Shape Constrained Tensor Decompositions

Bethany Lusch
Leadership Computing Facility
Argonne National Laboratory
Lemont, IL 60439
Email: blusch@anl.gov

Eric C. Chi
Department of Statistics
North Carolina State University
Raleigh, NC 27695
Email: eric.chi@ncsu.edu

J. Nathan Kutz
Department of Applied Mathematics
University of Washington
Seattle, WA 98195
Email: kutz@uw.edu

Abstract—We propose a new low-rank tensor factorization where one mode is coded as a sparse linear combination of elements from an over-complete library. Our method, Shape Constrained Tensor Decomposition (SCTD) is based upon the CANDECOMP/PARAFAC (CP) decomposition which produces $r$-rank approximations of data tensors via outer products of vectors in each dimension of the data. The SCTD model can leverage prior knowledge about the shape of factors along a given mode, for example in tensor data where one mode represents time. By constraining the vector in the temporal dimension to known analytic forms which are selected from a large set of candidate functions, more readily interpretable decompositions are achieved and analytic time dependencies discovered. The SCTD method circumvents traditional flattening techniques where an $N$-way array is reshaped into a matrix in order to perform a singular value decomposition. A clear advantage of the SCTD algorithm is its ability to extract transient and intermittent phenomena which is often difficult for SVD-based methods. We motivate the SCTD method using several intuitively appealing results before applying it on a real-world data set to illustrate the efficiency of the algorithm in extracting interpretable spatio-temporal modes. With the rise of data-driven discovery methods, the decomposition proposed provides a viable technique for analyzing multitudes of data in a more comprehensible fashion.

Index Terms—tensor decomposition, multiway arrays, multilinear algebra, higher-order singular value decomposition (HOSVD), over-complete libraries, sparse regression.

I. INTRODUCTION

Tensor decompositions are a generalization of the singular value decomposition (SVD) to higher dimensions, allowing for $N$-way arrays ($N \geq 3$) of data to be decomposed into their constitutive, low-rank subspaces without flattening, which is especially advantageous for categorical data types. With the rise of data science and data-driven discovery, tensor decompositions are of increasing value and importance in many areas of science and engineering.

There are a variety of tensor decompositions that can be applied to $N$-way arrays ($N \geq 3$) of data. We consider the CANDECOMP/PARAFAC (CP) decomposition [2]–[5], which is an outer product of three vectors and is of the form

\[ M = \sum_{j=1}^{r} \lambda_j a_j \odot b_j \odot c_j \]

illustrated in Fig. 1, which arranges $r$-rank data into a series of $r$ outer products of $N$ vectors. Under weak conditions, a rank-$r$ tensor has a unique CP tensor decomposition into $r$ rank-one component tensors, with the exception of scaling and permutation of the $r$ components [1]. It has been applied in several areas of signal processing, including multiple invariance sensor array processing [6], identification of a Multiple-Input Multiple-Output (MIMO) system [7], antennas [8], and MIMO radar [9].

Several constrained versions of the CP tensor decomposition have been proposed. The constraints considered so far include non-negativity [10]–[15], linear constraints [16], symmetry [3], orthogonality [17], [18], Vandermonde structure [19], and banded and/or structured matrix factors (such as Hankel or Toeplitz) [20], [21].

There are other decompositions available, including the Tucker tensor decomposition [1] and the recently developed tensor-based method [22] for Dynamic Mode Decomposition (DDM) [23]. The former method is widely used in the tensor community and is also known as the higher-order SVD (HOSVD) since it essentially involves $N$ matrix SVDs [24]. The latter method provides a regression that enforces Fourier mode behavior in the time mode. All of these tensor-based methods fall short of our primary goal, which is to

Fig. 1. CP tensor decompositions. This type of decomposition approximates a data set $X$ with a tensor $M$ consisting of $r$ components. Each component is an outer product of three vectors and is of the form $\lambda_j a_j \odot b_j \odot c_j$. This research was primarily conducted while B. Lusch was employed at the University of Washington.
provide a tensor decomposition with analytically tractable time dynamics capable of modeling transient phenomena. The DMD algorithm solves the first part of this objective but fails in modeling transient phenomena. Although a multi-resolution DMD method has been proposed to handle transients [25], it has a multiple pass architecture that sometimes struggles to extract spatio-temporal structures in an unsupervised manner.

In applications, one of the important dimensions of the data set is a time variable which measures how the other quantities of interest evolve over a prescribed time course. Many tensor decompositions produce the low-rank time variable evolution. However, the low-rank time modes often are complicated and noisy due to the form of the data itself. In contrast, we often expect simple and highly structured temporal signatures, whether it be oscillations of a prescribed frequency or exponential growth/decay of a signal, for instance. The natural remedy is to constrain the form of the temporal modes extracted from the tensor decomposition.

In this work, we demonstrate that the CP tensor decomposition can be modified to constrain the time dynamic mode to a broad range of analytic solutions that are selected from a large and over-complete library of candidate functions. By using sparse $\ell_1$ regression techniques, a minimal, but most informative, set of candidate functions are selected for representing the temporal dynamics. We call this technique Shape Constrained Tensor Decomposition (SCTD). The clear benefit of the SCTD over standard CP decompositions is that it gives analytic solutions which are readily interpretable. This is highly advantageous for characterizing the structure of the data and for data-driven discovery of underlying processes responsible for producing the dynamics observed. We demonstrate the method on a number of examples, including high-dimensional data generated from global temperature measurements.

