Unsupervised discovery of temporal sequences in high-dimensional datasets, with applications to neuroscience

Emily L. Mackevicius[1†], Andrew H. Bahle[1†], Alex H. Williams[2], Shijie Gu[1,3], Natalia I. Denissenko[1], Mark S. Goldman[4*], Michale S. Fee[1*]

***For correspondence:**
fee@mit.edu (MSF);
msgoldman@ucdavis.edu (MSG)

[†]These authors contributed equally to this work

[1]McGovern Institute for Brain Research, Department of Brain and Cognitive Sciences, MIT; [2]Stanford Neurosciences Program; [3]School of Life Sciences and Technology, ShanghaiTech University; [4]Center for Neuroscience, Physiology and Behavior, UC Davis

Abstract

The ability to identify interpretable, low-dimensional features that capture the dynamics of large-scale neural recordings is a major challenge in neuroscience. Repeated temporal patterns (sequences) are not succinctly captured by traditional dimensionality reduction techniques, so neural data is often aligned to behavioral task references. We describe a task-independent, unsupervised method, which we call seqNMF, that provides a framework for extracting sequences from high-dimensional datasets, and assessing the significance in held-out data. We test seqNMF on simulated datasets under a variety of noise conditions, and also on several neural datasets. In a hippocampal dataset, seqNMF identifies neural sequences that match those calculated manually by reference to behavioral events. In a songbird dataset, seqNMF discovers abnormal motor sequences in birds that lack stereotyped songs. Thus, by identifying temporal structure directly from neural data, seqNMF enables dissection of complex neural circuits in the absence of reliable temporal references from stimuli or behavioral outputs.

Introduction

The ability to detect and analyze temporal sequences embedded in a complex sensory stream is an essential cognitive function, and as such is a necessary capability of neuronal circuits in the brain [13, 28, 5, 26], as well as artificial intelligence systems [14, 57]. The detection and characterization of temporal structure in signals is also useful for the analysis of many forms of physical and biological data. In neuroscience, recent advances in technology for electrophysiological and optical measurements of neural activity have enabled the simultaneous recording of hundreds or thousands of neurons [9, 33, 52, 29], in which neuronal dynamics are often structured in sparse sequences [23, 24, 44, 19, 45].

Such sequences can be identified by averaging across multiple trials, but only in cases where an animal receives a temporally precise sensory stimulus, or generates a sufficiently stereotyped motor output.

However, it could be useful to extract sequences on a moment-to-moment basis (without averaging), for example to study internal neuronal dynamics in the brain during learning, sleep, or diseased states. In these applications, it is not possible to use external timing references, and sequences must be extracted directly from the neuronal data. A traditional unsupervised approach for directly extracting structure in neuronal data is dimensionality reduction. Intuitively, sequences may be thought of as low dimensional, and yet dimensionality reduction techniques such as PCA and NMF do not work for sequences, because those methods only model synchronous patterns of activity.

Alternative approaches that search for repeating neural patterns require surprisingly challenging statistical analysis [7, 41, 49]. While progress has been made in analyzing non-synchronous sequential patterns using statistical models that capture cross-correlations between pairs of neurons [51, 21, 53, 59, 22], such methods may not have statistical power to scale to patterns that include many (more than a few dozen) neurons, may require long periods ($\geq 10^5$ timebins) of stationary data, and may have challenges in dealing with (non-sequential) background activity. For a review highlighting features and limitations of these methods see [49]. Here we took an alternative, matrix factorization-based, approach that aims to extract sequences. We reasoned that this approach would complement existing methods by providing a more holistic and potentially simpler description of neural firing dynamics.

One promising method for the unsupervised detection of temporal patterns is convolutional non-negative matrix factorization (convNMF) [56, 55] (Figure 1), which has been applied to the analysis of audio signals such as speech [43, 55, 61], as well as neural signals [47]. ConvNMF identifies exemplar patterns (factors) in conjunction with the times and amplitudes of pattern occurrences. This strategy eliminates the need to average activity aligned to any external behavioral references. While convNMF produces excellent reconstructions of the data, it does not automatically produce the minimal number of factors required. Indeed, if the number of factors in the convNMF model is greater than the true number of sequences, the algorithm returns overly complex and redundant factorizations. These redundant factorizations are different each time the algorithm is run, producing inconsistent results [47]. Notably, there is nothing in the convNMF algorithm that favors the minimal factorization, as would be favored by the principle of 'Occam's Razor'.

Here we describe a modification of the convNMF algorithm that suppresses redundant factors, biasing the results toward factorizations with a minimal number of factors. This is achieved by adding a penalty term to the convNMF cost function. Unlike other common approaches such as sparsity regularization [65, 43, 50, 47] that constrain the make-up of each factor, our regularization penalizes the correlations between factors that result from redundant factorizations. We build on earlier applications of soft-orthogonality constraints to NMF [10] to capture the types of temporally offset correlations that may occur in the convolutional case.

Our algorithm, which we call seqNMF, produces minimal and consistent factorizations in synthetic data, including under a variety of noise conditions, with high similarity to ground-truth sequences. We further tested seqNMF on hippocampal spiking data in

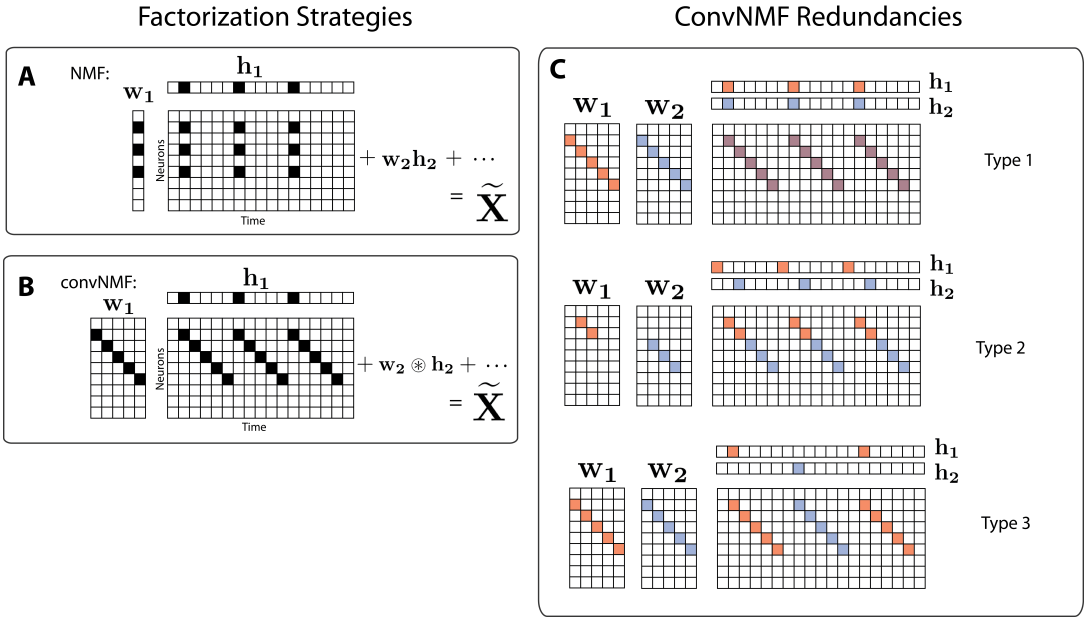

Figure 1. Introduction to convNMF factorization
(A) NMF (non-negative matrix factorization) approximates a dataset containing N neurons at T timepoints as a sum of K rank-one matrices. Each matrix is generated as the outer product of two nonnegative vectors: $\mathbf{w}_k$ of length N, which stores a neural ensemble, and $\mathbf{h}_k$ of length T, which holds the times at which the neural ensemble is active, and the relative amplitudes of this activity. **(B)** Convolutional NMF also approximates an $N \times T$ dataset as a sum of K matrices. Each matrix is generated as the convolution of two components: a non-negative matrix $\mathbf{w}_k$ of dimension $N \times L$ that stores a sequential pattern of the N neurons at L lags, and a vector of temporal loadings, $\mathbf{h}_k$, which holds the times at which each factor pattern is active in the data, and the relative amplitudes of this activity. **(C)** Three types of inefficiencies are present in unregularized convNMF: Type 1 in which two factors are used to reconstruct the same instance of a sequence, Type 2 in which two factors reconstruct a sequence in a piecewise manner, and Type 3 in which two factors are used to reconstruct different instances of the same sequence. For each case, the factors ($\mathbf{W}$ and $\mathbf{H}$) are shown, as well as the reconstruction ($\widetilde{\mathbf{X}} = \mathbf{W} \circledast \mathbf{H} = \mathbf{w}_1 \circledast \mathbf{h}_1 + \mathbf{w}_2 \circledast \mathbf{h}_2$).

which neural sequences have previously been described. Finally, we use seqNMF to extract sequences in a functional calcium imaging dataset recorded in vocal/motor cortex of untutored songbirds that sing pathologically variable songs. We found that repeatable neural sequences are activated in an atypical and overlapping fashion, suggesting potential neural mechanisms for this pathological song variability.

Results

Matrix factorization framework for unsupervised discovery of features in neural data

Matrix factorization underlies many well known unsupervised learning algorithms [60] with applications to neuroscience [15], including principal component analysis (PCA) [46], non-negative matrix factorization (NMF) [34], dictionary learning, and k-means clustering. We start with a data matrix, $\mathbf{X}$, containing the activity of N neurons at T times. If the neurons exhibit a single repeated pattern of synchronous activity, the entire data matrix can be reconstructed using a column vector $\mathbf{w}$ representing the neural pattern, and a row vector $\mathbf{h}$ representing the times and amplitudes at which that pattern occurs (temporal loadings). In this case, the data matrix $\mathbf{X}$ is mathematically reconstructed as the outer

product of $\mathbf{w}$ and $\mathbf{h}$). If multiple patterns are present in the data, then each pattern can be reconstructed by a separate outer product, where the reconstructions are summed to approximate the entire data matrix (Figure 1A) as follows:

$$\mathbf{X}_{nt} \approx \widetilde{\mathbf{X}}_{nt} = \sum_{k=1}^{K} \mathbf{W}_{nk}\mathbf{H}_{kt} = (\mathbf{WH})_{nt} \tag{1}$$

Where $\mathbf{X}_{nt}$ is the $(nt)^{th}$ element of matrix $\mathbf{X}$. Here, in order to store K different patterns, $\mathbf{W}$ is a $N \times K$ matrix containing the K exemplar patterns, and $\mathbf{H}$ is a $K \times T$ matrix containing the K timecourses:

$$\mathbf{W} = \begin{bmatrix} | & | & & | \\ \mathbf{w}_1 & \mathbf{w}_2 & \cdots & \mathbf{w}_K \\ | & | & & | \end{bmatrix} \qquad \mathbf{H} = \begin{bmatrix} - & \mathbf{h}_1 & - \\ - & \mathbf{h}_2 & - \\ & \vdots & \\ - & \mathbf{h}_K & - \end{bmatrix} \tag{2}$$

Given a data matrix with unknown patterns, the goal of these unsupervised learning algorithms is to discover a small set of patterns ($\mathbf{W}$) and a corresponding set of temporal loading vectors ($\mathbf{H}$) that approximate the data. In the case that the number of patterns (K) is sufficiently small (less than N and T), this corresponds to a dimensionality reduction, whereby the data is expressed in more compact form. NMF additionally requires that $\mathbf{W}$ and $\mathbf{H}$ must contain only non-negative numbers. The discovery of unknown factors is often accomplished by minimizing the following cost function, which measures (using the Frobenius norm, $\|\mathbf{M}\|_F = \sqrt{\sum_{ij} \mathbf{M}_{ij}}$) the element-by-element sum of all squared errors between a reconstruction $\widetilde{\mathbf{X}} = \mathbf{WH}$ and the original data matrix $\mathbf{X}$:

$$(\mathbf{W}^*, \mathbf{H}^*) = \underset{\mathbf{W},\mathbf{H}}{\arg\min} \|\widetilde{\mathbf{X}} - \mathbf{X}\|_F^2 \tag{3}$$

The factors $\mathbf{W}^*$ and $\mathbf{H}^*$ that minimize this cost function produce an optimal reconstruction $\widetilde{\mathbf{X}}^* = \mathbf{W}^*\mathbf{H}^*$ While this general strategy works well for extracting synchronous activity, it is unsuitable for discovering temporally extended patterns—first, because each element in a sequence must be represented by a different factor, and second, because NMF assumes that the columns of the data matrix are independent 'samples' of the data, so permutations in time have no effect on the factorization of a given dataset. It is therefore necessary to adopt a different strategy for temporally extended features.

Convolutional non-negative matrix factorization (convNMF)

Convolutional NMF (convNMF) [56, 55] extends NMF to provide a framework for extracting temporal patterns and sequences from data. While classical NMF represents each pattern as a single vector (Figure 1A), convNMF explicitly represents an exemplar pattern of neural activity over a brief period of time; the pattern is stored as an $N \times L$ matrix, where each column (indexed by $\ell = 1$ to L) indicates the activity of neurons at different timelags within the pattern (Figure 1B, where we call this matrix pattern $\mathbf{w_1}$ by analogy with NMF). The times at which this pattern/sequence occurs are stored using timeseries vector $\mathbf{h_1}$, as for NMF. The reconstruction is produced by convolving the $N \times L$ pattern with the timeseries $\mathbf{h_1}$ (Figure 1B).

If the dataset contains multiple patterns, each pattern is captured by a different $N \times L$ matrix and a different associated timeseries vector $\mathbf{h}$. A collection of K different patterns

Table 1. Notation for convolutional matrix factorization

Shift operator
The operator $\overrightarrow{(\mathbf{H})}^{\ell}$ shifts a matrix $\mathbf{H}$ in the $\rightarrow$ direction by ℓ timebins:
$\overrightarrow{(\mathbf{H})}^{\ell}_{\cdot t} = \mathbf{H}_{\cdot (t-l)}$ and likewise $\overleftarrow{(\mathbf{H})}^{\ell}_{\cdot t} = \mathbf{H}_{\cdot (t+\ell)}$
where $\cdot$ indicates all elements along the respective matrix dimension.
The shift operator inserts zeros when $(t-\ell) < 0$ or $(t+\ell) > T$
Tensor convolution operator
Convolutive matrix factorization reconstructs a data matrix $\mathbf{X}$
using a $N \times K \times L$ tensor $\mathbf{W}$ and a $K \times T$ matrix $\mathbf{H}$:
$\widetilde{\mathbf{X}} = \mathbf{W} \circledast \mathbf{H} = \sum_{\ell} \mathbf{W}_{\cdot\cdot\ell} \overrightarrow{\mathbf{H}}^{\ell}$
Note that each neuron n is reconstructed as the sum of k convolutions:
$\widetilde{\mathbf{X}}_{nt} = \sum_k \sum_{\ell} \mathbf{W}_{nk\ell} \mathbf{H}_{k(t-\ell)} \equiv (\mathbf{W} \circledast \mathbf{H})_{nt}$
Transpose tensor convolution operator
The following quantity is useful in several contexts:
$\mathbf{W} \overset{\mathsf{T}}{\circledast} \mathbf{X} = \sum_{\ell} (\mathbf{W}_{\cdot\cdot\ell})^{\mathsf{T}} \overleftarrow{\mathbf{X}}^{\ell}$
Note that each element $(\mathbf{W} \overset{\mathsf{T}}{\circledast} \mathbf{X})_{kt} = \sum_l (\mathbf{W}_{\cdot k\ell})^{\mathsf{T}} \mathbf{X}_{\cdot (t+\ell)} = \sum_l \sum_n \mathbf{W}_{nk\ell} \mathbf{X}_{n(t+\ell)}$ measures
the overlap (correlation) of factor k with the data at time t
CNMF reconstruction
$\mathbf{X} \approx \widetilde{\mathbf{X}} = \sum_k \mathbf{W}_{\cdot k \cdot} \circledast \mathbf{H}_{k \cdot} = \mathbf{W} \circledast \mathbf{H}$
Note that NMF is a special case of CNMF, where $L = 1$
$L1$ norm excluding diagonal elements
For any $K \times K$ matrix $\mathbf{C}$, $\|\mathbf{C}\|_{1,i \neq j} \equiv \sum_k \sum_{j \neq k} \mathbf{C}_{jk}$
Special matrices
$\mathbf{1}$ is a $K \times K$ matrix of ones
$\mathbf{I}$ is the $K \times K$ identity matrix
$\mathbf{S}$ is a $T \times T$ smoothing matrix: $\mathbf{S}_{ij} = 1$ when $\lvert i-j \rvert < L$ and otherwise $\mathbf{S}_{ij} = 0$

can be compiled together into an $N \times K \times L$ array (also known as a tensor) $\mathbf{W}$ and a corresponding $K \times T$ timeseries matrix $\mathbf{H}$. Analogous to NMF, convNMF generates a reconstruction of the data as a sum of K convolutions between each neural activity pattern ($\mathbf{W}$), and its corresponding temporal loadings ($\mathbf{H}$):

$$\mathbf{X}_{nt} \approx \widetilde{\mathbf{X}}_{nt} = \sum_{k=1}^{K} \sum_{\ell=0}^{L-1} \mathbf{W}_{nk\ell} \mathbf{H}_{k(t-\ell)} \equiv (\mathbf{W} \circledast \mathbf{H})_{nt} \tag{4}$$

where the tensor/matrix convolution operator $\circledast$ (notation summary, Table 1) reduces to matrix multiplication in the $L = 1$ case, which is equivalent to standard NMF. The quality of this reconstruction can be measured using the same cost function shown in Equation 3, and $\mathbf{W}$ and $\mathbf{H}$ may be found iteratively using similar multiplicative gradient descent updates to standard NMF [34, 56, 55].

While convNMF can perform extremely well at reconstructing sequential structure, it can be challenging to use when the number of sequences in the data is not known [47]. In this case, a reasonable strategy would be to choose K at least as large as the number of sequences that one might expect in the data. However, if the number of sequences is than the actual number of sequences, convNMF will identify more significant factors than are minimally required. This is because each sequence in the data may be approximated equally well by a single sequential pattern or by a linear combination of multiple partial patterns. A related problem is that running convNMF from different random initial conditions produces inconsistent results, finding different combinations of partial patterns on each run [47]. These inconsistency errors fall into three main categories (Figure 1C):

- *Type 1:* Two or more factors are used to reconstruct the same instances of a sequence.
- *Type 2:* Two or more factors are used to reconstruct temporally different parts of the same sequence, for instance the first half and the second half.
- *Type 3:* Identical factors are used to reconstruct different instances of a sequence.

Together, these inconsistency errors manifest as strong correlations between different redundant factors, as seen in the similarity of their temporal loadings ($\mathbf{H}$) and/or their exemplar activity patterns ($\mathbf{W}$).

SeqNMF: A constrained convolutional non-negative matrix factorization

Regularization is a common technique in optimization that allows the incorporation of constraints or additional information with the goal of improving generalization performance or simplifying solutions to resolve degeneracies [25]. To reduce the occurrence of redundant factors (and inconsistent factorizations) in convNMF, we sought a principled way of penalizing the correlations between factors by introducing a penalty term, $\mathscr{R}$, into the convNMF cost function of the following form:

$$(\mathbf{W}^*, \mathbf{H}^*) = \underset{\mathbf{W},\mathbf{H}}{\arg\min} \left(\|\widetilde{\mathbf{X}} - \mathbf{X}\|_F^2 + \mathscr{R} \right) \tag{5}$$

In this section, we will motivate a novel cost function that effectively minimizes the number of factors by penalizing spatial and temporal correlations between different factors. We will build up the full cost function by addressing, one at a time, the types of correlations generated by each type of error.

Regularization has previously been used in NMF to address the problem of duplicated factors, which, similar to Type 1 errors above, present as correlations between the $\mathbf{H}$'s [10]. Such correlations are measured by computing the correlation matrix $\mathbf{H}\mathbf{H}^T$, which contains the correlations between the temporal loadings of every pair of factors. The regularization may be implemented using the penalty term $\mathscr{R} = \lambda\|\mathbf{H}\mathbf{H}^T\|_{1,i\neq j}$, where the norm $\|\cdot\|_{1,i\neq j}$ sums the absolute value of every matrix entry except those along the diagonal (notation summary, Table 1) so that correlations between different factors are penalized, while the requisite correlation of each factor with itself is not. Thus, during the minimization process, similar factors compete, and a larger amplitude factor drives down the $\mathbf{H}$ of a correlated smaller factor. The parameter λ controls the magnitude of the penalty term $\mathscr{R}$.

