# Combining tensor decomposition and time warping models for multi-neuronal spike train analysis

Alex H. Williams
Statistics Department, Stanford University, Stanford, CA, USA, 94305
ahwillia@stanford.edu

**Abstract**

Recordings from large neural populations are becoming an increasingly popular and accessible method in experimental neuroscience. While the activity of individual neurons is often too stochastic to interrogate circuit function on a moment-by-moment basis, multi-neuronal recordings enable us to do so by pooling statistical power across many cells. For example, groups of neurons often exhibit correlated gain or amplitude modulation across trials, which can be statistically formalized in a tensor decomposition framework (Williams et al. 2018). Additionally, the time course of neural population dynamics can be shifted or stretched/compressed, which can be modeled by time warping methods (Williams et al. 2019). Here, we compare and contrast these two modeling frameworks and demonstrate how they can be combined. We demonstrate that combining these methods may be highly advantageous—for example, the presence of random time shifts hampers the performance and interpretability of tensor decomposition, while a time-shifted variant of this model corrects for these disruptions and uncovers ground truth structure in simulated data.

## Introduction

The recent proliferation of high-density neural recording technologies (T. H. Kim et al. 2016; Stringer et al. 2018; Chen et al. 2018) has fueled research into statistical models of single-trial neural dynamics (Churchland et al. 2007; Petreska et al. 2011; Pandarinath et al. 2018; Duncker and Sahani 2018; Williams et al. 2018). In recent work myself and collaborators explored models that capture two complementary forms of single-trial variability. First, in Williams et al. (2018) we studied the possibility of correlated amplitude variability or gain modulation amongst sub-populations of co-recorded neurons. We found that such a model could be formalized as a well-studied **tensor decomposition** problem (T. Kolda and Bader 2009). Second, in Williams et al. (2019) we studied variability in the timing of dynamics, and formalized this possibility as a **time warping** model.

For clarity, we explored the merits of these models in separate papers. Nonetheless, a natural question is whether these concepts can be combined into a unified model. After all, in many experiments we would expect both the amplitude and timing of neural dynamics to vary from trial-to-trial. This manuscript provides a self-contained technical summary of this prior work, and describes how concepts from both models may be combined. In particular, I explore two hybrid models: the first extends tensor decomposition by adding per-neuron and per-trial temporal shift parameters (I refer to this as **time-shifted tensor decomposition**); the second extends the time warping model by incorporating additional dynamical components (I refer to this as a **multi-shift model**).

Previous works have explored similar ideas, though with different motivating applications or proposed algorithms (Harshman et al. 2003; S. Hong and Harshman 2003; Mørup et al. 2008; S. Hong 2009; Q. Wu et al. 2014). In chemometrics, for example, absorption and emission spectra are often shifted on an instance-by-instance basis due to measurement variability, and this has been incorporated into tensor decomposition models of these datasets (Harshman et al. 2003; S. Hong 2009). In fMRI data, Mørup et al. (2008) introduced and applied a shifted tensor decomposition model with periodic boundary conditions—while periodic boundaries ameliorate computational costs, it may be inappropriate in many cases where evoked neural responses are transient and not rhythmic or periodic. A more recent paper by Duncker and Sahani (2018) incorporates warping functions into a low-dimensional Gaussian Process factor model for spike train analysis; they fit a single nonlinear warping function to all components on each trial, whereas the models I describe apply separate shift parameters to each component, thus enabling the possibility of discovering sub-populations of neurons that are independently time-shifted on a trial-by-trial basis. I also incorporate the possibility that individual units or neurons have characteristic time-shifts in their dynamics, which is not considered by Mørup et al. (2008) or Duncker and Sahani

(2018). Additionally, I describe how time warping, tensor decomposition, and hybrids of these two models can be fit by a common optimization strategy based on coordinate and block-coordinate descent (Wright 2015). Overall, this paper aims to synthesize many of these prior works into a unified conceptual framework so that they can be readily understood and applied by practitioners in neuroscience.

# 1 Models

## 1.1 Notation

Let $X_{tnk}$ denote the activity of neuron $n$, on trial $k$, in time bin $t$. Let $N$ denote the total number of neurons, $K$ be the total number of trials, and $T$ be the number of timebins in each trial. While we will focus on multi-neuronal recordings as a motivating example, it should be emphasized that the methods we describe are all applicable to *any* multi-dimensional time series dataset with repeated trials. For example, in the case of behavioral time series, $X_{tnk}$ could denote the position of body marker $n$, on trial $k$, at time bin $t$. These methods could also be applied to fMRI data with $n$ indexing over voxels, $k$ indexing trials, and $t$ indexing timebins (Mørup et al. 2008).

Lowercase bold symbols denote vectors, e.g. a vector with $n$ elements: $\mathbf{v} \in \mathbb{R}^n$. Uppercase bold symbols denote matrices, e.g. a matrix with $m$ rows and $n$ columns: $\mathbf{M} \in \mathbb{R}^{m \times n}$. We denote matrix transposes and inverses as $\mathbf{M}^\mathsf{T}$ and $\mathbf{M}^{-1}$, respectively. The trace of a square matrix is denoted $\mathrm{Tr}[\mathbf{M}]$. We will frequently refer to slices of arrays. For example, $\mathbf{X}_k \in \mathbb{R}^{T \times N}$ will denote the activity of all neurons on trial $k$, and $\mathbf{x}_{nk} \in \mathbb{R}^T$ will denote the activity of the $n^{\text{th}}$ neuron on the $k^{\text{th}}$ trial. These slicing operations should be familiar to readers that use scientific programming languages; for example, in Python $\mathbf{X}_k$ and $\mathbf{x}_{nk}$ respectively correspond the indexing operations `X[:, :, k]` and `X[:, n, k]`.

Our goal is to define a low-dimensional and interpretable model that approximates this multi-trial dataset. We will use $\widehat{X}_{tnk}$ to denote the predicted activity of neuron $n$, on trial $k$, in timebin $t$. For simplicity, will will assume a quadratic loss function (least-squares criterion) throughout this manuscript:

$$\text{loss} = \sum_{n=1}^{N} \sum_{k=1}^{K} \sum_{t=1}^{T} \left( X_{tnk} - \widehat{X}_{tnk} \right)^2 \tag{1}$$

Minimizing this quadratic loss is equivalent to performing maximum likelihood inference in a probabilistic model with Gaussian noise. Different loss functions can be incorporated into tensor decomposition and time warping models (Chi and T. G. Kolda 2012; D. Hong et al. 2018). In particular, a Poisson likelihood criterion is a popular loss function to use for binned spike counts, particularly in low firing rate regimes (Paninski 2004). However, the quadratic loss enables much simpler and faster optimization routines, and I provide some empirical evidence that this loss function can perform well even with low firing rates (see fig. 1; see also Fig 6 in Williams et al. 2019). A more thorough empirical comparison of potential loss functions on spike train data is a worthy direction of future research.

## 1.2 Tensor Decomposition

In Williams et al. (2018) the following **tensor decomposition model** is proposed for neural data:

$$\widehat{X}_{tnk} = \sum_{r=1}^{R} u_{nr} v_{kr} w_{tr} \tag{2}$$

This model approximates the data with $R$ components; this is also called a rank-$R$ decomposition or a rank-$R$ model. The variables $u_{nr}$ provide a low-dimensional description of the measured features (*neuron factors*); the variables $v_{kr}$ provide a low-dimensional description of each trial (*trial factors*); the variables $w_{tr}$ provide a low-dimensional description of the temporal dynamics within every trial (*temporal factors*). This model is also known as Canonical Polyadic (CP) tensor decomposition, parallel factor analysis (PARAFAC), and tensor components analysis (TCA). This general method dates back to (Carroll and Chang 1970); for an authorative and contemporary review see T. Kolda and Bader (2009).

We will see that it is useful to reformulate eq. (2) in terms of standard matrix/vector operations. One can show, for example, that eq. (2) is equivalent to:

$$\widehat{\mathbf{X}}_k = \sum_r v_{kr} \mathbf{u}_r \mathbf{w}_r^\mathsf{T} \tag{3}$$

with $\mathbf{X}_k \in \mathbb{R}^{N \times T}$ denoting the neural population activity on trial $k$, and $\mathbf{u}_r \in \mathbb{R}^N$ and $\mathbf{w}_r \in \mathbb{R}^T$ respectively denoting the neural factor and temporal factor for the $r^{\text{th}}$ component. This reformulation makes the role of $v_{kr}$ as a per-trial gain parameter for component $r$ more readily apparent—the full population activity on trial $k$ is modeled as a linear combination of rank-one matrices, $\mathbf{u}_r \mathbf{w}_r^{\mathsf{T}}$, with weights given by $v_{kr}$.[1] We will see that eq. (3) is a useful formulation when drawing conceptual connections between tensor decomposition and time warping models.

This model has an intuitive interpretation and connection to existing neuroscience research—namely, that ensembles of neurons with similar firing rate profiles are *modulated in amplitude* across trials. This form of trial-to-trial variability, often called *gain modulation*, is consistent with a variety of experimental observations and theoretical models of neural circuits (Salinas and Thier 2000; Goris et al. 2014; Carandini and Heeger 2011; Rabinowitz et al. 2015).

