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 Neuroscience Institute; [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. Dynamics that include repeated temporal patterns (which we call sequences), are not succinctly captured by traditional dimensionality reduction techniques such as principal components analysis (PCA) and non-negative matrix factorization (NMF). The presence of neural sequences is commonly demonstrated using visual display of trial-averaged firing rates [15, 32, 19]. However, the field suffers from a lack of task-independent, unsupervised tools for consistently identifying sequences directly from neural data, and cross-validating these sequences on held-out data. We propose a tool that extends a convolutional NMF technique to prevent its common failure modes. Our method, which we call seqNMF, provides a framework for extracting sequences from a dataset, and is easily cross-validated to assess the significance of each extracted factor. We apply seqNMF to recover sequences in both a previously published dataset from rat hippocampus, as well as a new dataset from the songbird pre-motor area, HVC. In the hippocampal data, our algorithm automatically identifies neural sequences that match those calculated manually by reference to behavioral events [15, 32]. The second data set was recorded in birds that never heard a tutor, and therefore sang pathologically variable songs. Despite this variable behavior, seqNMF is able to discover stereotyped neural sequences. These sequences are deployed in an overlapping and disorganized manner, strikingly different from what is seen in tutored birds. Thus, by identifying temporal structure directly from neural data, seqNMF can enable dissection of complex neural circuits with noisy or changing behavioral readouts.

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 [10, 23, 3, 21], as well as artificial intelligence systems [11, 42]. 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 recording of hundreds or thousands of neurons [6, 26, 38, 24], in which neuronal dynamics are often structured in sparse sequences [18, 19, 31, 32].

While sequential patterns are simple to conceptualize, identifying these patterns in high-dimensional datasets is surprisingly challenging. Traditional techniques for identifying low dimensional structure in high dimensional datasets such as PCA and NMF do not work for sequences, because those methods only model zero-time-lag correlations in data. It is sometimes possible to identify neural sequences by heuristically aggregating pairwise cross-correlations across neurons or across timebins [37, 17], but these correlations are easily confounded [4], leading to mathematically complex and computationally expensive procedures. In some cases, sequences can be identified by simply averaging across multiple behavioral trials, but this approach requires stereotyped behavior.

Of increasing interest is the study of internal dynamics in the brain, without reference to behavior, for example, neural dynamics during learning, sleep, or diseased states. A promising approach for the unsupervised detection of temporal patterns is convolutive matrix factorization (CNMF) [41, 40] (Figure 1), which has primarily been applied to audio signals such as speech [30, 40, 45]. CNMF identifies exemplar patterns in conjunction with the times at which each pattern occurs. This strategy eliminates the need to average activity aligned to any external behavioral variables, and CNMF has recently been used to extract repeated patterns in spontaneous neural activity [34]. While CNMF factorizations produce an excellent reconstruction of the data, this algorithm will find a much larger number of factors than minimally required. Because of this redundancy, there are many different possible factorizations that explain the data equally well, and the algorithm arbitrarily chooses among them each time it is run, producing inconsistent results [34].

When describing and interpreting data, the principle of 'Occam's razor', a key scientific doctrine, tells us to prefer minimal models. In this paper, we describe a modification of the CNMF algorithm that penalizes redundant factors, biasing the results toward factorizations with the smallest number of factors and providing a simple explanation of the data. We do this by incorporating a regularization term into the CNMF cost function. Unlike other common approaches [20] such as sparsity regularization [47, 30, 36] 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 [7] 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 under a variety of noise conditions, with high similarity to ground-truth sequences. We further tested seqNMF on hippocampal spiking data in 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 [44] with applications to neuroscience [12], including principal component analysis (PCA) [33], non-negative matrix factorization (NMF) [27], 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 at which that pattern occurs (temporal loadings). In this case, the data matrix $\mathbf{X}$ is mathematically reconstructed as the outer product of these two vectors ($\widetilde{\mathbf{X}}_{nt} = w_n h_t$). 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:

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

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:

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

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 vector of temporal loadings ($\mathbf{H}$) that approximate the data. This corresponds to a dimensionality reduction, whereby the data is expressed in more compact form ($K < N, T$). NMF additionally requires that $\mathbf{W}$ and $\mathbf{H}$ must contain only positive numbers. The discovery of unknown factors is often accomplished by minimizing the following cost function, which measures (using the Frobenius norm) the sum of all squared errors between the reconstruction $\widetilde{\mathbf{X}} = \mathbf{WH}$ and the original data matrix $\mathbf{X}$:

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

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 (CNMF)

Convolutional NMF (CNMF) [41, 40] 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), CNMF 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 $\mathbf{w_1}$ for 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 can be compiled together into an $N \times K \times L$ tensor $\mathbf{W}$ and a corresponding $K \times T$ timeseries matrix $\mathbf{H}$. Analogously to NMF, CNMF reconstructs the data as a sum of K convolutions between each neural activity pattern ($\mathbf{W}$), and its corresponding temporal loadings ($\mathbf{H}$):

$$\mathbf{X}_{nt} \approx \widetilde{\mathbf{X}}_{nt} = \sum_k \sum_\ell w_{nk\ell} h_{k(t-\ell)} = (\mathbf{W} \circledast \mathbf{H})_{nt} \tag{4}$$

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 the same multiplicative gradient descent updates often used for standard NMF [27, 41, 40].

While CNMF can perform extremely well at reconstructing sequential structure, it suffers from a significant problem—namely, it reconstructs data using many more factors than are minimally required. This is because an individual temporal pattern may be approximated equally well by a single pattern or by a linear combination of multiple sub-patterns. A related problem is that running the CNMF algorithm from different random initial conditions produces inconsistent results, finding different combinations of sub-patterns on each run [34]. 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 failure modes manifest as strong correlations between different redundant factors, as seen in the similarity of their temporal loadings ($\mathbf{H}$) and of their exemplar activity patterns ($\mathbf{W}$).

SeqNMF: A regularized 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 or simplifying solutions [20]. To reduce the occurrence of redundant factors (and inconsistent factorizations) in CNMF, we sought a principled way of penalizing the correlations between

factors by introducing a regularization term into the CNMF cost function of the following form:

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

In the next section, we will motivate a novel cost function that effectively minimizes the number of factors by penalizing 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 failure mode.

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 [7]. Such correlations are measured by computing the correlation matrix $\mathbf{H}\mathbf{H}^\mathsf{T}$, which contains the correlations between the temporal loadings of every pair of factors. The regularization may be implemented using the cost term $\mathscr{R} = \lambda||\mathbf{H}\mathbf{H}^\mathsf{T}||_{1,i\neq j}$. The norm $||\cdot||_{1,i\neq j}$ sums the absolute value of every matrix entry except the diagonal (notation summary, Table 1) so that correlations between different factors are penalized, while the obvious correlation of each factor with itself is not. Thus, during the minimization process, similar factors compete, and a larger factor drives down the $\mathbf{H}$ of a correlated smaller factor. The parameter λ is controls the magnitude of the regularization term $\mathscr{R}$.

In CNMF, a regularization term based on $\mathbf{H}\mathbf{H}^\mathsf{T}$ 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 loadingss that have a small temporal offset and thus are not detected by $\mathbf{H}\mathbf{H}^\mathsf{T}$. To address this problem, we smoothed the $\mathbf{H}$'s in the regularization term with a square window of length $2L-1$ using the smoothing matrix $\mathbf{S}$ ($s_{ij} = 1$ when $|i-j| < L$ and otherwise $s_{ij} = 0$). The resulting regularization, $\mathscr{R} = \lambda||\mathbf{H}\mathbf{S}\mathbf{H}^\mathsf{T}||$, allows factors with small temporal offsets to compete, effectively preventing errors of Type 1 and 2.

Unfortunately this regularization 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 $||\mathbf{H}\mathbf{S}\mathbf{H}^\mathsf{T}||$ 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 CNMF framework, there is no constraint on temporal translations of the sequence within $\mathbf{W}$. For example, if two redundant factors containing identical sequences that are simply offset by one timebin (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 2), analogous to $||\mathbf{H}\mathbf{S}\mathbf{H}^\mathsf{T}||$. 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 cost term. We note that redundant factors of this type 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 compute the pairwise correlation between the temporal loading of each factor and the overlap of every other factor with the data, given by $\mathbf{W} \overset{\mathsf{T}}{\circledast} \mathbf{X}_{i\neq j}$ (notation

summary, Table 1). The regularization then sums up these correlations across all pairs of factors, implemented as follows:

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

When incorporated into the update rules, this causes any factor that has a high overlap with the data to suppress the temporal loading ($\mathbf{H}$) 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:

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

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 CNMF [27, 41, 40] (Table 2).

