Multi-way Functional Principal Components Analysis

Genevera I. Allen
Departments of Statistics and Electrical & Computer Engineering,
Rice University; Houston, USA; gallen@rice.edu

***Abstract*—Many examples of multi-way or tensor-valued data, such as in climate studies, neuroimaging, chemometrics, and hyperspectral imaging, are structured meaning that variables are associated with locations. Tensor decompositions, or higher-order principal components analysis (HOPCA), are a classical method for dimension reduction and pattern recognition for this multi-way data. In this paper, we introduce novel methods for *Functional HOPCA* that decompose the tensor data into components that are smooth with respect to the known data structure. Through numerical experiments we demonstrate the comparative advantages of our methods for smooth signal recovery from multi-way data.**

***Keywords*-tensors, higher-order PCA, functional data analysis, Tucker decomposition, CP decomposition.**

I. Introduction

Tensor decompositions, sometimes referred to as Higher-Order Principal Components Analysis (HOPCA), are used for dimension reduction, data compression, pattern recognition, exploratory data analysis, and visualization of multi-way data [1]. With the advent of new technologies, multi-way data is becoming more common; and, several examples of this data are structured, meaning that variables along one or more mode are associated with a location. Consider, for example, multi-subject neuroimaging data such as from fMRI, EEG or MEG that can be arranged as a multi-way array of subjects by brain locations by time points; variables along the latter two modes are clearly structured. Thus, factors that are smooth with respect to the data structure would lead to improved signal recovery and more interpretable factors. For matrix data, several have achieved smooth factors in the context of PCA through regularization, an approach termed Functional PCA (FPCA) [2; 3]. Our objective is to develop a framework that extends Functional PCA to decompose a tensor such that one, two, or all factors are smooth with respect to the known structure.

In this paper, we develop several novel approaches to formulating Functional HOPCA that are based on algorithms in the FPCA or tensor decomposition literature: Functional Higher-Order SVD (HOSVD), an extension of the HOSVD algorithm for Tucker decompositions of [4]; Functional Higher-Order Orthogonal Iteration (HOOI), an extension of the HOOI or Tucker-ALS algorithm of [5]; Functional Tucker, an extension of the half-smoothing approaches to FPCA of [3; 6]; Functional CP-ALS, an extension of the CP-ALS algorithm of [7; 8] for functional factors; and finally, the Functional CP-TPA (Tensor Power Algorithm), an optimization-based approach that is an extension of the two-way FPCA method of [6] using the greedy TPA algorithm of [9; 10]. We present a thorough numerical study to compare these approaches and classical tensor decomposition methods; results indicate that some of our methods offer substantial improvements for signal recovery of smooth underlying higher-order factors.

II. Functional Higher-Order PCA

First, we briefly review notation. Tensors are denoted by $\boldsymbol{\mathcal{X}}$, matrices $\mathbf{X}$, vectors $\mathbf{x}$, and scalars x. For simplicity, we will only consider 3-mode tensors in this paper and always denote the mode dimensions as $\boldsymbol{\mathcal{X}}_{n\times p\times q}$; our results can be easily extended to tensors of higher orders. Matricizing or stacking a tensor along a particular mode is denoted as $\mathbf{X}_{(1)}$ which would be of dimension $n \times pq$. We will also employ several types of multiplication: the outer product, $\mathbf{x} \circ \mathbf{y} = \mathbf{x}\,\mathbf{y}^T$; multiplication along a tensor mode, $\boldsymbol{\mathcal{X}} \times_1 \mathbf{Y}$, denotes regular multiplication of $\mathbf{Y}$ along mode one of $\boldsymbol{\mathcal{X}}$; and the Khatri-Rao product $\mathbf{X} \odot \mathbf{Y}$ is the Kronecker product of the columns of $\mathbf{X}$ and $\mathbf{Y}$.

