Robust Low-rank Tensor Decomposition with the L_2 Criterion

Qiang Heng[*], Eric C. Chi[†], and Yufeng Liu[‡]

Abstract

The growing prevalence of tensor data, or multiway arrays, in science and engineering applications motivates the need for tensor decompositions that are robust against outliers. In this paper, we present a robust Tucker decomposition estimator based on the L_2 criterion, called the Tucker-L_2E. Our numerical experiments demonstrate that Tucker-L_2E has empirically stronger recovery performance in more challenging high-rank scenarios compared with existing alternatives. The appropriate Tucker-rank can be selected in a data-driven manner with cross-validation or hold-out validation. The practical effectiveness of Tucker-L_2E is validated on real data applications in fMRI tensor denoising, PARAFAC analysis of fluorescence data, and feature extraction for classification of corrupted images.

Keywords: inverse problem, L_2 criterion, nonconvexity, robustness, Tucker decomposition

1 Introduction

There has been growing interest in tensors, or multi-way arrays, since many real-world datasets have a multi-dimensional structure that is not well exploited by two-dimensional matrix-based data analysis. Some of the most important tensor-based data analysis tools are low-rank tensor decompositions, which primarily take two forms: the CANDECOMP/PARAFAC (CP) decomposition (Carroll and Chang, 1970; Harshman et al., 1970)

[*]Department of Statistics, North Carolina State University
[†]Department of Statistics, Rice University
[‡]Department of Statistics and Operations Research, The University of North Carolina at Chapel Hill

and the Tucker decomposition (Tucker, 1966). In the last decade, several new tensor decomposition paradigms based on alternative notions of tensor rank have been proposed, including low-tubal-rank factorization (Kilmer and Martin, 2011), tensor-train decomposition (Oseledets, 2011) and tensor ring decomposition (Zhao et al., 2016). Each decomposition has suitable applications as well as limitations.

A major challenge of low-rank tensor decomposition is that the observed tensor may be grossly corrupted with outliers or sparse noise. This paper addresses the robust tensor decomposition problem when the underlying tensor has low Tucker-rank. When an N-way data tensor $\boldsymbol{\mathcal{X}} \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_N}$ is fully observed, we assume that $\boldsymbol{\mathcal{X}}$ is generated from the following model

$$\boldsymbol{\mathcal{X}} \;=\; \boldsymbol{\mathcal{L}} + \boldsymbol{\mathcal{S}} + \boldsymbol{\mathcal{E}},$$

where $\boldsymbol{\mathcal{L}}$ denotes an underlying low Tucker-rank tensor, $\boldsymbol{\mathcal{S}}$ denotes a sparse tensor of outlying entries, and $\boldsymbol{\mathcal{E}}$ denotes a tensor of dense noise.

We also consider the case when $\boldsymbol{\mathcal{X}}$ is only partially observed over a subset Ω of its indices. Let $[I]$ denote the set of consecutive integers $\{1, \ldots, I\}$. Then the set $\Omega \subset [I_1] \times \cdots \times [I_N]$ is an index set of observed entries, and we assume that

$$x_{i_1 i_2 \cdots i_N} \;=\; l_{i_1 i_2 \cdots i_N} + s_{i_1 i_2 \cdots i_N} + e_{i_1 i_2 \cdots i_N},$$

for $(i_1, i_2, \cdots, i_N) \in \Omega$.

Our goal is to recover the latent factors of the underlying low-rank tensor $\boldsymbol{\mathcal{L}}$. Ideally, a robust method should remain effective in the absence of $\boldsymbol{\mathcal{S}}$ or $\boldsymbol{\mathcal{E}}$. If the goal is estimating the low-rank tensor $\boldsymbol{\mathcal{L}}$ instead of its latent factors, we refer to robust tensor decomposition as robust tensor recovery. We focus on the Tucker decomposition in this paper. Since the CP decomposition is a special case of the Tucker decomposition when the CP-rank does not exceed any of the tensor dimensions, the proposed method may also be applied to denoise or reconstruct low CP-rank tensors.

Many CP and Tucker decomposition methods have been proposed in the literature. We discuss these in Section 3. Our robust formulation adapts the L_2E method (Scott, 2001, 2009) to the Tucker decomposition. The L_2E is a minimum distance estimator that minimizes the integrated squared error (ISE) for parametric estimation. The integrated squared error is also referred to as the L_2 criterion (Hjort, 1994; Terrell, 1990) in nonpara-

metric density estimation, hence the name L_2E. Consequently, we call our formulation of Tucker decomposition the Tucker-L_2E. Minimum distance estimators are well known to possess robustness properties (Donoho and Liu, 1988). Moreover, minimization of the L_2 criterion has been employed in developing a wide range of robust statistical models including structured sparse models (Lozano et al., 2016; Chi and Chi, 2022; Liu et al., 2023), quantile regression (Lane, 2012), mixture models (Lee, 2010), classification (Chi and Scott, 2014), forecast aggregation (Ramos, 2014), and survival analysis (Yang and Scott, 2013). It also has successes in engineering applications including signal processing tasks such as wavelet-based image denoising (Scott, 2006) and image registration (Ma et al., 2013, 2015; Yang et al., 2017). The L_2E is attractive among minimum distance estimators since it strikes a good balance between robustness, efficiency, and computational tractability (Scott, 2001, 2009).

The rest of this article is organized as follows. In Section 2, we introduce tensor notation and terminology. In Section 3, we briefly review several prominent CP and Tucker decomposition formulations, both non-robust and robust, in the existing literature. In Section 4, we present our formulation of robust Tucker decomposition and its solution algorithm. In Section 5 and Section 6, we demonstrate the practical effectiveness of our approach and its advantage over existing methods in terms of recovery capability with numerical experiments and real data applications.

2 Background on Tensors and their Decompositions

We review basic operations on matrices and tensors using the terminology and notation in Kolda and Bader (2009). A tensor $\boldsymbol{\mathcal{X}} \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_N}$ is an element of the tensor product of N real vector spaces. The number of dimensions, or ways, N is called the *order* of tensor $\boldsymbol{\mathcal{X}}$. Each dimension is also called a *mode*. A *fiber* of $\boldsymbol{\mathcal{X}}$ is a column vector subset of $\boldsymbol{\mathcal{X}}$, defined by fixing all but one of the indices. For a matrix, an order-2 tensor, a mode-1 fiber is a matrix column, and a mode-2 fiber is a matrix row. A *slice* of $\boldsymbol{\mathcal{X}}$ is a matrix subset of $\boldsymbol{\mathcal{X}}$, defined by fixing all but two of the indices.

2.1 Basic Tensor Operations

It is often convenient to reshape a tensor into a matrix or a vector. The former is referred to as *matricization*, while the latter is referred to as *vectorization*. The mode-n

matricization of a tensor $\mathcal{X} \in \mathbb{R}^{I_1 \times \cdots \times I_N}$, denoted $\mathbf{X}_{(n)} \in \mathbb{R}^{I_n \times I_{-n}}$ with $I_{-n} = \prod_{k=1,k\neq n}^{N} I_k$, arranges the mode-$n$ fibers as the columns of the matrix $\mathbf{X}_{(n)}$ in the following lexicographic order. The tensor element $x_{i_1,\ldots,i_N}$ is mapped to the matrix element of $\mathbf{X}_{(n)}$ with index (i_n, j) where $j = 1 + \sum_{k=1,k\neq n}^{N}(i_k - 1)J_k$ and

$$J_k \quad = \quad \begin{cases} 1 & \text{if } k = 1 \text{ or if } k = 2 \text{ and } n = 1, \\ \prod_{k'=1,k'\neq i}^{k-1} I_{k'} & \text{otherwise.} \end{cases}$$

The vectorization of $\mathcal{X}$, denoted as $\text{vec}(\mathcal{X})$, is the vector obtained by stacking the columns of its mode-1 matricization $\mathbf{X}_{(1)}$ on top of each other.

We will use two kinds of products involving tensors and matrices throughout this paper. The elementwise *Hadamard product* of two tensors $\mathcal{X}$ and $\mathcal{Y}$ of identical size $I_1 \times \cdots \times I_N$ is denoted by $\mathcal{X} * \mathcal{Y}$ and is the tensor whose $(i_1, \ldots, i_N)$-th element is given by $x_{i_1 i_2 \ldots i_N} y_{i_1 i_2 \ldots i_N}$. The *n-mode product* of a tensor $\mathcal{X} \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_N}$ with a matrix $\mathbf{A} \in \mathbb{R}^{J \times I_n}$ is denoted by $\mathcal{X} \times_n \mathbf{A}$, which is a tensor of size $I_1 \times I_2 \times \cdots \times I_{n-1} \times J \times I_{n+1} \times \cdots \times I_N$ with elements

$$(\mathcal{X} \times_n \mathbf{A})_{i_1 \cdots i_{n-1} j i_{n+1} \cdots \iota_N} \quad = \quad \sum_{i_n=1}^{I_n} x_{i_1 i_2 \cdots i_N} a_{j i_n}$$

for $j \in [J]$. Note that the mode-n matricization of the n-mode product $\mathcal{X} \times_n \mathbf{A}$ can be expressed as

$$[\mathcal{X} \times_n \mathbf{A}]_{(n)} \quad = \quad \mathbf{A}\mathbf{X}_{(n)}.$$

The *Frobenius norm* and ℓ_1*-norm* of a tensor $\mathcal{X} \in \mathbb{R}^{I_1 \times \cdots \times I_N}$ are defined as

$$\|\mathcal{X}\|_{\text{F}} \quad = \quad \sqrt{\sum_{i_1=1}^{I_1} \sum_{i_2=1}^{I_2} \cdots \sum_{i_N=1}^{I_N} x_{i_1 i_2 \ldots i_N}^2} \qquad \text{and} \qquad \|\mathcal{X}\|_1 \quad = \quad \sum_{i_1=1}^{I_1} \sum_{i_2=1}^{I_2} \cdots \sum_{i_N=1}^{I_N} |x_{i_1 i_2 \ldots i_N}|.$$

Finally, we use $\mathcal{X}^{*2}$ to denote the tensor obtained by raising each entry of $\mathcal{X}$ to the power of 2. We use $a\mathcal{X} + b$ to denote the tensor obtained by multiplying every entry of $\mathcal{X}$ by a and then adding b to every element of the resulting scaled tensor. We denote the sum of all tensor entries as $\text{sum}(\mathcal{X})$ and the elementwise exponential of a tensor as $\exp(\mathcal{X})$.

2.2 Tensor decompositions and ranks

The Tucker decomposition of $\mathcal{X} \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_N}$ with rank $R = (r_1, r_2, \ldots, r_N)$ aims to find a *core tensor* $\mathcal{G} \in \mathbb{R}^{r_1 \times r_2 \times \cdots \times r_N}$ and *factor matrices* $\mathbf{A}^{(n)} \in \mathbb{R}^{I_n \times r_n}$ for $n \in [N]$ such that

$$\mathcal{X} \approx \mathcal{G} \times_1 \mathbf{A}^{(1)} \times_2 \mathbf{A}^{(2)} \times_3 \cdots \times_N \mathbf{A}^{(N)} = [\![\mathcal{G}; \mathbf{A}^{(1)}, \ldots, \mathbf{A}^{(N)}]\!],$$

where the equality uses the more compact notation $[\![\mathcal{G}; \mathbf{A}^{(1)}, \ldots, \mathbf{A}^{(N)}]\!]$ introduced in Kolda (2006). Sometimes the columns of $\mathbf{A}^{(n)}$ are required to be orthogonal so that the columns of $\mathbf{A}^{(n)}$ can be interpreted as the principal components of the n-th mode, but we do not require this in this work. The tensor $\mathcal{X}$ is said to have *Tucker-rank* $R = (r_1, r_2, \ldots, r_N)$ if $\text{rank}(\mathbf{X}_{(n)}) = r_n$ for $n \in [N]$.