The rest of the paper is organized as follows: Sec. II develops the basic mathematical architecture of the CP tensor decomposition and our refinement, the SCTD algorithm. This is followed by Sec. III in which we discuss practical details such as how to select tuning parameters and how to construct an appropriate over-complete library. Sec. IV tests the algorithm on simulated data, and Sec. V provides examples demonstrating the effectiveness of the algorithm on real-world data sets. Conclusions and an outlook for the SCTD algorithm are discussed in Sec. VI.

II. METHODOLOGY

In this manuscript, we present a modification to the standard CP tensor decomposition that is intuitively appealing and improves interpretability of the low-rank modes extracted from data. Specifically, we introduce an over-complete library of temporal responses that constrains the time mode dynamics. A sparsity-promoting algorithm selects a small number of these modes to represent the data. Thus the procedure can be thought of as a sparsity-promoting, constrained optimization problem.

The SCTD method is illustrated at a high level in Figs. 2 and 3. Fig. 2 shows the selection process whereby a small number of modes from an over-complete library of temporal functions are selected to best represent the temporal evolution of the data. We rely on an $\ell_1$ optimization procedure so as to obtain a sparse representation of the temporal dynamics. The algorithm thus restricts the temporal mode in the decomposition. While the temporal functions populating the library may be arbitrary, the power of the SCTD relies on the fact that temporal dynamics are typically far from arbitrary. Fig. 3 illustrates some temporal functions that serve as prototypes that characterize real-world temporal dynamics. Note that some of these functions are ideally suited for handling transient dynamics. Indeed, the success of the method is directly related to the temporal library functions included in the regression procedure.

A specific demonstration of the SCTD is shown in Fig. 4. In this example, three different spatial mode structures are combined with three specific time dynamics. The imposed time dynamics are representative of simple functional forms that are often difficult for the standard CP or DMD methods to model or resolve, i.e. temporal responses that have finite time windows of activity. In this example, the sequence of data snapshots are gathered into a 3-way data tensor $\mathbf{M}$. Different snapshots of the dynamics depict the spatial structure arising from the combination of the different modal structures. The objective of the SCTD is to solve the inverse problem: Given the data tensor $\mathbf{M}$, find the low-rank decomposition that correctly reconstructs the spatial modes and their time dynamics. The algorithm proposed here, which is based on the CP decomposition, can indeed recover the three modes and their time dynamics as shown in Fig. 4.

In the subsections that follow, the technical details of the CP tensor decomposition algorithm are considered along with strategies for building an over-complete library and enforcing a parsimonious combination of temporal dynamics prototypes.
Let \( \mathbf{M} \) represent an \( N \)-way data tensor of size \( I_1 \times I_2 \times \cdots \times I_N \). We are interested in an \( R \)-component CANDECOMP/PARAFAC (CP) \([2]–[4]\) factor model

\[
\mathbf{M} = \sum_{r=1}^{R} \lambda_r \mathbf{a}_r^{(1)} \odot \cdots \odot \mathbf{a}_r^{(N)},
\]

where \( \odot \) represents outer product and \( \mathbf{a}_r^{(n)} \) represents the \( r \)th column of the factor matrix \( \mathbf{A}^{(n)} \) of size \( I_n \times R \). We refer to each summand as a component. Assuming each factor matrix has been column-normalized to have unit Euclidean length, we refer to the \( \lambda_r \)'s as weights. We will use the shorthand notation \( \mathbf{M} = [\lambda; \mathbf{A}^{(1)}, \ldots, \mathbf{A}^{(N)}] \), where \( \lambda = (\lambda_1, \ldots, \lambda_R)^T \) [26]. A tensor that has a CP decomposition is sometimes referred to as a Kruskal tensor.

For the rest of this article, we consider a 3-way tensor where two modes index state variation and the third mode indexes time variation.

\[
\mathbf{M} = \sum_{r=1}^{R} \lambda_r \mathbf{a}_r \odot \mathbf{b}_r \odot \mathbf{c}_r.
\]

Let \( \mathbf{A} \in \mathbb{R}^{I_1 \times R} \) and \( \mathbf{B} \in \mathbb{R}^{I_2 \times R} \) denote the factor matrices corresponding to the two state modes and \( \mathbf{C} \in \mathbb{R}^{I_3 \times R} \) denote the factor matrix corresponding to the time mode. This 3-way decomposition is illustrated in Fig. 1.

**B. Sparse Representations in Over-Complete Libraries**

We can further impose structure on the factors of the low-rank decomposition. For example, we could impose sparsity and smoothness on the factors \([27]–[29]\). Here we assume that the time mode can be coded as a sparse linear combination of functions from a known over-complete library \( \mathbf{D} \in \mathbb{R}^{I_3 \times F} \), namely \( \mathbf{C} = \mathbf{DZ} \), where the elements of \( \mathbf{Z} \in \mathbb{R}^{F \times R} \) are predominantly zero, i.e. \( \text{nnz}(\mathbf{Z}) \ll PR \), and \( I_3 \ll P \), where \( \text{nnz}(\mathbf{Z}) \) is the number of non-zero elements in \( \mathbf{Z} \). This set up can be thought of as a sparse version of CANDELINC (canonical decomposition with linear constraints) [16]. Figs. 2 and 3 show both the constrained decomposition and library functions used. We make the following assumption on \( \mathbf{D} \).

**Assumption 1:** The column space of \( \mathbf{X}_{(3)} \) is not orthogonal to the column space of \( \mathbf{D} \).

We will rely on this assumption later for a convergence proof. Note that this assumption is reasonable and not demanding as we seek a dictionary \( \mathbf{D} \) whose columns will be used to approximate the time mode.