This tensor decomposition model has several other attractive properties. First, it is a relatively straightforward generalization of principal components analysis (PCA), which is already a popular method in neural data analysis. Second, while tensor decomposition is related to PCA, it has favorable properties from the standpoint of *statistical identifiability*. Specifically, under mild conditions the decomposition is *essentially unique*, meaning that non-orthogonal features can be identified by this method (Kruskal 1977; Rhodes 2010; Lim and Comon 2009). PCA, in contrast, is non-unique under rotations and invertible linear transformations of the factors, which limits the interpretability of the model. Third and finally, the tensor decomposition model contains exponentially fewer parameters than a PCA model, since it simultaneously reduces the dimensionality of the data across all three axes of the data array (neurons, timepoints, and trials). Thus, tensor decomposition produces a much more aggressive compression of the data, leading to more digestable and visualizable summary of the data. See Williams et al. (2018) for further details and discussion.

## 1.3 Time Warping

In principle, tensor decomposition can be applied to any multi-neuronal recording with a repeated trial structure; however, in certain situations it may not produce a low-dimensional description of the data that is both accurate and interpretable. Specifically, an accurate tensor decomposition may require a large number of components when (a) neurons within an ensemble fire at slightly different times, or (b) neural ensembles are activated at different times on each trial. Intuitively, these two scenarios result in high-dimensional structure across neurons and high-dimensional structure across trials, respectively. Since tensor decomposition assumes that the data are simultaneously low-dimensional across neurons, trials, and timebins, *high-dimensional structure across any one of these axes may cause difficulties*.

To capture variability in timing across trials, Williams et al. (2019) proposed the following **time warping model**:

$$\widehat{X}_{tnk} = \widetilde{x}_n(\omega_k(t)) \tag{4}$$

Here $\omega_k(t)$ is a monotonically increasing **warping function** mapping integers $\{1, 2, \ldots, T\}$ to real numbers on the interval $[1, T]$. The vector $\widetilde{\mathbf{x}}_n$ is a length-$T$ vector holding the *response template* for each neuron $n$. The response templates for each neuron, $\widetilde{\mathbf{x}}_n$, and the warping functions for each trial, $\omega_k(t)$, are optimized to minimize the least-squares criterion as noted in eq. (1).

Since $\omega_k(t)$ is not generally an integer it cannot be used as an index into the discrete vector $\widetilde{\mathbf{x}}_n$. Equation (4) thus contains an implied **linear interpolation** step:

$$\widetilde{x}_n(\omega_k(t)) = \Big(\omega_k(t) - \lfloor \omega_k(t) \rfloor\Big) \cdot \widetilde{x}_{n,\lfloor \omega_k(t) \rfloor} + \Big(\lceil \omega_k(t) \rceil - \omega_k(t)\Big) \cdot \widetilde{x}_{n,\lceil \omega_k(t) \rceil} \tag{5}$$

where $\lfloor \cdot \rfloor$ and $\lceil \cdot \rceil$ respectively denote flooring (rounding down to the nearest integer) and ceiling (rounding up to the nearest integer) operations. We will continue to use parenthetical indexing—e.g., $x(t)$—to denote indexing with linear interpolation, and subscript indexing—e.g., $x_t$—to denote direct indexing with integer variables.

By its virtue of being linear, the interpolation step in eq. (5) can be expressed as a matrix multiplication. Thus, every warping function, $\omega_k(t)$, is uniquely associated with a warping matrix, $\mathbf{\Omega}_k$. For example, suppose the warping function on each trial has the form $\omega_k(t) = t + \beta_k$, where $\beta_k$ is a per-trial shift parameter. In Williams et al. (2019) we refer to this a **shift-only warping model**. Of course, when $\beta_k = 0$, we have $\omega_k(t) = t$ and $\mathbf{\Omega}_k$ is a $T \times T$ identity matrix. The warping matrices associated with some other example warping functions are show below:

[1] In the general case, the per-trial weights $v_{kr}$ may be positive or negative, which perhaps complicates our interpretation of these weights as "amplitude" parameters (typically one does not think of "negative amplitudes"). Often, it is desirable to restrict the factor parameters, $\{u_{nr} v_{kr} w_{tr}\}$, to be nonnegative as this results in a more interpretable model (see section 1.4.1). In this case, the interpretation of $v_{kr}$ as an amplitude is more clear.

$$\boxed{\omega_k(t) = t - 1} \qquad \boxed{\omega_k(t) = t + 2} \qquad \boxed{\omega_k(t) = t - 0.3}$$

$$\mathbf{\Omega}_k = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix}, \quad \mathbf{\Omega}_k = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \end{bmatrix}, \quad \mathbf{\Omega}_k = \begin{bmatrix} 0.7 & 0.3 & 0 & 0 & 0 \\ 0 & 0.7 & 0.3 & 0 & 0 \\ 0 & 0 & 0.7 & 0.3 & 0 \\ 0 & 0 & 0 & 0.7 & 0.3 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix} \tag{6}$$

Using this notation, we can reformulate eq. (4) as follows:

$$\widehat{\mathbf{X}}_k = \widetilde{\mathbf{X}}\mathbf{\Omega}_k \tag{7}$$

which is the time warping analogue to eq. (3). Note that the above shift model clamps the endpoint values of the response template; if desired, this can be modified to accommodate periodic boundary conditions (Mørup et al. 2008).

In contrast to tensor decomposition, the time warping model defined in eq. (4) does *not* assume that firing rate dynamics are low-dimensional, since a different firing rate template is learned for each neuron. Further, by transforming the time axis on each trial by a learned warping function, the model can account for variability in the onset and duration of neural dynamics. Thus, time warping accounts for the two "failure modes" of tensor decomposition outlined at the beginning of this section.

On the other hand, there are some drawbacks to this time warping model. The model does not explicitly account for trial-by-trial variations in amplitude, which, as discussed in section 1.2, are often of interest. Perhaps more importantly, the time warping model assumes that *all neurons share the same warping function on a trial-by-trial basis* and thus does not explore potential variations in timing expressed across multiple sub-populations of neurons. Thus, the time warping and tensor decomposition models have complementary strengths and weaknesses. The following two sections discuss modeling extensions that aim to achieve the best of both models.

## 1.4 Time-Shifted Tensor Decomposition

As discussed above, there are two principal "failure modes" in which tensor decomposition *nearly* works, but nonetheless splits cell ensembles into multiple factors. First, neurons may fire with slightly different latencies to each other, for example in a short temporal sequence (see Mackevicius et al. 2019 and references therein). Second, the ensemble itself may fire at a different latency on a trial-by-trial basis (as discussed above in section 1.3).

One way to correct these discrepancies is by incorporating additional time warping functions into each component of the tensor decomposition. The resulting model can be very complex if each time warping function is allowed to be highly nonlinear; however, a key empirical observation in Williams et al. (2018) is that very simple forms of time warping are sufficient to uncover interesting features in neural data. Thus, we will introduce the simplest form of time warping, *shift-only warping*, leading to the following **time-shifted tensor decomposition** model (see Harshman et al. 2003; Mørup et al. 2008 for related prior work):

$$\widehat{X}_{tnk} = \sum_{r=1}^{R} u_{nr} v_{kr} w_r(t + \alpha_{nr} + \beta_{kr}) \tag{8}$$

Here, the parenthetical indexing notation, $w_r(\cdot)$, denotes linear interpolation as in eq. (5). In addition to optimizing the neural, trial, and temporal factors $\{u_{nr}, v_{kr}, w_{tr}\}$, we now must also optimize $\alpha_{nr}$ and $\beta_{kr}$ which may be interpreted as a per-neuron and per-trial shift parameters for each low-dimensional component. The time-shifted tensor decomposition has a total of $(2NR + KR + 2TR)$ parameters, which is a very modest increase over the number of parameters in vanilla tensor decomposition.

Equation (8) can be reformulated as:

$$\widehat{\mathbf{x}}_{nk} = \sum_{r=1}^{R} u_{nr} v_{kr} \mathbf{\Omega}_{nkr} \mathbf{w}_r \tag{9}$$

Where $\mathbf{\Omega}_{nkr}$ denotes a time warping matrix as in eq. (6). Note that a different warping function is applied to each low-dimensional temporal factor, $\mathbf{w}_r$, for every trial and neuron.

Optimizing time-shifted tensor decomposition is potentially much more challenging than vanilla tensor decomposition, due to the additional shift parameters which are not easily fit by gradient-based methods. Nonetheless, we have found this to be possible in practice, as long as the magnitudes of the shift parameters are not too large.

### 1.4.1 A note on nonnegative factorizations

Nonnegative tensor decomposition (Lim and Comon 2009; Paatero 1997) is a higher-order analogue of the well-known nonnegative matrix factorization model (Lee and Seung 1999). In essence, fitting this model entails constraining the neural factors ($u_{nr}$), trial factors ($v_{kr}$), and temporal factors ($w_{tr}$) to be greater than or equal to zero during optimization. In Williams et al. (2018), we reported that this constraint improves the consistency and interpretability of tensor decomposition without greatly hurting the model's accuracy. In section 2, we will see that similar nonnegativity constraints can be incorporated into time-shifted tensor decomposition as well as the multi-shift model (described in the next section). Enforcing nonnegativity has been useful in the vast majority of practical applications that I've encountered. Thus, one may safely read this manuscript with the assumption that $u_{nr} \geq 0$, $v_{kr} \geq 0$, and $w_{tr} \geq 0$ for all $n$, $t$, $k$, and $r$, as this case is often the rule, rather than the exception.