Suppose we observe data, $\boldsymbol{\mathcal{X}} \in \mathbb{R}^{n\times p\times q}$, and assume that this data arises from a low-rank model, either the CANDECOMP / PARAFAC (CP) model [7; 8]: $\boldsymbol{\mathcal{X}} = \sum_{k=1}^{K} d_k\, \mathbf{u}_k \circ \mathbf{v}_k \circ \mathbf{w}_k + \epsilon$, or the Tucker model [4]: $\boldsymbol{\mathcal{X}} = \boldsymbol{\mathcal{D}} \times_1 \mathbf{U} \times_2 \mathbf{V} \times_3 \mathbf{W} + \epsilon$, where $\boldsymbol{\mathcal{D}}$ is the 3-mode tensor core; in both models, ϵ denotes i.i.d. additive noise. In addition to the classic CP and Tucker models, we assume that one or more of the factors are smooth with respect to the known data structure. Thus, we assume that variables along one or more modes of $\boldsymbol{\mathcal{X}}$ arise as discretized functional curves, for example, pixels in images or time points. Our goal in this paper is to develop methods to estimate one or more smooth tensor factors. As the case of three-way smooth factors is the most challenging, we will develop our methods assuming data along all three tensor modes arise as discretized functional curves; all other combinations of smooth factors will be special cases of this three-way approach.

Before introducing our methods for Functional HOPCA, we first review methods for Functional PCA with matrix data. There are many approaches to encourage smoothness in matrix factorizations including basis expansions and smoothing splines, reproducing kernel Hilbert spaces, and wavelet decompositions, but by far the most popular approach is a direct method using generalized ℓ_2 regularization. First introduced

by [2] and latter modified by [3], this method regularizes the original variance maximization PCA criterion:

$$\underset{\mathbf{v}_k}{\text{maximize}} \ \mathbf{v}_k^T \mathbf{X}^T \mathbf{X} \mathbf{v}_k \ \text{ subject to } \ \mathbf{v}_k^T \mathbf{v}_k + \alpha \mathbf{v}_k^T \mathbf{\Omega} \mathbf{v}_k = 1 \\ \& \ \mathbf{v}_k^T \mathbf{v}_j + \alpha \mathbf{v}_k^T \mathbf{\Omega} \mathbf{v}_j = 0 \ \forall \ j < k.$$

Here, $\mathbf{X}_{n \times p}$ is the data matrix, $\mathbf{v}_k$ is the p length k^{th} FPC factor, $\alpha \geq 0$ is a regularization parameter controlling the amount of smoothness imposed on $\mathbf{v}$, and $\mathbf{\Omega} \succ 0$ is a $p \times p$ matrix that penalizes roughness in $\mathbf{v}$. Examples of $\mathbf{\Omega}$ include the squared second or fourth differences matrix which encourage smoothness by forcing second or fourth adjacent elements in $\mathbf{v}$ to be close to each other in the ℓ_2 sense; several other instances of $\mathbf{\Omega}$ matrices that are often application specific and can be used in this context are given in [11]. This FPCA framework also has a desirable interpretation related to smoothing: Letting $\mathbf{S} = (\mathbf{I} + \alpha \mathbf{\Omega})^{-1}$ be the smoothing matrix, then the above FPC solution can be obtained by half-smoothing the data, $\tilde{\mathbf{X}} = \mathbf{X} \mathbf{S}^{1/2}$, taking the SVD, $\tilde{\mathbf{X}} = \tilde{\mathbf{U}} \tilde{\mathbf{D}} \tilde{\mathbf{V}}$, and then half-smoothing the resulting PC factors, $\mathbf{V} = \mathbf{S}^{1/2} \tilde{\mathbf{V}}$. Later, [6] extended this approach to two-way FPCA assuming both the row and column PC factors are smooth by performing two-way half-smoothing; they also elegantly show the mathematical criteria that is optimized by the two-way half-smoothing procedure.