In convNMF, a penalty term based on $\mathbf{H}\mathbf{H}^T$ yields an effective method to prevent errors of Type 1, because it penalizes the associated zero lag correlations. However, it does not prevent errors of the other types, which exhibit different types of correlations. For example Type 2 errors result in correlated temporal loadings that have a small temporal offset and thus are not detected by $\mathbf{H}\mathbf{H}^T$. One simple way to address this problem is to smooth the $\mathbf{H}$'s in the penalty term with a square window of length $2L-1$ using the smoothing matrix $\mathbf{S}$ ($\mathbf{S}_{ij} = 1$ when $|i-j| < L$ and otherwise $\mathbf{S}_{ij} = 0$). The resulting penalty, $\mathscr{R} = \lambda\|\mathbf{H}\mathbf{S}\mathbf{H}^T\|$, allows factors with small temporal offsets to compete, effectively preventing errors of Type 1 and 2.

Unfortunately this penalty does not prevent errors of Type 3, in which redundant factors with highly similar patterns in $\mathbf{W}$ are used to explain different instances of the same sequence. Such factors have temporal loadings that are segregated in time, and thus have low correlations, to which the cost term $\|\mathbf{H}\mathbf{S}\mathbf{H}^T\|$ is insensitive. One way to resolve errors of Type 3 might be to include an additional cost term that penalizes the similarity of the factor patterns in $\mathbf{W}$. A challenge with this approach is that, in the convNMF framework, there is no constraint on temporal translations of the sequence within $\mathbf{W}$. For example, if two redundant factors contain identical sequences that are simply offset by one time bin (in the L dimension), then these patterns would have zero correlation. Such offsets might be accounted for by smoothing the $\mathbf{W}$ matrices in time before computing the correlation (Table 3), analogous to $\|\mathbf{H}\mathbf{S}\mathbf{H}^T\|$. The general approach of adding an additional cost term for $\mathbf{W}$ correlations has the disadvantage that it requires setting an extra parameter, namely the λ associated with this cost.

Thus, we chose an alternative approach to resolve errors of Type 3 that simultaneously detects correlations in $\mathbf{W}$ and $\mathbf{H}$ using a single correlation cost term. We note that, for Type 3 errors, redundant $\mathbf{W}$ patterns have a high degree of overlap with the data at the same times, even though their temporal loadings are segregated at different times. To introduce competition between these factors, we first compute, for each pattern in $\mathbf{W}$ its overlap with the data at each time t. This quantity is captured in symbolic form by $\mathbf{W} \overset{\top}{\circledast} \mathbf{X}$ (See Table 1). We then compute the pairwise correlation between the temporal loading of each factor and the overlap of every other factor with the data. The correlation cost sums up these correlations across all pairs of factors, implemented as follows:

$$\mathscr{R} = \lambda\|(\mathbf{W} \overset{\top}{\circledast} \mathbf{X})\mathbf{S}\mathbf{H}^T\|_{1,i\neq j} \tag{6}$$

When incorporated into the update rules, this causes any factor that has a high overlap with the data to suppress the temporal loadings ($\mathbf{H}$) of any other factors active at that time. Thus, factors compete to explain each feature of the data, favoring solutions that

use a minimal set of factors to give a good reconstruction. We refer to this minimal set as an efficient factorization. The resulting global cost function is:

$$(\mathbf{W}^*, \mathbf{H}^*) = \arg\min_{\mathbf{W},\mathbf{H}} \left(\|\widetilde{\mathbf{X}} - \mathbf{X}\|_F^2 + \lambda \|(\mathbf{W} \overset{\mathsf{T}}{\circledast} \mathbf{X})\mathbf{S}\mathbf{H}^\mathsf{T}\|_{1,i\neq j} \right) \tag{7}$$

The update rules for $\mathbf{W}$ and $\mathbf{H}$ are based on the derivatives of this global cost function, leading to a simple modification of the standard multiplicative update rules used for NMF and convNMF [34, 56, 55] (Table 3). Note that the addition of this correlation cost term does not formally constitute regularization, because it also includes a contribution from the data matrix $\mathbf{X}$. rather than just the model variables $\mathbf{W}$ and $\mathbf{H}$.

Below, we test the performance of this penalty based on correlations between factors. We will later consider different approaches to adding penalties to the convNMF cost function, including an L1 norm penalty. We will also examine a parameter sweep of the number of factors (K), as well as additional penalties to bias the tradeoff between temporal or pattern correlations.

Testing the performance of seqNMF on simulated sequences

To compare the performance of seqNMF to unregularized convNMF, we simulated neural sequences of a sort commonly encountered in neuronal data (Figure 2A). The simulated data were used to test several aspects of the seqNMF algorithm: convergence, consistency of factorizations, the ability of the algorithm to discover the correct number of sequences in the data, and robustness to noise. As an initial pass, simulated datasets were constructed by placing three ground-truth sequences at random non-overlapping times. Each sequence ensemble consisted of 10 neurons evenly spaced throughout a duration of 30 timesteps. The resulting data matrix had a total duration of 15000 timesteps and contained on average 60±6 instances of each of the three sequences. The seqNMF algorithm was run for 1000 iterations and reliably converged to a stable asymptotic value of root-mean-squared-error (RMSE) (Figure 2B). RMSE reached to within 10% of the asymptotic value within 100 iterations.

Consistency of seqNMF factorization

We set out to determine if seqNMF exhibits the desirable property of consistency—namely whether it returns similar sequences each time it is run on the same dataset using different random initializations of $\mathbf{W}$ and $\mathbf{H}$. Consistency was assessed as the extent to which there is a good one-to-one match between factors across different runs (Methods 10). Due to the inefficiencies outlined in Figure 1C, with K larger than the true number of sequences, convNMF yielded low consistency scores typically ranging from 0.2 to 0.4 on a scale from zero to one (Figure 2C, orange). In contrast, seqNMF factorizations were nearly identical across different fits of noiseless data, producing consistency scores that were always higher than any we measured for convNMF, and typically (>80% of the time) higher than 0.99 (Figure 2C, gray). Both convNMF and seqNMF had near-perfect reconstruction error for all combinations of K and L that exceed the number and duration of sequences in the data (not shown). However, convNMF exhibited low consistency scores, a problem that was further exacerbated for larger values of K. In contrast, seqNMF exhibited high consistency scores across a wide range of values of both K and L.

We also tested the consistency of seqNMF factorizations for the case in which a population of neurons is active in multiple different sequences. Such neurons that are

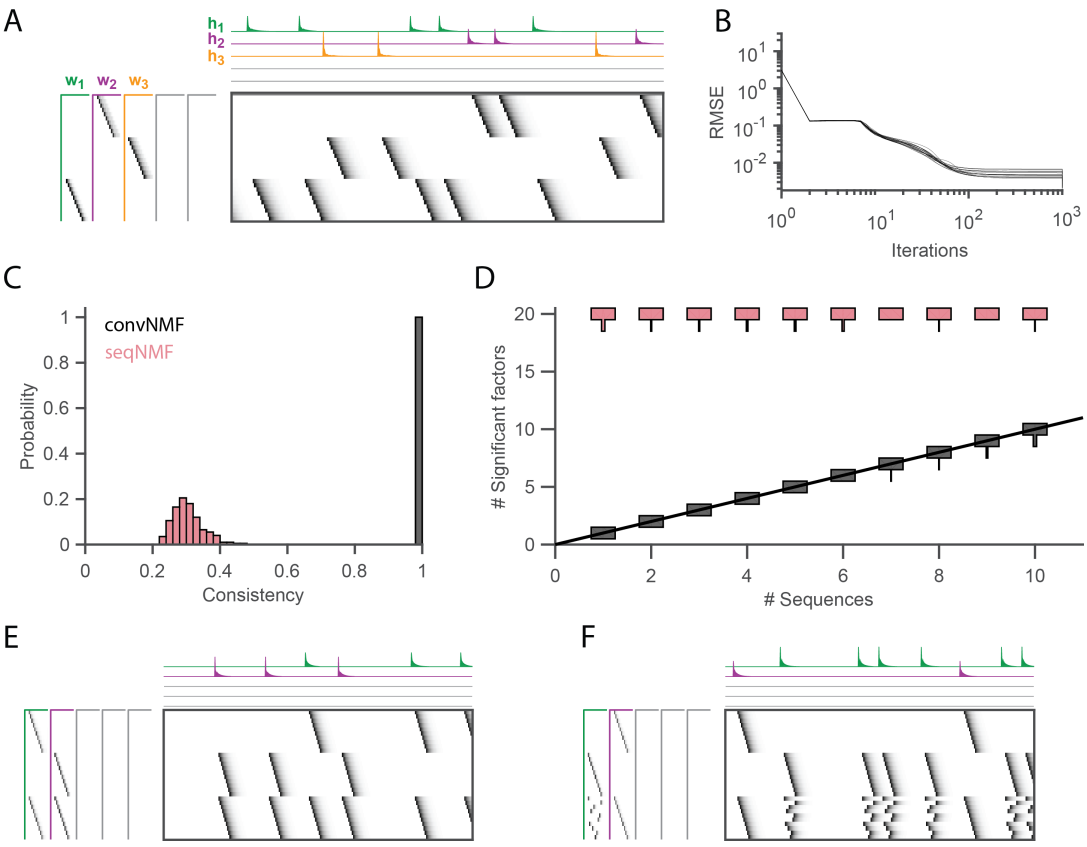

Figure 2. Testing seqNMF on simulated data

(A) A simulated dataset with three sequences and a seqNMF factorization ($K = 20$, $L = 50$, $\lambda = 0.003$). Each significant seqNMF factor is shown in a different color. At left are the exemplar patterns ($\mathbf{W}$) and on top are the timecourses ($\mathbf{H}$). **(B)** RMSE as a function of seqNMF iteration number. SeqNMF was run on a simulated dataset with three sequences and 15000 timebins (≈ 60 instances of each sequence). Twenty independent runs of seqNMF are shown. On this dataset, seqNMF converges to within 10% of the asymptotic error value within ≈ 100 iterations. **(C)** SeqNMF is more consistent than unregularized convNMF across 400 independent fits ($K = 20$, $L = 50$, $\lambda = 0.003$). **(D)** Plot showing the number of statistically significant factors vs. true number of simulated sequences for seqNMF and convNMF, for data containing between 1 and 10 sequences. Shown for each case is a vertical histogram representing the number of significant factors over 20 runs ($K = 20$, $L = 50$, $\lambda = 0.003$). **(E)** A seqNMF factorization of two simulated neural sequences with shared neurons that participate at the same latency in both sequences. **(F)** A seqNMF factorization of two simulated neural sequences with shared neurons that participate at different latencies in each sequence.

shared across different sequences have been observed in several neuronal datasets [44, 45, 24]. For one test, we constructed two sequences in which shared neurons were active at a common pattern of latencies in both sequences; in another test, shared neurons were active in a different pattern of latencies in each sequence. In both tests, seqNMF achieved near-perfect reconstruction error, and consistency was similar to the case with no shared neurons (Figure 2E, F).

Validating the statistical significance of extracted sequences

To assess statistical significance, one can apply seqNMF to a subset of the data and measure whether the extracted sequences appear in held-out data substantially more than sequences drawn from a null model. We measured the appearance of sequences in held-out data by measuring their overlap with held-out data, $\mathbf{W} \circledast^{\top} \mathbf{X}$. The overlap is high at timepoints at which the sequence occurs (relative to other timepoints). For a sequence that matches ground truth in synthetic data, this distribution of overlap values exhibits a heavy tail, indicating the presence of large outliers that correspond to times where the extracted sequence appears in held out data. In contrast, a candidate sequence that does not reliably occur in the held-out data produces a distribution of overlaps that appears more symmetric (Figure S1).

While there are many ways of detecting outliers and quantifying "heavy-tailedness" of a distribution, we use the skewness (the third central moment) as a simple measure. In particular, we generate null distributions by circularly shifting the pattern matrices $\mathbf{W}$ along the time lag dimension (see Methods 10) and compare the skewness of these distributions to the skewness of the distribution produced by the unshifted $\mathbf{W}$.

Runs of seqNMF on simulated and real data have revealed that the algorithm produces two types of factors that can be immediately ruled out as candidate sequences: 1) empty factors with zero amplitude in all neurons at all lags and 2) factors that have amplitude in only one neuron. The latter case occurs often in datasets where one neuron is substantially more active than other neurons, and thus accounts for a large amount of variance in the data. SeqNMF also occasionally generates factors that appear to capture one moment in the test data, especially in short datasets, where this can account for a substantial fraction of the data variance. Such sequences are easily identified as non-significant when tested on held-out data using the skewness test.

Note that if λ is set too small, seqNMF will produce multiple redundant factors to explain one sequence in the data. In this case, each redundant candidate sequence will pass the significance test outlined here. We will address below a procedure for choosing λ and methods for determining the number of sequences.

SeqNMF extracts the correct number of sequences in noise-free synthetic data

A successful factorization should contain the same number of significant factors as exist sequences in the data, at least in datasets for which the number of sequences is unambiguous. To compare the ability of seqNMF and convNMF to recover the true number of patterns in a dataset, we generated simulated noise-free data containing between 1 and 10 different sequences. We then ran many independent fits of these data, using both seqNMF and convNMF, and measured the number of significant factors. We found that convNMF overestimates the number of sequences in the data, returning K significant factors on nearly every run. In contrast, seqNMF tends to return a number of

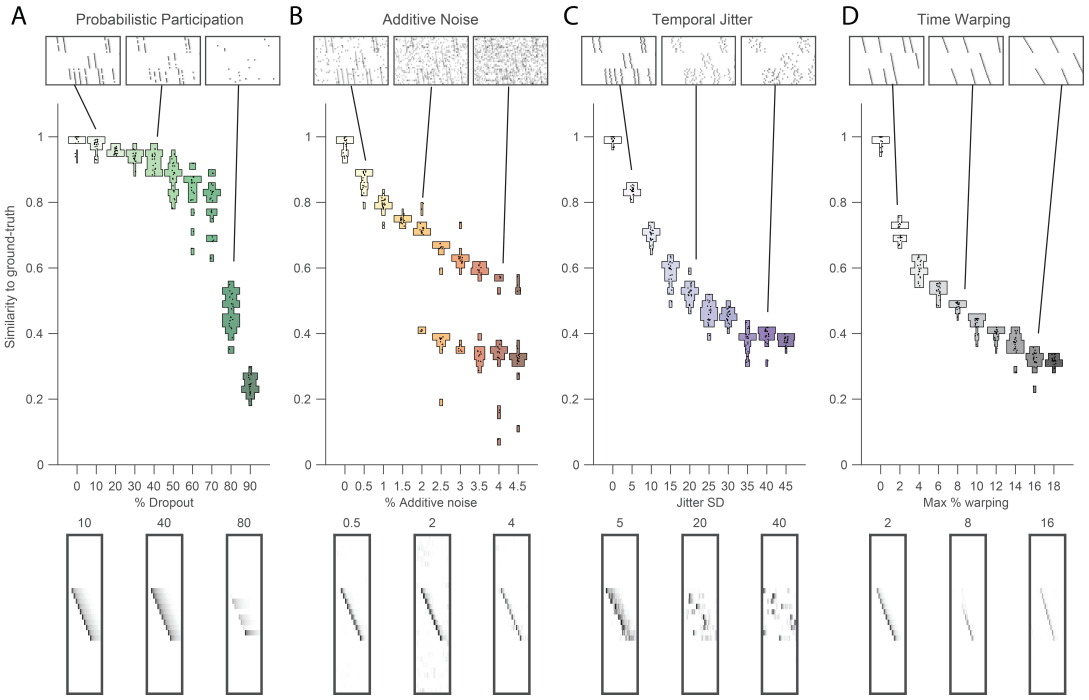

Figure 3. Testing seqNMF performance on sequences contaminated with noise
Performance of seqNMF was tested under 4 different noise conditions: **(A)** probabilistic participation, **(B)** additive noise, **(C)** timing jitter, and **(D)** sequence warping. For each noise type, we show: (top) examples of synthetic data at 3 different noise levels; (middle) similarity of seqNMF factors to ground-truth factors across a range of noise levels, showing 20 fits for each noise level; and (bottom) example of one of the $\mathbf{W}$'s extracted at 3 different noise levels (same noise levels as data shown in top row). SeqNMF was run with $K = 20$, $L = 50$. In these examples, the algorithm was run with λ chosen using the automated procedure outlined in Figure 4. For results at different values of λ, see Figure S2.

significant factors that closely matches the actual number of sequences (Figure 2D).

Robustness to noisy data

SeqNMF was able to correctly extract sequences even in data corrupted by noise of types commonly found in neural data. We consider four common types of noise: participation noise, in which individual neurons participate probabilistically in instances of a sequence; additive noise, in which neuronal events occur randomly outside of normal sequence patterns; temporal jitter, in which the timing of individual neurons is shifted relative to their typical time in a sequence; and finally, temporal warping, in which each instance of the sequence occurs at a different randomly selected speed. To test the robustness of seqNMF to each of these noise conditions, we factorized data containing three neural sequences at a variety of noise levels. The value of λ was chosen using methods described in the next section. SeqNMF proved relatively robust to all four noise types, as measured by quantifying the similarity between seqNMF factors and ground-truth sequences (Methods section 10, Figure 3). For low noise conditions, seqNMF produced factors that were highly similar to ground-truth; this similarity gracefully declined as noise increased. Visualization of the extracted factors revealed that they tend to qualitatively match ground-truth sequences even in the presence of high noise (Figure 3). Together, these findings suggest that seqNMF is suitable for extracting sequence patterns from neural data with realistic forms of noise.

We also tested the performance of seqNMF as a function of dataset size. To do so, we generated data of different sizes containing different numbers of instances of the underlying ground-truth sequences, ranging from 1 to 20. For intermediate levels of additive noise, we found that 3 examples of each sequence were sufficient for seqNMF to correctly extract factors with similarity scores within 10% of asymptotic performance (Figure S3).

Method for choosing an appropriate value of λ

Here we present procedures for guiding the choice of λ in seqNMF that address two goals of regularization: to simplify the solution space of ill-posed problems and to reduce overfitting. The choice of λ controls a trade-off between reconstruction accuracy and the efficiency/consistency of the resulting factorizations (Figure 4). The goal is to reconstruct only the repeating temporal patterns in the data and to do so with an efficient, maximally uncorrelated set of factors. We will first describe a procedure that balances a measure of correlation between factors with reconstruction error. We then describe a procedure based on cross-validation in held-out data. Both of these procedures are validated under a variety of noise conditions using simulated data for which the ground truth factors are known.

In the first procedure we measure the effect of λ on both reconstruction error and correlation cost in synthetic datasets containing three sequences (Figure 4). For any given factorization, the reconstruction error is $\|\mathbf{W} \circledast \mathbf{H} - \mathbf{X}\|_F^2$, and the efficiency may be estimated using the correlation cost term ($\|(\mathbf{W} \overset{\top}{\circledast} \mathbf{X})\mathbf{S}\mathbf{H}^\top\|_{1,i\neq j}$). SeqNMF was run with many random initializations over a range of λ spanning six orders of magnitude. For small λ, the behavior of seqNMF approaches that of convNMF, producing a large number of redundant factors with high correlation cost. In the regime of small λ, correlation cost saturates at a large value and reconstruction error saturates at a minimum value (Figure 4A). At the opposite extreme, in the limit of large λ, seqNMF returns a single significant factor with zero correlation cost because all other factors have been suppressed to zero amplitude. In this limit, the single factor is unable to reconstruct multi-sequence data, resulting in large reconstruction error. Between these extremes, there exists a region in which increasing λ produces a rapidly increasing reconstruction error and a rapidly decreasing correlation cost. Following the intuition that the optimal choice of λ for seqNMF would lie in this cross-over region where the costs are balanced, we set out to quantitatively identify, for known synthetic sequences, the optimal λ at which seqNMF has the highest probability of recovering the correct number of significant factors, and at which these factors most closely match the ground truth sequences.