## 1.5 Multi-Shift Model

The time-shifted tensor decomposition still assumes that the same temporal firing rate function (i.e., the temporal factors) are shared across all neurons in an ensemble, up to a temporal shift. This assumption may be too restrictive for neural dynamics that span many dimensions (Stringer et al. 2019). We can relax this assumption by replacing the low-dimensional neuron and temporal factors used in tensor decomposition ($u_{nr}$ and $w_{tr}$), with firing rate templates, as used in the vanilla time warping model ($\tilde{x}^r_{nt}$). This results in what we call the **multi-shift model**:

$$\widehat{X}_{tnk} = \sum_{r=1}^{R} v_{kr} \tilde{x}_{nr}(t + \beta_{kr}) \tag{10}$$

Using the warping matrix notation, the model can also be viewed as:

$$\widehat{\mathbf{X}}_k = \sum_{r=1}^{R} v_{kr} \mathbf{\Omega}_{kr} \tilde{\mathbf{X}}_r \tag{11}$$

This model extends the power of a vanilla shift-only time warping model by introducing multiple cell ensembles (indexed by $r$), which are independently modulated both in amplitude (by trial factors, $v_{kr}$) and in timing (by per-trial shift parameters $\beta_{kr}$). The multi-shift model contains $RNT + 2KR$ parameters, which can be considerably larger than time-shifted tensor decomposition in large-scale neural recordings. Nonetheless, since the number of total parameters grows slowly as $K$ increases, the model is still often feasible to work with in practical circumstances. Additionally, we can enforce nonnegativity constraints on $\tilde{x}^r_{nt}$ in a manner analogous to tensor decompositions (see section 1.4.1); this mitigates the potential for overfitting by limiting the expressivity of the model.

In summary, we have outlined two unsupervised learning models to model single-trial data. The first (eq. 8) closely resembles tensor decomposition, and incorporates shift-only time warping across neural and trial dimensions. The second (eq. 10) more closely resembles a time warping model, but incorporates multiple dynamical templates with per-trial amplitude modulation.

## 1.6 Choosing the number of components

Choosing the number of model components, $R$, is a challenging and well-known problem. In practice, this problem is often ill-posed in the sense that that there is rarely a "true" value of $R$, since our models are almost always *misspecified*—i.e., the data generation process does not follow the exact structure of our proposed model.

Nonetheless, there are a couple simple diagnostic tools to help practitioners decide on a reasonable guess for $R$. First, one can plot the model error as a function of $R$, and visually identify an inflection or "knee" in this monotonically decreasing curve (this visualization is called a *scree plot* in the context of PCA). A more rigorous approach is to adopt a cross-validation approach: the data is split which into separate training and testing partitions, which are respectively used for parameter fitting and model comparison. In this case, the model test error should eventually begin to increase for large enough values of $R$, indicating overfitting.

Second, one can use so-called *model stability* measures to choose an appropriate model. In this context, a model is considered "stable" if it consistently converges to the same solution across multiple optimization runs from different

random initializations. Conversely, a model is "unstable" if it converges to different solutions. Stability can be quantified by quantifying the similarity (e.g. with correlation coefficients) across optimization runs.

Stability criteria have previously been used to choose the number of components in clustering models (Luxburg 2010) and matrix factorization models (S. Wu et al. 2016), with the supposition that over-parameterized models (i.e. those with $R$ too large) are less stable than well-tuned models. Intuitively, this occurs because models with excess degrees of freedom can identify degenerate, redundant solutions, while models with fewer parameters are more constrained. Indeed, under weak assumptions the global solution of tensor decomposition is "essentially unique" (ignoring permutations and rescalings of factors) for small enough values of $R$; this uniqueness is eventually lost as $R$ increases (Kruskal 1977; Rhodes 2010).[2] In the context of exploratory data analysis, model stability is a clearly desirable feature—if multiple solutions with similar approximation error exist, then it becomes challenging to interpret and bestow meaning onto any particular solution.

Given two tensor decomposition models, with parameters $\{\mathbf{u}_r, \mathbf{v}_r, \mathbf{w}_r\}$ and $\{\mathbf{u}'_r, \mathbf{v}'_r, \mathbf{w}'_r\}$, Williams et al. (2018) used the following similarity metric to quantify stability:

$$\max_{\sigma \in \mathcal{P}} \frac{1}{R} \sum_{r=1}^{R} S(r, \sigma(r)) \tag{12}$$

where $\mathcal{P}$ is the set of all permutations of length $R$, $\sigma(\cdot)$ denotes one such permutation, and $S(i,j)$ computes the similarity between component $i$ in model 1 and component $j$ in model 2 as follows:

$$S(i,j) = \left(1 - \frac{|\lambda_i - \lambda_j|}{\max(\lambda_i, \lambda_j)}\right) \cdot \mathbf{u}_i^\mathsf{T} \mathbf{u}'_j \cdot \mathbf{v}_i^\mathsf{T} \mathbf{v}'_j \cdot \mathbf{w}_i^\mathsf{T} \mathbf{w}'_j \tag{13}$$

here $\lambda_r$ is a scale representing the norm of the $r^{\text{th}}$ component found by normalizing all factors $\mathbf{u}_r$, $\mathbf{v}_r$, $\mathbf{w}_r$ to unit length; see T. Kolda and Bader (2009). This similarity measure can be efficiently computed by enumerating an $R \times R$ matrix of pairwise similarity scores between all pairs of factors; then the optimal permutation of the factors, $\sigma$, can be found by the Hungarian algorithm in $O(R^3)$ running time (Jonker and Volgenant 1987; Burkard et al. 2012).

Equation (13) must be modified to be applicable to time-shifted tensor decomposition models. In particular, it is difficult to directly compare the temporal factors between two models due to invariances in the model structure. For example, adding a constant to the per-trial shift parameters, e.g. $\beta_{kr} \leftarrow \beta_{kr} + 1$ for all $k$, could result in very little change in the model prediction if the values in $\mathbf{w}_r$ are shifted in the opposite direction, i.e. $w_r(t) \leftarrow w_r(t+1)$ for all $t$.[3] To circumvent this problem we can compute similarity by defining:

$$\widehat{X}^{(r)}_{ntk} = u_{nr} v_{kr} w_r(t + \alpha_{nr} + \beta_{kr}) \tag{14}$$

as the contribution of component $r$ to the model's reconstruction. Then, we define the similarity between component $i$ in model 1 and component $j$ in model 2 as the norm of the residuals, $\widehat{X}^{(i)}_{ntk} - \widehat{X}^{(j)}_{ntk}$, where the first term is computed using model 1's parameters $\{\mathbf{u}_i, \mathbf{v}_i, \mathbf{w}_i, \alpha_{ni}, \beta_{ki}\}$ and the second term is computed model 2's parameters $\{\mathbf{u}'_j, \mathbf{v}'_j, \mathbf{w}'_j, \alpha'_{nj}, \beta'_{kj}\}$. Using this new definition of $S(i,j)$, which accounts for invariances introduced by the shift parameters, the optimal permutation can be computed as described above and eq. (12) can be applied to compute an overall similarity score between the two models.

## 2 Optimization with a Quadratic Loss

Here we derive methods to fit the models described in section 1 to neural data. Implementations in Python are provided at: `https://github.com/ahwillia/tensortools`

For a quadratic loss function, we can derive efficient and exact coordinate-wise updates for all model parameters, except for the shift parameters involved in time warping. These coordinate descent routines are known hierarchical alternating least squares (HALS) in the matrix and tensor factorization literature (Cichocki et al. 2007; Gillis and Glineur 2012). The key idea is to focus on updating one component at a time, which reduces the complex, nonconvex optimization problem to solving a series of easy-to-solve problems. Additionally, optimizing over single components makes it very simple to enforce nonnegativity constraints on parameter values.

[2] Note that while the global solution is provably "essentially unique" for small enough $R$, we can only guarantee that iterative optimization methods converge to a local minimum. Thus, even when the uniqueness conditions of Kruskal (1977) are met, tensor decomposition may still exhibit some instability due to convergence to distinct local minima. Nonetheless, in practice, instability tends to increase as $R$ increases, and can thus be used as a heuristic to choose the number of components.

[3] For the sake of intuition, we ignore can ignore the effect of boundary conditions where $t = 1$ and $t = T$. These boundary effects will only create small discrepancies if $T$ is large and the shifts are small.

