A Novel Algebraic Framework for Processing Multidimensional Data: Theory and Application

A dissertation submitted by

Zemin Zhang

in partial fulfillment of the requirements for the degree of

Doctor of Philosophy

in

Department of Electrical Engineering

Tufts University

February 2017

This dissertation by Zemin Zhang is accepted in its present form by

the Department of Electrical Engineering as satisfying the dissertation requirement

for the degree of Doctor of Philosophy.

Date _____ _____

Shuchin Aeron, Ph.D., Advisor
Department of Electrical and Computer Engineering

Recommended to the Graduate Council

Date _____ _____

Eric Miller, Ph.D., Reader

Date _____ _____

Misha Kilmer, Ph.D., Reader

Date _____ _____

Dehong Liu, Ph.D., Reader

Approved by the Graduate Council

Date _____ _____

Dean of the Graduate School

Abstract

Tensor related analysis and applications are more and more popular in computer vision, machine learning, data mining, psychometrics, signal processing and other areas. In this thesis, we first discuss, then build on a recently proposed tensor algebraic framework[1, 2, 3, 4], in which one can obtain a factorization for multidimensional data, referred to as the tensor-SVD (t-SVD) as similar to the Singular Value Decomposition (SVD) for matrices.t-SVD results in a notion of rank referred to as the tubal-rank.

Using this approach we consider the problem of sampling and recovery of 3-D arrays with what we will call low tubal-rank. We show that by solving a convex optimization problem, which minimizes a convex surrogate to the tubal-rank, one can guarantee exact recovery with high probability as long as number of samples is of the order $O(rnk \log(nk))$ given a tensor of size $n \times n \times k$ with tubal-rank r. The conditions under which this result holds are similar to the incoherence conditions for low-rank matrix completion under random sampling. The difference is that we define

incoherence under the algebraic set-up of the t-SVD, which is different from the standard matrix incoherence conditions. We also compare the numerical performance of the proposed algorithm with some state-of-the-art approaches on real-world datasets.

After that, we discuss t-SVD-based robust PCA methods, in both the batch and the online manner. Applications in image denoising and fusing cloud-contaminated satellite images demonstrate that the proposed method shows superiority in both convergence speed and performance compared to the state-of-the-art approaches.

In the end, a new dictionary learning algorithm for multidimensional data is proposed. Unlike most conventional dictionary learning methods which are derived for dealing with vectors or matrices, our algorithm, named K-TSVD, learns a multidimensional dictionary directly based on t-SVD. We propose to extend the K-SVD algorithm used for 1-D data to a K-TSVD algorithm for handling 2-D and 3-D data. Our algorithm, based on the idea of sparse coding (using group-sparsity over multidimensional coefficient vectors), alternates between estimating a compact representation and dictionary learning. We analyze our K-TSVD algorithm and demonstrate its result on video completion and video/multispectral image denoising.

Acknowledgements

Foremost, I would like to express my sincere gratitude to my advisor Prof. Shuchin Aeron. Without his guidance and support in the research, I would not be where I am right now. I consider that all I have achieved during my doctorate, and the fun I have had would not have been possible without his support and patience. During my last year in Tufts, which is also the hardest year of my entire life, Shuchin fully held my back and helped me through those dark days. No words will express how much I appreciate it.

I would like to thank my thesis committee: Prof. Eric Miller, Prof. Misha Kilmer and Dr. Dehong Liu, for their insightful comments and suggestions. I thank Prof. Eric Miller for his great courses and research advice, I have benefited immensely from them. I'm grateful to Prof. Misha Kilmer for her technical crucial support in my research. I also thank Dr. Dehong Liu for his guidance and great work during my summer internship in Mitsubishi Electrical Research Lab.

I owe my beloved wife Jiaqi Cai a lot. Thank you for your love, patience, encouragement and support. I know it is also very hard for you during those days. I can't

make all these happen without you. I love you, and thank you for everything you have done for me.

At last, my deepest love to my parents, Guangrui Zhang and Meiying Zhang, for their unconditional love. Even in the last days of her life, my mom Meiying Zhang was always caring about my work and life. Mom I love you. I'm so sad for losing you and not letting you see me graduate. Hope you rest peacefully in heaven.

Contents

List of Tables

List of Figures

Chapter 1

Introduction to Tensors

For the past 4 decades there have been a lot active research on tensor representations and decompositions, which are used to analyze the data, extract and explain their properties. The goal of this chapter is to provide an overview of high-order tensors and their decompositions.

In 1927, tensor decompositions were originally proposed in [5, 6], but did not received a lot attention. Later in the 1960s, the work of Tucker [7, 8, 9], Carroll & Chang [10] and Harshman [11] appeared in psychometrics literature and attracted more research attention since then. In recent years, tensor decomposition has drawn more interests in various fields, such as signal processing [12, 13, 14], numerical linear algebra [15, 16, 17], data mining [18, 19, 20] and computer vision [21, 22, 23, 24] and so on. In various applications, it is appropriate to store the data in multidimensional

arrays, rather than matrices, to apply some advantages and useful properties of tensors in solving different problems [2, 22, 25, 26]. Therefore, tensor related research is very important and worth studying.

In the following we will give a brief introduction to some classic tensor decompositions and models, summarize their advantages and drawbacks, and survey some related applications.

1.1 What are tensors?

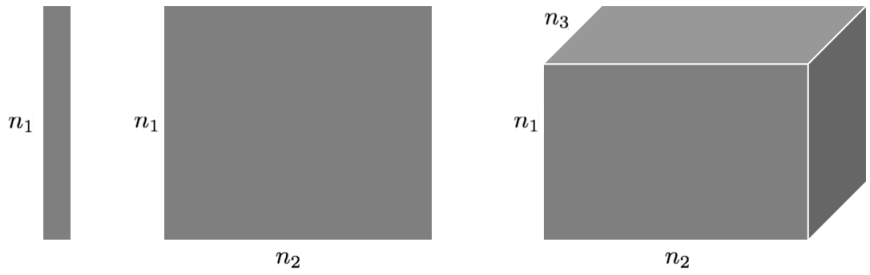


Figure 1.1: First order tensor, second order tensor and third order tensor.

Tensors are multidimensional arrays. The "order" of a tensor is simply the number of dimensions of this array. For instance, a vector is a first-order tensor, a matrix is a second-order tensor, and a d-dimensional array is a d-th order tensor. Figure 1.1 illustrates them. In general, a tensor of order 3 or higher is called a higher-order tensor. In this thesis, matrices are represented by uppercase boldface letters and vectors by lower case boldface letters. Tensors are represented in bold script font. For instance, a third-order tensor is represented as $\boldsymbol{\mathcal{A}}$, and its $(i, j, k)th$ entry is

represented as $\boldsymbol{\mathcal{A}}_{ijk}$.

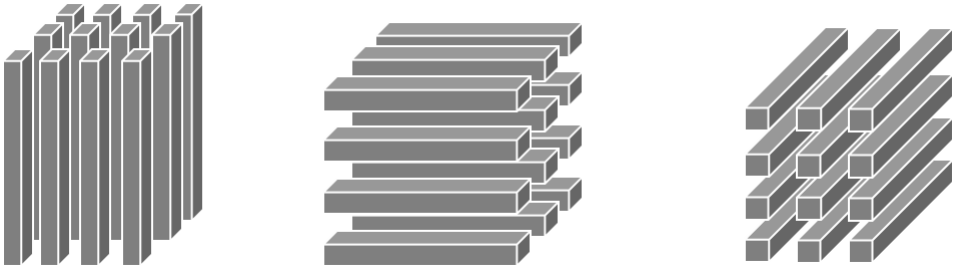


Figure 1.2: Mode-1, mode-2 and mode-3 fibers.

A fiber is a one-dimensional section, which is defined by fixing all the indices but one. A third-order tensor example is shown in Figure 1.2. Often we will use the following notation to extract the tensor fibers. $\boldsymbol{\mathcal{A}}(i, j, :)$ denotes a tube/fiber oriented into the board obtained by fixing the first two indices and varying the third. The mode-1, mode-2 and mode-3 fibers of a third-order tensor $\boldsymbol{\mathcal{A}}$ are represented as $\boldsymbol{\mathcal{A}}(:, j_1, k_1)$, $\boldsymbol{\mathcal{A}}(i_2, :, k_2)$ and $\boldsymbol{\mathcal{A}}(i_3, j_3, :)$.

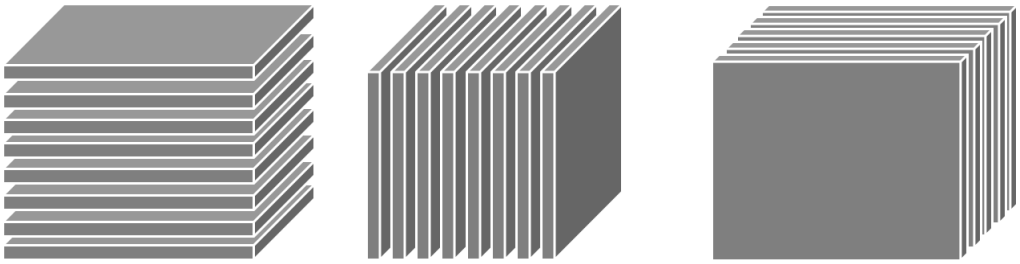


Figure 1.3: Mode-1, mode-2 and mode-3 slices.

Like fibers, in third-order tensors the slices also have different directions. A slice of a tensor is defined by fixing all indices but two. Figure 1.3 shows horizontal (mode-1), lateral (mode-2) and frontal (mode-3) slices of a third order tensor $\boldsymbol{\mathcal{A}}$. They are

denoted by $\boldsymbol{\mathcal{A}}(i,:,:)$, $\boldsymbol{\mathcal{A}}(:,j,:)$ and $\boldsymbol{\mathcal{A}}(:,:,k)$, respectively. For the frontal slices, we will frequently use a more compact notation $\boldsymbol{\mathcal{A}}^{(i)} \triangleq \boldsymbol{\mathcal{A}}(:,:,i) \in \mathbb{R}^{n_1 \times n_2}$.

