SIAM J. IMAGING SCIENCES
Vol. 11, No. 4, pp. 2629–2664

# Fast Randomized Algorithms for t-Product Based Tensor Operations and Decompositions with Applications to Imaging Data[*]

Davoud Ataee Tarzanagh[†] and George Michailidis[‡]

**Abstract.** Tensors of order three or higher have found applications in diverse fields, including image and signal processing, data mining, biomedical engineering, and link analysis, to name a few. In many applications that involve, for example, time series or other ordered data, the corresponding tensor has a distinguishing orientation that exhibits a low tubal structure. This has motivated the introduction of the tubal rank and the corresponding tubal singular value decomposition in the literature. In this work, we develop randomized algorithms for many common tensor operations, including tensor low-rank approximation and decomposition, together with tensor multiplication. The proposed tubal focused algorithms employ a small number of lateral and/or horizontal slices of the underlying third order tensor that come with *relative error guarantees* for the quality of the obtained solutions. The performance of the proposed algorithms is illustrated on diverse imaging applications, including mass spectrometry data and image and video recovery from incomplete and noisy data. The results show both good computational speed-up vis-a-vis conventional completion algorithms and good accuracy.

**Key words.** tensor slices selection, circulant algebra, low rank decomposition, nuclear norm minimization

**AMS subject classifications.** 15A69, 15A23, 68W20, 65F30

**DOI.** 10.1137/17M1159932

## 1. Introduction.

Tensors are multidimensional arrays that have been used in diverse fields of applications, including chemometrics [45], psychometrics [30], image/video and signal processing [24, 25, 35, 44, 54], and link analysis [29]. They have also been the object of intense mathematical study (see, for example, the review paper by Kolda and Bader [28] and references therein).

Analogously to the matrix case, a number of tensor decompositions have been proposed in the literature, briefly described next for the case of 3-way tensors. Consider a 3-way tensor $\mathfrak{Z} \in \mathbb{R}^{n_1 \times n_2 \times n_3}$. The equation

$$\mathfrak{Z} = \sum_{r_1=1}^{R_1} \sum_{r_2=1}^{R_2} \sum_{r_3=1}^{R_3} g_{r_1 r_2 r_3} \left( u_{r_1}^{(1)} \circ u_{r_2}^{(2)} \circ u_{r_3}^{(3)} \right) = \mathfrak{G} \times_1 U^{(1)} \times_2 U^{(2)} \times_3 U^{(3)} \tag{1.1}$$

provides the Tucker decomposition [22] of $\mathfrak{Z}$, with $\mathfrak{G} \in \mathbb{R}^{R_1 \times R_2 \times R_3}$ being the core tensor associated with this decomposition, and where $\times_i$ is the mode-$i$ (matrix) product. For a core

[*]Received by the editors December 5, 2017; accepted for publication (in revised form) August 13, 2018; published electronically November 13, 2018.
http://www.siam.org/journals/siims/11-4/M115993.html
**Funding:** The work of the second author was supported in part by NSF grants CCF 1540093 and IIS 1632730 and NIH grant 5 R01 GM114029-03.

[†]Department of Mathematics & UF Informatics Institute, University of Florida, Gainesville, FL 32611 (tarzanagh@ufl.edu).

[‡]Department of Statistics & UF Informatics Institute, University of Florida, Gainesville, FL 32611 (gmichail@ufl.edu).

tensor of minimal size, $R_1$ is the column rank (the dimension of the subspace spanned by mode-1 fibers), $R_2$ is the row rank (the dimension of the subspace spanned by mode-2 fibers), and so on. A key difference from the matrix case is that the values of $R_1, R_2, R_3$ can be different. The 3-tuple $(R_1, R_2, R_3)$ is consequently called the Tucker rank or *multilinear rank* of tensor $\mathfrak{Z}$ [8, 22]. The conventional Tucker decomposition corresponds to the orthonormal Tucker decomposition, which is also known as the higher-order singular value decomposition (SVD). De Lathauwer, De Moor, and Vandewalle [10] proposed an algorithm to compute a higher-order SVD decomposition. Soon afterwards they proposed the higher-order orthogonal iteration [11] to provide an inexact Tucker decomposition. The CANDECOMP/PARAFAC (CP) decomposition of a tensor is another important notion of tensor-decomposition, which leads to the definition of CP rank. The CP model can be considered as a special case of the Tucker model with a superdiagonal core tensor. Further, the CP rank of a tensor equals that of its Tucker core [21].

## 1.1. Third order tensor as operator on matrices.

While the Tucker-based factorization may be sufficient for many applications, in this paper we consider an entirely different tensor decomposition based on circulant algebra [26]. In this factorization, a tensor in $\mathbb{R}^{n_1 \times n_2 \times n_3}$ is viewed as a $n_1 \times n_2$ matrix of "tubes," also known as elements of the ring $\mathbb{R}^{n_3}$ where addition is defined as vector addition and multiplication as circular convolution. This "matrix-of-tubes" viewpoint leads to definitions of a new multiplication for tensors ("tubal multiplication"), a new rank for tensors ("tubal rank"), and a new notion of a SVD for tensors ("tubal SVD"). The tubal SVD (t-SVD) provides the "best" tubal rank-$r$ approximation to $\mathfrak{Z}$, as measured with respect to any unitary invariant tensor norm.

A limitation of the t-SVD decomposition is that it depends directly on the orientation of the tensor, whereas the CP and Tucker decompositions do not. This suggests that the latter decompositions are well suited for data applications where the tensor's orientation is not critical, e.g., chemometrics [45] and/or psychometrics [30]. However, in applications involving time series or other ordered data, the orientation of the tensor is fixed. Examples include, but not limited to, computed tomography (CT) [44], facial recognition [18], and video compression [54], where the tensor decomposition is dependent on the third dimension. Analogous to compressing a two-dimensional image using the matrix SVD (a classic linear algebra example, with detailed writeup in [23]), the t-SVD decomposition can be used to compress several images taken over time (e.g. successive frames from a video). Since such images do not change substantially from frame to frame, we expect tubal compression strategies to provide better results than performing a matrix SVD on each separate image frame. The former consider the tensor as a whole object, rather than as a collection of distinct images [27, 26, 54, 44, 35]. Further, t-SVD is essentially based on a group theoretic approach, where the multidimensional structure is unraveled by constructing group rings along the tensor fibers.[1] The advantage of such an approach over existing ones is that the resulting algebra and corresponding analysis enjoys many similar properties to matrix algebra and analysis. For example, it is shown in [20] that recovering a 3-way tensor of length $n$ and Tucker rank $(r, r, r)$ from random measurements requires $O(rn^2 \log^2(n))$ observations under a matrix coherence condition on all mode-$n$ unfoldings. However, the number of samples needed for exact recovery of a 3-way

[1]We consider the group rings constructed out of cyclic groups, resulting in an algebra of circulants. However, the results presented in this paper hold true for the general group-ring construction.

tensor of length $n$ and tensor tubal-rank $r$ is $O(rn^2 \log(n^2))$ under a weaker tensor coherence assumption [53]. Further, consider the decomposition $\mathcal{X} = \mathcal{L} + \mathcal{E}$, where $\mathcal{L} \in \mathbb{R}^{n_1 \times n_2 \times n_3}$ is low rank and $\mathcal{E}$ is sparse. Let $n_{(1)} = \max(n_1, n_2)$ and $n_{(2)} = \min(n_1, n_2)$. The work in [35] shows that for tensor $\mathcal{L}$ with coherence parameter $\mu$, the recovery is guaranteed with high probability for the tubal rank of order $n_{(2)}n_3/(\mu(\log n_{(1)}n_3)^2)$ and a number of nonzero entries in $\mathcal{E}$ of order $O(n_1n_2n_3)$. Hence, under the same coherence condition (see Definitions 2.14 and 2.15), the tubal robust tensor factorization problem perfectly recovers the low-rank and sparse components of the underlying tensor.

A shortcoming of these three classical decompositions is their brittleness with respect to severely corrupted or outlier data entries. To that end, a number of approaches have been developed in the literature to recover a low-rank tensor representation from data subject to noise and corrupted entries. We focus on two instances of the problem based on the t-SVD algorithm: (i) noisy tensor completion, i.e., recovering a low-rank tensor from a small subset of noisy entries, and (ii) noisy robust tensor factorization, i.e., recovering a low-rank tensor from corrupted data by noise and/or outliers of arbitrary magnitude [35, 54]. These two classes of tensor factorization problems have attracted significant interest in the research community [2, 6, 44, 54]. In particular, convex formulations of noisy tensor factorization have been shown to exhibit strong theoretical recovery guarantees and a number of algorithms has been developed for solving them [54, 44, 35].

It is frequently mentioned that (noisy) tensor factorization, despite its numerous advantages, also exhibits a number of drawbacks listed below:

- The available methods [4, 27, 26, 54, 44, 35] are inherently sequential and all rely on the repeated and costly computation of t-SVD factors, discrete Fourier transform (DFT) and its inverse, that limit their scalability.
- The basis tensor vectors resulting from t-SVD have little concrete meaning, which makes it challenging for practitioners to interpret the obtained results. For instance, the vector $[(1/2)$ age $- (1/\sqrt{2})$ height $+ (1/2)$income$]$, being one of the significant uncorrelated factors from a data set of people's features is not easily interpretable (see discussion in [15]). Kuruvilla, Park, and Schreiber in [31] have also claimed that "it would be interesting to try to find basis vectors for all experiment vectors, using actual experiment vectors and not artificial bases that offer little insight."
- The t-SVD decomposition for sparse tensors does not preserve sparsity in general, which for large size tensors leads to excessive computations and storage requirements. Hence, it is important to compute low-rank tensor factorizations that preserve such structural properties of the original data tensor.

**1.2. Main contributions.** In this paper, we study scalable *randomized* tensor multiplication (rt-product algorithm) and tensor factorization (rt-project) operations and extend the matrix CX and CUR type decompositions [12, 13, 15, 36] to third order tensors using a circulant algebra embedding. To that end, we develop a basic algorithm (t-CX), together with a more general one (t-CUR) based on tensor slice selection that come with relative error guarantees. Finally, we propose a new tensor nuclear norm minimization method, called CUR t-NN, which solves the noisy tensor factorization problem using a small number of lateral and/or horizontal slices of the underlying tensor. Specifically, CUR t-NN uses an adaptive technique to sample slices of the tensor based on a *leverage score* for them and subsequently solves a convex optimization problem followed by a projection step to recover a low rank approximation to the underlying tensor. Advantages of CUR t-NN include the following:

- Contrary to nuclear norm minimization based approaches, which minimize the sum of the singular values of the underling tensor, our approach only minimizes the sum of singular values of the set of sampled slices corresponding to the largest leverage scores. Thus, we obtain a more accurate and robust approximation to the rank function.
- Using subspace sampling, we obtain provable relative-error recovery guarantees for tubal product based tensor factorization. This technique is likely to be useful for other tensor approximation and data analysis problems.
- The proposed algorithm for noisy tensor factorization tasks has polynomial time complexity.

### 1.3. Related work on randomized tensor factorization.

Note that there has been prior work on the topic of randomized methods for tensor low rank approximations/decompositions. For example, recent work includes [37, 5]. Our methods are different from these studies, since we rely on the *t-product* construct in [4, 26, 27] to provide *tubal* tensor multiplication and low rank approximation. In [37], the proposed tensor-CUR algorithm employs a CUR matrix approximation to one of the unfolding matrix modes (the distinguished mode) providing an approximation based on few tube fibers and few slices. Hence, that algorithm achieves an additive error guarantee in terms of the flattened (unfolded) tensor, rather than the original one. However, the algorithms presented in this work offer *relative error guarantees* for the obtained approximations, in terms of the original low tubal rank tensor. Further, we propose a new tensor nuclear norm minimization method, which solves the noisy tensor factorization problem in the fully and partially observed setting using a small number of lateral and/or horizontal slices of the underlying tensor. It is worth mentioning that randomized algorithms were used to efficiently solve the robust matrix principal component analysis (PCA) problem using small sketches constructed from random linear measurements of the low rank matrix [43, 55, 34, 36, 39]. Our proposed nuclear norm minimization approach is different from these studies, since we relay on slice selection and projection (mainly t-CUR factorization). Further, our proposed algorithm uses an adaptive sampling strategy and provides high probability recovery guarantees of the exact low rank tensor approximation under a weaker coherence assumption.

The remainder of the paper is organized as follows. In section 2, we review some relevant mathematical concepts including the tensor circulant algebra, basic definitions and theorems of tensors, and the t-SVD decomposition based on the t-product concept. In section 3, we provide the randomized tensor decompositions and provide their relative error guarantees and introduce the concept of slice selection and projection. In section 4, we introduce, evaluate and analyze our proposed algorithm (CUR t-NN) for large scale noisy tensor decomposition. Experimental results are presented in section 5 and some concluding remarks are drawn in section 6.

The detailed proofs of the main results established are delegated to the Appendix.

## 2. Mathematical preliminaries.

Next, we introduce key definitions and concepts used in subsequent developments.

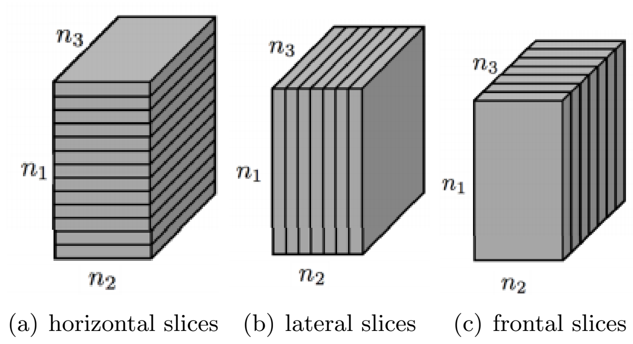


**Figure 2.1.** *Slices of an* $n_1 \times n_2 \times n_3$ *tensor* $\mathcal{X}$.

**Tensor indexing.** We denote tensors by Euler script letters, e.g., $\mathcal{X}$; matrices by capital letters, e.g., $X$; vectors by lowercase letters, e.g., $x$. The order of a tensor is the number of dimensions (also refereed to as ways or modes). In this work, we focus on 3-way tensors.

**Fibers and slices** [51]**.** A *fiber* of tensor $\mathcal{X}$ is a one-dimensional array defined by fixing two of the indices. Specifically, $\mathcal{X}_{:jk}$ is the (j,k)th *column* fiber, $\mathcal{X}_{i:k}$ is the (i,k)th *row* fiber, and $\mathcal{X}_{ij:}$ is the (i,j)th *tube* fiber. A slice of tensor $\mathcal{X}$ is a two-dimensional array defined by fixing one index only. Specifically, $\mathcal{X}_{i::}$ is the ith *horizontal* slice, $\mathcal{X}_{:j:}$ is the jth *lateral* slice, and $\mathcal{X}_{::k}$ is the kth *frontal* slice (see Figure 2.1). The vectorization of $\mathcal{X}$ is denoted by $vec(\mathcal{X})$. For a 3-way tensor $\mathcal{X} \in \mathbb{R}^{n_1 \times n_2 \times n_3}$, we denote its $(i, j, k)$th entry as $\mathcal{X}_{ijk}$.

**Norms.** We denote the $\ell_1$ norm as $\|\mathcal{X}\|_1 := \sum_{ijk} |x_{ijk}|$, the infinity norm as $\|\mathcal{X}\|_\infty := \max_{ijk} |x_{ijk}|$, and the Frobenius norm as $\|\mathcal{X}\|_F := \sqrt{\sum_{ijk} |x_{ijk}|^2}$. The above norms reduce to the vector or matrix norms if $\mathcal{X}$ is a vector or a matrix.

**Operators.** For $\mathcal{X} \in \mathbb{R}^{n_1 \times n_2 \times n_3}$ and using the Matlab commands `fft,ifft`, we denote by $\hat{\mathcal{X}}$ the result of applying the DFT on $\mathcal{X}$ along the third dimension, i.e., $\hat{\mathcal{X}} = \texttt{fft}(\mathcal{X}, [], 3)$. Analogously, one can also compute $\mathcal{X}$ from $\hat{\mathcal{X}}$ via the inverse DFT, using $\texttt{ifft}(\hat{\mathcal{X}}, [], 3)$. In particular, we denote by $\hat{X}$ the block diagonal matrix with each block corresponding to the frontal slice $\hat{X}_{::k}$ of $\hat{\mathcal{X}}$. That is,

$$\hat{X} = \mathrm{bdiag}_{k \in [n_3]}(\hat{\mathcal{X}}_{::k}) := \begin{pmatrix} \hat{\mathcal{X}}_{::1} & & & \\ & \hat{\mathcal{X}}_{::2} & & \\ & & \ddots & \\ & & & \hat{\mathcal{X}}_{::n_3} \end{pmatrix}.$$

## 2.1. Tensor basics.

Next, we review relevant mathematical concepts including the tubal SVD (t-SVD), basic definitions and operations, and other technical results of tensors [27, 44, 53] that are used throughout the paper.

Definition 2.1 (tensor product). *Given two tensors $\mathfrak{Z} \in \mathbb{R}^{n_1\times n_2\times n_3}$ and $\mathfrak{X} \in \mathbb{R}^{n_2\times n_4\times n_3}$, the t-product $\mathfrak{Z} * \mathfrak{X}$ is the $n_1 \times n_4 \times n_3$ tensor,*

$$\mathfrak{C} = \mathfrak{Z} * \mathfrak{X} = \text{fold}\left(\text{circ}\left(\mathfrak{Z}\right)\right).\text{unfold}\left(\mathfrak{X}\right), \tag{2.1}$$

*where*

$$\text{circ}\left(\mathfrak{Z}\right) := \begin{pmatrix} \mathfrak{Z}_{::1} & \mathfrak{Z}_{::n_3} & \dots & \mathfrak{Z}_{::2} \\ \mathfrak{Z}_{::2} & \mathfrak{Z}_{::1} & \dots & \mathfrak{Z}_{::3} \\ \vdots & \vdots & \ddots & \vdots \\ \mathfrak{Z}_{::n_3} & \mathfrak{Z}_{::n_3-1} & \dots & \mathfrak{Z}_{::1} \end{pmatrix}$$

*and*

$$\text{unfold}\left(\mathfrak{X}\right) := \begin{pmatrix} X_{::1} \\ X_{::2} \\ \vdots \\ X_{::n_3} \end{pmatrix}, \qquad \text{fold}\left(\text{unfold}\left(\mathfrak{X}\right)\right) = \mathfrak{X}.$$

Because the circular convolution of two tube fibers can be computed by the DFT, the t-product can be alternatively computed in the Fourier domain, as shown in Algorithm 2.1.