Testing the performance of seqNMF on simulated sequences

To compare the performance of seqNMF to unregularized CNMF, 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: consistency of factorizations, the ability of the algorithm to discover the correct number of sequences in the data, and robustness to noise.

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 1, CNMF yielded low consistency scores typically ranging from 0.2 to 0.4 on a scale from zero to one. 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 CNMF, and typically (>80% of the time) higher than 0.99 (Figure 2B). Both CNMF 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, CNMF 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 interesting case in which a population of neurons is active in multiple different sequences. In fact neurons that are shared across different sequences have been observed in several different neuronal datasets [31, 32, 19]. 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 2).

Cross-validating to assess the statistical significance of sequences

SeqNMF allows a simple procedure for assessing the statistical significance of each extracted sequence. Candidate sequences are extracted by applying SeqNMF to a subset of the data; the significance of each candidate sequence is then assessed on separate held-out data. If an extracted sequence corresponds to a real sequence present in the data, then the overlap of that factor with the held-out data ($\mathbf{W}^{\top} \circledast \mathbf{X}$) will have large values at the times at which the sequence occurs (relative to other times). The resulting abundance of high overlap values will create a distribution of overlaps with high skewness compared to a null distribution. In contrast, a candidate sequence that does not reliably occur in the held-out data will have a smaller number of high overlaps, and a distribution of overlaps with lower skewness. We compare the skewness of the actual distribution of overlaps with that of distributions generated from null factors to determine the significance of each candidate sequence (Figure S1, Methods 10). Null factors were created by random circular shifts in time lag, along the L dimension, of the pattern matrices $\mathbf{W}$.

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.

Estimating the number of sequences in a dataset

A successful factorization should contain the same number of significant factors as exist sequences in the data. To compare the ability of seqNMF and CNMF to recover the true number of patterns in a dataset, we generated simulated data containing between 1 and 10 different sequences. We then ran many independent fits of these data, using both seqNMF and CNMF, and measured the number of significant factors. We found that CNMF 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 significant factors (N_{sig}) that closely matches the actual number of sequences (N_{seq}). The standard deviation of the error ($N_{seq} - N_{sig}$) tended to grow linearly with the actual number of sequences (Figure 2C).

Robustness to noisy and challenging data

Having established that seqNMF can produce both consistent and efficient factorizations of noiseless synthetic data, we next probed the capacity of seqNMF to detect sequences in the presence of common types of noise. These included: 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 two neural sequences at 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 the similarity of the factors to the ground-truth. We defined the ground-truth sequences those used to generate the synthetic data prior to the addition of noise. We then quantified the correlation 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 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.

Method for choosing an appropriate value of λ

In general, the seqNMF algorithm performs differently using different values of λ, and application to the noisy datasets revealed that the optimal choice of this parameter may depend on the degree and type of noise contamination. Choosing λ involves a trade off between reconstruction accuracy and the efficiency and consistency of the resulting factorizations (Figure 4). Indeed, perfect reconstruction is no longer a goal in noisy data, since it would imply fitting all of the noise as well as the signal. Rather, 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. For any given factorization, the reconstruction error may be estimated as $||\widetilde{\mathbf{X}} - \mathbf{X}||_F^2$, and the efficiency may be estimated using the seqNMF regularization term ($||\mathbf{W} \overset{\top}{\circledast} \mathbf{XSH}^\top||_{1,i\neq j}$) which we refer to as correlation cost.

We have developed a quantitative strategy to guide the choice of λ, by analyzing the dependence on λ of both reconstruction error and correlation cost in synthetic datasets containing two sequences (Figure 4). 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 CNMF, 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 terminal reconstruction cost and correlation cost

are recorded. These 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 4). 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 λ_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 other noise types, performance was better at higher ($\approx 5\lambda_0$). Note that this procedure 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.

Sometimes there is not a clear correct answer for how many sequences exist in a dataset. In fact, different values of λ can lead to different sensible factorizations. It can be useful to explore the factorization for different values of λ between λ_0 and $10\lambda_0$. We observed a notable example of this in datasets that included sequences with a high degree of temporal warping. In this case, high λ led seqNMF to extract a single factor for each ground truth sequence. In contrast, at low λ seqNMF extracted multiple factors for each ground truth sequence, corresponding to slow and fast variations of the sequence. Thus, seqNMF clusters sequences with different granularity depending on the strength of the regularization term λ.

Adding additional sparsity regularization to seqNMF

Sparsity regularization is a widely used strategy for achieving more interpretable results across a variety of algorithms and datasets [47], including CNMF [30, 36]. In some of our datasets, we found it useful to add $L1$ regularization for sparsity, in addition to regularizing for factor competition. The multiplicative update rules for these variants are included in Table 2, and as part of our code package. Sparsity on the matrices $\mathbf{W}$ and $\mathbf{H}$ may particularly useful in cases when sequences are repeated rhythmically (Figure S2). 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 $\mathbf{H}$. Of course, these are both equally valid interpretations of the data, but each may be more useful in different contexts.

Further considerations of shared neurons

The existence of neurons that are shared between different sequences raises an interesting ambiguity in the types of factorizations that seqNMF can produce, an example of which is illustrated in Figure S3. In this case, there are two different, but equally valid, factorizations: in one factorization, there are two types of events, one in which a population of neurons generates a sequence by itself, and another in which a second population of neurons is also simultaneously active. In another factorization, these same data are interpreted by seqNMF as two different populations of neurons that are sometimes active separately and sometimes active together. Note that these two factorizations produce very different correlations between the factors. In the first, 'events-based' factorization, the $\mathbf{H}$s are orthogonal (uncorrelated) while the $\mathbf{W}$s have high overlap. In the second, 'parts-based' factorization, the $\mathbf{W}$s are orthogonal while the $\mathbf{H}$s are strongly correlated.

We have found that seqNMF will produce both types of factorizations depending on initial conditions and the structure of shared neurons in the data. We note that these different factorizations may correspond to different intuitions about underlying mechanisms. Therefore, it may be useful to explicitly bias the probability of these different factorizations by the addition of further regularization on either $\mathbf{W}$ or $\mathbf{H}$ correlations, as demonstrated in Figure S3. Update rules to implement both of these regularizations are derived in Appendix 1, and shown in Table 2, and included as options in our code.

Application of seqNMF to hippocampal sequences

To test the ability of seqNMF to discover patterns in electrophysiological data, we analyzed the activity of a set of simultaneously recorded hippocampal neurons in a publicly available dataset in which sequences have previously been reported [32]. 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 different types of sequences in two different rats. The automated method described above was used to choose λ (Figure 5). In Rat 1, a single significant factor was extracted, corresponding to a sequence active throughout the running wheel delay period (Figure 5B). In Rat 2, three significant factors were identified (Figure 5C). The first two corresponded 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 [18, 35, 29]. In contrast to adult birds, young birds sing highly variable babbling vocalizations, known as subsong, for which HVC is not necessary [1]. The emergence of sequences in HVC occurs gradually over development, as the song matures from subsong to adult song [31].

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 [14]. 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.

Using seqNMF, we were able to identify repeating neural sequences in isolate songbirds (Figure 6A). We found that the HVC network generates several distinct premotor sequences (Figure 6B-C), including sequences deployed during syllables of abnormally long and variable durations (Figure 6D-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 6). We observe that a short sequence occurs alone on some syllable renditions, while on other syllable renditions, a second longer sequences is generated simultaneously. This probabilistic overlap of different sequences is highly atypical in normal adult birds [18, 28, 35, 29]. 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 [14]. 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 [31]. 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 7). The resulting classifications agree with previously published hand-labeled syllable types [31]. 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.

Discussion

As neuroscientists strive to record larger datasets, there is a need for rigorous new tools to reveal underlying structure in high-dimensional data [16, 39, 8, 5]. In particular, sequential structure is increasingly regarded as a fundamental property of neuronal circuits [18, 19, 31, 32], but tools for extracting such structure in neuronal data have been lacking. While convolutional NMF provides a promising framework for extracting sequential structure in high-dimensional datasets, it suffers from a number of weaknesses: It is highly unconstrained, producing many redundant factors that provide a large number of factorizations with equally low reconstruction error. Others have approached the problem of achieving a minimal set of factors by running unregularized CNMF many times from different initial conditions and identifying a subset of the resultant factors that are most reliably produced [34]. Our approach has been to construct a regularizer that, when

incorporated into the multiplicative update rules, drives competition between factors and produces highly consistent factorizations.