We seek the CP model that maximizes a penalized correlation with the data tensor \( \mathcal{X} \)

\[
\hat{\mathbf{M}} = \arg \max_{\mathbf{M}} f(\mathbf{M}) \equiv \langle \mathcal{X} \mid \mathbf{M} \rangle - \tau \| \mathbf{Z} \|_F
\]

such that

\[
\mathbf{M} = [\lambda; \mathbf{A}, \mathbf{B}, \mathbf{C}], \quad \mathbf{C} = \mathbf{DZ}, \quad \| \lambda \|_2 \leq \| \mathcal{X} \|_F, \quad \| \mathbf{a}_r \|_2, \| \mathbf{b}_r \|_2, \| \mathbf{z}_r \|_2 \leq 1 \text{ for } r = 1, \ldots, R.
\]
In the problem statement above and in what follows, we express the sparsity penalty as a $\| \cdot \|_1$ penalty. In practice, however, we use an adaptive LASSO technique [31] which augments the standard LASSO with a weighting on each term. For ease of presentation and clarity, we have chosen to represent the sparsity penalty by a simple $\ell_1$ penalization.

C. Algorithm

The CP constraint $\mathcal{M} = [\lambda; A, B, C]$ and orthogonality constraints render the optimization problem in (2) non-convex. Thus, we propose a greedy procedure for building a SCTD one rank-1 tensor at a time. Our SCTD algorithm has two “levels” of iteration. In the outer level, we estimate a sequence of rank-1 terms. We refer to each outer level “iteration” as a round. Thus, after 5 rounds we have estimated 5 rank-1 terms in our CP decomposition. Within each of these rounds, we employ block coordinate ascent (BCA) to iteratively estimate a rank-1 term.

Outer level: In the first round, we estimate the first rank-1 term in $\mathcal{M}$ by solving the optimization problem

$$\hat{\mathcal{M}}_1 = \arg \max_{\mathcal{M}_1 \in \mathcal{C}} f(\mathcal{M}_1) \equiv \langle \mathcal{X} | \mathcal{M}_1 \rangle - \tau \|\mathbf{z}_1\|_1,$$

(3)

where $\mathcal{C} = \{ \mathcal{M}_1 = a_1 \circ b_1 \circ d_2 : \|a_r\|_2, \|b_r\|_2, \|d_r\|_2 \leq 1 \}$. For all subsequent rounds, we employ a modest modification to the update (3) to ensure orthogonality of the columns of the matrices $A, B, C$. This orthogonal procedure was suggested but not studied in [28]. After completing $r - 1$ rounds, we have the factor matrix $A_{r-1} = (a_1 \cdots a_{r-1})$ which we use to compute the projection matrix $\Pi_{A_{r-1}} = I - A(A^T A)^{-1}A^T$. The matrix $\Pi_{A_{r-1}}$ projects vectors onto the orthogonal complement of the column space of $A$. We similarly compute $\Pi_{B_{r-1}}$ and $\Pi_{C_{r-1}}$. Let $y_r$ denote the tensor obtained by multiplying the data tensor $X$ by the projection matrices $\Pi_{A_{r-1}}, \Pi_{B_{r-1}}$ and $\Pi_{C_{r-1}}$ along $X$’s appropriate modes, namely $X \times_1 \Pi_{A_{r-1}} \times_2 \Pi_{B_{r-1}} \times_3 \Pi_{C_{r-1}}$. Within the $r$th round, we solve the following maximization problem

$$\hat{\mathcal{M}}_r = \arg \max_{\mathcal{M}_r \in \mathcal{C}} f(\mathcal{M}_r) \equiv \langle y_r | \mathcal{M}_r \rangle - \tau \|\mathbf{z}_r\|_1.$$  

(4)

Algorithm 1 summarizes the outer level of iteration.

Algorithm 1: Shape Constrained Tensor Decomposition

Fig. 4. Extracting patterns from spatio-temporal data. (a) We begin with a data set where spatial information is collected over time. If we collect two-dimensional data at each time step, we may informally think of the data as a sequence of “frames.” (b) The sequence of frames can be saved as a tensor (one data cube) where the third dimension is time. (c) Our goal is to decompose that tensor into a sum of important frame components where each frame component has its own time dynamics. In this example, we see the three components coming in and out of the frames as time passes. The color coding demonstrates how the sample frames in part (a) are combinations of the components shown in part (c).
Let $M$ be a tensor of size $I_1 \times I_2 \times I_3$.

1. Let $\{a_1, b_1, z_1\} \leftarrow \text{arg max } \langle Y | M_1 \rangle - \tau \|z\|_1$

2. for $r = 2, 3, \ldots$

3. \[ A_r \leftarrow \left\{ a_1, \ldots, a_{r-1} \right\} \]

4. \[ B_r \leftarrow \left\{ b_1, \ldots, b_{r-1} \right\} \]

5. \[ C_r \leftarrow D \{ z_1, \ldots, z_{r-1} \} \]

6. \[ Y_r \leftarrow X \times_1 A_{r-1} \times_2 B_{r-1} \times_3 C_{r-1} \]

7. \[ \{ a_r, b_r, z_r \} \leftarrow \text{arg max } \langle Y_r | M_r \rangle - \tau \|z\|_1 \{ \text{BCA} \} \]

8. \[ \lambda_r \leftarrow \langle X | a_r \odot b_r \odot Dz_r \rangle \]

9. end for

Before describing the inner level iteration, we briefly elaborate on one of the merits of the scheme described above. In addition to improving interpretability of the factor matrices by enforcing orthogonality, the above outer iteration scheme avoids deflation. Recall that deflation seeks to construct a rank-$R$ approximation by computing best rank-1 approximations to $\Pi \times_1 A_{r-1} \times_2 B_{r-1} \times_3 C_{r-1}$. The optimization problem posed in (7) is a modified lasso problem and has appeared in similar settings [27], [28], [33]. Recall that the elementwise soft-thresholding operator applied to the vector $w$, denoted by $S_r(w)$, sends $w_i$ to the value $\text{sign}(w_i) \max\{ |w_i| - \tau, 0 \}$. We can express the $z$ update by

\[ z_r^{(n+1)} = \begin{cases} \frac{S_r(w_r)}{\|S_r(w_r)\|_2} & \text{if } S_r(w_r) \neq 0 \\ 0 & \text{otherwise.} \end{cases} \]

For a derivation of this update rule see the proof of Lemma 2.2 in [27]. We then update $\lambda_r$ according to (5).