**Algorithm 2.1.** t-product $\mathfrak{C} = \mathfrak{Z} * \mathfrak{X}$ in the Fourier domain.

```
1: Input: $\mathfrak{Z} \in \mathbb{R}^{n_1\times n_2\times n_3}$; $\mathfrak{X} \in \mathbb{R}^{n_2\times n_4\times n_3}$
2: $\hat{\mathfrak{Z}} \leftarrow$ fft($\mathfrak{Z}$, [], 3);
3: $\hat{\mathfrak{X}} \leftarrow$ fft($\mathfrak{X}$, [], 3);
4: for $k = 1, \dots, n_3$ do
5:     $\hat{\mathfrak{C}}_{::k} = \hat{\mathfrak{Z}}_{::k}\hat{\mathfrak{X}}_{::k}$;
6: end for
7: $\mathfrak{C} \leftarrow$ ifft($\hat{\mathfrak{C}}$, [], 3);
8: return $\mathfrak{C}$
```

Definition 2.2 (conjugate transpose). *The conjugate transpose of a tensor $\mathfrak{X} \in \mathbb{R}^{n_1\times n_2\times n_3}$ is tensor $\mathfrak{X}^* \in \mathbb{R}^{n_2\times n_1\times n_3}$ obtained by conjugate transposing each of the frontal slices and then reversing the order of transposed frontal slices 2 through $n_3$.*

Definition 2.3 (identity tensor). *The identity tensor $\mathfrak{I} \in \mathbb{R}^{n\times n\times n_3}$ is the tensor whose first frontal slice is the $n \times n$ identity matrix, and whose other frontal slices are all zeros.*

Definition 2.4 (orthogonal tensor). *A tensor $\mathfrak{Q} \in \mathbb{R}^{n\times n\times n_3}$ is orthogonal if it satisfies $\mathfrak{Q}^* * \mathfrak{Q} = \mathfrak{Q} * \mathfrak{Q}^* = \mathfrak{I}$.*

Definition 2.5 (f-diagonal Tensor). *A tensor is called $f-$diagonal if each of its frontal slices is a diagonal matrix.*

Theorem 2.6 (t-SVD). *Let $\mathfrak{X} \in \mathbb{R}^{n_1\times n_2\times n_3}$. Then, it can be factored as*

$$\mathfrak{X} = \mathfrak{U} * \Sigma * \mathfrak{V}^T, \tag{2.2}$$

*where $\mathfrak{U} \in \mathbb{R}^{n_1\times n_1\times n_3}$, $\mathfrak{V} \in \mathbb{R}^{n_2\times n_2\times n_3}$ are orthogonal and $\Sigma \in \mathbb{R}^{n_1\times n_2\times n_3}$ is a f-diagonal tensor.*

Note that t-SVD can be efficiently computed based on the matrix SVD in the Fourier domain. This is based on the key property that the block circulant matrix can be mapped to a block diagonal matrix in the Fourier domain, i.e.,

$$(F_{n_3} \otimes I_{n_1}) \cdot \operatorname{circ}(\mathcal{X}) \cdot (F_{n_3}^{-1} \otimes I_{n_2}) = \hat{X}, \tag{2.3}$$

where $F_{n_3}$ denotes the $n_3 \times n_3$ DFT matrix and $\otimes$ denotes the Kronecker product.

Definition 2.7 (tensor multi and tubal rank). *The tensor multirank of $\mathcal{X} \in \mathbb{R}^{n_1 \times n_2 \times n_3}$ is a vector $v \in \mathbb{R}^{n_3}$ with its ith entry being the rank of the ith frontal slice of $\hat{\mathcal{X}}$, $r_i = \operatorname{rank}(\hat{\mathcal{X}}_{::i})$. The tensor tubal rank, denoted by $r = \operatorname{rank}_t(\mathcal{X})$, is defined as the number of nonzero singular tubes of $\Sigma$, where $\Sigma$ is obtained from the t-SVD of $\mathcal{X} = \mathcal{U} * \Sigma * \mathcal{V}^T$. That is,*

$$r = card\{i : \Sigma_{ii:} \neq 0\} = \max_i r_i, \tag{2.4}$$

*where card denotes the cardinality of a set.*

The tensor tubal rank shares some properties of the matrix rank, e.g.,

$$\operatorname{rank}_t(\mathcal{X} * \mathcal{Z}) \leq \min(\operatorname{rank}_t(\mathcal{X}), \operatorname{rank}_t(\mathcal{Z})).$$

Many tensor completion and decomposition techniques for video and seismic noise reduction rely on a low-rank factorization of a time-frequency transform. Further, certain energy methods broadly used in image processing, e.g., PDEs [1] and belief propagation techniques mainly focus on local relationships. The basic assumption is that the missing entries depend primarily on their neighbors. Hence, the further apart two pixels are, the smaller their dependance is. However, for video and time series of images the value of the missing entry also depends on entries which are relatively far away in the time/sequence dimension. Thus, it is necessary to develop a tool to directly capture such global information in the data. Using (2.4), $r_1$ is the rank of the "mean image" across the video sequence. Meanwhile, $r_2$ is the rank of the next frequency's content across frames, etc. Under a smoothness assumption that captures global information at given pixels across time, the frontal slices (after FFT) for bigger $i$ have smaller singular values.

Definition 2.8 (tensor nuclear norm). *The tensor nuclear norm of a tensor $\mathcal{X} \in \mathbb{R}^{n_1 \times n_2 \times n_3}$, denoted by $\|\mathcal{X}\|_{\circledast}$, is defined as the average of the nuclear norm of all frontal slices of $\hat{\mathcal{X}}$, i.e.,*

$$\|\mathcal{X}\|_{\circledast} := \frac{1}{n_3} \sum_{k=1}^{n_3} \|\hat{\mathcal{X}}_{::k}\|_*. \tag{2.5}$$

The above tensor nuclear norm is defined in the Fourier domain. It is closely related to the nuclear norm of the block circulant matrix in the original domain. Indeed,

$$\begin{aligned}\|\mathcal{X}\|_{\circledast} &= \frac{1}{n_3} \sum_{k=1}^{n_3} \|\hat{\mathcal{X}}_{::k}\|_* = \frac{1}{n_3}\|\hat{X}\|_* \\ &= \frac{1}{n_3}\|(F_{n_3} \otimes I_{n_1}) \cdot \operatorname{circ}(\mathcal{X}) \cdot (F_{n_3}^{-1} \otimes I_{n_2})\|_* \\ &= \frac{1}{n_3}\|\operatorname{circ}(\mathcal{X})\|_*.\end{aligned}$$

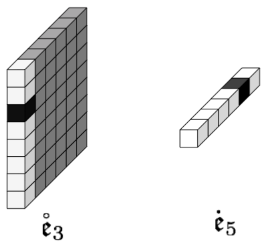


Figure 2.2. *The lateral basis $\mathring{\mathfrak{e}}_3$ and the tube basis $\dot{\mathfrak{e}}_5$. The black cubes are* 1*, gray and white cubes are* 0*. The white cubes stand for the potential entries that could be* 1*.*

The above relationship gives an equivalent definition of the tensor nuclear norm in the original domain. Thus, the tensor nuclear norm is the nuclear norm (with a factor $1/n_3$) of a new matricization (block circulant matrix) of a tensor.

Definition 2.9 (tensor spectral norm). *The tensor spectral norm of $\mathfrak{X} \in \mathbb{R}^{n\times n\times n_3}$, denoted as $\|\mathfrak{X}\|$, is defined as*

$$\|\mathfrak{X}\| := \|\hat{X}\|_2 = \|(F_{n_3} \otimes I_{n_1}) \cdot \operatorname{circ}(\mathfrak{X}) \cdot (F_{n_3}^{-1} \otimes I_{n_2})\|_2,$$

*where $\|\cdot\|_2$ denotes the spectral norm of a matrix.*

Definition 2.10 (inverse of tensor). *The inverse of $\mathfrak{X} \in \mathbb{R}^{n\times n\times n_3}$, denoted by $\mathfrak{X}^{-1}$, satisfies*

$$\mathfrak{X}^{-1} * \mathfrak{X} = \mathfrak{X} * \mathfrak{X}^{-1} = \mathfrak{I}, \tag{2.6}$$

*where $\mathfrak{I}$ is the identity tensor of size $n \times n \times n_3$.*

Definition 2.11 (standard tensor basis). *The lateral basis $\mathring{\mathfrak{e}}_i$, is of size $n_1 \times 1 \times n_3$ with only one entry equal to* 1 *and the remaining equal to zero, in which the nonzero entry* 1 *will only appear at the first frontal slice of $\mathring{\mathfrak{e}}_i$. Normally its transpose $\mathring{\mathfrak{e}}_i^\top$ is called the horizontal basis. The other standard tensor basis is called tube basis $\dot{\mathfrak{e}}_i$, and corresponds to a tensor of size $1 \times 1 \times n_3$ with one entry equal to* 1 *and the rest equal to* 0*. Figure* 2.2 *illustrates these bases.*

**2.2. Linear algebra with tensors: Free submodules.** The set of complex numbers $\mathbb{C}$ with standard scalar addition and multiplication is a field and $\mathbb{C}^{n_3}$ forms a vector space over this field. However, as pointed out in [25, 27], the set of tubes $\mathbb{C}^{1\times 1\times n_3}$ equipped with the tensor product form a ring with unity. A module over a ring can be thought of as a generalization of a vector space over a field, where the corresponding scalars are the elements of the ring. In linear algebra over a ring, the analog of a subspace is a *free submodule*. Our algorithm relies on submodules and the following theorem.

Theorem 2.12. *The set of slices*

$$\Upsilon := \{\mathfrak{X}_{:j:} \mid \quad \mathfrak{X}_{:j:} \in \mathbb{C}^{n_1\times 1\times n_3}, \quad j \in [n_2]\} \tag{2.7}$$

*forms a free module over the set of tubes $\mathbb{C}^{1\times 1\times n_3}$.*

*Proof.* A detailed proof is provided in [4]. ■

Using Theorem 2.12, $\Upsilon$ has a *basis* so that any element of it can be written as a "t-linear combination" of elements of a set of basis slices. A t-linear combination is defined as a sum of slices multiplied, based on the t-product, by coefficients from $\mathbb{C}^{1\times 1\times n_3}$.

Definition 2.13 (slice-wise linear independence). *The slices in a subset $\Lambda = \{\mathcal{X}_{:1:}, \dots, \mathcal{X}_{:n:}\}$ of $\Upsilon$ are said to be* linearly dependent *if there exists a finite number of distinct slices $\mathcal{X}_{:1:}, \dots, \mathcal{X}_{:m:}$ in $\Lambda$, and tubes $\mathcal{C}_{11:}, \dots, \mathcal{C}_{mm:}$, not all zero, such that*

$$\sum_{j=1}^{m} \mathcal{X}_{:j:} * \mathcal{C}_{jj:} = O \tag{2.8}$$

*where $O$ denotes the lateral slice comprising of all zeros.*

*The slices in a subset $\Lambda = \{\mathcal{X}_{:1:}, \dots, \mathcal{X}_{:n:}\}$ of $\Upsilon$ are said to be* linearly independent *if the equation*

$$\sum_{j=1}^{n} \mathcal{X}_{:j:} * \mathcal{C}_{jj:} = O$$

*can only be satisfied by all zero tubes $\mathcal{C}_{jj:}$, $j = 1, \dots, n$.*

Based on the computable t-SVD, the tensor nuclear norm [44] is used to replace the tubal rank for low-rank tensor recovery (from incomplete/corrupted tensors) by solving the following convex program:

$$\min_{\mathcal{L}} \|\mathcal{L}\|_{\circledast} \qquad \text{such that (s.t.)} \qquad \|P_\Omega(\mathcal{X} - \mathcal{L})\| \leq \Delta, \tag{2.9}$$

where $\|\mathcal{L}\|_{\circledast}$ denotes the tensor nuclear norm and

$$(P_\Omega(\mathcal{X})_{ij} = \mathcal{X}_{ij} \quad \text{if} \quad (i,j) \in \Omega \quad \text{and} \quad (P_\Omega(\mathcal{X}))_{ij} = 0 \quad \text{otherwise.}$$

Similarly to the matrix completion problem, recovery of tensor $\mathcal{X}$ from its observed entries is essentially infeasible if the large majority of the entries are equal to zero [7]. For the tensor completion case, it is the case that if tensor $\mathcal{X}$ only has a few entries which are not equal to zero, in its t-SVD $\mathcal{U} * \mathcal{S} * \mathcal{V}^\top = \mathcal{X}$, the singular tensors $\mathcal{U}$ and $\mathcal{V}$ will be highly concentrated. Indeed, not all tensors can be recovered from data sets with missing entries and/or large outliers. Hence, in analogy to the main idea in matrix completion [7], tensor slices $\mathcal{U}(:, i, :)$ and $\mathcal{V}(:, i, :), i = 1, 2, \dots, r$, need to be sufficiently spread out, which in turn implies that they should be uncorrelated with the standard tensor basis. Our analysis in section 4 will focus on noisy tensor completion based on a robust factorization algorithm whose estimation/recovery guarantees are expressed in terms of the coherence of the target low-rank tensor $\mathcal{L}$. It establishes that lower values of tensor coherence provide better recovery results. In addition, we propose a randomized approximation algorithm, whose guarantees are also related to the notion of tensor coherence. The following three notions of tensor coherence are defined next.

Definition 2.14 (tensor $\mu_0$-coherence). *Let $\boldsymbol{\mathcal{V}} \in \mathbb{R}^{n\times r\times n_3}$ contain orthonormal lateral basis with $r \le n$. Then, the $\mu_0$-coherence of $\boldsymbol{\mathcal{V}}$ is given by*

$$\mu_0(\boldsymbol{\mathcal{V}}) := \frac{nn_3}{r}\max_{1\le i\le n}\|\boldsymbol{\mathcal{V}}^\top * \mathring{\mathfrak{e}}_i\|_F^2 = \frac{nn_3}{r}\max_{1\le i\le n}\|\boldsymbol{\mathcal{V}}_{i::}\|_F^2,$$

*where $\mathring{\mathfrak{e}}_i$ is standard lateral basis.*

Definition 2.15 (tensor $\mu_1$-coherence). *For a tensor $\boldsymbol{\mathcal{L}} \in \mathbb{R}^{n_1\times n_2\times n_3}$ assume that rank$_t(\boldsymbol{\mathcal{L}}) = r$. Then, the $\mu_1$-coherence of $\boldsymbol{\mathcal{L}}$ is defined as*

$$\mu_1(\boldsymbol{\mathcal{L}}) := \frac{n_1n_2n_3^2}{r}\|\boldsymbol{\mathcal{U}} * \boldsymbol{\mathcal{V}}^T\|_\infty^2,$$

*where $\boldsymbol{\mathcal{L}}$ has the skinny t-SVD $\boldsymbol{\mathcal{L}} = \boldsymbol{\mathcal{U}} * \boldsymbol{\mathcal{S}} * \boldsymbol{\mathcal{V}}^\top$.*

Definition 2.16 (tensor $(\mu, r)$-coherence). *For any $\boldsymbol{\mu} > 0$, we call a tensor $\boldsymbol{\mathcal{L}}$ $(\mu, r)$-coherent if*

$$\begin{aligned} rank_t(\boldsymbol{\mathcal{L}}) &= r,\\ \max\{\mu_0(\boldsymbol{\mathcal{U}}), \mu_0(\boldsymbol{\mathcal{V}})\} &\le \mu,\\ \mu_1(\boldsymbol{\mathcal{L}}) &\le \mu. \end{aligned}$$

Note that the standard tensor coherence condition is much weaker than the *matrix weak coherence* one for each frontal slice of $\hat{\boldsymbol{\mathcal{X}}}$ [53]. Hence, in the analysis of the proposed randomized algorithms, we will use the standard tensor coherence condition.

## 3. Approximate tensor low rank decompositions.

We develop extensions of the matrix CX and CUR decompositions [13, 15] to third-order tensors. The proposed algorithms result in computing low-rank tensor approximations that are explicitly expressed in terms of a small number of slices of the input tensor. We start by introducing a basic computational tool—a randomized tensor multiplication procedure—that is used in subsequent developments.

### 3.1. Approximate tensor multiplication.

Next, we present a randomized tensor-product and provide key results on the quality of the resulting approximation. Given two tensors $\boldsymbol{\mathcal{A}}$ and $\boldsymbol{\mathcal{B}}$, using Definition 2.1, the t-product may be written as follows:

$$\boldsymbol{\mathcal{A}} * \boldsymbol{\mathcal{B}} = \sum_{t=1}^{n_2} \boldsymbol{\mathcal{A}}_{:t:} * \boldsymbol{\mathcal{B}}_{t::}, \tag{3.1}$$

where $*$ denotes the tensor product.

It can be easily shown that the left-hand side of (3.1) is equivalent to the block multiplication and then summation in the Fourier domain. When tensor multiplication is formulated as (3.1), we can develop a randomized algorithm to approximate the product $\boldsymbol{\mathcal{A}} * \boldsymbol{\mathcal{B}}$.

Algorithm 3.1 takes as input two tensors $\boldsymbol{\mathcal{A}} \in \mathbb{R}^{n_1\times n_2\times n_3}$ and $\boldsymbol{\mathcal{B}} \in \mathbb{R}^{n_2\times n_4\times n_3}$, a positive integer $c \le n_2$, and a probability distribution $\{p_i\}_{i=1}^{n_2}$ over $[n_2]$. It returns as output two tensors $\boldsymbol{\mathcal{C}}$ and $\boldsymbol{\mathcal{R}}$, where the lateral slices of $\boldsymbol{\mathcal{C}}$ correspond to a small number of sampled and rescaled slices of $\boldsymbol{\mathcal{A}}$, and similarly the horizontal slices of $\boldsymbol{\mathcal{R}}$ constitute a small number of sampled and rescaled slices of $\mathcal{B}$. Specifically, consider at most $c$ lateral slices (in expectation) of $\mathcal{A}$ selected, with the $i$th lateral slice of $\mathcal{A}$ in $\mathcal{C}$ be chosen with probability $p_i = \min\{1, cp_i\}$. Then, define the sampling tensor $\mathcal{S} \in \mathbb{R}^{n_2\times n_2\times n_3}$ to be binary where $S_{ii1} = 1$ if the $i$th slice is selected and $\mathcal{S}_{ij2:n_3} = 0$ otherwise. Define the rescaling tensor $\mathcal{D} \in \mathbb{R}^{n_2\times c\times n_3}$ to be the tensor with $\mathcal{D}_{ij1} = 1/\sqrt{cp_j}$ if $i-1$ of the previous slices have been selected and $\mathcal{D}_{ij2:n_3} = 0$ otherwise. Analogously, the rt-product samples and rescales the corresponding horizontal slices of tensor $\mathcal{B}$. In both cases, $\mathcal{C} = \mathcal{A} * \mathcal{S} * \mathcal{D}$ is an $n_1 \times c \times n_3$ tensor consisting of sampled and rescaled copies of the lateral slices of $\mathcal{A}$, and $\mathcal{R} = (\mathcal{S} * \mathcal{D})^\top * \mathcal{B} = \mathcal{D} * \mathcal{S}^\top * \mathcal{B}$ is a $c \times n_4 \times n_3$ tensor comprising of sampled and rescaled copies of the horizontal slices of $\mathcal{B}$. For the case of $n_3 = 1$, the algorithm selects column-row pairs in Algorithm 3.1 and is identical to the algorithm in [13].