## 2.1 Tensor Decomposition

In the case of a vanilla tensor decomposition (starting from eq. (2)) the objective function is:

$$\sum_{n,t,k}\left(X_{tnk}-\sum_{r}u_{nr}v_{kr}w_{tr}\right)^2=\sum_{n,t,k}\left(\underbrace{X_{tnk}-\sum_{r'\neq r}u_{nr'}v_{kr'}w_{tr'}}_{Z_{nktr}}-u_{nr}v_{kr}w_{tr}\right)^2 \tag{15}$$

where we have defined $Z_{nktr}$ as the residual, excluding the $r^{\text{th}}$ component of the model. If we are optimizing over any parameter in the $r^{\text{th}}$ component, then the $Z_{nktr}$ term is a constant. Now we can focus on minimizing:

$$\sum_{n,t,k}\left(Z_{nktr}-u_{nr}v_{kr}w_{tr}\right)^2 \tag{16}$$

which can be done in closed form. For example, consider optimizing over $w_{tr}$ while treating the other factors ($u_{nr}$ and $v_{kr}$) and other components (all parameters where $r'\neq r$) as fixed constants. The objective function is convex in $w_{tr}$, so there is a unique minimum, at which the gradient must be zero. A short calculation reveals this optimal value for $w_{tr}$:

$$\begin{aligned}
&\frac{\partial}{\partial w_{tr}}\sum_{n,k,t'}\left(Z_{nktr}-u_{nr}v_{kr}w_{t'r}\right)^2=0\\
\Rightarrow\quad&\sum_{n,k}\left(Z_{nktr}-u_{nr}v_{kr}w_{tr}\right)u_{nr}v_{kr}=0\\
\Rightarrow\quad&\sum_{n,k}Z_{nktr}u_{nr}v_{kr}=w_{tr}\sum_{n,k}\left(u_{nr}\right)^2\left(v_{kr}\right)^2\\
\Rightarrow\quad&w_{tr}=\frac{\sum_{n,k}Z_{nktr}u_{nr}v_{kr}}{\sum_n\left(u_{nr}\right)^2\sum_k\left(v_{kr}\right)^2}
\end{aligned} \tag{17}$$

Further manipulation of the numerator reveals that we need not form the residual tensor, $Z_{nktr}$, explicitly since:

$$\begin{aligned}
\sum_{n,k}Z_{nktr}u_{nr}v_{kr}&=\sum_{n,k}\left(X_{tnk}-\sum_{r'\neq r}u_{nr'}v_{kr'}w_{tr'}\right)u_{nr}v_{kr}\\
&=\sum_{n,k}X_{tnk}u_{nr}v_{kr}-\sum_{n,k}u_{nr}v_{kr}\sum_{r'\neq r}u_{nr'}v_{kr'}w_{tr'}\\
&=\sum_{n,k}X_{tnk}u_{nr}v_{kr}-\sum_{r'\neq r}\left(\sum_n u_{nr}u_{nr'}\sum_k v_{kr}v_{kr'}\right)
\end{aligned} \tag{18}$$

It is worthwhile to connect these derivations with the terminology and notation in T. Kolda and Bader (2009) and related works. For example, the term $\sum_{n,k}X_{tnk}u_{nr}v_{kr}$ represents a matrix multiplication between an unfolded (matricized) tensor and a Khatri-Rao product of vectors $\mathbf{u}_r$ and $\mathbf{v}_r$. To keep a streamlined narrative, we will not cover these connections in any further detail.

The tensor decomposition model we consider has a clear symmetry across the three sets of low-dimensional factors. Thus, we can easily repeat the above derivation for the other factors, $u_{nr}$ and $v_{kr}$, and arrive at the following update rules:

$$\begin{aligned}
u_{nr}&\leftarrow\frac{\sum_{k,t}X_{tnk}v_{kr}w_{tr}-\sum_{r'\neq r}\left(\sum_k v_{kr}v_{kr'}\sum_t w_{tr}w_{tr'}\right)}{\sum_k\left(v_{kr}\right)^2\sum_t\left(w_{tr}\right)^2}\\
v_{kr}&\leftarrow\frac{\sum_{n,k}X_{tnk}u_{nr}w_{tr}-\sum_{r'\neq r}\left(\sum_n u_{nr}u_{nr'}\sum_t w_{tr}w_{tr'}\right)}{\sum_n\left(u_{nr}\right)^2\sum_t\left(w_{tr}\right)^2}\\
w_{tr}&\leftarrow\frac{\sum_{n,k}X_{tnk}u_{nr}v_{kr}-\sum_{r'\neq r}\left(\sum_n u_{nr}u_{nr'}\sum_k v_{kr}v_{kr'}\right)}{\sum_n\left(u_{nr}\right)^2\sum_k\left(v_{kr}\right)^2}
\end{aligned} \tag{19}$$

In practice, we often incorporate **nonnegativity constraints** on the low-dimensional factors for the purposes of interpretability and identifiability. By appealing to the Karush–Kuhn–Tucker (KKT) conditions, one can show that incorporating

these constraints simply involves truncating any negative values to zero:

$$
\begin{aligned}
u_{nr} &\leftarrow \max\left(0, \frac{\sum_{k,t} X_{tnk} v_{kr} w_{tr} - \sum_{r' \neq r} \left(\sum_k v_{kr} v_{kr'} \sum_t w_{tr} w_{tr'}\right)}{\sum_k \left(v_{kr}\right)^2 \sum_t \left(w_{tr}\right)^2}\right) \\
v_{kr} &\leftarrow \max\left(0, \frac{\sum_{n,k} X_{tnk} u_{nr} w_{tr} - \sum_{r' \neq r} \left(\sum_n \left(u_{nr}\right)^2 \sum_t w_{tr} w_{tr'}\right)}{\sum_n \left(u_{nr}\right)^2 \sum_t \left(w_{tr}\right)^2}\right) \\
w_{tr} &\leftarrow \max\left(0, \frac{\sum_{n,k} X_{tnk} u_{nr} v_{kr} - \sum_{r' \neq r} \left(\sum_n u_{nr} u_{nr'} \sum_k v_{kr} v_{kr'}\right)}{\sum_n \left(u_{nr}\right)^2 \sum_k \left(v_{kr}\right)^2}\right)
\end{aligned}
\tag{20}
$$

It should be emphasized that these updates are **not** valid when multiple components are optimized at once—in this case, the updates can no longer be written in closed form and involve solving a quadratic program (nonnegative least-squares) problem (Gillis 2014; J. Kim et al. 2014).

## 2.2 Time Warping

Utilizing the warping matrix notation (see eq. 6), we can formulate the time warping model's objective function as:

$$
\sum_k \left\|\mathbf{X}_k - \mathbf{\Omega}_k \widetilde{\mathbf{X}}\right\|_F^2 = \sum_k \mathrm{Tr}\left[\mathbf{X}_k^\mathsf{T}\mathbf{X}_k - 2\mathbf{X}_k^\mathsf{T}\mathbf{\Omega}_k\widetilde{\mathbf{X}} + \widetilde{\mathbf{X}}_k^\mathsf{T}\mathbf{\Omega}_k^\mathsf{T}\mathbf{\Omega}_k\widetilde{\mathbf{X}}_k\right] \tag{21}
$$

We adopt a block-coordinate descent approach, first treating the warping matrices, $\mathbf{\Omega}_k$, as fixed and optimizing $\widetilde{\mathbf{X}}$. As before, we can find a closed form update by identifying where the gradient is zero:

$$
\begin{aligned}
&\sum_k \left[-2\mathbf{\Omega}_k^\mathsf{T}\mathbf{X}_k + 2\mathbf{\Omega}_k^\mathsf{T}\mathbf{\Omega}_k\widetilde{\mathbf{X}}_k\right] = 0 \\
\Rightarrow \quad &\sum_k \mathbf{\Omega}_k^\mathsf{T}\mathbf{\Omega}_k\widetilde{\mathbf{X}}_k = \sum_k \mathbf{\Omega}_k^\mathsf{T}\mathbf{X}_k \\
\Rightarrow \quad &\widetilde{\mathbf{X}} = \left(\sum_k \mathbf{\Omega}_k^\mathsf{T}\mathbf{\Omega}_k\right)^{-1} \sum_k \mathbf{\Omega}_k^\mathsf{T}\mathbf{X}_k
\end{aligned}
\tag{22}
$$

This update is very efficient to compute, as it can be shown that each $\mathbf{\Omega}_k^\mathsf{T}\mathbf{\Omega}_k$ is symmetric and tridiagonal (see Williams et al. 2019 for details). In python, such linear systems can be efficiently solved by the `scipy.linalg.solveh_banded` function. Note that there is no reason to enforce nonnegativity on the warping template $\widetilde{\mathbf{X}}$ as it will naturally be nonnegative for nonnegative data.

After updating the response templates according to eq. (22) we must then update the per-trial warping functions. Unfortunately, this can be quite complicated. The optimal *nonlinear warping* can be found by the celebrated *Dynamic Time Warping* algorithm (Berndt and Clifford 1994). Such nonlinear warping paths can be very complex and are thus susceptible to overfit to noise, prompting ongoing work on regularized and smoothed time warping functions (Cuturi and Blondel 2017; Duncker and Sahani 2018). In Williams et al. (2019), we propose a simple approach: we constrain the warping functions to be piecewise linear and specified by a small number of learned parameters. In the simplest case of *shift-only warping* we learn only a single shift parameter on each trial. This can be optimized by a brute force search, done in parallel over all trials.

## 2.3 Time-Shifted Tensor Decomposition

Utilizing warping matrices, we can formulate the model's objective function as:

$$
\sum_{nk} \left\|\mathbf{x}_{nk} - \sum_{r=1}^{R} u_{nr} v_{kr} \mathbf{\Omega}_{nkr} \mathbf{w}_r\right\|_2^2 \tag{23}
$$

We follow a similar strategies to those outlined in sections 2.1 and 2.2. First, to optimize the warping functions, we perform a randomized searches as described in section 2.2. Recall from eqs. (6) and (8) that we assume the warping functions to

correspond to per-neuron shifts, $\alpha_{nr}$, and per-trial shifts, $\beta_{kr}$. Due to interactions between these two sets of parameters, it is not easy to simultaneously optimize $\alpha_{nr}$ and $\beta_{kr}$ in parallel. However, we can perform update these parameters in two blocks—first over neurons, then over trials—each done parallel.