Matricization is the process of reordering the elements of an N-way array into matrix. It is also know as flattening and unfolding. For instance, a tensor of order-N has N different matricizations along different directions. Given a $3 \times 4 \times 2$ tensor $\boldsymbol{\mathcal{X}}$ as follows,

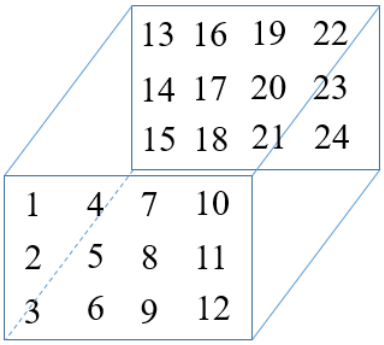


Figure 1.4: Tensor $\boldsymbol{\mathcal{X}}$ of size $3 \times 4 \times 2$

Then the mode-1, mode-2 and mode-3 matricization of $\boldsymbol{\mathcal{X}}$ is given as,

$$X_{(1)} = \begin{bmatrix} 1 & 4 & 7 & 10 & 13 & 16 & 19 & 22 \\ 2 & 5 & 8 & 11 & 14 & 17 & 20 & 23 \\ 3 & 6 & 9 & 12 & 15 & 18 & 21 & 24 \end{bmatrix} \tag{1.1}$$

$$X_{(2)} = \begin{bmatrix} 1 & 2 & 3 & 13 & 14 & 15 \\ 4 & 5 & 6 & 16 & 17 & 18 \\ 7 & 8 & 9 & 19 & 20 & 21 \\ 10 & 11 & 12 & 22 & 23 & 24 \end{bmatrix} \tag{1.2}$$

$$X_{(3)} = \begin{bmatrix} 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 & 10 & 11 & 12 \\ 13 & 14 & 15 & 16 & 17 & 18 & 19 & 20 & 21 & 22 & 23 & 24 \end{bmatrix} \tag{1.3}$$

When the data is stored in multidimensional tensors, we will need different tensor manipulation of the data according to the applications. In many problems, a very important step is to compress the high-order tensors to guarantee that the compressed representation of the tensor retains certain properties. Therefore, a lot of research interests are focusing on developing new tensor decompositions in the literature, such as CANDECOMP/PARAFAC decomposition [5, 6], Tucker decomposition [7, 8, 9], Hierarchical Tucker [27], Tensor Train [28] and so on.

In the following, we will go over two famous tensor decompositions: CP decompositon and Tucker decomposition.

1.2 CANDECOMP/PARAFAC Decomposition

In this section, we will briefly introduce the CANDECOMP/PARAFAC decomposition, which expresses a tensor as the sum of a finite number of vector outer-products. In details, given a third-order tensor $\boldsymbol{\mathcal{X}} \in \mathbb{R}^{I \times J \times K}$, the CP decomposition is written as follows

$$\boldsymbol{\mathcal{X}} = \sum_{r=1}^{R} a_r \circ b_r \circ c_r \tag{1.4}$$

where R is a positive integer, the symbol "$\circ$" represents the vector outer products,

$a_r \in \mathbb{R}^I$, $b_r \in \mathbb{R}^J$ and $c_r \in \mathbb{R}^K$ for $r = 1, ..., R$. Each term $a_r \circ b_r \circ c_r, r = 1, 2, ..., R$ is called a CP-rank 1 tensor.The smallest number of R is called the tensor CP rank.

Elementwise, (1.4) is equivalent to

$$\mathcal{X}_{ijk} = \sum_{r=1}^{R} a_{ir} b_{jr} c_{kr} \tag{1.5}$$

for $i = 1, ..., I$, $j = 1, ..., J$ and $k = 1, ..., K$. Figure 1.5 shows the CP decomposition of a third-order tensor.

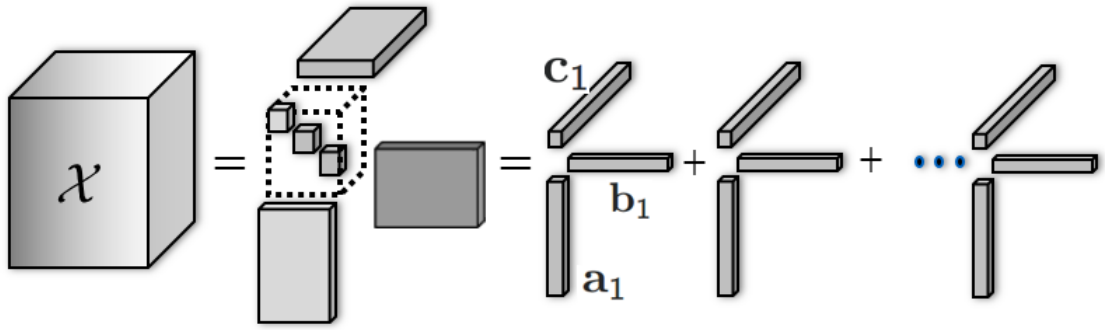


Figure 1.5: CP decomposition of a third-order tensor.

The CP rank of a tensor $\mathcal{X}$, is defined as the smallest number of CP rank-1 tensors that generates $\mathcal{X}$ as their sum. In other words, it is the smallest number of components in the CP decomposition. There are two important properties of this CP rank. First, the rank of a real-valued tensor may be different over $\mathbb{R}$ and $\mathbb{C}$. For example, let $\mathcal{X}$ be a tensor with frontal slices be

$$X(:,:,1) = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}, X(:,:,2) = \begin{bmatrix} 0 & 1 \\ -1 & 0 \end{bmatrix} \tag{1.6}$$

This is a tensor of CP rank 3 over $\mathbb{R}$ and CP rank 2 over $\mathbb{C}$ [29]. For detailed proof and the methodology for computing the factors, please see [30].

Second, given a specific tensor, there is no straightforward algorithm to determine the CP rank. Moreover, it is proved to be NP-hard [31]. This actually prevents the CP decomposition from some theoretical guarantees on certain bounds. In practice, the CP rank of a tensor is estimated numerically by fitting various rank-R CP models, and choose the best one with respect to the Frobenius norm.

Another property of the CP decomposition is the rank-R decomposition is unique. Let $\mathcal{X} \in \mathbb{R}^{I \times J \times K}$ be a third-order tensor of rank-R. Then its CP decomposition

$$\mathcal{X} = \sum_{r=1}^{R} a_r \circ b_r \circ c_r = [\![A, B, C]\!]$$

is unique, with the exception of the elementary indeterminacies of scaling and permutation. The last inequality is a shorthand introduced by [32]. There are many algorithms to compute a CP decomposition given a CP-rank R, among which the most frequently used one is the alternating least square (ALS) algorithm [10].

There are a lot of applications based on CP. In the 1970s, it was originally used in psychometrics [10]. Appellof and Davidson [33] pioneered the use of CP in chemometrics in 1981. There is also some research in neuroscience [34, 35], data mining [18, 36]. In computer vision, the CP decomposition is used in in image compression and classification [37]. The authors of [38] applied CP to bidirectional texture functions in order to build a compressed texture database. In the following chapters,

we will see some CP decomposition related algorithms comparing to our methods on different problems [39, 40, 41].

1.3 Tucker Decomposition

In 1963, the Tucker Decomposition was proposed by Tucker [7, 8, 9]. It decompose a tensor into a core tensor multiplied by a matrix along each mode. For a third-order tensor $\mathcal{X} \in \mathbb{R}^{I \times J \times K}$, we have

$$\mathcal{X} = \mathcal{G} \times_1 A \times_2 B \times_3 C = \sum_{p=1}^{P} \sum_{q=1}^{Q} \sum_{r=1}^{R} g_{pqr} a_p \circ b_q \circ c_r = [\mathcal{G}; A, B, C] \tag{1.7}$$

where $A \in \mathbb{R}^{I \times P}$, $B \in \mathbb{R}^{J \times Q}$ and $C \in \mathbb{R}^{K \times R}$. The symbol $\times_k$ is the n-mode (matrix) product defined as follows: Given a tensor $\mathcal{X} \in \mathbb{R}^{I_1 \times I_2 \times \ldots \times I_N}$ and a matrix $U \in \mathbb{R}^{J \times I_n}$, the n-mode (matrix) product of $\mathcal{X}$ and u is an $I_1 \times \ldots \times I_{n-1} \times J \times I_{n+1} \times \ldots \times I_N$ tensor. Elementwise, we have

$$(\mathcal{X} \times_n U)_{i_1 \ldots i_{n-1} j i_{n+1} \ldots i_N} = \sum_{i_n=1}^{I_n} x_{i_1 i_2 \ldots i_N} u_{j i_n}$$

and this could also be expressed using the mode-n matricization:

$$\mathcal{Y} = \mathcal{X} \times_n U \Leftrightarrow Y_{(n)} = U X_{(n)}$$

where the mode-n matricization is defined in (1.1) to (1.3) . The Tucker Decomposition of a third-order tensor is shown in Figure 1.6. In (1.7), if $P < I$, $Q < J$ and

$R < K$, the core tensor $\mathcal{G}$ can be regarded as a compression of tensor $\mathcal{X}$.

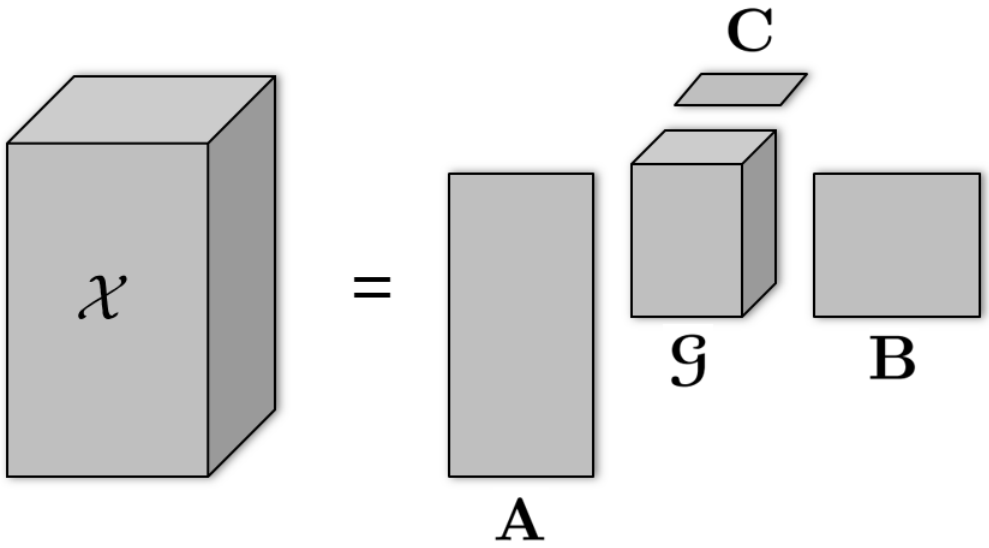


Figure 1.6: Tucker decomposition of a third-order tensor.

The Tucker n-rank of a tensor $\mathcal{X} \in \mathbb{R}^{I_1 \times I_2 \times \ldots \times I_N}$, denoted as $\text{rank}_n(\mathcal{X})$, is the column rank of its mode-n matricization $X_{(n)}$. If $R_n = \text{rank}_n(\mathcal{X})$ for $n = 1, ..., N$, then we can say the Tucker rank of tensor $\mathcal{X}$ is $(R_1, R_2, ..., R_N)$.