We are now ready to introduce our first set of methods for Functional HOPCA which are based on the Tucker decomposition model. The two most common algorithms to estimate Tucker models are the Higher-Order SVD (HOSVD) and Higher-Order Orthogonal Iteration (HOOI) or Tucker Alternating Least Squares (ALS) algorithms [4; 5]. Both of these methods proceed by matricizing the tensor along a mode and performing PCA to estimate the Tucker factor for that mode. The HOSVD takes $\mathbf{U}$ as the left singular vectors of $\mathbf{X}_{(1)}$, and takes $\mathbf{V}$ and $\mathbf{W}$ analogously. The HOOI is an iterative procedure that sets $\mathbf{U}$ to the left singular vectors of the matricized, projected tensor, $(\boldsymbol{\mathcal{X}} \times_2 \mathbf{V} \times_3 \mathbf{W})_{(1)}$, and sets $\mathbf{V}$ and $\mathbf{W}$ analogously, iterating until convergence. To estimate smooth Tucker factors then, one could simply replace the SVD / PCA step in both the HOSVD and HOOI algorithms with Functional PCA for matrix data as described above. This yields our so-called *Functional HOSVD* and *Functional HOOI* methods.

Another approach to estimating smooth factors in the Tucker model is to extend the half-smoothing approach of [3; 6]. In particular, one could define a smoothing matrix for each mode, $\mathbf{S}_\mathbf{u}$, $\mathbf{S}_\mathbf{v}$, and $\mathbf{S}_\mathbf{w}$, and perform our so-called *Functional-Tucker* method as follows: Half-smooth the tensor, $\tilde{\boldsymbol{\mathcal{X}}} = \boldsymbol{\mathcal{X}} \times_1 \mathbf{S}_\mathbf{u}^{1/2} \times_2 \mathbf{S}_\mathbf{v}^{1/2} \times_3 \mathbf{S}_\mathbf{w}^{1/2}$, take the Tucker decomposition, $\tilde{\boldsymbol{\mathcal{X}}} = \tilde{\boldsymbol{\mathcal{D}}} \times_1 \tilde{\mathbf{U}} \times_2 \tilde{\mathbf{V}} \times_3 \tilde{\mathbf{W}}$, and then half-smooth each of the Tucker factors, $\mathbf{U} = \mathbf{S}_\mathbf{u}^{1/2} \tilde{\mathbf{U}}$, $\mathbf{V} = \mathbf{S}_\mathbf{v}^{1/2} \tilde{\mathbf{V}}$, $\mathbf{W} = \mathbf{S}_\mathbf{w}^{1/2} \tilde{\mathbf{W}}$. Note that this approach is similar in spirit to the Smoothed Tucker method of [12] which uses B-splines to perform iterative smoothing. While [6] were able to show that two-way half smoothing corresponds to directly optimizing a mathematical criteria, our three-way extension does not. This results from the fact that the Tucker model has a non-diagonal core. Thus, our three methods to estimate Functional HOPCA via Tucker models are algorithmic approaches achieved by extensions of existing tensor algorithms.

We develop another algorithmic approach to Functional HOPCA by extending the commonly used algorithm to estimate the CP decomposition, the CP-ALS algorithm. This method estimates the CP model by iteratively regressing the other factor matrices on the tensor matricized along the mode of interest and then scaling the resulting factor [7; 8]. Since each step is performed by least squares regression, one could simply add a smoothing generalized ℓ_2 penalty as described above to the regression problem to achieve smoothness in the tensor factors. Consider, for example, the multi-factor penalty $\text{tr}(\mathbf{U}^T \mathbf{\Omega}_\mathbf{u} \mathbf{U}) = \sum_{k=1}^K \mathbf{u}_k^T \mathbf{\Omega}_\mathbf{u} \mathbf{u}_k$; then, one could solve the following penalized regression problem to estimate a smooth factor $\mathbf{U}$:

$$\underset{\mathbf{U}}{\text{minimize}} \ \ || \mathbf{X}_{(1)} - \mathbf{U}(\mathbf{V} \odot \mathbf{W})^T ||_F^2 + \alpha_\mathbf{u} \text{tr}(\mathbf{U}^T \mathbf{\Omega}_\mathbf{u} \mathbf{U}). \tag{1}$$