To optimize the remaining parameters (the low-rank neural, temporal and trial trial factors), we follow the approach of section 2.1 and consider optimizing a single component at a time. Denote the residual for neuron $n$ on trial $k$ as $\mathbf{z}_{nkr} \in \mathbb{R}^T$; that is:

$$\mathbf{z}_{nkr} = \mathbf{x}_{nk} - \sum_{r' \neq r} u_{nr'} v_{kr'} \boldsymbol{\Omega}_{nkr'} \mathbf{w}_{r'} \tag{24}$$

The objective function with respect to the $r^{\text{th}}$ component can be expressed as:

$$\sum_{nk} \left\| \mathbf{z}_{nkr} - u_{nr} v_{kr} \boldsymbol{\Omega}_{nkr} \mathbf{w}_r \right\|_2^2 \;=\; \sum_{nk} (\mathbf{z}_{nkr}^\mathsf{T} \mathbf{z}_{nkr} - 2 u_{nr} v_{kr} \mathbf{z}_{nkr}^\mathsf{T} \boldsymbol{\Omega}_{nkr} \mathbf{w}_r + u_{nr}^2 v_{kr}^2 \mathbf{w}_r^\mathsf{T} \boldsymbol{\Omega}_{nkr}^\mathsf{T} \boldsymbol{\Omega}_{nkr} \mathbf{w}_r) \tag{25}$$

As before, closed form parameter updates can be derived for the neural, temporal, and trial factor by identifying where the gradient is zero.

For the temporal factor, $\mathbf{w}_r$, we obtain:

$$\begin{aligned}
&\nabla_{\mathbf{w}^r} \left[ \sum_{nk} (\mathbf{z}_{nkr}^\mathsf{T} \mathbf{z}_{nkr} - 2 u_{nr} v_{kr} \mathbf{z}_{nkr}^\mathsf{T} \boldsymbol{\Omega}_{nkr} \mathbf{w}_r + u_{nr}^2 v_{kr}^2 \mathbf{w}_r^\mathsf{T} \boldsymbol{\Omega}_{nkr}^\mathsf{T} \boldsymbol{\Omega}_{nkr} \mathbf{w}_r) \right] = 0 \\
\Rightarrow \quad & \sum_{nk} (-2 u_{nr} v_{kr} \boldsymbol{\Omega}_{nkr}^\mathsf{T} \mathbf{z}_{nkr} + 2 u_{nr}^2 v_{kr}^2 \boldsymbol{\Omega}_{nkr}^\mathsf{T} \boldsymbol{\Omega}_{nkr} \mathbf{w}_r) = 0 \\
\Rightarrow \quad & \sum_{nk} u_{nr}^2 v_{kr}^2 \boldsymbol{\Omega}_{nkr}^\mathsf{T} \boldsymbol{\Omega}_{nkr} \mathbf{w}_r = \sum_{nk} u_{nr} v_{kr} \boldsymbol{\Omega}_{nkr}^\mathsf{T} \mathbf{z}_{nkr} \\
\Rightarrow \quad & \mathbf{w}_r = \left( \sum_{nk} u_{nr}^2 v_{kr}^2 \boldsymbol{\Omega}_{nkr}^\mathsf{T} \boldsymbol{\Omega}_{nkr} \right)^{-1} \sum_{nk} u_{nr} v_{kr} \boldsymbol{\Omega}_{nkr}^\mathsf{T} \mathbf{z}_{nkr}
\end{aligned} \tag{26}$$

which can be viewed as a re-weighted version of eq. (22).

For the trial factor, $\mathbf{v}_r$, we obtain:

$$\begin{aligned}
&\frac{\partial}{\partial v_{kr}} \left[ \sum_{nk'} (\mathbf{z}_{nk'r}^\mathsf{T} \mathbf{z}_{nk'r} - 2 u_{nr} v_{k'r} \mathbf{z}_{nk'r}^\mathsf{T} \boldsymbol{\Omega}_{nk'r} \mathbf{w}_r + u_{nr}^2 v_{k'r}^2 \mathbf{w}_r^\mathsf{T} \boldsymbol{\Omega}_{nk'r}^\mathsf{T} \boldsymbol{\Omega}_{nk'r} \mathbf{w}_r) \right] = 0 \\
\Rightarrow \quad & \sum_{n} (-2 u_{nr} \mathbf{z}_{nkr}^\mathsf{T} \boldsymbol{\Omega}_{nkr} \mathbf{w}_r + 2 u_{nr}^2 v_{kr} \mathbf{w}_r^\mathsf{T} \boldsymbol{\Omega}_{nkr}^\mathsf{T} \boldsymbol{\Omega}_{nkr} \mathbf{w}_r) = 0 \\
\Rightarrow \quad & v_{kr} \sum_{n} u_{nr}^2 \mathbf{w}_r^\mathsf{T} \boldsymbol{\Omega}_{nkr}^\mathsf{T} \boldsymbol{\Omega}_{nkr} \mathbf{w}_r = \sum_{n} u_{nr} \mathbf{z}_{nkr}^\mathsf{T} \boldsymbol{\Omega}_{nkr} \mathbf{w}_r \\
\Rightarrow \quad & v_{kr} = \frac{\sum_n u_{nr} \mathbf{z}_{nkr}^\mathsf{T} \boldsymbol{\Omega}_{nkr} \mathbf{w}_r}{\sum_n u_{nr}^2 \mathbf{w}_r^\mathsf{T} \boldsymbol{\Omega}_{nkr}^\mathsf{T} \boldsymbol{\Omega}_{nkr} \mathbf{w}_r}
\end{aligned} \tag{27}$$

Deriving an analogous update rule for the neuron factors follows a nearly identical series of computations, ultimately obtaining:

$$u_{nr} = \frac{\sum_k v_{kr} \mathbf{z}_{nkr}^\mathsf{T} \boldsymbol{\Omega}_{nkr} \mathbf{w}_r}{\sum_k v_{kr}^2 \mathbf{w}_r^\mathsf{T} \boldsymbol{\Omega}_{nkr}^\mathsf{T} \boldsymbol{\Omega}_{nkr} \mathbf{w}_r} \tag{28}$$

In practice, we find that it is beneficial to constrain the neural and trial factors to be nonnegative. Enforcing this constraint is simple and can be done in analogy to eq. (20). Enforcing nonnegativity on the temporal factors is a bit more challenging, since the corresponding update rule (derived in eq. 26) involves multiple variables at once. As discussed in Gillis (2014) and J. Kim et al. (2014), a more appropriate update rule must use *nonnegative least squares* methods. This could be accomplished by projected gradient descent, as we'll see in the following section. However, in practice we have found that is not typically necessary to force the temporal factors to be nonnegative—the model is already sufficiently constrained by the nonnegativity constraints on the neural and trial factors.

## 2.4 Multi-Shift Model

Optimizing the multi-shift model can also be accomplished by coordinate descent approach. First, define the residual matrix $\mathbf{Z}_{kr} \in \mathbb{R}^{T \times N}$ as:

$$\mathbf{Z}_{kr} = \mathbf{X}_k - \sum_{r' \neq r} v_{kr'} \mathbf{\Omega}_{kr'} \widetilde{\mathbf{X}}_{r'} \tag{29}$$

for trial $k$. Then, to update component $r$, we can focus on minimizing the objective function:

$$\sum_k \|\mathbf{Z}_{kr} - v_{kr} \mathbf{\Omega}_{kr} \widetilde{\mathbf{X}}_r\|_F^2 \tag{30}$$

Again, we resort to a randomized search over per-trial shift parameters to optimize $\mathbf{\Omega}_{kr}$. We have found it useful to constrain both the trial factors, $v_{kr}$, and response templates, $\widetilde{\mathbf{X}}_r$, to be nonnegative. Updating $v_{kr}$ under this constraint can be done in closed form, but we resort to a fast projected gradient descent routine to update the response templates.

The update rule for $v_{kr}$ is found by identifying where the gradient is zero:

$$\begin{aligned}
&\frac{\partial}{\partial v_{kr}} \|\mathbf{Z}_k - v_{kr} \mathbf{\Omega}_{kr} \widetilde{\mathbf{X}}_r\|_F^2 = 0 \\
\Rightarrow \quad & \frac{\partial}{\partial v_{kr}} \left[ \operatorname{Tr}\left[\mathbf{Z}_k^\mathsf{T} \mathbf{Z}_k\right] - 2 v_{kr} \operatorname{Tr}[\mathbf{Z}_{kr}^\mathsf{T} \mathbf{\Omega}_{kr} \widetilde{\mathbf{X}}_r] + v_{kr}^2 \operatorname{Tr}[\widetilde{\mathbf{X}}_r^\mathsf{T} \mathbf{\Omega}_{kr}^\mathsf{T} \mathbf{\Omega}_{kr} \widetilde{\mathbf{X}}_r] \right] = 0 \\
\Rightarrow \quad & -2 \operatorname{Tr}[\mathbf{Z}_{kr}^\mathsf{T} \mathbf{\Omega}_{kr} \widetilde{\mathbf{X}}_r] + 2 v_{kr} \operatorname{Tr}[\widetilde{\mathbf{X}}_r^\mathsf{T} \mathbf{\Omega}_{kr}^\mathsf{T} \mathbf{\Omega}_{kr} \widetilde{\mathbf{X}}_r] = 0 \\
\Rightarrow \quad & v_{kr} = \frac{\operatorname{Tr}[\mathbf{Z}_{kr}^\mathsf{T} \mathbf{\Omega}_{kr} \widetilde{\mathbf{X}}_r]}{\operatorname{Tr}[\widetilde{\mathbf{X}}_r^\mathsf{T} \mathbf{\Omega}_{kr}^\mathsf{T} \mathbf{\Omega}_{kr} \widetilde{\mathbf{X}}_r]}
\end{aligned} \tag{31}$$