The following procedure was implemented: For a given dataset, seqNMF is run several times at a range of values of λ, and saturating values of reconstruction cost and correlation cost are recorded (at the largest and smallest values of λ). Costs are normalized to vary between 0 and 1, and the value of λ at which the reconstruction and correlation cost curves intersect is determined (Figure 4B). This intersection point, λ_0, then serves as a precise reference by which to determine the correct choice of λ. We then separately calibrated the reference λ_0 to the λ's that performed well in synthetic datasets, with and without noise, for which the ground-truth is known. This analysis revealed that values of λ between $2\lambda_0$ and $5\lambda_0$ performed well across different noise types and levels (Figure 4B,C). For additive noise, performance was better when λ was chosen to be near λ_0, while with

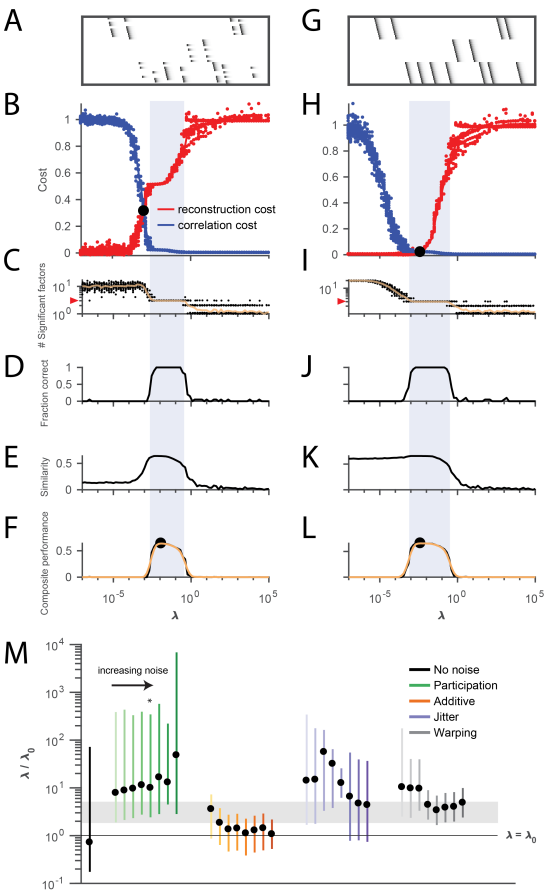

Figure 4. Procedure for choosing λ for a new dataset based on finding a balance between reconstruction cost and correlation cost

(A) Simulated data containing three sequences in the presence of participation noise (50% participation probability). This noise condition is used for the tests in (B-F). **(B)** Normalized reconstruction cost ($||\widetilde{\mathbf{X}} - \mathbf{X}||_F^2$) and correlation cost ($||(\mathbf{W} \overset{\top}{\circledast} \mathbf{X})\mathbf{S}\mathbf{H}^\top||_{1,i\neq j}$) as a function of λ for these data. The cross-over point λ_0 is marked with a black circle. **(C)** The number of significant factors obtained from 20 fits of these data as a function of λ (mean number plotted in orange). The correct number of factors (three) is marked by a red triangle. **(D)** The fraction of fits returning the correct number of significant factors as a function of λ. **(E)** Similarity of the top three factors to ground-truth (noiseless) factors as a function of λ. **(F)** Composite performance measured as the product of the curves shown in (D) and (E), (smoothed curve plotted in orange with a circle marking the peak). Shaded region indicates the range of λ that works well ($\pm$ half height of composite performance). **(G)** Simulated data containing three noiseless sequences. **(H-L)** Same as (B-F) but for the noiseless data. **(M)** Summary plot showing the range of values of λ (vertical bars), relative to the cross-over point λ_0, that work well for each noise condition ($\pm$ half height points of composite performance; note that this curve is a product of two other curves, and thus narrower, giving a conservative estimate of the range of effective λs). Circles indicate the value of λ at the peak composite performance. For each noise type, results for the all non-zero noise levels from Figure 3 are shown (increasing color saturation at high noise levels; Green, participation: 90, 80, 70, 60, 50, 40, 30, and 20%; Orange, additive noise 0.5, 1, 2, 2.5, 3, 3.5, and 4%; Purple, jitter: SD of the distribution of random jitter: 5, 10, 15, 20, 25, 30, 35, 40, and 45 timesteps; Grey, timewarp: 13, 16, 20, 26, 33, 40, 46, and 53%). The noise type and level in panels (A-F) is indicated by *. Gray band indicates a range between $2\lambda_0$ and $5\lambda_0$. Values of λ in this range tended to perform well across the different noise conditions. In real data, it may be useful to explore a wider range of λ.

other noise types, performance was better at higher λs ($\approx 5\lambda_0$). For all of the data shown in Figure 3, we chose $\lambda = 2\lambda_0$. Figure S2 shows how choosing $\lambda = \lambda_0$ for additive noise and $\lambda = 5\lambda_0$ for the other noise types yields slightly improved performance. Note that the procedure for choosing λ does not need to be run on every dataset analyzed, rather, only when seqNMF is applied to a new type of data for which a reasonable range of λ is not already known. Similar ranges of λ appeared to work for datasets with different numbers of ground-truth sequences—for the datasets used in Figure 2D, a range of λ between 0.001 and 0.01 returned the correct number of sequences at least 90% of the time for datasets containing between 1 and 10 sequences (Figure S4). Furthermore, this method for choosing λ also works on datasets containing sequences with shared neurons (Figure S5).

Our second method for choosing λ directly tests generalization error by randomly holding out a small subset of elements in the data matrix [64, 6] (Figure S6). This held-out set is only used to test the performance of seqNMF, but is not used for fitting. At high values of λ, seqNMF extracts only one factor, which exhibits similar reconstruction error on training data and held-out test data. At low values of λ, seqNMF extracts a large number of factors, yielding better reconstruction error on the training data, but the performance of these factors on held-out data is often far worse, corresponding to overfitting. At intermediate values of λ, within the optimal range described above, there was often a minimum in the reconstruction error on held-out data (test error). This corresponds to the classical approach for choosing regularization strength using cross-validation. In some datasets, the minimum in test error can be subtle or nonexistent, so we instead identify the λ corresponding to the rapid divergence between training error and test error (Figure S6C). In many of our test datasets, this divergence point agrees with the ground-truth and with the procedure described above based on the crossover between correlation cost and reconstruction cost. One caution in using the cross-validation method to choose an optimal λ is that it fails on synthetic datasets that have zero or very low noise (because of a lack of overfitting), as well as in datasets with temporal warping. More broadly, difficulties using cross-validation to choose λ may reflect that the primary function of the seqNMF penalty term is to reduce factor correlations and redundancies, not to minimize over-fitting.

Can we choose K rather than choosing λ?

A goal of the seqNMF correlation cost term is to limit the factorization to a small number of non-redundant factors. An alternative approach may be to directly constrain the number of factors (K) in the convNMF algorithm without regularization. If the number of underlying sequences in the data is unambiguous and is precisely known, as for the simulated datasets described above, then this approach works well, yielding factorizations close to ground truth sequences. We have found that the number of underlying sequences can sometimes be estimated by running convNMF for all reasonable values of K and selecting the value that yields the best cross-validated performance on held-out data. This method works reasonably well for simulated datasets with participation noise, additive noise, or temporal jitter over a range of noise levels that might be expected in real neural data. In some cases, there is a clear minimum in the test error at the correct K. In other cases there is a distinguishing feature such as a kink or a plateau in the test error as a function of K that could potentially be used to estimate the correct number of sequences

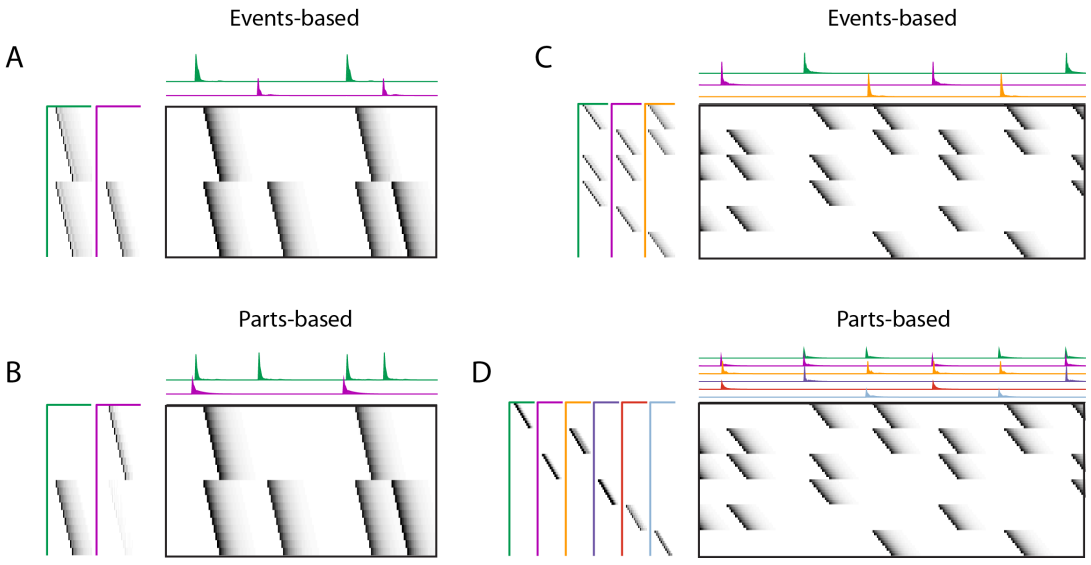

Figure 5. Using penalties to bias towards events-based and parts-based factorizations
Datasets that have neurons shared between multiple sequences can be factorized in different ways, emphasizing discrete temporal events (events-based) or component neuronal ensembles (parts-based), by penalizing correlations in $\mathbf{H}$ or $\mathbf{W}$ respectively. (Left) A dataset with two different ensembles of neurons that participate in two different types of events, with **(A)** events-based and **(B)** parts-based factorizations. (Right) A dataset with six different ensembles of neurons that participate in three different types of events, with **(C)** events-based and **(D)** parts-based factorizations.

(Figure S7). Notably, this method fails to identify the number of underlying sequences in the case of temporal warping—an issue to which we will return in the next section.

Strategies for dealing with ambiguous cases

In some datasets, there is not a unique answer for the desired factorization of sequences. A common example of such ambiguity arises when neurons are shared between different sequences, as is shown in Figure 5A and B. In this case, there are two ensembles of neurons (1 and 2), that participate in two different types of events. In one event type, ensemble 1 is active alone, while in the other event type, ensemble 1 is coactive with ensemble 2. There are two different reasonable factorizations of these data. In one factorization the two different ensembles are separated into two different factors, while in the other factorization the two different event types are separated into two different factors. We refer to these as 'parts-based' and 'events-based' respectively. Note that these different factorizations may correspond to different intuitions about underlying mechanisms. 'Parts-based' factorizations will be particularly useful for clustering neurons into ensembles, and 'events-based' factorizations will be particularly useful for correlating neural events with behavior.

We have found that seqNMF and convNMF can produce either type of factorization, depending on initial conditions and the structure of shared neurons in the data. It may therefore be useful to explicitly control the tendency to produce these different factorizations by the addition of penalties on either $\mathbf{W}$ or $\mathbf{H}$ correlations. Note that in the 'events-based' factorization, the $\mathbf{H}$s are orthogonal (uncorrelated) while the $\mathbf{W}$s have high overlap; in the 'parts-based' factorization, the $\mathbf{W}$s are orthogonal while the $\mathbf{H}$s are strongly correlated. Note that these correlations in $\mathbf{W}$ or $\mathbf{H}$ are unavoidable in the presence of

shared neurons and the presence of such correlations does not indicate a redundant factorization. Update rules to implement penalties on correlations in $\mathbf{W}$ or $\mathbf{H}$ are provided in Table 3 with derivations in Appendix 1. Figure S9 shows examples of using these penalties on the songbird dataset described in Figure 7.

Another type of ambiguity arises from the presence of systematic variations in the amplitude or timing of neuronal participation in a sequence. A notable example of this is data with temporal warping. In the case of high λ, seqNMF extracts a single factor for the underlying ground truth sequence. In contrast, at lower λ seqNMF extracts multiple factors for the underlying ground truth sequence, corresponding to slower and faster variations of the sequence, effectively tiling the space of warped sequences at a finer granularity depending on the strength of the penalty (λ). Note that each of these factorizations corresponds to a reasonable interpretation, in the context of seqNMF, for the same underlying timewarping process. Different neural datasets may require estimating warping with different degrees of precision, depending on the behavior being studied, leading to different reasonable choices of λ.

Another case requiring a choice between different reasonable levels of λ occurs when a sequence exhibits two variants in which, for example, two subensembles of neurons participate with different amplitudes in different instances of the sequence. Depending on the desired level of granularity, controlled by the choice of λ, this dataset could be factorized either as a single sequence or as two sequences. Any example in which a sequence has multiple close variants, either in the timing or activity of different neurons, can lead to this type of ambiguity. Depending on what type of factorization is desired, a different value of λ might be preferable. In real datasets, it can be useful to explore the factorization for different values of λ between λ_0 and $10\lambda_0$. There may often be a range of λ that give rise to different reasonable factorizations. Note that high λ risks missing sequences, especially sequences that occur rarely or include only a small number of neurons, and low λ may give rise to redundant factors.

Addition of a sparsity penalty to seqNMF or convNMF

Sparsity regularization is a widely used strategy for achieving more interpretable and generalizable results across a variety of algorithms and datasets [65], including convNMF [43, 50]. In some of our datasets, we found it useful to include $L1$ regularization for sparsity. The multiplicative update rules in the presense of $L1$ regularization are included in Table 3, and as part of our code package. Sparsity on the matrices $\mathbf{W}$ and $\mathbf{H}$ may be particularly useful in cases when sequences are repeated rhythmically (Figure S8). For example, the addition of a sparsity regularizer on the $\mathbf{W}$ update will bias the $\mathbf{W}$ exemplars to include only a single repetition of the repeated sequence, while the addition of a sparsity regularizer on $\mathbf{H}$ will bias the $\mathbf{W}$ exemplars to include multiple repetitions of the repeated sequence. This gives one fine control over how much structure in the signal to pack into $\mathbf{W}$ versus $\mathbf{H}$. Like the ambiguities described above, these are both valid interpretations of the data, but each may be more useful in different contexts.

Application of seqNMF to hippocampal sequences

To test the ability of seqNMF to discover patterns in electrophysiological data, we analyzed spiking activity in datasets of simultaneously recorded hippocampal neurons acquired in the Buszaki lab and available from a public repository (https://crcns.org/data-sets/hc) [2, 1].

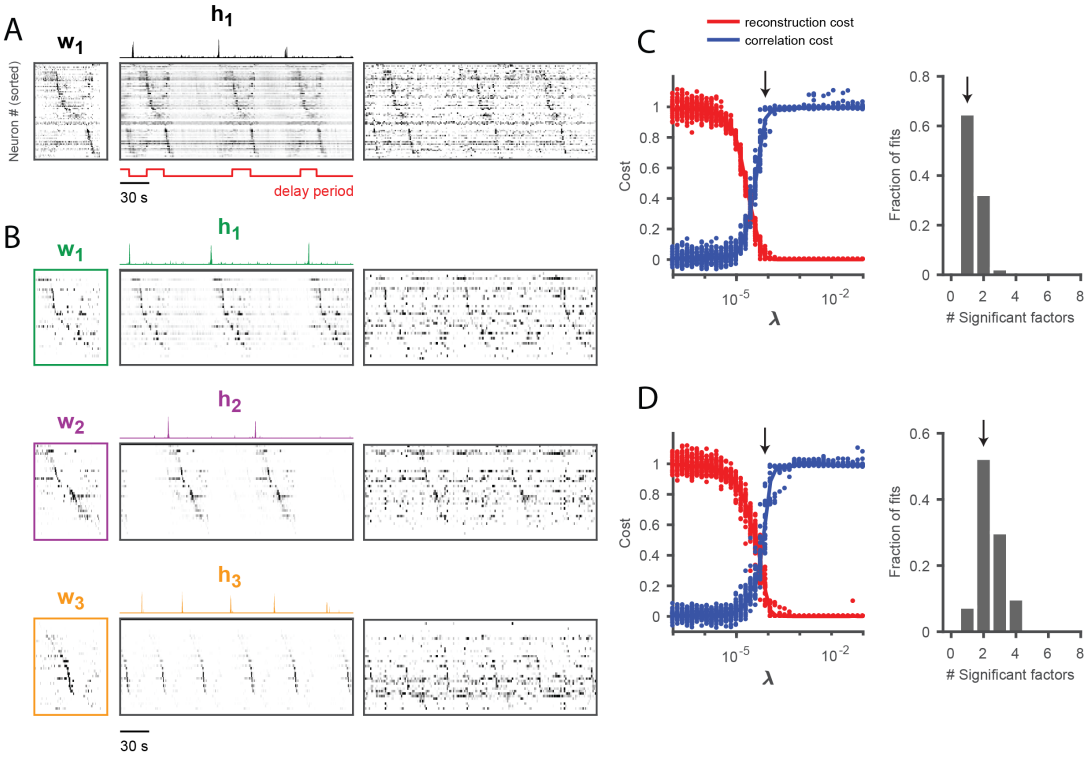

Figure 6. Application of seqNMF to extract hippocampal sequences from two rats
(A) Firing rates of 110 neurons recorded in the hippocampus of Rat 1 during an alternating left-right task with a delay period [2], as well as the single significant extracted seqNMF factor. Neurons are sorted according to the latency of their peak activation within the factor. The red line shows the onset and offset of the forced delay periods, during which the animal ran on a treadmill **(B)** Firing rates of 43 hippocampal neurons recorded in Rat 2 during the same task [1]. Neurons are sorted according to the latency of their peak activation within each of the three significant extracted sequences. Both seqNMF reconstruction of each factor (left) and raw data (right) are shown. The first two factors correspond to left and right trials, and the third corresponds to running along the stem of the maze. **(C)** (Left) Reconstruction (red) and correlation (blue) costs as a function of λ for Rat 1. Arrow indicates $\lambda = 8x10^{-5}$, used for seqNMF factorization shown in (A). (Right) Histogram of the number of significant factors across 30 runs of seqNMF. **(D)** Same as in (C) but for Rat 2. Arrow indicates $\lambda = 8x10^{-5}$ used for factorization shown in (B).

The data were acquired in two rats as part of published studies describing sequences in the hippocampus [45, 16]. In these experiments, rats were trained to alternate between left and right turns in a T-maze to earn a water reward. Between alternations, the rats ran on a running wheel during an imposed delay period lasting either 10 or 20 seconds. By averaging spiking activity during the delay period, the authors reported long temporal sequences of neural activity spanning the delay. In some rats, the same sequence occurred on left and right trials, while in other rats, different sequences were active in the delay period during the different trial types.

Without reference to the behavioral landmarks, seqNMF was able to extract sequences in both datasets. The automated method described above was used to choose λ (Figure 6). In Rat 1, with $\lambda = 2\lambda_0$, most runs of seqNMF extracted a single significant factor, corresponding to a sequence active throughout the running wheel delay period and immediately after, when the rat runs up the stem of the maze (Figure 6B). Some runs of seqNMF extracted two factors, splitting the delay period sequence and the maze stem sequence; this is a reasonable interpretation of the data, and likely results from variability in the relative timing of running wheel and maze stem traversal. At somewhat lower values of λ, seqNMF more often split these sequences into two factors. At even lower values of λ, seqNMF produced more significant factors. Such higher granularity factorizations may correspond to real variants of the sequences, as they generalize to held-out data (Figure S7J).

In Rat 2, at a λ of 1.5 λ_0, three significant factors were typically identified (Figure 6C). The first two correspond to distinct sequences active for the duration of the delay period on alternating trials. The third sequence was active immediately following each of the alternating sequences, corresponding to the time at which the animal exits the wheel and runs up the stem of the maze. Taken together, these results suggest that seqNMF can detect multiple neural sequences without the use of any behavioral landmarks. Having validated this functionality in both simulated data and previously published neural sequences, we then applied seqNMF to find structure in a novel dataset, in which the ground truth is unknown and difficult to ascertain using previous methods.

Application of seqNMF to abnormal sequence development in avian motor cortex

We applied seqNMF to analyze new functional imaging data recorded in songbird HVC during singing. Normal adult birds sing a highly stereotyped song, making it possible to detect sequences by averaging neural activity aligned to the song. Using this approach, it has been shown that HVC neurons generate precisely timed sequences that tile each song syllable [23, 48, 37]. In contrast to adult birds, young birds sing highly variable babbling vocalizations, known as subsong, for which HVC is not necessary [3]. The emergence of sequences in HVC occurs gradually over development, as the song matures from subsong to adult song [44].

Songbirds learn their song by imitation and must hear a tutor to develop normal adult vocalizations. Birds isolated from a tutor sing highly variable and abnormal songs as adults [18]. Such 'isolate' birds provide an opportunity to study how the absence of normal auditory experience leads to pathological vocal/motor development. However, the high variability of pathological 'isolate' song makes it difficult to identify neural sequences using the standard approach of aligning neural activity to vocal output.