To compute an orthogonal Tucker decomposition, a well-known method is called higher-order SVD (HOSVD) from [15]. The idea is to find the decomposition components of the tensor in each mode, independent of the other modes. If for at least one $R_n < \text{rank}_n(\mathcal{X})$, the decomposition is called the *truncated* HOSVD. Note that the truncated HOSVD is not optimal in terms of the best approximation with respect to to the norm of difference. As we will see later, the t-SVD, which we will talk about and use throughout this thesis, has this optimality condition in the sense that the tubal rank-k approximation is the "best fit" to the original tensor under Frobenius

norm of the difference.

Aside from this non-optimality intruncation, the Tucker decomposition is also not unique [29]. Consider the decomposition of a third-order tensor $\mathcal{X}$ in (1.7). Let $U \in \mathbb{R}^{P\times P}$, $V \in \mathbb{R}^{Q\times Q}$ and $W \in \mathbb{R}^{R\times R}$ be nonsingular matrices. Then the Tucker decomposition could also be

$$[\![\mathcal{G}; A, B, C]\!] = [\![\mathcal{G} \times_1 U \times_2 V \times_3 W; AU^{-1}, BV^{-1}, CW^{-1}]\!]$$

which means we are able to modify the core tensor $\mathcal{G}$ with multiplying inverse modifications to the factor matrices.

There are various applications in which the Tucker decomposition is used. In [42] and [43], the authors applied Tucker decomposition in signal processing. In computer vision, [21] is the first one to use Tucker decomposition in TensorFaces to apply facial recognition. The authors of [44] showed that the facial recognition results using TensorFaces is much more accurate than the standard PCA techniques. In [45], Tucker decomposition is applied to human motions, while in [46, 47] Tucker decomposition is used to model facial expressions and compress images.

Even though CP and Tucker decomposition have various applications in multiple fields, in this thesis, we will use an entirely different approach which obtains an energy-revealing factorization based on the framework present in [1, 2, 3, 4]. Later in this thesis, we will demonstrate some applications based on this novel factorization, and compare the its results to the methods based on CP, Tucker and other decomposition.

One important thing to mention is that there is no a tensor decomposition that works universally well on all the applications. Unlike the matrix case, where the compression is often accomplished via Singular Value Decomposition (SVD), in which the number of non-zero singular values is exactly the matrix rank, for higher-order tensors, there is no such universal concept of rank that is well established. Different tensor decompositions have different definitions of rank. Facing a real applications, we need to carefully analyze the data and choose a proper tensor decomposition based on the problems and the data itself.

Chapter 2

Notations and preliminaries

In this chapter we will talk about the related notations of t-product and t-SVD. The development is adapted from [1, 2, 3, 4]

Some more notations are needed in this chapter. For convinience, a mode-3 fiber is called *t*ensor tube as $\mathring{\boldsymbol{a}} \in \mathbb{R}^{1\times 1\times n_3}$, and a mode-2 slice is denoted as *t*ensor column of size $\vec{\boldsymbol{b}} \in \mathbb{R}^{n_1\times 1\times n_3}$. These are illustrated in Figure 2.1.

$\widehat{\boldsymbol{\mathcal{A}}}$ denotes a third-order tensor obtained by taking the Fourier Transform of all the tubes along the third dimension of $\boldsymbol{\mathcal{A}}$, i.e., for $i = 1, ..., n_1$ and $j = 1, ..., n_2$, i.e.,

$$vec(\widehat{\boldsymbol{\mathcal{A}}}(i, j, :)) = \boldsymbol{\mathcal{F}}(vec(\boldsymbol{\mathcal{A}}(i, j, :))), \tag{2.1}$$

where *vec* is the vectorization operator that takes the tensor tube and makes it a vector, and $\boldsymbol{\mathcal{F}}$ stands for the Discrete Fourier Transform (DFT). For compactness, we will use the following notation for the DFT along the 3rd dimension: $\widehat{\boldsymbol{\mathcal{A}}} = \texttt{fft}(\boldsymbol{\mathcal{A}}, [\,], 3)$. In the same fashion, one can also compute $\boldsymbol{\mathcal{A}}$ from $\widehat{\boldsymbol{\mathcal{A}}}$ via $\texttt{ifft}(\widehat{\boldsymbol{\mathcal{A}}}, [\,], 3)$ using the

inverse FFT operation along the 3-rd dimension. For vectors $\mathbf{x}$ and $\mathbf{y}$ of length n, $\mathbf{y} = \texttt{fft}(\mathbf{x})$ and $\mathbf{x} = \texttt{ifft}(\mathbf{y})$ are defined as follows.

$$\begin{aligned} \mathbf{y}(k) &= \sum_{j=1}^{n} \mathbf{x}(j) w_n^{(j-1)(k-1)} \\ \mathbf{x}(j) &= \frac{1}{n} \sum_{k=1}^{n} \mathbf{y}(k) w_n^{-(j-1)(k-1)} \ , \end{aligned} \tag{2.2}$$

where $w_n = e^{(-2\pi i)/n}$ is one of the n roots of unity.

Definition 2.0.1. ***(Tensor transpose [3])*** *The conjugate transpose of a tensor $\boldsymbol{\mathcal{A}} \in \mathbb{R}^{n_1 \times n_2 \times n_3}$ is the $n_2 \times n_1 \times n_3$ tensor $\boldsymbol{\mathcal{A}}^\top$ obtained by conjugate transposing each of the frontal slice and then reversing the order of transposed frontal slices 2 through n_3:*

$$\begin{aligned} \left(\boldsymbol{\mathcal{A}}^\top\right)^{(1)} &= \left(\boldsymbol{\mathcal{A}}^{(1)}\right)^\top \\ \left(\boldsymbol{\mathcal{A}}^\top\right)^{(i)} &= \left(\boldsymbol{\mathcal{A}}^{(n_3+2-i)}\right)^\top, \ i = 2, ..., n_3 \end{aligned}$$

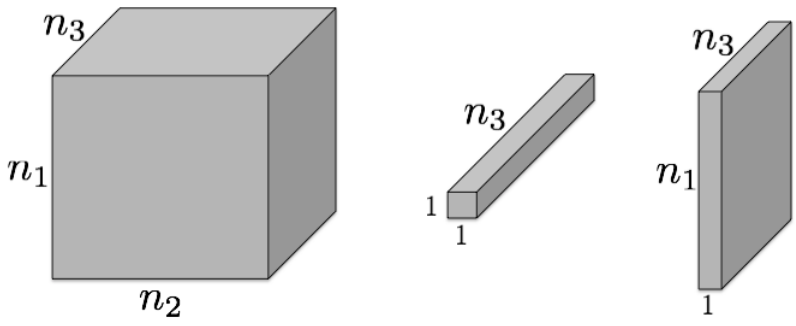


Figure 2.1: General third-order tensor, tensor tube and tensor column.

Definition 2.0.2. ***(t-product [3])*** *The t-product* $\boldsymbol{\mathcal{A}} * \boldsymbol{\mathcal{B}}$ *of* $\boldsymbol{\mathcal{A}} \in \mathbb{R}^{n_1 \times n_2 \times n_3}$ *and* $\boldsymbol{\mathcal{B}} \in \mathbb{R}^{n_2 \times n_4 \times n_3}$ *is an* $n_1 \times n_4 \times n_3$ *tensor whose* (i,j)*th tube* $\mathring{\boldsymbol{c}}_{ij}$ *is given by*

$$\mathring{\boldsymbol{c}}_{ij} = \boldsymbol{\mathcal{C}}(i,j,:) = \sum_{k=1}^{n_2} \boldsymbol{\mathcal{A}}(i,k,:) * \boldsymbol{\mathcal{B}}(k,j,:) \tag{2.3}$$

where $*$ *denotes the circular convolution between two tubes of same size.*

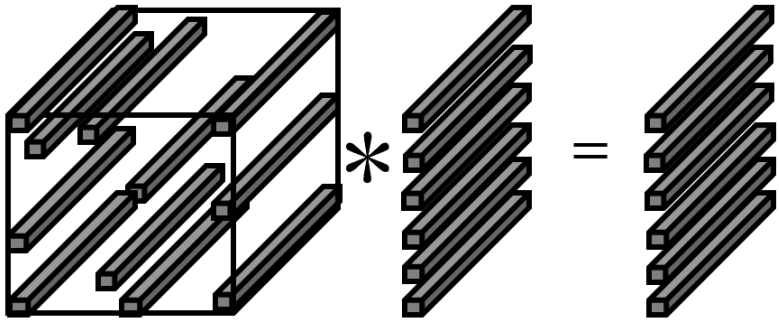


Figure 2.2: T-product of a third-order tensor and mode-2 slice.

Interpreted in another way, a 3-D tensor of size $n_1 \times n_2 \times n_3$ can be viewed as an $n_1 \times n_2$ matrix of fibers (tubes) oriented along the third dimension. So the t-product of two tensors can be regarded as a matrix-matrix multiplication, except that the multiplication operation between scalars is replaced by circular convolution between the tubes. This allows one to consider 3-D tensors as linear operators over matrices. That is, when $\boldsymbol{\mathcal{A}}$ is a $n_1 \times n_2 \times n_3$ tensor and $\boldsymbol{\mathcal{B}}$ is an $n_2 \times 1 \times n_3$ tensor (essentially matrix oriented into the paper), $\boldsymbol{\mathcal{A}} * \boldsymbol{\mathcal{B}}$ is an $n_1 \times 1 \times n_3$ tensor. This perspective has been recently used in [48] for problems of unsupervised clustering of 2-D data. For sake of brevity we direct the interested readers to [3, 1, 49].

