ANNUAL REVIEWS

Annual Review of Statistics and Its Application

Tensors in Statistics

Xuan Bi,[1,*] Xiwei Tang,[2,*] Yubai Yuan,[3,*] Yanqing Zhang,[4,*] and Annie Qu[3,*]

[1]Information and Decision Sciences, Carlson School of Management, University of Minnesota, Minneapolis, Minnesota 55455, USA

[2]Department of Statistics, University of Virginia, Charlottesville, Virginia 22904, USA

[3]Department of Statistics, University of California, Irvine, California 92697, USA; email: aqu2@uci.edu

[4]Department of Statistics, Yunnan University, Kunming 650504, China

Annu. Rev. Stat. Appl. 2021. 8:345–68

First published as a Review in Advance on August 14, 2020

The *Annual Review of Statistics and Its Application* is online at statistics.annualreviews.org

https://doi.org/10.1146/annurev-statistics-042720-020816

*These authors contributed equally to this article

Keywords

high-order networks, imaging analyses, recommender systems, tensor applications, tensor properties

Abstract

This article provides an overview of tensors, their properties, and their applications in statistics. Tensors, also known as multidimensional arrays, are generalizations of matrices to higher orders and are useful data representation architectures. We first review basic tensor concepts and decompositions, and then we elaborate traditional and recent applications of tensors in the fields of recommender systems and imaging analysis. We also illustrate tensors for network data and explore the relations among interacting units in a complex network system. Some canonical tensor computational algorithms and available software libraries are provided for various tensor decompositions. Future research directions, including tensors in deep learning, are also discussed.

1. INTRODUCTION

In this article, we provide a review of tensors in statistics and its applications. Tensors are generalizations of matrices for higher-order data and provide useful data representation formats. Tensors originally appeared in 1927 (Hitchcock 1927). Fueled by increased computing capacity during the past decade, tensors have expanded to many domains, including statistics, data science, and machine learning (Kolda & Bader 2009). For example, in context-aware recommender systems (CARS), users tend to interact with different items (e.g., movies or products) with various preferences depending on different contexts (e.g., time or promotion strategies). Such events represent different behavior patterns of users under different situations and can be effectively modeled by a third-order tensor as *users* × *items* × *context*; modeling could be challenging using traditional data formats.

In this article, we first introduce basic tensor concepts and notations, which provide a foundation for the upcoming sections. Specifically, we introduce some of the most widely used tensor decompositions, including canonical decomposition/parallel factor analysis (CANDECOMP/PARAFAC; hereafter, CP) decomposition (Kiers 2000) and Tucker decomposition (Tucker 1966), and their important properties and applications. Then, we elaborate on dynamic tensors, their modeling, and their key properties. Commonly used algorithms for CP and Tucker decompositions are illustrated, and relevant software is summarized in the **Supplemental Appendix**.

Supplemental Material >

Next, we provide traditional and recent applications of tensors in the field of recommender systems. As a personalization marketing strategy that has been successfully implemented in the retail and entertainment industries for about 20 years, recommender systems are now relied upon by companies and individual customers. Tensors, as flexible tools for data storage, arrangement, and analyses, are considered one of the most important techniques for modern recommender systems. We review the utilization of tensors mainly from two aspects: tensors in CARS (Adomavicius & Tuzhilin 2011) and tensor completion as a theoretical and fundamental statistical framework.

Imaging analysis is another major area in which tensor-based models have successful and widespread applications. This is because imaging data, such as magnetic resonance imaging (MRI) and functional magnetic resonance imaging (fMRI), are naturally stored in a multidimensional array such as a matrix (e.g., a two-dimensional MRI image), a cube (e.g., a three-dimensional MRI image), or a fourth-order tensor (e.g., a three-dimensional fMRI image over time). In biomedical studies, imaging data can be utilized to better understand human physiology, such as brain activity and tumor development, associated with biological, psychological, and clinical traits. However, the sheer size and complexity of medical imaging data pose unprecedented challenges to classical statistical methods, which are mostly based on vectorized data. Therefore, tensor-based models that preserve the higher-order structure of imaging data have received increasing interest and gained success in recent years. In this article, we review various applications and recent developments of tensor-based models in biomedical imaging analysis.

Another area of tensor application is in network data, which has arisen as one of the most common forms of information collection due to the high demand for collecting relations or associations among interacting units from observations. In many applications, the relations to be considered are complex in the sense that they are typically high order, multitype, and multirelational. For example, a biological interaction or process in biological networks involves multiple participating units, such as proteins or genes. Also, in social networks, there usually exist multiple types of social relations among people. The primary challenge in network data analyses is to infer specific network relations through modeling and learning based on observed complex relational data.

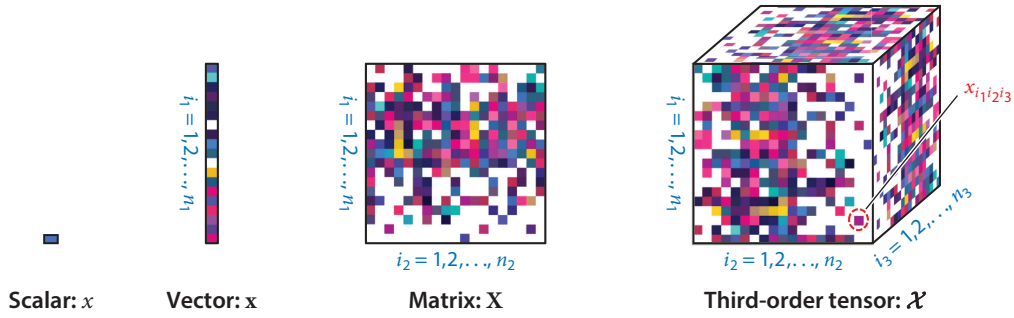

Figure 1

Arrays with different dimensions corresponding to a scalar, a vector, a matrix, and a third-order tensor, where different colors represent different values.

In traditional network analyses, the network is generally formulated as an adjacency matrix, which is mainly applicable for homogeneous or pairwise relations. Tensors, viewed as high-order generalizations of adjacency matrices, offer a more flexible framework to represent high-order relations in modeling network data. More importantly, the tensor technique provides great flexibility in modeling the heterogeneous relations with various types and orders on relations. In this article, we review the tensor applications of two important tasks on complex networks: link prediction and node clustering.

This article is organized as follows. Section 2 introduces the notation and background of tensors. Section 3 presents tensor applications in recommender systems. Section 4 reviews cutting-edge tensor applications in biomedical imaging analyses. Section 5 illustrates tensor applications in networks. In Section 6, we provide principal algorithms for tensor computations. Concluding remarks and discussions of future directions are summarized in Section 7.

2. NOTATION AND BACKGROUND

Tensors can be regarded as multiway collections of data. For example, the simplest tensor that would be a three-dimensional array can be considered as a data cube. In this section, we introduce the notation and different concepts about tensors.

2.1. Basics

A tensor is defined as a multidimensional array; specifically, a dth-order tensor is an array with d dimensions, denoted by Euler script letter $\mathcal{X} \in \mathbb{R}^{n_1 \times n_2 \times \cdots \times n_d}$, which is an extension of a matrix to higher order. Here, d is the number of dimensions of the array $\mathcal{X}$ representing the tensor's order, also known as ways or modes. The n_k is the marginal dimension of the kth mode ($k = 1, 2, \ldots, d$). **Figure 1** shows the arrays from a scalar to a third-order tensor. The $(i_1, i_2, \ldots, i_d)$th element of the tensor $\mathcal{X}$ is denoted by $x_{i_1 i_2 \cdots i_d}$ for $i_k = 1, 2, \ldots, n_k$ and $k = 1, 2, \ldots, d$. We can create subarrays by fixing some of the given tensor's indices. A fiber of a tensor is a vector created by fixing all but one index of a particular mode. A third-order tensor $\mathcal{X}$ has column, row, and tube fibers, denoted by $\mathbf{x}_{:i_2 i_3}$, $\mathbf{x}_{i_1 : i_3}$, and $\mathbf{x}_{i_1 i_2 :}$, respectively. A slice of a tensor is a two-dimensional matrix, defined by fixing all but the indices of two particular modes. **Figure 2** shows the horizontal, lateral, and frontal slides of a third-order tensor $\mathcal{X}$, denoted by $\mathbf{X}_{i_1 ::}$, $\mathbf{X}_{:i_2 :}$, and $\mathbf{X}_{::i_3}$, respectively.

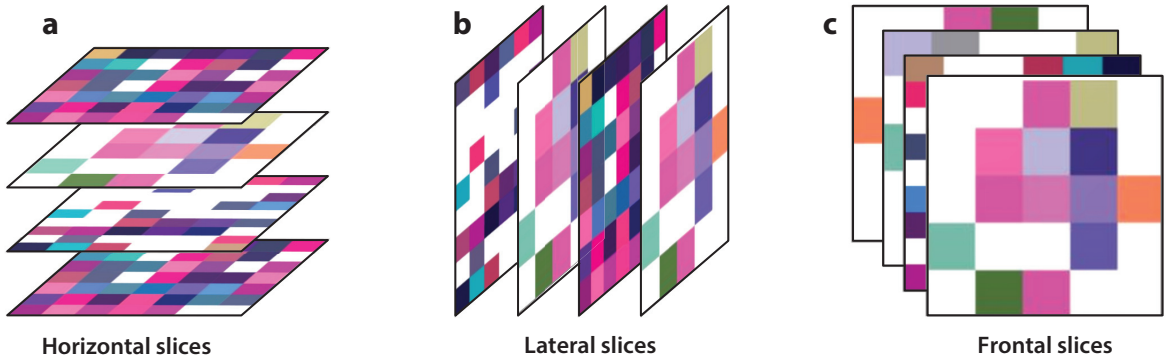

Figure 2

Slices of a third-order tensor $\mathcal{X}$: (*a*) horizontal slices $\mathbf{X}_{i_1::}$, (*b*) lateral slices $\mathbf{X}_{:i_2:}$, and (*c*) frontal slices $\mathbf{X}_{::i_3}$.

The norm of a tensor $\mathcal{X} \in \mathbb{R}^{n_1 \times n_2 \times \cdots \times n_d}$ is the square root of the sum of the squares of all elements, i.e.,

$$\|\mathcal{X}\| = \sqrt{\sum_{i_1=1}^{n_1} \sum_{i_2=1}^{n_2} \cdots \sum_{i_d=1}^{n_d} (x_{i_1 i_2 \cdots i_d})^2}.$$

The inner product of two tensors $\mathcal{X}, \mathcal{Y} \in \mathbb{R}^{n_1 \times n_2 \times \cdots \times n_d}$ with the same size is the sum of the products of their entries, i.e.,

$$\langle \mathcal{X}, \mathcal{Y} \rangle = \sum_{i_1=1}^{n_1} \sum_{i_2=1}^{n_2} \cdots \sum_{i_d=1}^{n_d} x_{i_1 i_2 \cdots i_d} y_{i_1 i_2 \cdots i_d},$$

implying that $\langle \mathcal{X}, \mathcal{X} \rangle = \|\mathcal{X}\|^2$. Moreover, an outer product, $\circ$, operating on multiple vectors $\mathbf{p}^k \in \mathbb{R}^{n_k}$ $(k = 1, 2, \ldots, d)$ creates a dth-order tensor $\mathcal{X} = \sum_{j=1}^{r} \mathbf{p}_j^1 \circ \mathbf{p}_j^2 \circ \cdots \circ \mathbf{p}_j^d$, where r is the rank of the tensor $\mathcal{X}$. When $r = 1$, the tensor $\mathcal{X}$ is a rank-1 tensor, that is, $\mathcal{X} = \mathbf{p}^1 \circ \mathbf{p}^2 \circ \cdots \circ \mathbf{p}^d$, where $\mathbf{p}^k = (p_1^k, \ldots, p_{n_k}^k)^\top$. Each element of the tensor $\mathcal{X}$ is the product of the corresponding vector elements: $x_{i_1 i_2 \cdots i_d} = p_{i_1}^1 p_{i_2}^2 \cdots p_{i_d}^d$ for $i_k = 1, 2, \ldots, n_k$ and $k = 1, 2, \ldots, d$, and $\mathbf{p}^k = (p_1^k, \ldots, p_{n_k}^k)^\top$ is the factor vector at the kth mode. The n_t-mode product of a tensor $\mathcal{X} \in \mathbb{R}^{n_1 \times n_2 \times \cdots \times n_d}$ with a matrix $\mathbf{A} \in \mathbb{R}^{n_t \times m}$ is denoted by $\mathcal{X} \times_t \mathbf{A}$ and is of size $n_1 \times \cdots \times n_{t-1} \times m \times n_{t+1} \times \cdots n_d$. Elementwise, we have $(\mathcal{X} \times_t \mathbf{A})_{i_1 \cdots i_{t-1} j i_{t+1} \cdots i_d} = \sum_{i_t=1}^{n_t} x_{i_1 \cdots i_{t-1} i_t i_{t+1} \cdots i_d} a_{j i_t}$ for $j = 1, 2, \ldots, m$.