While seqNMF regularization is particularly useful when the number of sequences in the data is not known *a priori*, seqNMF does more than simply minimize the number of factors. Even in the context of a minimal set of factors, there are often several different reasonable factorizations. SeqNMF provides a framework for biasing factorizations in a principled way between alternative interpretations of the data. For example, the choice of λ can control the granularity of the clustering of sequences into different factors. At high λ, seqNMF tends to combine similar sequences into a single factor, while at lower λ it tends to place different variants of a sequence into different factors, as shown for the case of temporally warped sequences. As another example, addition of a sparseness regularizer can be used to control the trade off placing features in the pattern exemplars or in the temporal loadings. Similarly, we have found that by including additional orthogonality constraints on $\mathbf{W}$ and $\mathbf{H}$, one can bias factorizations toward parts-based or events-based factorizations, respectively.

While seqNMF is generally quite robust, proper preprocessing of the data can be important to obtaining reasonable factorizations. 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. 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.

SeqNMF addresses a key challenge in extracting neural sequences in complex animal behaviors. Prior analysis methods required aligning neural activity to behavioral events, such as animal position for the case of hippocampal and cortical sequences [19, 32], or vocal output for the case of songbird vocalizations [31]. But this method is not ideally suited for the case highly variable behaviors, such as in early learning and development [31], either normal or abnormal. For example, by applying seqNMF, we were able to identify neural sequences underlying a pathologically variable vocal behavior in the songbird. This technique should enable similar approaches in other cases, expanding the repertoire of behaviors available to neuroscience from those that are repeated and stereotyped to include those that may be variable and rapidly changing.

Acknowledgements

This work was supported by the National Institutes of Health [grant number R01 DC009183], The G. Harold & Leila Y. Mathers Charitable Foundation, and the Simons Collaboration for the Global Brain. ELM received support through the NDSEG Fellowship program. 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 CNMF discussions. Thanks to Michael Stetner, Galen Lynch, Nhat Le, Dezhe Jin and Jane Van Velden for comments on the manuscript and on our code package. Special thanks to the 2017 Methods in Computational Neuroscience course at the Woods Hole Marine Biology Lab, where this collaboration was started.

Author contributions

ELM, AHB, AHW, MSG and MSF conceived the project. ELM, AHB and MSF designed and tested the seqNMF regularizers, the method for cross-validation, and the method for choosing λ. 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. ELM, AHB and MSF wrote the manuscript with input from AHW and MSG.

Methods and Materials

Table of key resources

Key resources, and references for how to access them, are listed in Table 3.

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 and 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, and plot the seqNMF factorization.

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. The binary activity matrix was convolved with an exponential kernel to resemble neural calcium imaging activity.

SeqNMF algorithm details

Our algorithm for seqNMF (CNMF with additional regularization to promote efficient factorizations) is a direct extension of the multiplicative update CNMF algorithm [41], and draws on previous work regularizing NMF to encourage factor orthogonality [7].

The uniqueness and consistency of traditional NMF has been better studied than CNMF, but in special cases, NMF has a unique solution comprised of sparse, 'parts-based' features that can be consistently identified by known algorithms [13, 2]. 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, and instead encourage the model to extract sparse or approximately orthogonal features [22, 25]. Both theoretical work and empirical observations suggest that these modifications result in more consistently identified features [43, 25].

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

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

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

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

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

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 Section 1 by taking the derivative of the cost function in Equation 8.

In addition to the multiplicative updates outlined in Table 2, we also shift factors to be centered in time, renormalize so rows of $\mathbf{H}$ have unit norm, and in the final iteration run one additional step of unregularized CNMF to prioritize the cost of reconstruction error over the regularization (Algorithm 1).