**Algorithm 3.1.** `(rt-product)`, a fast Monte Carlo algorithm for approximate tensor multiplication.

```
1:  Input: $\mathcal{A} \in \mathbb{R}^{n_1\times n_2\times n_3}$, $\mathcal{B} \in \mathbb{R}^{n_2\times n_4\times n_3}$, $p_i \geq 0, i \in [n_2]$ s.t. $\sum_{i\in[n_2]} p_i = 1$, positive
    integer $c \leq n_2$.
2:  Initialize $\mathcal{S} \in \mathbb{R}^{n_2\times n_2\times n_3}$ and $\mathcal{D} \in \mathbb{R}^{n_2\times c\times n_3}$ to the all zeros tensors;
3:  $t = 1$;
4:  for $i = 1, \dots, n_2$ do
5:      Pick $i$ with probability $\min\{1, cp_i\}$;
6:      if $i$ is picked then
7:          $\mathcal{S}_{it1} = 1, \quad \mathcal{S}_{it2:n_3} = 0$;
8:          $\mathcal{D}_{tt1} = 1/\min\{1, \sqrt{cp_i}\}, \quad \mathcal{D}_{tt2:n_3} = 0$;
9:          $t = t + 1$;
10:     end if
11: end for
12: $\mathcal{C} = \mathcal{A} * \mathcal{S} * \mathcal{D}, \quad \mathcal{R} = \mathcal{D} * \mathcal{S}^\top * \mathcal{B}$;
13: $\hat{\mathcal{C}} \leftarrow \texttt{fft}(\mathcal{C}, [], 3), \quad \hat{\mathcal{R}} \leftarrow \texttt{fft}(\mathcal{R}, [], 3)$;
14: for $k = 1, \dots, n_3$ do
15:     $\hat{\mathcal{Z}}_{::k} = \hat{\mathcal{C}}_{::k}\hat{\mathcal{R}}_{::k}$;
16: end for
17: $\mathcal{Z} \leftarrow \texttt{ifft}(\hat{\mathcal{Z}}, [], 3)$;
18: return $\mathcal{Z}$
```

## 3.2. Running time of rt-product.

The rt-product is computationally efficient, has small memory requirements, and is well suited for large scale problems. Indeed, the rt-product algorithm can be implemented without storing tensors $\mathcal{A} \in \mathbb{R}^{n_1\times n_2\times n_3}$ and $\mathcal{B} \in \mathbb{R}^{n_2\times n_4\times n_3}$ and uses $O(\max(n_1, n_4)cn_3 \log(n_3))$ flops to transform the input to the Fourier domain and $O(c(n_1 + n_2 + n_4)n_3)$ flops to construct $\mathcal{C} \in \mathbb{R}^{n_1\times c\times n_3}$ and $\mathcal{R} \in \mathbb{R}^{c\times n_4\times n_3}$, where

$$\mathcal{A} * \mathcal{B} \sim \mathcal{C} * \mathcal{R}. \tag{3.2}$$

It is worth mentioning that we do not require storing the sampling and rescaling tensors $\mathcal{S}$ and $\mathcal{D}$ in our implementation.

Next, we provide the main result on the quality of the approximation obtained by Algorithm 3.1 that specifies the assumptions under which (3.2) holds. The most interesting of these assumptions is that the sampling probabilities used to randomly sample the lateral slices of $\mathcal{A}$ and the corresponding horizontal slices of $\mathcal{B}$ are *nonuniform* and depend on the product of the norms of the lateral slices of $\mathcal{A}$ and/or the corresponding horizontal slices of $\mathcal{B}$. Following [13], we consider two examples of nonuniform sampling probabilities:

i. If we would like to use information from both tensors $\mathcal{A}$ and $\mathcal{B}$, we consider sampling probabilities $\{p_i\}_{i=1}^{n_2}$ such that

$$p_i \geq \beta \frac{\|\mathcal{A}_{:i:}\|_F \|\mathcal{B}_{i::}\|_F}{\sum_{i=1}^{n_2} \|\mathcal{A}_{:i:}\|_F \|\mathcal{B}_{i::}\|_F}, \qquad \beta \in (0,1]. \tag{3.3}$$

ii. If only information on $\mathcal{A}$ is easily available, then we use sampling probabilities $\{p_i\}_{i=1}^{n_2}$ such that

$$p_i \geq \beta \frac{\|\mathcal{A}_{:i:}\|_F^2}{\|\mathcal{A}\|_F^2}, \qquad \beta \in (0,1]. \tag{3.4}$$

The following theorem gives the main result for Algorithm 3.1 and generalizes Theorem 6 in [15] to third order tensors.

Theorem 3.1. *Suppose $\mathcal{A} \in \mathbb{R}^{n_1 \times n_2 \times n_3}$, $\mathcal{B} \in \mathbb{R}^{n_2 \times n_4 \times n_3}$, and $c \leq n_2$. In Algorithm 3.1, if the sampling probabilities $\{p_i\}_{i=1}^{n_2}$ satisfy (3.3) or (3.4), then the following holds:*

$$\mathbf{E}[\|\mathcal{A} * \mathcal{B} - \mathcal{C} * \mathcal{R}\|_F] \leq \frac{1}{\sqrt{\beta c}} \|\mathcal{A}\|_F \|\mathcal{B}\|_F. \tag{3.5}$$

The following lemma establishes a bound for tensor multiplication with respect to the *spectral norm* with improved sampling complexity.

Lemma 3.2. *Given a tensor $\mathcal{A} \in \mathbb{R}^{n_1 \times n_2 \times n_3}$, choose $c \geq 48r \log(4r/(\beta\delta))/(\beta\epsilon^2)$. Let $\mathcal{C} \in \mathbb{R}^{n_1 \times c \times n_3}$ and $\mathcal{R} = \mathcal{C}^\top$ be the corresponding subtensors of $\mathcal{A}$. If sampling probabilities $\{p_i\}_{i=1}^{n_2}$ satisfy (3.4), then with probability at least $1-\delta$, we have*

$$\begin{aligned} \|\mathcal{A} * \mathcal{A}^\top - \mathcal{C} * \mathcal{C}^\top\| &\leq \max_{k \in [n_3]} \left\{ \|\hat{\mathcal{A}}_{::k} \hat{\mathcal{A}}_{::k}^\top - \hat{\mathcal{C}}_{::k} \hat{\mathcal{C}}_{::k}^\top\|_2 \right\} \\ &\leq \epsilon/2 \|\hat{\mathcal{A}}_{::k_\pi}\|_2^2, \end{aligned} \tag{3.6}$$

*where $k_\pi = k \in [n_3]$ is the index of the tensor's frontal slice with the maximum spectral norm $\|\hat{\mathcal{A}}_{::k} \hat{\mathcal{A}}_{::k}^\top - \hat{\mathcal{C}}_{::k} \hat{\mathcal{C}}_{::k}^\top\|_2$.*

## 3.3. Slice-based tensor CX decomposition.

Next, using the concept of free submodules and Definition 2.13, we introduce the novel notion of *slice selection* and *projection* before formulating a tensor CX decomposition.

Definition 3.3. *Let $\mathcal{C} \in \mathbb{R}^{n_1 \times c \times n_3}$ and $\mathcal{X} \in \mathbb{R}^{n_1 \times n_2 \times n_3}$. Then, the tensor project (t-project) operator for $\mathcal{X}$, $\Pi_{\mathcal{C}}(\mathcal{X})$ is an $n_1 \times n_2 \times n_3$ tensor obtained as*

$$\Pi_{\mathcal{C}}(\mathcal{X}) := \mathcal{C} * \mathcal{C}^\dagger * \mathcal{X}, \tag{3.7}$$

*where $*$ denotes the t-product, $\Pi_{\mathcal{C}}(\mathcal{X})$ is the projection of $\mathcal{X}$ onto the subspace spanned by the lateral slices of $\mathcal{C}$, and $\mathcal{C}^\dagger$ is the Moore–Penrose generalized inverse of tensor $\mathcal{C}$ [26].*

Because the circulant convolution of two tube fibers can be computed by the DFT, the t-project can be alternatively computed in the Fourier domain, as shown in Algorithm 3.2.

**Algorithm 3.2.** t-project $\Pi_{\mathcal{C}}(\mathcal{X})$ computation in the Fourier domain.

```
1: Input: $\mathcal{X} \in \mathbb{R}^{n_1 \times n_2 \times n_3}$; $\mathcal{C} \in \mathbb{R}^{n_1 \times c \times n_3}$.
2: $\hat{\mathcal{X}} \leftarrow \texttt{fft}(\mathcal{X}, [], 3)$;
3: $\hat{\mathcal{C}} \leftarrow \texttt{fft}(\mathcal{C}, [], 3)$;
4: for $k = 1, \ldots, n_3$ do
5:     $\hat{\mathcal{Z}}_{::k} = \hat{\mathcal{C}}_{::k} \hat{\mathcal{C}}^{\dagger}_{::k} \hat{\mathcal{X}}_{::k}$;
6: end for
7: $\mathcal{Z} \leftarrow \texttt{ifft}(\hat{\mathcal{Z}}, [], 3)$;
8: return $\Pi_{\mathcal{C}}(\mathcal{X}) = \mathcal{Z}$
```

Definition 3.4 (t-CX). *Given an $n_1 \times n_2 \times n_3$ tensor $\mathcal{X}$, let $\mathcal{C}$ be an $n_1 \times c \times n_3$ tensor whose lateral slices correspond to $c$ lateral slices from tensor $\mathcal{X}$. Then, the $n_1 \times n_2 \times n_3$ tensor*

$$\Pi_{\mathcal{C}}(\mathcal{X}) = \mathcal{C} * \mathcal{C}^{\dagger} * \mathcal{X}$$

*is called a t-CX decomposition of $\mathcal{X}$.*

The following remarks are of interest for the previous definition.

- The choice for the number of lateral slices $c$ in the t-CX approximation depends on the application under consideration; neverthelees, we are primarily interested in the case $c \ll n_2$. For example, $c$ could be constant, independent of the dimension of tensor, or logarithmic in the size of $n_2$, or a large constant factor less than $n_2$.
- The t-CX decomposition expresses each $\mathcal{X}$ slice in terms of a linear combination of basis slices (see Definition 2.13), each of which corresponds to an actual lateral slice of $\mathcal{X}$. Hence, t-CX provides a low-rank approximation to the original tensor $\mathcal{X}$, even though its structural properties are different than those of the t-SVD.
- Given a set of lateral slices $\mathcal{C}$, the approximation $\Pi_{\mathcal{C}}(\mathcal{X}) = \mathcal{C} * \mathcal{C}^{\dagger} * \mathcal{X}$ is the "best" approximation to $\mathcal{X}$ in the following sense:

$$\|\mathcal{X} - \mathcal{C} * (\mathcal{C}^{\dagger} * \mathcal{X})\|_F = \min_{\mathcal{Y} \in \mathbb{R}^{c \times n_2 \times n_3}} \|\mathcal{X} - \mathcal{C} * \mathcal{Y}\|_F. \tag{3.8}$$

Next, we provide the t-CX decomposition algorithm and provide its relative error. Algorithm 3.3 takes as input an $n_1 \times n_2 \times n_3$ tensor $\mathcal{A}$, a tubal rank parameter $r$, and an error parameter $\epsilon$. It returns as output an $n_1 \times c \times n_3$ tensor $\mathcal{C}$ comprising of a small number of slices of $\mathcal{A}$. Let $\mathcal{L} = \mathcal{U} * \Sigma * \mathcal{V}^{\top}$ be a tubal rank-$r$ approximation of tensor $\mathcal{A}$. Central to our proposed randomized algorithm is the concept of sampling slices of the tensor based on a leverage score [15], defined by

$$p_i \geq \frac{\beta}{r n_3} \|\hat{\mathcal{V}}_{i::}\|_F^2 \qquad \forall\ i \in [n_2], \qquad \beta \in (0, 1], \tag{3.9}$$

where $\hat{\mathcal{V}} = \texttt{fft}(\mathcal{V}, [], 3)$. The goal is to select a small number of lateral slices of $\mathcal{A}$.

Note that we define rescaling and sampled tensors $\hat{\mathcal{D}}$ and $\hat{\mathcal{S}}$ in the Fourier domain. This leads to a significant reduction in time complexity of the FFT and its inverse. For the case $n_3 = 1$, the algorithm selects column-row pairs in Algorithm 3.3 and is identical to the algorithm of [15].

**Algorithm 3.3. (t-CX)**, a fast Monte Carlo algorithm for tensor low rank approximation.

```
1: **Input**: $\mathcal{A} \in \mathbb{R}^{n_1\times n_2\times n_3}$, $p_i \geq 0, i \in [n_2]$ s.t. $\sum_{i\in[n_2]} p_i = 1$, a rank parameter $r$, positive
   integer $c \leq n_2$.
2: $\hat{\mathcal{A}} \leftarrow$ fft($\mathcal{A}$, [], 3);
3: Initialize $\hat{\mathcal{S}}$ and $\hat{\mathcal{D}}$ to the all zeros tensors;
4: $t = 1$;
5: **for** $i = 1, \ldots, n_2$ **do**
6:    Pick $i$ with probability $\min\{1, cp_i\}$;
7:    **if** $i$ is picked, **then**
8:       $\hat{\mathcal{S}}_{it1} = 1, \quad \hat{\mathcal{S}}_{it2:n_3} = 0$;
9:       $\hat{\mathcal{D}}_{tt1} = 1/\min\{1, \sqrt{cp_i}\}, \quad \hat{\mathcal{D}}_{tt2:n_3} = 0$;
10:      $t = t + 1$;
11:   **end if**
12: **end for**
13: **for** $k = 1, \ldots, n_3$ **do**
14:    $\hat{\mathcal{Z}}_{::k} = \hat{\mathcal{A}}_{::k}\hat{\mathcal{S}}_{::k}\hat{\mathcal{D}}_{::k}(\hat{\mathcal{A}}_{::k}\hat{\mathcal{S}}_{::k}\hat{\mathcal{D}}_{::k})^{\dagger}\hat{\mathcal{A}}_{::k}$;
15: **end for**
16: $\mathcal{Z} \leftarrow$ ifft($\hat{\mathcal{Z}}$, [], 3);
17: **return** $\mathcal{Z}$
```

Next, we present a lemma of general interest. The derived properties aid in obtaining tensor based randomized $\ell_2$ regression and low rank estimation guarantees.

Lemma 3.5. *Given a target tensor $\mathcal{A} \in \mathbb{R}^{n_1\times n_2\times n_3}$, let $\epsilon \in (0, 1]$ and $\mathcal{L} = \mathcal{U} * \Sigma * \mathcal{V}^\top$ be a* rank$_t$*-$r$ approximation of $\mathcal{A}$. Let $\Gamma = (\mathcal{V}^\top * \mathcal{S} * \mathcal{D})^{\dagger} - (\mathcal{V}^\top * \mathcal{S} * \mathcal{D})^\top$. If the sampling probabilities $\{p_i\}_{i=1}^{n_2}$ satisfy (3.9) and if $c \geq 48r\log(4r/(\beta\delta))/(\beta\epsilon^2)$, then with probability at least $1 - \delta$, the following hold:*

$$\operatorname{rank}_t(\mathcal{V}^\top * \mathcal{S}) = \operatorname{rank}_t(\mathcal{V}) = \operatorname{rank}_t(\mathcal{L}), \tag{3.10a}$$

$$\|\Gamma\| = \|\Sigma^{-1}_{\mathcal{V}^\top *\mathcal{S}*\mathcal{D}} - \Sigma_{\mathcal{V}^\top *\mathcal{S}*\mathcal{D}}\|, \tag{3.10b}$$

$$(\mathcal{L} * \mathcal{S} * \mathcal{D})^{\dagger} = (\mathcal{V}^\top * \mathcal{S} * \mathcal{D})^{\dagger}\Sigma^{-1}\mathcal{U}^\top, \tag{3.10c}$$

$$\|\Sigma^{-1}_{\mathcal{V}^\top *\mathcal{S}*\mathcal{D}} - \Sigma_{\mathcal{V}^\top *\mathcal{S}*\mathcal{D}}\|_2 \leq \frac{\epsilon\sqrt{2}}{2}, \tag{3.10d}$$

*where $\Sigma_{\mathcal{V}^\top *\mathcal{S}*\mathcal{D}}$ is an F-diagonal tensor and contains the $r$ nonzero singular tubes of $\mathcal{V}^\top * \mathcal{S} * \mathcal{D}$.*

Note that (3.10d) shows that in terms of its singular tubes, the tensor $\mathcal{V}^\top * \mathcal{S} * \mathcal{D}$, i.e., the slice sampled and rescaled version of $\mathcal{V}$, is almost an orthogonal tensor. A useful property of an orthogonal tensor $\mathcal{V}$ is that $\mathcal{V}^{\dagger} = \mathcal{V}^\top$ (see Definition 2.4). Equation (3.10b) shows that although this property does not hold for $\mathcal{V}^\top * \mathcal{S} * \mathcal{D}$, the difference between $(\mathcal{V}^\top * \mathcal{S} * \mathcal{D})^\dagger$ and $(\mathcal{V}^\top * \mathcal{S} * \mathcal{D})^\top$ can be bounded.

Using Theorem 3.1 and Lemma 3.5, we provide in proposition 3.6 a sampling complexity for a tensor based randomized "$\ell_2$-regression" as in [15]. Indeed, Lemmas 2 and 3 of [15] follow by using the Frobenius norm bound of Theorem 3.1 and applying Markov's inequality. The claim of Lemma 1 of [15] follows by applying Lemma 3.5. If we consider the failing probability $\delta = 0.05$, the claims of all three lemmas hold simultaneously with probability at least 0.85, and in the following proposition we condition on this event. The proof follows along similar lines to the proof of Theorem 5 in [15].

Proposition 3.6. *Given a target tensor $\mathcal{A} \in \mathbb{R}^{n_1\times n_2\times n_3}$, let $\mathcal{L} = \mathcal{U} * \Sigma * \mathcal{V}^\top$ be a* rank$_t$*-$r$ approximation of $\mathcal{A}$. Choose $c = O(r\log(r/\beta)/(\beta\epsilon^2))$ slices of $\mathcal{A}$ with their sampling probabilities $\{p_i\}_{i=1}^{n_2}$ satisfying (3.9). Then, with probability at least* 0.85*, the following holds:*

$$\|\mathcal{A} - \mathcal{A} * \mathcal{S} * \mathcal{D} * (\mathcal{L} * \mathcal{S} * \mathcal{D})^\dagger * \mathcal{L}\|_F \le (1+\epsilon)\|\mathcal{A} - \mathcal{A} * \mathcal{L}^\dagger * \mathcal{L}\|_F. \tag{3.11}$$

Let $\mathcal{L} = \mathcal{U} * \Sigma * \mathcal{V}^\top$ be a t-SVD of tensor $\mathcal{L} \in \mathbb{R}^{n_1\times n_2\times n_3}$. In the remainder of the paper, we use the following two parameters related to the coherence of tensor $\mathcal{L}$:

$$\varrho := \frac{r\mu_0(\mathcal{V})}{n_3},$$
$$\varrho_c := \frac{c\mu_0(\mathcal{U}_c)}{n_3}, \tag{3.12}$$

where $\mathcal{U}_c$ is the left singular tensor of $\mathcal{C} \in \mathbb{R}^{n_1\times c\times n_3}$ lateral slices of $\mathcal{L}$.