We can also transform a tensor into a matrix via reordering the elements of a dth-order tensor, known as matricization, unfolding or flattening. There are many methods of reordering the elements into a matrix. Here, we introduce a special case of mode-k matricization. More details can be found in Kolda (2006). The mode-k matricization of a tensor $\mathcal{X} \in \mathbb{R}^{n_1 \times n_2 \times \cdots \times n_d}$ is denoted by $\mathbf{X}_{(k)}$ for $k = 1, 2, \ldots, d$ and arranges the mode-k fibers to be the columns of $\mathbf{X}_{(k)}$. The element $(i_1, i_2, \ldots, i_d)$ of tensor $\mathcal{X}$ corresponds to the element (i_k, j) of $\mathbf{X}_{(k)}$, where $j = 1 + \sum_{t=1, t \neq k}^{d} (i_t - 1) J_t$ with $J_t = \prod_{m=1, m \neq k}^{t-1} n_m$. Here, we present an example of the matricization of a third-order tensor from Kolda & Bader (2009) for illustration. Let the frontal slices of $\mathcal{X} \in \mathbb{R}^{2 \times 3 \times 2}$ be

$$\mathbf{X}_{::1} = \begin{pmatrix} 1 & 3 & 5 \\ 2 & 4 & 6 \end{pmatrix} \quad \text{and} \quad \mathbf{X}_{::2} = \begin{pmatrix} 7 & 9 & 11 \\ 8 & 10 & 12 \end{pmatrix}.$$

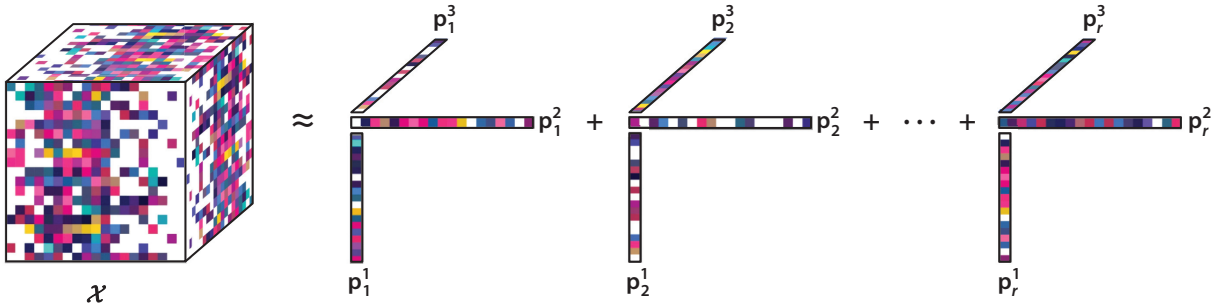

Figure 3

Canonical decomposition/parallel factor analysis (CANDECOMP/PARAFAC, or CP) decomposition for a third-order tensor: $\mathcal{X} \approx \sum_{j=1}^{r} \mathbf{p}_j^1 \circ \mathbf{p}_j^2 \circ \mathbf{p}_j^3$.

Then the three mode-k matricizations are

$$\mathbf{X}_{(1)} = \begin{pmatrix} 1 & 3 & 5 & 7 & 9 & 11 \\ 2 & 4 & 6 & 8 & 10 & 12 \end{pmatrix}, \ \mathbf{X}_{(2)} = \begin{pmatrix} 1 & 2 & 7 & 8 \\ 3 & 4 & 9 & 10 \\ 5 & 6 & 11 & 12 \end{pmatrix}, \text{ and } \mathbf{X}_{(3)} = \begin{pmatrix} 1 & 2 & 3 & 4 & 5 & 6 \\ 7 & 8 & 9 & 10 & 11 & 12 \end{pmatrix}.$$

2.2. Tensor Decomposition

Tensor decomposition refers to expressing a tensor as a sequence of elementary operations on other simple arrays, which has applications in many areas, such as signal processing, recommender systems, neuroimaging, and network analysis. Two major tensor decompositions are CP decomposition (Kiers 2000) and Tucker decomposition (Tucker 1966). There are many other tensor decompositions, including CANDELINC (canonical decomposition with linear constraints) (Carroll et al. 1980), DEDICOM (decomposition into direct components) (Harshman 1978), and PARATUCK2 (parallel factor analysis and Tucker 2 decomposition) (Harshman & Lundy 1996), as well as nonnegative variants of these. We introduce CP decomposition and Tucker decomposition as follows. Extensive references can be found in the review articles by McCullagh (1987), Kolda (2006), and Kolda & Bader (2009).

2.2.1. CANDECOMP/PARAFAC decomposition. The CP decomposition is a type of rank decomposition and approximates a tensor into a sum of component rank-1 tensors. That is,

$$\mathcal{X} \approx \sum_{j=1}^{r} \mathbf{p}_j^1 \circ \mathbf{p}_j^2 \circ \cdots \circ \mathbf{p}_j^d \equiv [\![\mathbf{P}^1, \mathbf{P}^2, \ldots, \mathbf{P}^d]\!], \qquad 1.$$

where the $\mathbf{p}_j^k$s ($j = 1, 2, \ldots, r$) are n_k-dimensional vectors ($k = 1, 2, \ldots, d$), and the factor matrices $\mathbf{P}^k = (\mathbf{p}_1^k, \mathbf{p}_2^k, \ldots, \mathbf{p}_r^k)$ ($k = 1, 2, \ldots, d$). That is, $\text{rank}(\mathcal{X}) = \min\{r : \mathcal{X} \approx \sum_{j=1}^{r} \mathbf{p}_j^1 \circ \mathbf{p}_j^2 \circ \cdots \circ \mathbf{p}_j^d\}$. Equation 1 is illustrated in **Figure 3**. An element of $\mathcal{X}$ is written according to the CP decomposition as $x_{i_1 i_2 \cdots i_d} \approx \sum_{j=1}^{r} p_{i_1 j}^1 p_{i_2 j}^2 \cdots p_{i_d j}^d$.

The CP decomposition is unique under certain conditions, where uniqueness of the CP decomposition provides an only possible combination of rank-1 tensors that sums to $\mathcal{X}$, with the exception of the elementary indeterminacies of scaling and permutation. The permutation indeterminacy means that the rank-1 component tensors can be reordered arbitrarily. That is, $\mathcal{X} = [\![\mathbf{P}^1, \mathbf{P}^2, \ldots, \mathbf{P}^d]\!] = [\![\mathbf{P}^1\Pi, \mathbf{P}^2\Pi, \ldots, \mathbf{P}^d\Pi]\!]$ for any $r \times r$ permutation matrix Π. The scaling indeterminacy refers to scaling the individual vectors, that is, $\mathcal{X} = \sum_{j=1}^{r} (\alpha_j^1 \mathbf{p}_j^1) \circ (\alpha_j^2 \mathbf{p}_j^2) \circ \cdots \circ (\alpha_j^d \mathbf{p}_j^d)$

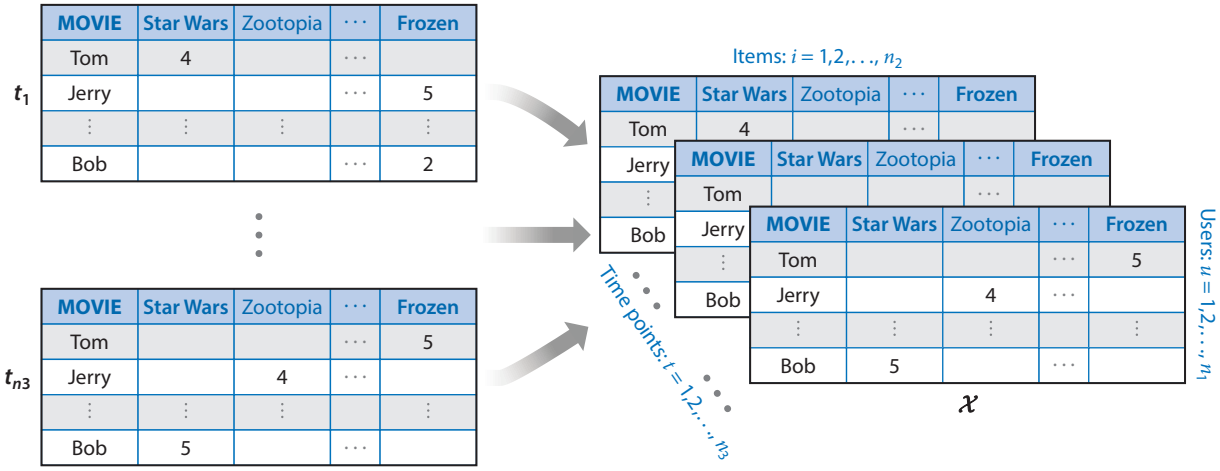

Figure 4

Rating of movies provided by users at different time points as data in a tensor $\mathcal{X}$; for example, the rating score of the movie (or item) i provided by the user u at the time point t can be used as the (u, i, t)th element x_{uit} of the tensor $\mathcal{X}$.

as long as $\prod_{k=1}^{d} \alpha_j^k = 1$ for $j = 1, \ldots, r$. A sufficient condition for uniqueness (Kruskal 1977, 1989; Sidiropoulos & Bro 2000) is

$$\sum_{k=1}^{d} K_{\mathbf{P}^k} \geqslant 2r + (d-1),$$

where $K_{\mathbf{P}^k}$ is the K-rank of the matrix $\mathbf{P}^k$, satisfying $K_{\mathbf{P}^k} = \max\{K : \text{any } K \text{ columns of } \mathbf{P}^k \text{ are linearly independent}\}$. Ten Berge & Sidiropoulos (2002) show that the sufficient condition is also necessary for tensors with rank $r = 2$ and $r = 3$, but not for tensors with rank $r > 3$. More general necessary conditions for uniqueness of CP decomposition can be found in the framework of Liu & Sidiropoulos (2001).

The CP decomposition has been widely used in many fields, such as recommender systems (Xiong et al. 2010, Bi et al. 2018), medical imaging (Karahan et al. 2015, Li et al. 2018, Tang et al. 2019), and networks (Mahyari et al. 2016). For example, movie recommender systems often deal with rating data provided by users or audiences for different movies watched at different time points as tensor data (see **Figure 4**).