Definition 2.0.3. ***(Identity tensor [3])*** *The identity tensor* $\boldsymbol{\mathcal{I}} \in \mathbb{R}^{n_1 \times n_1 \times n_3}$ *is defined to be a tensor whose first frontal slice* $\boldsymbol{\mathcal{I}}^{(1)}$ *is the* $n_1 \times n_1$ *identity matrix and*

all other frontal slices $\mathfrak{I}^{(i)}, i = 2, ..., n_3$ *are zero.*

Definition 2.0.4. ***(Orthogonal tensor [3])*** *A tensor* $\mathfrak{Q} \in \mathbb{R}^{n \times n \times n_3}$ *is orthogonal if it satisfies*

$$\mathfrak{Q}^\top * \mathfrak{Q} = \mathfrak{Q} * \mathfrak{Q}^\top = \mathfrak{I} \tag{2.4}$$

Definition 2.0.5. ***(Block diagonal form of third-order tensor [22])*** *Let* $\overline{\mathcal{A}}$ *denote the block-diagonal matrix of the tensor* $\widehat{\mathcal{A}}$ *in the Fourier domain, i.e.,*

$$\begin{aligned} \overline{\mathcal{A}} &\triangleq \textit{blockdiag}(\widehat{\mathcal{A}}) \\ &\triangleq \begin{bmatrix} \widehat{\mathcal{A}}^{(1)} & & & \\ & \widehat{\mathcal{A}}^{(2)} & & \\ & & \ddots & \\ & & & \widehat{\mathcal{A}}^{(n_3)} \end{bmatrix} \in \mathbb{C}^{n_1 n_3 \times n_2 n_3} \end{aligned} \tag{2.5}$$

It is easy to verify that the block diagonal matrix of $\mathcal{A}^\top$ *is equal to the transpose of the block diagonal matrix of* $\mathcal{A}$*:*

$$\overline{\mathcal{A}^\top} = \overline{\mathcal{A}}^\top \tag{2.6}$$

Remark 2.0.1. *The following fact will be used through out the paper. For any tensor* $\mathcal{A} \in \mathbb{R}^{n_1 \times n_2 \times n_3}$ *and* $\mathcal{B} \in \mathbb{R}^{n_2 \times n_4 \times n_3}$*, we have*

$$\mathcal{A} * \mathcal{B} = \mathcal{C} \Longleftrightarrow \overline{\mathcal{A}}\overline{\mathcal{B}} = \overline{\mathcal{C}}$$

The t-product allows us to define a tensor Singular Value Decomposition (t-SVD). We need one more definition to state the decomposition.

Definition 2.0.6. ***(f-diagonal tensor [3])*** *A tensor $\boldsymbol{\mathcal{A}}$ is called f-diagonal if each frontal slice $\boldsymbol{\mathcal{A}}^{(i)}$ is a diagonal matrix.*

Definition 2.0.7. ***(Tensor Singular Value Decomposition: t-SVD [3])*** *For $\boldsymbol{\mathcal{M}} \in \mathbb{R}^{n_1 \times n_2 \times n_3}$, the t-SVD of $\boldsymbol{\mathcal{M}}$ is given by*

$$\boldsymbol{\mathcal{M}} = \boldsymbol{\mathcal{U}} * \boldsymbol{\mathcal{S}} * \boldsymbol{\mathcal{V}}^{\top} \tag{2.7}$$

where $\boldsymbol{\mathcal{U}}$ and $\boldsymbol{\mathcal{V}}$ are orthogonal tensors of size $n_1 \times n_1 \times n_3$ and $n_2 \times n_2 \times n_3$ respectively. $\boldsymbol{\mathcal{S}}$ is a rectangular f-diagonal tensor of size $n_1 \times n_2 \times n_3$, and the entries in $\boldsymbol{\mathcal{S}}$ are called the singular values of $\boldsymbol{\mathcal{M}}$. $*$ denotes the t-product here.

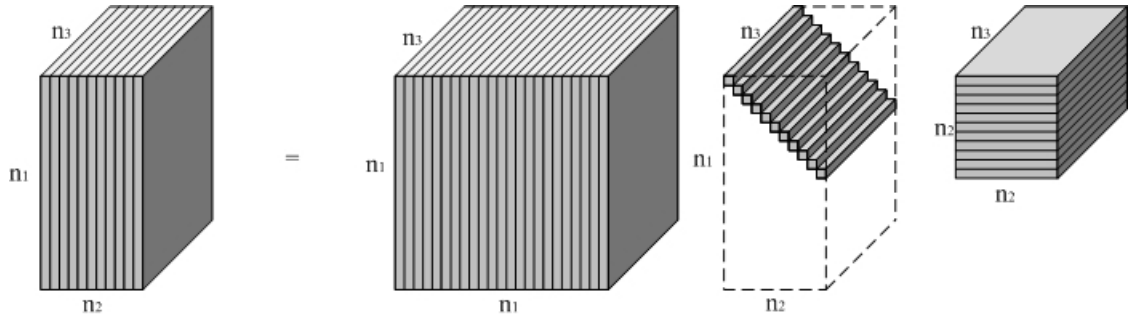


Figure 2.3: The t-SVD of an $n_1 \times n_2 \times n_3$ tensor.

One can obtain this decomposition by computing matrix SVDs in the Fourier domain as shown in Algorithm 1. Figure 2.3 illustrates the decomposition for the 3-D case.

Based on the t-SVD one can define the following notion of tensor rank.

Definition 2.0.8. ***(Tensor tubal-rank [22])*** *The tensor* ***tubal-rank*** *r of $\boldsymbol{\mathcal{A}}$ is defined to be the number of non-zero singular tubes of $\boldsymbol{\mathcal{S}}$, where $\boldsymbol{\mathcal{S}}$ comes from the*

Algorithm 1 t-SVD for third order tensors

Input: $\boldsymbol{\mathcal{M}} \in \mathbb{R}^{n_1 \times n_2 \times n_3}$.
Output: $\boldsymbol{\mathcal{U}} \in \mathbb{R}^{n_1 \times n_1 \times n_3}$, $\boldsymbol{\mathcal{S}} \in \mathbb{R}^{n_1 \times n_2 \times n_3}$, $\boldsymbol{\mathcal{V}} \in \mathbb{R}^{n_2 \times n_2 \times n_3}$.

$\boldsymbol{\mathcal{D}} \leftarrow \text{fft}(\boldsymbol{\mathcal{M}}, [\,], 3)$
for $i = 1$ to n_3 **do**
  $[\mathbf{U}, \mathbf{S}, \mathbf{V}] = \texttt{svd}(\boldsymbol{\mathcal{D}}^{(i)})$
  $\widehat{\boldsymbol{\mathcal{U}}}^{(i)} = \mathbf{U}$; $\widehat{\boldsymbol{\mathcal{S}}}^{(i)} = \mathbf{S}$; $\widehat{\boldsymbol{\mathcal{V}}}^{(i)} = \mathbf{V}$;
end for
$\boldsymbol{\mathcal{U}} \leftarrow \text{ifft}(\widehat{\boldsymbol{\mathcal{U}}}, [\,], 3)$; $\boldsymbol{\mathcal{S}} \leftarrow \text{ifft}(\widehat{\boldsymbol{\mathcal{S}}}, [\,], 3)$; $\boldsymbol{\mathcal{V}} \leftarrow \text{ifft}(\widehat{\boldsymbol{\mathcal{V}}}, [\,], 3)$

t-SVD of $\boldsymbol{\mathcal{A}} : \boldsymbol{\mathcal{A}} = \boldsymbol{\mathcal{U}} * \boldsymbol{\mathcal{S}} * \boldsymbol{\mathcal{V}}^\top$. *An alternative definition of tubal-rank is that it is the largest rank of all the frontal slices of* $\widehat{\boldsymbol{\mathcal{A}}}$ *in Fourier domain. If we say a third order tensor* $\boldsymbol{\mathcal{A}}$ *is of* ***full tubal-rank****, it means* $r = min\{n_1, n_2\}$.

Remark 2.0.2. *It is usually sufficient to compute the reduced version of the t-SVD using the tensor tubal-rank. It's faster and more economical for storage. In details, suppose* $\boldsymbol{\mathcal{M}} \in \mathbb{R}^{n_1 \times n_2 \times n_3}$ *has tensor tubal-rank* r*, then the reduced t-SVD of* $\boldsymbol{\mathcal{M}}$ *is given by*

$$\boldsymbol{\mathcal{M}} = \boldsymbol{\mathcal{U}} * \boldsymbol{\mathcal{S}} * \boldsymbol{\mathcal{V}}^\top \tag{2.8}$$

where $\boldsymbol{\mathcal{U}} \in \mathbb{R}^{n_1 \times r \times n_3}$ *and* $\boldsymbol{\mathcal{V}} \in \mathbb{R}^{n_2 \times r \times n_3}$ *satisfying* $\boldsymbol{\mathcal{U}}^\top * \boldsymbol{\mathcal{U}} = \boldsymbol{\mathcal{I}}, \boldsymbol{\mathcal{V}}^\top * \boldsymbol{\mathcal{V}} = \boldsymbol{\mathcal{I}}$. $\boldsymbol{\mathcal{S}} \in \mathbb{R}^{r \times r \times n_3}$ *is an* f*-diagonal tensor. This reduced version of the t-SVD will be used throughout the paper unless otherwise noted.*

An important property of the t-SVD is the optimality of the truncated t-SVD for data approximation as stated in the following Lemma.

Lemma 2.0.1 ([3])**.** *Let the t-SVD of* $\boldsymbol{\mathcal{M}} \in \mathbb{R}^{n_1 \times n_2 \times n_3}$ *be given by* $\boldsymbol{\mathcal{M}} = \boldsymbol{\mathcal{U}} * \boldsymbol{\mathcal{S}} * \boldsymbol{\mathcal{V}}^T$

and for $k < \min(n_1, n_2)$ *define* $\boldsymbol{\mathcal{M}}_k = \sum_{i=1}^{k} \boldsymbol{\mathcal{U}}(:,i,:) * \boldsymbol{\mathcal{S}}(i,i,:) * \boldsymbol{\mathcal{V}}(:,i,:)^T$, *Then*

$$\boldsymbol{\mathcal{M}}_k = \arg\min_{\widetilde{\boldsymbol{\mathcal{M}}} \in \mathbb{M}} \|\boldsymbol{\mathcal{M}} - \widetilde{\boldsymbol{\mathcal{M}}}\|_F$$

where $\mathbb{M} = \{\boldsymbol{\mathcal{C}} = \boldsymbol{\mathcal{X}} * \boldsymbol{\mathcal{Y}} | \boldsymbol{\mathcal{X}} \in \mathbb{R}^{n_1 \times k \times n_3}, \boldsymbol{\mathcal{Y}} \in \mathbb{R}^{k \times n_2 \times n_3}\}$ *is the set of all tensors with tensor tubal-rank at most* k *and* $\|\cdot\|_F$ *is the Frobenius norm defined in Definition 2.0.14.*

Based on Lemma 2.0.1, one can perform dimensionality reduction and hence tensor compression using the truncated t-SVD. Specifically, given a tensor $\boldsymbol{\mathcal{M}} \in \mathbb{R}^{n_1 \times n_2 \times n_3}$, as in Lemma 2.0.1, we take the tubal rank k approximation $\boldsymbol{\mathcal{X}}_k = \sum_{i=1}^{k} \boldsymbol{\mathcal{U}}(:,i,:)\boldsymbol{\mathcal{S}}(i,i,:)\boldsymbol{\mathcal{V}}(:,i,:)^\top$ as a compression of $\boldsymbol{\mathcal{X}}$. Define the compression ratio as the number of entries used to represent $\boldsymbol{\mathcal{X}}_k$ over the number of entries in $\boldsymbol{\mathcal{X}}$,

$$\text{ratio} = \frac{n_1 n_2 n_3}{n_1 k n_3 + k n_3 + k n_2 n_3} = \frac{n_1 n_2}{k(n_1 + n_2 + 1)}$$

In Figure 2.4 we provide some video examples, which can be compressed (approximated) well using the truncated t-SVD, compared to vectorizing or flattening the data and using the truncated SVD [22]. From the Relative Square Error (RSE) versus the compression ratio plots, we see better performance of the t-SVD over SVD in compression. Here all the videos share a similar feature that the camera is horizontally panning and with linear motion in the video as well. In such videos, a compact representation of one frame to the next frame can be effectively represented as a shift operation, which is captured by a convolution type operation. The t-SVD is based on such an operation along the third dimension, so it results in a much efficient representation. Since efficiency in representation implies efficiency in recovery, and as

we will show later, such data will have a better performance in completion using the t-SVD as well.