The following theorem shows that projection based on slice sampling leads to near optimal estimation in tensor regression when the covariate tensor has small coherence, $\mu_0(\mathcal{V})$.

Theorem 3.7. *Given $\mathcal{A} \in \mathbb{R}^{n_1\times n_2\times n_3}$, let $\mathcal{L} = \mathcal{U} * \Sigma * \mathcal{V}^\top$ be a* rank$_t$*-$r$ approximation of $\mathcal{A}$. Choose $c = O(\varrho\log(\varrho)/\epsilon^2)$, and let $\mathcal{A}_c \in \mathbb{R}^{n_1\times c\times n_2}$ be a tensor of $c$ lateral slices of $\mathcal{A}$ sampled uniformly without replacement. Further, let $\mathcal{L}_c \in \mathbb{R}^{n_1\times c\times n_3}$ consist of the corresponding slices of $\mathcal{L}$. Then, with probability at least* 0.85*, the following holds:*

$$\|\mathcal{A} - \mathcal{A}_c * \mathcal{L}_c^\dagger * \mathcal{L}\|_F \le (1+\epsilon)\|\mathcal{A} - \mathcal{A} * \mathcal{L}^\dagger * \mathcal{L}\|_F.$$

Next, by using Lemma 3.5, we provide a low rank estimation bound for the t-CX algorithm under the coherence assumption. Indeed, we will take advantage of a constant $\beta$ in (3.9) to provide relative error estimation guarantees for the t-CX algorithm under uniform slice sampling when tensor $\mathcal{A}$ under consideration exhibits sufficient incoherence.

Corollary 3.8. *Given a target tensor $\mathcal{A} \in \mathbb{R}^{n_1\times n_2\times n_3}$, let $\mathcal{L} = \mathcal{U} * \Sigma * \mathcal{V}^\top$ be a* rank$_t$ *$-r$ approximation of $\mathcal{A}$. Choose $c = O(\varrho\log(\varrho)\log(1/\delta)/\epsilon^2)$, and let $\mathcal{C}$ be a tensor of $c$ lateral slices of $\mathcal{A}$ sampled uniformly without replacement. Then, the following holds:*

$$\|\mathcal{A} - \mathcal{C} * \mathcal{C}^\dagger * \mathcal{A}\|_F \le (1+\epsilon)\|\mathcal{A} - \mathcal{L}\|_F \tag{3.13}$$

*with probability at least $1-\delta$.*

**3.4. Slice-based tensor CUR decomposition.** Similar to the matrix case, the t-CX decomposition suffices when $n_1 \ll n_2$ because $\mathfrak{C}^\dagger * \mathcal{A}$ is small in size. However, when $n_1$ and $n_2$ are almost equal, computing and storing the dense matrix $\mathfrak{C}^\dagger * \mathcal{A}$ in memory becomes prohibitive. The CUR decomposition provides a very useful alternative. Next, we introduce a tensor CUR (t-CUR) decomposition based on the rt-product.

Definition 3.9 (t-CUR). *Let $\mathfrak{X}$ be an $n_1 \times n_2 \times n_3$ tensor. For any given $\mathfrak{C}$, an $n_1 \times c \times n_3$ tensor whose slices comprise of $c$ lateral slices of tensor $\mathfrak{X}$, and $\mathfrak{R}$, an $l \times n_2 \times n_3$ tensor whose slices comprise of $l$ horizontal slices of tensor $\mathfrak{X}$, the $n_1 \times n_2 \times n_3$ tensor*

$$\mathfrak{C} * \mathfrak{U} * \mathfrak{R}$$

*is a lateral-horizontal-based tensor approximation to $\mathfrak{X}$ for any $c \times l \times n_3$ tensor $\mathfrak{U}$.*

The following remarks are of interest regarding the previous definition.

- The t-CUR decomposition is most appropriate as a data analysis tool when the data consist of one and/or two modes that are qualitatively different than the remaining ones. In this case, the t-CUR decomposition approximately expresses the original data tensor in terms of a basis consisting of underlying subtensors that are actual data slices and not artificial bases.
- The t-CUR approximation is a t-CX approximation, but one with a very special structure; i.e., every lateral slice of $\mathfrak{X}$ can be expressed in terms of the basis provided by $\mathfrak{C}$ (see Definition 2.13) using only the information contained in a small number of horizontal slices of $\mathfrak{X}$ and a low-dimensional encoding tensor.
- In terms of its t-SVD structure, $\mathfrak{U}$ contains the "inverse-of-$\mathfrak{X}$" information. For the proposed t-CUR decomposition, $\mathfrak{U}$ will be a generalized inverse of the intersection between selected tensors $\mathfrak{C}$ and $\mathfrak{R}$.

Note that the structural simplicity of the t-CUR tensor decomposition becomes apparent in the Fourier domain, as detailed in Algorithm 3.4. The latter takes as input an $n_1 \times n_2 \times n_3$ tensor $\mathcal{A}$, an $n_1 \times c \times n_3$ tensor $\mathfrak{C}$ consisting of a small number of lateral slices of $\mathcal{A}$, and an error parameter $\epsilon$. Letting $\hat{\mathfrak{C}} = \hat{\mathfrak{U}} * \hat{\Sigma} * \hat{\mathfrak{V}}^\top$, Algorithm 3.4 uses sampling probabilities

$$p_i \geq \frac{\beta}{cn_3} \|\hat{\mathfrak{U}}_{i::}\| \qquad \forall i \in [n_1], \qquad \beta \in (0, 1], \tag{3.14}$$

to select a small number of horizontal slices of $\mathcal{A}$. It returns an $l \times n_2 \times n_3$ tensor $\mathfrak{R}$ comprising of a small number of horizontal slices of $\mathcal{A}$ and an $c \times l \times n_3$ tensor $\mathfrak{U}$ consisting of the corresponding slices of $\mathfrak{C}$.

Corollary 3.10. *Given $\mathcal{A} \in \mathbb{R}^{n_1 \times n_2 \times n_3}$, let $\mathcal{L} = \mathfrak{U} * \Sigma * \mathfrak{V}^\top$ be a $\text{rank}_t -r$ approximation of $\mathcal{A}$. In Algorithm 3.4, choose $c = O(\varrho \log(\varrho) \log(1/\delta)/\epsilon^2)$, and let $\mathfrak{C} \in \mathbb{R}^{n_1 \times c \times n_3}$ be a tensor of $c$ slices of $\mathcal{A}$ sampled uniformly without replacement. Further, choose $l = O(\varrho_c \log(\varrho_c) \log(1/\delta)/\epsilon^2)$, and let $\mathfrak{R} \in \mathbb{R}^{l \times n_2 \times n_3}$ be a tensor of $l$ horizontal slices of $\mathcal{A}$ sampled uniformly without replacement. Then, the following holds:*

$$\|\mathcal{A} - \mathfrak{C} * \mathfrak{U} * \mathfrak{R}\|_F \leq (1 + \epsilon) \|\mathcal{A} - \mathcal{L}\|_F$$

*with probability at least $1 - \delta$.*

**Algorithm 3.4. (t-CUR)**, a fast Monte Carlo algorithm for tensor low rank approximation.

```
1: Input: $\mathcal{A} \in \mathbb{R}^{n_1 \times n_2 \times n_3}$, $\mathcal{C} \in \mathbb{R}^{n_1 \times c \times n_3}$ consisting of $c$ lateral slices of $\mathcal{A}$, $p_i \geq 0, i \in [n_2]$
   s.t. $\sum_{i \in [n_2]} p_i = 1$, a rank parameter $r$, and positive integer $l \leq n_1$.
2: $\hat{\mathcal{A}} \leftarrow \texttt{fft}(\mathcal{A}, [], 3)$;
3: $\hat{\mathcal{C}} \leftarrow \texttt{fft}(\mathcal{C}, [], 3)$;
4: Initialize $\hat{\mathcal{S}}$ and $\hat{\mathcal{D}}$ to the all zeros tensors;
5: $t = 1$;
6: for $i = 1, \ldots, n_1$ do
7:   Pick $i$ with probability $\min\{1, lp_i\}$;
8:   if $i$ is picked, then
9:     $\hat{\mathcal{S}}_{it1} = 1, \quad \hat{\mathcal{S}}_{it2:n_3} = 0$;
10:    $\hat{\mathcal{D}}_{tt1} = 1/\min\{1, \sqrt{lp_i}\}, \quad \hat{\mathcal{D}}_{tt2:n_3} = 0$;
11:    $t = t + 1$;
12:  end if
13: end for
14: for $k = 1, \ldots, n_3$ do
15:   $\hat{\mathcal{R}}_{::k} = \hat{\mathcal{D}}_{::k} \hat{\mathcal{S}}_{::k}^{\top} \hat{\mathcal{A}}_{::k}$;
16:   $\hat{\mathcal{U}}_{::k} = \left(\hat{\mathcal{D}}_{::k} \hat{\mathcal{S}}_{::k}^{\top} \hat{\mathcal{C}}_{::k}\right)^{\dagger}$;
17:   $\hat{\mathcal{Z}}_{::k} = \hat{\mathcal{C}}_{::k} \hat{\mathcal{U}}_{::k} \hat{\mathcal{R}}_{::k}$;
18: end for
19: $\mathcal{Z} \leftarrow \texttt{ifft}(\hat{\mathcal{Z}}, [], 3)$;
20: return $\mathcal{Z}$
```

*Remark* 3.11. In many applications such as tensor completion problems discussed in section 5.3, it may not be feasible to compute the t-SVD of the entire tensor $\mathcal{A}$ due to either computational cost or large set of missing values. In these cases, algorithms requiring knowledge of the leverage scores can not be applied. Hence, we may use an estimate where a subset of the lateral slices are chosen uniformly and without replacement, and the horizontal leverage scores (3.14) are calculated using the top-$r$ left singular slices of this lateral tensor instead of the entire $\mathcal{A}$ tensor.

*Remark* 3.12. Note that for any subspace, the smallest $\mu_0$ can be is 1. In such case, we are using the optimal subspace sampling with $\beta = 1$. Note also that we have provided Corollaries 3.8 and 3.10 based on uniform sampling. However, it can be easily seen that the relative error guarantees hold with nonuniform sampling probabilities (3.9) and (3.14), if we set $c = O(r \log(r/\beta) \log(1/\delta)/(\beta\epsilon^2))$, and $l = O(c \log(c/\beta) \log(1/\delta)/(\beta\epsilon^2))$.

## 4. Tensor completion and robust factorization.

Next, we provide a CUR tensor nuclear norm minimization (CUR t-NN) procedure that solves a noisy tensor factorization problem, using a small number of lateral and/or horizontal slices of the underlying tensor and exhibits favorable computational complexity and in addition comes with performance guarantees.

### 4.1. Related work on matrix problems.

*Low rank plus sparse matrix decomposition.* In many image processing and computer vision applications, the given data matrix $X$ can be decomposed as a sum of a low-rank and a sparse component. To that end, Candés et al. proposed robust PCA [6] to model data matrices generated according to the following mechanism:

$$\min_{L,E} \operatorname{rank}(L) + \|E\|_0 \qquad \text{s.t.} \qquad X = L + E. \tag{4.1}$$

Note that the solution of (4.1) is an NP-hard problem. It is established in [7] that if $L$ exhibits a certain degree of low-rankness, while $E$ is sparse enough, then the formulation of (4.1) can be relaxed into a convex problem of the form:

$$\min_{L,E} \|L\|_* + \|E\|_1 \qquad \text{s.t.} \qquad X = L + E. \tag{4.2}$$

This model implicitly assumes that the underlying data structure lies in a single low-rank subspace. However, in many applications (e.g., image classification) it is more likely that the data are obtained from a union of multiple subspaces, and hence recovery of the structure based on the above decomposition would be inaccurate. In order to segment the data into their respective subspaces, one needs to compute an affinity matrix that encodes the pairwise affinities between data vectors. Liu et al. [32] proposed a more general rank minimization problem, where the data matrix itself is used as the dictionary, resulting in the following convex optimization problem:

$$\min_{L,E} \|L\|_* + \|E\|_{2,1} \qquad \text{s.t.} \qquad X = XL + E, \qquad \operatorname{diag}(L) = 0. \tag{4.3}$$

When the subspaces are *globally* independent, the data are noiseless and sampling is sufficient; Liu et al. [32] show that the optimal solution, denoted by $L^*$, to the problem given by (4.3) corresponds to the widely used shape iteration matrix method [9]. The latter is a "block-diagonal" affinity matrix that indicates the true segmentation of the data. To handle data corrupted by noise, the popular low rank representation (LRR) introduced in [32] adopts a regularized formulation that introduces an extra penalty term to fit the noise component. Further, after obtaining the self-representation matrix $L$, the affinity matrix $C$ is usually constructed as $C = \frac{1}{2}(|L| + |L^\top|)$, where $|\cdot|$ represents the absolute operator. Then, the obtained affinity matrix $C$ will be processed through a spectral clustering algorithm [40] to produce the final clustering result and obtain the corresponding data generating subspaces.

It is established in [52, 47] that combining sparse and low-rank regularization can improve the performance of image classification. The basic objective function of this combination [52] is as follows:

$$\min_{L,E} \lambda_1\|L\|_* + \lambda_2\|L\|_1 + \lambda_3\|E\|_\ell \qquad \text{s.t.} \qquad X = XL + E, \qquad \operatorname{diag}(L) = 0, \tag{4.4}$$

where $\lambda_1, \lambda_2, \lambda_3$ are tuning parameters and $\|E\|_\ell$ indicates different norms suitable for different types for corrupting the data by noise, for example, the squared Frobenius norm for Gaussian noise and the $\ell_1$ norm for random spiked noise. Equation (4.4) is similar to the objective functions in [47, 16], where a detailed explanation of the formulation given in (4.4) is also provided.

#### 4.1.1. Low rank tensor decomposition.
Lu et al. [35] extended robust PCA [6] to the third order tensor based on t-SVD and proposed the following convex optimization problem:

$$\min_{\mathcal{L},\mathcal{E}} \|\mathcal{L}\|_{\circledast} + \lambda\|\mathcal{E}\|_1 \qquad \text{s.t.} \qquad \mathcal{X} = \mathcal{L} + \mathcal{E}, \tag{4.5}$$

where $\lambda$ is a tuning parameter.

This model implicitly assumes that the underlying data come from a single low-rank subspace. When the data is drawn from a union of multiple subspaces, which is common in image classification, the recovery based on the above formulation may lack in accuracy. To that end, Xie et al. [49] extended LRR based subspace clustering to a multiview one by employing the rank sum of different mode unfoldings to constrain the subspace coefficient tensor, resulting in the following convex optimization problem:

$$\min_{\mathcal{L},\mathcal{E}} \|\mathcal{L}\|_{\circledast} + \lambda\|\mathcal{E}\|_{2,1} \qquad \text{s.t.} \qquad \mathcal{X} = \mathcal{X} * \mathcal{L} + \mathcal{E}. \tag{4.6}$$

Equation (4.6) is similar to the objective functions in [25, 42], and a detailed explanation of (4.6) is provided in that paper.

### 4.2. Proposed algorithm.
Next, we propose an algorithm for large scale tensor decomposition of noisy data. Our proposal, called CUR t-NN, extends Algorithm 3.4 to the *noisy* tensor factorization. The main steps are outlined in Algorithm 4.1.

Note that CUR t-NN can be used in combination with an arbitrary optimization algorithm. In this paper, we have chosen to solve the noisy tensor factorization formulations (2.9), (4.5), and (4.6) using an alternating direction method of multiplier (ADMM) algorithm [3]. ADMM is the most widely used approach for robust tensor PCA in both the fully and partially observed settings. Indeed, ADMM achieves much higher accuracy than (accelerated) proximal gradient algorithm using fewer iterations. It works well across a wide range of problem settings and does not require careful tuning of the regularization parameters. Further, the following empirical finding has been frequently observed: namely, the rank of the iterates often remains bounded by the rank of the initializer, thus enabling efficient computations [6]. This feature is not shared by the block coordinate decent algorithm. We provide a variant of ADMM for solving problem (4.5) in the Appendix (see Algorithm A.1). With a small modification, Algorithm A.1 can be also used to solve the tensor completion problem (2.9), and tensor subspace clustering problem (4.6). In the following algorithm, $\mathbf{ADMM}(\mathcal{X}, \lambda)$ denotes the ADMM algorithm for solving regularized tensor nuclear norm minimization problems (2.9), (4.5), and (4.6), where $\mathcal{X}$ is the (sampled) data tensor and $\lambda$ is a regularization parameter.

### 4.3. Running time of CUR t-NN.
Algorithm 4.1 significantly reduces the per-iteration complexity of nuclear norm minimization problems. Indeed, in each iteration, a base tensor nuclear norm minimization algorithm requires $O(n_1 n_2 n_3 \log(n_3))$ flops to transform the tensor to the Fourier domain, $O(n_1 n_2 n_3 \min(n_1, n_2))$ flops for the t-SVD computation and factorization, and $O(n_1 n_2 n_3 \log(n_3))$ flops to transform it back to the original domain. On the other hand, Algorithm 4.1 only requires $O(\max(cn_1, ln_2)n_3 \log(n_3))$, $O(\max(n_1 c, ln_2)n_3 r)$, and $O(\max(cn_1, ln_2)n_3 \log(n_3))$ flops for the respective steps. Further, Algorithm 4.1 can be implemented without storing the data tensor $\mathcal{X}$ and can be advantageous when $r \ll \min(n_1, n_2)$, which occurs frequently in real data sets.

**Algorithm 4.1.** `CUR t-NN`, tensor nuclear norm minimization based on CUR factorization.

```
1: $\mathcal{X} \in \mathbb{R}^{n_1 \times n_2 \times n_3}$, positive integer $c$ and $l$, and a regularization parameter $\lambda$;
2: Let $\mathcal{C}$ be a tensor of $c$ selected lateral slices of $\mathcal{X}$ using probabilities (3.9);
3: $\tilde{\mathcal{C}} \leftarrow$ **ADMM**$(\mathcal{C}, \lambda)$;
4: Let $\mathcal{R}$ be a tensor of $l$ selected horizontal slices of $\mathcal{X}$ using probabilities (3.14);
5: $\tilde{\mathcal{R}} \leftarrow$ **ADMM**$(\mathcal{R}, \lambda)$;
6: Let $\tilde{\mathcal{U}} = \tilde{\mathcal{W}}^{\dagger}$, where $\tilde{\mathcal{W}}$ is the $l \times c \times n_3$ tensor formed by sampling the corresponding $l$
   horizontal slices of $\tilde{\mathcal{C}}$;
7: $\tilde{\mathcal{L}} = \tilde{\mathcal{C}} * \tilde{\mathcal{U}} * \tilde{\mathcal{R}}$;
8: **return** $\tilde{\mathcal{L}}$
```

## 4.4. Theoretical guarantees.

Next, using Lemma 3.5, we establish that Algorithm 4.1 exhibits high probability recovery guarantees comparable to those of the exact algorithms that use the full data tensor. Our first result bounds the $\mu_0$ and $\mu_1$-coherence (see Definitions 2.14 and 2.15) of a randomized tensor in terms of the coherence of the full tensor.