Algorithm 1: SeqNMF

Input: Data matrix $\mathbf{X}$, factor number 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 < NIter) & (Δ cost > tolerance) do
    Update H using multiplicative update from Table 2
    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 2
    Iter = Iter+1
Do one final unregularized CNMF update of W and H
return
```

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 $\widetilde{\mathbf{X}}_k^B = \mathbf{W}_{\cdot k\cdot}^B \circledast \mathbf{H}_{k\cdot}^B$

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 match pair of the remaining factors, etc.
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 overlap of the factor with the held-out data, and compare this to the null case (Figure S1). Overlap with the data is measured as $\mathbf{W} \overset{\top}{\circledast} \mathbf{X}$, so this quantity will be high at moments when the sequence occurs, producing a distribution of $\mathbf{W} \overset{\top}{\circledast} \mathbf{X}$ 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 moments 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; within the null factors, the timecourse of each neuron is circularly shifted 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, uncorrelated 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 CNMF on neural data [34].

During validation of our lambda choosing strategy 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. The product of these two curves was smoothed using a three sample boxcar sliding window and the width was found as the lambda on either side of the peak value which was nearest the half-maximum.

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.

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 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 CNMF [46], 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. [34]).

Hippocampus data

The hippocampal data we used was collected in the Buzsaki lab [32], 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 reviewed and approved by the Massachusetts Institute of Technology Committee on Animal Care.

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 [6] 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 [48].

References

[1] **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.

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

[3] **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.

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

[5] **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.

[6] **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.

[7] **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. .

[8] **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.

[9] **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.

[10] **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.

[11] **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, pMID: 27626963.

[12] **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.

[13] **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.

[14] **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):564–8. http://www.pubmedcentral.nih.gov/articlerender.fcgi?artid=2693086{&}tool=pmcentrez{&}rendertype=abstract, doi: 10.1038/nature07994.

[15] **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.

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

[17] **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.

[18] **Hahnloser RHR**, Kozhevnikov AA, Fee MS. An ultra-sparse code underliesthe generation of neural sequences in a songbird. Nature. 2002 09; 419:65–70. http://dx.doi.org/10.1038/nature00974.

[19] **Harvey CD**, Coen P, Tank DW. Choice-specific sequences in parietal cortex during a virtual-navigation decision task. Nature. 2012 03; 484:62–68. http://dx.doi.org/10.1038/nature10918.

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

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

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

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

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

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