2.2.2. Tucker decomposition. The Tucker decomposition is a higher-order form of principal component analysis (PCA) and decomposes a tensor into a core tensor multiplied by a matrix along each mode. That is, a tensor $\mathcal{X} \in \mathbb{R}^{n_1 \times n_2 \times \cdots \times n_d}$ is expressed as

$$\mathcal{X} \approx \mathcal{C} \times_1 \mathbf{Q}^1 \times_2 \mathbf{Q}^2 \cdots \times_d \mathbf{Q}^d = \sum_{j_1=1}^{m_1} \sum_{j_2=1}^{m_2} \cdots \sum_{j_d=1}^{m_d} c_{j_1 j_2 \ldots j_d} \mathbf{q}_{j_1}^1 \circ \mathbf{q}_{j_2}^2 \circ \cdots \mathbf{q}_{j_d}^d \equiv [\![\mathcal{C}; \mathbf{Q}^1, \mathbf{Q}^2, \ldots, \mathbf{Q}^d]\!], \tag{2.}$$

where $\mathbf{Q}^k \in \mathbb{R}^{n_k \times m_k}$ $(k = 1, \ldots, d)$ are the factor matrix and are usually orthogonal, and $\mathbf{q}_{j_k}^k \in \mathbb{R}^{n_k}$ can be treated as the principal components in each mode. All $c_{j_1 j_2 \ldots j_d}$s form a tensor $\mathcal{C} \in \mathbb{R}^{m_1 \times m_2 \times \cdots \times m_d}$, which is called the core tensor and is illustrated in **Figure 5**. An element of $\mathcal{X}$ is represented by the Tucker decomposition as

$$x_{i_1 i_2 \cdots i_d} \approx \sum_{j_1=1}^{m_1} \sum_{j_2=1}^{m_2} \cdots \sum_{j_d=1}^{m_d} c_{j_1 j_2 \ldots j_d} q_{i_1 j_1}^1 q_{i_2 j_2}^2 \cdots q_{i_d j_d}^d.$$

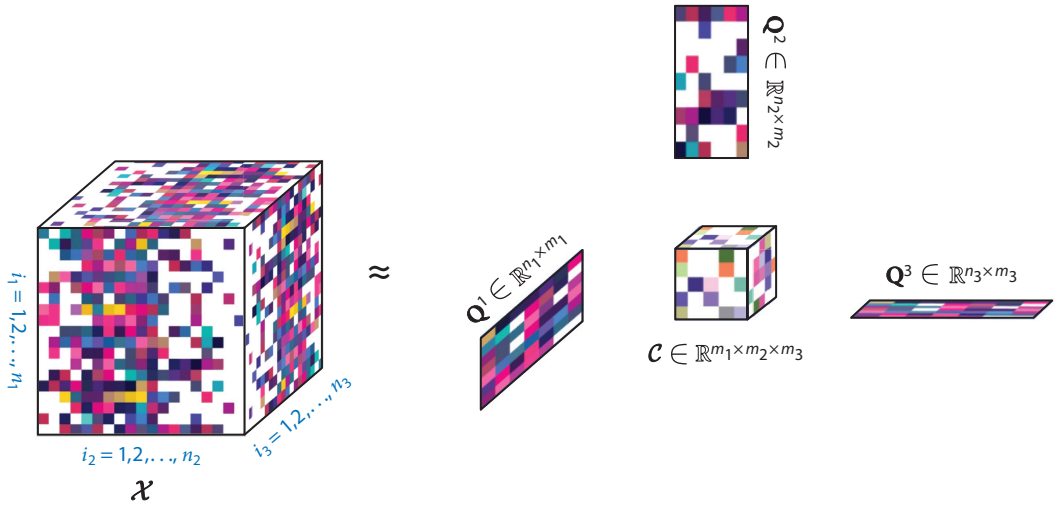

Figure 5

Tucker decomposition for a third-order tensor $\mathcal{X}$, where $\mathcal{C} \in \mathbb{R}^{m_1 \times m_2 \times m_3}$ is the core tensor, and $\mathbf{Q}^k \in \mathbb{R}^{n_k \times m_k}$ $(k = 1, 2, 3)$ are the factor matrices.

In contrast to the CP decomposition, the Tucker decomposition is generally not unique. This is because the core tensor $\mathcal{C}$ can be structured arbitrarily and might allow interactions between any components; that is, $\mathcal{X} \approx [\![\mathcal{C} \times_1 \mathbf{U}_1 \cdots \times_d \mathbf{U}_d; \mathbf{Q}^1\mathbf{U}_1^{-1}, \ldots, \mathbf{Q}^d\mathbf{U}_d^{-1}]\!]$, where $\mathbf{U}_k \in \mathbb{R}^{m_k}$ is any nonsingular matrices for $k = 1, \ldots, d$. Imposing additional constraints on the structure of $\mathcal{C}$ can generally lead to more relaxed uniqueness properties.

There are many applications for Tucker decompositions—for example, facial image recognition (Vasilescu & Terzopoulos 2002) and human motion signatures recognition (Vasilescu 2002). A concrete application of Tucker decomposition is analyzing the multifactor structure of facial image ensembles (Vasilescu & Terzopoulos 2002). In this example, there is a face database of 28 male subjects photographed in 5 different poses, 3 illuminations, and 3 expressions, where each original image has 512 × 352 pixels. Using a global rigid optical flow algorithm (Vasilescu & Terzopoulos 2002), the original images are aligned to one reference image and then cropped as a total of 7,943 pixels per image within an elliptical cropping window (illustrated in **Figure 6**). Hence, the database can be constructed as a fifth-order tensor $\mathcal{X}$, e.g., $\mathcal{X}$ is a 28 × 5 × 3 × 3 × 7,943 tensor. Tucker decomposition is able to capture the important features in a compact form, that is, $\mathcal{X} \approx \mathcal{C} \times_1 \mathbf{Q}_{people} \times_2 \mathbf{Q}_{poses} \times_3 \mathbf{Q}_{illum} \times_4 \mathbf{Q}_{express} \times_5 \mathbf{Q}_{pixels}$, where *illum* indicates illuminations, and *express* indicates expressions. The mode matrices $\mathbf{Q}_{people}$, $\mathbf{Q}_{poses}$, $\mathbf{Q}_{illum}$, $\mathbf{Q}_{express}$, and $\mathbf{Q}_{pixels}$ can be interpreted as principal components. By the core tensor $\mathcal{C}$, the eigenimages present in $\mathbf{Q}_{pixels}$ can be transformed into eigenmodes representing the principal axes of variations across the various factors (people, poses, illuminations, expressions) by forming $\mathcal{C} \times_5 \mathbf{Q}_{pixels}$.

2.3. Dynamic Tensors

In many applications, one particular tensor of interest is the dynamic tensor. For example, in recommender systems, it measures users' taste on items over time; in brain imaging, it may represent functional magnetic resonance imaging; also, in network analysis, it may formulate the dynamic relations among entities.

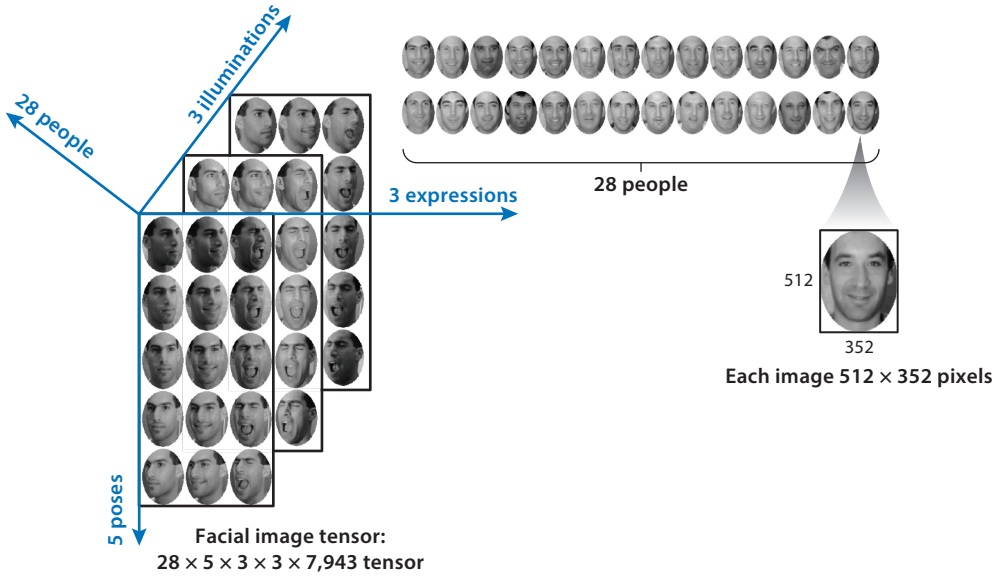

Figure 6

An illustration of Tucker decomposition for facial image ensembles. Adapted with permission from Vasilescu & Terzopoulos (2002, p. 8).

To the best of our knowledge, there are two types of formulations for dynamic tensors. The first is dynamic tensors with slice-wise updates, also known as tensor streams and incremental tensors (Sun et al. 2006, 2008). The dynamic tensors assume that one particular mode (say, the dth mode) of the tensor $\mathcal{X}$ represents time. One application of such a formulation is forecasting. Suppose we are interested in h-point ahead forecasting. Then, ultimately, $\mathcal{X}_{n_1 \times n_2 \times \cdots \times n_d}$ is extended to $\tilde{\mathcal{X}}_{n_1 \times n_2 \times \cdots \times (n_d + h)}$. For example, we may consider sales forecasting in this framework, where $\mathcal{X}$ is a third-order tensor with three modes being customer, product, and season, and entries of $\mathcal{X}$ represent sales volumes (Xiong et al. 2010).

Specifically, one way to achieve forecasting is through tensor factorization followed by extrapolating time-specific latent factors via time series models. Through applying the CP decomposition in Section 2.2, the $(n_d \times r)$-dimensional time-specific latent factor matrix $\mathbf{P}^d$ can be considered as r time series. Existing techniques to extrapolate $\mathbf{P}^d$ include kernel methods (Koren 2010), the Holt-Winters method (Dunlavy et al. 2011), and autoregression (Wang et al. 2016, Yu et al. 2016). Once an extended $[(n_d + h) \times r]$-dimensional $\tilde{\mathbf{P}}^d$ is acquired, the extended $\mathcal{X}$ can be estimated as $\tilde{\mathcal{X}} \approx [\![\mathbf{P}^1, \mathbf{P}^2, \ldots, \tilde{\mathbf{P}}^d]\!]$.

There are two major advantages associated with forecasting after tensor decomposition, compared with a single time series analysis for each individual. First is scalability (Yu et al. 2016). If time series models are directly applied to $\mathcal{X}$, then we expect $n_1 \times n_2 \times \cdots \times n_{d-1}$ time series, which could be extremely huge. For the sales forecasting example, this implies thousands of stores and millions of products, leading to billions of time series. After tensor decomposition, the number of time series is dropped down to r, which is usually less than 100. Second, the tensor decomposition incorporates information across all time series simultaneously. This allows each time series to borrow information from other time series to improve forecasting accuracy.

However, one drawback of estimating time as a tensor mode is a fixed granularity—that is, simply discretizing time into time intervals such as weeks or months, which do not update tensor values at every instance and are unable to capture intricate, nonlinear relationships in data.

Alternatively, one could consider dynamical data streams, which are tensors with element-wise updates (Zhou et al. 2017), or a tensor-valued function (Padovani 2000, Mahyari et al. 2016, Lund 2020, Zhang et al. 2020). In other words, $\mathcal{X} = \mathcal{X}(t)$, where each element is a function of time $x_{i_1 i_2 \ldots i_d}(t)$. For example, we can consider new sales volumes at a given time point t as a tensor-value function $\mathcal{X}(t) \in \mathbb{R}^{n_1 \times n_2 \times n_3}$ with three modes of stores, products, and promotion strategies. The element $x_{i_1 i_2 i_3}(t)$ represents a new sale volume record, that is, the i_1th store sale volume for the i_2th product given the i_3th promotion strategy at time t (Zhang et al. 2020).

One viable approach to fit the tensor function for forecasting is through the CP decomposition with time-varying coefficients using the spline approximation method. Then, forecasting can be obtained via estimating tensor function $\widehat{\mathcal{X}}(t)$ given a future point t. Specifically, each component of $\mathcal{X}(t)$ is approximated via the CP decomposition, with time-varying coefficients as follows: $x_{i_1 i_2 \cdots i_d}(t) \approx \sum_{j=1}^{r} h_j(t) p^1_{i_1 j} p^2_{i_2 j} \cdots p^d_{i_d j}$. The time-varying coefficient $h_j(t)$ can be estimated by spline approximation, that is, $\hat{h}_j(t) = \sum_{i=1}^{M} \alpha_{ji} B_{ji}(t)$, where $B_{ji}(t)$s are spline functions. Once the estimators $\hat{p}^k_{i_k j}$ and $\hat{h}_j(t)$ at given time points t are acquired, the forecasting estimator $\hat{x}_{i_1 i_2 \cdots i_d}(t) \approx \sum_{j=1}^{r} \hat{h}_j(t) \hat{p}^1_{i_1 j} \hat{p}^2_{i_2 j} \cdots \hat{p}^d_{i_d j}$ can be achieved at a given time point t.

There are two major advantages associated with using tensor-valued functions. First, the tensor-valued function can forecast tensor values at any point of time intervals, not just discrete time points with a fixed granularity. The tensor-valued function regards every element of a tensor as a function of time. The forecasting is achieved via fitting functions of time, which ensures that the number of parameters after tensor factorization does not increase as time moves forward and that the computational complexity would be reasonable, while the extended tensors $\tilde{\mathcal{X}}_{n_1 \times n_2 \times \cdots \times (n_d + b)}$ have the dimension at time mode increasing over time. Second, nonparametric estimation methods after tensor decomposition are able to capture intricate relationships from observed data without any model assumption like in time series models. This ensures the robustness of forecasting values.

3. TENSOR ANALYSIS IN RECOMMENDER SYSTEMS

In this section, we discuss tensor applications in recommender systems. A recommender system is one type of information filtering system that tracks user preferences and recommends personalized items to each user. Usually a recommender system can be formulated into a matrix of user-item interactions, for example, movie rating, or online purchasing, denoted by $\mathbf{X} \in \mathbb{R}^{n_1 \times n_2}$, where the (i_1, i_2)th element represents the interaction between user i_1 and item i_2. In practice, $\mathbf{X}$ can be extremely large and sparse. The goal of a recommender system is to complete the matrix and recommend personalized items to each user. Among many existing works, one promising approach is matrix completion (e.g., Candès & Recht 2012, Mazumder et al. 2010, Mao et al. 2019).