Lemma 4.1. *Let $\mathcal{L}_c$ be a tensor formed by selecting $c$ slices of a* $\text{rank}_t - r$ *tensor $\mathcal{L} = \mathcal{U} * \Sigma * \mathcal{V}$ that satisfy the probabilistic conditions in* (3.9). *If $c \geq 48r\log(4r/(\beta\delta))/(\beta\epsilon^2)$, then with probability at least $1-\delta$,*

- $\mu_0(\mathcal{U}_{\mathcal{L}_c}) = \mu_0(\mathcal{U})$,
- $\mu_0(\mathcal{V}_{\mathcal{L}_c}) \leq \dfrac{1}{1-\epsilon/2}\mu_0(\mathcal{V})$,
- $\mu_1(\mathcal{L}_c) \leq \dfrac{r}{1-\epsilon/2}\mu_0(\mathcal{U})\mu_0(\mathcal{V})$,

*where $\epsilon \in (0, 1]$.*

Our next theorem provides a bound for the estimation error of the CUR t-NN algorithm.

Theorem 4.2. *Under the notion of Algorithm 4.1, choose $c = O(\varrho\log(\varrho)\log(1/\delta)/\epsilon^2)$ and $l = O(\varrho_c\log(\varrho_c)\log(1/\delta)/\epsilon^2)$. Let $\mathcal{C}^*$ and $\mathcal{R}^*$ be the corresponding lateral and horizontal subtensors of the exact solution $\mathcal{L}^*$. If $\mathcal{L}^*$ is $(\mu, r)$-coherent, then with probability at least $1-\delta$, $\mathcal{C}^*$ and $\mathcal{R}^*$ are $(\frac{r\mu^2}{1-\epsilon/2}, r)$-coherent and*

$$\|\mathcal{L}^* - \tilde{\mathcal{L}}\|_F \leq (2+\epsilon)\sqrt{\|\mathcal{C}^* - \tilde{\mathcal{C}}\|_F^2 + \|\mathcal{R}^* - \tilde{\mathcal{R}}\|_F^2},$$

*where $\tilde{\mathcal{L}}$ is a solution obtained by Algorithm 4.1.*

## 5. Experimental results.

Next, we investigate the efficiency of the proposed randomized algorithms on both synthetic and real data sets. The results are organized in the following three subsections: in section 5.1, we compare the performance of the rt-product to that of the t-product using synthetic data. In section 5.2, we use the proposed t-CX and t-CUR algorithms for finding important ions and positions in two mass spectrometry imaging (MSI) data sets. Finally, in section 5.3, we apply the proposed tensor factorization CUR t-NN algorithm on data sets related to image and video processing.

All algorithms have been implemented in the MATLAB R2018a environment and run on a Mac machine equipped with a 1.8 GHz Intel Core i5 processor and 8 GB 1600 MHz DDR3 of memory.

Table 5.1
*Relative errors and running times of tensor multipication algorithms.*

| Rank, selected slices | t-product | rt-product | | | | | |
|---|---|---|---|---|---|---|---|
| | | Nonuniform sampling | | | Uniform sampling | | |
| | Time (s) | Time(s) | RFE | RSPE | Time(s) | RFE | RSPE |
| r=100, l=460 | 9.09 | 8.13 | 70e-3 | 43e-4 | 1.03 | 39e-2 | 11e-2 |
| r=200, l=1000 | 36.36 | 14.27 | 64e-3 | 25e-4 | 1.08 | 12e-2 | 82e-2 |
| r=300, l=1700 | 162.99 | 34.89 | 56e-3 | 29e-4 | 1.39 | 41e-2 | 34e-2 |
| r=400, l=2500 | 321.43 | 41.04 | 31e-3 | 13e-3 | 1.64 | 90e-2 | 38e-3 |
| r=500, l=3100 | 684.01 | 67.07 | 71e-3 | 12e-4 | 3.09 | 23e-2 | 43e-3 |

**5.1. Experimental results for fast tensor multipication.** A random data tensor $\mathcal{X} \in \mathbb{R}^{n_1\times n_2\times n_3}$ of dimension $n_1 = 10^5$, $n_2 = 10^3$, and $n_3 = 5$ is generated as follows: the first frontal slice of $\mathcal{X}$ is set to $\mathcal{X}(:,:,1) = \texttt{sprandn}(n_1, n_2, 1)$, i.e., comprising of $n_1 \times n_2$ sparse normally distributed random matrix. The remaining frontal slices are generated by $\texttt{sprandn}(\mathcal{X}(:,:,1))$, which results in the *same sparsity* structure as $\mathcal{X}(:,:,1)$, with normally distributed random entries with mean 0 and variance 1.

We perform a t-SVD decomposition on $\mathcal{X}$ and select the left singular slices $\mathcal{U}$ for our experiments. We consider both uniform and nonuniform sampling schemes. For nonuniform sampling, we use a randomized approach (similar to Algorithm 4 in [14]) to obtain normalized leverage scores of $\mathcal{U}$. Further, the Frobenius and spectral bounds given in (3.5) and (3.6), respectively, are used as performance metrics for Algorithm 3.1. Table 5.1 shows the relative Frobenius error (RFE)—$\|\mathcal{U}_r^\top\mathcal{U}_r - \tilde{\mathcal{U}}_r^\top\tilde{\mathcal{U}}_r\|_F/\|\mathcal{U}_r\|_F^2$—and relative spectral error (RSPE)—$\|\mathcal{U}_r^\top\mathcal{U}_r - \tilde{\mathcal{U}}_r^\top\tilde{\mathcal{U}}_r\|_2/\|\mathcal{U}_r\|_2^2$—for the rt-product algorithm to recover $\tilde{\mathcal{U}}_r^\top\tilde{\mathcal{U}}_r = \mathcal{U}_r^\top * \mathcal{S} * \mathcal{D} * \mathcal{D} * \mathcal{S}^\top * \mathcal{U}_r$ based on $l \approx r\log(r)$ slices and its deterministic counterpart. The depicted results are averaged over 10 independent replications.

Very large improvements in computational speed can be seen when using the rt-product, especially when coupled with a nonuniform sampling scheme. Further, the gains become more pronounced for larger number of slices sampled and larger rank.

**5.2. Finding important ions and positions in MSI.** MSI is used to visualize the spatial distribution of chemical compounds, such as metabolites, peptides or proteins by their molecular masses. The ability of MSI to localize panels of biomolecules in tissue samples has led to a rapid and substantial impact in both clinical and pharmacological research, aided in uncovering biomolecular changes associated with disease and finally provided low cost imaging of drugs. Typical techniques used require finding important ions and positions from a three-dimensional image: ions to be used in fragmentation studies to identify key compounds and positions for follow-up validation measurements using microdissection or other orthogonal techniques. Unfortunately, with modern imaging machines, these must be selected from an overwhelming amount of raw data. Existing popular techniques used to reduce the volume of data include PCA and nonnegative matrix factorization, but they return difficult-to-interpret linear combinations of actual data elements. A recent paper [50] shows that CX and CUR matrix decompositions can be used directly to address this selection need. One major shortcoming of CX and/or CUR matrix decompositions is that they can only handle

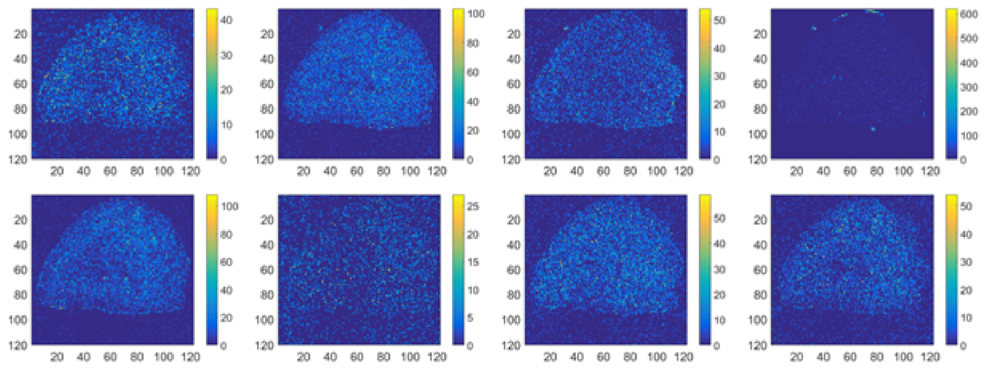


**Figure 5.1.** *Sample ions in the brain data set.*

2-way (matrix) data. However, MSI data form a multidimensional array. Hence, in order to use CX/CUR matrix decompositions, one has first to reformulate the multiway array as a matrix. Such preprocessing usually leads to information loss, which in turn could cause significant performance degradation.

By using instead the t-CX and t-CUR decompositions (Algorithms 3.3 and 3.4, respectively) one can obtain good low-rank approximations of the available data, expressed as combinations of actual ions and positions, as opposed to difficult-to-interpret eigen-ions and eigen-positions produced by matrix factorization techniques. We show that this leads to effective prioritization of information for both actual ions and actual positions. In particular, important ions can be discovered by using leverage scores as the importance sampling distribution. Further, selection of important positions from the original tensor can be accomplished based on the random sampling algorithm in [50], since the distribution of the leverage scores of positions is uniform. To this end, we consider the following two ways of computing leverage scores of a given data set.

- Deterministic: Compute the normalized tensor leverage scores exactly using probabilities (3.9) and (3.14).
- Randomized: Compute an approximation to the normalized leverage scores of tensor (mapped to a block diagonal matrix in the Fourier domain) by using Algorithm 4 of [14].

**5.2.1. Description and analysis of MSI data sets.** Next, we use the following two data sets for illustration purposes that are publicly available at the OpenMSI Web gateway.[2] They represent two diverse acquisition modalities, including one mass spectrometry image of the left coronal hemisphere of a mouse brain (see Figure 5.1) acquired using a time-of-flight mass analyzer and one MSI data set of a lung acquired using an Orbitrap mass analyzer. These data sets form a $122\times120\times80339$ and a $122\times120\times500000$ tensor, respectively. As described in [50], the brain data set is processed using peak-finding to identify the most intense ions. Using this technique, the original brain data is reduced from $122 \times 120 \times 80339$ values to a data set of size $122 \times 120 \times 2000$. To compute the CX decomposition, we reshape the MSI

[2]https://openmsi.nersc.gov/openmsi/client/.

data cube into a two-dimensional ($14640 \times 2000$) matrix, where each row corresponds to the spectrum of a pixel in the image, and each column to the intensity of an ion across all pixels, thus describing the distribution of the ion in physical space. No peak-finding was applied to the lung data set.

For the brain data set, we evaluate the quality of the approximation of the leverage scores based on a rank $r = 5$ approximation. The distribution of the deterministic leverage scores for the ions and pixels is shown in Figure 5.2. Table 5.2 shows the relative square error (RSE) $\|\mathcal{X} - \tilde{\mathcal{X}}\|_F / \|\mathcal{X}\|_F$ using t-CX decomposition to recover rank $-5$ tensor $\tilde{\mathcal{X}}$ for selection of $c = 25, 35, 45$, and 55 ions, using both randomized and deterministic leverage scores. Table 5.3 provides the reconstruction errors to the best rank$-5$ approximation, based on the t-CX decomposition coupled with horizontal slice selection. The results obtained running both randomized and deterministic CX and t-CX algorithms are based on 10 independent replicates

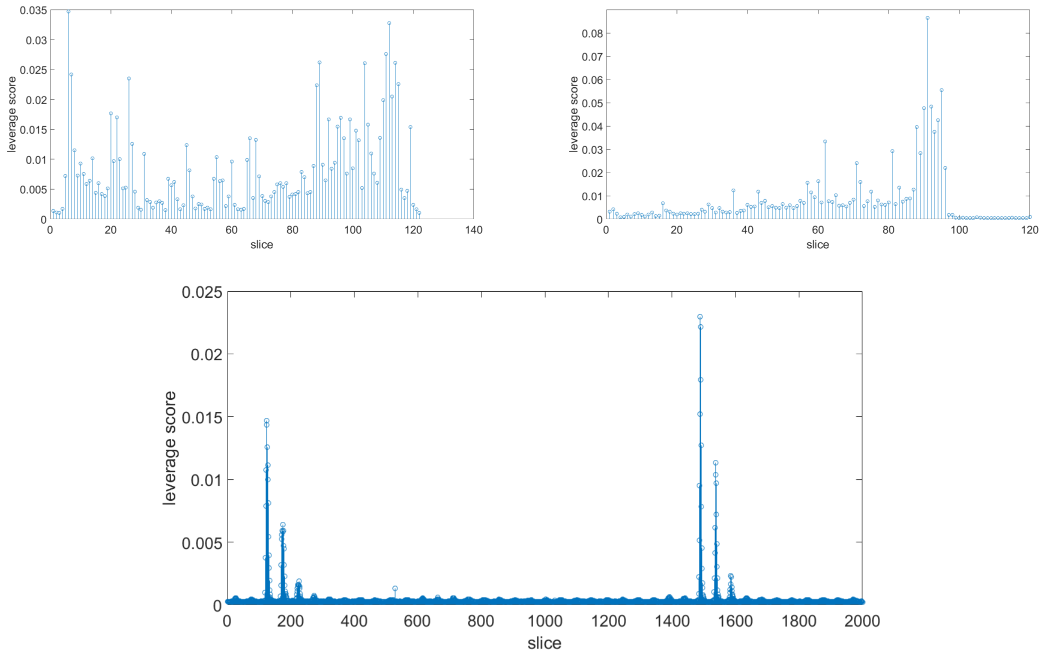


**Figure 5.2.** *Distribution of leverage scores of tensor $\mathcal{X}$, relative to the best rank-5 space for the brain data set. Left: Horizontal scores. Right: Lateral scores. Bottom: Frontal scores.*

**Table 5.2**

*RSE of matrix and tensor low rank decomposition relative to the best rank-5 space for identifying important ions in the brain data set.*

| Number of | CX | | | | t-CX | | | |
|---|---|---|---|---|---|---|---|---|
| selected slices | Randomized | | Deterministic | | Randomized | | Deterministic | |
| | RSE | Time | RSE | Time | RSE | Time | RSE | Time |
| 25 | 19.13e-2 | 4.56 | 18.84e-2 | 8.46 | 13.65e-2 | 4.13 | 16.05e-2 | 4.47 |
| 35 | 17.24e-2 | 5.12 | 17.59e-2 | 8.95 | 17.43e-2 | 4.87 | 13.13e-2 | 5.99 |
| 45 | 16.35e-2 | 7.01 | 16.93e-2 | 14.99 | 11.32e-2 | 6.14 | 11.01e-2 | 7.01 |
| 55 | 15.14e-2 | 8.16 | 16.52e-2 | 15.86 | 16.26e-2 | 6.63 | 10.16e-2 | 8.99 |

Table 5.3

*RSE of matrix and tensor low rank decomposition relative to the best rank-5 space for finding important pixels in the brain data set.*

| Number of selected slices | CX | | | | t-CX | | | |
|---|---|---|---|---|---|---|---|---|
| | Randomized | | Deterministic | | Randomized | | Deterministic | |
| | RSE | Time | RSE | Time | RSE | Time | RSE | Time |
| 25 | 45.24e-2 | 3.15 | 46.03e-2 | 7.41 | 37.65e-2 | 3.01 | 16.05e-2 | 4.06 |
| 35 | 35.87e-2 | 3.81 | 35.68e-2 | 7.63 | 28.00e-2 | 3.19 | 13.13e-2 | 4.55 |
| 45 | 24.13e-2 | 5.29 | 24.19e-2 | 11.89 | 23.98e-2 | 4.15 | 21.01e-2 | 5.59 |
| 55 | 23.16e-2 | 5.83 | 24.39e-2 | 12.93 | 15.23e-2 | 4.23 | 15.17e-2 | 5.71 |

Table 5.4

*RSE of matrix and tensor low rank decomposition relative to the best rank-15 space for finding important ions and pixels in the lung data set.*

| Number of selected slices | Decomposition using ions | | | | Decomposition using ions and pixels | | | |
|---|---|---|---|---|---|---|---|---|
| | CX | | t-CX | | CUR | | t-CUR | |
| | RSE | Time | RSE | Time | RSE | Time | RSE | Time |
| 25 | 40.82e-2 | 3.31 | 46.89e-2 | 3.05 | 59.76e-2 | 2.89 | 24.67e-2 | 2.14 |
| 35 | 41.98e-2 | 4.13 | 29.63e-2 | 3.24 | 42.57e-2 | 3.16 | 22.71e-2 | 2.67 |
| 45 | 30.16e-2 | 5.78 | 22.35e-2 | 3.36 | 12.18e-2 | 4.25 | 18.15e-2 | 3.13 |
| 55 | 30.74e-2 | 6.67 | 20.81e-2 | 4.05 | 19.00e-2 | 5.47 | 18.15e-2 | 3.26 |

and then averaging them. Note that for pixel selection, the deterministic CX decomposition results in larger reconstruction errors than its randomized CX counterpart. The reason for this behavior lies in the distribution of the leverage scores for the pixels, which are fairly uniform (see Figure 5.2).

For the lung data set, reconstruction errors to the best rank $r = 15$ approximation based on randomized t-CX and t-CUR decompositions are given in Table 5.4 based on averages over 10 independent replicates. It can be seen that the t-CX and t-CUR match or outperform their matrix variants in terms of accuracy, while improving on computing time.

These results introduce the concept of t-CX/ t-CUR tensor factorizations to MSI, describing their utility and illustrating principled algorithmic approaches to deal with the overwhelming amount of data generated by this technology and their ability to select important and intepretable ions/pixels.

**5.3. Robust PCA (RPCA) in the fully and partially observed settings.** Many images exhibit an inherent low rank structure and are suitably denoised by low-rank modeling methods, such as RPCA [6]. In this section, we assess the performance of the CUR t-NN on two popular data sets and for the typical use cases they represent. We compare the performance of the proposed CUR t-NN algorithm to the following techniques:

- **EXACT NN**, the exact matrix completion [7];
- **RPCA NN**, the robust matrix completion [6];
- **E-TUCKER NN**, the TUCKER based tensor completion [33];
- **R-TUCKER NN**, the robust TUCKER based tensor completion [20];
- **EXACT t-NN**, the t-SVD based tensor completion [53];
- **RPCA t-NN**, the robust t-SVD based tensor completion [35].

All algorithms are terminated either when the relative square error (RSE),

$$\text{RSE} := \frac{\|\mathcal{L}^* - \tilde{\mathcal{L}}\|_F}{\|\mathcal{L}^*\|_F} \leq 10^{-3},$$

or the number of iterations and CPU times exceed 1,000 and 20 minutes, respectively.