The CP decomposition for $\mathcal{X}$ with rank $R = r$ aims to find $\mathbf{a}_i^{(n)} \in \mathbb{R}^{I_n}$ for $n \in [N], i \in [r]$, and a weight vector $\boldsymbol{\gamma} \in \mathbb{R}^N$ such that

$$\mathcal{X} \approx \sum_{i=1}^{r} \gamma_i \mathbf{a}_i^{(1)} \circ \mathbf{a}_i^{(2)} \circ \cdots \circ \mathbf{a}_i^{(N)},$$

where $\circ$ denotes the outer product. Just as the outer product of two vectors yields a rank-1 matrix, the outer product of N vectors yields an N-way rank-1 tensor. Thus, the CP model aims to approximate a tensor with a linear combination of rank-1 tensors. Following Kolda and Bader (2009), we write the linear combination of rank-1 tensors $\sum_{i=1}^{r} \gamma_i \mathbf{a}_i^{(1)} \circ \mathbf{a}_i^{(2)} \circ \cdots \circ \mathbf{a}_i^{(N)}$ more compactly as $[\![\boldsymbol{\gamma}; \mathbf{A}^{(1)}, \ldots, \mathbf{A}^{(N)}]\!]$, where $\mathbf{A}^{(n)} = \left[\mathbf{a}_1^{(n)} \ \mathbf{a}_2^{(n)} \ \ldots \ \mathbf{a}_r^{(n)}\right] \in \mathbb{R}^{I_n \times r}$ are the *CP factor matrices*. The tensor $\mathcal{X}$ is said to have *CP-rank* r if r is the smallest integer possible for the approximation to hold with equality. When $r \leq \min\{I_1, I_2, \cdots, I_N\}$, the CP decomposition can be viewed as a special case of Tucker decomposition. This is because if $\mathcal{G}$ has dimension $(r, r, \ldots, r)$ and is "superdiagonal," i.e., its only nonzero entries are $g_{ii\ldots i}$ for $i \in [r]$, then

$$\mathcal{G} \times_1 \mathbf{A}^{(1)} \times_2 \mathbf{A}^{(2)} \times_3 \cdots \times_N \mathbf{A}^{(N)} = [\![\mathbf{g}; \mathbf{A}^{(1)}, \mathbf{A}^{(2)}, \ldots, \mathbf{A}^{(N)}]\!],$$

where $\mathbf{g} \in \mathbb{R}^r$ is a vector containing the superdiagonal of nonzero entries of $\mathcal{G}$. A tensor with CP-rank r has Tucker-rank $(r, r, \ldots, r)$, but the converse does not hold in general.

For notational simplicity, we will often "absorb" the weight vector into one of the

factor matrices when writing the CP model, e.g.,

$$\left[\!\left[\boldsymbol{\gamma}; \mathbf{A}^{(1)}, \mathbf{A}^{(2)}, \ldots, \mathbf{A}^{(N)}\right]\!\right] \;=\; \left[\!\left[\tilde{\mathbf{A}}^{(1)}, \mathbf{A}^{(2)}, \ldots, \mathbf{A}^{(N)}\right]\!\right],$$

where $\tilde{\mathbf{A}}^{(1)} = \mathbf{A}^{(1)}\text{diag}(\boldsymbol{\gamma})$ and $\text{diag}(\boldsymbol{\gamma})$ is the diagonal matrix with i-th diagonal entry γ_i.

3 Related Work

Tensor decompositions based on least squares Non-robust CP and Tucker decompositions are formulated as the solutions to nonlinear least squares problems that minimize the Frobenius norm of the residual tensor. Formally, the CP decomposition solves the following optimization problem

$$\underset{\mathbf{A}^{(1)},\ldots,\mathbf{A}^{(n)}}{\text{minimize}} \quad \left\|\boldsymbol{\mathcal{X}} - \left[\!\left[\mathbf{A}^{(1)}, \mathbf{A}^{(2)}, \cdots, \mathbf{A}^{(N)}\right]\!\right]\right\|_{\text{F}}^{2}. \tag{1}$$

Historically, the alternating least squares (ALS) algorithm (Carroll and Chang, 1970; Harshman et al., 1970) has been the "work-horse" of solving the above CP decomposition problem, which updates one of the factor matrices while holding the others fixed. Acar et al. (2011a), however, showed that a direct optimization approach can obtain more accurate estimates of the low-rank tensor, especially when the specified rank is greater than the true rank. By direct optimization, we mean that the gradients with respect to the factor matrices are computed and all the factor matrices are updated "all-at-once" or simultaneously with a local optimization method like nonlinear conjugate gradient method (NCG) or Limited-memory Broyden–Fletcher–Goldfarb–Shanno algorithm (L-BFGS). We refer to this direct optimization approach as CP-OPT.

Similarly, the Tucker decomposition, or the best rank-$(r_1, r_2, \ldots, r_N)$ approximation of $\boldsymbol{\mathcal{X}}$, is formulated as

$$\underset{\mathcal{G},\mathbf{A}^{(1)},\ldots,\mathbf{A}^{(N)}}{\text{minimize}} \quad \left\|\boldsymbol{\mathcal{X}} - \left[\!\left[\mathcal{G}; \mathbf{A}^{(1)}, \ldots, \mathbf{A}^{(N)}\right]\!\right]\right\|_{\text{F}}^{2}. \tag{2}$$

The first method to compute the Tucker decomposition introduced in Tucker (1966) was later shown by De Lathauwer et al. (2000a) to be a generalization of the matrix singular value decomposition (SVD), known today as *Higher-order Singular Value Decomposition* (HOSVD). However, it does not produce the best fit in terms of relative error. An

alternating least squares algorithm named *Higher-order Orthogonal Iteration* (HOOI) (Kroonenberg and De Leeuw, 1980; Kapteyn et al., 1986; De Lathauwer et al., 2000b) has stronger empirical performance and is the most widely adopted method to compute the Tucker decomposition. In fact, it has also been shown that HOSVD achieves a sub-optimal rate of estimation error, while HOOI is information-theoretic optimal (Zhang and Xia, 2018).

The above least squares formulations reflect a Gaussian assumption. This assumption is a natural starting point to develop a robust tensor decomposition formulation. Consequently, our Tucker-L_2E method is derived under it. Nonetheless, it is important to note that there have been recent works extending tensor decomposition under non-Gaussian modeling assumptions. For example, Hong et al. (2020) studied generalized CP decomposition with various statistically motivated loss functions. Han et al. (2022) studied generalized low Tucker-rank tensor estimation, which establishes an upper bound for statistical error and a linear computational convergence rate.

Robust Principal Component Analysis (RPCA) Perhaps the most classic robust matrix recovery method is *Principal Component Pursuit* (Candès et al., 2011), which decomposes a corrupted matrix $\mathbf{X} \in \mathbb{R}^{m\times n}$ as the sum of a low-rank matrix $\mathbf{L}$ and a matrix of sparse outliers $\mathbf{S}$. This is achieved by solving the following convex optimization problem.

$$\underset{\mathbf{X},\mathbf{S}}{\text{minimize}} \quad \|\mathbf{L}\|_* + \lambda\|\mathbf{S}\|_1 \qquad \text{subject to} \quad \mathbf{L}+\mathbf{S}=\mathbf{X}, \tag{3}$$

where $\|\cdot\|_*$ denotes the nuclear norm and λ is a nonnegative tuning parameter. Candès et al. (2011) proved that (3) achieves exact recovery of the low-rank component $\mathbf{L}$ under low-rank and incoherence assumptions. Since the matricizations of a low Tucker-rank tensor are low-rank matrices, RPCA is often used as a baseline for robust tensor recovery.

Higher-order RPCA Goldfarb and Qin (2014) proposed *Higher-order Robust Principal Component Analysis* (HoRPCA) as a generalization of RPCA to tensors. HoRPCA comes in several different variations. We discuss the three best-performing variants in

this section. The first variant is the *singleton model* (HoRPCA-S), formulated as

$$\underset{\boldsymbol{\mathcal{L}},\boldsymbol{\mathcal{S}}}{\text{minimize}} \quad \sum_{i=1}^{N} \|\mathbf{L}_{(i)}\|_* + \lambda \|\boldsymbol{\mathcal{S}}\|_1 \qquad \text{subject to} \quad \boldsymbol{\mathcal{L}} + \boldsymbol{\mathcal{S}} = \boldsymbol{\mathcal{X}}. \tag{4}$$

HoRPCA-S minimizes the sum of nuclear norms of all the matricizations of $\boldsymbol{\mathcal{L}}$ to encourage each mode to be low-rank. The descriptor "singleton" is in contrast with the *mixture model* (HoRPCA-W), formulated as

$$\underset{\boldsymbol{\mathcal{L}}_i,\boldsymbol{\mathcal{S}}}{\text{minimize}} \quad \sum_{i=1}^{N} \|\mathbf{L}_{i,(i)}\|_* + \lambda \|\boldsymbol{\mathcal{S}}\|_1 \qquad \text{subject to} \quad \sum_{i=1}^{N} \boldsymbol{\mathcal{L}}_i + \boldsymbol{\mathcal{S}} = \boldsymbol{\mathcal{X}}. \tag{5}$$

HoRPCA-W represents the underlying tensor as the sum of N tensors that are only low-rank in one mode. Tomioka et al. (2011) first introduced the mixture model which can be considered a relaxation of the singleton model. Yang et al. (2015) later proposed robust tensor recovery also using the mixture model along with more robust loss functions. The mixture model can automatically detect the rank-deficient modes and yields better recovery results when the underlying tensor is only low-rank in certain modes. However, in our experience, the limitation of the mixture model is that it does not approximate the low-rank tensor well when the minimum rank in the Tucker rank tuple is relatively large. The variant with the strongest recovery performance presented in Goldfarb and Qin (2014) is the *constrained nonconvex model* (HoRPCA-C), formulated as

$$\underset{\boldsymbol{\mathcal{L}},\boldsymbol{\mathcal{S}}}{\text{minimize}} \quad \|\boldsymbol{\mathcal{S}}\|_1 \qquad \text{subject to} \quad \boldsymbol{\mathcal{L}} + \boldsymbol{\mathcal{S}} = \boldsymbol{\mathcal{X}}, \quad \text{rank}(\mathbf{L}_{(i)}) \leq r_i, \ i \in [N]. \tag{6}$$

The key difference between the formulation in (6) and those in (4) and (5) is that (6) enforces a hard constraint on the Tucker-rank $(r_1, r_2, \ldots, r_N)$ whereas the other formulations trade off the rank of the latent tensor $\boldsymbol{\mathcal{L}}$ with the ℓ_1-norm of the outlier tensor $\boldsymbol{\mathcal{S}}$ via the penalty parameter λ.

Goldfarb and Qin (2014) iteratively solves (4), (5) and (6) using the alternating direction method of multipliers (ADMM) algorithm (Boyd et al., 2011). In particular, since (6) is nonconvex, the standard convergence guarantees of ADMM for convex programs do not apply. However, (6) demonstrates strong empirical convergence in practice.

Bayesian Robust Tensor Factorization Zhao et al. (2015) approached the robust CP decomposition problem in a generative manner with *Bayesian Robust Tensor Factor-*

ization. (BRTF). Normal-gamma priors are used to induce column sparsity of the factor matrices and elementwise sparsity of the outliers. We direct readers to Zhao et al. (2015) for the detailed hierarchical model setup. What makes BRTF attractive is that it can automatically infer the appropriate CP-rank.

Riemannian Gradient Descent Recently, Cai et al. (2022) introduced a general framework under a low-rank plus sparse tensor model. The algorithm is based on Riemannian gradient descent and a novel gradient pruning procedure, which is able to estimate both the low-rank tensor and the outlying sparse tensor. The appropriate Tucker-rank and sparsity level of outliers can be tuned with a BIC-type criterion. Performance bounds for both the low-rank and the sparse tensors are established under suitable conditions. The proposed algorithm is also applicable to Bernoulli and Poisson distributed data. We refer to the algorithm described in Cai et al. (2022) as RGrad in this article.

Partial observations Real-world tensor data is often not fully observed. In the tensor completion literature, a binary weight tensor $\mathcal{W}$ whose entries are 0/1 to indicate missing or observed is often used as a mask to model missing entries. For example, CP and Tucker decomposition in the presence of missing data can be formulated as

$$\underset{\mathbf{A}^{(1)},\ldots,\mathbf{A}^{(N)}}{\text{minimize}} \quad \left\| \mathcal{W} * \left(\mathcal{X} - \left[\!\left[\mathbf{A}^{(1)}, \ldots, \mathbf{A}^{(N)} \right]\!\right] \right) \right\|_{\mathrm{F}}^2, \tag{7}$$

and

$$\underset{\mathcal{G},\mathbf{A}^{(1)},\ldots,\mathbf{A}^{(N)}}{\text{minimize}} \quad \left\| \mathcal{W} * \left(\mathcal{X} - \left[\!\left[\mathcal{G}; \mathbf{A}^{(1)}, \ldots, \mathbf{A}^{(N)} \right]\!\right] \right) \right\|_{\mathrm{F}}^2. \tag{8}$$

Similar to (1) and (2), (7) and (8) can also be solved with direct optimization, e.g., CP-WOPT (Acar et al., 2011b) and Tucker-WOPT (Filipović and Jukić, 2015). For RPCA and HoRPCA, an easy way to deal with missing data is to enforce the equality constraints only on observed entries. Similarly for BRTF, we can choose to incorporate only the observed tensor entries into the hierarchical model.

It is worth mentioning that there is significant interest in tensor completion in recent research. In the last few years there are many advances on the theoretical front for low CP/Tucker-rank tensor completion. For example, Cai et al. (2019) studied the reconstruction of a low-rank symmetric tensor and proposed a two-stage nonconvex algorithm

which achieves optimal ℓ_∞ statistical accuracy. Building upon Cai et al. (2019), Cai et al. (2023) studied the nonconvex tensor completion problem from an uncertainty quantification perspective. Xia and Yuan (2019) studied the sample size requirement for the exact recovery of a low-rank tensor from a subset of its entries, using a spectral initialization method and gradient descent. Zhang (2019) proposed a novel tensor measurement scheme for low-rank tensor completion. Xia et al. (2021) proposed a procedure for low-rank tensor completion from noisy entries based on spectral initialization and power iteration that is computationally efficient and achieves the optimal rates of convergence. Recently, Tong et al. (2022) developed a scaled gradient descent approach to low-rank tensor completion and regression which converges at a linear rate independent of the condition number of the true tensor.