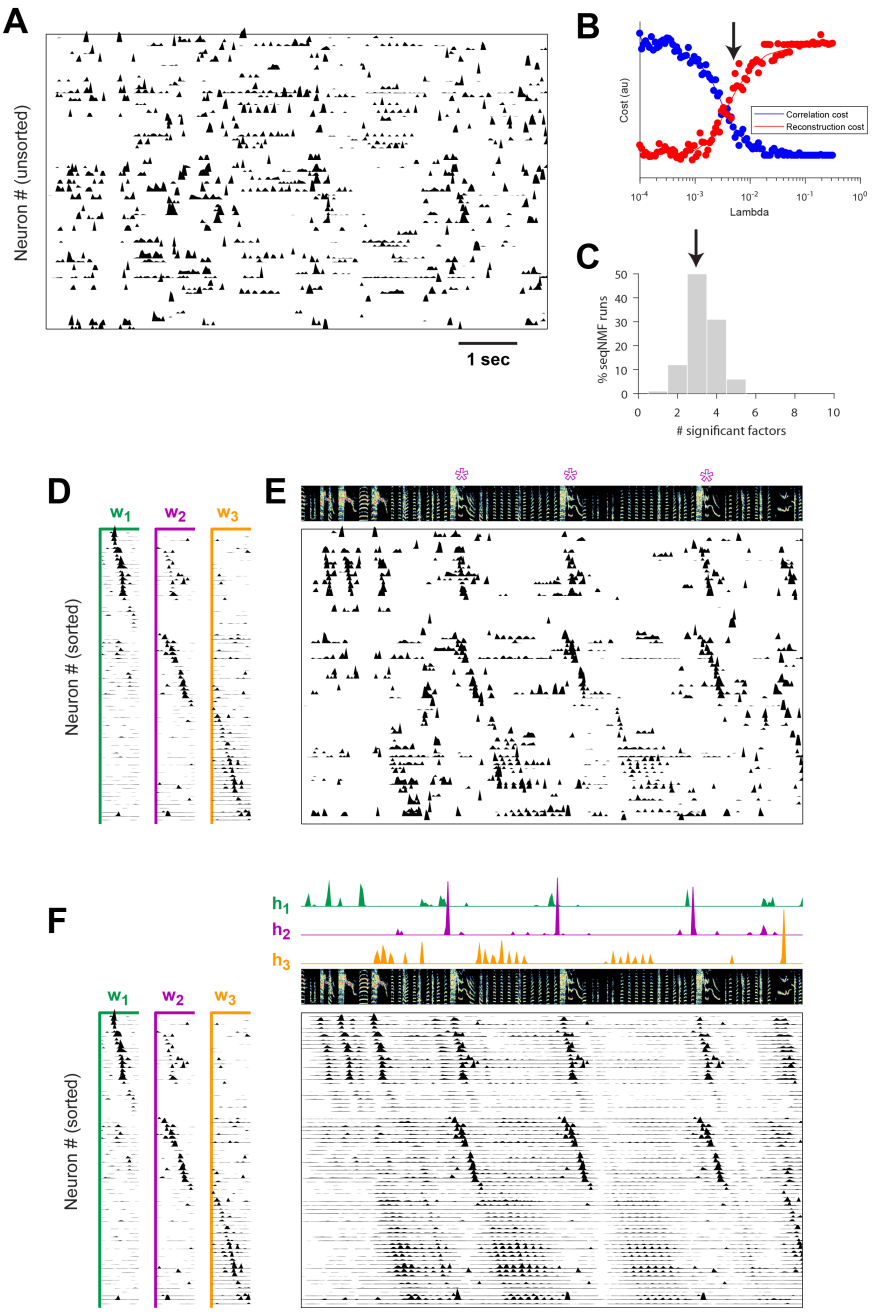

Figure 7. SeqNMF applied to calcium imaging data from a singing isolate bird reveals abnormal sequence deployment

(A) Functional calcium signals recorded from 75 neurons, unsorted, in a singing isolate bird. **(B)** Reconstruction and correlation cost as a function of λ. The arrow at $\lambda = 0.005$ indicates the value selected for the rest of the analysis. **(C)** Number of significant factors for 100 runs of seqNMF with $K = 10$, $\lambda = 0.005$. Arrow indicates 3 is the most common number of significant factors. **(D)** SeqNMF factor exemplars ($\mathbf{W}$'s), Neurons are grouped according to the factor in which they have peak activation, and within each group neurons are sorted by the latency of their peak activation within the factor **(E)** The same data shown in (A), after sorting neurons by their latency within each factor as in (D). A spectrogram of the bird's song is shown at top, with a purple '*' denoting syllable variants correlated with $\mathbf{w}_2$. **(F)** Same as (E), but showing reconstructed data rather than calcium signals. Shown at top are the temporal loadings ($\mathbf{H}$) of each factor.

Using seqNMF, we were able to identify repeating neural sequences in isolate songbirds (Figure 7A). At the chosen λ (Figure 7B), seqNMF typically extracts three significant sequences (Figure (7C). Similarly, our masked cross-validation test indicated good convNMF performance at $K = 3$, with over-fitting starting at $K = 4$ (Figure S7I). The extracted sequences include sequences deployed during syllables of abnormally long and variable durations (Figure 7D-F).

In addition, the extracted sequences exhibit properties not observed in normal adult birds. We see an example of two distinct sequences that sometimes, but not always, co-occur (Figure 7). We observe that a short sequence occurs alone on some syllable renditions, while on other syllable renditions, a second longer sequence is generated simultaneously. This probabilistic overlap of different sequences is highly atypical in normal adult birds [23, 36, 48, 37]. Furthermore, this pattern of neural activity is associated with abnormal variations in syllable structure—in this case resulting in a longer variant of the syllable when both sequences co-occur. This acoustic variation is a characteristic pathology of isolate song [18]. Thus, even though we observe HVC generating some sequences in the absence of a tutor, it appears that these sequences are deployed in a highly abnormal fashion.

Application of seqNMF to a behavioral dataset: song spectrograms

Although we have focused on the application of seqNMF to neural activity data, this method naturally extends to other types of high-dimensional datasets, including behavioral data with applications to neuroscience. The neural mechanisms underlying song production and learning in songbirds is an area of active research. However, the identification and labeling of song syllables in acoustic recordings is challenging, particularly in young birds where song syllables are highly variable. Because automatic segmentation and clustering often fail, song syllables are still routinely labelled by hand [44]. We tested whether seqNMF, applied to a spectrographic representation of zebra finch vocalizations, is able to extract meaningful features in behavioral data. SeqNMF correctly identified repeated acoustic patterns in juvenile songs, placing each distinct syllable type into a different factor (Figure 8). The resulting classifications agree with previously published hand-labeled syllable types [44]. A similar approach could be applied to other behavioral data, for example movement data or human speech, and could facilitate the study of neural mechanisms underlying even earlier and more variable stages of learning. Indeed, convNMF was originally developed for application to spectrograms [56]; notably it has been suggested that auditory cortex may use similar computations to represent and parse natural song statistics [40].

Discussion

As neuroscientists strive to record larger datasets, there is a need for rigorous tools to reveal underlying structure in high-dimensional data [20, 54, 11, 8]. In particular, sequential structure is increasingly regarded as a fundamental property of neuronal circuits [23, 24, 44, 45], but standardized statistical approaches for extracting such structure have not been widely adopted or agreed upon.

Here, we explored a simple matrix factorization-based approach to identify neural sequences [47]. The convNMF model elegantly captures sequential structure in an unsupervised manner [56, 55]. However, in datasets where the number of sequences is

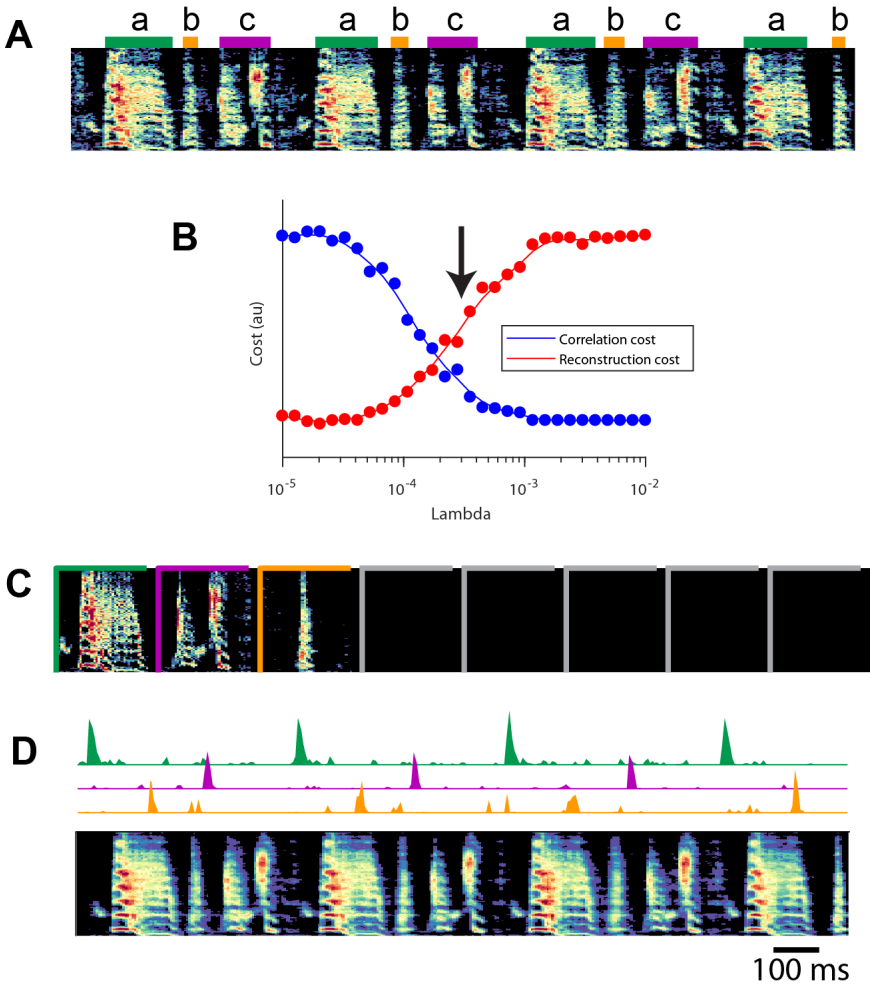

Figure 8. SeqNMF applied to song spectrograms
(A) Spectrogram of juvenile song, with hand-labeled syllable types [44]. **(B)** Reconstruction cost and correlation cost for these data as a function of λ. Arrow denotes $\lambda = 0.0003$, which was used to run seqNMF **(C)** SeqNMF $\mathbf{W}$'s for this song, fit with $K = 8$, $L = 200ms$, $\lambda = 0.0003$. Note that there are three non-empty factors, corresponding to the three hand-labeled syllables a, b, and c. **(D)** SeqNMF $\mathbf{H}$'s (for the three non-empty factors) and seqNMF reconstruction of the song shown in (A) using these factors.

not known, convNMF may return redundant, inefficient, or inconsistent factorizations. In order to resolve these challenges, we introduced a new regularization (penalty) term to encourage the model to identify sparse and non-redundant sequential firing patterns. Furthermore, we carefully explored the robustness of this method to noise and developed procedures for choosing hyperparameters (K and λ) based on cross-validation and assessing the significance of identified sequences based on shuffled null distributions. Our results show that seqNMF is highly robust to many forms of noise. For example, even when (synthetic) neurons participate probabilistically in sequences at a rate of 50%, the model typically identifies factors with greater than 80% similarity to the ground truth (Figure 3A). Additionally, seqNMF performs well even with limited data, successfully extracting sequences that only appear a handful of times in a noisy data stream (Figure S3)

Prior investigations of neural sequences have relied on manual alignment of neural activity to behavioral events, such as animal position for the case of hippocampal and cortical sequences [24, 45], or syllable onset for the case of songbird vocalizations [23]. This approach is not ideally suited for the case of highly variable behaviors, such as in early learning and development [44]. For example, the analysis of neural activity in singing juvenile birds has been challenging because of the difficulty in identifying distinct syllable types on which to perform the temporal alignment. This problem would also apply to isolate song birds because of the pathologically variable nature of their vocalizations. By applying seqNMF, we were able to identify neural sequences without reference to song syllables, enabling future work into the neural basis of singing in isolate birds.

As in many data analysis scenarios, a variety of statistical approaches may be brought to bear on finding sequences in neural data. A classic method is to construct cross-correlogram plots, showing spike time correlations between pairs of neurons at various time lags. However, other forms of spike rate covariation, such as trial-to-trial gain modulation, can produce spurious peaks in this measure [7]; recent work has developed statistical corrections for these effects [51]. After significant pairwise correlations are identified, one can heuristically piece together pairs of neurons with significant interactions into a sequence. This bottom-up approach may be better than seqNMF at detecting sequences involving small numbers of neurons if such microsequences contribute only a small amount of variance in the overall dataset. On the other hand, this bottom-up approach may fail to identify long sequences with high participation noise or jitter in each neuron [49]. One can think of seqNMF as a complementary top-down approach, which performs very well in the high-noise regime since it learns a template sequence at the level of the full population that is robust to noise at the level of individual units.

Statistical models with a dynamical component, such as Hidden Markov Models (HMMs) [38], linear dynamical systems [30], and models with switching dynamics [35], can also capture sequential firing patterns. These methods will typically require many hidden states or latent dimensions to capture sequences, similar to PCA and NMF which require many components to recover sequences. However, since dynamical models are much more constrained than PCA or NMF, they can yield more interpretable results. For example, visualizing the transition matrix of an HMM can provide insight into the order in which hidden states of the model are visited, mapping onto different sequences that manifest in population activity [38]. One advantage of this approach is that it can model sequences that occasionally end prematurely, while seqNMF will always produce the full sequence. On the other hand, this pattern completion property makes seqNMF robust

to participation noise and jitter. In contrast, a standard HMM must pass through each hidden state to model a sequence, and therefore will have trouble whenever one of these hidden states is skipped. Thus, we expect HMMs (or related models) and seqNMF to exhibit complementary strengths and weaknesses.

Another contribution of our work is a natural framework in which to bias factorizations towards parts-based versus events-based solutions. While existing computational work has focused on neural sequences that do not have ensembles of shared neurons, such shared populations have been observed during song learning [44], demonstrating that neural sequences in real biological data can substantially overlap. Such shared sequences can lead to different reasonable factorizations of the data that may correspond to different interpretations of underlying mechanisms. For example, we found that neural sequences in HVC of isolated songbirds are well-described by both parts- or events-based factorizations (figure S9), each of which could correspond to a different biophysical model of sequence generation. This capacity for a combinatorial description of overlapping sequences distinguishes convNMF and seqNMF from clustering methods [22, 39] and methods based on hypothesis testing [49, 51], which seek to identify full snapshots of repeated population firing patterns rather than parts- or events-based representations. Another difference between these methods and seqNMF, particularly when using an events-based factorization, is its ability to model different amplitudes in the sequences by changing the magnitude of the event loadings in $\mathbf{H}$.

More generally, a key strength of seqNMF is that it can be easily tuned to the requirements and goals of a particular analysis. In addition to changing between a parts- and events-based factorization, one can tune the overall sparsity in the model by classic L1 regularization. Future work could incorporate outlier detection into the objective function as has been done in other matrix factorization models [42]. One could also incorporate additional parameters to model changes in neural sequences across trials or days during development or learning of a new behavior, similar to extensions of PCA and NMF to multi-trial data [63]. Thus, adding convolutional structure to factorization-based models of neural data represents a rich opportunity for future developments in statistical methodology.

Despite limiting ourselves to a relatively simple model for the purposes of this paper, we extracted biological insights that were difficult to achieve by other methods in practical experimental datasets. Overall, seqNMF can extract neural sequences from large-scale population recordings without reference to stereotyped behavior or rigid sensory stimuli, enabling the dissection of neural circuit activity during rich and variable animal behaviors.

Acknowledgements

This work was supported by a grant from the Simons Collaboration for the Global Brain, the National Institutes of Health (NIH) [grant number R01 DC009183] and the G. Harold & Leila Y. Mathers Charitable Foundation. ELM received support through the NDSEG Fellowship program. AHB received support through NIH training grant 5T32EB019940-03. MSG received support from the NIH [grant number U19NS104648]. AHW received support from the U.S. Department of Energy Computational Science Graduate Fellowship (CSGF) program. Thanks to Pengcheng Zhou for advice on his CNMF_E calcium data cell extraction algorithm. Thanks to Wiktor Młynarski for helpful convNMF discussions. Thanks to Michael Stetner, Galen Lynch, Nhat Le, Dezhe Jin, Edward Nieh, Adam Charles

and Jane Van Velden for comments on the manuscript and on our code package. Special thanks to the 2017 Methods in Computational Neuroscience course [supported by NIH grant R25 MH062204 and Simons Foundation] at the Woods Hole Marine Biology Lab, where this collaboration was started.

Author contributions

ELM, AHB, AHW, MSG and MSF conceived the project, based on previous discussions of MSG, MSF and ELM. ELM, AHB and MSF designed and tested the seqNMF regularizers, the method for validating the significance of sequences in a held-out dataset, and the method for choosing λ. ELM, AHB, AHW, and MSF designed and tested the method for measuring RMSE on a masked test set. ELM and AHB wrote the algorithm and demo code. ELM and NID collected the imaging data in singing birds. ELM and SG analyzed imaging data. All authors contributed to writing the manuscript.

Methods and Materials

Table of key resources

Key resources, and references for how to access them, are listed in Table 2.

Contact for resource sharing

Further requests should be directed to Michale Fee (fee@mit.edu).

Software and data availability

Our seqNMF MATLAB code is publicly available as a github repository, along with some of our data for demonstration:

https://github.com/FeeLab/seqNMF

The repository includes the seqNMF function, as well as helper functions for selecting λ, testing the significance of factors, plotting, and other functions. It also includes a demo script that goes through an example of how to select λ for a new dataset, test for significance of factors, plot the seqNMF factorization, switch between parts-based and events-based factorizations, and calculate cross-validated performance on a masked test set.

We plan to post more of our data publicly on the CRCNS data-sharing platform.

Generating simulated data

We simulated neural sequences containing between 1 and 10 distinct neural sequences in the presence of various noise conditions. Each neural sequence was made up of 10 consecutively active neurons, each separated by three timebins. The binary activity matrix was convolved with an exponential kernel ($\tau = 10$ timebins) to resemble neural calcium imaging activity.

SeqNMF algorithm details

Our algorithm for seqNMF (convNMF with additional regularization to promote efficient factorizations) is a direct extension of the multiplicative update convNMF algorithm [56], and draws on previous work regularizing NMF to encourage factor orthogonality [10].

Table 2. Key resources

Software/algorithm	Source	Link to code
seqNMF	This paper	https://github.com/FeeLab/seqNMF
convNMF	[56, 55]	https://github.com/colinvaz/nmf-too
Sparse convNMF	[43, 50]	https://github.com/colinvaz/nmf-too
Soft orthogonal NMF	[10]	
Other NMF extensions	[12]	
NMF	[34]	
CNMF_E (cell extraction)	[66]	https://github.com/zhoupc/CNMF_E
MATLAB	MathWorks	www.mathworks.com
Dataset	**Source**	**Link to data**
HVC, Isolate songbird	This paper	To be uploaded to CRCNS after publ
Hippocampus, running wheel task (rat 1)	[2]	/hc-5
hline Hippocampus, running wheel task (rat 2)	[1]	https://crcns.org/data-sets/hc/hc-3
Other	**Source**	**Link**
Zebra finches (Taeniopygia guttata)	MIT animal facility	
AAV9.CAG.GCaMP6f.WPRE.SV40	[9]	https://pennvectorcore.med.upenn.
Miniature microscope	Inscopix	https://www.inscopix.com/nvista

Table 3. Regularized NMF and convNMF: cost functions and algorithms

NMF	
$\mathscr{L} = \frac{1}{2}\|\|\widetilde{\mathbf{X}} - \mathbf{X}\|\|_2^2 + \mathscr{R}$	$\mathbf{W} \leftarrow \mathbf{W} \times \frac{\mathbf{X}\mathbf{H}^\top}{\widetilde{\mathbf{X}}\mathbf{H}^\top + \frac{d\mathscr{R}}{d\mathbf{W}}}$
$\widetilde{\mathbf{X}} = \mathbf{W}\mathbf{H}$	$\mathbf{H} \leftarrow \mathbf{H} \times \frac{\mathbf{W}^\top\mathbf{X}}{\mathbf{W}^\top\widetilde{\mathbf{X}} + \frac{d\mathscr{R}}{d\mathbf{H}}}$
convNMF	
$\mathscr{L} = \frac{1}{2}\|\|\widetilde{\mathbf{X}} - \mathbf{X}\|\|_2^2 + \mathscr{R}$	$\mathbf{W}_{..\ell} \leftarrow \mathbf{W}_{..\ell} \times \frac{\overrightarrow{\mathbf{X}\ \mathbf{H}}^{\ell\top}}{\widetilde{\mathbf{X}}\ \overrightarrow{\mathbf{H}}^{\ell\top} + \frac{d\mathscr{R}}{d\mathbf{W}_{..\ell}}}$
$\widetilde{\mathbf{X}} = \mathbf{W} \circledast \mathbf{H}$	$\mathbf{H} \leftarrow \mathbf{H} \times \frac{\mathbf{W}^\top \circledast \mathbf{X}}{\mathbf{W}^\top \circledast \widetilde{\mathbf{X}} + \frac{d\mathscr{R}}{d\mathbf{H}}}$
$L1$ regularization for $\mathbf{H}$ ($L1$ for $\mathbf{W}$ is analogous)	
$\mathscr{R} = \lambda\|\|\mathbf{H}\|\|_1$	$\frac{d\mathscr{R}}{d\mathbf{W}_{..\ell}} = 0$
	$\frac{d\mathscr{R}}{d\mathbf{H}} = \lambda\mathbf{1}$
Soft orthogonality for $\mathbf{H}$	
$\mathscr{R} = \frac{\lambda}{2}\|\|\mathbf{H}\mathbf{H}^\top\|\|_{1,i\neq j}$	$\frac{d\mathscr{R}}{d\mathbf{W}_{..\ell}} = 0$
	$\frac{d\mathscr{R}}{d\mathbf{H}} = \lambda(\mathbf{1} - \mathbf{I})\mathbf{H}$
Smoothed soft orthogonality for $\mathbf{H}$ (favors 'events-based')	
$\mathscr{R} = \frac{\lambda}{2}\|\|\mathbf{H}\mathbf{S}\mathbf{H}^\top\|\|_{1,i\neq j}$	$\frac{d\mathscr{R}}{d\mathbf{W}_{..\ell}} = 0$
	$\frac{d\mathscr{R}}{d\mathbf{H}} = \lambda(\mathbf{1} - \mathbf{I})\mathbf{H}\mathbf{S}$
Smoothed soft orthogonality for $\mathbf{W}$ (favors 'parts-based')	
$\mathscr{R} = \frac{\lambda}{2}\|\|\mathbf{W}_{flat}^\top\mathbf{W}_{flat}\|\|_{1,i\neq j}$	$\frac{d\mathscr{R}}{d\mathbf{W}_{..\ell}} = \lambda\mathbf{W}_{flat}(\mathbf{1} - \mathbf{I})$
where $(\mathbf{W}_{flat})_{nk} = \sum_\ell \mathbf{W}_{nk\ell}$	$\frac{d\mathscr{R}}{d\mathbf{H}} = 0$
Smoothed cross-factor orthogonality (main seqNMF $\mathscr{R}$)	
$\mathscr{R} = \lambda\|\|\mathbf{W}^\top \circledast \mathbf{X}\mathbf{S}\mathbf{H}^\top\|\|_{1,i\neq j}$	$\frac{d\mathscr{R}}{d\mathbf{W}_{..\ell}} = \lambda\overleftarrow{\mathbf{X}}^{\ell}\mathbf{S}\mathbf{H}^\top(\mathbf{1} - \mathbf{I})$
	$\frac{d\mathscr{R}}{d\mathbf{H}} = \lambda(\mathbf{1} - \mathbf{I})\mathbf{W}^\top \circledast \mathbf{X}\mathbf{S}$

The uniqueness and consistency of traditional NMF has been better studied than convNMF, but in special cases, NMF has a unique solution comprised of sparse, 'parts-based' features that can be consistently identified by known algorithms [17, 4]. However, this ideal scenario does not hold in many practical settings. In these cases, NMF is sensitive to initialization, resulting in potentially inconsistent features. This problem can be addressed by introducing additional constraints or regularization terms that encourage the model to extract particular, e.g. sparse or approximately orthogonal, features [27, 31]. Both theoretical work and empirical observations suggest that these modifications result in more consistently identified features [58, 31].