Remark 2.0.3. ***(Relation to CP decomposition)*** *Suppose a tensor* $\boldsymbol{\mathcal{A}} \in \mathbb{R}^{n_1 \times n_2 \times n_3}$ *has CP rank* r *and its CP decomposition is given by*

$$\boldsymbol{\mathcal{A}} = \sum_{i=1}^{r} \mathbf{a}_i^{(1)} \circ \mathbf{a}_i^{(2)} \circ \mathbf{a}_i^{(3)}$$

where $\mathbf{a}_i^{(k)} \in \mathbb{R}^{n_k}, k = 1, 2, 3$. *Then tensor* $\widehat{\boldsymbol{\mathcal{A}}}$ *which is obtained by taking the FFT along the third dimension of* $\boldsymbol{\mathcal{A}}$*, has the CP decomposition as follows,*

$$\widehat{\boldsymbol{\mathcal{A}}} = \sum_{i=1}^{r} \mathbf{a}_i^{(1)} \circ \mathbf{a}_i^{(2)} \circ \widehat{\mathbf{a}_i^{(3)}}$$

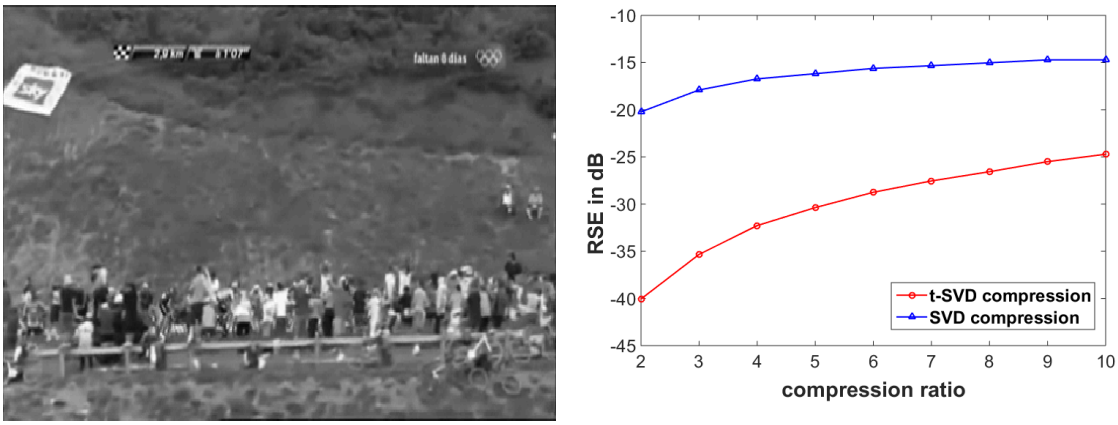
where $\widehat{\mathbf{a}_i^{(3)}} = \texttt{fft}(\mathbf{a}_i^{(3)}), i = 1, 2, ..., r$. *We can see that* $\widehat{\boldsymbol{\mathcal{A}}}$ *also has CP rank* r*, and each frontal slice of* $\widehat{\boldsymbol{\mathcal{A}}}$ *is the sum of* r *rank-*1 *matrices, so the rank of each frontal slice is at most* r*. It implies that if a tensor is of CP rank* r*, its tensor tubal-rank is at most* r*. This means that for a third-order tensor with low CP rank, as we will show later, we can recover it from random samples using the t-SVD structure.*

Definition 2.0.9. ***(Inverse of tensor [3])*** *The inverse of a tensor* $\boldsymbol{\mathcal{A}} \in \mathbb{R}^{n \times n \times n_3}$ *is written as* $\boldsymbol{\mathcal{A}}^{-1}$ *satisfying*

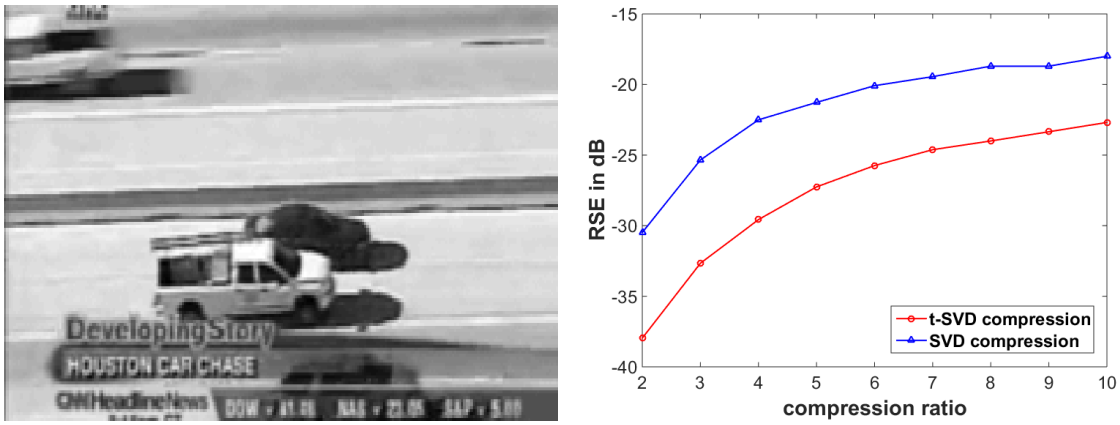
$$\boldsymbol{\mathcal{A}}^{-1} * \boldsymbol{\mathcal{A}} = \boldsymbol{\mathcal{A}} * \boldsymbol{\mathcal{A}}^{-1} = \boldsymbol{\mathcal{I}} \tag{2.9}$$

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

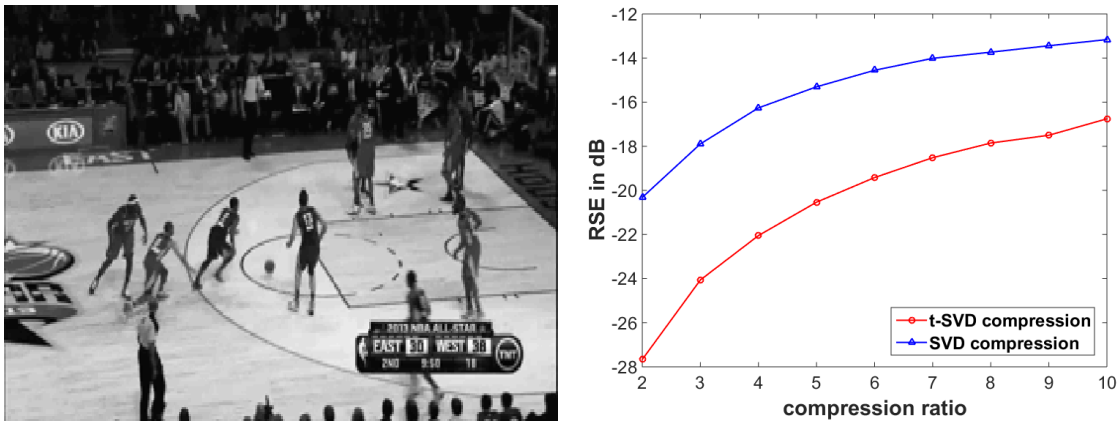
Definition 2.0.10. ***(Tensor operator)*** *Tensor operators are denoted by Calligraphic letters. Suppose* $\mathfrak{L} : \mathbb{R}^{n_1 \times n_2 \times n_3} \to \mathbb{R}^{n_4 \times n_2 \times n_3}$ *is a tensor operator mapping an*



(a) Bicycle video of size $270 \times 480 \times 50$.



(b) Car video of size $136 \times 240 \times 50$.



(c) Basketball video of size $360 \times 640 \times 80$.

Figure 2.4: Some videos and their compression performance using the SVD and the t-SVD. The left figures show one frame of each video and the right figures are the compression performance comparisons of the SVD and t-SVD. The relative square error (RSE) is defined in dB as $\text{RSE} = 20\log_{10}(\|\mathcal{X}_{\text{com}} - \mathcal{X}\|_F / \|\mathcal{X}\|_F)$, where $\mathcal{X}$ is the original video and $\mathcal{X}_{\text{com}}$ is the compressed video, and tensor Frobenius norm $\|\cdot\|_F$ is defined in Definition 1.1.14.

$n_1 \times n_2 \times n_3$ tensor $\boldsymbol{\mathcal{A}}$ to an $n_4 \times n_2 \times n_3$ tensor $\boldsymbol{\mathcal{B}}$ via the t-product as follows:

$$\boldsymbol{\mathcal{A}} = \mathfrak{L}(\boldsymbol{\mathcal{B}}) = \boldsymbol{\mathcal{L}} * \boldsymbol{\mathcal{B}} \tag{2.10}$$

where $\boldsymbol{\mathcal{L}}$ is an $n_4 \times n_1 \times n_3$ tensor. Note that (2.10) is equivalent to the following equation, which lies in the Fourier domain:

$$\overline{\boldsymbol{\mathcal{A}}} = \overline{\boldsymbol{\mathcal{L}}}\,\overline{\boldsymbol{\mathcal{B}}} \tag{2.11}$$

where $\overline{\boldsymbol{\mathcal{A}}} \in \mathbb{C}^{n_1 n_3 \times n_2 n_3}$, $\overline{\boldsymbol{\mathcal{L}}} \in \mathbb{C}^{n_4 n_3 \times n_1 n_3}$ and $\overline{\boldsymbol{\mathcal{B}}} \in \mathbb{C}^{n_4 n_3 \times n_2 n_3}$ are block diagonal matrices.