Remark Among the previously reviewed methods, RPCA, HoRPCA-S and HoRPCA-W induce a low-rank structure using nuclear norm penalties while CP-OPT, HOOI, HoRPCA-C and RGrad require the rank to be explicitly specified. We categorize methods that employ nuclear norm penalties as "penalized formulations" and methods requiring specification of rank as "rank-constrained formulations." Our approach, Tucker-L_2E, is an instance of the latter. In Section 5, we demonstrate that rank-constrained formulations are generally more robust and handle dense noise better than penalized formulations, provided that the ranks are appropriately specified. We also illustrate in Section 5.3 that HoRPCA-C and Tucker-L_2E can tolerate some level of rank overestimation if the noise is sparse.

We acknowledge several other robust CP/Tucker decomposition methods that we have not detailed in this section due to the limitation of space. Anandkumar et al. (2016), for example, proposed an iterative thresholding algorithm for robust tensor decompositions which is designed to recover CP models with orthogonal factors. Gu et al. (2014) studies the statistical performance of a convex formulation of robust tensor decomposition. Wu et al. (2017) uses the Cauchy distribution to handle long-tail noise in CP and Tucker decomposition.

4 Methodology

In this section, we introduce our proposed Tucker-L_2E method. We briefly review the L_2E method in Section 4.1 and then develop Tucker-L_2E in Section 4.2. The algorithm

and implementation details for Tucker-L_2E can be found in Section 4.3.

4.1 The L_2E Method

We first review the parametric estimation framework using the L_2 criterion proposed by Scott (2001, 2009). Let $\phi(x)$ be the unknown true density we aim to estimate and $\tilde{\phi}(x \mid \theta)$ be the density of a member of the family of parametric models specified by the parameter $\theta \in \Theta$. We seek the parameter θ that minimizes the ISE between $\phi(x)$ and $\tilde{\phi}(x \mid \theta)$

$$\int \left[\tilde{\phi}(x \mid \theta) - \phi(x)\right]^2 dx. \tag{9}$$

Of course, recovering θ in this way is impossible in practice since ϕ is unknown. Fortunately, although we cannot minimize the L_2 distance between $\phi(x)$ and $\tilde{\phi}(x \mid \theta)$ directly, we can minimize an unbiased estimate of the distance. To do this, we first expand (9) as

$$\int \tilde{\phi}(x \mid \theta)^2 \, dx - 2 \int \tilde{\phi}(x \mid \theta) \, \phi(x) \, dx + \int \phi(x)^2 \, dx.$$

The second integral is the expectation $\mathbb{E}_X[\tilde{\phi}(X \mid \theta)]$, where X is a random variable drawn from ϕ. Therefore, the sample mean provides an unbiased estimate of this quantity. The third integral does not involve the parameter of interest θ and may be ignored in the computation of a minimizer. The first integral has a closed-form expression for many parametric models. In this work, we assume that $\tilde{\phi}(x \mid \theta)$ is a normal density where θ consists of a mean and precision (inverse standard deviation) parameter. Under this assumption, the integral $\int \tilde{\phi}(x \mid \theta)^2 dx$ can be written as an explicit function of the precision parameter alone. As a concrete example, consider the univariate case and assume that $\theta = (\mu, \tau)$ and $\tilde{\phi}(x \mid \theta)$ is the density function of a normal random variable $X \sim \mathcal{N}(\mu, \tau^{-2})$. Then the L_2E for the univariate mean and precision is

$$\hat{\theta}_{L_2E} \;=\; \arg\min_{\mu,\tau} \, h(\mu, \tau), \tag{10}$$

where

$$h(\mu, \tau) \;=\; \frac{\tau}{2\sqrt{\pi}} - \frac{\tau}{n}\sqrt{\frac{2}{\pi}} \sum_{i=1}^{n} \exp\left(-\frac{\tau^2}{2}(x_i - \mu)^2\right). \tag{11}$$

For a fixed τ, the μ that minimizes (11) approaches the MLE of μ as τ approaches

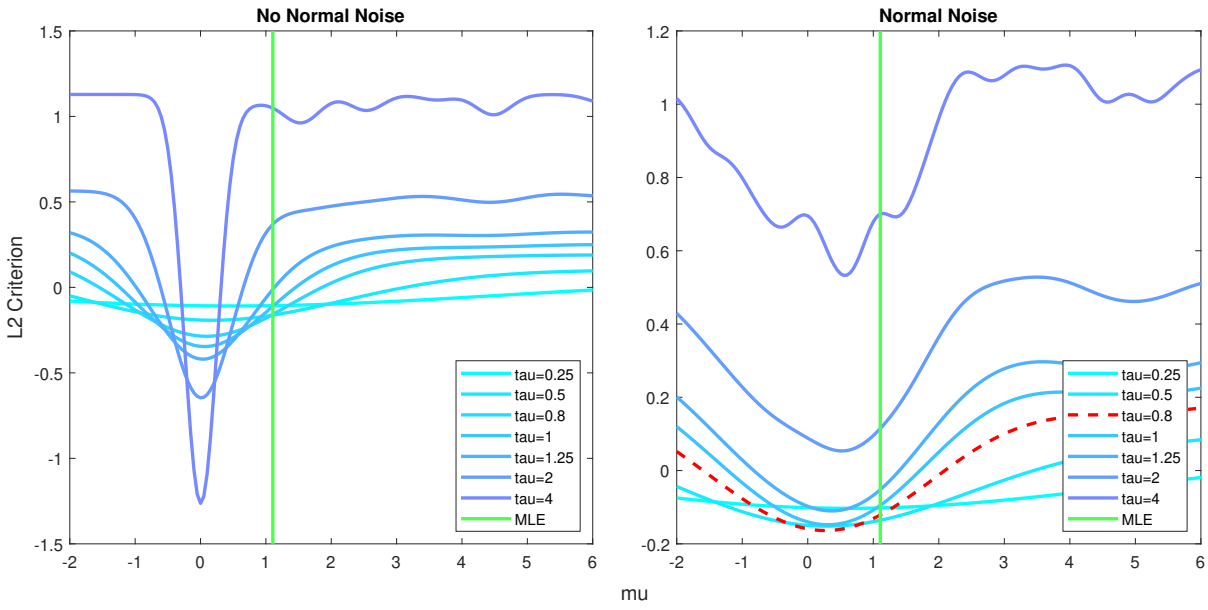


Figure 1: The L_2 criterion as a function of μ with different values of τ. The green vertical line indicates the maximum likelihood estimator of μ, in this case simply $\bar{x}$.

zero. To see this, note that the Taylor expansion of $-\exp(-t)$ around 0 is $-1+t+o(t)$. Therefore for sufficiently small τ,

$$h(\mu,\tau) \approx \frac{\tau}{2\sqrt{\pi}} - \sqrt{\frac{2}{\pi}}\tau + \frac{\tau^3}{n\sqrt{2\pi}}\sum_{i=1}^{n}(x_i-\mu)^2.$$

We can visualize how the L_2 criterion $h(\mu,\tau)$ varies with μ for fixed values of τ to illustrate how the L_2E achieves robustness. We consider two examples. In the first example, $\phi(x)$ is a three-to-one mixture of the constant 0 and Unif$[0,10]$. The second example is identical to the first but the observations are further corrupted with additive $\mathcal{N}(0,1)$ noise. One hundred observations $x_1, x_2, \ldots, x_{100}$ are generated for each example. In both examples, the true value for μ is 0 and the 25% uniformly distributed observations can be regarded as outliers. Intuitively, the true value for τ is $+\infty$ for the first example and 1 for the second example. Figure 1 shows $h(\mu,\tau)$ as a function of μ for different values of τ with the first example in the left panel and the second example in the right panel.

The left panel of Figure 1 shows that the μ that minimizes $h(\mu,\tau)$ is nearly identical to the MLE of μ when τ is very small. As τ increases, the minimizing μ becomes closer to the true value 0. In other words, the L_2 criterion is less influenced by the outliers and consequently, the L_2E for μ is more robust. We also see that if τ becomes too large, however, many spurious local minima appear in the optimization landscape, which may cause difficulties for gradient-based local optimization algorithms.

The right panel of Figure 1 shows that the L_2 criterion curve with $\tau = 0.8$ (highlighted in red dashed line) attains the smallest minimum value. Moreover, the minimum of $h(\mu, \tau)$ with $\tau = 0.8$ is the closest (0.25) to the true value 0. This suggests that (10) is able to automatically choose suitable τ in the presence of normal noise, although it slightly underestimates the precision parameter. Scott (2009) also observed this underestimation phenomenon.

4.2 Tucker-L_2E

We set up the optimization problem for estimating the Tucker-L_2E model in stages. We start with a natural adaptation of (10) to accommodate tensor data which replaces the location parameter μ with a latent mean tensor $\boldsymbol{\mathcal{L}}$. A preliminary formulation of robust tensor estimation based on the L_2 criterion is to minimize the following objective function

$$h\left(\boldsymbol{\mathcal{L}}, \tau\right) \;=\; \frac{\prod_{n=1}^{N} I_n}{2\sqrt{\pi}}\tau - \sqrt{\frac{2}{\pi}}\tau \,\mathrm{sum}\left(\exp\left[-\frac{\tau^2}{2}\left(\boldsymbol{\mathcal{X}} - \boldsymbol{\mathcal{L}}\right)^{*2}\right]\right), \tag{12}$$

where $\boldsymbol{\mathcal{X}} \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_N}$ is the observed noisy tensor.

When the data tensor $\boldsymbol{\mathcal{X}}$ has missing entries and is observed only on the index set Ω, similar to CP-WOPT and Tucker-WOPT, we can sum over only the observed entries in the objective to account for missing data and minimize the natural generalization of the objective function in (12)

$$h_\Omega\left(\boldsymbol{\mathcal{L}}, \tau\right) \;=\; \frac{\mathrm{sum}\left(\boldsymbol{\mathcal{W}}\right)}{2\sqrt{\pi}}\tau - \sqrt{\frac{2}{\pi}}\tau \,\mathrm{sum}\left(\boldsymbol{\mathcal{W}} * \exp\left(-\frac{\tau^2}{2}(\boldsymbol{\mathcal{X}} - \boldsymbol{\mathcal{L}})^{*2}\right)\right). \tag{13}$$

Recall that the tensor $\boldsymbol{\mathcal{W}} \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_N}$ is binary and depends on Ω in the following manner. The $(i_1, i_2, \ldots, i_N)$-th entry of $\boldsymbol{\mathcal{W}}$ is 1 if $x_{i_1 i_2 \ldots i_N} \in \Omega$ and is 0 otherwise. Note that $h_\Omega(\boldsymbol{\mathcal{L}}, \tau)$ and $h(\boldsymbol{\mathcal{L}}, \tau)$ coincide when $\boldsymbol{\mathcal{X}}$ is fully observed, i.e., $\Omega = [I_1] \times \cdots \times [I_N]$. Thus, we will work with the more general objective function h_Ω moving forward.

To estimate a low Tucker-rank tensor, we parameterize $\boldsymbol{\mathcal{L}}$ as $\boldsymbol{\mathcal{L}} = \left[\!\left[\mathcal{G}; \mathbf{A}^{(1)}, \ldots, \mathbf{A}^{(N)}\right]\!\right]$. Notice that this parameterization is equivalent to imposing the constraints $\mathrm{rank}(\mathbf{L}_{(n)}) \leq r_n, n \in [N]$ on $\boldsymbol{\mathcal{L}}$ (see Zhang and Xia (2018)). Thus we seek the solution to the following optimization problem over the parameters $\mathcal{G}, \mathbf{A}^{(1)}, \ldots, \mathbf{A}^{(N)}$ and τ.

$$\text{minimze } h_\Omega\left(\left[\!\left[\mathcal{G}; \mathbf{A}^{(1)}, \ldots, \mathbf{A}^{(N)}\right]\!\right], \tau\right) \qquad \text{subject to} \qquad \tau > 0.$$

We now turn our attention to details concerning the parameter τ. Notice that τ must be positive since it is a precision parameter and consequently introduces an additional constraint over an open set. An easy way to ensure a positive precision while at the same time eliminating the strict positivity constraint is to reparameterize τ as $\tau = \exp(\eta)$ and optimize over $\eta \in \mathbb{R}$. Moreover, recall in Section 4.1, we discussed that although the L_2E is more robust when τ is larger, the accompanying spurious local minima may make computing solutions of (12) harder. In other words, the precision parameter τ trades-off robustness and the "roughness" of the optimization landscape. We also see from the left panel of Figure 1 that the minimum value of $h(\mu, \tau)$ always decreases as τ increases. This suggests that (12) may not even have a finite infimum if we allow τ to diverge to infinity. Therefore, it is reasonable to impose an upper bound on τ, or equivalently on η. Thus, we seek the solution to the following optimization problem over the parameters $\mathcal{G}, \mathbf{A}^{(1)}, \ldots, \mathbf{A}^{(N)}$ and η.

$$\text{minimize } h_\Omega\left(\left[\!\left[\mathcal{G}; \mathbf{A}^{(1)}, \ldots, \mathbf{A}^{(N)}\right]\!\right], e^\eta\right) \qquad \text{subject to} \qquad \eta \leq \eta_{\max}. \tag{14}$$

Theorem 4.1 gives some justification for reparameterizing τ as $\exp(\eta)$ and placing an upper bound on η in (14).