**5.3.1. CUR t-NN on the Extended Yale B data set.** We apply the CUR t-NN on the Extended Yale B data set to evaluate the accuracy of the proposed low-rank representations, as well as its computation time. The database consists of 2,432 images of 38 individuals, each under 64 different lighting conditions [17]. We used 30 images from each subject and kept them at full resolution. Each image comprises of $192 \times 168$ pixels on a grayscale range. The data are organized into a $192 \times 1140 \times 168$ tensor that exhibits low tubal rank, which is an expected feature due to the spatial correlation within lateral slices [18]. Laplacian (salt and pepper[3]) noise was introduced separately in all frontal slices of the observation tensor for 20% of the entries.

We provide visualizations of the reconstructed first image from the first subject at the 20% noise level in Figure 5.3. We compare the performance of the CUR t-NN, RPCA t-NN, R-TUCKER NN, and RPCA NN for solving the robust tensor low rank approximation problem (4.5). Table 5.5 shows the accuracy of the CUR t-NN together with that of competing algorithms for the Extended Yale B data set. As shown in Figure 5.3 and Table 5.5, CUR t-NN estimates nearly the same face model as the RPCA t-NN requiring only a small fraction of time. On the other hand, all algorithms exhibit small RSE, but the CUR t-NN proves essentially as competitive as the best method RPCA t-NN, but achieves almost the same performance at approximately 1/4 of the time.

Beyond just speed-ups and/or accuracy improvements of the CUR t-NN algorithm, its output can be directly used in place of the singular slices and tubes that standard methods provide. The latter represent linear combinations of the slices of the tensor, which for an image data set capture an "average eigenface." On the other hand, CUR t-NN reconstructs the tensor through selection of real faces in the data set, thus giving the opportunity for researchers to inspect them and examine their representativeness. Hence, similar to the original CUR decomposition of matrices, CUR t-NN enhances the interpretability of the tensor decomposition [18, 48, 41].

**5.3.2. CUR t-NN on a video data set.** Next, we compare the CUR t-NN to the aforementioned listed competing methods for video data representation and compression from randomly missing entries. The video data, henceforth referred to as the basketball video (source: YouTube) is mapped to a $144 \times 256 \times 80$ tensor, obtained from with a nonstationary panning camera moving from left to right horizontally following the running players. We randomly sampled 50% entries from the basketball video (Figure 5.4). We compare the performance of the CUR t-NN, EXACT t-NN, E-TUCKER NN, and EXACT NN for solving the tensor completion problem (2.9).

[3]This noise can be caused by sharp and sudden disturbances in the image signal. It demonstrates itself as sparsely occurring white and black pixels.

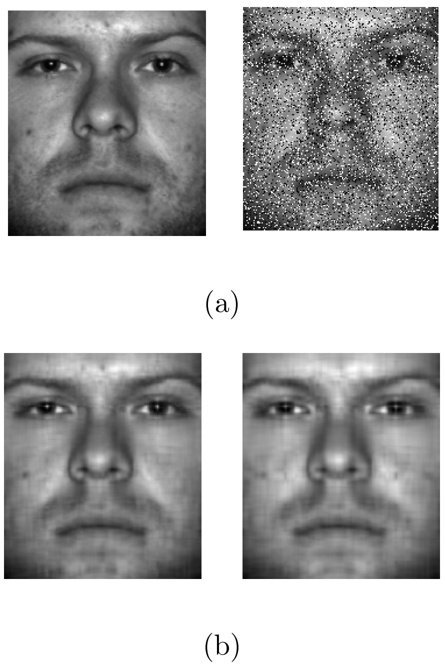


**Figure 5.3.** *The first frame of the noisy tensor factorization result for the Extended Yale B data set.* (a) *Left: The original frame.* (a) *Right: Noisy image (20% pixels corrupted).* (b) *Left: RPCA t-NN* [35]. (b) *Right: CUR t-NN.*

**Table 5.5**
*RSE of tensor robust completion methods on Extended Yale B dataset.*

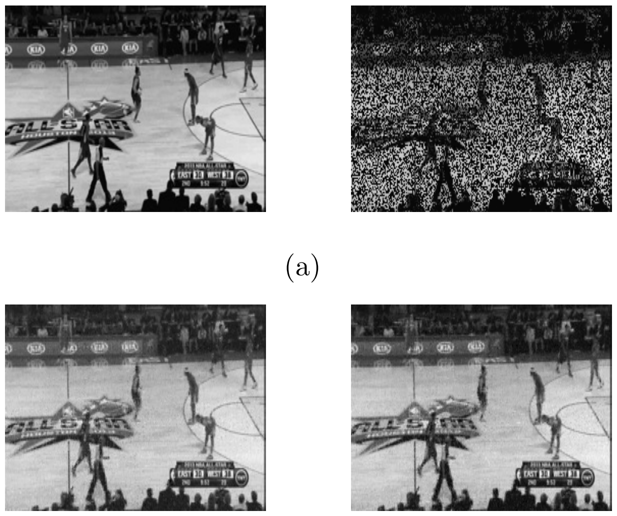
| **Robust completion approach** | **RSE** | **Time(s)** |
|---|---|---|
| `RPCA NN` [6] | 0.0056 | 417 |
| `R-TUCKER NN` [20] | 0.0034 | 513 |
| `RPCA t-NN` [35] | 0.0021 | 495 |
| **CUR t-NN** | **0.0026** | **136** |

The result is shown in Table 5.6. It can be seen that the CUR t-NN outperforms almost all its competitors in terms of CPU running time and accuracy and essentially matches that of EXACT t-NN.

## 6. Conclusion.

This paper introduced two randomized algorithms for basic tensor operations—rt-product and rt-project—and then used in tensor CX and CUR decomposition algorithms, whose aim is to select informative slices. The randomized tensor operations together with the tensor decompositions algorithms comes with relative error recovery guarantees. These algorithms can be effectively used in the analysis of large scale tensors with small tubal rank.

In addition, we proposed the CUR t-NN algorithm that exploits the advantages of randomization for dimensionality reduction and can be used effectively for large scale noisy tensor decompositions. Indeed, CUR t-NN uses an adaptive technique to sample slices of the tensor based on a *leverage score* for them and subsequently solves a convex optimization problem



(a)

(b)

**Figure 5.4.** *The 20th frame of the tensor completion result for the basketball video.* (a) *Left: The original video.* (a) *Right: Sampled video (50% sampling rate).* (b) *Left: EXACT t-NN based reconstruction* [53]. (b) *Right: CUR t-NN based reconstruction.*

**Table 5.6**
*RSE of tensor completion results for the basketball video.*

| Completion approach | RSE | Time(s) |
|---|---|---|
| `EXACT NN` [7] | 0.1001 | 687 |
| `E-TUCKER NN` [33] | 0.0900 | 718 |
| `EXACT t-NN` [53] | 0.0715 | 695 |
| **CUR t-NN** | **0.0850** | **205** |

followed by a projection step to recover a low rank approximation to the underlying tensor. The proposed algorithm has linear running time, and provably maintains the recovery guarantees of the exact algorithm with full data tensor.

## Appendix A.

**A.1. Proof of Theorem 3.1.** Let $F_{n_3}$ denote the discrete Fourier matrix. Then, Definition 2.1 implies that

$$\begin{aligned}(F_{n_3}\otimes I)\mathrm{circ}(\boldsymbol{\mathcal{A}})(F_{n_3}^{-1}\otimes I)(F_{n_3}\otimes I)\mathrm{unfold}(\boldsymbol{\mathcal{A}}^\top) &= \begin{pmatrix}\hat{\boldsymbol{\mathcal{A}}}_{::1}\hat{\boldsymbol{\mathcal{A}}}_{::1}^\top & & & \\ & \hat{\boldsymbol{\mathcal{A}}}_{::2}\hat{\boldsymbol{\mathcal{A}}}_{::2}^\top & & \\ & & \ddots & \\ & & & \hat{\boldsymbol{\mathcal{A}}}_{::n_3}\hat{\boldsymbol{\mathcal{A}}}_{::n_3}^\top\end{pmatrix}\\ &= \mathrm{bdiag}_{k\in[n_3]}(\hat{\boldsymbol{\mathcal{A}}}_{::k}\hat{\boldsymbol{\mathcal{A}}}_{::k}^\top). \qquad \text{(A.1)}\end{aligned}$$

Using (A.1) and the unitary invariance of the Frobenius norm, we have

$$\begin{aligned}\|\boldsymbol{\mathcal{A}} * \boldsymbol{\mathcal{B}} - \boldsymbol{\mathcal{C}} * \boldsymbol{\mathcal{R}}\|_F^2 &= \|(F_{n_3} \otimes I)\text{circ}(\boldsymbol{\mathcal{A}})(F_{n_3}^{-1} \otimes I)(F_{n_3} \otimes I)\text{unfold}(\boldsymbol{\mathcal{B}}) \\ &\quad - (F_{n_3} \otimes I)\text{circ}(\boldsymbol{\mathcal{C}})(F_{n_3}^{-1} \otimes I)(F_{n_3} \otimes I)\text{unfold}(\boldsymbol{\mathcal{R}})\|_F^2 \\ &= \frac{1}{n_3}\| \,\text{bdiag}_{k\in n_3}(\hat{\boldsymbol{\mathcal{A}}}_{::k}\hat{\boldsymbol{\mathcal{B}}}_{::k} - \hat{\boldsymbol{\mathcal{C}}}_{::k}\hat{\boldsymbol{\mathcal{R}}}_{::k})\|_F^2 \\ &= \frac{1}{n_3}\sum_{k=1}^{n_3} \|\hat{\boldsymbol{\mathcal{A}}}_{::k}\hat{\boldsymbol{\mathcal{B}}}_{::k} - \hat{\boldsymbol{\mathcal{C}}}_{::k}\hat{\boldsymbol{\mathcal{R}}}_{::k}\|_F^2 \\ &= \frac{1}{n_3}\sum_{k=1}^{n_3} \|\hat{\boldsymbol{\mathcal{A}}}_{::k}\hat{\boldsymbol{\mathcal{B}}}_{::k} - \hat{\boldsymbol{\mathcal{A}}}_{::k}\hat{\boldsymbol{\mathcal{S}}}_{::k}\hat{\boldsymbol{\mathcal{D}}}_{::k}\hat{\boldsymbol{\mathcal{D}}}_{::k}\hat{\boldsymbol{\mathcal{S}}}_{::k}^\top\hat{\boldsymbol{\mathcal{B}}}_{::k}\|_F^2. \end{aligned} \tag{A.2}$$

In Algorithm 3.1, we define $I_j$, $j \in [n_2]$ as an indicator variable, which is set to 1 if both the $j$th lateral slice of $\boldsymbol{\mathcal{A}}$ and the $j$th horizontal slice of $\boldsymbol{\mathcal{B}}$ are selected. In this case, the selected lateral and horizontal slices are scaled by score $1/\sqrt{\min\{1, cp_j\}}$. Note that if $\min\{1, cp_j\} = 1$, then $I_j = 1$ with probability 1 and $1 - I_j/\min\{1, cp_j\} = 0$. Then, taking expectation on both sides of (A.2) and considering the set $\Upsilon = \{j \in [n_2] : cp_j < 1\} \subseteq [n_2]$, we get

$$\begin{aligned}\mathbf{E}\left[\|\boldsymbol{\mathcal{A}} * \boldsymbol{\mathcal{B}} - \boldsymbol{\mathcal{C}} * \boldsymbol{\mathcal{R}}\|_F^2\right] &= \frac{1}{n_3}\sum_{k=1}^{n_3}\mathbf{E}\left[\|\sum_{j\in\Upsilon}\left(1-\frac{I_j}{cp_j}\right)\hat{\boldsymbol{\mathcal{A}}}_{:jk}\hat{\boldsymbol{\mathcal{B}}}_{j:k}\|_F^2\right] \\ &= \frac{1}{n_3}\sum_{k=1}^{n_3}\mathbf{E}\left[\sum_{i_1=1}^{n_1}\sum_{i_2=1}^{n_2}\left(\sum_{j\in\Upsilon}\left(1-\frac{I_j}{cp_j}\right)\hat{\boldsymbol{\mathcal{A}}}_{:jk}\hat{\boldsymbol{\mathcal{B}}}_{j:k}\right)_{i_1i_2}^2\right] \\ &= \frac{1}{n_3}\sum_{k=1}^{n_3}\mathbf{E}\left[\sum_{i_1=1}^{n_1}\sum_{i_2=1}^{n_2}\left(\sum_{j\in\Upsilon}\left(1-\frac{I_j}{cp_j}\right)\hat{\boldsymbol{\mathcal{A}}}_{i_1jk}\hat{\boldsymbol{\mathcal{B}}}_{ji_2k}\right)^2\right] \end{aligned}$$

$$\begin{aligned} &= \frac{1}{n_3}\sum_{k=1}^{n_3}\mathbf{E}\left[\sum_{i_1=1}^{n_1}\sum_{i_2=1}^{n_2}\sum_{j_1\in\Upsilon}\sum_{j_2\in\Upsilon}\hat{p}_{i_1i_2j_1j_2}\right] \\ &= \frac{1}{n_3}\sum_{k=1}^{n_3}\sum_{i_1=1}^{n_1}\sum_{i_2=1}^{n_2}\sum_{j_1\in\Upsilon}\sum_{j_2\in\Upsilon}\mathbf{E}\left[\hat{p}_{i_1i_2j_1j_2}\right], \end{aligned} \tag{A.3}$$

where $\hat{p}_{i_1i_2j_1j_2} = (1-\frac{I_{j_1}}{cp_{j_1}})(1-\frac{I_{j_2}}{cp_{j_2}})\hat{\boldsymbol{\mathcal{A}}}_{i_1j_1k}\hat{\boldsymbol{\mathcal{B}}}_{j_1i_2k}\hat{\boldsymbol{\mathcal{A}}}_{i_1j_2k}\hat{\boldsymbol{\mathcal{B}}}_{j_2i_2k}$.

Since for $j \in [\Upsilon]$, $\mathbf{E}[1 - I_j/cp_j] = 0$ and $\mathbf{E}[(1 - I_j/cp_j)^2] = (1/cp_j) - 1 \le 1/cp_j$, we get

$$\begin{aligned}\mathbf{E}\left[\|\boldsymbol{\mathcal{A}} * \boldsymbol{\mathcal{B}} - \boldsymbol{\mathcal{C}} * \boldsymbol{\mathcal{R}}\|_F^2\right] &= \frac{1}{n_3}\sum_{k=1}^{n_3}\sum_{i_1=1}^{n_1}\sum_{i_2=1}^{n_2}\sum_{j\in\Upsilon}\mathbf{E}(1-I_j/cp_j)^2\hat{\boldsymbol{\mathcal{A}}}_{i_1jk}^2\hat{\boldsymbol{\mathcal{B}}}_{ji_2k}^2 \\ &\le \frac{1}{n_3}\sum_{k=1}^{n_3}\sum_{j\in\Upsilon}\frac{1}{cp_j}\sum_{i_1=1}^{n_1}\sum_{i_2=1}^{n_2}\hat{\boldsymbol{\mathcal{A}}}_{i_1jk}^2\hat{\boldsymbol{\mathcal{B}}}_{ji_2k}^2 \\ &= \frac{1}{n_3}\sum_{k=1}^{n_3}\frac{1}{c}\sum_{j\in\Upsilon}\frac{\|\hat{\boldsymbol{\mathcal{A}}}_{:jk}\|_2^2\|\hat{\boldsymbol{\mathcal{B}}}_{j:k}\|_2^2}{p_j}\end{aligned}$$

$$= \frac{1}{c}\sum_{j\in\Upsilon}\frac{\|\boldsymbol{\mathcal{A}}_{:j1}\|_2^2\|\boldsymbol{\mathcal{B}}_{j:1}\|_2^2}{p_j} + \cdots + \frac{1}{c}\sum_{j\in\Upsilon}\frac{\|\boldsymbol{\mathcal{A}}_{:jn_3}\|_2^2\|\boldsymbol{\mathcal{B}}_{j:n_3}\|_2^2}{p_j}$$

$$= \frac{1}{c}\sum_{j\in\Upsilon}\frac{\|\boldsymbol{\mathcal{A}}_{:j:}\|_F^2\|\boldsymbol{\mathcal{B}}_{j::}\|_F^2}{p_j}. \tag{A.4}$$

Equation (3.5) follows from (A.4) by using Jensen's inequality and the fact that the sampling probabilities (3.3) and (3.4) are defined in the original domain.

**A.2. Proof of Lemma 3.2.** Using Definitions 2.1 and 2.9, we have that

$$\begin{aligned}\|\boldsymbol{\mathcal{A}} * \boldsymbol{\mathcal{A}}^\top - \boldsymbol{\mathcal{C}} * \boldsymbol{\mathcal{C}}\| &= \|(F_{n_3}\otimes I)\mathrm{circ}(\boldsymbol{\mathcal{A}})(F_{n_3}^{-1}\otimes I)(F_{n_3}\otimes I)\mathrm{unfold}(\boldsymbol{\mathcal{A}}^\top)\\ &\quad - (F_{n_3}\otimes I)\mathrm{circ}(\boldsymbol{\mathcal{C}})(F_{n_3}^{-1}\otimes I)(F_{n_3}\otimes I)\mathrm{unfold}(\boldsymbol{\mathcal{C}}^\top)\|_2\\ &= \|\,\mathrm{bdiag}_{k\in n_3}(\hat{\boldsymbol{\mathcal{A}}}_{::k}\hat{\boldsymbol{\mathcal{A}}}_{::k}^\top - \hat{\boldsymbol{\mathcal{C}}}_{::k}\hat{\boldsymbol{\mathcal{C}}}_{::k}^\top)\|_2\\ &= \max_{k\in[n_3]}\{\|\hat{\boldsymbol{\mathcal{A}}}_{::k}\hat{\boldsymbol{\mathcal{A}}}_{::k}^\top - \hat{\boldsymbol{\mathcal{C}}}_{::k}\hat{\boldsymbol{\mathcal{C}}}_{::k}^\top\|_2\},\end{aligned} \tag{A.5}$$

where the first equality follows from the unitary invariance of the spectral norm, the second equality from (A.1), and the third equality follows since the spectral norm of a block matrix is equal to the maximum of block norms.

Let $k_\pi = k \in [n_3]$ be the index of the tensor's frontal slice with maximum spectral norm $\|\hat{\boldsymbol{\mathcal{A}}}_{::k}\hat{\boldsymbol{\mathcal{A}}}_{::k}^\top - \hat{\boldsymbol{\mathcal{C}}}_{::k}\hat{\boldsymbol{\mathcal{C}}}_{::k}^\top\|_2$. Using (A.5) and Example 4.3 in [19], we obtain

$$\|\hat{\boldsymbol{\mathcal{A}}}_{::k_\pi}\hat{\boldsymbol{\mathcal{A}}}_{::k_\pi}^\top - \hat{\boldsymbol{\mathcal{C}}}_{::k_\pi}\hat{\boldsymbol{\mathcal{C}}}_{::k_\pi}^\top\|_2 \le \frac{\epsilon}{2}\|\hat{\boldsymbol{\mathcal{A}}}_{::,k_\pi}\|_2^2 \tag{A.6}$$

with probability at least $1-\delta$.