Note that the first term of the criterion is the same as that of the CP-ALS algorithm [7; 8]. The solution to this problem is obtained by the following:

Proposition 1: The solution to (1) is given by the solution to the following Sylvester equation: $\mathbf{U}(\mathbf{V} \odot \mathbf{W})^T (\mathbf{V} \odot \mathbf{W}) + \alpha_\mathbf{u} \mathbf{\Omega}_\mathbf{u} \mathbf{U} = \mathbf{X}_{(1)} (\mathbf{V} \odot \mathbf{W})$.

Thus, our so-called *Functional CP-ALS* algorithm proceeds by iteratively solving the above penalized regression problem (or analogous versions for $\mathbf{V}$ and $\mathbf{W}$), scaling the resulting factors to have column-norm one, and repeating until convergence. While it may seem that this algorithm solves the traditional CP-ALS Frobenius-norm loss with additional penalties on each of the factors, this is not the case due to the scaling step that ensures numerically stable factors. See a discussion in [6] for further details on the analogous result with matrix data.

While we have introduced several new methods for estimating Functional HOPCA, all of these approaches modify existing algorithms and thus, do not directly optimize a unified mathematical objective. Here, we introduce a novel optimization-based method that extends the two-way FPCA methods of [6] to multi-way data and is closely related to the tensor power algorithm of [9; 10]. We consider estimating a rank-one CP model one at a time in a greedy manner:

$$\underset{\mathbf{u},\mathbf{v},\mathbf{w}}{\text{minimize}} \ \ || \boldsymbol{\mathcal{X}} - \mathbf{u} \circ \mathbf{v} \circ \mathbf{w} ||_F^2 - || \mathbf{u} ||_2^2 || \mathbf{v} ||_2^2 || \mathbf{w} ||_2^2 \\ + \mathbf{u}^T \mathbf{S}_\mathbf{u}^{-1} \mathbf{u} \, \mathbf{v}^T \mathbf{S}_\mathbf{v}^{-1} \mathbf{v} \, \mathbf{w}^T \mathbf{S}_\mathbf{w}^{-1} \mathbf{w} \, . \tag{2}$$

This criterion is a direct extension of the two-way FPCA objective in [6]. Notice that this problem is tri-convex problem in the factors, meaning that it is convex in $\mathbf{u}$ with $\mathbf{v}$ and $\mathbf{w}$ fixed and vice versa. Thus, we can consider optimizing (2) one factor at a time. Since the objective is smooth, there exists an analytical solution for each factor as given in Algorithm 1. Additionally, notice that the factor updates are similar to those of the tensor power algorithm (TPA) [9; 10], a higher-order

extension of the power algorithm for computing eigenvalues. Thus, we term this method the *Functional CP-TPA*.