For seqNMF, we added to the convNMF cost function a term that promotes competition between overlapping factors, resulting in the following cost function:

$$(\widetilde{\mathbf{W}}, \widetilde{\mathbf{H}}) = \arg\min_{\mathbf{W},\mathbf{H}} \left(||\widetilde{\mathbf{X}} - \mathbf{X}||_F^2 + \lambda ||(\mathbf{W} \overset{\top}{\circledast} \mathbf{X})\mathbf{S}\mathbf{H}^\top||_{1,i\neq j} \right) \tag{8}$$

We derived the following multiplicative update rules for $\mathbf{W}$ and $\mathbf{H}$ (Appendix 1):

$$\mathbf{W}_{\cdot\cdot\ell} \leftarrow \mathbf{W}_{\cdot\cdot\ell} \times \frac{\mathbf{X} \left(\overset{\ell\rightarrow}{\mathbf{H}} \right)^\top}{\widetilde{\mathbf{X}} \left(\overset{\ell\rightarrow}{\mathbf{H}} \right)^\top + \lambda \overset{\leftarrow\ell}{\mathbf{X}} \mathbf{S}\mathbf{H}^\top(\mathbf{1} - \mathbf{I})} \tag{9}$$

$$\mathbf{H} \leftarrow \mathbf{H} \times \frac{\mathbf{W} \overset{\top}{\circledast} \mathbf{X}}{\mathbf{W} \overset{\top}{\circledast} \widetilde{\mathbf{X}} + \lambda(\mathbf{1} - \mathbf{I})(\mathbf{W} \overset{\top}{\circledast} \mathbf{X}\mathbf{S})} \tag{10}$$

Where the division and $\times$ are element-wise. The operator $\overset{\ell\rightarrow}{(\cdot)}$ shifts a matrix in the $\rightarrow$ direction by ℓ timebins, i.e. a delay by ℓ timebins, and $\overset{\leftarrow\ell}{(\cdot)}$ shifts a matrix in the $\leftarrow$ direction by ℓ timebins (notation summary, Table 1). Note that multiplication with the $K \times K$ matrix $(\mathbf{1} - \mathbf{I})$ effectively implements factor competition because it places in the kth row a sum across all other factors. These update rules are derived in Appendix 1 by taking the derivative of the cost function in Equation 8.

In addition to the multiplicative updates outlined in Table 3, we also renormalize so rows of $\mathbf{H}$ have unit norm; shift factors to be centered in time such that the center of mass of each $\mathbf{W}$ pattern occurs in the middle; and in the final iteration run one additional step of unregularized convNMF to prioritize the cost of reconstruction error over the regularization (Algorithm 1). This final step is done to correct a minor suppression in the amplitude of some peaks in $\mathbf{H}$ that may occur within $2L$ timebins of neighboring sequences.

Calculating consistency

The consistency between two factorizations measures the extent to which it is possible to create a one-to-one match between factors in factorization A and factors in factorization B. Specifically, given two factorizations $(\mathbf{W}^A, \mathbf{H}^A)$ and $(\mathbf{W}^B, \mathbf{H}^B)$ respectively, consistency is measured with the following procedure:

1. For each factor number k, compute the part of the reconstruction explained by this factor in each reconstruction, $\widetilde{\mathbf{X}}_k^A = \mathbf{W}_{\cdot k \cdot}^A \circledast \mathbf{H}_{k \cdot}^A$ and $\widetilde{\mathbf{X}}_k^B = \mathbf{W}_{\cdot k \cdot}^B \circledast \mathbf{H}_{k \cdot}^B$

Algorithm 1: SeqNMF

Input: Data matrix $\mathbf{X}$, number of factors K, factor duration L, regularization strength λ

Output: Factor exemplars $\mathbf{W}$, and factor timecourses $\mathbf{H}$

```
Initialize W and H randomly
Iter = 1
while (Iter < maxIter) & (Δ cost > tolerance) do
    Update H using multiplicative update from Table 3
    Shift W and H to center W's in time
    Renormalize W and H so rows of H have unit norm
    Update W using multiplicative update from Table 3
    Iter = Iter+1
Do one final unregularized convNMF update of W and H
return
```

2. Reshape $\widetilde{\mathbf{X}}_k^A$ and $\widetilde{\mathbf{X}}_k^B$ into vectors containing all the elements of each matrix respectively, then compute $\mathbf{C}$, a $K \times K$ correlation matrix where $\mathbf{C_{ij}}$ is the correlation between the vectorized $\widetilde{\mathbf{X}}_i^A$ and $\widetilde{\mathbf{X}}_j^B$
3. Permute the factors greedily so factor 1 is the best matched pair of factors, factor 2 is the best matched pair of the remaining factors, etc. The quality of the match is measured by the correlation between the reconstructions computed using just each factor individually.
4. Measure consistency as the ratio of the power (sum of squared matrix elements) contained on the diagonal of the permuted $\mathbf{C}$ matrix to the total power in $\mathbf{C}$

Thus, two factorizations are perfectly consistent when there exists a permutation of factor numbers for which there is a one-to-one match between what parts of the reconstruction are explained by each factor.

Testing the significance of each factor on held-out data

In order to test whether a factor is significantly present in held-out data, we measure the distribution across timebins of the overlaps of the factor with the held-out data, and compare the skewness of this distrubution to the null case (Figure S1). Overlap with the data is measured as $\mathbf{W}^T \circledast \mathbf{X}$, so this quantity will be high at timepoints when the sequence occurs, producing a distribution of $\mathbf{W}^T \circledast \mathbf{X}$ with high skew. In contrast, a distribution of overlaps exhibiting low skew indicates a sequence is not present in the data, since there are few timepoints of particularly high overlap. We estimate what skew levels would appear by chance by constructing null factors where temporal relationships between neurons have been eliminated. To create such null factors, we start from the real factors then circularly shift the timecourse of each neuron by a random amount between 0 and L. We measure the skew of the overlap distributions for each null factor, and ask whether the skew we measured for the real factor is significant at p-value α, that is, if it exceeds the $((1-\frac{\alpha}{K}) \times 100)^{th}$ percentile of the null skews. Note the required Bonferroni correction for K comparisons when testing K factors.

Choosing appropriate parameters for a new dataset

Choice of appropriate parameters (λ, K and L) will depend on the data type (sequence length, number, and density; amount of noise; etc.).

In practice, we find that results are relatively robust to choice of parameters. When K or L is set larger than necessary, seqNMF tends to simply leave the unnecessary factors or time bins empty. For λ, the goal is to find the 'sweet spot' (Figure 4) to explain as much data as possible while still producing sensible factorizations, that is, minimally correlated factors, with low values of $||(\mathbf{W} \overset{\top}{\circledast} \mathbf{X})\mathbf{S}\mathbf{H}^{\top}||_{1,i\neq j}$. Our software package includes demo code for determining the best parameters for a new type of data, using the following strategy:

1. Start with K slightly larger than the number of sequences anticipated in the data
2. Start with L slightly longer than the maximum expected factor length
3. Run seqNMF for a range of λ's, and for each λ measure the reconstruction error $\left(||\mathbf{X} - \mathbf{W} \circledast \mathbf{H}||_F^2\right)$ and the factor competition regularization term $\left(||(\mathbf{W} \overset{\top}{\circledast} \mathbf{X})\mathbf{S}\mathbf{H}^{\top}||_{1,i\neq j}\right)$
4. Choose a λ slightly above the crossover point λ_0
5. Decrease K if desired, as otherwise some factors will be consistently empty
6. Decrease L if desired, as otherwise some time bins will consistently be empty

In some applications, achieving the desired accuracy may depend on choosing a λ that allows some inconsistency. It is possible to deal with this remaining inconsistency by comparing factors produced by different random initializations, and only considering factors that arise from several different initializations, a strategy that has been previously applied to standard convNMF on neural data [47].

During validation of our procedure for choosing λ, we compared factorizations to ground truth sequences as shown in Figure 4. To find the optimal lambda we used the product of two curves. The first curve was obtained by calculating the fraction of fits in which the true number of sequences was recovered as a function of λ. The second curve was obtained by calculating similarity to ground truth as a function of λ. Similarity to ground truth is measured as the consistency the factorization and the noiseless sequences used to generate the data. The product of these two curves was smoothed using a three-sample boxcar sliding window, and the width was found as the values of λ on either side of the peak value that correspond mose closely to the half-max points of the curve.

Measuring performance on noisy data by comparing seqNMF sequences to ground-truth sequences

We wanted to measure the ability of seqNMF to recover ground-truth sequences even when the sequences are obstructed by noise. Our noisy data consisted of two ground-truth sequences, obstructed by a variety of noise types. We first took the top seqNMF factor, and made a reconstruction with only this factor. We then measured the correlation between this reconstruction and reconstructions generated from each of the ground-truth factors, and chose the best match. Next, we measured the correlation between the remaining ground-truth reconstruction and the second seqNMF factor. The mean of these two correlations was used as a measure of similarity between the seqNMF factorization and the ground-truth (noiseless) sequences.

Testing generalization of factorization to randomly held-out (masked) data entries

The data matrix $\mathbf{X}$ was divided into training data and test data by randomly selecting 5 or 10% of matrix entries to hold out. Specifically, the objective function (equation 5, in the Results section) was modified to:

$$\underset{\mathbf{W},\mathbf{H}}{\arg\min} \quad ||\mathbf{M} \times (\mathbf{W} \circledast \mathbf{H} - \mathbf{X})||_F^2 + \mathscr{R} \tag{11}$$

where $\times$ indicates elementwise multiplication (Hadamard product) and $\mathbf{M}$ is a binary matrix with 5 or 10% of the entries randomly selected to be zero (held-out test set) and the remaining 95 or 90% set to one (training set). To search for a solution, we reformulate this optimization problem as:

$$\begin{aligned} \underset{\mathbf{W},\mathbf{H},\mathbf{Z}}{\arg\min} \quad & ||\mathbf{W} \circledast \mathbf{H} - \mathbf{Z}||_F^2 + \mathscr{R} \\ \text{subject to} \quad & \mathbf{M} \times \mathbf{Z} = \mathbf{M} \times \mathbf{X} \end{aligned} \tag{12}$$

where we have introduced a new optimization variable $\mathbf{Z}$, which can be thought of as a surrogate dataset that is equal to the ground truth data only on the training set. The goal is now to minimize the difference between the model estimate, $\widetilde{\mathbf{X}} = \mathbf{W} \circledast \mathbf{H}$, and the surrogate, $\mathbf{Z}$, while constraining $\mathbf{Z}$ to equal $\mathbf{X}$ at unmasked elements (where $m_{ij} = 1$) and allowing $\mathbf{Z}$ to be freely chosen at masked elements (where $m_{ij} = 0$). Clearly, at masked elements, the best choice is to make $\mathbf{Z}$ equal to the current model estimate $\widetilde{\mathbf{X}}$ as this minimizes the cost function without violating the constraint. This leads to the following update rules which are applied cyclically to update $\mathbf{Z}$, $\mathbf{W}$, and $\mathbf{H}$.

$$\mathbf{Z}_{nt} \leftarrow \begin{cases} \mathbf{X}_{nt} & \text{if } \mathbf{M}_{nt} = 1 \\ (\mathbf{W} \circledast \mathbf{H})_{nt} & \text{if } \mathbf{M}_{nt} = 0 \end{cases} \tag{13}$$

$$\mathbf{W}_{\cdot\cdot\ell} \leftarrow \mathbf{W}_{\cdot\cdot\ell} \times \frac{\mathbf{Z} \left(\overset{\ell\rightarrow}{\mathbf{H}} \right)^\top}{\widetilde{\mathbf{X}} \left(\overset{\ell\rightarrow}{\mathbf{H}} \right)^\top + \lambda \overset{\leftarrow\ell}{\mathbf{Z}} \mathbf{S}\mathbf{H}^\top(\mathbf{1} - \mathbf{I})} \tag{14}$$

$$\mathbf{H} \leftarrow \mathbf{H} \times \frac{\mathbf{W}^\top \circledast \mathbf{Z}}{\mathbf{W}^\top \circledast \widetilde{\mathbf{X}} + \lambda(\mathbf{1} - \mathbf{I})(\mathbf{W}^\top \circledast \mathbf{Z}\mathbf{S})} \tag{15}$$

The measure used for testing generalization performance was RMSE. For the testing phase, RMSE was computed from the difference between $\widetilde{\mathbf{X}}$ and the data matrix $\mathbf{X}$ only for held-out entries.

Algorithm speed

In practice, our algorithm converges rapidly: fewer than 100 iterations on a typical 150 neuron by 10,000 time point data matrix. Typically, 100 iterations on such data take less than 30 seconds on a standard PC. However, applications to much larger datasets may require faster performance. In these cases, we recommend running seqNMF on smaller subsets of the dataset, perhaps by incorporating seqNMF regularization into an online version of convNMF [62], and/or parallelizing the algorithm by running it on shorter

datasets and merging/recombining factors that are common across these shorter runs (finding common factors by e.g. [47]).

Notes on data preprocessing

While seqNMF is generally quite robust, proper preprocessing of the data can be important to obtaining reasonable factorizations on real neural data. A key principle is that, in minimizing the reconstruction error, seqNMF is most strongly influenced by parts of the data that exhibit high variance. This can be problematic if the regions of interest in the data have relatively low amplitude. For example, high firing rate neurons may be prioritized over those with lower firing rate. As an alternative to subtracting the mean firing rate of each neuron, which would introduce negative values, neurons could be normalized divisively or by subtracting off a NMF reconstruction fit in method that forces a non-negative residual [32]. Additionally, variations in behavioral state may lead to seqNMF factorizations that prioritize regions of the data with high variance and neglect other regions. It may be possible to mitigate these effects by normalizing data, or by restricting analysis to particular subsets of the data, either by time or by neuron.

Hippocampus data

The hippocampal data we used was collected in the Buzsaki lab [2, 1], and is publicly available on the Collaborative Research in Computational Neuroscience (CRCNS) Data sharing website. The dataset we refer to as 'Rat 1' is in the hc-5 dataset, and the dataset we refer to as 'Rat 2' is in the hc-3 and dataset. Before running seqNMF, we processed the data by convolving the raw spike trains with a gaussian kernel of standard deviation 100ms.

Animal care and use

We used male zebra finches (*Taeniopygia guttata*) from the MIT zebra finch breeding facility (Cambridge, MA). Animal care and experiments were carried out in accordance with NIH guidelines, and reviewed and approved by the Massachusetts Institute of Technology Committee on Animal Care (protocol 0715-071-18).

In order to prevent exposure to a tutor song, birds were foster-raised by female birds, which do not sing, starting on or before post-hatch day 15. For experiments, birds were housed singly in custom-made sound isolation chambers.

Calcium imaging

The calcium indicator GCaMP6f was expressed in HVC by intercranial injection of the viral vector AAV9.CAG.GCaMP6f.WPRE.SV40 [9] into HVC. In the same surgery, a cranial window was made using a GRIN (gradient index) lens (1mm diamenter, 4mm length, Inscopix). After at least one week, in order to allow for sufficient viral expression, recordings were made using the Inscopix nVista miniature fluorescent microscope.

Neuronal activity traces were extracted from raw fluorescence movies using the CNMF_E algorithm, a constrained non-negative matrix factorization algorithm specialized for microendoscope data by including a local background model to remove activity from out-of-focus cells [66].

We performed several preprocessing steps before applying seqNMF to functional calcium traces extracted by CNMF_E. First, we estimated burst times from the raw traces

by deconvolving the traces using an AR-2 process. The deconvolution parameters (time constants and noise floor) were estimated for each neuron using the CNMF_E code package [66]. Some neurons exhibited larger peaks than others, likely due to different expression levels of the calcium indicator. Since seqNMF would prioritize the neurons with the most power, we renormalized by dividing the signal from each neuron by the sum of the maximum value of that row and the 95^{th} percentile of the signal across all neurons. In this way, neurons with larger peaks were given some priority, but not much more than that of neurons with weaker signals.

References

[1] Multiple single unit recordings from different rat hippocampal and entorhinal regions while the animals were performing multiple behavioral tasks. CRCNSorg. 2013; https://crcns.org/data-sets/hc/hc-5/about-hc-5, doi: http://dx.doi.org/10.6080/K09G5JRZ.

[2] Simultaneous extracellular recordings from left and right hippocampal areas CA1 and right entorhinal cortex from a rat performing a left / right alternation task and other behaviors. CRCNSorg. 2015; https://crcns.org/data-sets/hc/hc-5/about-hc-5, doi: http://dx.doi.org/10.6080/K0KS6PHF.

[3] **Aronov D**, Andalman AS, Fee MS. A specialized forebrain circuit for vocal babbling in the juvenile songbird. Science (New York, NY). 2008 may; 320(5876):630–4. http://www.ncbi.nlm.nih.gov/pubmed/18451295, doi: 10.1126/science.1155140.

[4] **Arora S**, Ge R, Kannan R, Moitra A. Computing a Nonnegative Matrix Factorization – Provably. ArXiv e-prints. 2011 nov; .

[5] **Bapi RS**, Pammi VSC, Miyapuram KP, Ahmed. Investigation of sequence processing: A cognitive and computational neuroscience perspective. Current Science. 2005; 89(10):1690–1698. http://www.jstor.org/stable/24111208.

[6] **Bro R**, Kjeldahl K, Smilde AK, Kiers HAL. Cross-validation of component models: A critical look at current methods. Analytical and Bioanalytical Chemistry. 2008 Mar; 390(5):1241–1251. https://doi.org/10.1007/s00216-007-1790-1, doi: 10.1007/s00216-007-1790-1.

[7] **Brody CD**. Correlations Without Synchrony. Neural Computation. 1999; 11(7):1537–1551. https://doi.org/10.1162/089976699300016133, doi: 10.1162/089976699300016133.

[8] **Bzdok D**, Yeo BTT. Inference in the age of big data: Future perspectives on neuroscience. NeuroImage. 2017; 155(Supplement C):549–564. http://www.sciencedirect.com/science/article/pii/S1053811917303816, doi: https://doi.org/10.1016/j.neuroimage.2017.04.061.