Algorithm 2 summarizes the inner level of iteration. Note that Algorithm 2 will terminate immediately if any of the vectors $a_r$, $b_r$, or $z_r$ become zero. Also, note that Algorithm 2 can be generalized in a straightforward manner for an $N$-way tensor. Additional unconstrained modes will introduce additional updates that are identical to the $a_r$ and $b_r$ updates while additional constrained modes will be identical to the $z_r$ update using an appropriate mode-specific dictionary.

Algorithm 2: Block Coordinate Ascent (BCA) at round $r$ Initialize $a^{(0)}$, $b^{(0)}$, and $z^{(0)}$.

1. for $n = 1, 2, 3, \ldots$

2. \[ c^{(n-1)} \leftarrow Dz^{(n-1)} \]

3. \[ u \leftarrow Y_1 \left[ c^{(n-1)} \odot b^{(n-1)} \right] \{ \text{Update } a_r \} \]

4. \[ a_r^{(n)} \leftarrow \begin{cases} u & \text{if } u \neq 0 \\ 0 & \text{otherwise.} \end{cases} \]

5. \[ v \leftarrow Y_2 \left[ c^{(n-1)} \odot a_r^{(n)} \right] \{ \text{Update } b_r \} \]

6. \[ b_r^{(n)} \leftarrow \begin{cases} v & \text{if } v \neq 0 \\ 0 & \text{otherwise.} \end{cases} \]

7. \[ z_r^{(n)} \leftarrow \begin{cases} w & \text{if } w \neq 0 \\ 0 & \text{otherwise.} \end{cases} \]

9. end for

D. Complexity of BCA

Precomputing $D^TY_3$ requires $O(p I_1 I_2 I_3)$ flops. Computing $c_r = Dz_r$ requires $O(I_3 \times \text{nnz}(z_r))$ operations. Constructing $u_r$ requires $O(I_1 I_2 I_3)$ flops. The $a_r^{(n)}$ update requires $O(I_1 I_2 I_3)$ flops. Constructing $v_r$ requires $O(I_1 I_2 I_3)$ flops. The $b_r^{(n)}$ update requires $O(I_2)$ flops. Constructing $w_r$ requires $O(p I_1 I_2)$ flops. The $z_r^{(n)}$ update requires $O(p)$ flops. Assuming that we have a large dictionary, namely $p \gg I_3$, the dominant cost will be the formation of the vector $w_r$; consequently the per iteration complexity is $O(p I_1 I_2)$. Thus, the per-iteration complexity grows linearly in the size of the dictionary.
E. Convergence of BCA

We now discuss the convergence properties of BCA within a given round \( r \). For notational convenience, we bundle the parameters \( a_n, b_n, z_n \) into the vector \( \theta \), namely at the \( n \)th iteration of BCA, we compute the vector \( \theta^{(n)} \in \mathbb{R}^{I_1 + I_2 + I_3} \), where
\[
\theta^{(n)} = \left( a^{(n)}_\tau^T, b^{(n)}_\nu^T, z^{(n)}_\nu^T \right).
\]

The convergence theory of monotonically increasing algorithms, like BCA, hinges on the properties of the map \( T : \mathbb{R}^{I_1 + I_2 + I_3} \rightarrow \mathbb{R}^{I_1 + I_2 + I_3} \), that sends the \( n \)th iterate to the \( n + 1 \)th iterate, namely \( \theta^{(n+1)} = T(\theta^{(n)}) \).

Note one issue with the BCA algorithm that needs to be carefully accounted for is that the all zeros vector is a fixed point of the algorithm map, namely \( T(0) = 0 \). This is one of our contributions as prior work does not define the algorithm carefully as we do. Since the all zeros vector is not useful, we will state the convergence algorithm in terms of a sequence of iterates that are never the all zeros vector. This can be maintained in practice by not taking \( \tau \) too large. An upper limit of \( \tau \) is the point at which all entries of \( z_\nu \) are zero.

We have the following convergence guarantees for the BCA algorithm.

Proposition II.1. The limit points of a non-zero sequence of BCA iterates \( \theta^{(n)} \) are stationary points of the optimization problem (4). Moreover, the distance between successive iterates converges to zero, namely \( \| \theta^{(n+1)} - \theta^{(n)} \| \rightarrow 0 \).

The proof is contained in the Appendix.

III. DETAILS OF SCTD ALGORITHM

Now that we are familiar with the basic procedure, we next discuss important details in the SCTD. We first describe how the sparsity-inducing parameter \( \tau \) is chosen. We then describe how the over-complete library is constructed.

A. Picking The Regularization Parameter

At each iteration, we use the Bayesian Information Criterion (BIC) [34] to pick regularization parameter \( \tau \). The BIC is a quantitative score that balances how well the model fits the data against how complicated the model is. In the context of the SCTD, a constrained rank-1 Kruskal tensor with low BIC corresponds to a rank-1 Kruskal tensor which fits the data well in light of how many free parameters were used in fitting it. As defined in [28], for this problem, the BIC criterion is
\[
\text{BIC}(\tau_r) = \log \left( \frac{\| y - \lambda_a a_n b_n \circ D_z_r \|_F^2}{I_1 I_2 I_3} \right) + \log \left( \frac{I_1 I_2 I_3}{I_1 I_2 I_3} \right) \left( \text{size of } \{ z_r \} \right),
\]
where \( \{ z_r \} \) is the number of non-zero elements of \( z_r \). This can be derived from each update being an \( \ell_1 \)-norm penalized regularization problem.