Since we are optimizing over a single variable, it is valid to simply truncate negative values to enforce $v_{kr} \geq 0$. In summary, the update rule, which can be computed in parallel across trials, is:

$$v_{kr} \leftarrow \max\left(0, \frac{\operatorname{Tr}[\mathbf{Z}_{kr}^\mathsf{T} \mathbf{\Omega}_{kr} \widetilde{\mathbf{X}}_r]}{\operatorname{Tr}[\widetilde{\mathbf{X}}_r^\mathsf{T} \mathbf{\Omega}_{kr}^\mathsf{T} \mathbf{\Omega}_{kr} \widetilde{\mathbf{X}}_r]}\right) \tag{32}$$

Finally, we must derive an update rule for the response templates, $\widetilde{\mathbf{X}}_r$. We begin by deriving the gradient:

$$\begin{aligned}
&\nabla_{\widetilde{\mathbf{X}}_r} \left[ \sum_k \operatorname{Tr}\left[\mathbf{Z}_k^\mathsf{T} \mathbf{Z}_k\right] - 2 v_{kr} \operatorname{Tr}[\mathbf{Z}_{kr}^\mathsf{T} \mathbf{\Omega}_{kr} \widetilde{\mathbf{X}}_r] + v_{kr}^2 \operatorname{Tr}[\widetilde{\mathbf{X}}_r^\mathsf{T} \mathbf{\Omega}_{kr}^\mathsf{T} \mathbf{\Omega}_{kr} \widetilde{\mathbf{X}}_r] \right] \\
&= 2 \sum_k \left( -v_{kr} \mathbf{\Omega}_{kr}^\mathsf{T} \mathbf{Z}_{kr} + v_{kr}^2 \mathbf{\Omega}_{kr}^\mathsf{T} \mathbf{\Omega}_{kr} \widetilde{\mathbf{X}}_r \right) \\
&= 2 \sum_k v_{kr} \mathbf{\Omega}_{kr}^\mathsf{T} \left( v_{kr} \mathbf{\Omega}_{kr} \widetilde{\mathbf{X}}_r - \mathbf{Z}_{kr} \right)
\end{aligned} \tag{33}$$

By differentiating once more can find the Hessian matrix with respect to $\widetilde{\mathbf{X}}_r$. The Hessian can be viewed as a $NT \times NT$ block diagonal matrix with $\mathbf{B} = 2 \sum_k v_{kr}^2 \mathbf{\Omega}_{kr}^\mathsf{T} \mathbf{\Omega}_{kr}$ repeated $N$ times as the blocks.[4] Since the objective function is quadratic, the gradient is globally Lipschitz continuous with constant $L = 2\|\mathbf{B}\|_2$, with $\|\cdot\|_2$ denoting the largest eigenvalue of a matrix (operator norm). Standard convex optimization theory tells us that projected gradient descent will converge as long as the step size is less than this Lipschitz constant (Boyd and Vandenberghe 2004). We can compute a very tight upper bound on this maximum eigenvalue very cheaply via the Gershgorin circle theorem. Recall that $\mathbf{B}$ is tridiagonal, nonnegative, and symmetric, due to the structure of the warping matrices. Thus, the maximum eigenvalue must be a real number and be less than $\gamma = \max(\text{diag}(\mathbf{B}))) + 2\max(\text{offdiag}(\mathbf{B}))$. We set the stepsize of gradient descent to be the inverse of this upper bound, assuring fast convergence.[5] In summary, we update $\widetilde{\mathbf{X}}_r$ according to:

$$\widetilde{\mathbf{X}}_r \leftarrow \max\left(0, \widetilde{\mathbf{X}}_r - \frac{1}{\gamma} \nabla_{\widetilde{\mathbf{X}}_r}\right) \tag{34}$$

[4] In eq. (33) we treated the gradient as $T \times N$ matrix, and thus by common convention we would treat the Hessian as a order-4 tensor. However, if we view the gradient as a vectorized version of eq. (33), then the Hessian would be an $NT \times NT$ matrix as described.

[5] The bound on the maximum eigenvalue is not tight and it is very easy to derive tighter approximations using the Gershgorin circle theorem. Additionally, we can use specialized eigenvalue solvers such as `scipy.linalg.eigh_tridiagonal` to quickly compute the Lipschitz constant exactly. Nonetheless, the upper bound on $\gamma$ listed here is very cheap to compute and has been sufficient in practice.

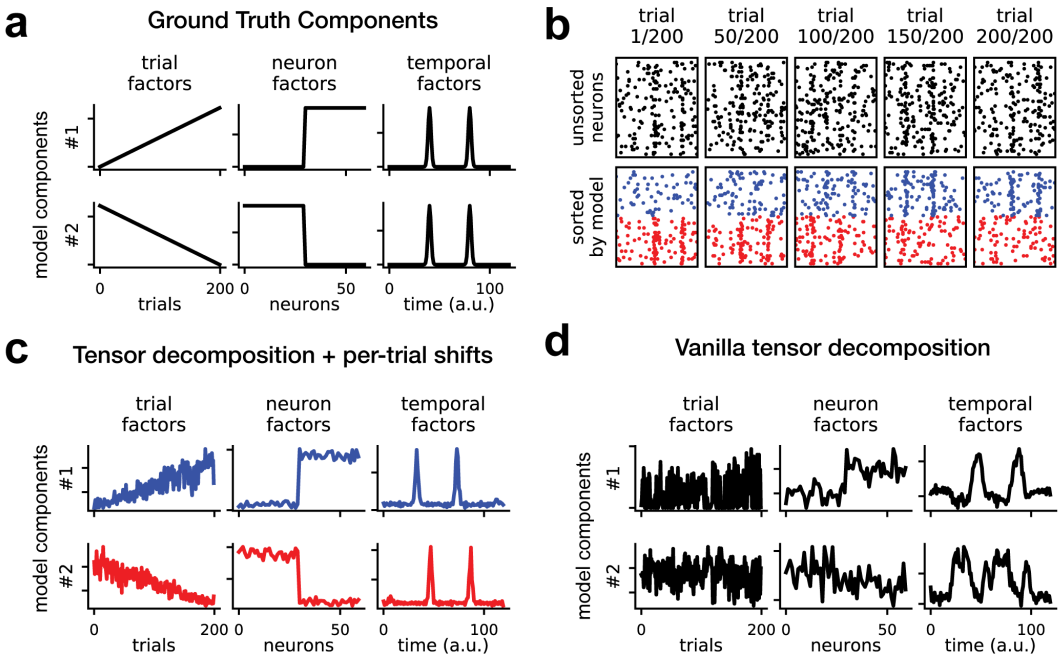


Figure 1: **Synthetic spiking activity of two neural populations with random per-trial latencies.** The ground truth spiking probabilities follow eq. (8) with $\alpha_{nr} = 0$ for each neuron $n$ and component $r$, and $\beta_{kr}$ randomized $\pm 15\%$ of the trial duration for each trial $k$ and component $r$. (A) Ground truth factors for simulated data. The $\beta_{kr}$ delay parameters are not visualized. (B) Simulated spike data of $N = 60$ neurons over $K = 200$ trials. Spike counts were sampled from a Poisson distribution, with time-varying firing rate function given by the model in panel A. Top row (black spikes) shows trials 1, 50, 100, 150, 200 with neurons shown in a randomized order. Bottom row (blue and red spikes) shows the same data with neurons grouped according to the learned neuron factors. The first component (in blue) grows in amplitude over the course of the simulated session, while the second component (in red) decreases in amplitude over the course of the session. (C) Low-dimensional factors recovered by the shifted tensor decomposition model (the learned $\beta_{kr}$ parameters are not visualized). The factors are colored to match panel B. (D) A tensor decomposition model without time warping fails to recover the ground-truth structure of the model.

with the gradient term computed according to eq. (33).

# 3 Demonstrations on Synthetic Data

As mentioned at the beginning of section 2.2, tensor decomposition can fail to uncover the intended low-dimensional structure of data when the timing of neural dynamics is variable. Figure 1 demonstrates this in a simulated dataset with $N = 60$ neurons, $T = 120$ timebins, and $K = 200$ trials. Poisson-distributed binned spike counts were simulated according to the 2-component ground truth model shown in Figure 1a, plus random, per-trial time-shifts $\pm 15\%$ of the full trial duration. The maximum Poisson rate parameter in any timebin was less than 0.3, so that the simulated spike patterns were highly sparse and divergent from the Gaussian likelihood/quadratic loss criterion associated with the models (see eq. (1)). The minimum Poisson rate parameter in any timebin was 0.02, so that some additional spikes were present as "background noise."