Theorem 4.1. *Problem* (14) *has a finite infimum and the infimum is negative.*

We provide a proof of Theorem 4.1 in the supplement. Theorem 4.1 is relevant in our problem setup since it has the following implications in our computation process. First, we have seen in the left panel of Figure 1 that if the noise is sparse, the L_2E problem may be ill-posed with no finite infimum. Theorem 4.1 assures us that if we place an upper bound on η, we can guarantee a finite infimum. Second, when $\tau = 0$, it is not hard to see from (13) that the objective value will be 0. If the infimum of (14) is guaranteed to be negative, then any point with $\tau = 0$ is suboptimal, thus we do not risk excluding a potential minimizer through reparameterizing τ as $\exp(\eta)$.

Before discussing how to compute Tucker-L_2E next, we emphasize a key feature of our approach is that it simultaneously optimizes or estimates a precision parameter as well as a latent low-rank tensor. As we saw earlier, this precision parameter controls the cutoff for when an entry is large enough to be effectively trimmed from the model fitting. As far as we are aware, Tucker-L_2E is unique in its ability to jointly optimize precision and latent low-rank tensor parameters. Other methods that employ a precision

parameter treat it as a hyperparameter and consequently employ a separate estimation or setting procedure for the precision parameter. For example, Wu et al. (2017) computes an estimate of the precision parameter based on the residuals of the least squares estimate of the low-rank tensor, but this limits the recovery capability of the model as the least squares estimates are of poor quality in challenging high-rank scenarios.

4.3 Solution Algorithm and Implementation Details

The optimization problem in (14) involves differentiable functions of all the model parameters with a simple box constraint on η. The high dimensionality of the parameter space renders second-order algorithms impractical. In contrast, the classic quasi-Newton method L-BFGS-B (Liu and Nocedal, 1989; Byrd et al., 1995; Zhu et al., 1997) is particularly well suited for solving (14). A detailed derivation of the gradient of the objective in (14) with respect to $\mathbf{A}^{(n)}$, $\mathcal{G}$, and η can be found in the supplement. Since the optimization problem in (14) is nonconvex, initialization is critical. We present two initialization strategies in this article. The first is a simple strategy from Filipović and Jukić (2015):

Algorithm 1 Mean Imputation + HOOI/HOSVD

Require: $\mathcal{X}, \mathcal{W} \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_N}, (r_1, r_2, \ldots, r_N)$.

1: Impute missing entries with the mean of the observed entries of $\mathcal{X}$.
2: Compute $\mathcal{G}_0$ and $\mathbf{A}_0^{(n)}$ via HOOI/HOSVD of $\mathcal{X}$ with rank $(r_1, r_2, \ldots, r_N)$.
3: **return** $(\mathcal{G}_0, \mathbf{A}_0^{(n)})$.

The second is a popular initialization procedure in the recent tensor completion literature, which we call spectral initialization with diagonal deletion, see for example Cai et al. (2019); Xia and Yuan (2019); Xia et al. (2021); Tong et al. (2022). In our experiments, we find that this initialization procedure offers some improvement over Algorithm 1 when the underlying tensor has low CP-rank and a large percentage of tensor entries is missing. However, we note that in most cases, Algorithm 1 works similarly or better, especially when the underlying Tucker-rank is relatively high. Therefore our default initialization procedure is Algorithm 1 in our following experiments unless explicitly specified otherwise. We discuss the details of the second initialization strategy in the supplement.

We find the initial value of η to have minimal impact on the solution, consequently we initialize η using a small value $\log(0.01)$, where log is the natural log function. We also observe that numerical issues can occur when the tensor entries are somewhat large in magnitude due to the exponent function in the objective of (14). Scientific computing

languages compute $\exp(-x)$ via a power series so that the numeric precision degrades when x becomes larger in absolute value. Therefore as a pre-processing step, the observed tensor entries are rescaled to have a mean absolute deviation (MAD) of 0.1. We revert the estimated tensor to the original scale after the decomposition is complete. Another practical concern is the choice of Tucker-rank $(r_1, r_2, \ldots, r_N)$ and the upper bound $\eta_{\max}$. In particular, for the upper bound $\eta_{\max}$, we want to select a value such that we have sufficient robustness but the loss landscape is still smooth enough for L-BFGS-B to find a good solution. In Section 5.3 and Section 6.1, we demonstrate that Tucker-rank $(r_1, r_2, \ldots, r_N)$ can be selected in a data-driven manner using cross-validation or hold-out validation if computation is intensive. For $\eta_{\max}$, we find that $\eta_{\max} = \log(50)$ works well for a wide range of problems and we have used it for all of our experiments except the feature extraction application in Section 6.3 where $\eta_{\max}$ is set at $\log(20)$ for optimal performance. It can also be tuned along with the Tucker-rank using cross-validation or hold-out validation if warranted or desired.

Algorithm 2 Tucker-L_2E

Require: $\mathcal{X}, \mathcal{W} \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_N}$, $(r_1, r_2, \ldots, r_N)$, and $\eta_{\max}$.

1: Calculate the MAD of the observed entries of $\mathcal{X}$, denoted as s.
2: Rescale $\mathcal{X}$ as $\frac{1}{10s}\mathcal{X}$.
3: Compute initial estimates $(\mathcal{G}_0, \mathbf{A}_0^{(n)})$ with Algorithm 1 or spectral initialization with diagonal deletion.
4: Set $\eta_0 = \log(0.01)$.
5: Using (14) as the objective and $(\mathcal{G}_0, \mathbf{A}_0^{(n)}, \eta_0)$ as the initial value, update $(\mathcal{G}, \mathbf{A}^{(n)}, \eta)$ with L-BFGS-B until convergence of objective value or the maximum number of iterations is reached. The final iterate is denoted as $(\mathcal{G}_*, \mathbf{A}_*^{(n)}, \eta_*)$.
6: **return** $(10s\mathcal{G}_*, \mathbf{A}_*^{(n)}, \eta_*)$.

Algorithm 2 summarizes our procedure for computing Tucker-L_2E. We implement our algorithm in the MATLAB R2021a computing environment. We use the Tensor Toolbox for MATLAB version 3.2.1[1] (Bader and Kolda, 2006, 2008) for basic tensor classes and operations. We also use the implementation of HOOI and HOSVD in the Tensor Toolbox to compute the initial estimate of $\mathcal{G}$ and $\mathbf{A}^{(n)}$ in Algorithm 1. We use the implementation of L-BFGS-B by Becker (2015)[2]. We direct readers to Byrd et al. (1995) for the algorithmic details of L-BFGS-B. We note that since L-BFGS-B is a gradient-based local optimization algorithm, Algorithm 2 is only able to find a locally optimal or critical point of (14). Additionally, since L-BFGS-B is a descent method (Byrd et al.,

[1] https://www.tensortoolbox.org/
[2] https://github.com/stephenbeckr/L-BFGS-B-C

1995) and Theorem 4.1 ensures that our objective in (14) is bounded below, the sequence of objective function evaluations over the iterate sequence is guaranteed to converge with L-BFGS-B updates.

5 Numerical Experiments

We consider both the CP model and the Tucker model in our simulation studies. The tensor dimension is set at $(50, 50, 50)$. For the CP model, the entries of the factor matrices are independently drawn from $\mathcal{N}(0, 1)$. For the Tucker model, we adopt a similar data generation protocol to Cai et al. (2022). A tensor of size $(50, 50, 50)$ with random normal entries is first generated. Then the tensor is truncated to have Tucker-rank (r, r, r) with HOSVD. We use relative error, defined as $\text{RE} = \|\hat{\boldsymbol{\mathcal{L}}} - \boldsymbol{\mathcal{L}}\|_{\text{F}}/\|\boldsymbol{\mathcal{L}}\|_{\text{F}}$, as the primary goodness-of-fit metric. After generating the low-rank tensor $\boldsymbol{\mathcal{L}}$, we randomly select a fraction δ of the tensor entries to be corrupted with outliers drawn from $\text{Unif}[-M, M]$. We use a relatively large magnitude $M = 10\,\text{std}(\text{vec}(\boldsymbol{\mathcal{L}}))$ in the following experiments. Optionally, a layer of dense Gaussian noise $\boldsymbol{\mathcal{E}}$ can also be added, whose scale is set such that $\|\boldsymbol{\mathcal{E}}\|_{\text{F}}/\|\boldsymbol{\mathcal{L}}\|_{\text{F}} = 0.1$. We use CP-OPT, HOOI, BRTF, HoRPCA-S, HoRPCA-C and RGrad as the baseline methods in this section. We use the implementation of CP-OPT and HOOI in the Tensor Toolbox. The software of HoRPCA-S, HoRPCA-C and BRTF can be found on the authors' websites. The software of RGrad can be found in the supplementary materials of Cai et al. (2022). We provide code and demo examples for our proposed method at `https://github.com/qhengncsu/TuckerL2E`.

5.1 Evaluating Robustness versus Rank

The inverse problem of robust Tucker decomposition becomes more challenging as the underlying Tucker rank or outlier percentage increases. In this section, we contrast the recovery performance of the baseline methods and Tucker-L_2E by generating third-order tensors with increasing CP or Tucker-rank ($R = 5, 10, \ldots, 45$ or $R = (5, 5, 5), (10, 10, 10), \ldots, (45, 45, 45)$) under outlier corruption and in the presence or absence of dense noise. Note that for a tensor with an underlying CP-rank of r, we can still compute a Tucker decomposition with Tucker-rank (r, r, r) to reconstruct the tensor. In this section we keep the outlier sparisty at 25%. For HoRPCA-S, we tune the penalty parameter with the ground truth. For rank-constrained formulations, the specified CP or Tucker-ranks are set to the true

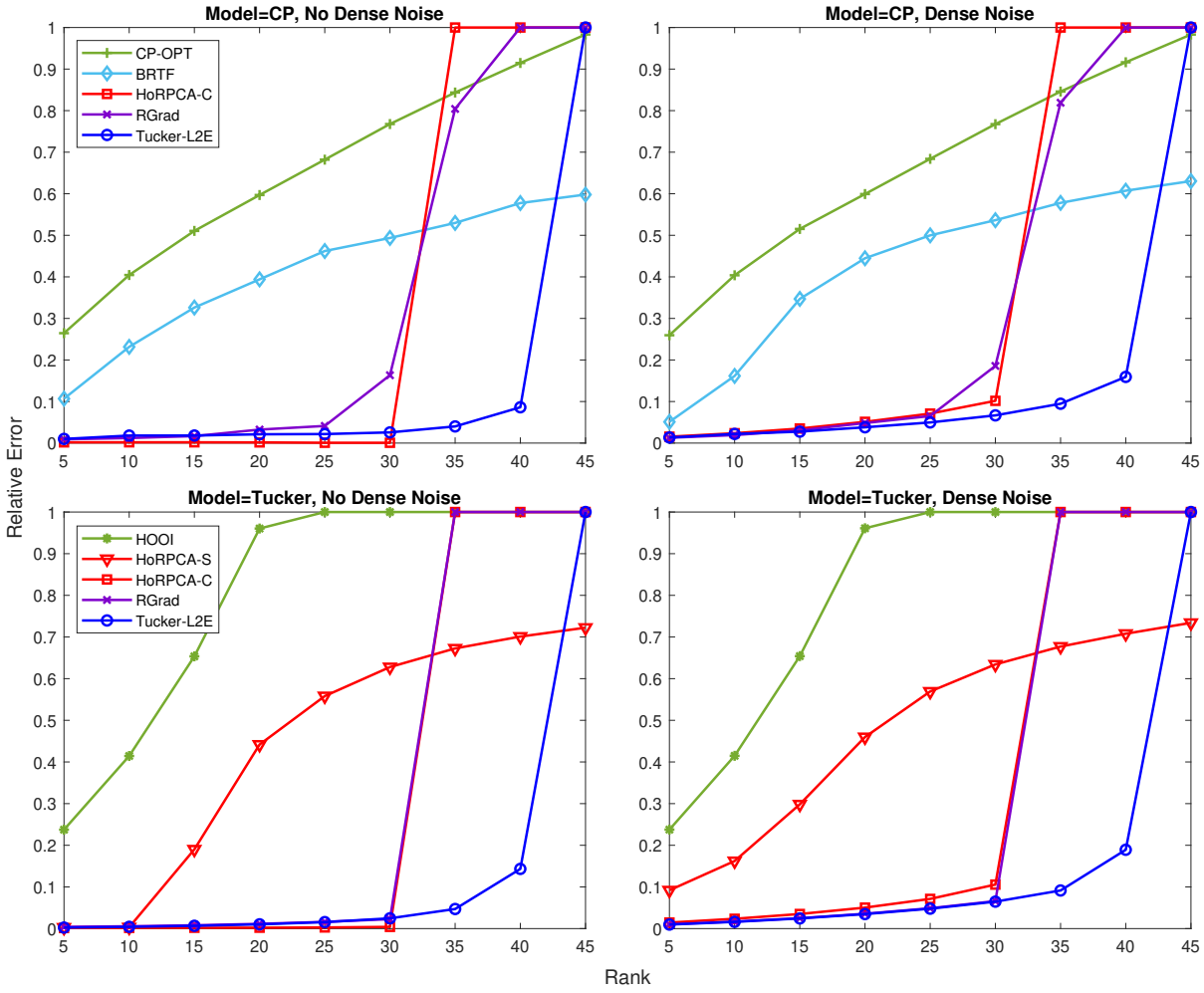


Figure 2: Recovery results on fully observed tensors with increasing CP or Tucker-rank. Outlier sparsity is set at 0.25. Data points are averaged over 50 random replicates. Average relative errors larger than 1 are capped at 1 to suit the display.

ranks. We provide additional details of parameter settings in the supplement.