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

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

[28] **Long MA**, Jin DZ, Fee MS. Support for a synaptic chain model of neuronal sequence generation. Nature. 2010 nov; 468(7322):394–9. http://www.pubmedcentral.nih.gov/articlerender.fcgi?artid=2998755{&}tool=pmcentrez{&}rendertype=abstract, doi: 10.1038/nature09514.

[29] **Lynch G**, Okubo T, Hanuschkin A, Hahnloser RR, Fee M. Rhythmic Continuous-Time Coding in the Songbird Analog of Vocal Motor Cortex. Neuron. 2016; 90(4):877 – 892. http://www.sciencedirect.com/science/article/pii/S0896627316301088, doi: https://doi.org/10.1016/j.neuron.2016.04.021.

[30] **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.

[31] **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.

[32] **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.

[33] **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.

[34] **Peter S**, Durstewitz D, Diego F, Hamprecht FA. Sparse convolutional coding for neuronal ensemble identification. ArXiv e-prints. 2016 Jun; .

[35] **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; 90(4):866 – 876. http://www.sciencedirect.com/science/article/pii/S0896627316001094, doi: https://doi.org/10.1016/j.neuron.2016.02.016.

[36] **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.

[37] **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.

[38] **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.

[39] **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.

[40] **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.

[41] **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.

[42] **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.

[43] **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.

[44] **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, doi: 10.1561/2200000055.

[45] **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.

[46] **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.

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

[48] **Zhou P**, Resendez SL, Rodriguez-Romaguera J, Jimenez JC, Neufeld SQ, 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. ArXiv e-prints. 2016 May; .

Figures

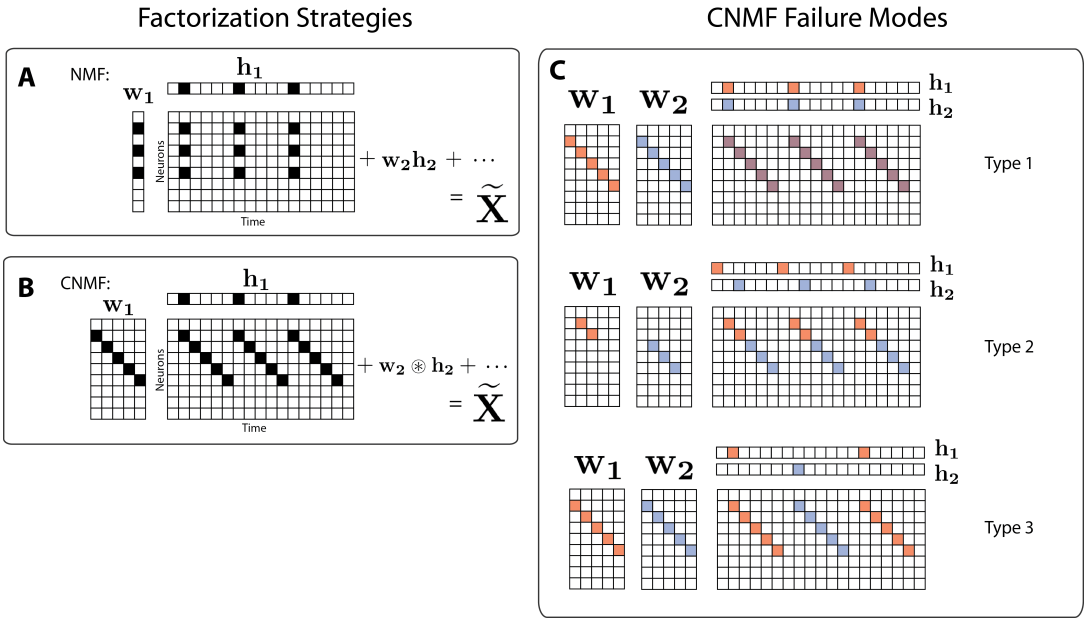

Figure 1. Introduction to CNMF factorization failure modes motivating seqNMF regularization
(A) NMF (non-negative matrix factorization) approximates a dataset containing N neurons at T timepoints as a sum of K rank-one matrices. Each matrix is generated as the outer product of two nonnegative vectors : $\mathbf{w}_k$ 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. **(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. **(C)** Three types of inefficiencies are present in unregularized CNMF: 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.

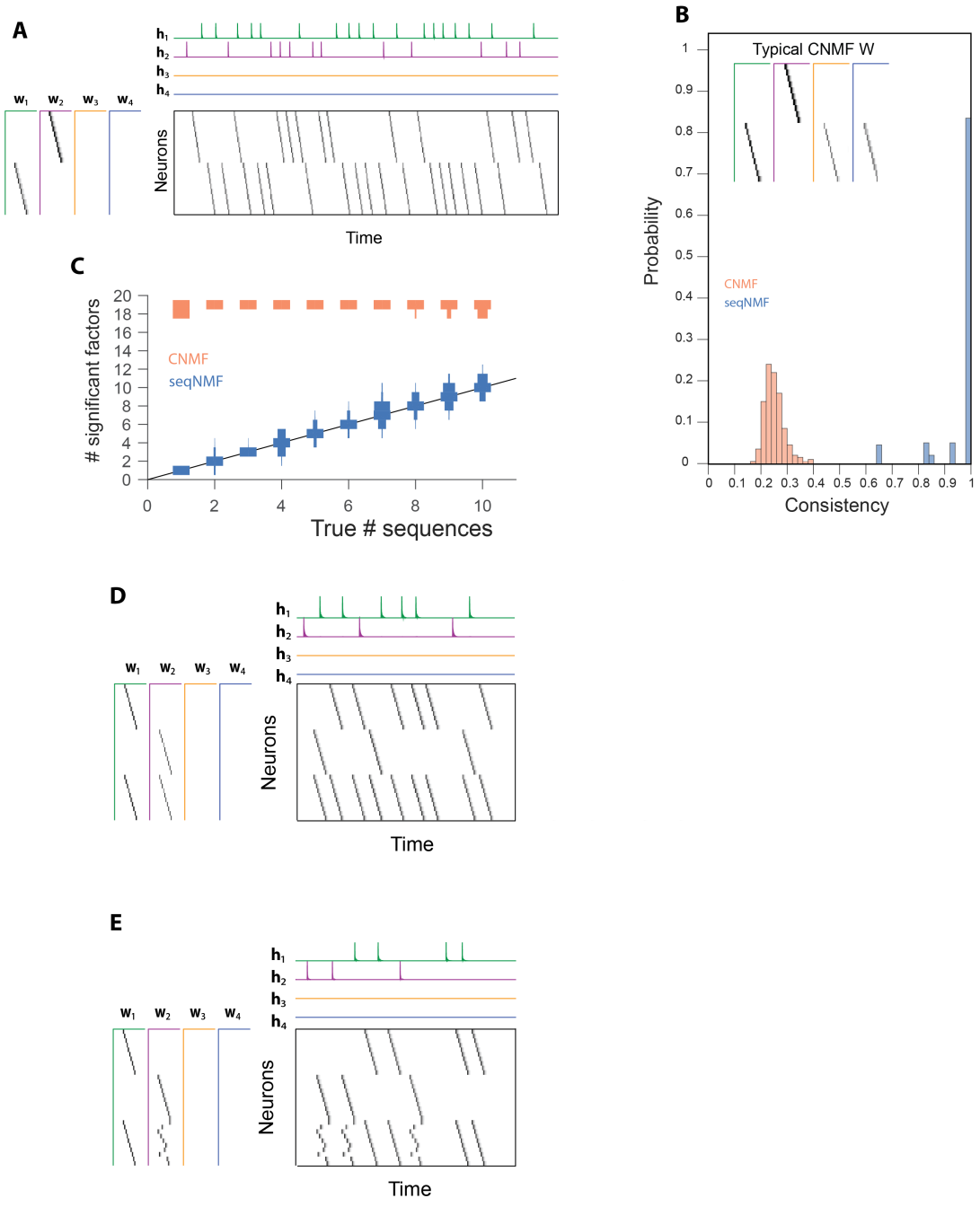

Figure 2. Testing seqNMF on simulated data

(A) A simulated dataset with two simulated neural sequences and a seqNMF factorization ($K = 4$, $L = 250$, $\lambda = 0.0005$) **(B)** SeqNMF is far more consistent than unregularized CNMF across 100 independent fits ($K = 20$, $L = 250$, $\lambda = 0.0005$). Inset: neural patterns for a typical CNMF factorization showing redundant copies of the lower sequence. **(C)** Discrete violin plots showing the number of statistically significant factors vs. true number of simulated sequences for seqNMF and CNMF for 100 fits of simulated data containing between 1 and 10 sequences ($K = 20$, $L = 250$, $\lambda = 0.0005$). **(D)** A seqNMF factorization of two simulated neural sequences with shared neurons that participate at the same latency in both sequences **(E)** A seqNMF factorization of two simulated neural sequences with shared neurons that participate at different latencies in each sequence.

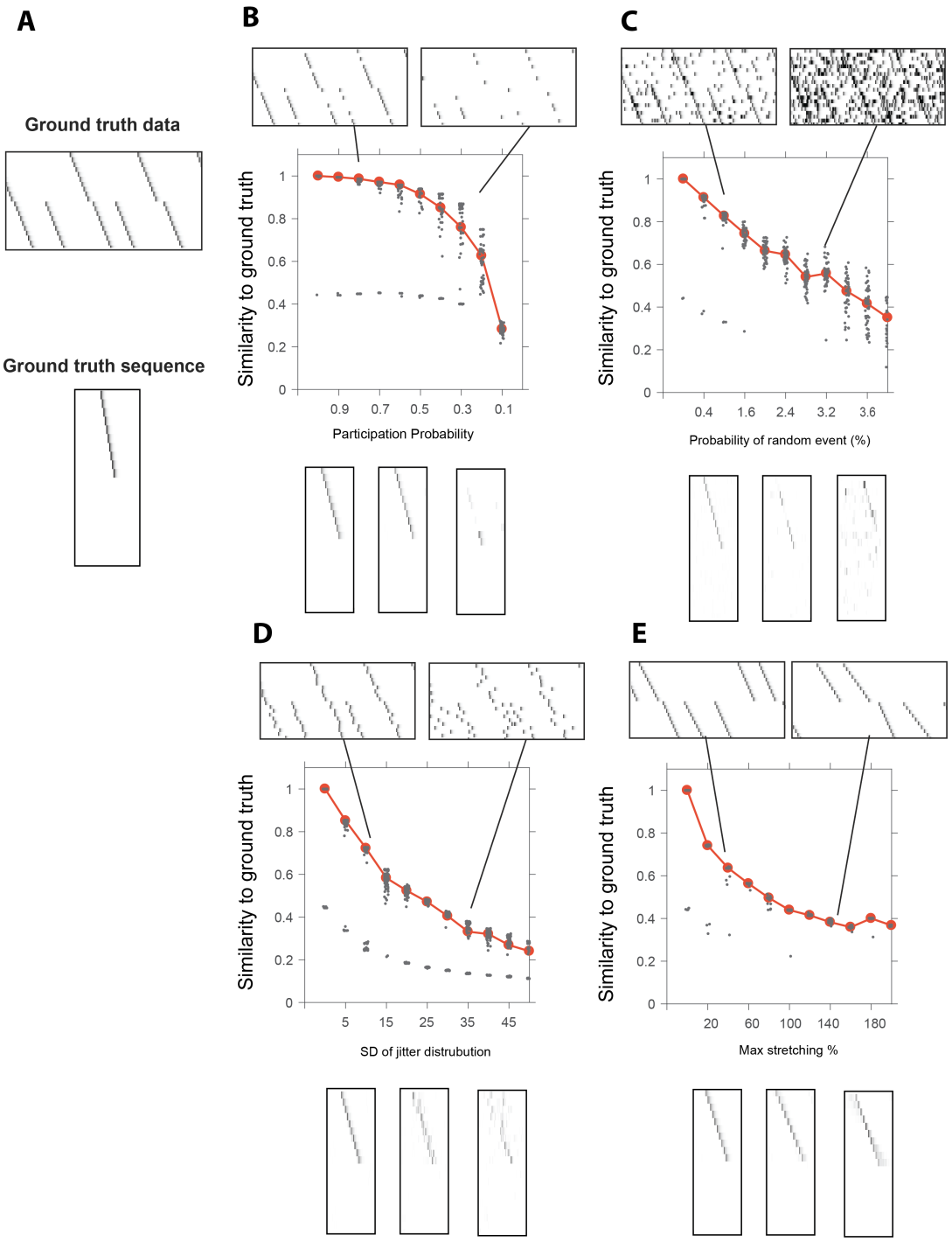

Figure 3. Testing seqNMF performance on sequences contaminated with noise