3.1. Context-Aware Recommender Systems

Modern recommender systems also collect other useful information in addition to user-item interactions. One type of information to collect is the context of such interactions, for example, time, location, device, companions, networks, or even combinations of these contexts. Recommender systems with contextual information are usually called CARS (Adomavicius & Tuzhilin 2011). Instead of having a single utility matrix $\mathbf{X}$ to represent all user-item interactions, CARS formulates a utility matrix $\mathcal{X}_{i_3}$ for each context $i_3 = 1, \ldots, n_3$, as illustrated in **Figure 7**. For example, we may formulate users' preferences as dynamic and changing over time, or shopping behaviors at different locations or with different devices. In **Figure 4**, each $t = 1, \ldots, n_3$ is a time point (e.g., a week, a season, a year, etc.) and each $\mathcal{X}_t$ represents a utility matrix at time t, corresponding to users'

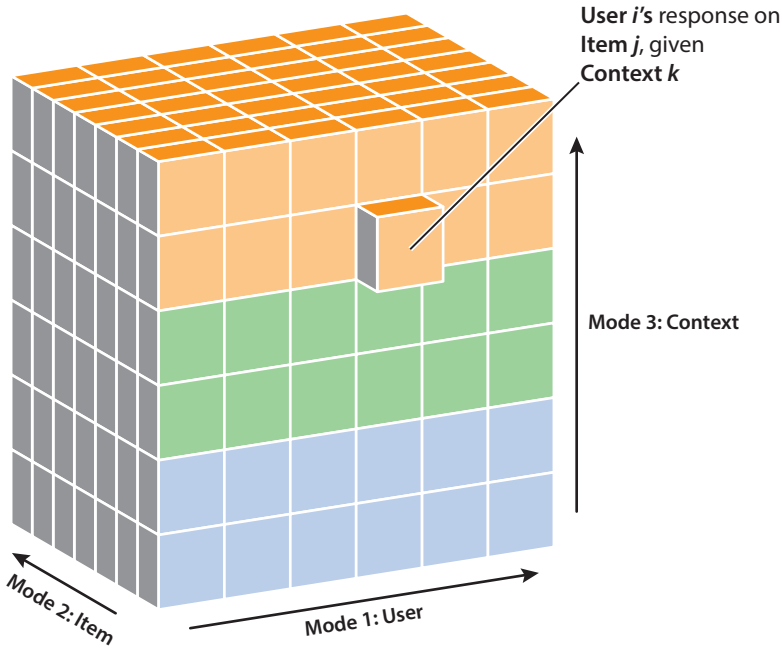

Figure 7

An illustration of a third-order tensor for recommender systems.

preference over movies at time t. In other words, all of the $\mathcal{X}_1, \ldots, \mathcal{X}_{n_3}$ matrices form a tensor $\mathcal{X}$. And the traditional matrix completion problem can be converted to a tensor completion problem for CARS.

For a dth-order tensor $\mathcal{X}$, let $\Omega = \{(i_1, i_2, \ldots, i_d) : x_{i_1 i_2 \cdots i_d} \text{ is observed}\}$ be a set of indices corresponding to observed tensor entries. Then a criterion function for tensor completion can be described as:

$$L(\mathbf{P}^1, \ldots, \mathbf{P}^d) = \sum_{(i_1, \cdots, i_d) \in \Omega} (x_{i_1 \cdots i_d} - \hat{x}_{i_1 \cdots i_d})^2 + \lambda f(\mathbf{P}^1, \ldots, \mathbf{P}^d),$$

where $\hat{x}_{i_1 \cdots i_d}$ is the estimated version of $x_{i_1 \cdots i_d}$, f is a regularization function, and λ is a tuning parameter. For example, one may adopt the L_2 regularization function. And we can achieve the estimation of $\hat{x}_{i_1 \cdots i_d}$ via either the CP decomposition or the Tucker decomposition, as illustrated in Section 6. In contrast to tensor decompositions as introduced in Section 2.2, one difference is that tensors in CARS could be very sparse; that is, $|\Omega|$ could be very small compared with $n_1 n_2 \cdots n_d$. For example, users may not be able to watch all movies on a platform, and the more contexts we collect, the sparser a tensor representation could be. Meanwhile, the tensor entries are not necessarily uniformly or independently drawn, as users have their own preferences and may refer to friends or online reviews for suggestions. To address these issues, Bi et al. (2018) propose a multilayer tensor factorization to accommodate dependency structures at each mode, such that decisions of users, or patterns of items and contexts, are correlated if they are in the same group. Tarzanagh & Michailidis (2019) propose a double core tensor factorization model, which incorporates smoothing loss functions and can accommodate heterogeneity across observation groups. An

alternative approach is to collect additional scalar, matrix, or tensor information (e.g., Li & Zhang 2017, Lock 2018). Another possible extension is to allow a low-rank plus sparse tensor, where the addition of the sparse tensor facilitates high-rank tensors without imposing much computational complexity. This line of work follows the low-rank plus sparse matrices approaches (e.g., Candès et al. 2011, Hao et al. 2020). Because of sparsity, the convergence properties may behave quite differently as well. At the algorithm level, Chen et al. (2012) propose a maximum block improvement algorithm that iteratively updates only the block with the maximum improvement, which guarantees convergence to a stationary point.

Once tensor is decomposed and recovered, CARS indicates not only which user is going to click, purchase, or interact with which item, but also when, where, or even how such clicks, purchases, or interactions would happen. This provides critical strategies to business decision making in personalized advertising, sales forecasting, marketing campaigns, or social networks' friend recommendations, regarding what product to introduce, what movies to be displayed on whose front page, and so on.

3.2. Tensor Completion

Many effective and powerful methods are proposed to achieve tensor completion, and some of these methods also have broad applications beyond the CARS settings (e.g., Zhang 2019). For example, Gandy et al. (2011) consider a tractable convex relaxation method for the low-n-rank tensor recovery, and Mu et al. (2014) introduce a new convex relaxation to reduce sample complexity and exploit several tensor structures jointly. Kressner et al. (2014) propose a nonlinear conjugate gradient method to perform Riemannian optimization techniques. Moreover, Yuan & Zhang (2016) achieve tensor completion via minimizing nuclear norm. And Shah et al. (2015) focus on separable measurement mechanisms to identify a random set of tensor entries.

Given the fact that sparsity is one of the key characteristics of CARS, a related and important topic is estimating the sample size requirement for tensor completion problems under various situations. For $d = 3$, Jain & Oh (2014) propose an alternating minimization–based method. Barak & Moitra (2016) investigate the noisy tensor completion problem and propose a sum-of-squares hierarchy method to achieve tensor completion given $n^{3/2}r$ randomly sampled entries where $n := \max\{n_1, n_2, n_3\}$ is the largest dimension across all modes, and r is the tensor rank. More generally, Krishnamurthy & Singh (2013) and Yuan & Zhang (2017) investigate the sample size requirement for higher-order tensor completion problems, both under the scenario when $d \geqslant 3$ and $n := n_1 = n_2 = \cdots = n_d$. Specifically, Krishnamurthy & Singh (2013) incorporate adaptive sampling in the absence of noise. And Yuan & Zhang (2017) show that the underlying dth-order tensor can be recovered perfectly via an incoherent nuclear norm minimization. These results are summarized in **Table 1**. We notice that Yuan & Zhang (2017) generalize the result of Jain & Oh (2014) and achieve a smaller number of required entries. For the case of $d = 3$, Yuan & Zhang (2017) require a smaller sample size when the tensor rank r is large. Meanwhile, the method proposed by Krishnamurthy & Singh (2013) is more advantageous when d and r are small but n is large.

Table 1 The requirement of uniformly random entries to recover a dth-order tensor of rank r and equal size n at each mode

Method reference	Tensor order	Number of required entries
Jain & Oh (2014)	$d = 3$	$O(n^{3/2}r^5\log^4 n)$
Krishnamurthy & Singh (2013)	$d \geqslant 3$	$O(r^{2(d-1)}d^2 n\log r)$
Yuan & Zhang (2017)	$d \geqslant 3$	$O((r^{(d-1)/2}n^{3/2} + r^{d-1}n)\log^2 n)$

4. TENSOR ANALYSIS IN BIOMEDICAL IMAGING APPLICATIONS

In this section, we review two types of tensor applications in biomedical imaging analysis: supervised predictive models and unsupervised feature learning models. Unlike the tensor data for recommender systems, the imaging data in biomedical studies are typically completely observed, and each entry of the tensor corresponds to an image pixel/voxel.

4.1. Tensor-Based Supervised Learning Models

One of the primary scientific goals in biomedical imaging analysis is to establish the associations between the imaging data and other variables such as clinical assessments, demographic factors, and genetic information. Under such a supervised learning framework, the imaging data are treated as either a covariate or a response.

Conventional statistical and machine learning approaches in imaging analysis are mostly based on the vectorized data, either by summarizing the imaging data through a small number of preidentified regions of interest (ROIs) or by unfolding the entire image into a long vector. The former highly depends on existing domain knowledge and does not fully utilize the raw imaging information, while the latter could become rather restrictive due to the ultrahigh dimensionality of the voxels involved in a high-resolution image. For instance, in a regression model taking each voxel as a variable, a three-dimensional imaging covariate with a size of $256 \times 256 \times 256$ is associated with $256^3 = 16{,}777{,}216$ parameters, which could be infeasible to estimate even with additional regularization techniques. More importantly, some key information, such as the spatial pattern, is likely to be lost while transferring the multidimensional imaging array to a vector form.

To address the limitations of vectorization-based approaches, a family of scalar-on-tensor regression models has been developed to preserve the tensor structure of imaging data (Guo et al. 2012, Zhou et al. 2013). Specifically, a univariate response Y (e.g., disease status) is linked with a vectorized predictor $\mathbf{z} \in \mathbb{R}^{p_0}$ (e.g., age and sex) and a tensor-valued predictor $\mathcal{X} \in \mathbb{R}^{p_1 \times \cdots \times p_d}$ (e.g., medical images, as in **Figure 8**) through a generalized linear model

$$g\{\mathbb{E}(Y)\} = \alpha + \boldsymbol{\gamma}^T \mathbf{z} + \langle \mathcal{B}, \mathcal{X} \rangle, \qquad 3.$$

where $g(\cdot)$ is a link function, and $\langle \cdot\,, \cdot \rangle$ denotes the point-wise inner product between two tensors (**Figure 9**). For dimension reduction purposes, the coefficient tensor $\mathcal{B}$ is assumed to preserve a

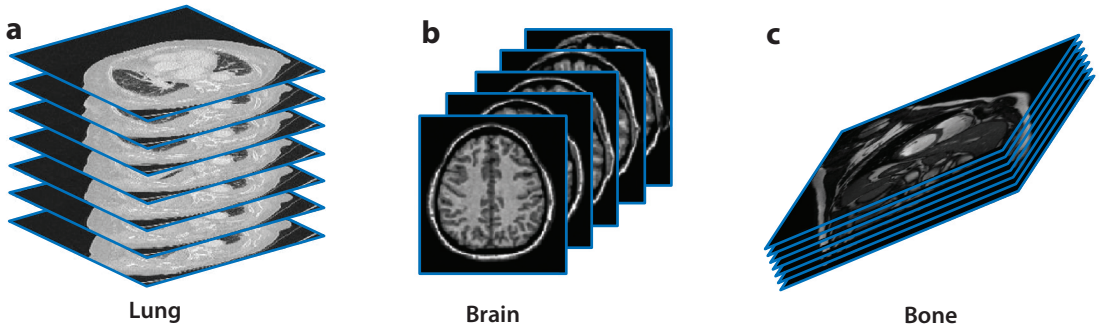

Figure 8

Illustrations of tensor-valued imaging data: (*a*) a 3D CT scan of a lung, (*b*) a 3D MRI of a brain, and (*c*) a 3D MRI of bone. Abbreviations: 3D, three-dimensional; CT, computerized tomography; MRI, magnetic resonance imaging.