Figure 2 shows that the convex penalized formulation HoRPCA-S works well when the tensor rank is low but loses accuracy as the rank increases. HoRPCA-C, RGrad and Tucker-L_2E demonstrate competitive recovery performance in most cases, except when the rank is too close to the data dimension, particularly at rank 35-45. Notably, Tucker-L_2E appears to be able to tolerate a higher Tucker-rank than HoRPCA-C and RGrad, both in the CP model and the Tucker model. More specifically, Tucker-L_2E is able to provide a reasonably good reconstruction at rank 35 and 40 while HoRPCA-C and RGrad break down.

5.2 Phase Transition of Rank and Outlier Sparsity

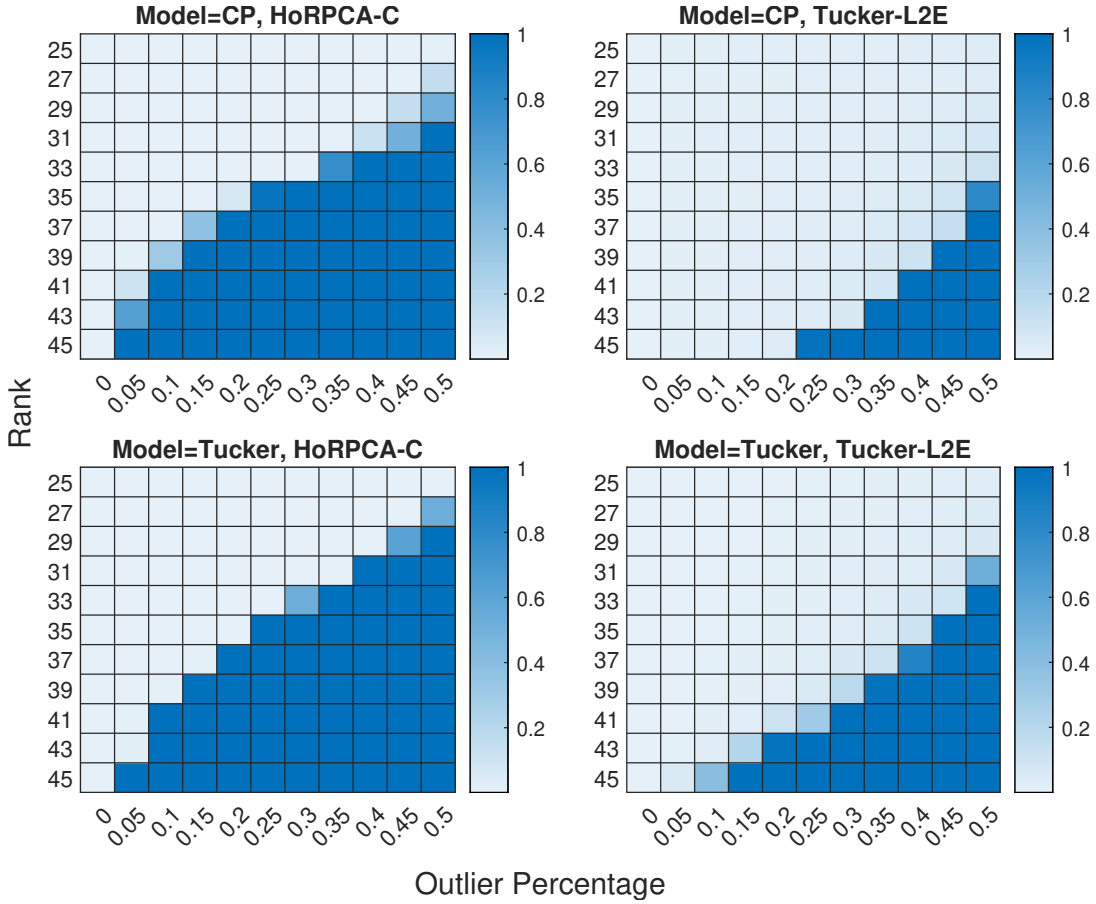


Figure 3: Phase transition diagrams in high-rank scenarios with varying percentages of outliers. Heatmap shows the average relative error of 20 random replicates. Average relative errors larger than 1 are capped at 1 to suit the display.

In the previous section, we saw that compared with HoRPCA-C and RGrad, Tucker-L_2E appears to be able to tolerate a higher rank at the given outlier percentage. Although

RGrad handles dense noise better than HoRPCA-C, HoRPCA-C appears to be a slightly stronger baseline when it comes to stability in high-rank scenarios. To further investigate the phase transition behavior of HoRPCA-C and Tucker-L_2E, we generate tensors with high rank ($R = 25, 27, \ldots, 45$ or $R = (25, 25, 25), (27, 27, 27), \ldots, (45, 45, 45)$) and vary the percentage of outliers taking $\delta = 0, 0.05, \ldots, 0.5$. Full observations are used and no dense noise is added. The specified Tucker-ranks are set to be equal to the true CP or Tucker-ranks.

Figure 3 shows that as the underlying tensor rank increases, the percentage of outliers that can be tolerated decreases for both methods. However, at a given rank, Tucker-L_2E can often handle a greater level of corruption. At rank 25, both methods are able to obtain an accurate reconstruction for any outlier percentage no greater than 0.5. From rank 29 to 41, Tucker-L_2E can generally handle 10-20% more outliers. At rank 43 or 45, Tucker-L_2E can handle 5% more outliers. Interestingly, the advantage of Tucker-L_2E over HoRPCA-C is notably more significant on data generated by the CP model, especially at ranks 43 and 45. This is potentially because a CP-rank of 45 is a more constrained low-rank structure than a general Tucker-rank of (45,45,45).

5.3 Rank Misspecification and Cross Validation

In the previous two sections, we set the specified ranks to be equal to the true ranks for the rank-constrained formulations. In practice, however, such prior knowledge of tensor rank may not always be available. In this section, we investigate how rank-constrained formulations behave when the tensor rank is underestimated or overestimated. We also demonstrate the application of cross-validation to choose the appropriate rank for HoRPCA-C and Tucker-L_2E. We consider three scenarios: 1) the noiseless tensor has CP-rank 15; 2) the noiseless tensor has Tucker-rank (30,10,5); 3) the noiseless tensor has Tucker-rank (35,35,35). After generating the true low-rank tensor, 25% percent of tensor entries are corrupted with outliers. The specified CP-ranks are $5, 10, \ldots, 45$ and the specified Tucker-ranks are $(5, 5, 5), (10, 10, 10), \ldots, (45, 45, 45)$. Notice that although in the second scenario, the noiseless tensor is not equally low-rank in every mode, we still set the specified Tucker-ranks to be equal in every mode simply to limit the number of Tucker-rank tuples that we need to consider.

The cross-validation scheme can be described as follows: the tensor entries are randomly split into 10 folds; robust tensor decomposition methods (HoRPCA-C and Tucker-

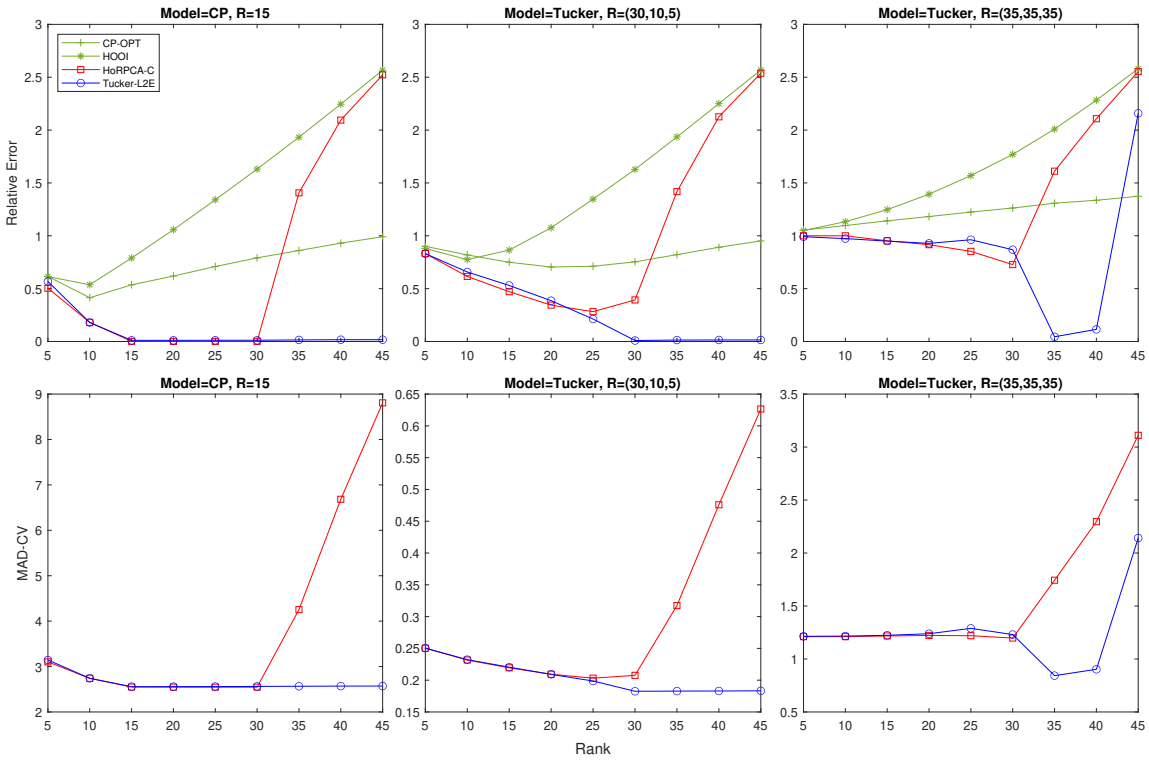


Figure 4: First row: recovery results when the rank is underestimated or overestimated. Second row: 10-fold cross-validation error for a generated tensor.

L_2E) are applied to 9 out of 10 folds, treating the hold-out fold as missing data. This process is repeated for each train/test split; we use the estimated values for the hold-out fold to form a new tensor which we call the predicted tensor; cross-validation error is computed as the MAD between entries of the predicted tensor and entries of the original noisy tensor. The MAD is chosen over the more common mean squared error (MSE) to make the cross-validation error less sensitive to large residuals, which likely coincide with outlying entries.

Figure 4 shows that the non-robust methods (CP-OPT and HOOI) will greatly overfit to the outliers if the tensor rank is overestimated. For the first scenario ($R = 15$), both HoRPCA-C and Tucker-L_2E exhibit a certain level of overfitting resistance. Remarkably, Tucker-L_2E remains unaffected by outliers even if the Tucker-rank is grossly overestimated to be (45,45,45). In the second scenario ($R = (30, 10, 5)$), unlike Tucker-L_2E, HoRPCA-C is not able to achieve perfect recovery if the Tucker-rank is specified to (30,30,30). The first two scenarios demonstrate that Tucker-L_2E is more robust to rank overestimation than HoRPCA-C. The third scenario ($R = (35, 35, 35)$) is chosen to be challenging. Contrary to our expectation, the relative error and cross-validation error for Tucker-L_2E will first increase before it decreases. It is reassuring that the cross-validation error still

achieves its minimum at the true Tucker-rank. Another surprising observation is that HoRPCA-C attains its minimum relative error at Tucker-rank $(30, 30, 30)$ instead of the true rank $(35, 35, 35)$. This is likely because while at Tucker-rank $(30, 30, 30)$, HoRPCA-C is only capable of an approximate reconstruction, it is still better than the true Tucker-rank $(35, 35, 35)$ where HoRPCA-C becomes unstable.