Remark 2.0.4. *For a tensor operator via t-product defined in (2.10), we are able to transform it into the equivalent form in Fourier domain (2.11) for computational efficiency. On the other hand, we can also transform an operator in Fourier domain back to the original domain as needed.*

Definition 2.0.11. ***(Inner product of tensors)*** *If $\boldsymbol{\mathcal{A}}$ and $\boldsymbol{\mathcal{B}}$ are third-order tensors of same size $n_1 \times n_2 \times n_3$, then the inner product between $\boldsymbol{\mathcal{A}}$ and $\boldsymbol{\mathcal{B}}$ is defined as the following,*

$$\langle \boldsymbol{\mathcal{A}}, \boldsymbol{\mathcal{B}} \rangle = \frac{1}{n_3} \text{trace}(\overline{\boldsymbol{\mathcal{B}}}^{\top} \overline{\boldsymbol{\mathcal{A}}}) \in \mathbb{R} \tag{2.12}$$

where $1/n_3$ comes from the normalization constant of the FFT. The reason that this inner product produces a real-valued result comes from the conjugate symmetric property of the FFT.

Definition 2.0.12. ***(Tensor basis and the corresponding decomposition)*** *We introduce 2 tensor bases here. The first one is called* ***column basis*** $\vec{\boldsymbol{e}}_i$ *of size* $n_1 \times 1 \times n_3$ *with one entry equaling 1 and the rest equaling zero. However, the nonzero entry 1 will only appear at the first frontal slice of* $\vec{\boldsymbol{e}}_i$*. Naturally its transpose* $\vec{\boldsymbol{e}}_i^\top$ *is called* ***row basis****. The other tensor basis is called* ***tube basis*** $\mathring{\boldsymbol{e}}_i$ *of size* $1 \times 1 \times n_3$ *with one entry equaling to 1 and rest equaling to 0. Figure 2.5 illustrates these 2 bases.*

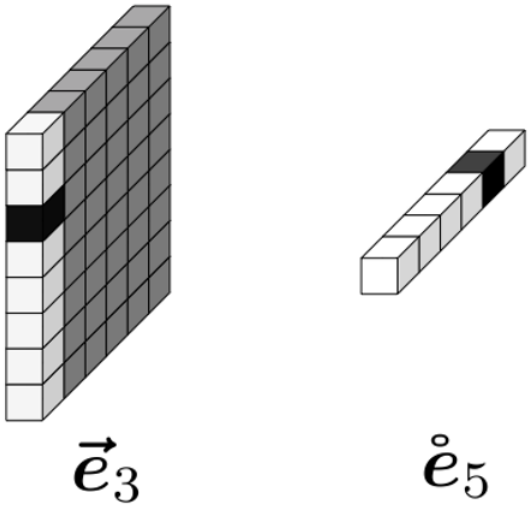


Figure 2.5: The column basis $\vec{\boldsymbol{e}}_3$ and tube basis $\mathring{\boldsymbol{e}}_5$. The black cubes are 1, gray and white cubes are 0. In the tensor column basis $\vec{\boldsymbol{e}}_i$ and row basis $\vec{\boldsymbol{e}}_j^\top$, only the entries in the frontal slice can be 1 (white cubes).

One can obtain a unit tensor $\boldsymbol{\mathcal{E}}$ with only one non-zero entry $\boldsymbol{\mathcal{E}}_{ijk}$ equal to 1 via

$$\boldsymbol{\mathcal{E}} = \vec{\boldsymbol{e}}_i * \mathring{\boldsymbol{e}}_k * \vec{\boldsymbol{e}}_j^\top \ . \tag{2.13}$$

Given any third order tensor $\boldsymbol{\mathcal{X}} \in \mathbb{R}^{n_1 \times n_2 \times n_3}$, we have the following decomposition

$$\begin{aligned}\boldsymbol{\mathcal{X}} &= \sum_{i=1}^{n_1}\sum_{j=1}^{n_2}\sum_{k=1}^{n_3}\langle \vec{\boldsymbol{e}}_i * \mathring{\boldsymbol{e}}_k * \vec{\boldsymbol{e}}_j^{\top}, \boldsymbol{\mathcal{X}}\rangle \vec{\boldsymbol{e}}_i * \mathring{\boldsymbol{e}}_k * \vec{\boldsymbol{e}}_j^{\top} \\ &= \sum_{i=1}^{n_1}\sum_{j=1}^{n_2}\sum_{k=1}^{n_3}\boldsymbol{\mathcal{X}}_{ijk}\vec{\boldsymbol{e}}_i * \mathring{\boldsymbol{e}}_k * \vec{\boldsymbol{e}}_j^{\top}\end{aligned}$$

The proof to such a decomposition is straightforward since $\langle \vec{\boldsymbol{e}}_i * \mathring{\boldsymbol{e}}_k * \vec{\boldsymbol{e}}_j^{\top}, \boldsymbol{\mathcal{X}}\rangle$ will give out the exact value of $\boldsymbol{\mathcal{X}}_{ijk}$.

The following norms on tensors will be used throughout the proof of our main result.

Definition 2.0.13. *(ℓ_{2^*} **norm of tensor column)** Let $\vec{\boldsymbol{x}}$ be an $n_1 \times 1 \times n_3$ tensor column, we define an ℓ_{2^*} norm on it as follows*

$$\|\vec{\boldsymbol{x}}\|_{2^*} = \sqrt{\sum_{i=1}^{n_1}\sum_{k=1}^{n_3}\vec{\boldsymbol{x}}_{i1k}^2}\,. \tag{2.14}$$

Moreover, we have the following relationship between the ℓ_{2^*} norm of $\vec{\boldsymbol{x}}$ and its FFT along the third dimension $\widehat{\vec{\boldsymbol{x}}}$,

$$\|\vec{\boldsymbol{x}}\|_{2^*} = \frac{1}{\sqrt{n_3}}\|\widehat{\vec{\boldsymbol{x}}}\|_{2^*}\ , \tag{2.15}$$

where $1/\sqrt{n_3}$ is the normalization constant.

Definition 2.0.14. ***(Tensor Frobenius norm)** The induced Frobenius norm from the inner product defined above is given by,*

$$\|\boldsymbol{\mathcal{A}}\|_F = \langle \boldsymbol{\mathcal{A}}, \boldsymbol{\mathcal{A}}\rangle^{1/2} = \frac{1}{\sqrt{n_3}}\|\widehat{\boldsymbol{\mathcal{A}}}\|_F = \sqrt{\sum_i\sum_j\sum_k \boldsymbol{\mathcal{A}}_{ijk}^2}$$

Definition 2.0.15. ***(Tubal nuclear norm [50, 25])** The tubal nuclear norm of a tensor $\boldsymbol{\mathcal{A}}$, denoted as $\|\boldsymbol{\mathcal{A}}\|_{TNN}$, is the sum of singular values of all the frontal*

slices of $\widehat{\boldsymbol{\mathcal{A}}}$, and is a convex relaxation of the tensor tubal-rank [22]. In particular,

$$\|\boldsymbol{\mathcal{A}}\|_{TNN} = \|\overline{\boldsymbol{\mathcal{A}}}\|_* \tag{2.16}$$

Definition 2.0.16. ***(Tensor spectral norm)*** *The tensor spectral norm $\|\boldsymbol{\mathcal{A}}\|$ of a third-order tensor $\boldsymbol{\mathcal{A}}$ is defined as the largest singular value of $\boldsymbol{\mathcal{A}}$. Moreover,*

$$\|\boldsymbol{\mathcal{A}}\| = \|\overline{\boldsymbol{\mathcal{A}}}\| \tag{2.17}$$

i.e. the tensor spectral norm of $\boldsymbol{\mathcal{A}}$ equals to the matrix spectral norm of $\overline{\boldsymbol{\mathcal{A}}}$.

Definition 2.0.17. ***(Tensor operator norm)*** *Suppose $\mathfrak{L}$ is a tensor operator defined via t-product,*

$$\mathfrak{L}(\boldsymbol{\mathcal{X}}) = \boldsymbol{\mathcal{L}} * \boldsymbol{\mathcal{X}} \tag{2.18}$$

where $\boldsymbol{\mathcal{L}} \in \mathbb{R}^{k\times n_1\times n_3}$ and $\boldsymbol{\mathcal{X}} \in \mathbb{R}^{n_1\times n_2\times n_3}$ are third-order tensors. Then the operator norm of $\mathfrak{L}$ is defined as,

$$\|\mathfrak{L}\|_{op} = \sup_{\boldsymbol{\mathcal{X}}:\|\boldsymbol{\mathcal{X}}\|_F\leq 1} \|\mathfrak{L}(\boldsymbol{\mathcal{X}})\|_F\,, \tag{2.19}$$

which is consistent with matrix case, where the spectral norm is equivalent to the operator norm $\|\mathfrak{L}\|_{op} = \|\boldsymbol{\mathcal{L}}\|$.

Definition 2.0.18. ***(Tensor infinity norm)*** *The tensor infinity norm $\|\boldsymbol{\mathcal{A}}\|_\infty$ is defined as follows:*

$$\|\boldsymbol{\mathcal{A}}\|_\infty = \max_{i,j,k} |\boldsymbol{\mathcal{A}}_{ijk}| \tag{2.20}$$

which is the entry with the largest absolute value of $\boldsymbol{\mathcal{A}}$.

2.1 Outline of This Thesis

The remainder of the thesis is organized as follows. In Chapter 2 we introduce the problem of tensor completion under limited samples. We show that by solving a convex optimization problem, which minimizes a convex surrogate to the tensor tubal-rank, one can guarantee exact recovery with high probability as long as number of samples is of the order $O(rnk\log(nk))$ given a tensor of size $n \times n \times k$ with tubal-rank r. We then give out the t-SVD based tensor completion algorithm and show that it is in practice working better than classic matrix completion dealing with horizontal panning video. Chapter 3 describes the problem tensor Robust PCA. In this chapter we introduce two types of robust PCA: the batch tensor RPCA and online tensor RPCA. We show that tensor robust PCA methods work very well in denoising and background separation. Chapter 4 is devoted in studying the tensor dictionary learning problems. We first introduce the t-linear combination of tensor dictionaries and coefficients, then propose the K-TSVD algorithm to learn the high dimensional dictionaries. As applications we show the learned dictionary perform very well in video/image completion and denoising, compared to the state-of-the-art dictionary learning techniques. In Chapter 5, we conclude the thesis, and give some discussions about future directions.

Chapter 3

Tensor Completion

3.1 Introduction

[1]Recovery of multidimensional array of numbers or *tensors*[2] under limited number of measurements is an important problem, which arises in a variety of applications, such as recommendation systems [52], dimensionality reduction [53], multi-class learning [54], data mining [55, 56], and computer vision [57, 22].