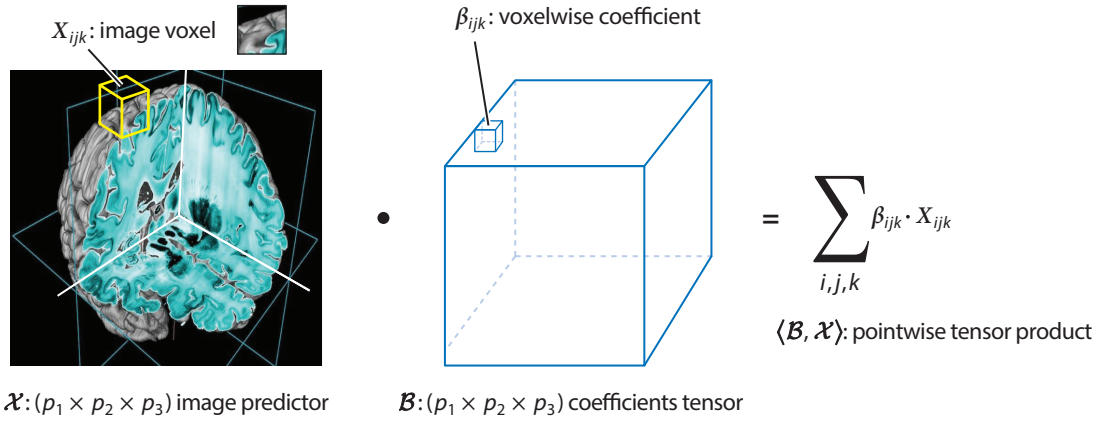

Figure 9

Tensor regression with a three-dimensional MRI brain image predictor. Abbreviation: MRI, magnetic resonance imaging. The left panel was adapted with permission from Amunts et al. (2013).

low-rank structure based on certain tensor decompositions (Zhou et al. 2013, Li et al. 2018). For example, a rank-R CP decomposition significantly reduces the number of model parameters from $(1 + p_0 + p_1 p_2 \cdots p_d)$ to $\{1 + p_0 + R(p_1 + p_2 + \cdots + p_d)\}$. Meanwhile, some regularizations can also be imposed to allow additional structure such as sparsity.

This tensor regression approach can be further extended to model a multivariate response $\mathbf{Y} = (Y_1, Y_2, \ldots, Y_q)^T$, where each marginal response Y_k $(1 \leqslant k \leqslant q)$ is associated with a coefficient tensor $\mathcal{B}_k \in \mathbb{R}^{p_1 \times \cdots \times p_d}$. In addition to reducing the parameter dimension, the individual coefficients are also believed to share some common structures. Hence, the coefficient tensors are formulated in a stack $\mathcal{B} = [\mathcal{B}_1, \ldots, \mathcal{B}_q] \in \mathbb{R}^{p_1 \times \cdots \times p_d \times q}$, which is assumed to be low rank, either based on tensor decomposition techniques (Li et al. 2016, Zhang & Li 2017, Miranda et al. 2018) or achieved by penalizing the tensor nuclear norm (Raskutti et al. 2019).

In some imaging analyses, it is also of great interest to investigate the effects of other factors on an image response, for example, to compare the MRI scans of brains between subjects with attention deficit hyperactivity disorder and normal controls (Li et al. 2016). In contrast to conventional statistical models that mostly focus on a limited number of ROIs, the novel tensor-based approaches enable us to investigate the covariates' effects on all imaging voxels jointly. One prevalent technique is to represent the image response in terms of a linear combination of some multiway feature arrays, and the associated weights could depend on other covariates (Li & Zhang 2017, Sun & Li 2017). In a similar fashion, Lock (2018) further extends it to a tensor-on-tensor regression model allowing a predictor with an arbitrary order.

Beyond the regression framework, similar techniques can be extended to other supervised learning models with tensor predictors, for example, discriminant analysis for tensor classification (Wimalawarne et al. 2016, Yang & Dunson 2016, Lyu et al. 2017, Pan et al. 2019) and multivariate point process models for analyzing the calcium imaging data (Tang & Li 2020). In most existing tensor-based supervised learning approaches, formulating the model parameters into an appropriate tensor plays a pivotal role. In addition to substantial dimension reduction, the imposed low-rank structure also makes it feasible to utilize the high-order information from imaging data and leads to more efficient estimations, as the voxel-wise parameters are jointly estimated through a small number of latent factors from the tensor decomposition.

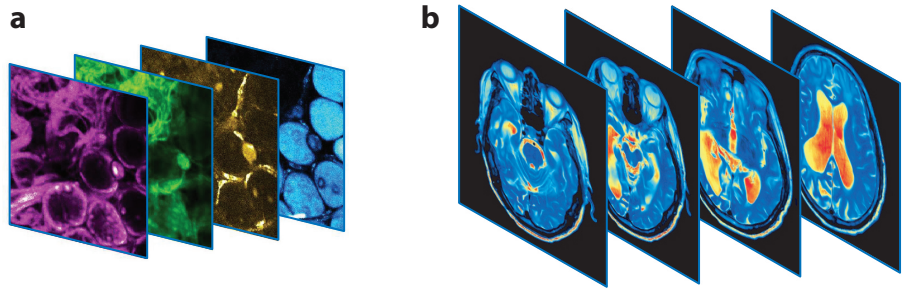

Figure 10

Higher-order image tensors with special modes. (*a*) Four-modality optical imaging data, where different modalities are generated by photons with different wavelengths. (*b*) fMRI data of brain activity, where different modalities correspond to imaging over time. Abbreviation: fMRI, functional magnetic resonance imaging.

4.2. Unsupervised Tensor Feature Learning

Most aforementioned supervised learning approaches focus on the structure of coefficients rather than on the image itself. In many other applications, such as cancer imaging analysis, it is crucial to process signals from images directly for effective feature learning and extraction. Nevertheless, due to the extremely large volume of individual images compared with the limited number of subjects, directly applying conventional techniques such as PCA to the vectorized imaging data could be infeasible in both model estimation and computation.

To tackle these challenges, multiple individual images are aligned into an integrated tensor using tensor decompositions. Consequently, each individual tensor can be represented as a linear combination of some basis features (Mørup et al. 2006, Cong et al. 2012, Karahan et al. 2015). In contrast to the vectorization-based methods, the obtained basis feature here is a layer of rank-1 tensors that can effectively preserve the higher-order structure information.

In addition to regular tensor decomposition methods, a variety of new decomposition techniques have been developed to accommodate specific structures of different tensor data. The classic tensor decomposition method assuming population-shared bases and discrete factors can be restrictive for more complex imaging data involving special modes, such as time (e.g., fMRI data) and modalities (e.g., multimodality imaging data), as in **Figure 10**.

To account for the heterogeneity along different modes of tensor data, Tang et al. (2019) propose an individualized multilayer tensor decomposition incorporating orthogonal mode-specific layers. For example, for multimodality imaging data, each single image tensor for the ith subject and the mth modality is formulated as

$$\mathcal{X}_i^{(m)} = \sum_{r=1}^{R} w_{ir}^{(m)} \mathcal{A}_r^{(m)} + \nu_i \mathcal{S}_i, \quad \text{s.t.} \quad \langle \mathcal{A}_r^{(m)}, \mathcal{S}_i \rangle = 0, \text{ for } 1 \leqslant r \leqslant R, \qquad 4.$$

where $\mathcal{A}_r^{(m)}$ is a population-shared, modality-specific basis feature, and $\mathcal{S}_i$ is an individualized layer shared by different imaging modalities capturing the cross-subject variations. Meanwhile, in the spirit of functional data analysis, some dynamic tensor decomposition methods are developed to model time-dependent tensor data. Section 2.3 provides more details.

Another important question in unsupervised learning is clustering. In imaging analysis, directly applying a clustering algorithm to the vectorized tensor data may suffer from poor clustering

accuracy and heavy computational burden. One option is to carry out the clustering based on the extracted low-dimensional features of tensors (Cao et al. 2013); however, this could be ineffective due to the two-stage estimation. Another, more appealing approach is to embed clustering into the decomposition for the integrated higher-order tensor (Madrid-Padilla & Scott 2017, Sun & Li 2019). For example, a fusion penalty, $\sum_{r=1}^{R} \sum_{i,i'} |w_{ir} - w_{i'r}|$, can be incorporated on the latent factors in the corresponding CP decomposition: $\sum_{r=1}^{R} \mathbf{b}_r^1 \circ \mathbf{b}_r^2 \circ \cdots \circ \mathbf{b}_r^d \circ \mathbf{w}_r$, yielding an estimated clustering structure along with specified tensor modes simultaneously.

5. TENSOR ANALYSIS IN NETWORK

Network data are collections of entities and measured relations between them. A network $G(V, E)$ is defined on a set of nodes V and a set of edges or links between nodes E. In a social network, each node represents an individual, and an edge encodes the existence of a relationship between two individuals. In a biological network, the node set V can represent a group of proteins, and the interactions between proteins are associated with edges in E.

In this section, we review tensor applications on multirelational networks and multiway relational networks. A multirelational network $\mathcal{G}_1(V, \mathcal{E})$ consists of a set of networks $\{G_1(V, E_1), G_2(V, E_2), \ldots, G_m(V, E_m)\}$ on the same node set V, where the edge set $\mathcal{E} = \{E_i\}_{i=1}^m$ consists of edges given m different relation types. Specifically, each element in E_i is an edge connecting a pair of nodes based on the ith relation. In contrast, a multiway relational network $\mathcal{G}_2(V, \mathcal{E})$ consists of a node set V and an edge set $\mathcal{E}$ where each element in $\mathcal{E}$ connects multiple nodes from V.

Traditionally, an ordinary network can be represented as a matrix with the rows and columns labeled as the nodes and binary entries indicating the presence or absence of an edge between two nodes. However, the matrix is incapable of representing multiway or multitype relations. In contrast, both multiway relational networks and multirelational networks can be formulated as a tensor to provide complex network structures. In the former case, each mode indicates a participating entity in a multiway relation, while an additional mode provides types of relations in the latter case. For example, we can utilize an mth-order tensor $\mathcal{X} = \{0, 1\}^{n^m}$ to represent a set of m-way relations among n nodes where the binary element $x_{i_1 i_2 \cdots i_m}$ indicates the presence or absence of m-way relations among nodes $i_1, i_2, \ldots i_m$. In addition, a third-order tensor $\mathcal{X} = \{0, 1\}^{n \times n \times m}$ represents m types of relations among n nodes where element x_{ijk} indicates whether a kth-type relation exists or not between nodes i and j.

5.1. Multirelational Modeling in Tensors

Applications for multirelational networks include knowledge graphs where factual information is represented as different types of relations among the entities in a database, and social networks where the relationships between people are characterized through their connections in terms of different social relations.

Specifically, let V be a set of users in a social network where the size of node set $|V|$ is n, and let $\mathcal{E} = \{E_i\}_{i=1}^m$ be a set of friendships from m different social media. Therefore, the multirelational social network illustrated in **Figure 11** can be formulated as a third-order tensor $\mathcal{X} = \{x_{ijk}\} \in \{0, 1\}^{n \times n \times m}$. A common tensor structure in multirelational modeling is termed partially symmetric in two specific modes if its tensor slices are symmetric. For example, a third-order tensor $\mathcal{X} \in \mathbb{R}^{p_1 \times p_2 \times p_3}$ is partially symmetric in the first two modes if $p_1 = p_2$ and $\mathcal{X}_{::i_3} = \mathcal{X}_{::i_3}^T$, and the corresponding CP decomposition can be formulated as $\mathcal{X} = \sum_{j=1}^{r} \tilde{\mathbf{p}}_j \circ \tilde{\mathbf{p}}_j \circ \mathbf{p}_j^3$. Given that the relationships are mutual, a multirelational social network consists of a stack of undirected networks and is partially symmetric in the two modes regarding users.