**A.3. Proof of Lemma 3.5.** Let $\sigma_{i,k}$ be the $i$th largest singular value of slice $k$. For all $1 \le i \le r$ and $1 \le k \le n_3$, we have

$$\begin{aligned}|1 - \max_{k\in[n_3]}\sigma_{i,k}^2(\boldsymbol{\mathcal{V}}^\top * \boldsymbol{\mathcal{S}} * \boldsymbol{\mathcal{D}})| &= |\max_{k\in[n_3]}\sigma_{i,k}(\boldsymbol{\mathcal{V}}^\top * \boldsymbol{\mathcal{V}}) - \max_{k\in[n_3]}\sigma_{i,k}(\boldsymbol{\mathcal{V}}^\top * \boldsymbol{\mathcal{S}} * \boldsymbol{\mathcal{D}} * \boldsymbol{\mathcal{D}} * \boldsymbol{\mathcal{S}}^\top * \boldsymbol{\mathcal{V}})|\\ &\le \|\boldsymbol{\mathcal{V}} * \boldsymbol{\mathcal{V}}^\top - \boldsymbol{\mathcal{V}}^\top * \boldsymbol{\mathcal{S}} * \boldsymbol{\mathcal{D}} * \boldsymbol{\mathcal{D}} * \boldsymbol{\mathcal{S}}^\top * \boldsymbol{\mathcal{V}}\|_2.\end{aligned} \tag{A.7}$$

Using Lemma 3.2 and (A.7), we get

$$|1 - \max_{k\in[n_3]}\sigma_{i,k}^2(\boldsymbol{\mathcal{V}}^\top * \boldsymbol{\mathcal{S}} * \boldsymbol{\mathcal{D}})| \le \epsilon/2\|\hat{\boldsymbol{\mathcal{V}}}_{::k_\pi}\|_2^2 \tag{A.8}$$

for all $1 \le i \le r$ and an index $k_\pi \in [n_3]$.

Since $\epsilon \in (0,1]$, each tubal singular value of $\boldsymbol{\mathcal{V}}^\top * \boldsymbol{\mathcal{S}}$ is positive, which implies that $\mathrm{rank}_t(\boldsymbol{\mathcal{V}}^\top * \boldsymbol{\mathcal{S}}) = \mathrm{rank}_t(\boldsymbol{\mathcal{V}}) = \mathrm{rank}_t(\boldsymbol{\mathcal{L}})$.

To prove (3.10b), we use the t-SVD of $\mathcal{V}^\top * \mathcal{S} * \mathcal{D}$ and note that

$$\begin{aligned}
\|\Omega\| &= \|\left(\mathcal{V}^\top * \mathcal{S} * \mathcal{D}\right)^\dagger - \left(\mathcal{V}^\top * \mathcal{S} * \mathcal{D}\right)^\top\| \\
&= \|\left(\mathcal{U}_{\mathcal{V}^\top * \mathcal{S} * \mathcal{D}} \Sigma_{\mathcal{V}^\top * \mathcal{S} * \mathcal{D}} \mathcal{V}^\top_{\mathcal{V}^\top * \mathcal{S} * \mathcal{D}}\right)^\dagger - \left(\mathcal{U}_{\mathcal{V}^\top * \mathcal{S} * \mathcal{D}} \Sigma_{\mathcal{V}^\top * \mathcal{S} * \mathcal{D}} \mathcal{V}^\top_{\mathcal{V}^\top * \mathcal{S} * \mathcal{D}}\right)^\top\| \\
&= \|\mathcal{V}_{\mathcal{V}^\top * \mathcal{S} * \mathcal{D}} \left(\Sigma^{-1}_{\mathcal{V}^\top * \mathcal{S} * \mathcal{D}} - \Sigma_{\mathcal{V}^\top * \mathcal{S} * \mathcal{D}}\right) \mathcal{U}^\top_{\mathcal{V}^\top * \mathcal{S} * \mathcal{D}}\| \\
&= \|\left(\Sigma^{-1}_{\mathcal{V}^\top * \mathcal{S} * \mathcal{D}} - \Sigma_{\mathcal{V}^\top * \mathcal{S} * \mathcal{D}}\right)\|. \qquad \text{(A.9)}
\end{aligned}$$

The claim follows since $\mathcal{V}_{\mathcal{V}^\top * \mathcal{S} * \mathcal{D}}$ and $\mathcal{U}_{\mathcal{V}^\top * \mathcal{S} * \mathcal{D}}$ are orthogonal tensors.

To prove (3.10c), note that

$$\begin{aligned}
(\mathcal{L} * \mathcal{S} * \mathcal{D})^\dagger &= \left(\mathcal{U} * \Sigma * \mathcal{V}^\top * \mathcal{S} * \mathcal{D}\right)^\dagger \\
&= \left(\mathcal{U} * \Sigma * \mathcal{U}_{\mathcal{V}^\top * \mathcal{S} * \mathcal{D}} * \Sigma_{\mathcal{V}^\top * \mathcal{S} * \mathcal{D}} * \mathcal{V}^\top_{\mathcal{V}^\top * \mathcal{S} * \mathcal{D}}\right)^\dagger \\
&= \mathcal{V}_{\mathcal{V}^\top * \mathcal{S} * \mathcal{D}} * \left(\Sigma * \mathcal{U}_{\mathcal{V}^\top * \mathcal{S} * \mathcal{D}} * \Sigma_{\mathcal{V}^\top * \mathcal{S} * \mathcal{D}}\right)^\dagger * \mathcal{U}^\top. \qquad \text{(A.10)}
\end{aligned}$$

To remove the pseudoinverse in the above derivations, we use the first part of the lemma. In this case,

$$\begin{aligned}
\left(\Sigma * \mathcal{U}_{\mathcal{V}^\top * \mathcal{S} * \mathcal{D}} * \Sigma_{\mathcal{V}^\top * \mathcal{S} * \mathcal{D}}\right)^\dagger &= \left(\Sigma * \mathcal{U}_{\mathcal{V}^\top * \mathcal{S} * \mathcal{D}} * \Sigma_{\mathcal{V}^\top * \mathcal{S} * \mathcal{D}}\right)^{-1} \\
&= \Sigma^{-1}_{\mathcal{V}^\top * \mathcal{S} * \mathcal{D}} * \mathcal{U}^\top_{\mathcal{V}^\top * \mathcal{S} * \mathcal{D}} \Sigma^{-1}. \qquad \text{(A.11)}
\end{aligned}$$

By combining (A.10) and (A.11), we get the result.

To prove (3.10d), we have that for all $1 \le i \le r$ and $1 \le k \le n_3$,

$$\begin{aligned}
\|\Omega\| &= \|\Sigma^{-1}_{\mathcal{V}^\top * \mathcal{S} * \mathcal{D}} - \Sigma_{\mathcal{V}^\top * \mathcal{S} * \mathcal{D}}\| && \text{from (A.9)} \\
&= \max_{i,k} \left|\sigma_{i,k}(\mathcal{V}^\top * \mathcal{S} * \mathcal{D}) - \frac{1}{\sigma_{i,k}(\mathcal{V}^\top * \mathcal{S} * \mathcal{D})}\right| && \text{by definition} \\
&= \max_{i,k} \frac{|\sigma^2_{i,k}(\mathcal{V}^\top * \mathcal{S} * \mathcal{D}) - 1|}{|\sigma_{i,k}(\mathcal{V}^\top * \mathcal{S} * \mathcal{D})|} && \text{by simple manipulation} \\
&\le \frac{\epsilon/2}{\sqrt{1-\epsilon/2}} && \text{from (A.7)} \\
&\le \epsilon/\sqrt{2} && \text{if } \epsilon < 1 \Rightarrow \sqrt{1-\epsilon/2} > 1/\sqrt{2}.
\end{aligned}$$

**A.4. Proof of Theorem 3.7.** From Definition 2.14 and using (3.9) with $\beta = n_3/\mu_0(\mathcal{V}) \in (0, 1]$, we have that

$$\frac{\beta}{rn_3}\|\hat{\mathcal{V}}_{i::}\|_F^2 = \frac{\beta}{r}\|\mathcal{V}_{i::}\|_F^2 \le \frac{\beta}{r}\frac{r}{n_2 n_3}\mu_0(\mathcal{V}) = \frac{1}{n_2} = p_i$$

for all $i \in [n_2]$.

Using Proposition 3.6, our choice of $c$ implies that

$$\begin{aligned}\|\mathcal{A} - \mathcal{A}_c * \mathcal{L}_c^\dagger * \mathcal{L}\|_F &= \|\mathcal{A} - \mathcal{A} * \mathcal{C} * \mathcal{D} * (\mathcal{L}_c * \mathcal{D})^\dagger * \mathcal{L}\|_F \\ &\leq (1+\epsilon)\|\mathcal{A} - \mathcal{A} * \mathcal{L}^\dagger * \mathcal{L}\|_F \end{aligned} \tag{A.12}$$

holds with probability at least 0.85.

**A.5. Proof of Corollary 3.8.** Since $\mathcal{C}^+ * \mathcal{A}$ minimizes $\|\mathcal{A} - \mathcal{C} * \mathcal{X}\|_F$ over all tensors $\mathcal{X} \in \mathbb{R}^{n_1 \times c \times n_3}$, it follows that

$$\|\mathcal{A} - \mathcal{C} * \mathcal{C}^\dagger * \mathcal{A}\|_F \leq \|\mathcal{A} - \mathcal{C} * \mathcal{L}_{\mathcal{C}}^\dagger * \mathcal{L}\|_F.$$

Now, using (A.12), we get

$$\|\mathcal{A} - \mathcal{C} * \mathcal{C}^\dagger * \mathcal{A}\|_F \leq (1+\epsilon)\|\mathcal{A} - \mathcal{L}\|_F \tag{A.13}$$

with probability at least 0.85.

We can trivially boost the success probability to $1-\delta$ by repeating Algorithm 3.3 $O(\log(1/\delta))$ rounds. Specifically, let $\mathcal{C}_i$ denote the output of Algorithm 3.3 at round $i$; using (A.13) for each $\mathcal{C}_i$ we have

$$\|\mathcal{A} - \mathcal{C}_i * \mathcal{C}_i^\dagger * \mathcal{A}\|_F \leq (1+\epsilon)\|\mathcal{A} - \mathcal{L}\|_F \tag{A.14}$$

with probability at least 0.85.

Now, let $\mathcal{C}$ denote the set of all columns used in the approximation. Since each $\mathcal{C}_i = \mathcal{C} * \mathcal{S}_i$ for some tensor $\mathcal{S}_i$ and $\mathcal{C}^+ * \mathcal{A}$ minimizes $\|\mathcal{A} - \mathcal{C} * \mathcal{X}\|_F$ over all tensors $\mathcal{X} \in \mathbb{R}^{n_1 \times c \times n_3}$, it follows that

$$\|\mathcal{A} - \mathcal{C} * \mathcal{C}^\dagger * \mathcal{A}\|_F \leq \|\mathcal{A} - \mathcal{C}_i * \mathcal{C}_i^\dagger * \mathcal{A}\|_F$$

for each $i$. Hence, if

$$\|\mathcal{A} - \mathcal{C} * \mathcal{C}^\dagger * \mathcal{A}\|_F \leq (1+\epsilon)\|\mathcal{A} - \mathcal{L}\|_F$$

fails to hold, then for each $i$, (A.14) also fails to hold. Since $0.15 < 1/e$, the desired conclusion must hold with probability at least $1 - (1/e)^{\log(1/\delta)} = 1 - \delta$.

**A.6. Proof of Corollary 3.10.** Using (A.12), it follows that

$$\|\mathcal{A} - \mathcal{C} * \mathcal{U} * \mathcal{R}\|_F \leq (1+\epsilon)\|\mathcal{A} - \mathcal{C} * \mathcal{C}^\dagger * \mathcal{A}\|_F \tag{A.15}$$

with probability at least 0.85.

Using (A.13) and (A.15), our choice of $c$ and $l$ implies the following holds with probabilities at least 0.7:

$$\|\mathcal{A} - \mathcal{C} * \mathcal{U} * \mathcal{R}\|_F \leq (1+\epsilon)^2\|\mathcal{A} - \mathcal{L}\|_F \leq (1+\epsilon')\|\mathcal{A} - \mathcal{L}\|_F, \tag{A.16}$$

where $\epsilon' = 3\epsilon$.

The inequality (A.16) holds with probability at least $1-\delta$ by following the boosting strategy employed in the proof of Corollary 3.8. Indeed, since in each trial inequality (A.16) fails with probability less than $0.3 < 1/e$, the claim of Corollary 3.10 will hold with probability greater than $1 - (1/e)^{\log(1/\delta)} = 1 - \delta$.

**A.7. Proof of Lemma 4.1.** Since from Lemma 3.5 we have $\text{rank}_t(\mathcal{L}_c) = \text{rank}_t(\mathcal{L})$, the first claim follows similarly by using Lemma 1 of [38].

To prove the second claim, using Lemma 3.5, assume that $\mathcal{S}^\top * \mathcal{V}$ consists of the first $c$ horizontal slices of $\mathcal{V}$. Then, if $\mathcal{L}_c = \mathcal{U} * \Sigma * \mathcal{V}^\top * \mathcal{S}$ has $\text{rank}_t(\mathcal{L}_c) = \text{rank}_t(\mathcal{L}) = r$, the tensor $\mathcal{V}^\top * \mathcal{S}$ must have full tubal rank. Thus, we can write

$$\begin{aligned}
\mathcal{L}_c^+ * \mathcal{L}_c &= (\mathcal{U} * \Sigma * \mathcal{V}^\top * \mathcal{S})^+ * \mathcal{U} * \Sigma * \mathcal{V}^\top * \mathcal{S} \\
&= (\Sigma * \mathcal{V}^\top * \mathcal{S})^+ * \mathcal{U}^\top * \mathcal{U} * \Sigma * \mathcal{V}^\top * \mathcal{S} \\
&= (\Sigma * \mathcal{V}^\top * \mathcal{S})^+ * \Sigma * \mathcal{V}^\top * \mathcal{S} \\
&= (\mathcal{V}^\top * \mathcal{S})^+ * \Sigma^\top * \Sigma * \mathcal{V}^\top * \mathcal{S} \\
&= (\mathcal{V}^\top * \mathcal{S})^+ * \mathcal{V}^\top * \mathcal{S} \\
&= \mathcal{V}^\top * \mathcal{S} * (\mathcal{V}^\top * \mathcal{S} * \mathcal{S}^\top * \mathcal{V})^{-1} \mathcal{V}^\top * \mathcal{S},
\end{aligned}$$

where the second and third equalities follow from $\mathcal{U}^\top * \mathcal{U} = \mathcal{I}_c$. The fourth and fifth equalities result from $\Sigma$ having full tubal rank and $\mathcal{V}$ having full lateral slice rank, and the sixth follows from $\mathcal{S}^\top * \mathcal{V}$ having full horizontal slice rank. Next, denote the right singular slices of $\mathcal{L}_c$ by $\mathcal{V}_c \in \mathbb{R}^{c \times r \times n_3}$. Define $\mathring{\mathbf{e}}_{i,c}$ as the $i$th lateral slice of $\mathcal{I}_c$ and $\mathring{\mathbf{e}}_{i,n_2}$ as the $i$th lateral slice of $\mathcal{I}$. Then we have

$$\begin{aligned}
\mu_0(\mathcal{V}_c) &= \frac{cn_3}{r} \max_{1 \le i \le c} \|\mathcal{V}_c^\top * \mathring{\mathbf{e}}_{i,c}\|_F^2 \\
&= \frac{cn_3}{r} \max_{1 \le i \le c} \text{trace}\{\mathring{\mathbf{e}}_{i,c}^\top * \mathcal{L}_c^+ * \mathcal{L}_c * \mathring{\mathbf{e}}_{i,c}\} \\
&= \frac{cn_3}{r} \max_{1 \le i \le c} \text{trace}\{\mathring{\mathbf{e}}_{i,c}^\top * (\mathcal{V}^\top * \mathcal{S})^+ * \mathcal{V}^\top * \mathcal{S} * \mathring{\mathbf{e}}_{i,c}\} \\
&= \frac{cn_3}{r} \max_{1 \le i \le c} \text{trace}\{\mathring{\mathbf{e}}_{i,c}^\top * \mathcal{S}^\top * \mathcal{V} * \mathcal{W}^{-1} * \mathcal{V}^\top * \mathcal{S} * \mathring{\mathbf{e}}_{i,c}\} \\
&= \frac{cn_3}{r} \max_{1 \le i \le c} \text{trace}\{\mathring{\mathbf{e}}_{i,n_2}^\top * \mathcal{V} * \mathcal{W}^{-1} * \mathcal{V}^\top * \mathring{\mathbf{e}}_{i,n_2}\},
\end{aligned}$$

where $\mathcal{W} = \mathcal{V}^\top * \mathcal{S} * \mathcal{S}^\top * \mathcal{V}$ and the final equality follows from $\mathcal{V}^\top * \mathcal{S} * \mathring{\mathbf{e}}_{i,c} = \mathcal{V}^\top * \mathring{\mathbf{e}}_{i,n_2}$ for all $1 \le i \le c$.

Next, we have

$$\begin{aligned}
\mu_0(\mathcal{V}_{\mathcal{L}_\mathfrak{C}}) &= \frac{cn_3}{r} \max_{1 \le i \le c} \text{trace}\{\mathring{\mathbf{e}}_{i,n_2}^\top * \mathcal{V} * \mathcal{W}^{-1} * \mathcal{V}^\top * \mathring{\mathbf{e}}_{i,n_2}\} \\
&= \frac{cn_3}{r} \max_{1 \le i \le c} \text{trace}\{\mathcal{W}^{-1} * \mathcal{V}^\top * \mathring{\mathbf{e}}_{i,n_2} * \mathring{\mathbf{e}}_{i,n_2}^\top * \mathcal{V}\} \\
&\le \frac{cn_3}{r} \|\mathcal{W}^{-1}\|^2 \max_{1 \le i \le c} \|\mathcal{V}^\top * \mathring{\mathbf{e}}_{i,n_2} * \mathring{\mathbf{e}}_{i,n_2}^\top * \mathcal{V}\|_{\circledast}^2,
\end{aligned}$$

where the last inequality follows form Hölder's inequality.