We use the BIC criterion to pick the best \( \tau_r \) from a range of options. We further refine the value of \( \tau_r \) by checking the neighborhood of the current best option.

B. Constructing the Over-Complete Library

We can choose an over-complete library based on knowledge of the application area or data set. Some natural candidates are displayed in Fig. 3. If we expect periodic but transient dynamics, we may choose to populate the library with windowed sines and cosines, varying the frequencies of the sines and cosines and the widths and shifts of the windows. If we anticipate transient phenomena that are non-periodic, it may be appropriate to include Gaussians with a range of means and variances. If the time domain itself is periodic, such as hour of the day, then we might improve the results by including dynamics that have this period. For example, to allow a Gaussian-like mode to vary smoothly through the night, we could generate cosines with varying frequencies and shifts, but only include one period of the cosine (see “wrapped cosines” in Fig. 3). We use the BIC criterion to choose the size of the library.

IV. SIMULATION EXPERIMENTS

We begin by testing the SCTD on a simulated data set similar to Fig. 4. Recall that this data set is composed of three spatio-temporal modes (specifically, it is a Kruskal tensor with rank three). The equations to generate this data set are listed in the Appendix. We can think of this data set as a video or sequence of frames. Our goal is to decompose it into three modes (a rank-three Kruskal tensor) with an analytical description for the temporal dimension. Although the SCTD is exceptional for the data in Fig. 4, the example is limited since no noise was included in the data.

A more realistic example is shown in Fig. 5. In this experiment, we added white Gaussian noise with \( \sigma \) in the frequency domain to the data \( (\omega, t) = F^{-1}[u(\omega) + \sigma N(0, 1)] \). In this case, we used \( \sigma = 3 \), resulting in a signal-to-noise ratio of 0.1374 (-8.620 dB), where signal-to-noise ratio is defined as the ratio of the summed squared magnitude of the signal to the summed squared magnitude of the noise. The algorithm outlined in Sec. II can now be applied to the data and a direct comparison can be made to a CP decomposition and a DMD reduction. In particular, for the CP decomposition, we use the CP_ALS function in the Matlab Tensor Toolbox [35], [36], which uses an alternating least squares algorithm [37]. Fig. 5 shows that despite the inclusion of noise, the modes and temporal dynamics can be cleanly extracted using the SCTD. Indeed, analytic forms for the time dynamics can be discovered. Although the true modes were not included in the library, the SCTD chooses functions with similar frequencies and windows to the functions used to generate the data (see the Appendix for the formulas). In comparison, the CP algorithm gives a decomposition with noisy time modes which lack analytic description. The DMD algorithm (using data flattening) can give analytic expressions for the time dynamics, but the temporal expressions are flawed because DMD cannot handle such transient and/or intermittent time dynamics, i.e. only time dynamics of the form \( \exp(\omega t) \) are allowed.

The SCTD also provides a diagnostic for performing an \( r \)-rank truncation. For an SVD decomposition, the singular val-
Real data

CP tensor decomposition

Dynamic Mode Decomposition

SCTD

e^{(-0.002+0.101i)t}
e^{(-0.013+0.045i)t}
e^{(-0.013-0.045i)t}
cos .1t, t ∈ [0, 128]
sin .4t, t ∈ [0, 63.5]
sin .7t, t ∈ [59.9, 97.7]

Fig. 5. Comparing methods on a simulated data set. The data set, a 3-way tensor, is generated as described in Fig. 4, except that noise is added. We hope that a method can decompose the tensor into its three noiseless components. A traditional CP tensor decomposition sometimes falls into a good local minimum and decomposes the data correctly. Clean spatial modes are found, but some noise in the time dynamics is maintained. The time dynamics are not fit to analytic expressions. The Dynamic Mode Decomposition tries to fit clean time dynamics functions to the spatial modes. However, it is restricted to Fourier modes and cannot handle the windowed behavior in this data set. It also does not correctly separate the third spatial mode. The SCTD finds clean spatial modes and fits smooth time dynamics to each component. The output includes the particular analytic functions that were fit to the time dynamics. Note that while the true modes were not included in the library, the SCTD chooses functions close to those used to generate the data.

Fig. 6. Reconstruction error curve. We can choose the number of components to keep in the SCTD by considering the trade-off between error and complexity. Here we see diminishing returns in reconstruction error after the inclusion of the first three components, suggesting that a rank-3 approximation sufficiently captures the majority of the systematic variation in the data.

To further explore the example shown in Fig. 5, we consider a number of different cases which highlight the use of the algorithm and the choice of library prototypes. Thus we consider the following:

Case (a): Library contains true modes. We start with an easy case. We construct a library with 2000 prototypes, including the true temporal modes, and we do not add noise to the data. We see in Fig. 7 and Table I that in each iteration, the SCTD picks exactly one prototype. The library size is chosen using the BIC criterion. See Fig. 9 for further experimentation. This is the only case in which the library contains the true modes.

Case (b): Library does not contain true modes. Next, we assess how robust the SCTD is to “model misspecification”: We construct another library of 3000 prototypes but do not “cheat” by including the true time dynamics. As we can see in Fig. 7 and Tab. I, in this case, the method uses extra prototypes (about 0.12% of the library) and accumulates more error, yet the factor accuracy only drops from 0.9998 to 0.9190.