5.4 Varying Degrees of Missingness

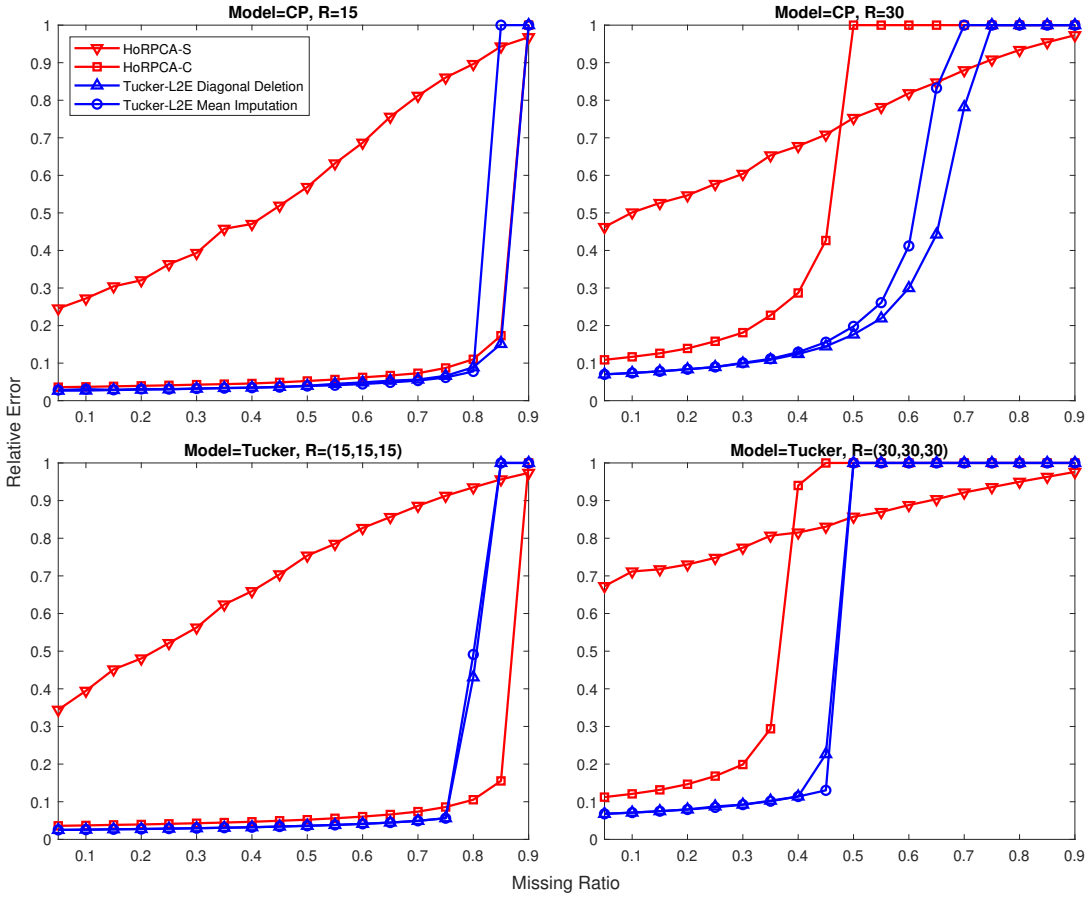


Figure 5: Recovery results under varying degrees of missingness. Outlier sparsity is set at 25%. Data points are averaged over 20 random replicates. Average relative errors larger than 1 are capped at 1 to suit the display.

In this section we investigate the recovery performance of HoRPCA-S, HoRPCA-C and Tucker-L_2E under varying degrees of missingness. RGrad is not considered in this section since its current form does not allow missing entries. We generate $(50, 50, 50)$ tensors with CP-rank 15, 30 or Tucker-rank $(15, 15, 15)$, $(30, 30, 30)$. After generating the low-rank tensor, 25% of the entries are corrupted with outliers and dense noise of relative scale 0.1 is added. Then $\rho \times 100\%$ of the tensor entries are set to be missing.

The missingness is assumed to be completely random and independent from the outlier corruption. We vary the missing ratio ρ from 0.05 to 0.9 in this section. We present the recovery results of both initialization methods.

In Figure 5, "Tucker-L_2E Diagonal Deletion" refers to the recovery results using spectral initialization with diagonal deletion and "Tucker-L_2E Mean Imputation" refers to the recovery results using Algorithm 1 as the initialization. We can see that the diagonal deletion procedure offers some improvement over Algorithm 1 when the underlying tensor is of low CP-rank. Tucker-L_2E appears to be less stable than HoRPCA-C when the rank is low and the missing percentage is very high (over 80%). This may be attributed to the fact that HoRPCA-C models missing data with an equality constraint so that the unobserved entries are still penalized for having a large magnitude, while for Tucker-L_2E the unobserved entries are masked and unconstrained. When the rank is relatively high, we can see that Tucker-L_2E still enjoys an empirical advantage over HoRPCA-C in terms of recovery capability.

6 Real Data Applications

6.1 Tensor Denoising on 3D fMRI Data

We consider a 3D MRI dataset INCISIX from the OsiriX repository[3], which contains 166 slices through a human brain, each having dimension 512×512. The dataset was first analyzed in Gandy et al. (2011) from a tensor completion perspective by randomly setting voxels to be missing. We approach this dataset from a tensor denoising perspective. Following Goldfarb and Qin (2014), we extract the first 50 slices and downsample each of them to have size 128×128. Therefore the noiseless tensor $\boldsymbol{\mathcal{X}}$ has dimension $128 \times 128 \times 50$. We then corrupt 25% of the tensor entries with outliers/noise drawn from $\text{Unif}[0, 2\,\text{std}(\text{vec}(\boldsymbol{\mathcal{X}}))]$. In addition to HoRPCA-C and RGrad, we consider a classic low-tubal-rank robust tensor recovery method called tensor robust principal component analysis (Lu et al., 2019) as another baseline. Scree plots of the different matrix unfoldings revealed that the mode-3 singular values decay rapidly, which indicates that the data tensor is approximately low-rank along mode-3. Therefore, we considered the following three Tucker-rank tuples, (64,64,10), (96,96,15) and (128,128,20), as candidate Tucker-rank tuples for HoRPCA-C, RGrad and Tucker-L_2E. For RGrad, the proposed

[3]https://www.osirix-viewer.com

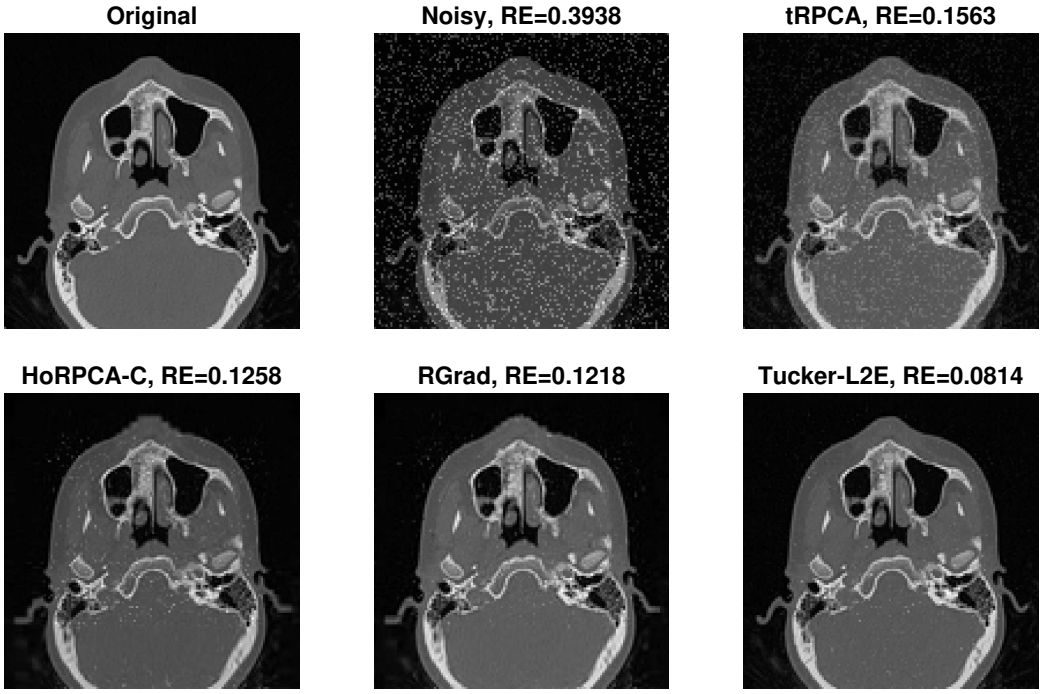


Figure 6: Recovery results for slice 30 of INCISIX dataset. The annotated relative errors are for the whole tensor instead of one slice.

BIC criterion selected the parameters $R = (96, 96, 15)$, $\alpha = 0.27$ and $\mu_0 = 1$. For HoRPCA-C and Tucker-L_2E, we use a hold-out validation approach to identify the best Tucker-rank tuple. We randomly sample 10% of the entries as the validation set and use the remaining 90% of entries to impute the missing 10% of entries. Then we can use the MAD between the imputed values and the actual values to determine the appropriate Tucker-rank. From Table 1, we see that the best performing Tucker-rank for HoRPCA-C is $(96, 96, 15)$, while for Tucker-L_2E it is $(128, 128, 20)$. We then reapply HoRPCA-C and Tucker-L_2E with the selected ranks to the fully observed tensor.

Tucker-rank	$(64, 64, 10)$	$(96, 96, 15)$	$(128, 128, 20)$
HoRPCA-C hold-out MAD	223.16	**216.85**	276.60
Tucker-L_2E hold-out MAD	236.51	211.12	**206.60**

Table 1: Hold-out MAD for HoRPCA-C and Tucker-L_2E at different Tucker-ranks.

We visualize the recovery results of tRPCA, HoRPCA-C, RGrad and Tucker-L_2E in Figure 6. The advantage of Tucker-L_2E over HoRPCA-C and RGrad is that at rank (128,128,20), HoRPCA-C and RGrad overfit to the sparse noise while Tucker-L_2E remains largely unaffected, in line with our observations in Section 5. By remaining stable at a larger rank, Tucker-L_2E is able to capture more structural information, resulting in the

smallest relative error and the best perceptual quality.

6.2 PARAFAC Analysis of Fluorescence Data

Parallel Factor analysis (PARAFAC) is a widely used tool in Chemometrics for decomposing fluorescence excitation-emission matrices (EEMs) into their underlying chemical components (Murphy et al., 2013). The CP model is particularly suitable for the analysis of EEMs since this type of data mostly conforms to a trilinear structure due to Beer's law, which states that absorbance is the product of molar concentration, molar absorption coefficient, and optical path length. Certain regions of the fluorescence landscape, however, may be corrupted by Raman and Rayleigh scattering. Therefore, EEM data is a natural candidate for the low CP-rank tensor plus sparse outliers model.

We consider a standard EEM dataset, the Dorrit data, originally introduced in Baunsgaard (1999). We use a preprocessed version[4] (Riu and Bro, 2003) of the Dorrit data, which consists of 27 mixed samples containing different concentrations of hydroquinone, tryptophan, phenylalanine, and dopa. Each sample has 121 emission wavelengths (241-481 nm) and 24 excitation wavelengths (200-315 nm). Following Riu and Bro (2003) and Goldfarb and Qin (2014), we exclude samples 2, 3, 5, and 10 as well as data corresponding to excitation wavelengths from 200 nm to 225 nm since this portion of the data is believed to be noisy for reasons other than scattering and amounts to slice-wise corruption, which greatly affect the global properties of the data. Therefore the tensor data to be analyzed have dimension $23 \times 121 \times 18$. The truncated fluorescence landscape of sample 1 is visualized in Figure 7. We set the CP-rank to 4 and the Tucker-rank to $(4, 4, 4)$ since we have prior knowledge that there are 4 pure compounds in the samples.