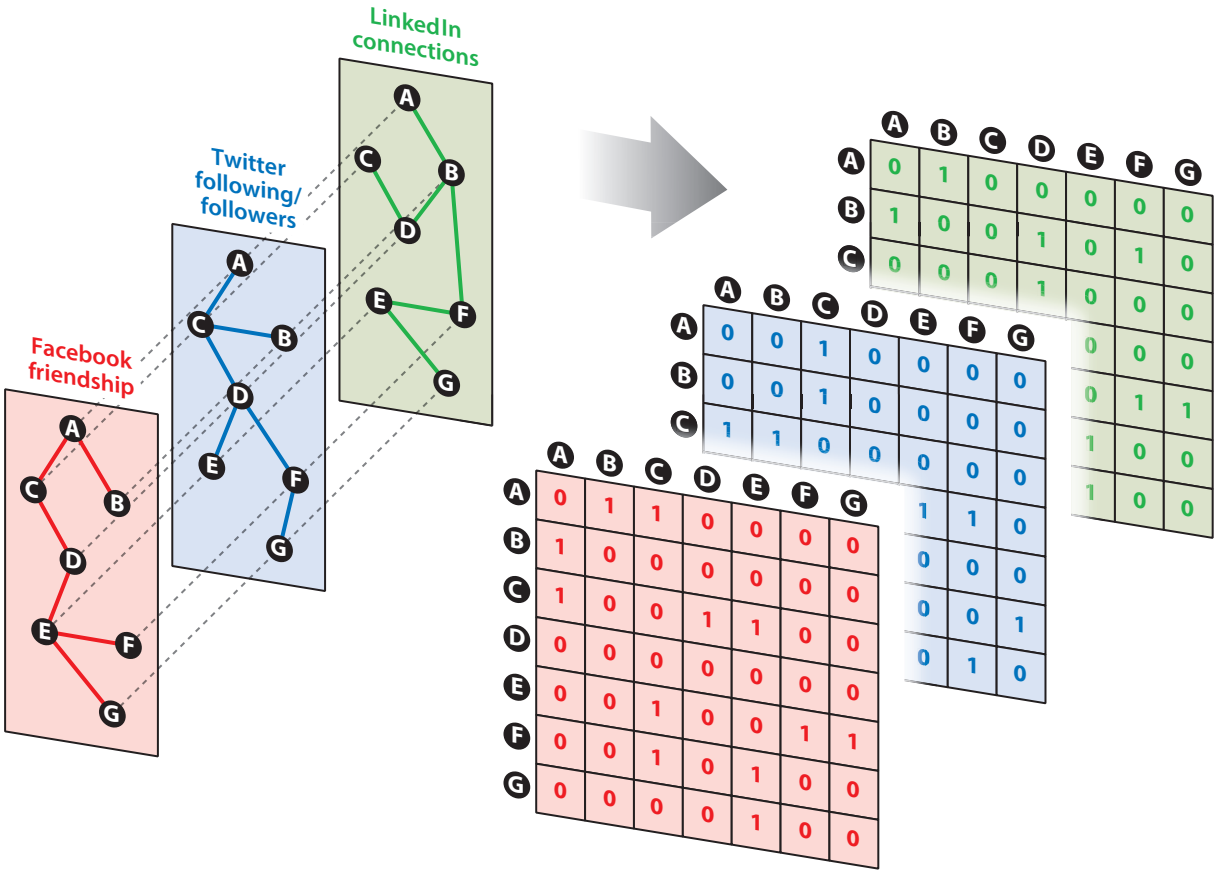

Figure 11

An illustration of a tensor formulation for multirelational social networks, where {*A*, *B*, *C*, *D*, *E*, *F*, *G*} denote seven social media users.

We model the observed relation as a Bernoulli distribution: $P(x_{ijk}) = [\sigma(f_{ijk})]^{x_{ijk}}[1 - \sigma(f_{ijk})]^{1-x_{ijk}}$, where $\sigma(\cdot)$ is the logistic link function. This differs from the existing methods via a partially symmetric score tensor $\{f_{ijk}\} \in R^{n \times n \times m}$. The tensor decomposition is adopted here to infer latent factor representations of $\{f_{ijk}\}$. Under this line of work, Kolda et al. (2005) and Franz et al. (2009) propose to factorize the score tensor through the CP decomposition to capture the heterogeneous interaction among entities. Nickel et al. (2011) propose a bilinear tensor decomposition to capture the interactions between two entities using a multiplicative term with a relation-related weighting matrix. This type of formulation generalizes the CP decomposition in the sense that it degenerates to CP decomposition given diagonal weighting matrices. Noticing that the CP decomposition considers three-way interactions among entities and relations, Jenatton et al. (2012) propose a saturated interaction model to include intercepts and two-way interaction. Under a similar data structure, Zhang et al. (2019) propose a decomposition to extract the network features shared by all individual networks and their loadings on the base networks. Specifically, the tensor network is decomposed as follows: $\mathcal{X} \approx \sum_{j=1}^{r} \lambda_j \mathbf{p}_j^1 \circ \mathbf{p}_j^1 \circ \mathbf{p}_j^2$, where $\{\mathbf{p}_j^1 \circ \mathbf{p}_j^1\}_{j=1}^r$ are the base networks that are the counterpart of principle components in PCA, and the ith row of $\{\mathbf{p}_1^2, \ldots, \mathbf{p}_r^2\}$ is the loadings for the individual networks $\mathcal{X}_{\cdot\cdot i}$.

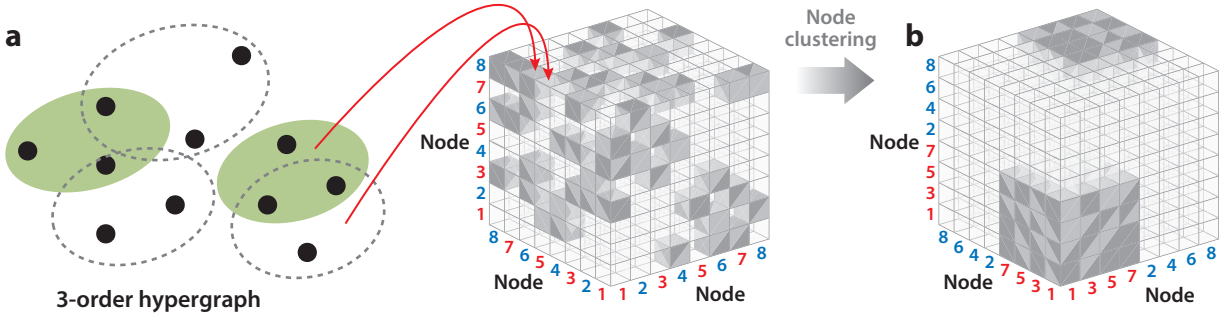

Figure 12

A hypergraph consists of a set of nodes and a set of multiway links among nodes. A hypergraph can be represented as a tensor with modes labeled as the nodes and binary entries indicating the presence or absence of a multiway link. (*a*) A 3-uniform hypergraph is formulated as a third-order tensor. (*b*) Clustering within a hypergraph tensor: Group nodes into sets such that each set of nodes is densely connected internally.

5.2. Multiway Relation Modeling in Tensors

In many applications of complex relational data, the system involves not only pairwise relations between entities but also multiway or high-order interactions among a group of entities. For example, in a protein-protein interaction network (Klamt et al. 2009), it is important to model and capture the collaborative high-order interactions among a subgroup of proteins, which is functionally associated with a protein complex.

A hypergraph tensor (Bretto 2013, Corsini & Leoreanu 2013, Pearson & Zhang 2014) is proposed to generalize the two-way links represented by elements in a matrix to multiway hyperlinks represented by elements in a tensor. Specifically, a multiway network with n nodes, where each relation involves m entities, is formulated as a mth-order adjacency tensor $\mathcal{X} = \{0, 1\}^{n^m}$. The encoding procedure is illustrated in **Figure 12*a***.

In an application such as protein-protein interactions, the order among entities could be indifferent within a hyperlink, which represents a subgroup of entities. The corresponding tensor is referred as supersymmetric (Kolda 2006), such that all elements remain constant under any permutation of the indices. For example, a rank-r m-order adjacency tensor is supersymmetric if all elements in $\{x_{\sigma(i_1)\cdots\sigma(i_m)} | \sigma(\cdot)$ is any index permutation$\}$ are equal to $x_{i_1\ldots i_m}$, and this imposes the CP representation in the form $\mathcal{X} = \sum_{j=1}^{r} \mathbf{p}_j \circ \mathbf{p}_j \circ \cdots \circ \mathbf{p}_j$.

One of the fundamental problems in multiway network analysis is link-based clustering (see **Figure 12*b***). This is also equivalent to identifying the block structure in an adjacency tensor. Due to the nature of unsupervised learning of clustering algorithms, Ghoshdastidar & Dukkipati (2014) introduce a planted partition model to enable performance comparison and theoretical analysis, and to characterize the randomness of the block structure in a hypergraph tensor. Consider an m-order hypergraph tensor containing n nodes and k communities; let $Z \in \{0, 1\}^{n \times k}$ be the assignment matrix of the nodes' membership. The generating probability of a hyperlink is governed by $E(\mathcal{X}_{i_1 i_2 \ldots i_m}) = \sum_{j_1,\ldots,j_m=1}^{k} \mathrm{B}_{j_1 j_2 \ldots j_m} Z_{i_1 j_1} \ldots Z_{i_m j_m}$, where $\mathrm{B} \in [0, 1]^{k^m}$ indicates the block-wise connectivity probabilities. This model generalizes the stochastic block model (Holland et al. 1983) from a simple graphical model to a hypergraph model. Specifically, a hypergraph tensor is treated as a random realization from the mixture model, and therefore, specific properties of a clustering algorithm can be established under this model assumption.

The general solution for hypergraph tensor clustering is first projecting each node into a latent factor vector and then performing a heuristic clustering algorithm on the latent space. However,

existing methods differ on the procedure used to project the tensor into a low-dimension representation. The first category of method (e.g., Ghoshdastidar & Dukkipati 2014, Lin et al. 2017, Kim et al. 2018) is to reconstruct a similarity matrix through the original hypergraph tensor and then apply node clustering methods to the graph.

The second line of works follows the idea of performing clustering on the node-wise latent factors obtained through the tensor decomposition introduced in Section 2.2. Ghoshdastidar & Dukkipati (2015) propose a method for partitioning uniform hypergraphs by higher-order singular value decomposition (HOSVD) of hypergraph tensors. In their subsequent work (Ghoshdastidar & Dukkipati 2017), the assumption of uniform hyperedge size can be relaxed, and a more general hypergraph model with mixing edge orders is considered. To overcome the suboptimality of HOSVD, Ke et al. (2019) propose a penalized higher-order orthogonal iteration to obtain the low-rank Tucker decomposition, where large entries are truncated at each iteration. This method extends the higher-order orthogonal iteration (HOOI) for tensors with Gaussian entities to a binary tensor and is able to handle the sparsity in hypergraph tensors. Related to the hypergraph clustering problem, it is often reasonable to investigate whether communities exist or not in an observed hypergraph prior to identifying these communities. Yuan et al. (2018) study the phase transition condition for differentiating between a tensor generated by a block model and one generated by a complete random process and propose the corresponding test statistic based on hypergraph cycles when the community structure is detectable.

6. ALGORITHMS FOR TENSOR COMPUTATION

Supplemental Material >

In this section, we introduce some canonical and commonly used tensor computation algorithms. Available software packages are listed in the **Supplemental Appendix**.

Assuming the number of components is fixed, existing algorithms to compute a CP decomposition include the alternating least squares (ALS) method (Carroll & Chang 1970) and the tensor power method (Anandkumar et al. 2017). The key idea for the ALS algorithm is to estimate each $\mathbf{P}^k$ cyclically, $k = 1, \ldots, d$, while fixing $\mathbf{P}^{k'}$s until a stopping criterion is satisfied. A commonly used criterion function for tensor CP decomposition (with L_2-penalty to guarantee convergence) is as follows:

$$L(\mathbf{P}^1, \ldots, \mathbf{P}^d | \mathcal{X}) = \sum_{i_1, \ldots, i_d} (x_{i_1 \cdots i_d} - \hat{x}_{i_1 \cdots i_d})^2 + \lambda \sum_{k=1}^{d} \|\mathbf{P}^k\|_F^2,$$

where $\hat{x}_{i_1 i_2 \cdots i_d} = \sum_{j=1}^{r} p^1_{i_1 j} p^2_{i_2 j} \cdots p^d_{i_d j}$ is an estimate of the tensor element $x_{i_1 i_2 \cdots i_d}$, λ is a tuning parameter to control the magnitude of the penalty, and $\|\cdot\|_F$ represents the Frobenius norm. Then the i_kth row of $\mathbf{P}^k$ can be estimated as

$$\hat{\mathbf{p}}^k_{i_k} = \underset{\mathbf{p}^k_{i_k}}{\operatorname{argmin}} \sum_{i_1, \ldots, i_{k-1}, i_{k+1}, \ldots, i_d} (x_{i_1 \cdots i_d} - \hat{x}_{i_1 \cdots i_d})^2 + \lambda \|\mathbf{p}^k_{i_k}\|_2^2, \; i_k = 1, \ldots, n_k,$$

which is essentially a ridge regression. Each iteration of the ALS algorithm consists of cyclically updating each row of $\mathbf{P}^k$, for $k = 1, \ldots, d$. The algorithm is considered converged if the improvement of the criterion function $L(\cdot|\mathcal{X})$ is smaller than a certain threshold (e.g., 10^{-4}). Other penalty functions can also be considered, such as L_1- or L_0-penalty for sparsity pursuit (Zhu et al. 2016).