The factors in Figure 1a can be interpreted as follows. First, as indicated by the neuron factors (middle column), there are two non-overlapping sub-populations or ensembles of cells. Second, as indicated by the temporal factors (right column), each ensemble fires twice within every trial—the responses are sharp and only last for a short time interval, making them somewhat difficult to detect, especially in the presence of trial-to-trial variation in response onset. Finally, as indicated by the trial factors (left column), the first neural ensemble grows linearly in amplitude over trials, while the second neural ensemble diminishes linearly over trials.

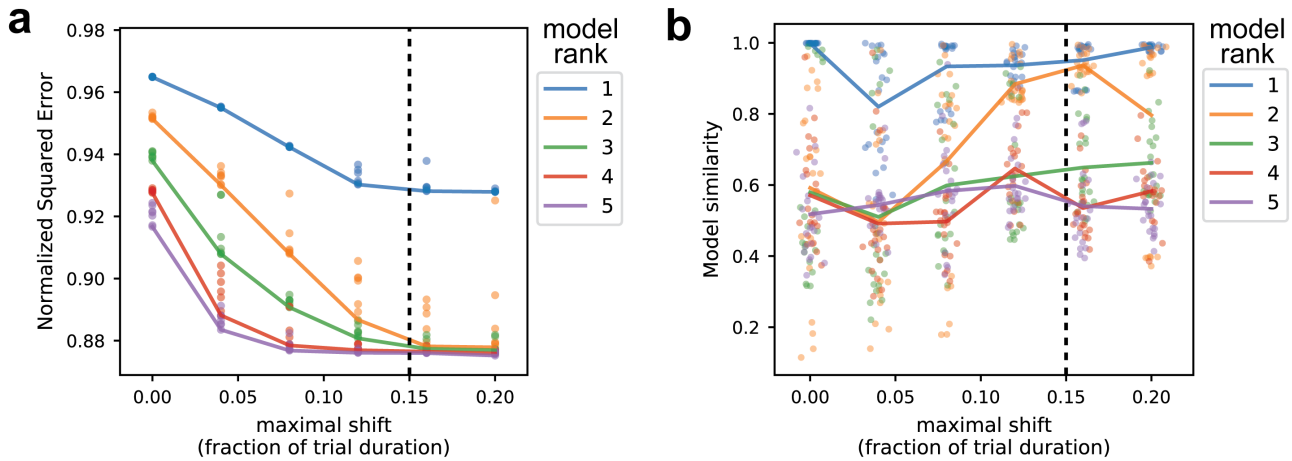


Figure 2: **Hyperparameter sweep for time-shifted tensor decomposition.** Colored lines denote models with different numbers of components (i.e., $R$, the rank of the tensor decomposition), while the horizontal axis measures the maximal per-trial shift, $\beta_{\text{max}}$. The ground truth model had $R = 2$ components (yellow line) and $\beta_{\text{max}} = 0.15$ (vertical dashed black line). For simplicity, no per-neuron shifts ($\alpha_{nr}$ in eq. (8)) were included in the model. (A) Normalized reconstruction error, $\|\mathbf{X} - \widehat{\mathbf{X}}\|/\|\mathbf{X}\|$, of all models. (B) Model similarity scores (see section 1.6) across different random initializations of the same hyperparameter set.

Figure 1b shows the activity of all neurons in five trials equally spaced across the full session of $K = 200$ trials. In the top row of raster plots, each of the $N = 60$ neurons are assigned to a random y-coordinate to simulate the appearance of raw experimental data. It is very hard to visually identify recurring temporal patterns within trials and longer-term changes in population activity across trials. The bottom row of raster plots shows the same data, but with neurons re-sorted and colored by the low-dimensional neuron factors identified by time-shifted tensor decomposition in Figure 1c. The model successfully groups neurons into their two ground truth ensembles, enabling one to visually identify the double banded structure in each ensemble's response pattern and the fact that ensemble #1 (in blue) intensifies over the session while ensemble #2 (in red) decreases in amplitude. The trial-to-trial variability in response timing is also visible upon close inspection of these re-organized raster plots. Overall, the time-shifted tensor decomposition model captures the correct qualitative structure of the simulated dataset (compare fig. 1a and fig. 1c). In contrast, a classic tensor decomposition model (see eq. (2)) fails to identify any of this interpretable structure in the neural data (fig. 1d); this is due entirely to the variability in neural response times.

Next, we performed a hyperparameter sweep to determine whether the diagnostic tools described in section 1.6 could successfully identify the number of components and the scale of the per-trial shifts in a similar synthetic dataset.[6] We fit models with $R \in \{1, 2, 3, 4, 5\}$. Further, we set a maximum absolute value to the per-trial shift parameters, $\beta_{kr}$; expressed as a fraction of the total trial duration, these limits were $\beta_{\text{max}} \in \{0, 0.04, 0.08, 0.12, 0.16, 0.2\}$. Note that $\beta_{\text{max}} = 0$ corresponds to traditional tensor decomposition. For each combination of $R$ and $\beta_{\text{max}}$ we fit seven models from different random initializations. Figure 2a shows that a model with $R = 2$ components (yellow line) with $\beta_{\text{max}} = 0.16$ closely matches the lowest error achieved by any model; the hyperparameters of this model were the closest to the ground truth (which has $R = 2$ and $\beta_{\text{max}} = 0.15$). Furthermore, when $\beta_{\text{max}}$ was set lower than its ground truth value, even models with additional components ($R > 2$) often performed suboptimally. Note, however, that for traditional tensor decomposition (i.e., $\beta_{\text{max}} = 0$) the model with $R = 5$ components noticeably outperforms models with fewer components. Thus, the presence of per-trial temporal shifts can cause us to mistakenly believe that high-rank tensor model is needed; introducing shift parameters into the tensor decomposition model reveals that $R = 2$ components is necessary, resulting in a simpler and much more interpretable model with similar (or even superior) performance.

Figure 2b shows the model stability criterion (see section 1.6) over the explored hyperparameter range. Every dot in fig. 2b

[6]The signal-to-noise ratio of the simulated data in fig. 1 is near the threshold of detectability. To achieve clear results in fig. 2 we increased the width of the peaks in the temporal factors by a factor of 2. An interesting avenue of future work would be to more rigorously define the conditions (e.g. the minimal signal-to-noise ratio) under which time-shifted tensor decomposition models can be reliably fit; Kadmon and Ganguli (2018) study this problem for the classic tensor decomposition model.

corresponds to the similarity score (ranging between zero and one) between a pair of models with the same hyperparameters. Models with $R = 1$ components were the most stable; however, these models had suboptimal performance as already shown in fig. 2a. Models with too many components ($R > 2$) were consistently unstable. Interestingly, models with the correct number of components ($R = 2$) was not always stable—if the maximal per-trial shift parameter was less than the ground truth value, the similarity between model fits was consistently low. The model closest to the ground truth ($R = 2$ and $\beta_{\mathsf{max}} = 0.16$) demonstrated relatively high stability, which, in conjunction with the results in fig. 2a, provides strong evidence for its superiority.

# 4 Conclusion and Future Work

Taken individually, tensor decomposition and time warping can be broadly applicable and insightful methods for neural data analysis (Williams et al. 2018; Williams et al. 2019). Nonetheless, there may be cases where these two models are insufficient. Neural ensemble activity may be shifted across neurons or on a trial-by-trial basis, hampering the interpretability of tensor decomposition. On the other hand, the time warping model described in Williams et al. (2019) assumes that all neurons shared the same trial-by-trial warping function and have a single canonical response pattern. We described how to combine the complementary strengths of these two models into two new models: time-shifted tensor decomposition, and a multi-shift model. Previous work has explored similar modeling ideas, but with different proposed algorithms and motivating applications (Harshman et al. 2003; Mørup et al. 2008). The unified treatment of these hybrid models provided here should clarify the conceptual connections between these previous works. Additionally, we provided an open source Python implementation for these shifted tensor decomposition models at: `https://github.com/ahwillia/tensortools`


## Acknowledgements

I wish to thank Tammy Kolda (Sandia National Labs) for introducing me to tensor decomposition methods and Surya Ganguli (Stanford) for helping me relate these ideas to a neuroscience audience. Jordan Sorokin (Stanford) helped me develop and beta-test early versions of the time-shifted tensor decomposition code. Jordan Sorokin, John Huguenard, and Isabel Low (all Stanford) provided experimental data and feedback that helped me develop these models—though no experimental data is discussed in these notes, their input nonetheless helped form and motivate the data analysis approaches described here. I also wish to thank the U.S. Department of Energy CSGF program for supporting my PhD research on these topics.


# References


Berndt, Donald J and James Clifford (1994). "Using dynamic time warping to find patterns in time series". In: *KDD workshop*. Vol. 10. aaai.org, pp. 359–370.

Boyd, Stephen and Lieven Vandenberghe (2004). *Convex optimization*. Cambridge University Press.

Burkard, Rainer, Mauro Dell'Amico, and Silvano Martello (2012). *Assignment Problems, Revised Reprint*. en. SIAM.

Carandini, Matteo and David J Heeger (2011). "Normalization as a canonical neural computation". en. In: *Nat. Rev. Neurosci.* 13.1, pp. 51–62.

Carroll, J Douglas and Jih-Jie Chang (1970). "Analysis of individual differences in multidimensional scaling via an n-way generalization of "Eckart-Young" decomposition". In: *Psychometrika* 35.3, pp. 283–319.