Algorithm 1 Functional CP-TPA Algorithm

1) Initialize $\hat{\boldsymbol{\mathcal{X}}} = \boldsymbol{\mathcal{X}}$ and norm-one vectors $\mathbf{u}$, $\mathbf{v}$, and $\mathbf{w}$.
2) For $k = 1 \dots K$
   a) Repeat until converge:
      i) $\mathbf{u}_k \leftarrow \dfrac{\mathbf{S}_\mathbf{u}(\hat{\boldsymbol{\mathcal{X}}} \times_2 \mathbf{v}_k \times_3 \mathbf{w}_k)}{\mathbf{v}_k^T \mathbf{S}_\mathbf{v}^{-1} \mathbf{v}_k \, \mathbf{w}_k^T \mathbf{S}_\mathbf{w}^{-1} \mathbf{w}_k}$.
      ii) $\mathbf{v}_k \leftarrow \dfrac{\mathbf{S}_\mathbf{v}(\hat{\boldsymbol{\mathcal{X}}} \times_1 \mathbf{u}_k \times_3 \mathbf{w}_k)}{\mathbf{u}_k^T \mathbf{S}_\mathbf{u}^{-1} \mathbf{u}_k \, \mathbf{w}_k^T \mathbf{S}_\mathbf{w}^{-1} \mathbf{w}_k}$.
      iii) $\mathbf{w}_k \leftarrow \dfrac{\mathbf{S}_\mathbf{w}(\hat{\boldsymbol{\mathcal{X}}} \times_1 \mathbf{u}_k \times_2 \mathbf{v}_k)}{\mathbf{u}_k^T \mathbf{S}_\mathbf{u}^{-1} \mathbf{u}_k \, \mathbf{v}_k^T \mathbf{S}_\mathbf{v}^{-1} \mathbf{v}_k}$.
   b) Scale $\mathbf{u}_k$, $\mathbf{v}_k$, and $\mathbf{w}_k$ to have norm-one.
   c) $d_k \leftarrow \hat{\boldsymbol{\mathcal{X}}} \times_1 \mathbf{u}_k \times_2 \mathbf{v}_k \times_3 \mathbf{w}_k$.
   d) $\hat{\boldsymbol{\mathcal{X}}} \leftarrow \hat{\boldsymbol{\mathcal{X}}} - d_k \, \mathbf{u}_k \circ \mathbf{v}_k \circ \mathbf{w}_k$.

The following result shows that the Functional CP-TPA method indeeds solves the unified objective (2):

Proposition 2: The rank-one Functional CP-TPA algorithm converges to a stationary point of (2).

Proof: Taking the gradient of (2) with respect to each factor, we see that the updates in Step 2 a) solve (2) for each factor. Furthermore, since the objective is tri-concave in the factors and each update is unique, we have that cyclical updates converge to a stationary point [13]. ∎

Since this method solves for a single rank-one CP decomposition at a time, multiple rank models can be estimated in a greedy manner by subtracting the previously estimated decomposition [9; 6; 10]. The simplicity and flexibility of this approach, however, must be weighed against the fact that greedy rank-one approximations can be sub-optimal [14]. Also, notice that while our method converges to a stationary point, the quality of the solution depends heavily on initial values for the factors, as is the case with all tensor decomposition methods [1].

We have presented several novel approaches to incorporating smoothness with respect to known data structure in tensor decompositions using regularization. Overall, as the Functional CP-TPA method solves a mathematical criterion, it offers certain advantages such as guaranteed convergence, optimality assurances, and a method of comparing the quality of different solutions. Computationally, the Functional HOOI and Functional CP-ALS methods may be intractable in high-dimensional settings as they involve repeatedly taking eigenvalue decompositions of matricized tensors or solving Sylvester equations respectively. Also, a consideration for practical use of these methods is a data-driven way to determine the appropriate amount of smoothing imposed on each factor moderated by the regularization parameter, α. Many methods such as cross-validation have been suggested and can be used with any of our methods [3]. But, for the Functional CP-TPA, an extension of the generalized cross-validation (GCV) procedure of [6] can be used which leaves out one tensor fiber at a time. The GCV criterion then has an analytical form that can be estimated in a nested manner within the alternating power updates for each CP factor. Thus, this data-driven method for estimating the regularization parameters has the advantage of being able to impose differing amounts of smoothness on the different tensor modes.