We provide more details for the case $d = 2$ (matrix) as an illustration, where we have $\hat{x}_{i_1 i_2} = (\mathbf{p}^1_{i_1})^T (\mathbf{p}^2_{i_2})$ in the loss function. We first initialize $\mathbf{P}^1$ and $\mathbf{P}^2$ as independent and identically distributed random normal variables with zero mean and small standard deviation. Then, minimizing $L(\cdot|\mathcal{X})$ can be achieved via iteratively updating the following two equations:

$$\mathbf{p}^2_{i_1} = \left\{(\mathbf{P}^2)^T (\mathbf{P}^2) + \lambda \mathbf{I}_r\right\}^{-1} (\mathbf{P}^2)^T \mathbf{x}_{i_1,\cdot}, \text{ and } \mathbf{p}^2_{i_2} = \left\{(\mathbf{P}^1)^T (\mathbf{P}^1) + \lambda \mathbf{I}_r\right\}^{-1} (\mathbf{P}^1)^T \mathbf{x}_{\cdot,i_2}.$$

The ALS algorithm for $d = 2$ converges to a stationary point. For tensors with $d \geqslant 3$, the ALS can be conducted in a similar fashion. However, it may converge only to a point where the criterion function ceases to decrease (Chen et al. 2012), and the performance of this algorithm is influenced by its initialization (Rabanser et al. 2017).

Algorithms to compute a Tucker decomposition mainly include HOSVD (De Lathauwer et al. 2000a), HOOI (De Lathauwer et al. 2000b), and Newton-Grassmann optimization (Eldén & Savas 2009). The key idea behind HOSVD (Tucker 1966) is to find the components that best capture the variation in a mode, while not considering the other modes at this point in time. This directly corresponds to the basic PCA concept. Specifically, computing a Tucker decomposition of a tensor $\mathcal{X} \in \mathbb{R}^{n_1 \times n_2 \cdots \times n_d}$ is done by solving the following optimization problem:

$$\min_{\mathcal{C}, \mathbf{Q}^1, \ldots, \mathbf{Q}^d} \left\| \mathcal{X} - [\![\mathcal{C}; \mathbf{Q}^1, \mathbf{Q}^2, \ldots, \mathbf{Q}^d]\!] \right\|, \quad 5.$$

subject to $\mathcal{C} \in \mathbb{R}^{m_1 \times m_2 \times \ldots \times m_d}$, $\mathbf{Q}^k \in \mathbb{R}^{n_k \times m_k}$ and columnwise orthogonal for $k = 1, \ldots, d$. The detailed HOSVD algorithm is provided in the **Supplemental Appendix**.

Supplemental Material >

HOOI is essentially an ALS algorithm, which uses the outcome of performing HOSVD on a tensor as a starting point for initializing the factor matrices (Kolda & Bader 2009). This algorithm is especially helpful in cases where we only have access to a truncated HOSVD, since the successive application of the ALS algorithm allows for more accurate decompositions. Since the core tensor $\mathcal{C}$ must satisfy $\mathcal{C} = \mathcal{X} \times_1 \mathbf{Q}^{1\top} \times_2 \mathbf{Q}^{2\top} \cdots \times_d \mathbf{Q}^{d\top}$, we can then rewrite the square of the above objective function as $\|\mathcal{X} - [\![\mathcal{C}; \mathbf{Q}^1, \mathbf{Q}^2, \ldots, \mathbf{Q}^d]\!]\|^2 = \|\mathcal{X}\|^2 - \|\mathcal{X} \times_1 \mathbf{Q}^{1\top} \times_2 \cdots \times_d \mathbf{Q}^{d\top}\|^2$. Therefore, solving the above minimization problem is equivalent to solving a series of the maximization problems, where the kth step solves for the kth component matrix: $\max_{\mathbf{Q}^k} \|\mathcal{X} \times_1 \mathbf{Q}^{1\top} \times_2 \mathbf{Q}^{2\top} \cdots \times_d \mathbf{Q}^{d\top}\|$ subject to $\mathbf{Q}^k \in \mathbb{R}^{n_k \times m_k}$ and columnwise orthogonal. The solution can be determined using the SVD. The detailed HOOI algorithm is provided in the **Supplemental Appendix**.

7. CONCLUSION AND FUTURE DIRECTIONS

In this article, we provide a review of the basic properties of tensors; important developments of tensor-based models in statistics; and various applications in recommender systems, medical imaging analyses, and multiway multirelational network data. In addition, we also summarize some canonical tensor computation algorithms.

We summarize the tensor methods and tools through parametric modeling based on a low-rank decomposition; however, nonparametric tensor modeling has also been proposed. One possible approach to nonparametric tensors is based on the reproducing kernel Hilbert space. The tensor is formulated using a tensor product kernel, and tensor completion and regression can be implemented based on the kernel-based interpolation. Another line of nonparametric tensors follows nonparametric Bayesian modeling via utilizing a Gaussian process to capture the interactions among tensor entities. In general, nonparametric modeling is able to characterize nonlinear relationships among data entities and is robust for noisy observations, but with the price of increasing computational cost.

The existing work on tensor completion also opens up several potential future directions. One of the most important areas is the relaxation of the uniformly sampled entries assumed in existing tensor completion approaches. One possible extension is to treat the unobserved tensor entries as not missing completely at random (Rubin 1976). Such models have been widely studied under the longitudinal or survival data framework.

In addition, there are also several unsolved problems for tensor applications in imaging analyses. One of the most challenging topics is how to effectively integrate tensor-value imaging

covariates with other predictors, such as demographic characteristics, treatments, and genetic information—for example, how to combine both imaging and genetic data. One possible direction is to borrow the idea of sufficient dimension reduction or partial least squares methods and project the data onto specifically designed subspaces. In addition, a hierarchal model could be useful to embed the additional factors into a tensor regression model.

Tensor decompositions have also been studied in research on deep learning architectures. For example, studies show that a shallow network is equivalent to rank-1 CP decomposition (Cohen et al. 2015), whereas a deep network corresponds to a hierarchical Tucker decomposition. In addition, tensor decomposition can be used to represent the weighting matrix of a fully-connected layer, and the core of the Tucker decomposition is used for reparameterizing the deep network layer (Novikov et al. 2015). Furthermore, tensors have been applied in clustering analyses via linear tensor discriminant analyses (Baudat & Anouar 2000) and tensor stack networks (Hutchinson et al. 2013). Both schemes have been successfully implemented in speech and facial recognition. These methodologies require a relatively large amount of data in order to construct indirect partitions of these data sets. Thus, new schemes or extensions of existing tensor clustering methods need to be further developed for deep learning.

Furthermore, complex network structures also motivate us to develop multiple-order tensors. One potential direction is clustering or link predictions for heterogeneous hypergraphs with varying sizes of hyperlinks, which would require a generalization of stacking tensors with different orders.

DISCLOSURE STATEMENT

The authors are not aware of any affiliations, memberships, funding, or financial holdings that might be perceived as affecting the objectivity of this review.

ACKNOWLEDGMENTS

The authors would like to acknowledge support for this project from the National Science Foundation grants DMS-1821198 and DMS-1952406 and the National Natural Science Foundation of P.R. China (11871420). The authors thank the Editorial Committee, Production Editor, and anonymous reviewers for their suggestions and helpful feedback, which improved the article significantly.

LITERATURE CITED

Adomavicius G, Tuzhilin A. 2011. Context-aware recommender systems. In *Recommender Systems Handbook*, ed. F Ricci, L Rokach, B Shapira, PB Kantor, pp. 217–53. New York: Springer

Anandkumar A, Ge R, Janzamin M. 2017. Analyzing tensor power method dynamics in overcomplete regime. *J. Mach. Learn. Res.* 18:752–91

Barak B, Moitra A. 2016. Noisy tensor completion via the sum-of-squares hierarchy. *PMLR* 49:417–45

Baudat G, Anouar F. 2000. Generalized discriminant analysis using a kernel approach. *Neural Comput.* 12:2385–404

Bi X, Qu A, Shen X. 2018. Multilayer tensor factorization with applications to recommender systems. *Ann. Stat.* 46:3308–33

Bretto A. 2013. *Hypergraph Theory: An Introduction*. New York: Springer

Candès EJ, Li X, Ma Y, Wright J. 2011. Robust principal component analysis? *J. ACM* 58:1–37

Candès EJ, Recht B. 2012. Exact matrix completion via convex optimization. *Commun. ACM* 55:111–19

Cao X, Wei X, Han Y, Yang Y, Lin D. 2013. Robust tensor clustering with non-greedy maximization. In *Proceedings of the Twenty-Third International Joint Conference on Artificial Intelligence*, ed. F Rossi, pp. 1254–59. Menlo Park, CA: AAAI

Carroll JD, Chang JJ. 1970. Analysis of individual differences in multidimensional scaling via an n-way generalization of "Eckart-Young" decomposition. *Psychometrika* 35:283–319
Carroll JD, Pruzansky S, Kruskal JB. 1980. Candelinc: a general approach to multidimensional analysis of many-way arrays with linear constraints on parameters. *Psychometrika* 45:3–24
Chen B, He S, Li Z, Zhang S. 2012. Maximum block improvement and polynomial optimization. *SIAM J. Optim.* 22:87–107
Cohen N, Sharir O, Shashua A. 2015. On the expressive power of deep learning: a tensor analysis. arXiv:1509.05009 [cs.NE]
Cong F, Phan AH, Zhao Q, Huttunen-Scott T, Kaartinen J, et al. 2012. Benefits of multi-domain feature of mismatch negativity extracted by non-negative tensor factorization from EEG collected by low-density array. *Int. J. Neural Syst.* 22:1250025
Corsini P, Leoreanu V. 2013. *Applications of Hyperstructure Theory*. New York: Springer
De Lathauwer L, De Moor B, Vandewalle J. 2000a. A multilinear singular value decomposition. *SIAM J. Matrix Anal. Appl.* 21:1253–78
De Lathauwer L, De Moor B, Vandewalle J. 2000b. On the best rank-1 and rank-($r_1, r_2, \ldots, r_n$) approximation of higher-order tensors. *SIAM J. Matrix Anal. Appl.* 21:1324–42
Dunlavy DM, Kolda TG, Acar E. 2011. Temporal link prediction using matrix and tensor factorizations. *ACM Trans. Knowl. Discov. Data* 5(10):1–27
Eldén L, Savas B. 2009. A Newton–Grassmann method for computing the best multilinear rank-(r_1, r_2, r_3) approximation of a tensor. *SIAM J. Matrix Anal. Appl.* 31:248–71
Franz T, Schultz A, Sizov S, Staab S. 2009. TripleRank: ranking semantic web data by tensor decomposition. In *The Semantic Web—ISWC 2009*, ed. A Bernstein, DR Karger, T Health, L Feigenbaum, D Maynard, et al., pp. 213–28. New York: Springer
Gandy S, Recht B, Yamada I. 2011. Tensor completion and low-n-rank tensor recovery via convex optimization. *Inverse Probl.* 27:025010
Ghoshdastidar D, Dukkipati A. 2014. Consistency of spectral partitioning of uniform hypergraphs under planted partition model. In *Advances in Neural Information Processing Systems 27 (NIPS 2014)*, ed. Z Ghahramani, M Welling, C Cortes, ND Lawrence, KQ Weinberger, pp. 397–405. Red Hook, NY: Curran
Ghoshdastidar D, Dukkipati A. 2015. A provable generalized tensor spectral method for uniform hypergraph partitioning. *PMLR* 37:400–9
Ghoshdastidar D, Dukkipati A. 2017. Consistency of spectral hypergraph partitioning under planted partition model. *Ann. Stat.* 45:289–315
Guo W, Kotsia I, Patras I. 2012. Tensor learning for regression. *IEEE Trans. Image Proc.* 21:816–27
Hao B, Zhang A, Cheng G. 2020. Sparse and low-rank tensor estimation via cubic sketchings. *PMLR* 108:1319–30
Harshman RA. 1978. *Models for analysis of asymmetrical relationships among n objects or stimuli.* Presented at First Joint Meeting of the Psychometric Society and the Society of Mathematical Psychology, Hamilton, Ontario
Harshman RA, Lundy ME. 1996. Uniqueness proof for a family of models sharing features of Tucker's three-mode factor analysis and PARAFAC/CANDECOMP. *Psychometrika* 61:133–54
Hitchcock FL. 1927. The expression of a tensor or a polyadic as a sum of products. *J. Math. Phys.* 6:164–89
Holland PW, Laskey KB, Leinhardt S. 1983. Stochastic blockmodels: first steps. *Soc. Netw.* 5:109–37
Hutchinson B, Deng L, Yu D. 2013. Tensor deep stacking networks. *IEEE Trans. Pattern Anal. Mach. Intell.* 35:1944–57
Jain P, Oh S. 2014. Provable tensor factorization with missing data. In *Advances in Neural Information Processing Systems 27 (NIPS 2014)*, ed. Z Ghahramani, M Welling, C Cortes, ND Lawrence, KQ Weinberger, pp. 1431–39. Red Hook, NY: Curran
Jenatton R, Roux NL, Bordes A, Obozinski GR. 2012. A latent factor model for highly multi-relational data. In *Advances in Neural Information Processing Systems 25 (NIPS 2012)*, ed. F Pereira, CJC Burges, L Bottou, KQ Weinberger, pp. 3167–75. Red Hook, NY: Curran
Karahan E, Rojas-Lopez PA, Bringas-Vega ML, Valdes-Hernandez PA, Valdes-Sosa PA. 2015. Tensor analysis and fusion of multimodal brain images. *Proc. IEEE* 103:1531–59