(A) Ground-truth (noiseless) data, as well as an example of one ground-truth sequence used to generate the data. Performance of seqNMF was tested under 4 different noise conditions: **(B)** probabilistic participation, **(C)** additive noise, **(D)** timing jitter, and **(E)** sequence warping. For each noise type, we show: (top) examples of synthetic data at 2 different noise levels, (middle) similarity of seqNMF factors to ground-truth factors across a range of noise levels, showing 50 fits for each noise level, with red lines indicating the median, and (bottom) example $\mathbf{W}$'s extracted at 3 different noise levels. SeqNMF was run with $K = 20$, $L = 250$, and λ chosen using the automated procedure outlined in Figure 4.

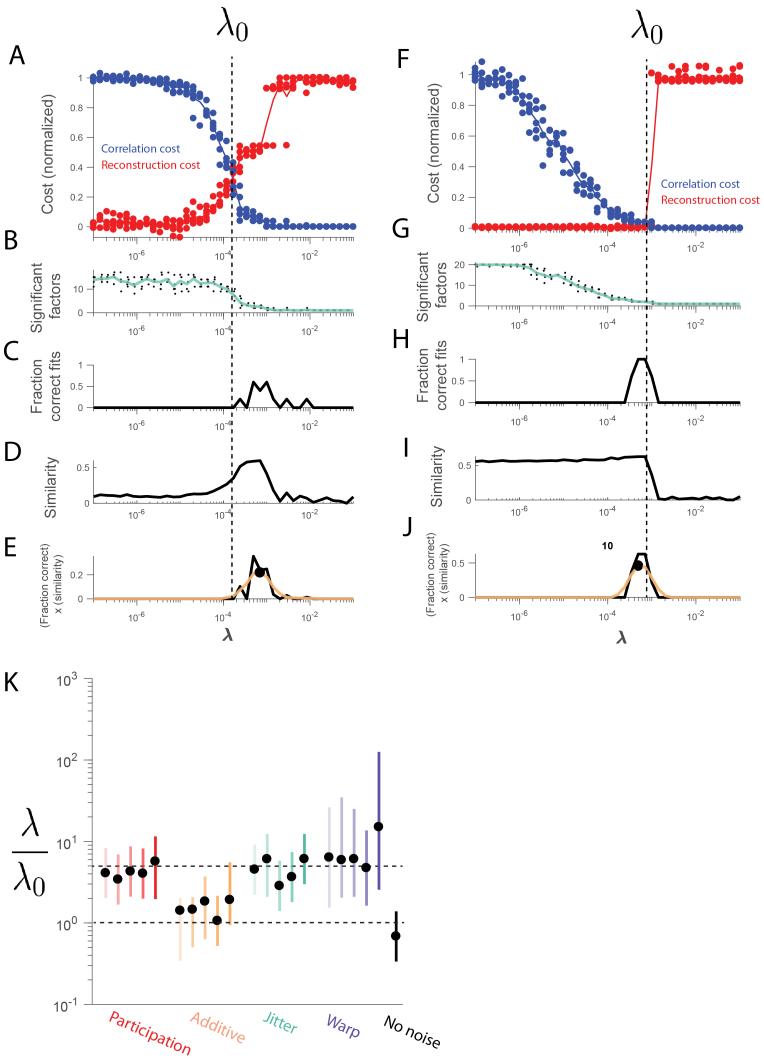

Figure 4. Procedure for choosing λ for a new dataset based on finding a balance between reconstruction cost and correlation cost in noisy and noiseless data

(A) Normalized reconstruction cost ($||\widetilde{\mathbf{X}} - \mathbf{X}||_F^2$) and correlation cost ($||\mathbf{W} \overset{\mathsf{T}}{\circledast} \mathbf{XSH}^{\mathsf{T}}||_{1,i\neq j}$) as a function of λ for simulated data containing two sequences in the presence of participation noise (70% participation probability). The cross-over point λ_0 is marked. **(B)** The number of significant factors obtained from 20 fits of these data as a function of λ (mean number plotted in green). **(C)** The fraction of fits returning the correct number of significant factors (two) as a function of λ. **(D)** Similarity of the top two factors to ground-truth (noiseless) factors as a function of λ. **(E)** The product of the curves shown in (C) and (D), (smoothed curve plotted in orange) with a circle marking the peak. **(F)** Normalized reconstruction cost and correlation cost as a function of λ for simulated data containing two noiseless sequences. **(G-J)** Same as (B-E) but for the noiseless data. **(K)** 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 the curve shown in panel E; 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 of the curves in (E). For each noise type, results for the first five non-zero noise levels from Figure 3 are shown (increasing color saturation at high noise levels; Red, participation: 90,80,70,60 and 50%; Orange, additive noise 0.4, 0.8, 1.2, 1.6 and 2%; Green, jitter: 5,10,15,20, and 25 timesteps; Purple, timewarp: 10,20,30,40,and 50%)

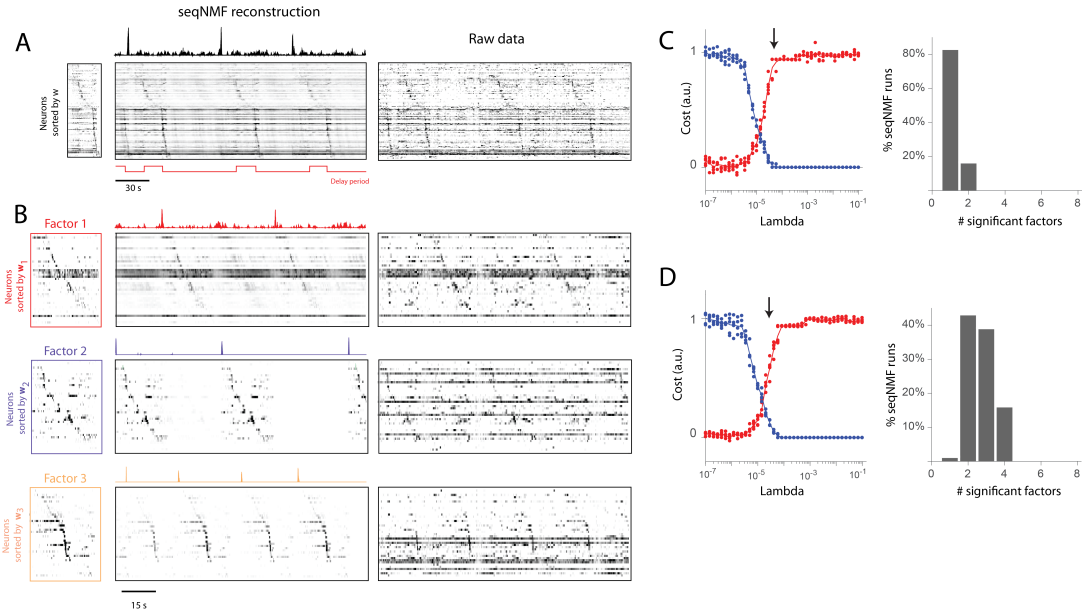

Figure 5. Application of seqNMF to extract hippocampal sequences from two rats
(A) Firing rates of 110 neurons recorded in the hippocampus of Rat 1 during an alternating left-right task with a delay period [32], as well as the seqNMF factor. Neurons are sorted according to their latency within the factor. 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 [32]. Neurons are sorted according to their latency 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 $\lambda = 6x10^{-5}$, 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 $\lambda = 3x10^{-5}$ used for factorization shown in (B).

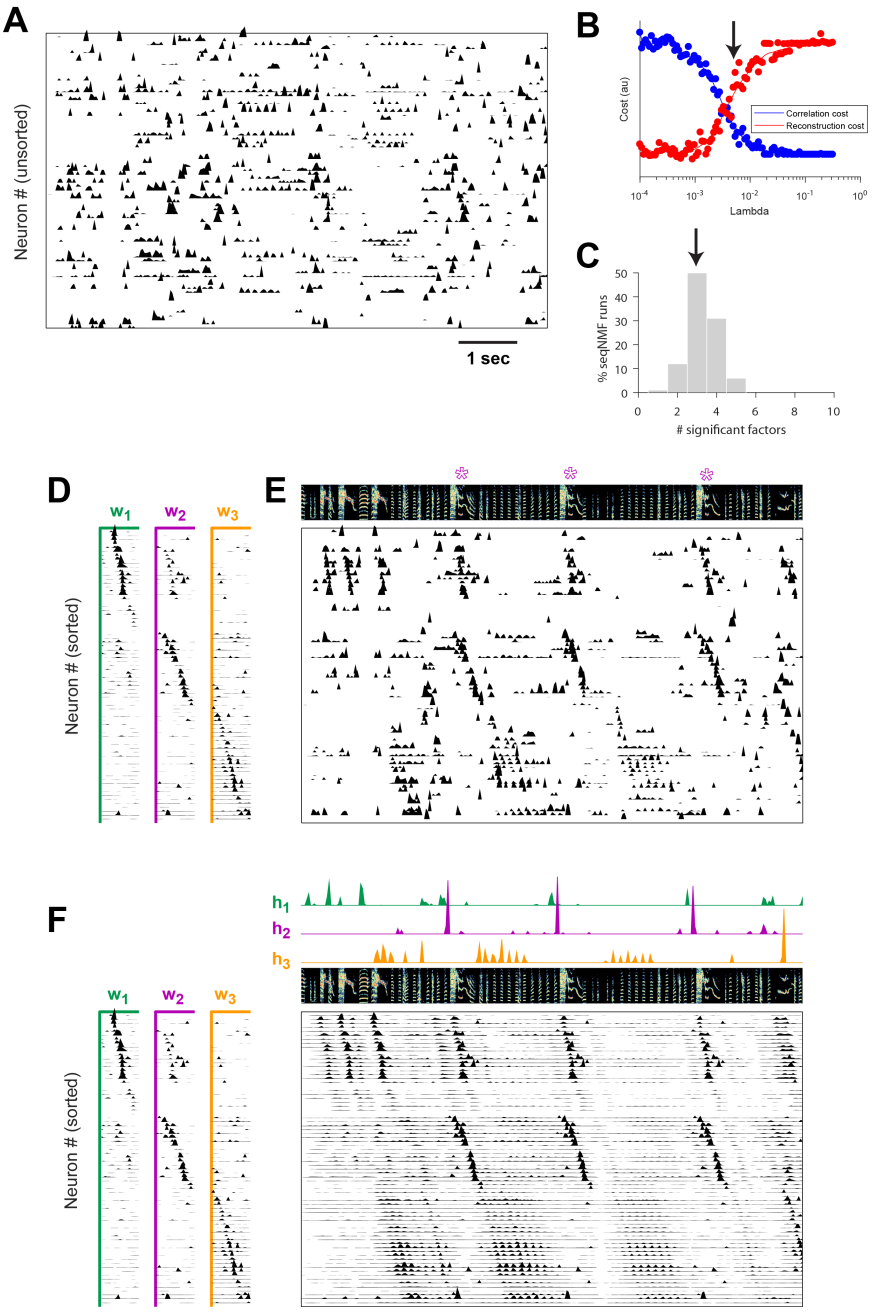

Figure 6. SeqNMF applied to calcium imaging data from a singing isolate bird reveals abnormal sequence deployment

(A) Functional calcium signals recorded from 75 neurons, unsorted, in a singing isolate bird. **(B)** Reconstruction and correlation cost as a function of lambda. 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), sorting neurons by their latency within each 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.

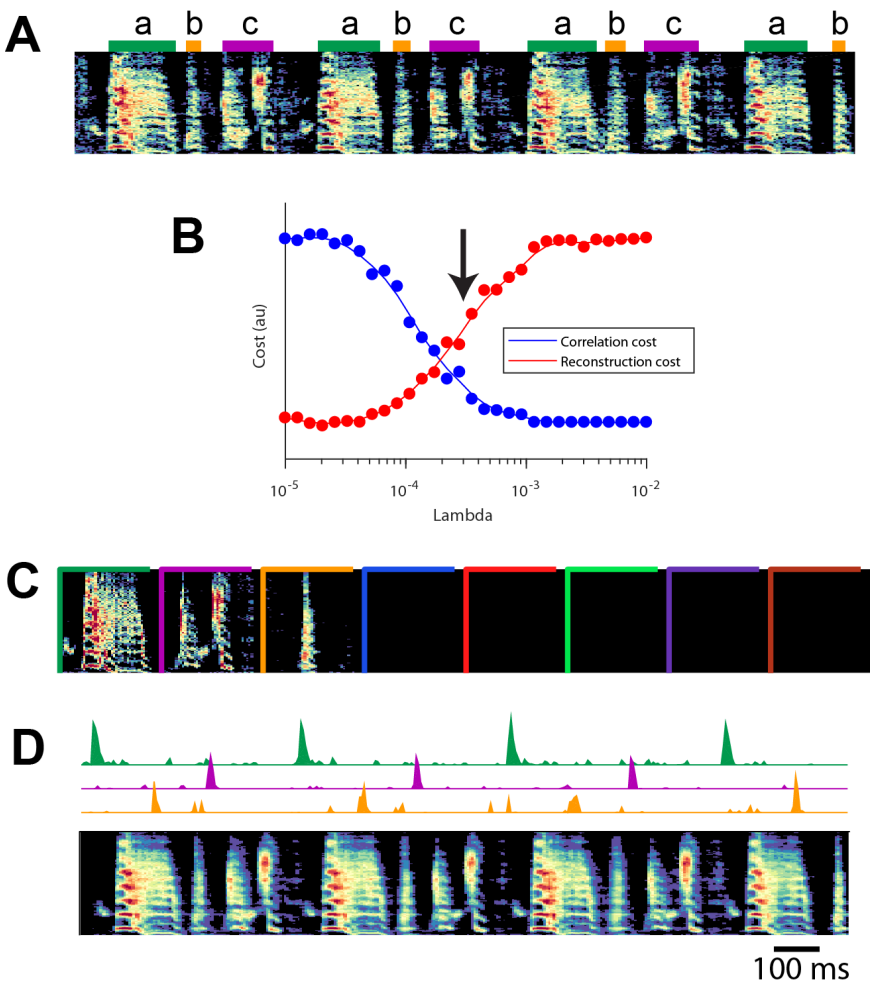

Figure 7. SeqNMF applied to song spectrograms

(A) Spectrogram of juvenile song, with hand-labeled syllable types [31]. **(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.

Table 1. Notation for convolutional matrix factorization

Shift operator
The operator $\overrightarrow{(\cdot)}^{\ell}$ shifts a matrix in the $\rightarrow$ direction by l timebins:
$(\overrightarrow{\mathbf{A}}^{\ell})_{\cdot t} = \mathbf{A}_{\cdot (t-l)}$ and likewise $(\overleftarrow{\mathbf{A}}^{\ell})_{\cdot t} = \mathbf{A}_{\cdot (t+\ell)}$
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:
$\mathbf{W} \overset{\mathsf{T}}{\circledast} \mathbf{X} = \sum_{\ell} \mathbf{W}^{\mathsf{T}}_{\cdot \cdot \ell} \overleftarrow{\mathbf{X}}^{\ell}$