Chen, Xiuye, Yu Mu, Yu Hu, Aaron T. Kuan, Maxim Nikitchenko, Owen Randlett, Alex B. Chen, Jeffery P. Gavornik, Haim Sompolinsky, Florian Engert, and Misha B. Ahrens (2018). "Brain-wide Organization of Neuronal Activity and Convergent Sensorimotor Transformations in Larval Zebrafish". In: *Neuron* 100.4, 876–890.e5.

Chi, Eric C and Tamara G Kolda (2012). "On Tensors, Sparsity, and Nonnegative Factorizations". In: *SIAM J. Matrix Anal. Appl.* 33.4, pp. 1272–1299.

Churchland, Mark M, Byron M Yu, Maneesh Sahani, and Krishna V Shenoy (2007). "Techniques for extracting single-trial activity patterns from large-scale neural recordings". en. In: *Curr. Opin. Neurobiol.* 17.5, pp. 609–618.

Cichocki, Andrzej, Rafal Zdunek, and Shun-Ichi Amari (2007). "Hierarchical ALS Algorithms for Nonnegative Matrix and 3D Tensor Factorization". In: *Independent Component Analysis and Signal Separation*. Springer Berlin Heidelberg, pp. 169–176.

Cuturi, Marco and Mathieu Blondel (2017). "Soft-DTW: A Differentiable Loss Function for Time-series". In: *Proceedings of the 34th International Conference on Machine Learning - Volume 70*. ICML'17. Sydney, NSW, Australia: JMLR.org, pp. 894–903.

Duncker, Lea and Maneesh Sahani (2018). "Temporal alignment and latent Gaussian process factor inference in population spike trains". In: *Advances in Neural Information Processing Systems 31*. Ed. by S Bengio, H Wallach, H Larochelle, K Grauman, N Cesa-Bianchi, and R Garnett. Curran Associates, Inc., pp. 10445–10455.

Gillis, Nicolas (2014). "The why and how of nonnegative matrix factorization". In: *Regularization, Optimization, Kernels, and Support Vector Machines* 12.257, pp. 257–291.

Gillis, Nicolas and François Glineur (2012). "Accelerated multiplicative updates and hierarchical ALS algorithms for nonnegative matrix factorization". en. In: *Neural Comput.* 24.4, pp. 1085–1105.

Goris, Robbe L T, J Anthony Movshon, and Eero P Simoncelli (2014). "Partitioning neuronal variability". en. In: *Nat. Neurosci.* 17.6, pp. 858–865.

Harshman, Richard A, Sungjin Hong, and Margaret E Lundy (2003). "Shifted factor analysis?Part I: Models and properties". In: *J. Chemom.* 17.7, pp. 363–378.

Hong, David, Tamara G Kolda, and Jed A Duersch (2018). "Generalized Canonical Polyadic Tensor Decomposition". In:

Hong, Sungjin (2009). "Warped factor analysis". In: *J. Chemom.* 23.7-8, pp. 371–384.

Hong, Sungjin and Richard A Harshman (2003). "Shifted factor analysis—Part II: Algorithms". In: *J. Chemom.* 17.7, pp. 379–388.

Jonker, R and A Volgenant (1987). "A shortest augmenting path algorithm for dense and sparse linear assignment problems". In: *Computing* 38.4, pp. 325–340.

Kadmon, Jonathan and Surya Ganguli (2018). "Statistical mechanics of low-rank tensor decomposition". In: *Advances in Neural Information Processing Systems 31*. Ed. by S Bengio, H Wallach, H Larochelle, K Grauman, N Cesa-Bianchi, and R Garnett. Curran Associates, Inc., pp. 8201–8212.

Kim, Jingu, Yunlong He, and Haesun Park (2014). "Algorithms for nonnegative matrix and tensor factorizations: a unified view based on block coordinate descent framework". In: *J. Global Optimiz.* 58.2, pp. 285–319.

Kim, Tony Hyun, Yanping Zhang, Jérôme Lecoq, Juergen C Jung, Jane Li, Hongkui Zeng, Cristopher M Niell, and Mark J Schnitzer (2016). "Long-Term Optical Access to an Estimated One Million Neurons in the Live Mouse Cortex". In: *Cell Reports* 17.12, pp. 3385–3394.

Kolda, T and B Bader (2009). "Tensor Decompositions and Applications". In: *SIAM Rev.* 51.3, pp. 455–500.

Kruskal, Joseph B (1977). "Three-way arrays: rank and uniqueness of trilinear decompositions, with application to arithmetic complexity and statistics". In: *Linear Algebra Appl.* 18.2, pp. 95–138.

Lee, D D and H S Seung (1999). "Learning the parts of objects by non-negative matrix factorization". en. In: *Nature* 401.6755, pp. 788–791.

Lim, Lek-Heng and Pierre Comon (2009). "Nonnegative approximations of nonnegative tensors". In: *J. Chemom.* 23.7-8, pp. 432–441.

Luxburg, Ulrike von (2010). "Clustering Stability: An Overview". In: *Foundations and Trends® in Machine Learning* 2.3, pp. 235–274.

Mackevicius, Emily L, Andrew H Bahle, Alex H Williams, Shijie Gu, Natalia I Denisenko, Mark S Goldman, and Michale S Fee (2019). "Unsupervised discovery of temporal sequences in high-dimensional datasets, with applications to neuroscience". en. In: *Elife* 8.

Mørup, Morten, Lars Kai Hansen, Sidse Marie Arnfred, Lek-Heng Lim, and Kristoffer Hougaard Madsen (2008). "Shift-invariant multilinear decomposition of neuroimaging data". en. In: *Neuroimage* 42.4, pp. 1439–1450.

Paatero, Pentti (1997). "A weighted non-negative least squares algorithm for three-way 'PARAFAC' factor analysis". In: *Chemometrics Intellig. Lab. Syst.* 38.2, pp. 223–242.

Pandarinath, Chethan, Daniel J O'Shea, Jasmine Collins, Rafal Jozefowicz, Sergey D Stavisky, Jonathan C Kao, Eric M Trautmann, Matthew T Kaufman, Stephen I Ryu, Leigh R Hochberg, Jaimie M Henderson, Krishna V Shenoy, L F Abbott, and David Sussillo (2018). "Inferring single-trial neural population dynamics using sequential auto-encoders". en. In: *Nat. Methods* 15.10, pp. 805–815.

Paninski, Liam (2004). "Maximum likelihood estimation of cascade point-process neural encoding models". en. In: *Network* 15.4, pp. 243–262.

Petreska, Biljana, Byron M Yu, John P Cunningham, Gopal Santhanam, Stephen I Ryu, Krishna V Shenoy, and Maneesh Sahani (2011). "Dynamical segmentation of single trials from population neural data". In: *Advances in Neural Information Processing Systems 24*. Ed. by J Shawe-Taylor, R S Zemel, P L Bartlett, F Pereira, and K Q Weinberger. Curran Associates, Inc., pp. 756–764.

Rabinowitz, Neil C, Robbe L Goris, Marlene Cohen, and Eero P Simoncelli (2015). "Attention stabilizes the shared gain of V4 populations". en. In: *Elife* 4, e08998.

Rhodes, John A (2010). "A concise proof of Kruskal's theorem on tensor decomposition". In: *Linear Algebra Appl.* 432.7, pp. 1818–1824.

Salinas, E and P Thier (2000). "Gain modulation: a major computational principle of the central nervous system". In: *Neuron* 27.1, pp. 15–21.

Stringer, Carsen, Marius Pachitariu, Nicholas Steinmetz, Charu Bai Reddy, Matteo Carandini, and Kenneth D. Harris (2018). "Spontaneous behaviors drive multidimensional, brain-wide population activity". In: *bioRxiv*.

Stringer, Carsen, Marius Pachitariu, Nicholas Steinmetz, Matteo Carandini, and Kenneth D Harris (2019). "High-dimensional geometry of population responses in visual cortex". en. In: *Nature* 571.7765, pp. 361–365.

Williams, Alex H, Tony Hyun Kim, Forea Wang, Saurabh Vyas, Stephen I Ryu, Krishna V Shenoy, Mark Schnitzer, Tamara G Kolda, and Surya Ganguli (2018). "Unsupervised Discovery of Demixed, Low-Dimensional Neural Dynamics across Multiple Timescales through Tensor Component Analysis". en. In: *Neuron* 98.6, 1099–1115.e8.

Williams, Alex H, Ben Poole, Niru Maheswaranathan, Ashesh K Dhawale, Tucker Fisher, Christopher D Wilson, David H Brann, Eric Trautmann, Stephen Ryu, Roman Shusterman, Dmitry Rinberg, Bence P Ölveczky, Krishna V Shenoy, and Surya Ganguli (2019). "Discovering precise temporal patterns in large-scale neural recordings through robust and interpretable time warping". en.

Wright, Stephen J (2015). "Coordinate descent algorithms". In: *Math. Program.* 151.1, pp. 3–34.

Wu, Q., J. Liu, F. Sun, J. Li, and A. Cichocki (2014). "Nonnegative Shifted Tensor Factorization in time frequency domain". In: *2014 International Joint Conference on Neural Networks (IJCNN)*, pp. 3009–3014.

Wu, Siqi, Antony Joseph, Ann S Hammonds, Susan E Celniker, Bin Yu, and Erwin Frise (2016). "Stability-driven nonnegative matrix factorization to interpret spatial gene expression and build local gene networks". en. In: *Proc. Natl. Acad. Sci. U. S. A.* 113.16, pp. 4290–4295.