Ke ZT, Shi F, Xia D. 2019. Community detection for hypergraph networks via regularized tensor power iteration. arXiv:1909.06503 [stat.ME]
Kiers HAL. 2000. Towards a standardized notation and terminology in multiway analysis. *J. Chemom.* 14:105–22
Kim C, Bandeira AS, Goemans MX. 2018. Stochastic block model for hypergraphs: statistical limits and a semidefinite programming approach. arXiv:1807.02884 [math.PR]
Klamt S, Haus UU, Theis F. 2009. Hypergraphs and cellular networks. *PLOS Comput. Biol.* 5:e1000385
Kolda TG. 2006. *Multilinear operators for higher-order decompositions.* Tech. Rep. SAND2006-2081, Sandia Natl. Lab., Albuquerque, NM
Kolda TG, Bader BW. 2009. Tensor decompositions and applications. *SIAM Rev.* 51:455–500
Kolda TG, Bader BW, Kenny JP. 2005. Higher-order web link analysis using multilinear algebra. In *Fifth IEEE International Conference on Data Mining*. New York: IEEE
Koren Y. 2010. Collaborative filtering with temporal dynamics. *Commun. ACM* 53:89–97
Kressner D, Steinlechner M, Vandereycken B. 2014. Low-rank tensor completion by Riemannian optimization. *BIT Numer. Math.* 54:447–68
Krishnamurthy A, Singh A. 2013. Low-rank matrix and tensor completion via adaptive sampling. In *Advances in Neural Information Processing Systems 26 (NIPS 2013)*, ed. CJC Burges, L Bottou, M Welling, Z Ghahramani, KQ Weinberger, pp. 836–44. Red Hook, NY: Curran
Kruskal JB. 1977. Three-way arrays: rank and uniqueness of trilinear decompositions, with application to arithmetic complexity and statistics. *Linear Algebra Appl.* 18:95–138
Kruskal JB. 1989. Rank, decomposition, and uniqueness for 3-way and N-way arrays. In *Multiway Data Analysis*, ed. R Coppi, S Bolasco, pp. 7–18. Amsterdam: North-Holland
Li L, Zhang X. 2017. Parsimonious tensor response regression. *J. Am. Stat. Assoc.* 112:1131–46
Li X, Xu D, Zhou H, Li L. 2018. Tucker tensor regression and neuroimaging analysis. *Stat. Biosci.* 10:520–45
Li Z, Suk HI, Shen D, Li L. 2016. Sparse multi-response tensor regression for Alzheimer's disease study with multivariate clinical assessments. *IEEE Trans. Med. Imaging* 35:1927–36
Lin CY, Chien IE, Wang IH. 2017. On the fundamental statistical limit of community detection in random hypergraphs. In *2017 IEEE International Symposium on Information Theory*, pp. 2178–82. New York: IEEE
Liu X, Sidiropoulos ND. 2001. Cramer-Rao lower bounds for low-rank decomposition of multidimensional arrays. *IEEE Trans. Signal Proc.* 49:2074–86
Lock EF. 2018. Tensor-on-tensor regression. *J. Comput. Gr. Stat.* 27:638–47
Lund K. 2020. The tensor t-function: A definition for functions of third-order tensors. *Numer. Linear Algebra Appl.* **https://doi.org/10.1002/nla.2288**
Lyu T, Lock EF, Eberly LE. 2017. Discriminating sample groups with multi-way data. *Biostatistics* 18:434–50
Madrid-Padilla OH, Scott J. 2017. Tensor decomposition with generalized lasso penalties. *J. Comput. Gr. Stat.* 26:537–46
Mahyari AG, Zoltowski DM, Bernat EM, Aviyente S. 2016. A tensor decomposition-based approach for detecting dynamic network states from EEG. *IEEE Trans. Biomed. Eng.* 64:225–37
Mao X, Chen SX, Wong RK. 2019. Matrix completion with covariate information. *J. Am. Stat. Assoc.* 114:198–210
Mazumder R, Hastie T, Tibshirani R. 2010. Spectral regularization algorithms for learning large incomplete matrices. *J. Mach. Learn. Res.* 11:2287–322
McCullagh P. 1987. *Tensor Methods in Statistics*. Boca Raton, FL: Chapman and Hall
Miranda MF, Zhu H, Ibrahim JG. 2018. TPRM: tensor partition regression models with applications in imaging biomarker detection. *Ann. Appl. Stat.* 12:1422–50
Mørup M, Hansen LK, Herrmann CS, Parnas J, Arnfred SM. 2006. Parallel factor analysis as an exploratory tool for wavelet transformed event-related EEG. *NeuroImage* 29:938–47
Mu C, Huang B, Wright J, Goldfarb D. 2014. Square deal: lower bounds and improved relaxations for tensor recovery. In *Proceedings of the 31st International Conference on International Conference on Machine Learning*, ed. EP Xing, T Jebara, pp. 73–81. Brookline, MA: Microtome
Nickel M, Tresp V, Kriegel HP. 2011. A three-way model for collective learning on multi-relational data. In *Proceedings of the 28th International Conference on International Conference on Machine Learning*, ed. L Getoor, T Scheffer, pp. 809–16. Madison, WI: Omnipress

Novikov A, Podoprikhin D, Osokin A, Vetrov D. 2015. Tensorizing neural networks. In *Advances in Neural Information Processing Systems 28 (NIPS 2015)*, ed. C Cortes, ND Lawrence, DD Lee, M Sugiyama, R Garnett, pp. 442–50. Red Hook, NY: Curran

Padovani C. 2000. On the derivative of some tensor-valued functions. *J. Elast.* 58:257–68

Pan Y, Mai Q, Zhang X. 2019. Covariate-adjusted tensor classification in high dimensions. *J. Am. Stat. Assoc.* 114:1305–19

Pearson KJ, Zhang T. 2014. On spectral hypergraph theory of the adjacency tensor. *Gr. Comb.* 30:1233–48

Rabanser S, Shchur O, Günnemann S. 2017. Introduction to tensor decompositions and their applications in machine learning. arXiv:1711.10781 [stat.ML]

Raskutti G, Yuan M, Chen H. 2019. Convex regularization for high-dimensional multiresponse tensor regression. *Ann. Stat.* 47:1554–84

Rubin DB. 1976. Inference and missing data. *Biometrika* 63:581–92

Shah P, Rao N, Tang G. 2015. Optimal low-rank tensor recovery from separable measurements: four contractions suffice. arXiv:1505.04085 [stat.ML]

Sidiropoulos ND, Bro R. 2000. On the uniqueness of multilinear decomposition of n-way arrays. *J. Chemom.* 14:229–39

Sun J, Tao D, Faloutsos C. 2006. Beyond streams and graphs: dynamic tensor analysis. In *Proceedings of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 374–83. New York: ACM

Sun J, Tao D, Papadimitriou S, Yu PS, Faloutsos C. 2008. Incremental tensor analysis: theory and applications. *ACM Trans. Knowl. Discov. Data* 2:1–37

Sun WW, Li L. 2017. STORE: sparse tensor response regression and neuroimaging analysis. *J. Mach. Learn. Res.* 18:4908–44

Sun WW, Li L. 2019. Dynamic tensor clustering. *J. Am. Stat. Assoc.* 114:1894–907

Tang X, Bi X, Qu A. 2019. Individualized multilayer tensor learning with an application in imaging analysis. *J. Am. Stat. Assoc.* 115:836–51

Tang X, Li L. 2020. Multivariate temporal point process regression. arXiv:2001.00719 [stat.ME]

Tarzanagh DA, Michailidis G. 2019. Regularized and smooth double core tensor factorization for heterogeneous data. arXiv:1911.10454 [stat.ML]

Ten Berge J, Sidiropoulos N. 2002. On uniqueness in CANDECOMP/PARAFAC. *Psychometrika* 67:399–409

Tucker LR. 1966. Some mathematical notes on three-mode factor analysis. *Psychometrika* 31:279–311

van den Berg E. 2019. The ocean tensor package. *J. Open Res. Softw.* 7:26

Vasilescu MAO. 2002. Human motion signatures: analysis, synthesis, recognition. In *Proceedings of the 16th International Conference on Pattern Recognition*, pp. 456–60. New York: IEEE

Vasilescu MAO, Terzopoulos D. 2002. Multilinear analysis of image ensembles: TensorFaces. In *Computer Vision—ECCV 2002*, ed. A Heyden, G Sparr, M Nielsen, P Johansen, pp. 447–60. New York: Springer

Wang X, Donaldson R, Nell C, Gorniak P, Ester M, Bu J. 2016. Recommending groups to users using user-group engagement and time-dependent matrix factorization. In *Thirtieth AAAI Conference on Artificial Intelligence*, pp. 1331–37. Menlo Park, CA: AAAI

Wimalawarne K, Tomioka R, Sugiyama M. 2016. Theoretical and experimental analyses of tensor-based regression and classification. *Neural Comput.* 28:686–715

Xiong L, Chen X, Huang TK, Schneider J, Carbonell JG. 2010. Temporal collaborative filtering with Bayesian probabilistic tensor factorization. In *Proceedings of the 2010 SIAM International Conference on Data Mining*, ed. S Parthasarathy, B Liu, B Goethals, J Pei, C Kamath, pp. 211–22. Philadelphia: SIAM

Yang Y, Dunson DB. 2016. Bayesian conditional tensor factorizations for high-dimensional classification. *J. Am. Stat. Assoc.* 111:656–69

Yu HF, Rao N, Dhillon IS. 2016. Temporal regularized matrix factorization for high-dimensional time series prediction. In *Advances in Neural Information Processing Systems 29 (NIPS 2016)*, ed. DD Lee, M Sugiyama, UV Luxburg, I Guyon, R Garnett, pp. 847–55. Red Hook, NY: Curran

Yuan M, Liu R, Feng Y, Shang Z. 2018. Testing community structures for hypergraphs. arXiv:1810.04617 [math.ST]

Yuan M, Zhang CH. 2016. On tensor completion via nuclear norm minimization. *Found. Comput. Math.* 16:1031–68

Yuan M, Zhang CH. 2017. Incoherent tensor norms and their applications in higher order tensor completion. *IEEE Trans. Inform. Theory* 63:6753–66

Zhang A. 2019. Cross: efficient low-rank tensor completion. *Ann. Stat.* 47:936–64

Zhang X, Li L. 2017. Tensor envelope partial least-squares regression. *Technometrics* 59:426–36

Zhang Y, Bi X, Tang N, Qu A. 2020. Dynamic tensor recommender systems. arXiv:2003.05568 [stat.ME]

Zhang Z, Allen GI, Zhu H, Dunson D. 2019. Tensor network factorizations: relationships between brain structural connectomes and traits. *Neuroimage* 197:330–43

Zhou H, Li L, Zhu H. 2013. Tensor regression with applications in neuroimaging data analysis. *J. Am. Stat. Assoc.* 108:229–39

Zhou S, Erfani SM, Bailey J. 2017. SCED: a general framework for sparse tensor decomposition with constraints and elementwise dynamic learning. In *2017 IEEE International Conference on Data Mining*, pp. 675–84. New York: IEEE

Zhu Y, Shen X, Ye C. 2016. Personalized prediction and sparsity pursuit in latent factor models. *J. Am. Stat. Assoc.* 111:241–52