Note that each element $(\mathbf{W} \overset{\mathsf{T}}{\circledast} \mathbf{X})_{kt} = \sum_l \mathbf{W}^{\mathsf{T}}_{\cdot k \ell} \mathbf{X}_{\cdot (t+\ell)}$ 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 special case of CNMF, where $L = 1$
$L1$ norm excluding diagonal
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 smoothing matrix: $s_{ij} = 1$ when $\|i-j\| < L$ and otherwise $s_{ij} = 0$

Table 2. Regularized NMF and CNMF: cost functions and algorithms

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

Table 3. Key resources

Software/algorithm	Source	Link to code
seqNMF	This paper	https://github.com/FeeLab/seqNMF
CNMF	[41, 40]	https://github.com/colinvaz/nmf-toolbox
Sparse CNMF	[30, 36]	https://github.com/colinvaz/nmf-toolbox
Soft orthogonal NMF	[7]	
Other NMF extensions	[9]	
NMF	[27]	
CNMF_E (cell extraction)	[48]	https://github.com/zhoupc/CNMF_E
MATLAB	MathWorks	www.mathworks.com
Dataset	**Source**	**Link to data**
HVC, Isolate songbird	This paper	will upload to CRCNS after publication
Hippocampus, running wheel task	[32]	https://crcns.org/data-sets/hc/hc-3 and /hc-5
Other	**Source**	**Link**
Zebra finches (Taeniopygia guttata)	MIT animal facility	
AAV9.CAG.GCaMP6f.WPRE.SV40	[6]	https://pennvectorcore.med.upenn.edu
Miniature microscope	Inscopix	https://www.inscopix.com/nvista

Supplemental Figures

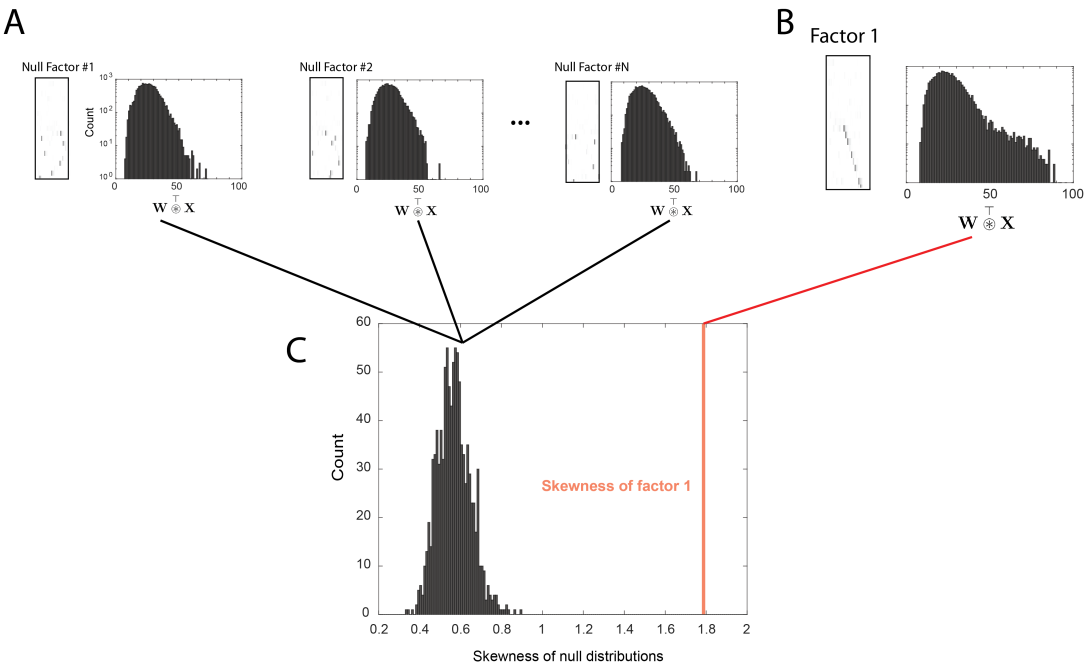

Figure S1. Outline of the procedure used to assess factor significance.
(A) In order to test the significance of a factor on held-out data, we constructed null (shifted) versions of the factor, and measured the distribution of overlap values ($\mathbf{W} \circledast^{\top} \mathbf{X}$) between each null factor and the held-out data. **(B)** We also measured the distribution of overlap values between the real 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. 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 additional smoothed soft orthogonality for $\mathbf{H}$, with $\lambda_{L1\mathbf{H}} = 1$. **(B)** Factorization achieved using additional smoothed soft orthogonality for $\mathbf{W}$, with $\lambda_{L1\mathbf{W}} = 1$.

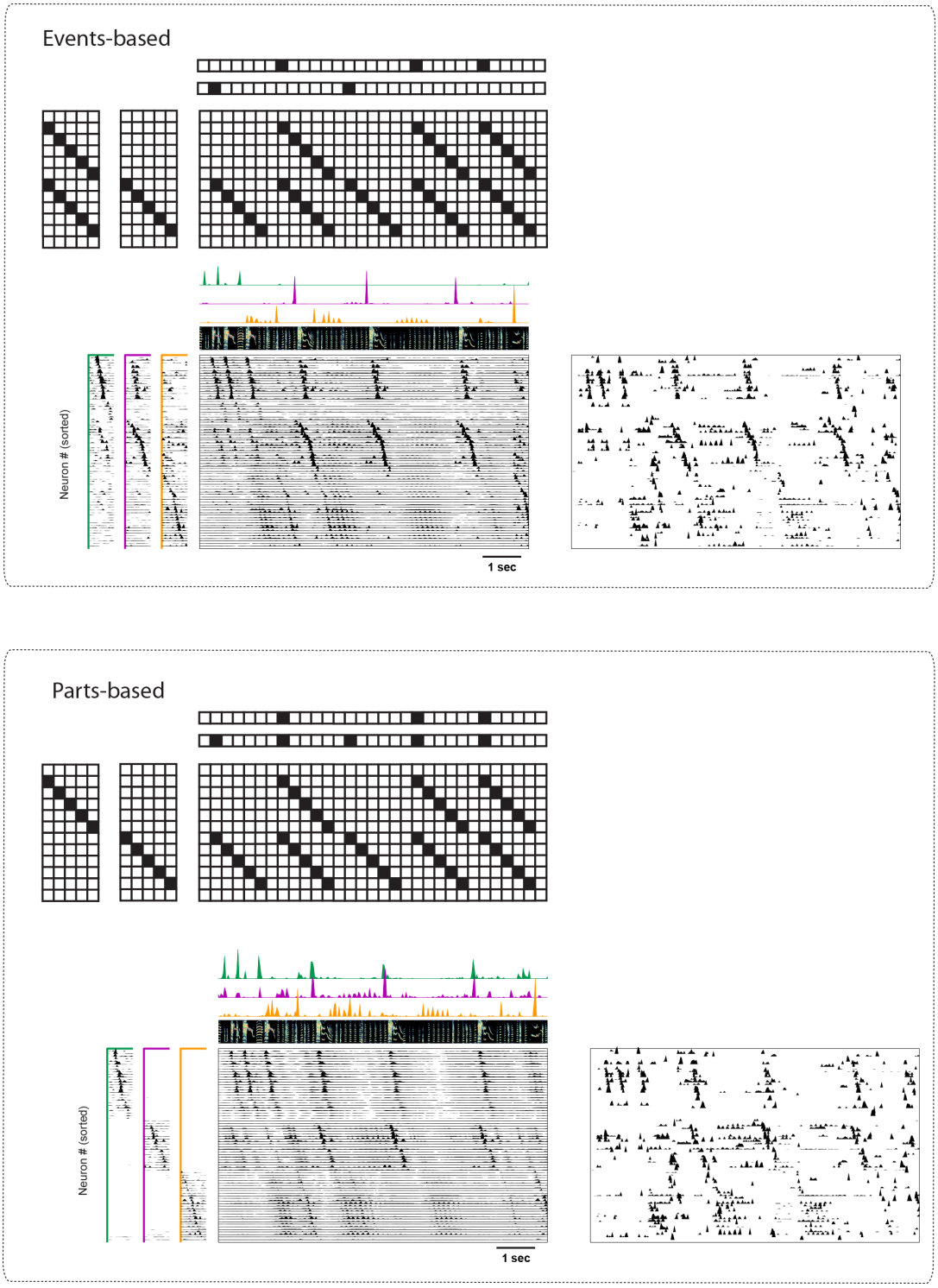

Figure S3. Biasing towards parts-based and events-based factorizations
Illustration of a trade-off between parts-based ($\mathbf{W}$ is more strictly orthogonal) and events-based ($\mathbf{H}$ is more strictly orthogonal) factorizations in a dataset where some neurons are shared between different sequences. The same data as in Figure 6 is factorized using smoothed soft orthogonality on $\mathbf{H}$ (top, events-based), or on $\mathbf{W}$ (bottom, 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 6. 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 [27]. 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 [41] apply several forms of regularization [30, 36, 7]. See Table 2 for a compilation of cost functions, derivatives and multiplicative updates for NMF and CNMF under several different regularization conditions.

Standard NMF

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

$$(\widetilde{\mathbf{W}}, \widetilde{\mathbf{H}}) = arg \min_{\mathbf{W},\mathbf{H}} \mathscr{L}(\mathbf{W}, \mathbf{H}) \tag{11}$$

$$\mathscr{L}(\mathbf{W}, \mathbf{H}) = \frac{1}{2}||\widetilde{\mathbf{X}} - \mathbf{X}||_F^2 \tag{12}$$

$$\widetilde{\mathbf{W}}, \widetilde{\mathbf{H}} \geq 0 \tag{13}$$

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:

$$\frac{d}{d\mathbf{W}}\mathscr{L}(\mathbf{W}, \mathbf{H}) = \widetilde{\mathbf{X}}\mathbf{H}^\top - \mathbf{X}\mathbf{H}^\top \tag{14}$$

$$\frac{d}{d\mathbf{H}}\mathscr{L}(\mathbf{W}, \mathbf{H}) = \mathbf{W}^\top\widetilde{\mathbf{X}} - \mathbf{W}^\top\mathbf{X} \tag{15}$$

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

$$\mathbf{W} \leftarrow \mathbf{W} - \eta_{\mathbf{W}}(\widetilde{\mathbf{X}}\mathbf{H}^\top - \mathbf{X}\mathbf{H}^\top) \tag{16}$$

$$\mathbf{H} \leftarrow \mathbf{H} - \eta_{\mathbf{H}}(\mathbf{W}^\top\widetilde{\mathbf{X}} - \mathbf{W}^\top\mathbf{X}) \tag{17}$$

To arrive at multiplicative updates, Lee and Seung [27] set:

$$\eta_{\mathbf{W}} = \frac{\mathbf{W}}{\mathbf{W}\mathbf{H}\mathbf{H}^\top} \tag{18}$$