Figure 8 displays the recovered emission/excitation loadings (mode-2 and mode-3 CP factors) produced by CP-OPT, RGrad, HoRPCA-C, and Tucker-L_2E. The last three methods are in fact applied to denoise the tensors. The CP factors are then extracted by applying CP-OPT to the reconstructed tensors. We assign the CP factors to the 4 analytes based on proximity to the pure component emission/excitation profiles presented in Baunsgaard (1999). For this dataset, with rank fixed at $(4, 4, 4)$, BIC suggests $\alpha = 0.11$ and $\mu_0 = 5$ for RGrad. This has an interesting implication that there are approximately 11% of entries that are affected by scattering. The emission/excitation loadings produced by HoRPCA-C and Tucker-L_2E appear more similar to the pure component

[4]http://www.models.life.ku.dk/dorrit

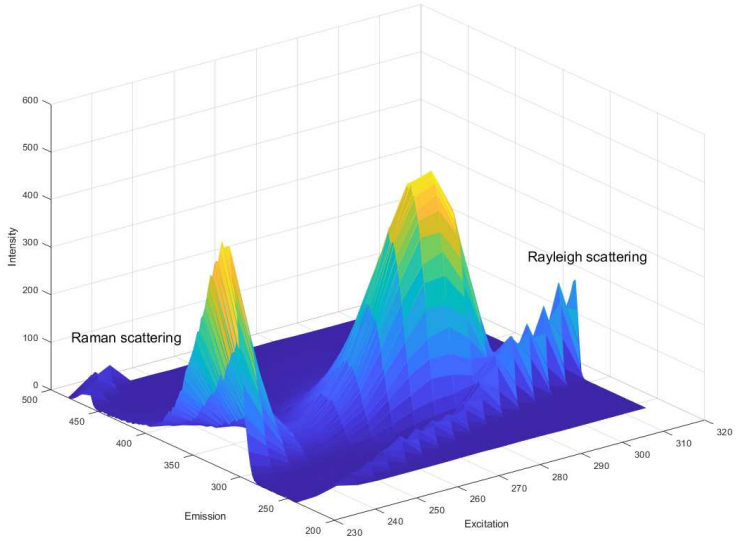


Figure 7: Truncated fluorescence landscape of sample 1 in the Dorrit data. Intensity peaks caused by Raman and Rayleigh scattering can be observed on the top left and the bottom right.

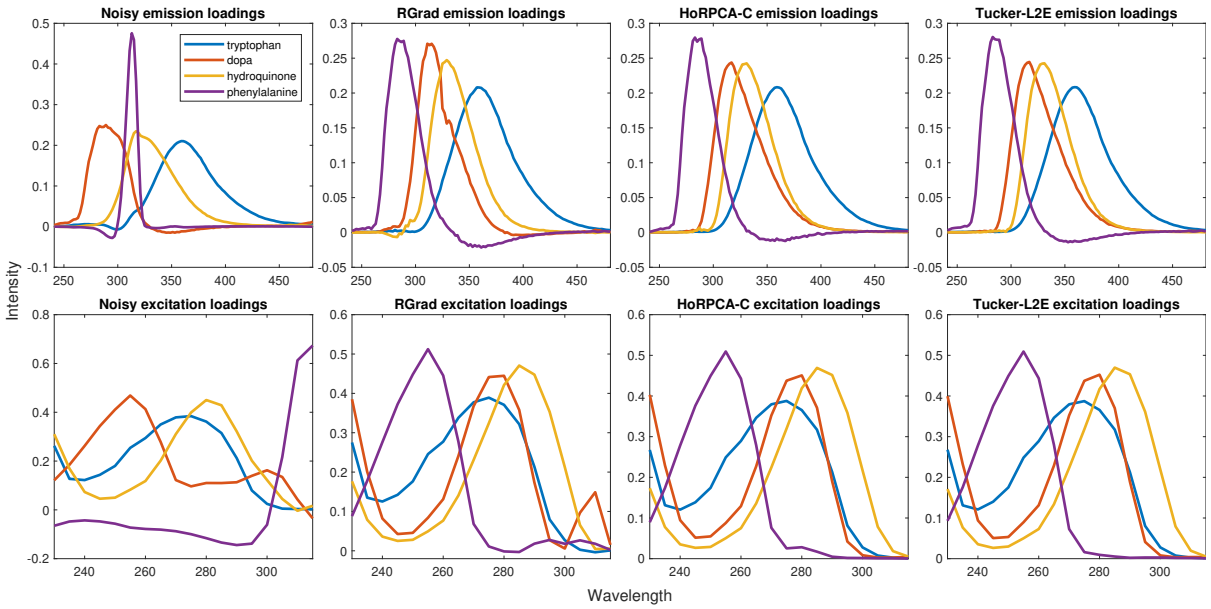


Figure 8: Reconstructed mode-2 and mode-3 CP factors for Dorrit data with scattering only.

profile than the ones produced by RGrad. Although the difference between HoRPCA-C and Tucker-L_2E is minuscule, the emission/excitation loadings of phenylalanine (purple lines in Figure 8) produced by Tucker-L_2E appear to be slightly more regular than those produced by HoRPCA-C.

6.3 Feature Extraction for Classification

Tucker decompositions are useful for extracting features from high-dimensional multi-way datasets for classification. The extracted features can then be used as input to standard classifiers such as *k-nearest-neighbors* (k-NN) or *support vector machines* (SVM). In this section we adopt the feature extraction framework based on Tucker decomposition originally presented in Phan and Cichocki (2010). Chachlakis et al. (2019) demonstrated that if the dataset is corrupted with sparse noise, strategic dimensionality reduction by a Tucker decomposition can reduce the impact of noise and leads to improved classification accuracy. Below we briefly describe the feature extraction framework. Suppose that we have K_1 tensor-valued training samples of size $I_1 \times I_2 \times \cdots \times I_N$, which can be classified into C categories. We concatenate training samples across mode $N+1$ to obtain $\mathcal{X}_1 \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_N \times K_1}$, which we call the "training tensor." Then $\mathcal{X}_1$ is Tucker decomposed with rank $(d_1, d_2, \ldots, d_N, K_1)$ to obtain the factor matrices $\mathbf{U}_n \in \mathbb{R}^{I_n \times d_n}$ for $n \in [N]$. We compress the training samples as follows:

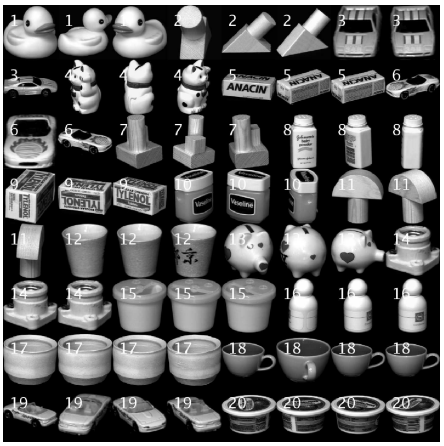
$$\mathcal{Z}_1 \;=\; \mathcal{X}_1 \times_1 \mathbf{U}_1^\mathsf{T} \times_2 \mathbf{U}_2^\mathsf{T} \times_3 \cdots \times_N \mathbf{U}_N^\mathsf{T} \;\in\; \mathbb{R}^{d_1 \times d_2 \times \cdots \times d_N \times K_1}.$$

Then $\mathcal{Z}_1$ is matricized in mode $N+1$ to become a data matrix $\mathbf{Z}_1$ of size $K_1 \times \prod_{n=1}^N d_n$ with labeled rows. We similarly concatenate testing samples to obtain the "testing tensor" $\mathcal{X}_2 \in \mathbb{R}^{d_1 \times d_2 \times \cdots \times d_N \times K_2}$, which we compress to obtain

$$\mathcal{Z}_2 \;=\; \mathcal{X}_2 \times_1 \mathbf{U}_1^\mathsf{T} \times_2 \mathbf{U}_2^\mathsf{T} \times_3 \cdots \times_N \mathbf{U}_N^\mathsf{T} \;\in\; \mathbb{R}^{d_1 \times d_2 \times \cdots \times d_N \times K_2},$$

which is then matricized in mode $N+1$ to become $\mathbf{Z}_2 \in \mathbb{R}^{K_2 \times \prod_{n=1}^N d_n}$. A suitable classifier is then trained on $\mathbf{Z}_1$ and tested on $\mathbf{Z}_2$. We apply the above feature extraction framework, with an added aspect of robustness, to image classification. We consider two classic image classification datasets, namely MNIST (Deng, 2012) and COIL-20 (Nene et al., 1996), which are also studied in Phan and Cichocki (2010). MNIST consists of 28×28 images of hand-written digits. COIL-20 consists of 128×128 images for 20 different objects with

(a) Randomly sampled images from the MNIST dataset.



(b) Randomly sampled images from the COIL-20 dataset.

Figure 9: Visualizations of the MNIST dataset and the COIL-20 dataset.

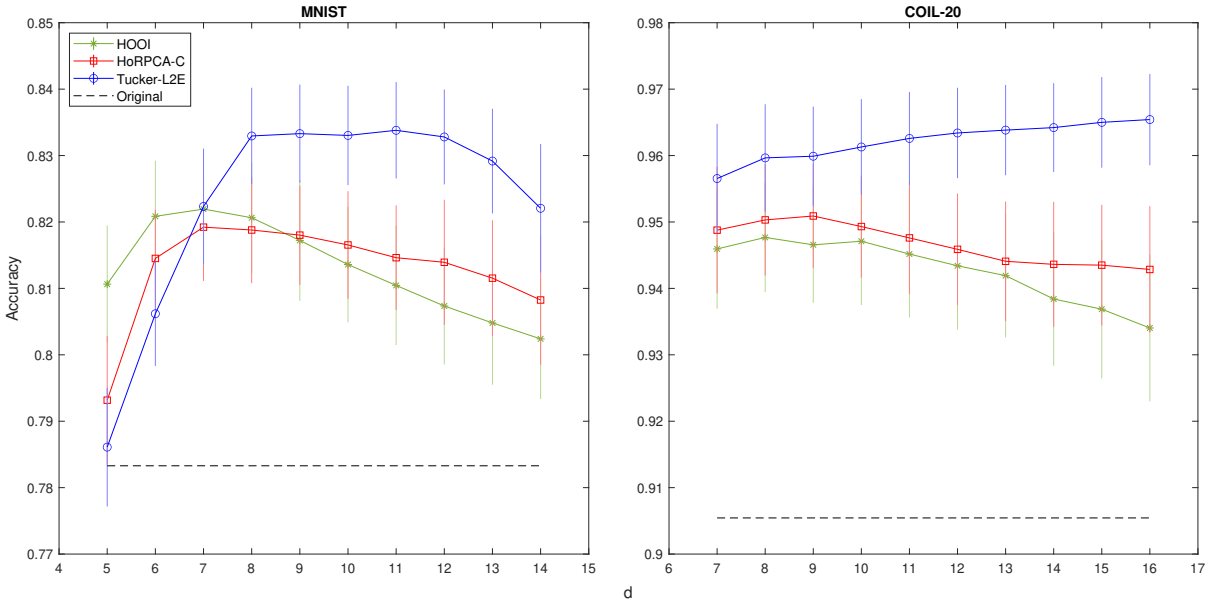


Figure 10: Average testing accuracy across 50 random realizations of training/testing sets. Error bars denote ±1 standard errors. Gray line shows the testing accuracy of *nearest-neighbor* without applying Tucker decomposition for dimensionality reduction.

each object having 72 images. Figure 9 depicts samples of images from both datasets.

As a preprocessing step, we remove the 4-pixel padding for MNIST since those regions contain no information. Images from COIL-20 are downsampled to 32×32 to speed up computation. To construct a training sample, we randomly sample 50 images for each digit from MNIST and 20 images for each object from COIL-20. The training images are then corrupted with salt-and-pepper noise added to a randomly selected sample of 25% of the pixels. Thus for MNIST, $\boldsymbol{\mathcal{X}}_1$ has dimension $20 \times 20 \times 500$ while for COIL-20, $\boldsymbol{\mathcal{X}}_1$ has dimension $32 \times 32 \times 400$. We set $d_1 = d_2 = d$ so that the total number of features is d^2. We randomly sample another 500 images for each digit and use the remaining 52 images for each object as the testing points. The classifier of choice is *nearest-neighbor*. We repeat the described procedure on 50 different random realizations of training and testing sets. The Tucker decomposition methods considered here are HOOI, HoRPCA-C and Tucker-L_2E. In particular, HoRPCA-C is only used to approximate the training tensor. The factor matrices are obtained by applying HOOI to the output tensor of HoRPCA-C.

Figure 10 highlights that by applying Tucker decomposition for feature extraction and dimensionality reduction, all methods achieve a substantial gain in accuracy compared with directly using the corrupted training images. As d increases, initially the accuracy of all Tucker decomposition methods will increase due to being able to create more meaningful features. However, eventually the feature extraction framework will overfit to the sparse noise and the accuracy starts decreasing. The testing accuracy on MNIST is generally much lower than on COIL-20 despite having fewer categories and more training images per category. This suggests that MNIST is a more challenging dataset to classify. Tucker-L_2E again exhibits greater stability in high-rank scenarios, especially in the case of COIL-20 with its accuracy steadily increasing throughout the range of d that we investigated. When d is small, Tucker-L_2E may not be advantageous. However, as the number of features increases, the best attainable accuracy of Tucker-L_2E outperforms that of HOOI and HoRPCA-C.

7 Conclusion

This paper describes a new formulation of the robust Tucker decomposition problem based on the L_2 criterion, Tucker-L_2E. We present two initialization strategies and a

solution algorithm based on L-BFGS-B. Numerical experiments and real data applications demonstrate that Tucker-L_2E exhibits stronger recovery capability in challenging high-rank scenarios compared with existing alternatives. This empirical property is useful since real-world tensors are often nearly low-rank instead of perfectly low-rank. By remaining stable at a higher rank, Tucker-L_2E is able to provide a more expressive reconstruction of the underlying low-rank tensor in the presence of sparse perturbations.