[9] **Chen TW**, Wardill TJ, Sun Y, Pulver SR, Renninger SL, Baohan A, Schreiter ER, Kerr RA, Orger MB, Jayaraman V, Looger LL, Svoboda K, Kim DS. Ultrasensitive fluorescent proteins for imaging neuronal activity. Nature. 2013 jul; 499(7458):295–300. http://www.ncbi.nlm.nih.gov/pubmed/23868258http://www.pubmedcentral.nih.gov/articlerender.fcgi?artid=PMC3777791, doi: 10.1038/nature12354.

[10] **Chen Z**, Cichocki A. Nonnegative matrix factorization with temporal smoothness and/or spatial decorrelation constraints. In: Laboratory for Advanced Brain Signal Processing, RIKEN, Tech. Rep; 2005. .

[11] **Churchland AK**, Abbott LF. Conceptual and technical advances define a key moment for theoretical neuroscience. Nature Neuroscience. 2016 feb; 19(3):348–349. http://www.nature.com/doifinder/10.1038/nn.4255, doi: 10.1038/nn.4255.

[12] **Cichocki A.** Nonnegative Matrix and Tensor Factorizations : Applications to Exploratory Multi-way Data Analysis and Blind Source Separation. Wiley; 2009. http://libproxy.mit.edu/login?url=http://search.ebscohost.com/login.aspx?direct=true{&}db=nlebk{&}AN=287301{&}site=ehost-live{&}scope=site.

[13] **Clegg BA**, Digirolamo GJ, Keele SW. Sequence learning. Trends in cognitive sciences. 1998 aug; 2(8):275–81. http://www.ncbi.nlm.nih.gov/pubmed/21227209, doi: 10.1016/S1364-6613(98)01202-9.

[14] **Cui Y**, Ahmad S, Hawkins J. Continuous Online Sequence Learning with an Unsupervised Neural Network Model. Neural Computation. 2016; 28(11):2474–2504. https://doi.org/10.1162/NECO{_}a{_}00893, doi: 10.1162/NECO_a_00893.

[15] **Cunningham JP**, Yu BM. Dimensionality reduction for large-scale neural recordings. Nature Neuroscience. 2014 nov; 17(11):1500–1509. http://www.nature.com/articles/nn.3776, doi: 10.1038/nn.3776.

[16] **Diba K**, Buzsáki G. Hippocampal Network Dynamics Constrain the Time Lag between Pyramidal Cells across Modified Environments. Journal of Neuroscience. 2008; 28(50):13448–13456. http://www.jneurosci.org/content/28/50/13448, doi: 10.1523/JNEUROSCI.3824-08.2008.

[17] **Donoho D**, Stodden V. When Does Non-Negative Matrix Factorization Give a Correct Decomposition into Parts? In: Thrun S, Saul LK, Schölkopf B, editors. Advances in Neural Information Processing Systems 16 MIT Press; 2004.p. 1141–1148. http://papers.nips.cc/paper/2463-when-does-non-negative-matrix-factorization-give-a-correct-decomposition-into-parts.pdf.

[18] **Fehér O**, Wang H, Saar S, Mitra PP, Tchernichovski O. De novo establishment of wild-type song culture in the zebra finch. Nature. 2009 may; 459(7246):564–568. http://www.pubmedcentral.nih.gov/articlerender.fcgi?artid=2693086{&}tool=pmcentrez{&}rendertype=abstract, doi: 10.1038/nature07994.

[19] **Fujisawa S**, Amarasingham A, Harrison M, Buzsáki G. Behavior-dependent short-term assembly dynamics in the medial prefrontal cortex. Nature Neuroscience. 2008; 11(7):823–833. https://www.nature.com/articles/nn.2134, doi: 10.1038/nn.2134.

[20] **Gao P**, Ganguli S. On Simplicity and Complexity in the Brave New World of Large-Scale Neuroscience. ArXiv e-prints. 2015 mar; .

[21] **Gerstein GL**, Williams ER, Diesmannc M, Gründ S, Trengove C. Detecting synfire chains in parallel spike data. Journal of Neuroscience Methods. 2012; 206:54–64. doi: 10.1016/j.jneumeth.2012.02.003.

[22] **Grossberger L**, Battaglia FP, Vinck M. Unsupervised clustering of temporal patterns in high-dimensional neuronal ensembles using a novel dissimilarity measure. bioRxiv. 2018; https://www.biorxiv.org/content/early/2018/04/30/252791, doi: 10.1101/252791.

[23] **Hahnloser RHR**, Kozhevnikov AA, Fee MS. An ultra-sparse code underlies the generation of neural sequences in a songbird. Nature. 2002 sep; 419(6902):65–70. http://www.ncbi.nlm.nih.gov/pubmed/12214232, doi: 10.1038/nature00974.

[24] **Harvey CD**, Coen P, Tank DW. Choice-specific sequences in parietal cortex during a virtual-navigation decision task. Nature. 2012 mar; 484(7392):62–68. http://www.ncbi.nlm.nih.gov/pubmed/22419153http://www.pubmedcentral.nih.gov/articlerender.fcgi?artid=PMC3321074http://www.nature.com/doifinder/10.1038/nature10918, doi: 10.1038/nature10918.

941 [25] **Hastie T**, Tibshirani R, Friedman JHJH. The elements of statistical learning : data mining,
942 inference, and prediction. Springer; 2009.

943 [26] **Hawkins J**, Ahmad S. Why Neurons Have Thousands of Synapses, a Theory of Sequence
944 Memory in Neocortex. Frontiers in Neural Circuits. 2016; 10:23. https://www.frontiersin.org/
945 article/10.3389/fncir.2016.00023, doi: 10.3389/fncir.2016.00023.

946 [27] **Huang K**, Sidiropoulos ND, Swami A. Non-Negative Matrix Factorization Revisited: Uniqueness
947 and Algorithm for Symmetric Decomposition. IEEE Transactions on Signal Processing. 2014
948 jan; 62(1):211–224. doi: 10.1109/TSP.2013.2285514.

949 [28] **Janata P**, Grafton ST. Swinging in the brain: shared neural substrates for behaviors related to
950 sequencing and music. Nature Neuroscience. 2003 jul; 6(7):682–687. http://www.nature.com/
951 articles/nn1081, doi: 10.1038/nn1081.

952 [29] **Jun JJ**, Steinmetz NA, Siegle JH, Denman DJ, Bauza M, Barbarits B, Lee AK, Anastassiou CA,
953 Andrei A, Aydın Ç, Barbic M, Blanche TJ, Bonin V, Couto J, Dutta B, Gratiy SL, Gutnisky DA,
954 Häusser M, Karsh B, Ledochowitsch P, et al. Fully integrated silicon probes for high-density
955 recording of neural activity. Nature. 2017 nov; 551(7679):232–236. http://www.nature.com/
956 doifinder/10.1038/nature24636, doi: 10.1038/nature24636.

957 [30] **Kao JC**, Nuyujukian P, Ryu SI, Churchland MM, Cunningham JP, Shenoy KV. Single-trial dynamics
958 of motor cortex and their applications to brain-machine interfaces. Nature Communications.
959 2015 07; 6:7759 EP –. http://dx.doi.org/10.1038/ncomms8759.

960 [31] **Kim J**, Park H. Sparse Nonnegative Matrix Factorization for Clustering. In: ; 2008. .

961 [32] **Kim M**, Smaragdis P. Efficient model selection for speech enhancement using a deflation
962 method for Nonnegative Matrix Factorization. In: 2014 IEEE Global Conference on Signal
963 and Information Processing (GlobalSIP) IEEE; 2014. p. 537–541. http://ieeexplore.ieee.org/
964 document/7032175/, doi: 10.1109/GlobalSIP.2014.7032175.

965 [33] **Kim TH**, Zhang Y, Lecoq J, Jung JC, Li J, Zeng H, Niell CM, Schnitzer MJ. Long-Term
966 Optical Access to an Estimated One Million Neurons in the Live Mouse Cortex.
967 Cell reports. 2016 dec; 17(12):3385–3394. http://www.ncbi.nlm.nih.gov/pubmed/
968 28009304http://www.pubmedcentral.nih.gov/articlerender.fcgi?artid=PMC5459490, doi:
969 10.1016/j.celrep.2016.12.004.

970 [34] **Lee DD**, Seung HS. Learning the parts of objects by non-negative matrix factorization. Nature.
971 1999; 401(6755):788–791.

972 [35] **Linderman S**, Johnson M, Miller A, Adams R, Blei D, Paninski L. Bayesian Learning and Inference
973 in Recurrent Switching Linear Dynamical Systems. In: Singh A, Zhu J, editors. Proceedings of the
974 20th International Conference on Artificial Intelligence and Statistics, vol. 54 of Proceedings
975 of Machine Learning Research Fort Lauderdale, FL, USA: PMLR; 2017. p. 914–922. http:
976 //proceedings.mlr.press/v54/linderman17a.html.

977 [36] **Long MA**, Jin DZ, Fee MS. Support for a synaptic chain model of neuronal sequence generation.
978 Nature. 2010 nov; 468(7322):394–399. http://www.pubmedcentral.nih.gov/articlerender.fcgi?
979 artid=2998755{&}tool=pmcentrez{&}rendertype=abstract, doi: 10.1038/nature09514.

980 [37] **Lynch G**, Okubo T, Hanuschkin A, Hahnloser RR, Fee M. Rhythmic Continuous-Time
981 Coding in the Songbird Analog of Vocal Motor Cortex. Neuron. 2016 may; 90(4):877–
982 892. http://www.ncbi.nlm.nih.gov/pubmed/27196977http://linkinghub.elsevier.com/retrieve/
983 pii/S0896627316301088, doi: 10.1016/j.neuron.2016.04.021.

[38] **Maboudi K**, Ackermann E, Pfeiffer BE, Foster DJ, Diba K, Kemere C. Uncovering temporal structure in hippocampal output patterns. bioRxiv. 2018; https://www.biorxiv.org/content/early/2018/01/04/242594, doi: 10.1101/242594.

[39] **van der Meij R**, Voytek B. Uncovering Neuronal Networks Defined by Consistent between-Neuron Spike Timing from Neuronal Spike Recordings. eNeuro. 2018; http://www.eneuro.org/content/early/2018/05/08/ENEURO.0379-17.2018, doi: 10.1523/ENEURO.0379-17.2018.

[40] **Młynarski W**, McDermott JH. Learning Midlevel Auditory Codes from Natural Sound Statistics. Neural Computation. 2018 mar; 30(3):631–669. https://www.mitpressjournals.org/doi/abs/10.1162/neco{_}a{_}01048, doi: 10.1162/neco_a_01048.

[41] **Mokeichev A**, Okun M, Barak O, Katz Y, Ben-Shahar O, Lampl I. Stochastic Emergence of Repeating Cortical Motifs in Spontaneous Membrane Potential Fluctuations In Vivo. Neuron. 2007; 53(3):413 – 425. http://www.sciencedirect.com/science/article/pii/S0896627307000372, doi: https://doi.org/10.1016/j.neuron.2007.01.017.

[42] **Netrapalli P**, U N N, Sanghavi S, Anandkumar A, Jain P. Non-convex Robust PCA. In: Ghahramani Z, Welling M, Cortes C, Lawrence ND, Weinberger KQ, editors. Advances in Neural Information Processing Systems 27 Curran Associates, Inc.; 2014.p. 1107–1115. http://papers.nips.cc/paper/5430-non-convex-robust-pca.pdf.

[43] **O'Grady PD**, Pearlmutter BA. Convolutive Non-Negative Matrix Factorisation with a Sparseness Constraint. In: 2006 16th IEEE Signal Processing Society Workshop on Machine Learning for Signal Processing; 2006. p. 427–432. doi: 10.1109/MLSP.2006.275588.

[44] **Okubo TS**, Mackevicius EL, Payne HL, Lynch GF, Fee MS. Growth and splitting of neural sequences in songbird vocal development. Nature. 2015 nov; 528(7582):352–357. http://www.ncbi.nlm.nih.gov/pubmed/26618871http://www.pubmedcentral.nih.gov/articlerender.fcgi?artid=PMC4957523http://www.nature.com/doifinder/10.1038/nature15741, doi: 10.1038/nature15741.

[45] **Pastalkova E**, Itskov V, Amarasingham A, Buzsáki G. Internally Generated Cell Assembly Sequences in the Rat Hippocampus. Science. 2008; 321(5894):1322–1327. http://science.sciencemag.org/content/321/5894/1322, doi: 10.1126/science.1159775.

[46] **Pearson K**. LIII. On lines and planes of closest fit to systems of points in space. The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science. 1901 nov; 2(11):559–572. https://www.tandfonline.com/doi/full/10.1080/14786440109462720, doi: 10.1080/14786440109462720.

[47] **Peter S**, Kirschbaum E, Both M, Campbell L, Harvey B, Heins C, Durstewitz D, Diego F, Hamprecht FA. Sparse convolutional coding for neuronal assembly detection. In: Guyon I, Luxburg UV, Bengio S, Wallach H, Fergus R, Vishwanathan S, Garnett R, editors. Advances in Neural Information Processing Systems 30 Curran Associates, Inc.; 2017.p. 3675–3685. http://papers.nips.cc/paper/6958-sparse-convolutional-coding-for-neuronal-assembly-detection.pdf.

[48] **Picardo M**, Merel J, Katlowitz K, Vallentin D, Okobi D, Benezra S, Clary R, Pnevmatikakis E, Paninski L, Long M. Population-Level Representation of a Temporal Sequence Underlying Song Production in the Zebra Finch. Neuron. 2016 may; 90(4):866–876. http://www.ncbi.nlm.nih.gov/pubmed/27196976http://www.pubmedcentral.nih.gov/articlerender.fcgi?artid=PMC4941616http://linkinghub.elsevier.com/retrieve/pii/S0896627316001094, doi: 10.1016/j.neuron.2016.02.016.

[49] **Quaglio P**, Rostami V, Torre E, Grün S. Methods for identification of spike patterns in massively parallel spike trains. Biological Cybernetics. 2018 apr; 112(1-2):57–80. http://www.ncbi.nlm.nih.gov/pubmed/29651582http://www.pubmedcentral.nih.gov/articlerender.fcgi?artid=PMC5908877http://link.springer.com/10.1007/s00422-018-0755-0, doi: 10.1007/s00422-018-0755-0.

[50] **Ramanarayanan V**, Goldstein L, Narayanan SS. Spatio-temporal articulatory movement primitives during speech production: Extraction, interpretation, and validation. The Journal of the Acoustical Society of America. 2013; 134(2):1378–1394. https://doi.org/10.1121/1.4812765, doi: 10.1121/1.4812765.

[51] **Russo E**, Durstewitz D. Cell assemblies at multiple time scales with arbitrary lag constellations. eLife. 2017; 6:e19428. https://doi.org/10.7554/eLife.19428, doi: 10.7554/eLife.19428.

[52] **Scholvin J**, Kinney JP, Bernstein JG, Moore-Kochlacs C, Kopell N, Fonstad CG, Boyden ES. Close-Packed Silicon Microelectrodes for Scalable Spatially Oversampled Neural Recording. IEEE Transactions on Biomedical Engineering. 2016 jan; 63(1):120–130. doi: 10.1109/TBME.2015.2406113.

[53] **Schrader S**, Grün S, Diesmann M, Gerstein GL. Detecting Synfire Chain Activity Using Massively Parallel Spike Train Recording. Journal of Neurophysiology. 2008 oct; 100(4):2165–2176. http://www.ncbi.nlm.nih.gov/pubmed/18632888http://www.pubmedcentral.nih.gov/articlerender.fcgi?artid=PMC2576207http://www.physiology.org/doi/10.1152/jn.01245.2007, doi: 10.1152/jn.01245.2007.

[54] **Sejnowski TJ**, Churchland PS, Movshon JA. Putting big data to good use in neuroscience. Nature neuroscience. 2014 nov; 17(11):1440–1. http://www.ncbi.nlm.nih.gov/pubmed/25349909http://www.pubmedcentral.nih.gov/articlerender.fcgi?artid=PMC4224030, doi: 10.1038/nn.3839.

[55] **Smaragdis P**. Convolutive Speech Bases and Their Application to Supervised Speech Separation. IEEE Transactions on Audio, Speech, and Language Processing. 2007 jan; 15(1):1–12. doi: 10.1109/TASL.2006.876726.

[56] **Smaragdis P**. In: Puntonet CG, Prieto A, editors. Non-negative Matrix Factor Deconvolution; Extraction of Multiple Sound Sources from Monophonic Inputs Berlin, Heidelberg: Springer Berlin Heidelberg; 2004. p. 494–499. https://doi.org/10.1007/978-3-540-30110-3{_}63, doi: 10.1007/978-3-540-30110-3_63.

[57] **Sutskever I**, Vinyals O, Le QV. Sequence to Sequence Learning with Neural Networks. In: Ghahramani Z, Welling M, Cortes C, Lawrence ND, Weinberger KQ, editors. Advances in Neural Information Processing Systems 27 Curran Associates, Inc.; 2014.p. 3104–3112. http://papers.nips.cc/paper/5346-sequence-to-sequence-learning-with-neural-networks.pdf.

[58] **Theis FJ**, Stadlthanner K, Tanaka T. First results on uniqueness of sparse non-negative matrix factorization. In: 2005 13th European Signal Processing Conference; 2005. p. 1–4.

[59] **Torre E**, Canova C, Denker M, Gerstein G, Helias M, Grün S. ASSET: Analysis of Sequences of Synchronous Events in Massively Parallel Spike Trains. PLOS Computational Biology. 2016 jul; 12(7):e1004939. http://www.ncbi.nlm.nih.gov/pubmed/27420734http://www.pubmedcentral.nih.gov/articlerender.fcgi?artid=PMC4946788http://dx.plos.org/10.1371/journal.pcbi.1004939, doi: 10.1371/journal.pcbi.1004939.

[60] **Udell M**, Horn C, Zadeh R, Boyd S. Generalized Low Rank Models. Foundations and Trends in Machine Learning. 2016; 9(1). http://dx.doi.org/10.1561/2200000055.

[61] **Vaz C**, Toutios A, Narayanan S. Convex Hull Convolutive Non-negative Matrix Factorization for Uncovering Temporal Patterns in Multivariate Time-Series Data. In: Interspeech San Francisco, CA; 2016. p. 963–967.

[62] **Wang D**, Vipperla R, Evans N, Zheng TF. Online Non-Negative Convolutive Pattern Learning for Speech Signals. IEEE Transactions on Signal Processing. 2013 jan; 61(1):44–56. doi: 10.1109/TSP.2012.2222381.

[63] **Williams AH**, Kim TH, Wang F, Vyas S, Ryu SI, Shenoy KV, Schnitzer M, Kolda TG, Ganguli S. Unsupervised discovery of demixed, low-dimensional neural dynamics across multiple timescales through tensor components analysis. Neuron. in press; https://www.biorxiv.org/content/early/2017/10/30/211128, doi: 10.1101/211128.

[64] **Wold S**. Cross-Validatory Estimation of the Number of Components in Factor and Principal Components Models. Technometrics. 1978; 20(4):397–405. https://www.tandfonline.com/doi/abs/10.1080/00401706.1978.10489693, doi: 10.1080/00401706.1978.10489693.

[65] **Zhang Z**, Xu Y, Yang J, Li X, Zhang D. A survey of sparse representation: algorithms and applications. ArXiv e-prints. 2016 feb; .

[66] **Zhou P**, Resendez SL, Rodriguez-Romaguera J, Jimenez JC, Neufeld SQ, Giovannucci A, Friedrich J, Pnevmatikakis EA, Stuber GD, Hen R, Kheirbek MA, Sabatini BL, Kass RE, Paninski L. Efficient and accurate extraction of in vivo calcium signals from microendoscopic video data. eLife. 2018 feb; 7:e28728. https://elifesciences.org/articles/28728, doi: 10.7554/eLife.28728.