Case (c): Library does not contain true modes and the data is noisy. Finally, we increase the difficulty by adding white Gaussian noise to the data (σ = 1). The results are very similar to Case (b) without noise (see Fig. 7 and Tab. I). Note that although the resulting analytic expression of 69 prototypes is

293
not simple, if you want a simple analytical expression, you can pick the mode with the largest coefficient and still maintain accuracy. The top mode is plotted in green on top of the linear combination (blue) and the true mode (black) in Fig. 7. In this case, the library size chosen by the BIC criterion is significantly larger, 12,000 prototypes.

Next, we consider the effects of changing $\sigma$ and the library size. We have seen two examples of Case (c). In Fig. 5, the library contains 3,000 prototypes and the white noise has $\sigma = 3$. In Fig. 7 (and Tab. I), the library contains 12,000 prototypes and the white noise has standard deviation $\sigma = 1$. We now vary $\sigma$ (Fig. 8) and the library size (Fig. 9).

In Fig. 8, we vary $\sigma$. For each case, we use the BIC criterion to choose the best library size out of 1, 000, 2, 000, \ldots 10, 000. As the magnitude of the noise increases, so does the error. However, this growth in error is slow when the error is measured against the original (noiseless) data. For context, see Fig. 5 for a visualization of data with $\sigma = 3$. In Fig. 9, we consider data that has no noise and vary the library size. Once the library is sufficiently large, the relative error does not improve. However, the number of prototypes chosen grows.

V. REAL DATA EXAMPLE: EL NIÑO

We now apply the SCTD to a real-world data set exhibiting complex spatio-temporal dynamics with intermittency to illustrate the power of the SCTD to produce interpretable results, especially in the constrained time dynamics. We demonstrate the SCTD on a data set of sea surface temperatures. The data are freely available from the NOAA/OAR/ESRL PSD, Boulder, Colorado, USA. We used the weekly sea surface temperature from the NOAA_OI_SST_V2 data set, which can be downloaded from http://www.esrl.noaa.gov/psd/. In particular, we consider the Pacific Ocean from 1995 through the end of 2000. We subtracted the background from the data using DMD [38], and then we created a library with a combination of Gaussians and windowed sines and cosines. Two of the first twelve modes that the SCTD extracted are shown in Fig. 10. The second component finds one-time phenomena related to the El Niño event of 1997–1998. In particular, we see unusually warm temperatures in the eastern Pacific ocean, especially near Peru, but almost stretching to New Guinea. By mid-to-late 1997, unusually cool waters occurred near the coast of Australia. The third component finds annual variation in temperature.

These results could not be obtained with standard DMD because the El Niño event is not a Fourier mode. However, recent innovations around multi-resolution analysis and DMD (the multi-resolution DMD algorithm [25]) do allow for a significantly improved description. Likewise, a traditional CP tensor decomposition might extract similar patterns, they would not be accompanied with a sparse analytic description. We show that the SCTD chooses a sparse linear combination of our over-complete library in Fig. 11.

VI. CONCLUSION

In this manuscript, we have developed what we think is a highly useful innovation to the standard CP tensor decomposition. By constraining the time dimension of the tensor decomposition, a more intuitively appealing and interpretable decomposition can be achieved. Indeed, analytic solution forms for the time dependency of the data decomposition can be extracted. This is done by using an over-complete library of potential temporal functions in order to select the best candidate functions via sparse regression. This work merges three distinct mathematical methods: tensor decompositions, sparse regression, and over-complete libraries. The success of the SCTD method is demonstrated on a number of simulated problems and one real-world application where preserving the tensor nature of the data is highly desirable and advantageous. The SCTD method provides a viable data-discovery algorithm that can be used in a host of settings where low-rank features of an $N$-way data tensor need to be analyzed. It should also be noted that one can easily envision also constraining other dimensions of the data, not just the time dimension. Potential future work includes learning the library, drawing inspiration from papers that incorporate shape learning [39]–[41]. Ultimately, the most useful data analysis techniques developed allow for interpretable diagnostics which are also predictive in nature. The SCTD advances a theoretical framework for tensor decompositions that provides an intuitively appealing framework for understanding the rich time dynamics of low-rank decompositions without requiring data-flattening. With the emergence of many categorical data structures, this can be especially appealing. Thus, we render a tensor decomposition package that is user-friendly and aids in identifying important dynamics structures in data, including intermittent phenomena, which are very difficult for standard tensor, DMD and PCA-like methods to deduce. All MATLAB and R codes used for this paper are available online at github.com/BethanyL/SCTD.

ACKNOWLEDGMENT

J. N. Kutz acknowledges support from the Air Force Office of Scientific Research (FA9550-15-1-0385). B. Lusch acknowledges fellowship support from the National Physical Science Consortium and National Security Agency. This research
True components  (a) Library contains true modes  (b) Library does not contain true modes  (c) Library does not contain true modes and data noisy

Fig. 7. Results on simulated data set. We repeat for reference the three true components that compose the data set. (a) When the library contains the correct time dynamics functions, the SCTD does a good job of recovering them. (b) When the library does not contain the exact right modes, the SCTD uses more prototypes to fit the data, but still chooses a sparse number. (c) When we additionally make the data noisy, the SCTD is robust. It chooses more prototypes, but if an especially simple output is desired, using just the prototype with the highest coefficient is accurate. See more detail in Tab. I.

Fig. 8. Varying the noise. In Figs. 5 and 7 and Tab. I, we displayed results on noisy data. Here we vary the amount of noise to display the robustness of the SCTD. The value of $\sigma$ ranges 0.1–4 while the SNR ranges from 20.93 to $-0.057$ dB. As the noise increases, the error in the reconstruction of the original data increases. Note that the cases of $\sigma = 3$ and $\sigma = 1$ are displayed in Figs. 5 and 7, respectively. The increase in error is slow when the error is in terms of the noiseless data.