In this article we used an off-the-self local optimization algorithm L-BFGS-B as the main computational tool. A projected-gradient type algorithm will likely have a smaller memory footprint, which presents an interesting venue for future work. We also note that the proposed robust tensor recovery paradigm can be adapted to other formats of low-rank tensor recovery, for example the low-tubal-rank format and the tensor-train format, with suitably designed computation algorithms.

Supplementary Materials

Supplement: A pdf file that contains derivation of gradient, an alternative initialization strategy named spectral initialization with diagonal deletion, proof of Theorem 4.1, details of parameter choices, and a run time comparison with the baseline methods.

Software: Matlab code of the described method, along with scripts to reproduce some of the figures in Section 5 and Section 6.

Acknowledgement

We are grateful to the associate editor, editor, and anonymous referee for their valuable comments and suggestions, which greatly improved the presentation of this paper. We thank Haixu Ma and Xiaoqian Liu for their assistance in testing the software.

Funding

This research was partially funded by grants from the National Institute of General Medical Sciences (R01GM135928: EC, R01GM126550: YL) and the National Science Foundation (DMS-2201136: EC, DMS-2100729: YL).

Disclosure Statement

The authors report there are no competing interests to declare.

References

Acar, E., Dunlavy, D. M., and Kolda, T. G. (2011a), "A scalable optimization approach for fitting canonical tensor decompositions," *Journal of Chemometrics*, 25, 67–86.

Acar, E., Dunlavy, D. M., Kolda, T. G., and Mørup, M. (2011b), "Scalable tensor factorizations for incomplete data," *Chemometrics and Intelligent Laboratory Systems*, 106, 41–56.

Anandkumar, A., Jain, P., Shi, Y., and Niranjan, U. N. (2016), "Tensor vs. matrix methods: Robust tensor decomposition under block sparse perturbations," in *Artificial Intelligence and Statistics*, PMLR, pp. 268–276.

Bader, B. W. and Kolda, T. G. (2006), "Algorithm 862: MATLAB tensor classes for fast algorithm prototyping," *ACM Transactions on Mathematical Software (TOMS)*, 32, 635–653.

— (2008), "Efficient MATLAB computations with sparse and factored tensors," *SIAM Journal on Scientific Computing*, 30, 205–231.

Baunsgaard, D. (1999), "Factors affecting 3-way modelling (PARAFAC) of fluorescence landscapes." *Internal Report, Department of Dairy and Food Science, The Royal Veterinary and Agricultural University Denmark*.

Becker, S. (2015), "L-BFGS-B-C," GitHub, https://github.com/stephenbeckr/L-BFGS-B-C.

Boyd, S., Parikh, N., Chu, E., Peleato, B., Eckstein, J., et al. (2011), "Distributed optimization and statistical learning via the alternating direction method of multipliers," *Foundations and Trends® in Machine learning*, 3, 1–122.

Byrd, R. H., Lu, P., Nocedal, J., and Zhu, C. (1995), "A limited memory algorithm for bound constrained optimization," *SIAM Journal on Scientific Computing*, 16, 1190–1208.

Cai, C., Li, G., Poor, H. V., and Chen, Y. (2019), "Nonconvex low-rank tensor completion from noisy data," *Advances in Neural Information Processing Systems*, 32.

Cai, C., Poor, H. V., and Chen, Y. (2023), "Uncertainty quantification for nonconvex tensor completion: Confidence intervals, heteroscedasticity and optimality," *IEEE Transactions on Information Theory*, 69, 407–452.

Cai, J.-F., Li, J., and Xia, D. (2022), "Generalized low-rank plus sparse tensor estimation by fast Riemannian optimization," *Journal of the American Statistical Association*, to appear.

Candès, E. J., Li, X., Ma, Y., and Wright, J. (2011), "Robust principal component analysis?" *Journal of the ACM (JACM)*, 58, 1–37.

Carroll, J. D. and Chang, J.-J. (1970), "Analysis of individual differences in multidimensional scaling via an N-way generalization of "Eckart-Young" decomposition," *Psychometrika*, 35, 283–319.

Chachlakis, D. G., Prater-Bennette, A., and Markopoulos, P. P. (2019), "L1-norm Tucker tensor decomposition," *IEEE Access*, 7, 178454–178465.

Chi, E. C. and Scott, D. W. (2014), "Robust parametric classification and variable selection by a minimum distance criterion," *Journal of Computational and Graphical Statistics*, 23, 111–128.

Chi, J. T. and Chi, E. C. (2022), "A user-friendly computational framework for robust structured regression with the L2 criterion," *Journal of Computational and Graphical Statistics*, 31, 1051–1062.

De Lathauwer, L., De Moor, B., and Vandewalle, J. (2000a), "A multilinear singular value decomposition," *SIAM Journal on Matrix Analysis and Applications*, 21, 1253–1278.

— (2000b), "On the best rank-1 and rank-$(r_1, r_2, ..., r_n)$ approximation of higher-order tensors," *SIAM Journal on Matrix Analysis and Applications*, 21, 1324–1342.

Deng, L. (2012), "The MNIST database of handwritten digit images for machine learning research," *IEEE Signal Processing Magazine*, 29, 141–142.

Donoho, D. L. and Liu, R. C. (1988), "The "automatic" robustness of minimum distance functionals," *The Annals of Statistics*, 16, 552–586.

Filipović, M. and Jukić, A. (2015), "Tucker factorization with missing data with application to low-n-rank tensor completion," *Multidimensional Systems and Signal Processing*, 26, 677–692.

Gandy, S., Recht, B., and Yamada, I. (2011), "Tensor completion and low-n-rank tensor recovery via convex optimization," *Inverse Problems*, 27, 025010.

Goldfarb, D. and Qin, Z. (2014), "Robust low-rank tensor recovery: Models and algorithms," *SIAM Journal on Matrix Analysis and Applications*, 35, 225–253.

Gu, Q., Gui, H., and Han, J. (2014), "Robust tensor decomposition with gross corruption," *Advances in Neural Information Processing Systems*, 27.

Han, R., Willett, R., and Zhang, A. R. (2022), "An optimal statistical and computational framework for generalized tensor estimation," *The Annals of Statistics*, 50, 1–29.

Harshman, R. A. et al. (1970), "Foundations of the PARAFAC procedure: Models and conditions for an "explanatory" multimodal factor analysis," *UCLA Working Papers in Phonetics*, 16, 1–84.

Hjort, N. (1994), "Minimum L2 and robust Kullback-Leibler estimation," in *Proceedings of the 12th Prague Conference*, pp. 102–105.

Hong, D., Kolda, T. G., and Duersch, J. A. (2020), "Generalized canonical polyadic tensor decomposition," *SIAM Review*, 62, 133–163.

Kapteyn, A., Neudecker, H., and Wansbeek, T. (1986), "An approach to n-mode components analysis," *Psychometrika*, 51, 269–275.

Kilmer, M. E. and Martin, C. D. (2011), "Factorization strategies for third-order tensors," *Linear Algebra and its Applications*, 435, 641–658.

Kolda, T. G. (2006), "Multilinear operators for higher-order decompositions," Tech. Rep. SAND2006-2081, Sandia National Laboratories, Albuquerque, NM, Livermore, CA.

Kolda, T. G. and Bader, B. W. (2009), "Tensor decompositions and applications," *SIAM Review*, 51, 455–500.

Kroonenberg, P. M. and De Leeuw, J. (1980), "Principal component analysis of three-mode data by means of alternating least squares algorithms," *Psychometrika*, 45, 69–97.

Lane, J. W. (2012), "Robust quantile regression using L2E," Ph.D. thesis.

Lee, J. (2010), "L2E estimation for finite mixture of regression models with applications and L2E with penalty and non-normal mixtures," Ph.D. thesis.

Liu, D. C. and Nocedal, J. (1989), "On the limited memory BFGS method for large scale optimization," *Mathematical Programming*, 45, 503–528.

Liu, X., Chi, E. C., and Lange, K. (2023), "A Sharper Computational Tool for L_2E Regression," *Technometrics*, 65, 117–126.

Lozano, A. C., Meinshausen, N., and Yang, E. (2016), "Minimum distance lasso for robust high-dimensional regression," *Electronic Journal of Statistics*, 10, 1296–1340.

Lu, C., Feng, J., Chen, Y., Liu, W., Lin, Z., and Yan, S. (2019), "Tensor robust principal component analysis with a new tensor nuclear norm," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 42, 925–938.

Ma, J., Qiu, W., Zhao, J., Ma, Y., Yuille, A. L., and Tu, Z. (2015), "Robust L_2E estimation of transformation for non-rigid registration," *IEEE Transactions on Signal Processing*, 63, 1115–1129.

Ma, J., Zhao, J., Tian, J., Tu, Z., and Yuille, A. L. (2013), "Robust estimation of nonrigid transformation for point set registration," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 2147–2154.

Murphy, K. R., Stedmon, C. A., Graeber, D., and Bro, R. (2013), "Fluorescence spectroscopy and multi-way techniques. PARAFAC," *Analytical Methods*, 5, 6557–6566.

Nene, S. A., Nayar, S. K., Murase, H., et al. (1996), "Columbia object image library (coil-20)," .

Oseledets, I. V. (2011), "Tensor-train decomposition," *SIAM Journal on Scientific Computing*, 33, 2295–2317.

Phan, A. H. and Cichocki, A. (2010), "Tensor decompositions for feature extraction and classification of high dimensional datasets," *Nonlinear Theory and its Applications, IEICE*, 1, 37–68.

Ramos, J. J. (2014), "Robust methods for forecast aggregation," Ph.D. thesis.

Riu, J. and Bro, R. (2003), "Jack-knife technique for outlier detection and estimation of standard errors in PARAFAC models," *Chemometrics and Intelligent Laboratory Systems*, 65, 35–49.

Scott, A. I. (2006), "Denoising by wavelet thresholding using multivariate minimum distance partial density estimation," Ph.D. thesis.

Scott, D. W. (2001), "Parametric statistical modeling by minimum integrated square error," *Technometrics*, 43, 274–285.

— (2009), "The L2E method," *Wiley Interdisciplinary Reviews: Computational Statistics*, 1, 45–51.

Terrell, G. R. (1990), *Linear density estimates*, Department of Statistics, Virginia Polytechnic Institute and State University.

Tomioka, R., Suzuki, T., Hayashi, K., and Kashima, H. (2011), "Statistical performance of convex tensor decomposition," *Advances in Neural Information Processing Systems*, 24, 972–980.

Tong, T., Ma, C., Prater-Bennette, A., Tripp, E., and Chi, Y. (2022), "Scaling and scalability: Provable nonconvex low-rank tensor estimation from incomplete measurements," *Journal of Machine Learning Research*, 23, 1–77.

Tucker, L. R. (1966), "Some mathematical notes on three-mode factor analysis," *Psychometrika*, 31, 279–311.

Wu, Y., Tan, H., Li, Y., Li, F., and He, H. (2017), "Robust tensor decomposition based on Cauchy distribution and its applications," *Neurocomputing*, 223, 107–117.

Xia, D. and Yuan, M. (2019), "On polynomial time methods for exact low-rank tensor completion," *Foundations of Computational Mathematics*, 19, 1265–1313.

Xia, D., Yuan, M., and Zhang, C.-H. (2021), "Statistically optimal and computationally efficient low rank tensor completion from noisy entries," *The Annals of Statistics*, 49, 76–99.

Yang, J. and Scott, D. W. (2013), "Robust fitting of a Weibull model with optional censoring," *Computational Statistics & Data Analysis*, 67, 149–161.

Yang, K., Pan, A., Yang, Y., Zhang, S., Ong, S. H., and Tang, H. (2017), "Remote sensing image registration using multiple image features," *Remote Sensing*, 9, 581.

Yang, Y., Feng, Y., and Suykens, J. A. (2015), "Robust low-rank tensor recovery with regularized redescending M-estimator," *IEEE Transactions on Neural Networks and Learning Systems*, 27, 1933–1946.

Zhang, A. (2019), "Cross: Efficient low-rank tensor completion," *The Annals of Statistics*, 47, 936–964.

Zhang, A. and Xia, D. (2018), "Tensor SVD: Statistical and computational limits," *IEEE Transactions on Information Theory*, 64, 7311–7338.

Zhao, Q., Zhou, G., Xie, S., Zhang, L., and Cichocki, A. (2016), "Tensor ring decomposition," *arXiv preprint arXiv:1606.05535*.

Zhao, Q., Zhou, G., Zhang, L., Cichocki, A., and Amari, S.-I. (2015), "Bayesian robust tensor factorization for incomplete multiway data," *IEEE transactions on neural networks and learning systems*, 27, 736–748.

Zhu, C., Byrd, R. H., Lu, P., and Nocedal, J. (1997), "Algorithm 778: L-BFGS-B: Fortran subroutines for large-scale bound-constrained optimization," *ACM Transactions on mathematical software (TOMS)*, 23, 550–560.