Since $\mathcal{V}^\top * \mathring{\mathbf{e}}_{i,n_2} * \mathring{\mathbf{e}}_{i,n_2}^\top * \mathcal{V}$ has tubal rank one, using the definition of $\mu_0$-coherence, we have

$$\mu_0(\mathcal{V}_{\mathcal{L}_\mathfrak{C}}) \le \frac{c}{n_2} \|\mathcal{W}^{-1}\|^2 \mu_0(\mathcal{V}).$$

Now, using (A.8), we have that $\|\mathcal{W}^{-1}\|^2 \leq \frac{n_2}{c(1-\epsilon/2)}$. Thus, it follows that

$$\mu_0(\mathcal{V}_{\mathcal{L}_\mathcal{C}}) \leq \mu_0(\mathcal{V})/(1-\epsilon/2).$$

To prove the last claim under Lemma 3.5, we note that

$$\begin{aligned}\mu_1(\mathcal{L}_c) &= \frac{n_1 c n_3^2}{r} \max_{\substack{1\leq i\leq n_1\\ 1\leq j\leq c}} \|\mathring{\mathfrak{e}}_{i,n_1}^\top * \mathcal{U}_c * \mathcal{V}_c^\top * \mathring{\mathfrak{e}}_{j,c}\|_F^2 \\ &\leq \frac{n_1 c n_3^2}{r} \max_{1\leq i\leq n_1} \|\mathcal{U}_c^\top * \mathring{\mathfrak{e}}_{i,n_1}\|_F^2 \max_{1\leq j\leq c} \|\mathcal{V}_c^\top * \mathring{\mathfrak{e}}_{j,c}\|_F^2 \\ &\leq \frac{r}{(1-\epsilon/2)} \mu_0(\mathcal{U})\mu_0(\mathcal{V}).\end{aligned}$$

## A.8. Proof of Theorem 4.2.

Define $A(\mathcal{X})$ as the event that a tensor $\mathcal{X}$ is $(\frac{r\mu^2}{1-\epsilon/2}, r)$-coherent. To prove Theorem 4.2, let $\tilde{\mathcal{L}}$ denote the solution obtained by CUR t-NN and let $\mathcal{L}^*$ be the exact solution of problems (2.9), (4.5), and (4.6). In Algorithm 4.1, define $\bar{\mathcal{L}}$ as

$$\bar{\mathcal{L}} = \begin{bmatrix} \tilde{\mathcal{C}}_1 & \tilde{\mathcal{R}}_2 \\ \tilde{\mathcal{C}}_2 & \mathcal{L}_{22}^* \end{bmatrix},$$

where $\tilde{\mathcal{C}} = \begin{bmatrix} \tilde{\mathcal{C}}_1 & \tilde{\mathcal{C}}_2 \end{bmatrix}^\top$, and $\tilde{\mathcal{R}} = \begin{bmatrix} \tilde{\mathcal{R}}_1 & \tilde{\mathcal{R}}_2 \end{bmatrix}$, and $\mathcal{L}_{22}^* \in \mathbb{R}^{(n_1-l)\times(n_2-c)\times n_3}$ is the bottom right subtensor of $\mathcal{L}^*$. It can easily be seen that

$$\|\mathcal{L}^* - \bar{\mathcal{L}}\|_F^2 \leq \|\mathcal{C}^* - \tilde{\mathcal{C}}\|_F^2 + \|\mathcal{R}^* - \tilde{\mathcal{R}}\|_F^2. \tag{A.17}$$

Now, define $W(\tilde{\mathcal{L}}, \bar{\mathcal{L}})$ as the event

$$\|\tilde{\mathcal{L}} - \bar{\mathcal{L}}\|_F \leq (1+\epsilon)\|\mathcal{L}^* - \bar{\mathcal{L}}\|_F. \tag{A.18}$$

If $W(\tilde{\mathcal{L}}, \bar{\mathcal{L}})$ holds, we have

$$\begin{aligned}\|\mathcal{L}^* - \tilde{\mathcal{L}}\|_F &\leq \|\mathcal{L}^* - \bar{\mathcal{L}}\|_F + \|\bar{\mathcal{L}} - \tilde{\mathcal{L}}\|_F && \text{by the triangle inequality} \\ &\leq \|\mathcal{L}^* - \bar{\mathcal{L}}\|_F + (1+\epsilon)\|\mathcal{L}^* - \bar{\mathcal{L}}\|_F && \text{from (A.18)} \\ &\leq (2+\epsilon)\|\mathcal{L}^* - \bar{\mathcal{L}}\|_F \\ &\leq (2+\epsilon)\sqrt{\|\mathcal{C}^* - \tilde{\mathcal{C}}\|_F^2 + \|\mathcal{R}^* - \tilde{\mathcal{R}}\|_F^2} && \text{from (A.17).}\end{aligned}$$

Next, we consider all three events $W(\tilde{\mathcal{L}}, \bar{\mathcal{L}})$, $A(\mathcal{R}^*)$, and $A(\mathcal{C}^*)$ with $\log(3/\delta)$. Using Lemma 4.1, our choice of $c$ and $l$ implies that $A(\mathcal{C}^*)$ and $A(\mathcal{R}^*)$ holds with probability at least $1-\delta/3$. Since $\tilde{\mathcal{L}}$ is a t-CUR approximation to $\bar{\mathcal{L}}$, from Corollary 3.10, we get that $W(\tilde{\mathcal{L}}, \bar{\mathcal{L}})$ holds with probability at least $1-\delta/3$.

Using the union bound, it follows that

$$\begin{aligned}\mathbf{Pr}(W(\tilde{\mathcal{L}}, \bar{\mathcal{L}}) \bigcap A(\mathcal{C}^*) \bigcap A(\mathcal{R}^*)) &\geq 1 - \mathbf{Pr}(W(\tilde{\mathcal{L}}, \bar{\mathcal{L}})^c) - \mathbf{Pr}(A(\mathcal{C}^*)^c) - \mathbf{Pr}(A(\mathcal{R}^*)^c) \\ &\geq 1 - \delta/3 - \delta/3 - \delta/3 \\ &= 1 - \delta.\end{aligned}$$

**A.9. Optimization by ADMM.** We provide the optimization and parameter setting details of ADMM used in Algorithm A.1 for solving a robust tensor factorization. As discussed in [54], the updates of $\mathcal{L}_{k+1}$ and $\mathcal{E}_{k+1}$ have closed form solutions. It is easy to see that the main per-iteration cost of Algorithm A.1 is in the update of $\mathcal{L}_{k+1}$, which requires computing the `fft` of $\mathcal{L}$ and the SVD of block matrix $\hat{L} = \texttt{fft}(\mathcal{L})$ in the Fourier domain.

```
Algorithm A.1. , $\mathcal{L} \leftarrow$ ADMM($\mathcal{X}, \lambda$).
Initialize: $\mathcal{L}_0 = \mathcal{E}_0 = \mathcal{Y}_0 = 0$, $\rho = 1.1$, $\mu_0 = 1e-3$, $\mu_{\max} = 1e+10$, $\epsilon = 1e-8$.
while not converged do
    • $\mathcal{L}_{k+1} \leftarrow \text{argmin}_{\mathcal{L}} \|\mathcal{L}\|_{\circledast} + \frac{\mu_k}{2}\|\mathcal{L} + \mathcal{E}_k - \mathcal{X} + \frac{\mathcal{Y}_k}{\mu_k}\|_F^2$;
    • $\mathcal{E}_{k+1} \leftarrow \text{argmin}_{\mathcal{E}} \lambda\|\mathcal{E}\|_1 + \frac{\mu_k}{2}\|\mathcal{L}_{k+1} + \mathcal{E} - \mathcal{X} + \frac{\mathcal{Y}_k}{\mu_k}\|_F^2$;
    • $\mathcal{Y}_{k+1} = \mathcal{Y}_k + \mu_k(\mathcal{L}_{k+1} + \mathcal{E}_{k+1} - \mathcal{X})$;
    • Update $\mu_{k+1}$ by $\mu_{k+1} = \min(\rho\mu_k, \mu_{\max})$;
    • Check $\|\mathcal{L}_{k+1} - \mathcal{L}_k\|_\infty \le \epsilon$, $\|\mathcal{E}_{k+1} - \mathcal{E}_k\|_\infty \le \epsilon$, $\|\mathcal{L}_{k+1} + \mathcal{E}_{k+1} - \mathcal{X}\|_\infty \le \epsilon$;
end while
```

The next result establishes the global convergence of the ADMM for solving problem (4.5) (for details on the convergence analysis, see [46]). Note that similar results hold for solving problems (2.9) and (4.6).

Theorem A.1. *The sequence* $(\mathcal{L}_k, \mathcal{E}_k)$ *generated by Algorithm* A.1 *from any starting point converges to a stationary point of problem* (4.5).

**Acknowledgments.** The authors would like to thank the Associate Editor and three anonymous referees for many constructive comments and suggestions that improved significantly the structure and readability of the paper.

## REFERENCES

[1] M. Bertalmio, G. Sapiro, V. Caselles, and C. Ballester, *Image inpainting*, in Proceedings of the 27th Annual Conference on Computer Graphics and Interactive Techniques, ACM Press/Addison-Wesley Publishing, New York, 2000, pp. 417–424.

[2] T. Bouwmans, A. Sobral, S. Javed, S. K. Jung, and E.-H. Zahzah, *Decomposition into low-rank plus additive matrices for background/foreground separation: A review for a comparative evaluation with a large-scale dataset*, preprint, arXiv:1511.01245 [cs.CV], (2015).

[3] S. Boyd, N. Parikh, E. Chu, B. Peleato, and J. Eckstein, *Distributed optimization and statistical learning via the alternating direction method of multipliers*, Found. Trends Mach. Learn., 3 (2011), pp. 1–122.

[4] K. Braman, *Third-order tensors as linear operators on a space of matrices*, Linear Algebra Appl., 433 (2010), pp. 1241–1253.

[5] C. F. Caiafa and A. Cichocki, *Generalizing the column–row matrix decomposition to multi-way arrays*, Linear Algebra Appl., 433 (2010), pp. 557–573.

[6] E. J. Candès, X. Li, Y. Ma, and J. Wright, *Robust principal component analysis?*, J. ACM, 58 (2011), p. 11.

[7] E. J. Candès and B. Recht, *Exact matrix completion via convex optimization*, Found. Comput. Math., 9 (2009), pp. 717–772.

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

[9] J. P. Costeira and T. Kanade, *A multibody factorization method for independently moving objects*, Int. J. Comput. Vis., 29 (1998), pp. 159–179.

[10] L. De Lathauwer, B. De Moor, and J. Vandewalle, *A multilinear singular value decomposition*, SIAM J. Matrix Anal. Appl., 21 (2000), pp. 1253–1278.

[11] L. De Lathauwer, B. De Moor, and J. Vandewalle, *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 (2000), pp. 1324–1342.

[12] A. Deshpande, L. Rademacher, S. Vempala, and G. Wang, *Matrix approximation and projective clustering via volume sampling*, in Proceedings of the Seventeenth Annual ACM-SIAM Symposium on Discrete Algorithm, SIAM, Philadelphia, 2006, pp. 1117–1126.

[13] P. Drineas, R. Kannan, and M. W. Mahoney, *Fast Monte Carlo algorithms for matrices* I: *Approximating matrix multiplication*, SIAM J. Comput., 36 (2006), pp. 132–157.

[14] P. Drineas, M. Magdon-Ismail, M. W. Mahoney, and D. P. Woodruff, *Fast approximation of matrix coherence and statistical leverage*, J. Mach. Learn. Res., 13 (2012), pp. 3475–3506.

[15] P. Drineas, M. W. Mahoney, and S. Muthukrishnan, *Relative-error CUR matrix decompositions*, SIAM J. Matrix Anal. Appl., 30 (2008), pp. 844–881.

[16] P. Foggia, G. Percannella, and M. Vento, *Graph matching and learning in pattern recognition in the last* 10 *years*, Int. J. Pattern Recognit. Artif. Intell., 28 (2014), 1450001.

[17] A. S. Georghiades, P. N. Belhumeur, and D. J. Kriegman, *From few to many: Illumination cone models for face recognition under variable lighting and pose*, IEEE Trans. Pattern Anal. Mach. Intell., 23 (2001), pp. 643–660.

[18] N. Hao, M. E. Kilmer, K. Braman, and R. C. Hoover, *Facial recognition using tensor-tensor decompositions*, SIAM J. Imaging Sci., 6 (2013), pp. 437–463.

[19] D. Hsu, S. Kakade, and T. Zhang, *Tail inequalities for sums of random matrices that depend on the intrinsic dimension*, Electron. Commun. Probab., 17 (2012).

[20] B. Huang, C. Mu, D. Goldfarb, and J. Wright, *Provable Low-Rank Tensor Recovery*, Optimization Online e-print, 4252, 2014.

[21] B. Jiang, F. Yang, and S. Zhang, *Tensor and its Tucker core: The invariance relationships*, Numer. Linear Algebra Appl., 24 (2017), e2086.

[22] I. Jolliffe, *Principal Component Analysis*, Springer, New York, 2002.

[23] D. Kalman, *A singularly valuable decomposition: The SVD of a matrix*, College Math. J., 27 (1996), pp. 2–23.

[24] E. Karahan, P. A. Rojas-Lopez, M. L. Bringas-Vega, P. A. Valdes-Hernandez, and P. A. Valdes-Sosa, *Tensor analysis and fusion of multimodal brain images*, Proc. IEEE, 103 (2015), pp. 1531–1559.

[25] E. Kernfeld, S. Aeron, and M. Kilmer, *Clustering multi-way data: A novel algebraic approach*, preprint, arXiv:1412.7056 [cs.LG], 2014.

[26] M. E. Kilmer, K. Braman, N. Hao, and R. C. Hoover, *Third-order tensors as operators on matrices: A theoretical and computational framework with applications in imaging*, SIAM J. Matrix Anal. Appl., 34 (2013), pp. 148–172.

[27] M. E. Kilmer and C. D. Martin, *Factorization strategies for third-order tensors*, Linear Algebra Appl., 435 (2011), pp. 641–658.

[28] T. G. Kolda and B. W. Bader, *Tensor decompositions and applications*, SIAM Rev., 51 (2009), pp. 455–500.

[29] T. G. Kolda, B. W. Bader, and J. P. Kenny, *Higher-order web link analysis using multilinear algebra*, Proceedings of the Fifth IEEE International Conference on in Data Mining, IEEE, New York, 2005, pp. 242–249.

[30] P. M. Kroonenberg, *Three-Mode Principal Component Analysis: Theory and Applications*, vol. 2, DSWO Press, Leiden, the Netherlands, 1983.

[31] F. G. Kuruvilla, P. J. Park, and S. L. Schreiber, *Vector algebra in the analysis of genome-wide expression data*, Genome Biol., 3 (2002), 0011.1.

[32] G. Liu, Z. Lin, S. Yan, J. Sun, Y. Yu, and Y. Ma, *Robust recovery of subspace structures by low-rank representation*, IEEE Trans. Pattern Anal. Mach. Intell., 35 (2013), pp. 171–184.

[33] J. Liu, P. Musialski, P. Wonka, and J. Ye, *Tensor completion for estimating missing values in visual data*, IEEE Trans. Pattern Anal. Mach. Intell., 35 (2013), pp. 208–220.

[34] R. Liu, Z. Lin, S. Wei, and Z. Su, *Solving Principal Component Pursuit in Linear Time via l_1 Filtering*, preprint, arXiv:1108.5359 [cs.NA], 2011.
[35] C. Lu, J. Feng, Y. Chen, W. Liu, Z. Lin, and S. Yan, *Tensor robust principal component analysis: Exact recovery of corrupted low-rank tensors via convex optimization*, in Proceedings of the 2016 IEEE Conference on Computer Vision and Pattern Recognition, (2016), pp. 5249–5257.
[36] L. W. Mackey, M. I. Jordan, and A. Talwalkar, *Divide-and-conquer matrix factorization*, in Proceedings of the Conference on Advances in Neural Information Processing Systems, 2011, pp. 1134–1142.
[37] M. W. Mahoney, M. Maggioni, and P. Drineas, *Tensor-CUR decompositions for tensor-based data*, SIAM J. Matrix Anal. Appl., 30 (2008), pp. 957–987.
[38] M. Mohri and A. Talwalkar, *Can matrix coherence be efficiently and accurately estimated?*, in Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics, 2011, pp. 534–542.
[39] Y. Mu, J. Dong, X. Yuan, and S. Yan, *Accelerated low-rank visual recovery by random projection*, in Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2011, pp. 2609–2616.
[40] A. Y. Ng, M. I. Jordan, and Y. Weiss, *On spectral clustering: Analysis and an algorithm*, in Proceedings of the Conference on Advances in Neural Information Processing Systems, pp. 849–856.
[41] F. Nie, H. Huang, X. Cai, and C. H. Ding, *Efficient and robust feature selection via joint* 2, 1*-norms minimization*, in Proceedings of the Conference on Advances in Neural Information Processing Systems, 2010, pp. 1813–1821.
[42] X. Piao, Y. Hu, J. Gao, Y. Sun, and Z. Lin, *A submodule clustering method for multi-way data by sparse and low-rank representation*, preprint, arXiv:1601.00149 [cs.CV], 2016.
[43] M. Rahmani and G. K. Atia, *Randomized robust subspace recovery and outlier detection for high dimensional data matrices*, IEEE Trans. Signal Process., 65, pp. 1580–1594.
[44] O. Semerci, N. Hao, M. E. Kilmer, and E. L. Miller, *Tensor-based formulation and nuclear norm regularization for multienergy computed tomography*, IEEE Trans. Image Process., 23 (2014), pp. 1678–1693.
[45] A. Smilde, R. Bro, and P. Geladi, *Multi-Way Analysis: Applications in the Chemical Sciences*, John Wiley & Sons, New York, 2005.
[46] D. A. Tarzanagh and G. Michailidis, *Estimation of graphical models through structured norm minimization*, J. Mach. Learn. Res., 18 (2018), pp. 1–48.
[47] Y.-X. Wang, H. Xu, and C. Leng, *Provable subspace clustering: When LRR meets SSC*, in Proceedings of the Conference on Advances in Neural Information Processing Systems, 2013, pp. 64–72.
[48] J. Wright, A. Y. Yang, A. Ganesh, S. S. Sastry, and Y. Ma, *Robust face recognition via sparse representation*, IEEE Trans. Pattern Anal. Mach. Intell., 31 (2009), pp. 210–227.
[49] Y. Xie, D. Tao, W. Zhang, and L. Zhang, *Multi-View Subspace Clustering via Relaxed l_1-norm of Tensor Multi-Rank*, preprint, arXiv:1610.07126 [cs.CV], 2016.
[50] J. Yang, O. Rubel, M. W. Mahoney, and B. P. Bowen, *Identifying important ions and positions in mass spectrometry imaging data using cur matrix decompositions*, Anal. Chem., 87 (2015), pp. 4658–4666.
[51] J. Zhang, A. K. Saibaba, M. Kilmer, and S. Aeron, *Divide-and-Conquer Matrix Factorization*, preprint, arXiv:1609.07086 [math.NA], 2016.
[52] Y. Zhang, Z. Jiang, and L. S. Davis, *Learning structured low-rank representations for image classification*, in Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2013, pp. 676–683.
[53] Z. Zhang and S. Aeron, *Exact tensor completion using t-SVD*, IEEE Trans. Signal Process., 65 (2016), pp. 1511–1526.
[54] Z. Zhang, G. Ely, S. Aeron, N. Hao, and M. Kilmer, *Novel methods for multilinear data completion and de-noising based on tensor-SVD*, in Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2014, pp. 3842–3849.
[55] T. Zhou and D. Tao, *GoDec: Randomized low-rank & sparse matrix decomposition in noisy case*, in Proceedings of the International Conference on Machine Learning, 2011.