Figures

Supplemental Figures

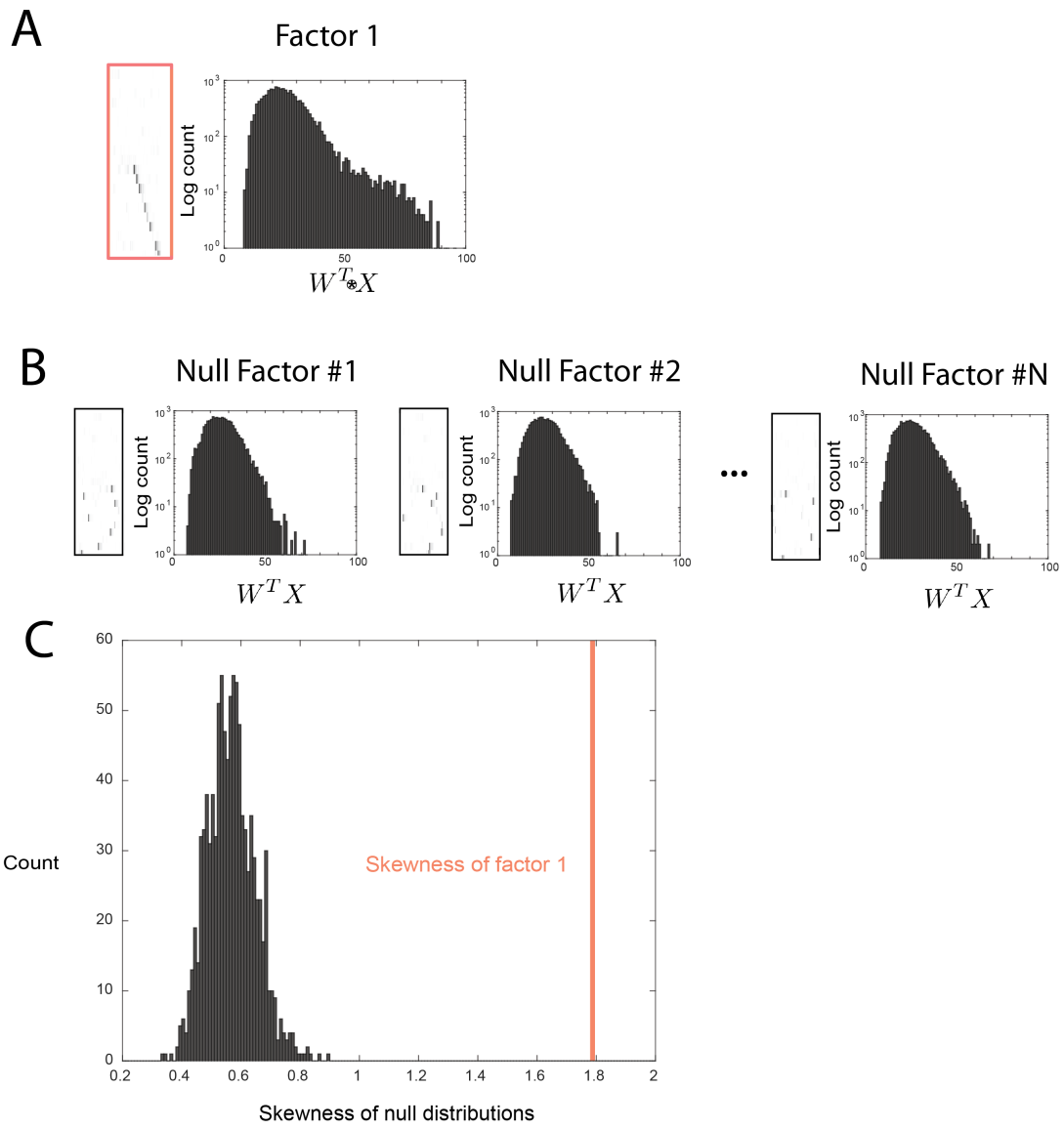

Figure S1. Outline of the procedure used to assess factor significance.
(A) The distribution of overlap values between the real factor and the held-out data. **(B)** In order to test the significance of a factor on held-out data, we constructed null (shifted) versions of the factor, and measured the distribution of overlap values ($\mathbf{W} \overset{T}{\circledast} \mathbf{X}$) between each null factor and the held-out data. **(C)** We then compared the skewness of the actual distribution to the skewness of null distributions, and asked whether it was significantly higher than the null case.

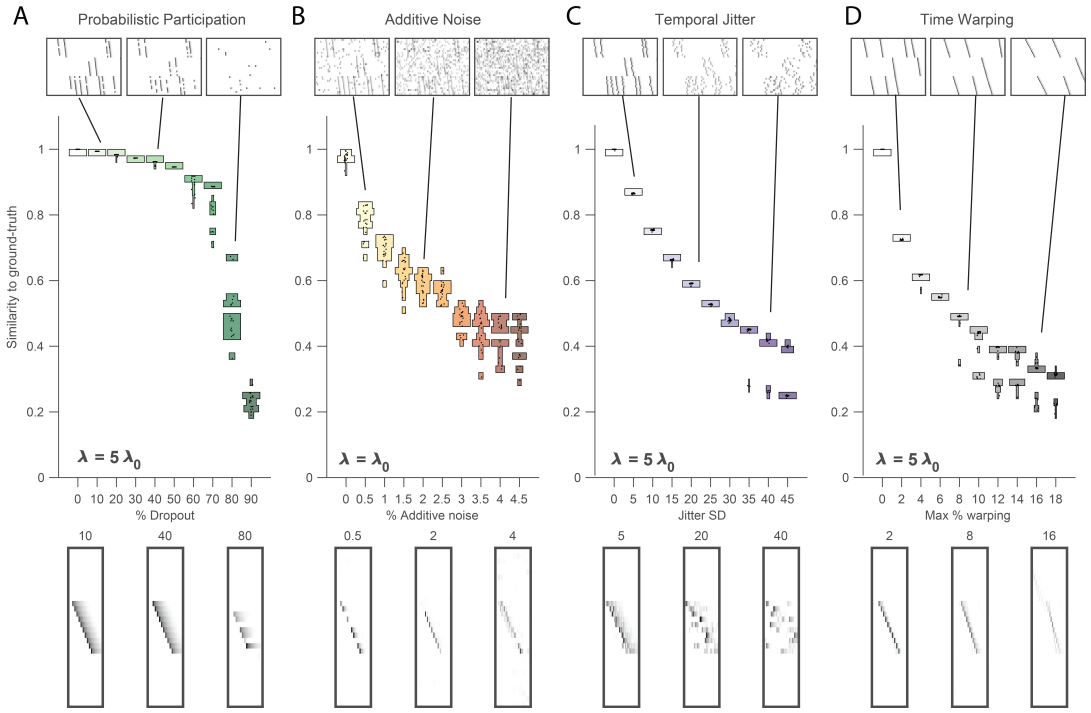

Figure S2. Robustness to noise at different values of λ

Performance of seqNMF was tested under 4 different noise conditions, at different values of λ than in Figure 3 (where $\lambda = 2\lambda_0$): **(A)** probabilistic participation, $\lambda = 5\lambda_0$, **(B)** additive noise, $\lambda = \lambda_0$ **(C)** timing jitter, $\lambda = 5\lambda_0$ and **(D)** sequence warping, $\lambda = 5\lambda_0$. For each noise type, we show: (top) examples of synthetic data at 3 different noise levels; (middle) similarity of seqNMF factors to ground-truth factors across a range of noise levels, showing 20 fits for each noise level; and (bottom) example of one of the $\mathbf{W}$'s extracted at 3 different noise levels (same conditions as data shown above). SeqNMF was run with $K = 20$, $L = 50$.

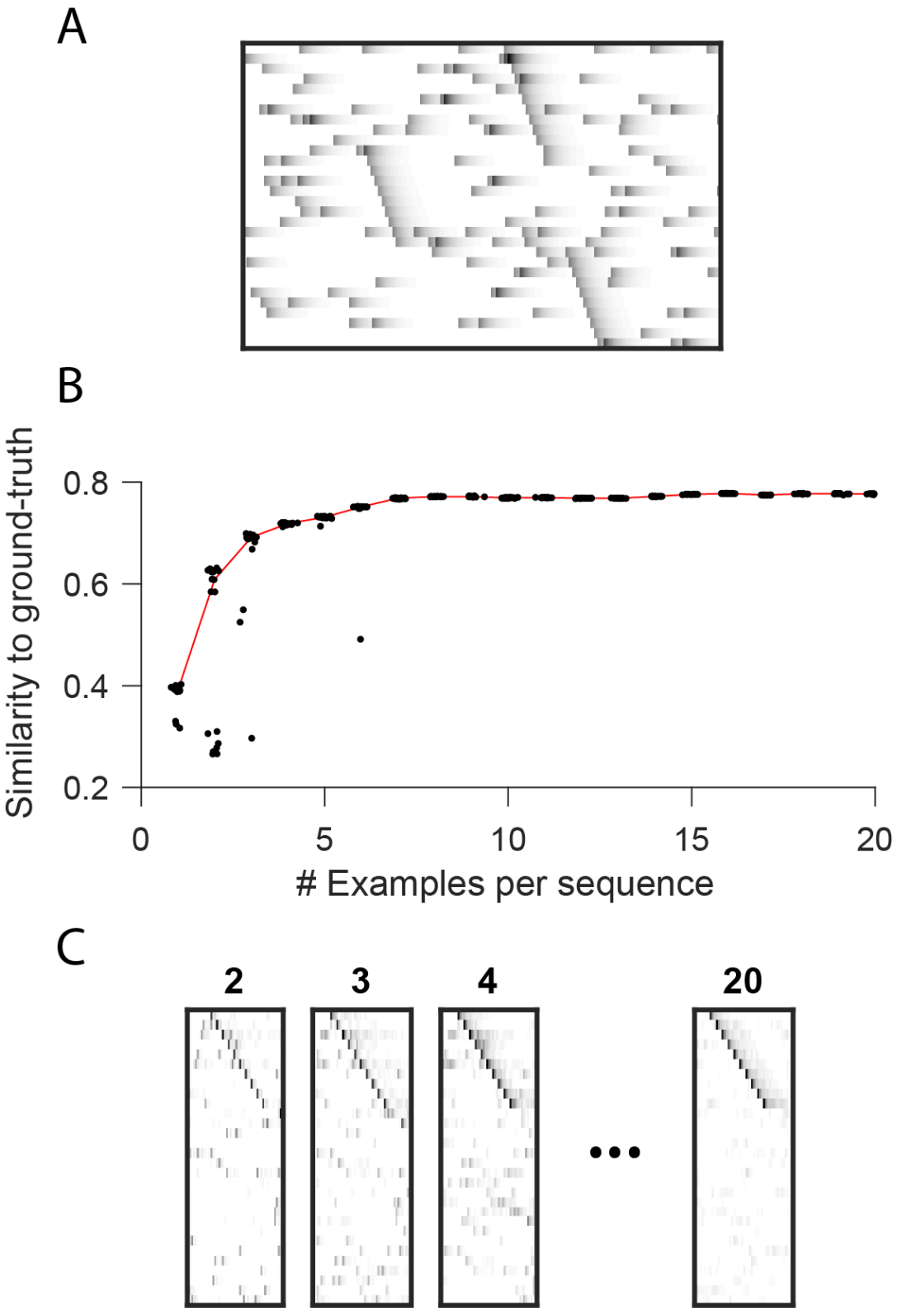

Figure S3. Effect of dataset size on seqNMF reconstruction

(A) A short (400 timestep) dataset containing one example each of three ground-truth sequences, as well as additive noise. **(B)** As a function of dataset size, similarity of extracted factors to noiseless, ground-truth factors. At each dataset size, 20 independent fits of seqNMF are shown. Median shown in red. **(C)** Example factors fit on data containing 2, 3, 4 or 20 examples of each sequence. Extracted factors were significant on held-out data compared to null (shuffled) factors even when training and test datasets each only contained only 2 examples of each sequence.

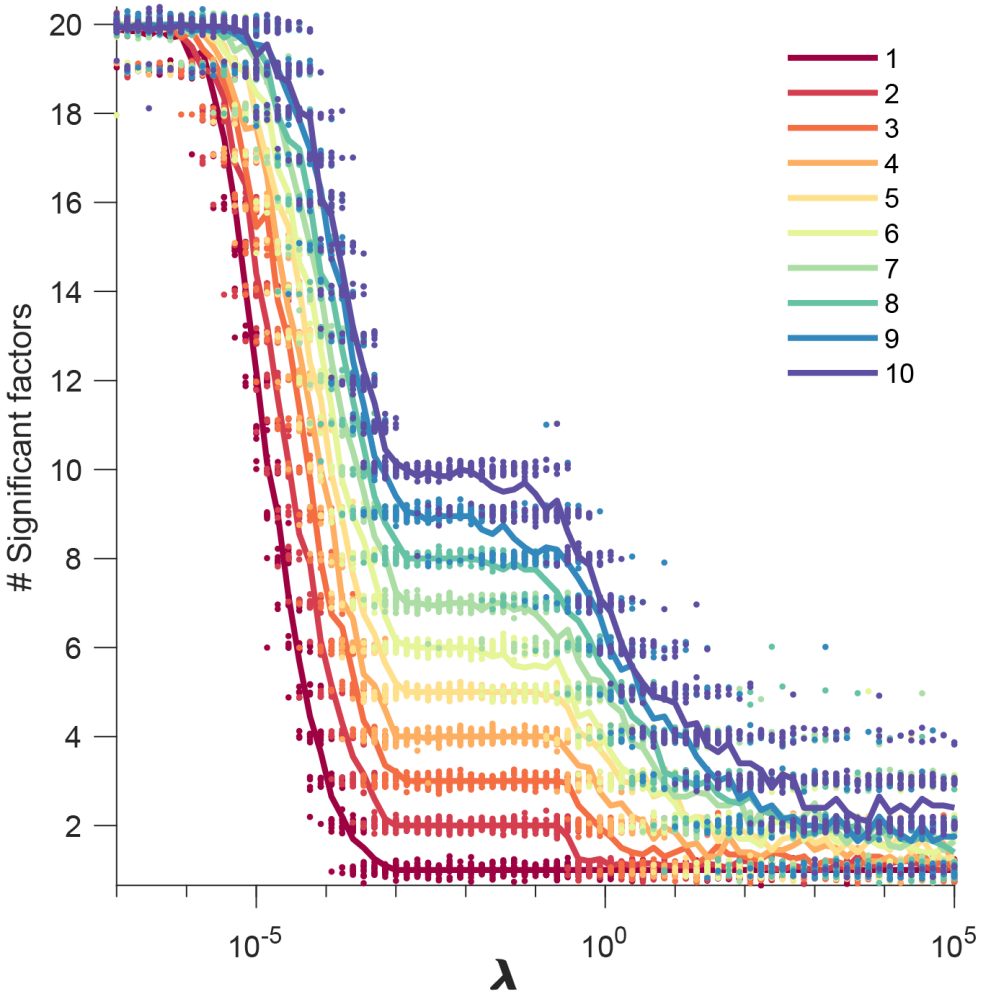

Figure S4. Number of significant factors as a function of λ for datasets containing between 1 and 10 sequences

Number of significant factors were obtained by fitting seqNMF to data containing between 1 and 10 ground truth sequences ($K = 20$, $L = 50$) for a large range of values of λ. For each dataset a λ ranging between 0.001 and 0.1 tended to return the correct number of significant sequences at least 90% of the time.

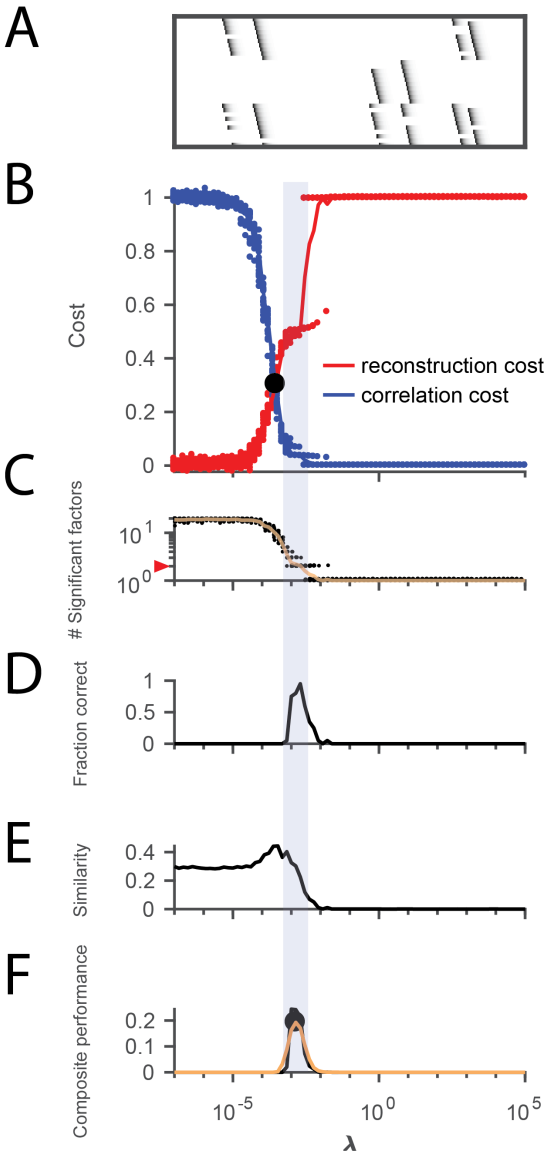

Figure S5. Procedure for choosing λ applied to data with shared neurons
(A) Simulated data containing two patterns which share 50% of their neurons, in the presence of participation noise (70% participation probability). **(B)** Normalized reconstruction cost ($||\widetilde{\mathbf{X}} - \mathbf{X}||_F^2$) and correlation cost ($||(\mathbf{W} \circledast \mathbf{X})\mathbf{S}\mathbf{H}^T||_{1,i\neq j}$) as a function of λ for these data. The cross-over point λ_0 is marked with a black circle. **(C)** The number of significant factors obtained from 20 fits of these data as a function of λ (mean number plotted in orange). The correct number of factors (two) is marked by a red triangle. **(D)** The fraction of fits returning the correct number of significant factors as a function of λ. **(E)** Similarity of the top two factors to ground-truth (noiseless) factors as a function of λ. **(F)** Composite performance measured as the product of the curves shown in (D) and (E), (smoothed curve plotted in orange with a circle marking the peak). Shaded region indicates the range of λ that works well ($\pm$ half height of composite performance). For this data set, the best performance occurs at $\lambda = 5\lambda_0$, while a range of λ between 2 λ_0 and 10 λ_0 performs well.

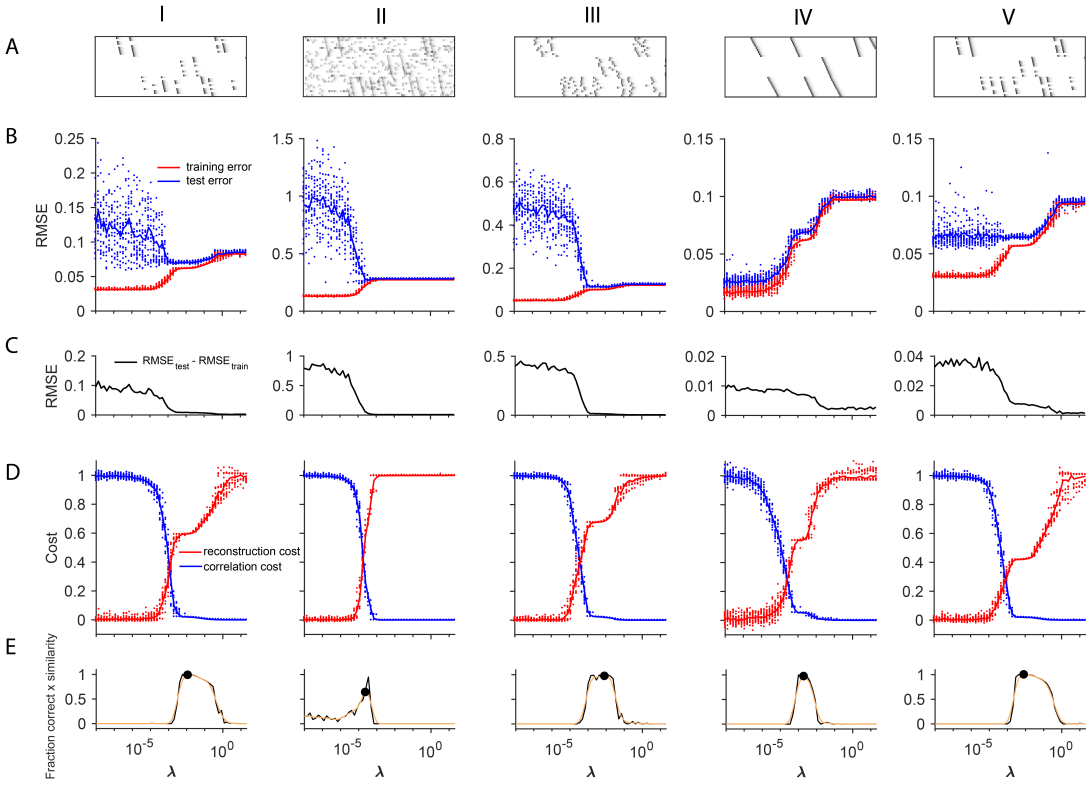

Figure S6. Using cross-validation on held-out (masked) data to choose λ

A method for choosing a reasonable value of λ based on cross validation is shown for five different noise types (each column shows a different noise type; from left to right: **(I)** participation probability, **(II)** additive noise, **(III)** jitter, **(IV)** temporal warping), and **(V)** a lower level of participation noise. For all fits, 10% of the data was held out as the test set. **(A)** Examples of each dataset. **(B)** Test error (blue) and training error (red) as a function of λ for each of the different noise conditions. **(C)** The difference between the test error and training error values shown above. **(D)** Normalized reconstruction cost ($||\widetilde{\mathbf{X}} - \mathbf{X}||_F^2$) and correlation cost ($||(\mathbf{W} \overset{\mathsf{T}}{\circledast} \mathbf{X})\mathbf{S}\mathbf{H}^\mathsf{T}||_{1,i\neq j}$) as a function of λ for each of the different noise conditions. **(E)** Composite performance as a function of λ. Panels D and E are identical to those in Figure 4, and are included here for comparison. **(V)** These data have a lower amount of participation noise than (I). Note that in low-noise conditions, test error may not exhibit a minima within the optimal range of λ.