Fig. 9. Varying the library size. In Fig. 7 and Tab. I, we displayed results on a library with 2,800 prototypes. Here we vary the size of the library to consider the tradeoffs. Once we have a reasonably large library, the relative error is consistent. However, the number of selected prototypes roughly grows with the library size. Thus to limit complexity, we may wish to pick a library size that is sufficient for low error reconstructions but is not larger than necessary.

Fig. 10. Results on ocean surface temperature data set. We start with a data set of ocean surface temperature over time. The dimensions are longitude, latitude, and time. Here we display a sample of the components found by the SCTD. The second component finds the El Niño event of 1997–1998, a warm band in the central and east-central equatorial Pacific. The third contains annual variation.

was funded in part by the Argonne Leadership Computing Facility, which is a DOE Office of Science User Facility supported under Contract DE-AC02-06CH11357.

REFERENCES

[1] T. G. Kolda and B. W. Bader, “Tensor decompositions and applications,” *SIAM Review*, vol. 51, no. 3, pp. 455–500, 2009.
[2] F. L. Hitchcock, “The expression of a tensor or a polyadic as a sum of products,” *Journal of Mathematics and Physics*, vol. 6, no. 1-4, pp. 164–189, 1927.
Fig. 11. Results on ocean surface temperature data set, continued. This figure gives further information about the results in Fig. 10. We demonstrate the sparsity of the time dynamics by plotting the magnitudes of the coefficients in each $z_k$.

[3] J. D. Carroll and J.-J. Chang, “Analysis of individual differences in multidimensional scaling via an N-way generalization of Eckart-Young decomposition,” Psychometrika, vol. 35, pp. 283–319, 1970.

[4] A. R. Harshman, “Foundations of the PARAFAC procedure: Models and conditions for an ‘explanatory’ multi-modal factor analysis,” UCLA working papers in phonetics, vol. 16, pp. 1–84, 1970.

[5] M. Sørensen, L. D. Lathauwer, P. Comon, S. Icart, and L. Deneire, “Canonical polyadic decomposition with a columnwise orthonormal factor matrix,” SIAM Journal on Matrix Analysis and Applications, vol. 33, no. 4, pp. 1190–1213, 2012.

[6] N. D. Sidiropoulos, R. Bro, and G. B. Giannakis, “Parallel factor analysis in sensor array processing,” IEEE Transactions on Signal Processing, vol. 48, no. 8, pp. 2377–2388, 2000.

[7] B. Chen, A. P. Petropulu, and L. De Lathauwer, “Blind identification of convolutive MIMO systems with 3 sources and 2 sensors,” EURASIP Journal on Advances in Signal Processing, no. 5, pp. 487–496, 2002.

[8] L. De Lathauwer and A. de Baynast, “Blind deconvolution of dsedm signals by means of decomposition in rank-(1,1,1) terms,” IEEE Transactions on Signal Processing, vol. 56, no. 4, pp. 1562–1571, 2008.

[9] D. Nion and N. D. Sidiropoulos, “Tensor algebra and multidimensional harmonic retrieval in signal processing for mimo radars,” IEEE Transactions on Signal Processing, vol. 58, no. 11, pp. 5693–5705, 2010.

[10] R. Bro and S. De Jong, “A fast non-negativity-constrained least squares algorithm,” Journal of Chemometrics, vol. 11, no. 5, pp. 393–401, 1997.

[11] L.-H. Lim and P. Comon, “Nonnegative approximations of nonnegative tensors,” Journal of Chemometrics, vol. 23, no. 7–8, pp. 432–441, 2009.

[12] M. P. Friedlander and K. Hatz, “Computing non-negative tensor factorizations,” Optimisation Methods and Software, vol. 23, no. 4, pp. 631–647, 2008.

[13] P. Paatero, “A weighted non-negative least squares algorithm for three-way PARAFAC factor analysis,” Chemometrics and Intelligent Laboratory Systems, vol. 38, no. 2, pp. 223–242, 1997.

[14] ——, “The multilinear engine - A table-driven, least squares program for solving multilinear problems, including the n-way parallel factor analysis model,” Journal of Computational and Graphical Statistics, vol. 8, no. 4, pp. 854–888, 1999.

[15] M. Welling and M. Weber, “Positive tensor factorization,” Pattern Recognition Letters, vol. 22, no. 12, pp. 1255–1261, 2001.

[16] J. Douglas Carroll, S. Pruzansky, and J. B. Kruskal, “Candelab: A general approach to multidimensional analysis of many-way arrays with linear constraints on parameters,” Psychometrika, vol. 45, no. 1, pp. 3–24, 1980.

[17] M. Sørensen, L. D. Lathauwer, P. Comon, S. Icart, and L. Deneire, “Canonical polyadic decomposition with a columnwise orthonormal factor matrix,” SIAM Journal on Matrix Analysis and Applications, vol. 33, no. 4, pp. 1190–1213, 2012.

[18] W. P. Krijnen, T. K. Dijkstra, and A. Stegeman, “On the non-existence of optimal solutions and the occurrence of ‘degeneracy’ in the CANDECOMP/PARAFAC model,” Psychometrika, vol. 73, no. 3, pp. 431–439, 2008.