III. Numerical Results

	F-HOSVD	F-HOOI	F-Tucker	F-CP-ALS	F-CP-TPA
$K = 1$					
I	0.104	0.355	0.160	1.402	0.141
III	0.898	2.016	0.870	5.81	1.705
V	0.096	0.400	0.126	2.572	0.392
VI	0.928	2.181	0.932	12.81	3.765
VII	125.8	575.9	117.7	8046	68.68
$K = 2$					
I	0.129	0.690	0.227	3.533	0.269
III	0.963	2.853	1.014	41.77	1.067
V	0.130	0.728	0.242	3.602	0.780
VI	0.995	4.25	1.153	21.75	3.854
VII	122.3	944.2	117.5	8893	102

TABLE I
Mean run time in seconds.

We evaluate the comparative effectiveness of our proposed Functional HOPCA methods through a series of numerical experiments. Three-way data is generated from a rank-one or rank-two CP model: $\boldsymbol{\mathcal{X}} = \sum_k d_k \, \mathbf{u}_k \circ \mathbf{v}_k \circ \mathbf{w}_k + \epsilon$ for $\epsilon \sim \; i.i.d. \; N(0, 1)$ and signal strength $d_1 = 50$ for $K = 1$ or $d_{1:2} = [60 \; 40]$ for $K = 2$. (Note that we simulate from a CP model for comparison purposes as the Tucker model is a special case with diagonal tensor core.) The norm-one signal factors and data-dimensions varying according to seven different scenarios: Data-dimensions are either $100 \times 100 \times 100$ (scenarios I, III, V), $1000 \times 50 \times 50$ (scenarios II, IV, VI), or $5000 \times 50 \times 50$ (scenario VII). Scenarios I - IV and VII consider the case where only $\mathbf{u}$ is smooth, with $\mathbf{u}$ either a sinusoidal curve (I, II, VII) or a Gaussian pulse (III, IV); $\mathbf{v}$ and $\mathbf{w}$ are generated as random orthonormal vectors. Scenarios V and VI consider the case where all factors are smooth: $\mathbf{u}$ a sinusoidal curve, $\mathbf{v}$ a Gaussian pulse, and $\mathbf{w}$ a product of two sinusoidal curves of different phases. For the rank-two models, the orthogonal signal to the specified curve (i.e. a cosine curve) was taken as the rank-two factor. Our roughness penalty, $\boldsymbol{\Omega}$ was taken as the squared second differences matrix, as commonly used for smoothing of equi-spaced discretized functional curves [11]. As all of our Functional HOPCA methods have a tuning parameter, α, that controls the amount of smoothing, we took $\alpha = n[.001, .01, .1, 1, 10]$ where n is the appropriate data-dimension; results for the best performing value of α are reported for each method.

In Table II, we report comparative results in terms of subspace recovery for six simulation scenarios. We compare the true generating subspace to that estimated by the Tucker decomposition, CP decomposition, and our Functional HOSVD, HOOI, Tucker, CP-ALS, and CP-TPA methods. Mean and median results for 50 replicates are reported for subspace recovery measured by $1 - \cos(\angle(\hat{\mathbf{U}}, \mathbf{U}^*))$, where $\angle$ denotes the principal angle between $\hat{\mathbf{U}}$ and $\mathbf{U}^*$; thus, smaller numbers indicate better signal recovery. Results indicate that all functional HOPCA methods offer substantial improvements