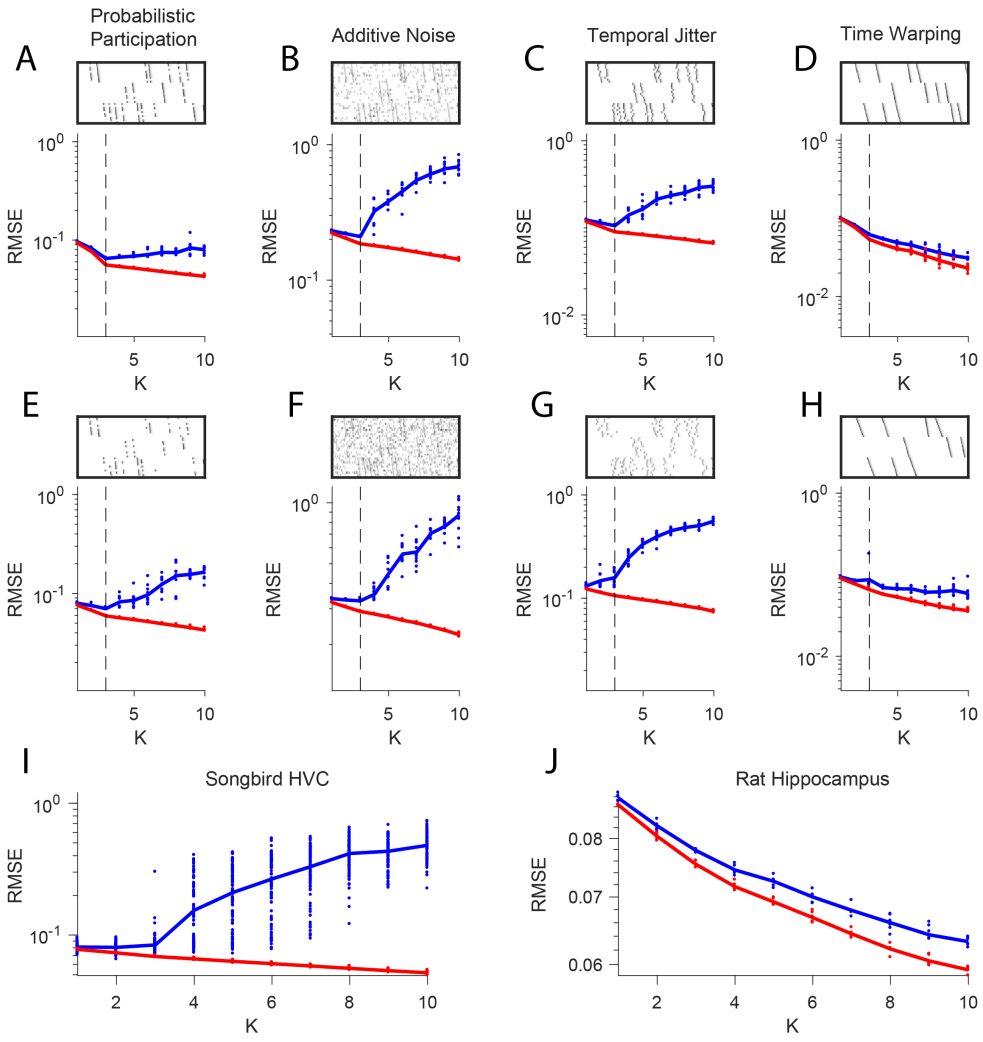

Figure S7. Estimating the number of sequences in a dataset using cross-validation on randomly masked held-out datapoints

(A) Reconstruction error (RMSE) for test data (red) and training data (blue) plotted as a function of the number of components (K) used in convNMF. Twenty independent convNMF fits are shown for each value of K. This panel shows results for 1% participation noise. For synthetic data fit fits, 10% of the data was held out as the test set. For neural data 5% of the data was held out. Other noise conditions are shown as follows: **(B)** jitter noise (10 timestep SD); **(C)** warping (13%); **(D)** higher additive noise (2.5%); **(E)** higher jitter noise (25 timestep SD); **(F)** higher warping (33%) **(G)** Reconstruction error vs. K for neuronal data collected from premotor cortex (area HVC) of a singing bird (Figure 7) and **(H)** hippocampus of a rat performing a left-right alternation task (Figure 6).

Figure S8. Biasing factorizations between sparsity in $\mathbf{W}$ or $\mathbf{H}$

Two different factorizations of the same simulated data, where a sequence is always repeated precisely three times. Both yield perfect reconstructions, and no cross-factor correlations. The factorizations differ in the amount of features placed in $\mathbf{W}$ versus $\mathbf{H}$. Both use $K = 3$ and $\lambda = 0.001$. **(A)** Factorization achieved using a sparsity penalty on $\mathbf{H}$, with $\lambda_{L1\mathbf{H}} = 1$. **(B)** Factorization achieved using a sparsity penalty on $\mathbf{W}$, with $\lambda_{L1\mathbf{W}} = 1$.

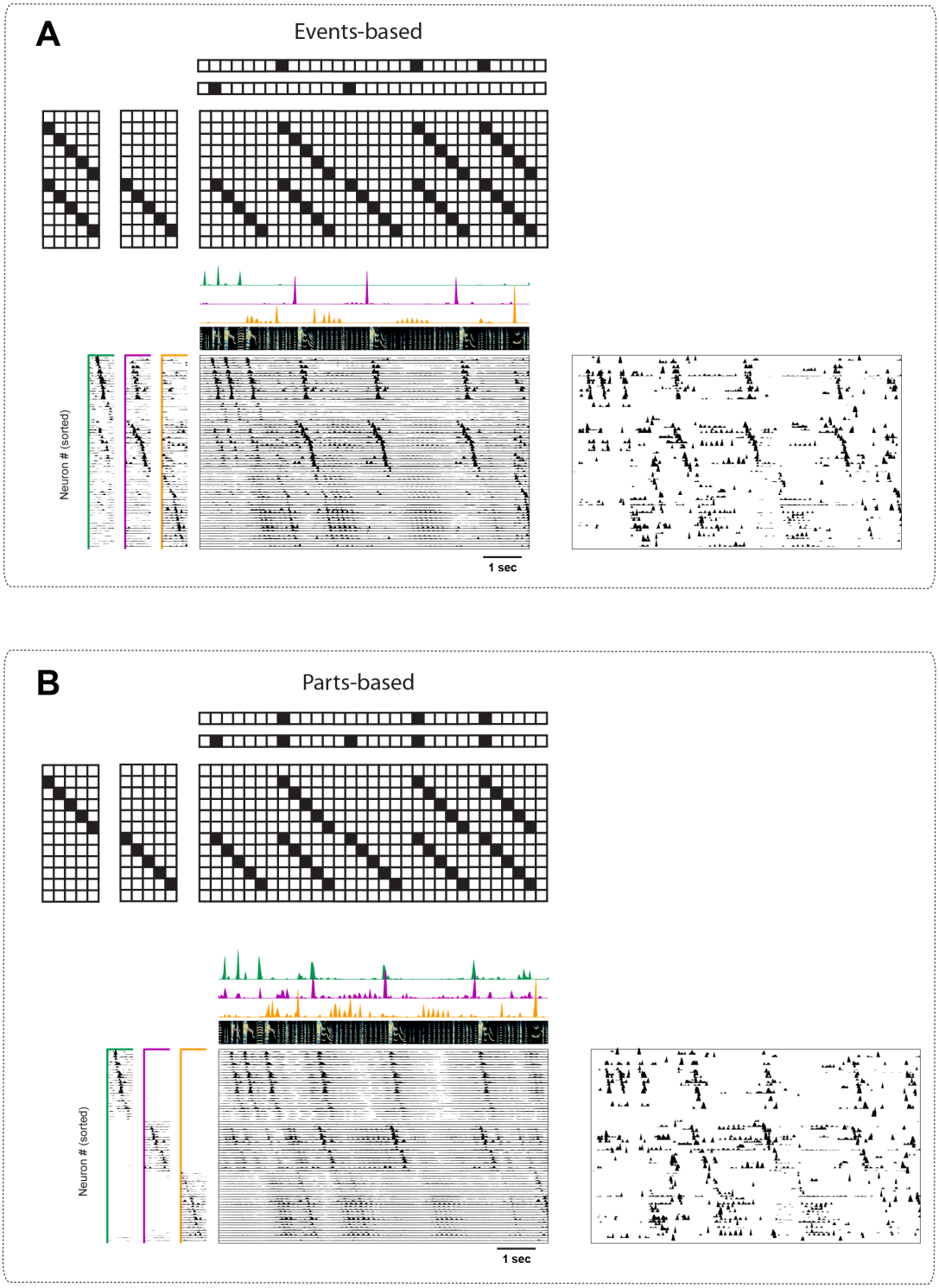

Figure S9. Events-based and parts-based factorizations of songbird data
Illustration of a trade-off between parts-based ($\mathbf{W}$ is more strictly orthogonal) and events-based ($\mathbf{H}$ is more strictly orthogonal) factorizations in a dataset where some neurons are shared between different sequences. The same data as in Figure 7 is factorized using smoothed soft orthogonality on $\mathbf{H}$ (**A**, events-based), or on $\mathbf{W}$ (**B**, parts-based). Below each motivating cartoon factorization, we show seqNMF fits ($\mathbf{W}$ and $\mathbf{H}$ together with the reconstruction) of the data in Figure 7. The right panels contain the raw data sorted according to these factorizations. Favoring events-based or parts-based factorizations is a matter of preference. Parts-based factorizations are particularly useful for separating neurons into ensembles. Events-based factorizations are particularly useful for identifying what neural events occur when.

Appendix 1

Deriving multiplicative update rules

Standard gradient descent methods for minimizing a cost function must be adapted when solutions are constrained to be non-negative, since gradient descent steps may result in negative values. Lee and Seung invented an elegant and widely-used algorithm for non-negative gradient descent that avoids negative values by performing multiplicative updates [34]. They derive these multiplicative updates by choosing an adaptive learning rate that makes additive terms cancel from standard gradient descent on the cost function. We will reproduce their derivation here, and detail how to extend it to the convolutional case [56] and apply several forms of regularization [43, 50, 10]. See Table 3 for a compilation of cost functions, derivatives and multiplicative updates for NMF and convNMF under several different regularization conditions.

Standard NMF

NMF performs the factorization $\mathbf{X} \approx \widetilde{\mathbf{X}} = \mathbf{WH}$. NMF factorizations seek to solve the following problem:

$$(\widetilde{\mathbf{W}}, \widetilde{\mathbf{H}}) = arg \min_{\mathbf{W},\mathbf{H}} \mathscr{L}(\mathbf{W}, \mathbf{H}) \tag{16}$$

$$\mathscr{L}(\mathbf{W}, \mathbf{H}) = \frac{1}{2}||\widetilde{\mathbf{X}} - \mathbf{X}||_F^2 \tag{17}$$

$$\widetilde{\mathbf{W}}, \widetilde{\mathbf{H}} \geq 0 \tag{18}$$

This problem is convex in $\mathbf{W}$ and $\mathbf{H}$ separately, not together, so a local minimum is found by alternating $\mathbf{W}$ and $\mathbf{H}$ updates. Note that:

$$\frac{d}{d\mathbf{W}}\mathscr{L}(\mathbf{W}, \mathbf{H}) = \widetilde{\mathbf{X}}\mathbf{H}^\top - \mathbf{X}\mathbf{H}^\top \tag{19}$$

$$\frac{d}{d\mathbf{H}}\mathscr{L}(\mathbf{W}, \mathbf{H}) = \mathbf{W}^\top\widetilde{\mathbf{X}} - \mathbf{W}^\top\mathbf{X} \tag{20}$$

Thus, gradient descent steps for $\mathbf{W}$ and $\mathbf{H}$ are:

$$\mathbf{W} \leftarrow \mathbf{W} - \eta_{\mathbf{W}}(\widetilde{\mathbf{X}}\mathbf{H}^\top - \mathbf{X}\mathbf{H}^\top) \tag{21}$$

$$\mathbf{H} \leftarrow \mathbf{H} - \eta_{\mathbf{H}}(\mathbf{W}^\top\widetilde{\mathbf{X}} - \mathbf{W}^\top\mathbf{X}) \tag{22}$$

To arrive at multiplicative updates, Lee and Seung [34] set:

$$\eta_{\mathbf{W}} = \frac{\mathbf{W}}{\mathbf{W}\mathbf{H}\mathbf{H}^\top} \tag{23}$$

$$\eta_{\mathbf{H}} = \frac{\mathbf{H}}{\mathbf{W}^\top\mathbf{W}\mathbf{H}} \tag{24}$$

Thus, the gradient descent updates become multiplicative:

$$\mathbf{W} \leftarrow \mathbf{W} \times \frac{\mathbf{X}\mathbf{H}^\top}{\mathbf{W}\mathbf{H}\mathbf{H}^\top} = \mathbf{W} \times \frac{\mathbf{X}\mathbf{H}^\top}{\widetilde{\mathbf{X}}\mathbf{H}^\top} \tag{25}$$

$$\mathbf{H} \leftarrow \mathbf{H} \times \frac{\mathbf{W}^\top\mathbf{X}}{\mathbf{W}^\top\mathbf{W}\mathbf{H}} = \mathbf{H} \times \frac{\mathbf{W}^\top\mathbf{X}}{\mathbf{W}^\top\widetilde{\mathbf{X}}} \tag{26}$$

where the division and $\times$ are element-wise.

Standard convNMF

Convolutional NMF factorizes data $\mathbf{X} \approx \widetilde{\mathbf{X}} = \sum_\ell \mathbf{W}_{\cdot\cdot\ell} \overrightarrow{\mathbf{H}}^{\ell} = \mathbf{W} \circledast \mathbf{H}$. convNMF factorizations seek to solve the following problem:

$$(\widetilde{\mathbf{W}}, \widetilde{\mathbf{H}}) = arg\min_{\mathbf{W},\mathbf{H}} \mathscr{L}(\mathbf{W}, \mathbf{H}) \tag{27}$$

$$\mathscr{L}(\mathbf{W}, \mathbf{H}) = \frac{1}{2}||\widetilde{\mathbf{X}} - \mathbf{X}||_F^2 \tag{28}$$

$$\widetilde{\mathbf{W}}, \widetilde{\mathbf{H}} \geq 0 \tag{29}$$

The derivation above for standard NMF can be applied for each ℓ, yielding the following update rules for convNMF [56]:

$$\mathbf{W}_{\cdot\cdot\ell} \leftarrow \mathbf{W}_{\cdot\cdot\ell} \times \frac{\mathbf{X}\overrightarrow{\mathbf{H}}^{\ell\top}}{\widetilde{\mathbf{X}}\overrightarrow{\mathbf{H}}^{\ell\top}} \tag{30}$$

$$\mathbf{H} \leftarrow \mathbf{H} \times \frac{\sum_\ell \mathbf{W}_{\cdot\cdot\ell}^\top \overleftarrow{\mathbf{X}}^{\ell}}{\sum_\ell \mathbf{W}_{\cdot\cdot\ell}^\top \overleftarrow{\widetilde{\mathbf{X}}}^{\ell}} = \mathbf{H} \times \frac{\mathbf{W} \circledast^\top \mathbf{X}}{\mathbf{W} \circledast^\top \widetilde{\mathbf{X}}} \tag{31}$$

Where the operator $\ell \rightarrow$ shifts a matrix in the $\rightarrow$ direction by ℓ timebins, i.e. a delay by ℓ timebins, and $\leftarrow \ell$ shifts a matrix in the $\leftarrow$ direction by ℓ timebins (Table 1). Note that NMF is a special case of convNMF where $L = 1$.

Incorporating regularization terms

Suppose we want to regularize by adding a new term, $\mathscr{R}$ to the cost function:

$$(\widetilde{\mathbf{W}}, \widetilde{\mathbf{H}}) = arg\min_{\mathbf{W},\mathbf{H}} \mathscr{L}(\mathbf{W}, \mathbf{H}) \tag{32}$$

$$\mathscr{L}(\mathbf{W}, \mathbf{H}) = \frac{1}{2}||\widetilde{\mathbf{X}} - \mathbf{X}||_F^2 + \mathscr{R} \tag{33}$$

$$\widetilde{\mathbf{W}}, \widetilde{\mathbf{H}} \geq 0 \tag{34}$$

Using a similar trick to Lee and Seung, we choose a $\eta_{\mathbf{W}}, \eta_{\mathbf{H}}$ to arrive at a simple multiplicative update. Below is the standard NMF case, which generalizes trivially to the convNMF case.

Note that:

$$\frac{d\mathscr{L}}{d\mathbf{W}} = \widetilde{\mathbf{X}}\mathbf{H}^\top - \mathbf{X}\mathbf{H}^\top + \frac{d\mathscr{R}}{d\mathbf{W}} \tag{35}$$

$$\frac{d\mathscr{L}}{d\mathbf{H}} = \mathbf{W}^\top\widetilde{\mathbf{X}} - \mathbf{W}^\top\mathbf{X} + \frac{d\mathscr{R}}{d\mathbf{H}} \tag{36}$$

We set:

$$\eta_{\mathbf{W}} = \frac{\mathbf{W}}{\widetilde{\mathbf{X}}\mathbf{H}^\top + \frac{d\mathscr{R}}{d\mathbf{W}}} \tag{37}$$

$$\eta_{\mathbf{H}} = \frac{\mathbf{H}}{\mathbf{W}^\top\widetilde{\mathbf{X}} + \frac{d\mathscr{R}}{d\mathbf{H}}} \tag{38}$$

Thus, the gradient descent updates become multiplicative:

$$\mathbf{W} \leftarrow \mathbf{W} - \eta_{\mathbf{W}}\frac{d\mathscr{L}}{d\mathbf{W}} = \mathbf{W} \times \frac{\mathbf{X}\mathbf{H}^\top}{\widetilde{\mathbf{X}}\mathbf{H}^\top + \frac{d\mathscr{R}}{d\mathbf{W}}} \tag{39}$$

$$\mathbf{H} \leftarrow \mathbf{H} - \eta_{\mathbf{H}}\frac{d\mathscr{L}}{d\mathbf{H}} = \mathbf{H} \times \frac{\mathbf{W}^\top\mathbf{X}}{\mathbf{W}^\top\widetilde{\mathbf{X}} + \frac{d\mathscr{R}}{d\mathbf{H}}} \tag{40}$$

where the division and $\times$ are element-wise.

This framework enables flexible incorporation of different types of regularization or penalty terms into the multiplicative NMF update algorithm. This framework also extends naturally to the convolutional case. See Table 3 for examples of several regularization terms, including $L1$ sparsity [43, 50] and soft orthogonality [10], as well as the terms we introduce here to combat the types of inefficiencies and cross correlations we identified in convolutional NMF, namely, smoothed orthogonality for $\mathbf{H}$ and $\mathbf{W}$, and smoothed cross-factor orthogonality, the primary seqNMF regularization term. For the seqNMF regularization term, $\lambda||(\mathbf{W} \overset{\top}{\circledast} \mathbf{X})\mathbf{S}\mathbf{H}^\top||_{1,i\neq j}$, the multiplicative update rules are:

$$\mathbf{W}_{..\ell} \leftarrow \mathbf{W}_{..\ell} \times \frac{\mathbf{X}\left(\overset{\ell\rightarrow}{\mathbf{H}}\right)^\top}{\widetilde{\mathbf{X}}\left(\overset{\ell\rightarrow}{\mathbf{H}}\right)^\top + \lambda\,\overset{\leftarrow\ell}{\mathbf{X}}\mathbf{S}\mathbf{H}^\top(\mathbf{1}-\mathbf{I})} \tag{41}$$

$$\mathbf{H} \leftarrow \mathbf{H} \times \frac{\mathbf{W} \overset{\top}{\circledast} \mathbf{X}}{\mathbf{W} \overset{\top}{\circledast} \widetilde{\mathbf{X}} + \lambda(\mathbf{1}-\mathbf{I})(\mathbf{W} \overset{\top}{\circledast} \mathbf{X}\mathbf{S})} \tag{42}$$

Where the division and $\times$ are element-wise. Note that multiplication with the $K \times K$ matrix $(\mathbf{1}-\mathbf{I})$ effectively implements factor competition because it places in the kth row a sum across all other factors.