The strategies and performance bounds for sampling and recovery of tensors rest heavily on the framework used to reveal a low complexity algebraic structure in the data, namely a *low-rank* decomposition. For example, for matrix data (a 2-dimensional tensor) when treated as a linear operator over a vector space, one defines the rank of the matrix via its minimal decomposition into a sum of rank-1 matrices.

[1]This research was supported in part by the National Science Foundation grant NSF:1319653.

[2]This terminology comes from representation of multilinear functionals on the outer product of finite dimensional vector spaces as an indexed array of numbers, for example see [51].

This is well-known to be obtained via the Singular Value Decomposition (SVD). In this case, it has been shown that as long as the left and right singular vectors are incoherent with the standard basis, a random sampling strategy with sampling complexity in proportion to the complexity of the decomposition is sufficient for recovery by solving a convex optimization problem, namely minimizing the nuclear norm of the matrix [58, 59].

For N-D tensors with $N \geq 3$ (**Note**: N is also referred to as the order of the tensor, and we will often refer to an N-D tensor as an order-N tensor), there are several ways to define algebraic complexity using the framework of classical multilinear algebra, where tensors are treated as multilinear operators over tensor product or outer product of vector spaces [51]. In this framework, decomposition of a tensor as sum of rank-1 *outer products* is referred to as CANDECOMP/PARAFAC (CP) [29] factorization and the minimal number of such factors required is referred to as the CP rank. However there are known computational and ill-posedness issues with CP [60]. Other kinds of decompositions, such as Tucker, Hierarchical-Tucker (H-Tucker) and Tensor Train (TT) [61], are also shown to reveal the algebraic structure in the data with the notion of rank extended to *multi-rank*, expressed as a vector of ranks of the factors in the contracted representation using matrix product states. In this context, to the best of our knowledge, provably optimal approaches based on Tucker decomposition work by *matricizing* the tensor in various ways, subsequently employing the theory and methods for matrix completion, see for example [62, 63].

This paper considers sampling and recovery for 3-D tensors using the algebraic framework proposed in [1, 2, 3, 4]. In this framework, 3-D tensors are treated as linear operators over 2-D tensors and one obtains an SVD-like factorization referred to as the tensor-SVD (t-SVD). Using this factorization, one can define a notion of rank, referred to as the tubal-rank. This algebraic framework is essentially based on a group theoretic approach where the multidimensional structure is unraveled by constructing group-rings along the tensor fibers [64]. This approach has recently been extended in [49] to construct a Banach algebra along tensor fibers. In this paper, we restrict ourselves to group rings constructed out of cyclic groups and also omit consideration of the extensions carried out in [49]. Nevertheless, the results presented here can be generalized to these settings.

It is important to note that the t-SVD algebraic framework is different from the classic multilinear algebraic framework for tensor decompositions [51]. Therefore, the notion of tensor rank using the t-SVD, namely the tubal-rank, differs from the CP rank and the Tucker rank. Hence, bounds and conditions for tensor completion for low CP rank and low Tucker rank tensors are not *directly* comparable to results in this paper.

The t-SVD has been recently exploited in [22] for the problem of 3-D tensor recovery from limited sampling with applications to computer vision. In the present paper, we derive theoretical performance bounds for the tensor recovery algorithm proposed in [22]. In this context, our work is greatly inspired by [59, 65], in which the

main tool, namely the Non-commutative Bernstein Inequality (NBI), is also helpful in deriving our results. We prove that with high probability one can exactly recover a tensor of size $n_1 \times n_2 \times n_3$ with tubal-rank r (as derived from the t-SVD), by solving a convex optimization problem, given $\mathcal{O}(rn_1n_3 \log((n_1 + n_2)n_3))$ samples when certain tensor incoherence conditions are satisfied. The notions of tensor incoherence and results are novel, and we show that, while related, they are *not* directly implied by the results in matrix completion using the standard matrix incoherence conditions.

In order to put our work into perspective and highlight our contributions, we now go over related work on tensor completion using different tensor factorizations and contrast our findings with existing literature.

3.1.1 Related Work

Apart from the t-SVD, there are two major types of low-rank tensor completion methods considered in the literature: methods that are based on the CP decomposition, and those that are based on the Tucker decomposition. The sampling methods include random downsampling, Gaussian measurements and adaptive sampling. We summarize these results in **Table** 3.1. Below we will provide details for each of these methods.

Table 3.1: A summary of existing tensor completion methods

Format	**Sampling Method**	**Samples needed for exact recovery (3rd-order tensor of size $n \times n \times n$)**	**Incoherent condition**
CP[66]	Gaussian measurements	$O(rn^2)$ for CP rank r	N/A
CP[40]	Random sampling	$O(n^{3/2}r^5 \log^4(n))$ for CP rank r on symmetric tensors	Incoherence condition of symmetric tensors with orthogonal decomposition
Tucker[66]	Gaussian measurements	$O(rn^2)$ for Tucker rank (r, r, r)	N/A
Tucker[62]	Random sampling	$O(rn^2 \log^2(n))$ for Tucker rank (r, r, r)	Matrix incoherence condition on all mode-n unfoldings
CP[67]	Adaptive Sampling	$O(nr \log(r))$ for CP rank r	Standard incoherence condition with orthogonal decomposition
t-SVD	Random sampling	$O(rn^2 \log(n))$ for Tensor tubal-rank r	Tensor incoherence condition

3.1.1.1 Tensor Completion Based on CP decomposition

The CP decomposition of an order-N tensor $\mathcal{X} \in \mathbb{R}^{n_1 \times n_2 \times \ldots \times n_N}$ is given by,

$$\mathcal{X} = \sum_{\ell=1}^{r} \mathbf{x}_\ell^{(1)} \circ \mathbf{x}_\ell^{(2)} \circ \cdots \circ \mathbf{x}_\ell^{(N)}, \;\; \mathbf{x}_\ell^{(i)} \in \mathbb{R}^{n_i}, i = 1, 2, ..., N \;, \tag{3.1}$$

where $\circ$ denotes the outer product [29]. The smallest r such that Equation (3.1) holds is called the CP rank of $\mathcal{X}$.

Suppose we sample $\mathcal{X}$ based on an index set Ω. Let P_Ω be the orthogonal projection onto Ω. Then in [39] the authors propose to complete the tensor by solving the following optimization problem,

$$\min \|P_\Omega(\mathcal{A} - \sum_{\ell=1}^{r} \mathbf{x}_\ell^{(1)} \circ \mathbf{x}_\ell^{(2)} \circ ... \circ \mathbf{x}_\ell^{(N)})\|_2^2 + \lambda \sum_{\ell=1}^{r} \sum_{i=1}^{N} \|\mathbf{x}_\ell^{(i)}\|_2^2 \;, \tag{3.2}$$

where $\lambda \geq 0$ is the regularization parameter. However, this approach has several drawbacks. The optimization problem is non-convex and hence only local minima can be guaranteed. Further, for practical problems it is often computationally difficult to determine the CP rank or the best low rank CP approximation of a tensor data beforehand. Recently in [40] it was shown that one can provably recover an $n \times n \times n$ *symmetric* tensor with CP rank r from $O(n^{3/2} r^5 \log^4 n)$ randomly sampled entries under standard incoherence conditions on the factors[3].

[3] A tensor $\mathcal{X} \in \mathbb{R}^{n \times n \times n}$ is called symmetric in the CP format if its CP decomposition has the format $\mathcal{X} = \sum_{\ell=1}^{r} \sigma_\ell (\mathbf{u}_\ell \circ \mathbf{u}_\ell \circ \mathbf{u}_\ell)$, where $\mathbf{u}_\ell \in \mathbb{R}^n$ with $\|\mathbf{u}_\ell\| = 1$.

3.1.1.2 Tensor Completion Based on Tucker Decomposition

In [68] tensor completion based on minimizing a convex surrogate to the tensor n-rank is proposed. Tensor n-rank is the sum of the ranks of matrices obtained by the matricizations of the tensor, i.e. it is the sum of the Tucker ranks. Specifically one solves for,

$$\begin{aligned} &\min_{\boldsymbol{\mathcal{X}}} \sum_{i=1}^{N} \alpha_i \|\boldsymbol{\mathcal{X}}_{(i)}\|_* \\ &\text{subject to } P_\Omega(\boldsymbol{\mathcal{X}}) = P_\Omega(\boldsymbol{\mathcal{T}}) \ , \end{aligned} \tag{3.3}$$

where $\boldsymbol{\mathcal{X}}_{(i)}$ denotes the mode-i matricization of $\boldsymbol{\mathcal{X}}$ [29], α_i are pre-specified positive constants satisfying $\sum_{i=1}^{N} \alpha_i = 1$, and $\|\cdot\|_*$ denotes the matrix nuclear norm. However, no theoretical guarantees for recovery are provided and it is not clear how to optimally choose the weights α_i's. Normally one ends up choosing the best matricization that is determined empirically, which turns it into a matrix completion problem. In [69] a tighter convex relaxation for the tensor n-rank is proposed, but again no provable recovery bounds are provided.

In [62] the authors solve the following convex problem,

$$\begin{aligned} &\min_{\boldsymbol{\mathcal{X}},\boldsymbol{\mathcal{E}}} \sum_{i=1}^{N} \alpha_i \|\boldsymbol{\mathcal{X}}_{(i)}\|_* + \|\boldsymbol{\mathcal{E}}\|_1 + \frac{\tau}{2}\|\boldsymbol{\mathcal{X}}\|_F^2 + \frac{\tau}{2}\|\boldsymbol{\mathcal{E}}\|_F^2 \\ &\text{s.t. } P_\Omega(\boldsymbol{\mathcal{X}} + \boldsymbol{\mathcal{E}}) = \boldsymbol{\mathcal{B}} \ , \end{aligned} \tag{3.4}$$

for some specific choices for α_i and τ. This can be viewed as a combination of the matrix completion and the matrix Robust Principal Component Analysis (RPCA) when extended to the case of tensors. It is shown that if the tensor satisfies the matrix incoherence conditions under all its matricizations, then solving for the above

optimization problem leads to accurate recovery if the number of samples exceeds that required for completion under each matricization.