		CP	Tucker	F-HOSVD	F-HOOI	F-Tucker	F-CP-ALS	F-CP-TPA
$K = 1$								
I	**u**	0.858 / 0.902	0.844 / 0.915	0.0466 / 0.0471	0.0192 / 0.0194	0.549 / 0.772	0.145 / 0.0175	0.0428 / 0.00687
II	**u**	0.879 / 0.901	0.854 / 0.924	0.0593 / 0.0602	0.03 / 0.0301	0.702 / 0.857	0.238 / 0.0185	0.196 / 0.00946
III	**u**	0.963 / 0.968	0.97 / 0.98	0.0816 / 0.0812	0.0708 / 0.0715	0.66 / 0.867	0.14 / 0.0304	0.0727 / 0.0168
IV	**u**	0.968 / 0.972	0.957 / 0.966	0.0711 / 0.0707	0.0631 / 0.0627	0.504 / 0.781	0.0677 / 0.0285	0.0468 / 0.0123
V	**u**	0.889 / 0.91	0.913 / 0.924	0.0454 / 0.0451	0.0191 / 0.0188	0.481 / 0.721	0.106 / 0.0192	0.0071 / 0.00702
	v	0.918 / 0.951	0.936 / 0.957	0.0603 / 0.06	0.0294 / 0.0294	0.563 / 0.842	0.109 / 0.0201	0.0081 / 0.0081
	w	0.905 / 0.935	0.913 / 0.932	0.0571 / 0.056	0.0262 / 0.0263	0.522 / 0.734	0.115 / 0.0207	0.0101 / 0.00983
VI	**u**	0.972 / 0.978	0.978 / 0.982	0.0824 / 0.0816	0.0717 / 0.072	0.869 / 0.924	0.172 / 0.0469	0.0746 / 0.0167
	v	0.934 / 0.957	0.931 / 0.937	0.926 / 0.939	0.0539 / 0.0539	0.913 / 0.963	0.141 / 0.0104	0.0697 / 0.0126
	w	0.931 / 0.936	0.927 / 0.932	0.103 / 0.0941	0.0198 / 0.0192	0.847 / 0.906	0.141 / 0.0101	0.0627 / 0.0132
$K = 2$								
I	**u**	0.908 / 0.946	0.917 / 0.948	0.0393 / 0.0357	0.0108 / 0.0106	0.581 / 0.757	0.344 / 0.0299	0.105 / 0.0114
II	**u**	0.923 / 0.935	0.938 / 0.946	0.898 / 0.92	0.0183 / 0.0176	0.826 / 0.911	0.318 / 0.0299	0.217 / 0.019
III	**u**	0.981 / 0.985	0.978 / 0.98	0.0748 / 0.0738	0.0669 / 0.0294	0.601 / 0.886	0.333 / 0.055	0.176 / 0.0303
IV	**u**	0.98 / 0.983	0.975 / 0.981	0.0332 / 0.0322	0.0575 / 0.02	0.547 / 0.759	0.39 / 0.0561	0.0375 / 0.02
V	**u**	0.936 / 0.952	0.939 / 0.947	0.0746 / 0.0703	0.0235 / 0.0233	0.302 / 0.0127	0.307 / 0.0309	0.0109 / 0.0109
	v	0.951 / 0.963	0.952 / 0.955	0.089 / 0.0891	0.0236 / 0.0229	0.321 / 0.0184	0.317 / 0.032	0.0159 / 0.0155
	w	0.943 / 0.95	0.939 / 0.951	0.0928 / 0.0904	0.0242 / 0.0235	0.315 / 0.0184	0.316 / 0.0325	0.017 / 0.0172
VI	**u**	0.983 / 0.983	0.979 / 0.984	0.0983 / 0.0989	0.132 / 0.0583	0.841 / 0.897	0.323 / 0.062	0.267 / 0.0319
	v	0.957 / 0.96	0.955 / 0.965	0.944 / 0.957	0.0908 / 0.0148	0.881 / 0.936	0.286 / 0.0174	0.275 / 0.0372
	w	0.951 / 0.958	0.945 / 0.949	0.319 / 0.225	0.0855 / 0.0121	0.818 / 0.873	0.282 / 0.0186	0.239 / 0.0139

TABLE II
MEAN / MEDIAN SUBSPACE RECOVERY SIMULATION RESULTS.

over the CP and Tucker decompositions for signal recovery of smooth multi-way factors. Out of our methods, the Functional-Tucker has markedly worse performance, along with the Functional HOSVD in a couple of the simulation scenarios. The Functional HOOI and CP-TPA methods consistently have the best performance across all scenarios. Interestingly, there is often a discrepancy between the mean and median subspace recovery results for the Functional CP-ALS and CP-TPA methods. Inspection of the replicates indicate that a handful of outlying poor results influence the reported mean. These occur as random initializations were used for both methods, and poor starting points can lead to sub-optimal results. As the Functional CP-TPA method solves a mathematical objective, one can simply run this algorithm from different initializations and take the solution yielding the best objective value.