[19] M. Sørensen and L. De Lathauwer, “Blind signal separation via tensor decomposition with Vandermonde factor: Canonical polyadic decomposition,” IEEE Transactions on Signal Processing, vol. 61, no. 22, pp. 5507–5519, 2013.
APPENDIX

Notation Details

Matricization of a tensor: The mode-\(n\) matricization or unfolding of a tensor \(A\) is denoted by \(A_{(n)}\) and is of size \(I_n \times J_n\) where \(J_n = \prod_{m \neq n} I_m\). In this case, the tensor element with index \(i\) maps to matrix element \((i, j)\) where

\[
i = i_n \quad \text{and} \quad j = 1 + \sum_{k=1}^{N} (i_k - 1) \left( \prod_{m \neq n=1}^{m} I_m \right).
\]

Generation of Simulated Data

The simulated data set used in the experiments in Section IV is generated by summing three spatio-temporal modes:

\[
f(x, y, t) = \Psi_1(x, y) \circ a_1(t) + \Psi_2(x, y) \circ a_2(t) + \Psi_3(x, y) \circ a_3(t),
\]

where

\[
\Psi_1(x, y) = \exp(-0.1(x+25)^2 - 0.1(y-25)^2),
\]

\[
\Psi_2(x, y) = \exp(-0.02(x-25)^2 - 0.05(y+25)^2),
\]

\[
\Psi_3(x, y) = \exp(-0.05(x+10)^2 - 0.25y^2) + \exp(-0.05(x-10)^2 - 0.25y^2),
\]

\[
a_1(t) = 2 \cos \left( \frac{\pi t}{32} \right),
\]

\[
a_2(t) = 2 \sin \left( \frac{\pi t}{8} \right) I_{[0,64]},
\]

\[
a_3(t) = 2 \sin \left( \frac{\pi t}{4} \right) I_{[64,96]},
\]

and \(I\) is the indicator function. The data set is created by evaluating \(f(x, y, t)\) for \(x = -50, -49, \ldots, 50\), \(y = -50, -49, \ldots, 50\), and \(t = 0, 1, \ldots, 128\).

Proof of Proposition VI.1

To streamline the notation, we drop the round subscript \(r\) of the factors \(a, b, z\). Also, since convergence in the case where one of the factors \(a^{(n)}, b^{(n)}, z^{(n)}\) becomes zero is trivial, we will only consider the case where the sequence \(\theta^{(n)} = \left( a^{(n)^T}, b^{(n)^T}, z^{(n)^T} \right)^T\) is non-zero.

To characterize the limiting behavior of the iterate sequence \(\{\theta^{(n)}\} = T(\theta^{(n-1)})\), we rely on a simple version of Meyer’s monotone convergence theorem [42].

Theorem VI.1. Let \(f(\theta)\) be a continuous function on a domain \(S\) and \(T(\theta)\) be a continuous map from \(S\) into \(S\) satisfying \(f(T(\theta)) < f(\theta)\) for all \(\theta \in S\) with \(T(\theta) \neq \theta\). Suppose for some initial point \(\theta^{(0)}\) that the set \(L_f(\theta^{(0)}) = \{ \theta \in S : f(\theta) \leq f(\theta^{(0)}) \}\) is compact. Then (a) all cluster points are fixed points of \(T(\theta)\), and (b) \(\lim_{n \to \infty} \| \theta^{(n+1)} - \theta^{(n)} \| = 0\).

Let \(S_0 = \{ \theta : \| a \|_2 \leq 1, \| b \|_2 \leq 1, \| z \|_2 \leq 1 \}\). Recall we seek the solution to the following problem.

\[
\arg \max_{\theta \in S} f(\theta) = \langle \text{vec}(y), Dz \otimes b \otimes a \rangle - \tau \| z \|_1,
\]

where \(S = \{ \theta \in S_0 : a \neq 0 \} \cap \{ \theta \in S : b \neq 0 \} \cap \{ \theta \in S : z \neq 0 \}\).

Both the objective function \(f\) and the algorithm map \(T\) are continuous on \(S\).

Note that the problem

\[
\max_{\| a \|_2 \leq 1} \langle u | a \rangle
\]

has a unique maximizer whenever \(u \neq 0\). Additionally,

\[
\max_{\| z \|_2 \leq 1} \langle f | a \rangle - \tau \| z \|_1
\]

also has a unique maximizer whenever \(f \neq 0\). By Assumption 1, it follows that \(f(T(\theta)) < f(\theta)\) for all \(\theta \in S\) with \(T(\theta) \neq \theta\).

Finally, note that for any \(\theta^{(0)} \in S\), the set \(L_f(\theta^{(0)})\) is closed since \(f\) is continuous on \(S\). Moreover \(L_f(\theta^{(0)})\) is compact since it is a closed subset of compact set \(S_0\). Thus, by Theorem VI.1, the limit points of the BCA iterate sequence \(\theta^{(n)}\) are fixed points of \(T\). Again we will argue that for \(\tau\) sufficiently small, the map \(T\) has fixed points in the set \(S\).

To complete the proof, note that the following optimization problem is equivalent to the one we wish to solve.

\[
\min_{\theta = a, b, z} \langle \text{vec}(y), (Dz^+ + z^-) \otimes b \otimes a \rangle + \tau \langle 1 | z^+ + z^- \rangle
\]

such that

\[
c_1(\theta) = \| a \|_2^2 - 1 \leq 0
\]

\[
c_2(\theta) = \| b \|_2^2 - 1 \leq 0
\]

\[
c_3(\theta) = \| z^+ - z^- \|_2^2 - 1 \leq 0
\]

\[
c_j^T(\theta) = -z_j^+ \leq 0
\]

\[
c_j^-(\theta) = -z_j^- \leq 0
\]

It is tedious but straightforward to verify that the above optimization problem satisfies linear independence constraint qualification (LICQ) for all \(\theta \in S\). Since LICQ holds at non-zero fixed points, one can then identify the fixed points of \(T\) as Karush-Kuhn-Tucker (KKT) points of the above problem.