$$\eta_{\mathbf{H}} = \frac{\mathbf{H}}{\mathbf{W}^\top\mathbf{W}\mathbf{H}} \tag{19}$$

Thus, the gradient descent updates become multiplicative:

$$\mathbf{W} \leftarrow \mathbf{W} \times \frac{\mathbf{X}\mathbf{H}^\top}{\mathbf{W}\mathbf{H}\mathbf{H}^\top} = \mathbf{W} \times \frac{\mathbf{X}\mathbf{H}^\top}{\widetilde{\mathbf{X}}\mathbf{H}^\top} \tag{20}$$

$$\mathbf{H} \leftarrow \mathbf{H} \times \frac{\mathbf{W}^\top\mathbf{X}}{\mathbf{W}^\top\mathbf{W}\mathbf{H}} = \mathbf{H} \times \frac{\mathbf{W}^\top\mathbf{X}}{\mathbf{W}^\top\widetilde{\mathbf{X}}} \tag{21}$$

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

Standard CNMF

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

$$(\widetilde{\mathbf{W}}, \widetilde{\mathbf{H}}) = arg\min_{\mathbf{W},\mathbf{H}} \mathscr{L}(\mathbf{W}, \mathbf{H}) \tag{22}$$

$$\mathscr{L}(\mathbf{W}, \mathbf{H}) = \frac{1}{2}||\widetilde{\mathbf{X}} - \mathbf{X}||_F^2 \tag{23}$$

$$\widetilde{\mathbf{W}}, \widetilde{\mathbf{H}} \geq 0 \tag{24}$$

The derivation above for standard NMF can be applied for each ℓ, yielding the following update rules for CNMF [41]:

$$\mathbf{W}_{\cdot\cdot\ell} \leftarrow \mathbf{W}_{\cdot\cdot\ell} \times \frac{\mathbf{X}\overset{\ell\rightarrow}{\mathbf{H}}^\top}{\widetilde{\mathbf{X}}\overset{\ell\rightarrow}{\mathbf{H}}^\top} \tag{25}$$

$$\mathbf{H} \leftarrow \mathbf{H} \times \frac{\sum_\ell \mathbf{W}_{\cdot\cdot\ell}^\top \overset{\leftarrow\ell}{\mathbf{X}}}{\sum_\ell \mathbf{W}_{\cdot\cdot\ell}^\top \overset{\leftarrow\ell}{\widetilde{\mathbf{X}}}} = \mathbf{H} \times \frac{\mathbf{W}^\top \circledast \mathbf{X}}{\mathbf{W}^\top \circledast \widetilde{\mathbf{X}}} \tag{26}$$

Note that NMF is a special case of CNMF where $L = 0$.

Incorporating regularization terms

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

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

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

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

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 CNMF case.

Note that:

$$\frac{d\mathscr{L}}{d\mathbf{W}} = \widetilde{\mathbf{X}}\mathbf{H}^\top - \mathbf{X}\mathbf{H}^\top + \frac{d\mathscr{R}}{d\mathbf{W}} \tag{30}$$

$$\frac{d\mathscr{L}}{d\mathbf{H}} = \mathbf{W}^\top\widetilde{\mathbf{X}} - \mathbf{W}^\top\mathbf{X} + \frac{d\mathscr{R}}{d\mathbf{H}} \tag{31}$$

We set:

$$\eta_{\mathbf{W}} = \frac{\mathbf{W}}{\widetilde{\mathbf{X}}\mathbf{H}^\top + \frac{d\mathscr{R}}{d\mathbf{W}}} \tag{32}$$

$$\eta_{\mathbf{H}} = \frac{\mathbf{H}}{\mathbf{W}^\top\widetilde{\mathbf{X}} + \frac{d\mathscr{R}}{d\mathbf{H}}} \tag{33}$$

Thus, the gradient descent updates become multiplicative:

$$\mathbf{W} \leftarrow \mathbf{W} - \eta_{\mathbf{W}}\frac{d\mathscr{L}}{d\mathbf{W}} = \mathbf{W} \times \frac{\mathbf{X}\mathbf{H}^\top}{\widetilde{\mathbf{X}}\mathbf{H}^\top + \frac{d\mathscr{R}}{d\mathbf{W}}} \tag{34}$$

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

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

This framework enables flexible incorporation of different types of regularization into the multiplicative NMF update algorithm. This framework also extends naturally to the convolutional case. See Table 2 for examples of several regularization terms, including $L1$ sparsity [30, 36] and soft orthogonality [7], 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:

$$\mathbf{W}_{..\ell} \leftarrow \mathbf{W}_{..\ell} \times \frac{\mathbf{X}\left(\overset{\ell\rightarrow}{\mathbf{H}}\right)^\top}{\widetilde{\mathbf{X}}\left(\overset{\ell\rightarrow}{\mathbf{H}}\right)^\top + \lambda\,\overset{\leftarrow\ell}{\mathbf{X}}\mathbf{S}\mathbf{H}^\top(\mathbf{1}-\mathbf{I})} \tag{36}$$

$$\mathbf{H} \leftarrow \mathbf{H} \times \frac{\mathbf{W} \overset{\top}{\circledast} \mathbf{X}}{\mathbf{W} \overset{\top}{\circledast} \widetilde{\mathbf{X}} + \lambda(\mathbf{1}-\mathbf{I})(\mathbf{W} \overset{\top}{\circledast} \mathbf{X}\mathbf{S})} \tag{37}$$

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 (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.