Finally, we compare our proposed methods for Functional HOPCA in terms of computational time. All methods were programmed solely in Matlab and consistent criteria were used to establish convergence. Table I reports the mean run time in seconds over five trials for each of our methods in the indicated simulation scenarios. Recall that the Functional HOOI and CP-TPA methods perform best in terms of subspace recovery; between these, Functional-CP-TPA offers substantially reduced computational time, especially for the high-dimensional scenario VII.

Overall, results demonstrate that our methods, especially the Functional HOOI and Functional CP-TPA methods, are better able to recover smooth underlying tensor factors than competing tensor decomposition methods. We recommend, however, the Functional CP-TPA method as it solves a coherent mathematical objective and is computationally the most efficient method.

ACKNOWLEDGMENTS

This work is supported by NSF DMS-1209017.

REFERENCES

[1] T. Kolda and B. Bader, "Tensor decompositions and applications," *SIAM Review*, vol. 51, no. 3, pp. 455–500, 2009.

[2] J. A. Rice and B. W. Silverman, "Estimating the mean and covariance structure nonparametrically when the data are curves," *J. of the Royal Statistical Society, Series B*, pp. 233–243, 1991.

[3] B. Silverman, "Smoothed functional principal components analysis by choice of norm," *The Annals of Statistics*, vol. 24, no. 1, pp. 1–24, 1996.

[4] L. Tucker, "Some mathematical notes on three-mode factor analysis," *Psychometrika*, vol. 31, no. 3, pp. 279–311, 1966.

[5] L. De Lathauwer, B. De Moor, and J. Vandewalle, "A multilinear singular value decomposition," *SIAM J. on Matrix Analysis and Applications*, vol. 21, no. 4, pp. 1253–1278, 2000.

[6] J. Huang, H. Shen, and A. Buja, "The analysis of two-way functional data using two-way regularized singular value decompositions," *Journal of the American Statistical Association*, vol. 104, no. 488, pp. 1609–1620, 2009.

[7] J. Carroll and J. Chang, "Analysis of individual differences in multidimensional scaling via an n-way generalization of eckart-young decomposition," *Psychometrika*, vol. 35, no. 3, pp. 283–319, 1970.

[8] R. Harshman, "Foundations of the parafac procedure: Models and conditions for an" explanatory" multimodal factor analysis," 1970, uCLA Working Papers in Phonetics.

[9] L. De Lathauwer, B. De Moor, and J. Vandewalle, "On the best rank-1 and rank-(r 1, r 2,..., rn) approximation of higher-order tensors," *SIAM Journal on Matrix Analysis and Applications*, vol. 21, no. 4, pp. 1324–1342, 2000.

[10] G. I. Allen, "Sparse higher-order principal components analysis," in *AISTATS*, vol. 15, 2012.

[11] J. O. Ramsay, *Functional Data Analysis*. Springer, 2005.

[12] M. E. Timmerman and H. A. Kiers, "Three-way component analysis with smoothness constraints," *Computational statistics & data analysis*, vol. 40, no. 3, pp. 447–470, 2002.

[13] P. Tseng, "Convergence of a block coordinate descent method for nondifferentiable minimization," *Journal of optimization theory and applications*, vol. 109, no. 3, pp. 475–494, 2001.

[14] A. Stegeman and P. Comon, "Subtracting a best rank-1 approximation may increase tensor rank," *Linear Algebra and its Applications*, vol. 433, no. 7, pp. 1276–1300, 2010.