3.1.1.3 Tensor Completion under Gaussian Measurements

Instead of random sampling, in [66] a different method for low CP or Tucker rank tensor completion under Gaussian measurements is proposed. The main idea is to *reshape* the tensor $\boldsymbol{\mathcal{X}} \in \mathbb{R}^{n_1 \times n_2 \times n_3 \times \ldots \times n_N}$ into a *square* matrix $\boldsymbol{\mathcal{X}}_{[j]} \in \mathbb{R}^{n_1 \ldots n_j \times n_{j+1} \ldots n_N}$ and apply matrix completion on it. If $\boldsymbol{\mathcal{X}}$ is a low-rank tensor (in either CP or Tucker sense), $\boldsymbol{\mathcal{X}}_{[j]}$ will be a low-rank matrix. It is shown that if $\boldsymbol{\mathcal{X}}_0$ has CP rank r, then $O(rn^{\lceil N/2 \rceil})$ Gaussian samples are sufficient to recover the original tensor. If $\boldsymbol{\mathcal{X}}_0$ has Tucker rank $(r, r, ..., r)$, then $O(r^{\lfloor N/2 \rfloor} n^{\lceil N/2 \rceil})$ Gaussian samples are needed.

3.1.1.4 Tensor Completion via Adaptive Sampling

In [67] a tensor completion approach based on adaptive sampling is developed. The key idea is to predict the tensor singular subspace given the sampled sub-tensor, and recursively update the subspace if a newly sampled sub-tensor lies out of it. It is shown that $O(nr^{3/2} \log r)$ adaptively chosen samples are sufficient for exact recovery of an $n \times n \times n$ tensor with CP rank r. This approach extends the matrix completion to the tensor case and yields a tighter bound, requiring only column incoherence conditions.

3.2 Tensor Completion via T-SVD

In this section, we will formally define the sampling model and the problem of tensor completion. Our main result is stated in Theorem 3.2.1.

3.2.1 Tensor Completion with Random Sampling

Given a third-order tensor $\boldsymbol{\mathcal{M}} \in \mathbb{R}^{n_1 \times n_2 \times n_3}$ of tubal-rank r, suppose there are m entries in $\boldsymbol{\mathcal{M}}$ sampled according to the Bernoulli model, which means each entry in the tensor is sampled with probability p independent of the others. The task of tensor completion problem is to recover $\boldsymbol{\mathcal{M}}$ from the observed entries.

In this paper, we follow the approach taken in [22] which solves the following convex optimization problem for tensor completion,

$$\begin{aligned} &\min_{\boldsymbol{\mathcal{X}}} \ \|\boldsymbol{\mathcal{X}}\|_{TNN} \\ &\text{subject to } \boldsymbol{\mathcal{X}}_{ijk} = \boldsymbol{\mathcal{M}}_{ijk}, (i,j,k) \in \Omega \end{aligned} \tag{3.5}$$

where Ω is the index set of observed entries. We will analyze the sufficient conditions under which, the solution to (3.5) is equal to $\boldsymbol{\mathcal{M}}$.

Before we state our main result, we need to introduce the notion of tensor incoherence, a condition that is required for the results to hold true under random sampling. Similar to the matrix completion case, recovery under random sampling is hopeless if most of the entries are equal to zero [58]. For tensor completion using t-SVD, if tensor $\boldsymbol{\mathcal{M}}$ is sparse, then in the reduced t-SVD of $\boldsymbol{\mathcal{M}} = \boldsymbol{\mathcal{U}} * \boldsymbol{\mathcal{S}} * \boldsymbol{\mathcal{V}}^\top$, the tensors $\boldsymbol{\mathcal{U}}$ and

$\boldsymbol{\mathcal{V}}$ will be highly concentrated on the tensor basis. Similar to the case of matrix completion [58], it is required that the tensor columns, $\boldsymbol{\mathcal{U}}(:,i,:)$ and $\boldsymbol{\mathcal{V}}(:,i,:), i = 1,2,...,r$ be sufficiently spread, i.e. should be uncorrelated with the tensor basis, for recovery under random sampling. This intuition motivates the following *tensor incoherence condition.*

Definition 3.2.1. *(Tensor Incoherence Condition) Let the reduced t-SVD of a tensor* $\boldsymbol{\mathcal{M}}$ *be* $\boldsymbol{\mathcal{U}} * \boldsymbol{\mathcal{S}} * \boldsymbol{\mathcal{V}}^\top$. $\boldsymbol{\mathcal{M}}$ *is said to satisfy the* tensor incoherence condition*, if there exists a* $\mu_0 > 0$ *such that*

$$\begin{aligned} \max_{i=1,\dots,n_1} \left\| \boldsymbol{\mathcal{U}}^\top * \vec{\boldsymbol{e}}_i \right\|_{2^*} &\leq \sqrt{\frac{\mu_0 r}{n_1}}, \\ \max_{j=1,\dots,n_2} \left\| \boldsymbol{\mathcal{V}}^\top * \vec{\boldsymbol{e}}_j \right\|_{2^*} &\leq \sqrt{\frac{\mu_0 r}{n_2}}, \end{aligned} \tag{3.6}$$

where $\vec{\boldsymbol{e}}_i$ *is the* $n_1 \times 1 \times n_3$ *column basis with* $\vec{\boldsymbol{e}}_{i11} = 1$ *and* $\vec{\boldsymbol{e}}_j$ *is the* $n_2 \times 1 \times n_3$ *column basis with* $\vec{\boldsymbol{e}}_{j11} = 1$*,* $i = 1,2,...,n_1, j = 1,2,...n_2$.

Note that the smallest μ_0 is equal to 1 and this value is achieved when each tensor column $\vec{\boldsymbol{u}}_i = \boldsymbol{\mathcal{U}}(:,i,:)$ has entries with magnitude $1/\sqrt{n_1 n_3}$, or each tensor column $\vec{\boldsymbol{v}}_i = \boldsymbol{\mathcal{V}}(:,i,:)$ has entries with magnitude $1/\sqrt{n_2 n_3}$. The largest possible value of μ_0 is $\min(n_1, n_2)/r$ when one of the tensor columns of $\boldsymbol{\mathcal{U}}$ (or $\boldsymbol{\mathcal{V}}$ respectively) is equal to the tensor column basis $\vec{\boldsymbol{e}}_i$ (or $\vec{\boldsymbol{e}}_j$ resp.). With low μ_0, each entry of $\boldsymbol{\mathcal{M}}$ carries approximately same amount of information.

In [58] for matrix completion case, another joint incoherence condition is needed, which bounds the maximum (absolute value) entry of $UV^\top$, where U and V correspond to left and right singular vectors in the SVD of the matrix. However, this joint

incoherence condition is regarded unintuitive and restrictive. In [65], the authors successfully removed this joint incoherence by using the $\ell_{\infty,2}$ norm to get a similar bound in the dual certificate step. In our tensor completion case, we apply this idea to our set-up and successfully avoid the joint incoherence condition. Now we will present our main result.

Theorem 3.2.1. *Suppose* $\boldsymbol{\mathcal{M}}$ *is an* $n_1 \times n_2 \times n_3$ *tensor and its reduced t-SVD is given by* $\boldsymbol{\mathcal{M}} = \boldsymbol{\mathcal{U}} * \boldsymbol{\mathcal{S}} * \boldsymbol{\mathcal{V}}^\top$ *where* $\boldsymbol{\mathcal{U}} \in \mathbb{R}^{n_1 \times r \times n_3}$, $\boldsymbol{\mathcal{S}} \in \mathbb{R}^{r \times r \times n_3}$ *and* $\boldsymbol{\mathcal{V}} \in \mathbb{R}^{n_2 \times r \times n_3}$. *Suppose* $\boldsymbol{\mathcal{M}}$ *satisfies the tensor incoherence condition(3.6) with parameter* $\mu_0 > 0$*, then there exists constants* $c_0, c_1, c_2 > 0$ *such that if*

$$p \geq c_0 \frac{\mu_0 r \log(n_3(n_1 + n_2))}{\min\{n_1, n_2\}}, \tag{3.7}$$

then $\boldsymbol{\mathcal{M}}$ *is the unique minimizer to (3.5) with probability at least* $1 - c_1((n_1+n_2)n_3)^{-c_2}$.

Note that the sampling model we use here is the Bernoulli model. There are some other widely used models including sampling with or without replacement. For matrix completion, the recovery guarantees for different models are consistent with only a change of the constants in the sampling complexity and recovery guarantees [70, 71] and we expect them to be the same in our case as well.

Note that although the proof of **Theorem** 3.2.1 follows closely the proof of matrix completion under various measurement models, there are some subtle differences. First of all, we are sampling in the *original* domain, while the tubal nuclear norm (TNN) is defined in the Fourier domain. In fact, let $\boldsymbol{\mathcal{P}}_\Omega(\boldsymbol{\mathcal{Z}})$ denote the same size

tensor as $\mathcal{Z}$ with $\mathcal{P}_\Omega(\mathcal{Z})_{ijk} = \mathcal{Z}_{ijk}$ if $(i,j,k) \in \Omega$ and zero otherwise. Then (3.5) can be rewritten as,

$$\min_{\mathcal{X}} \ \|\mathcal{X}\|_{TNN}$$
$$\text{subject to } \mathcal{P}_\Omega(\mathcal{X}) = \mathcal{P}_\Omega(\mathcal{M}) \ ,$$

Note that it is equivalent to the following

$$\begin{aligned} &\min_{\mathcal{X}} \ \|\overline{\mathcal{X}}\|_* \\ &\text{subject to } \mathcal{B}\mathcal{P}\mathcal{B}^{-1}(\overline{\mathcal{X}}) = \mathcal{B}\mathcal{P}\mathcal{B}^{-1}(\overline{\mathcal{M}}) \ , \end{aligned} \tag{3.8}$$

where $\mathcal{B}$ is a mapping which maps a third order tensor $\mathcal{Z}$ to $\overline{\mathcal{Z}}$, and $\mathcal{B}^{-1}$ is its inverse transform. Now the above problem is a matrix completion problem under linear constraint and it is completely defined in the Fourier domain.

We can regard this constraint as several observations in the form of inner products of $\overline{\mathcal{X}}$ and $\overline{\mathcal{E}_{ijk}}$ in the Fourier domain, where $\mathcal{E}_{ijk}$ is a unit tensor with only (i,j,k)th entry being 1 and $\overline{\mathcal{E}_{ijk}} = \texttt{fft}(\mathcal{E}_{ijk}, [\,], 3)$. In details, the problem can be equivalently rewritten as,

$$\begin{aligned} &\min \ \|\overline{\mathcal{X}}\|_* \\ &\text{subject to } \ \langle \overline{\mathcal{X}}, \overline{\mathcal{E}_{ijk}} \rangle = \langle \overline{\mathcal{M}}, \overline{\mathcal{E}_{ijk}} \rangle, \ \ (i,j,k) \in \Omega \\ &\qquad\qquad \overline{\mathcal{X}} \text{ is block diagonal (block size } n \times n). \end{aligned} \tag{3.9}$$

We now note that [72] gives provable guarantees on recovery for such cases, but without the block diagonal constraint, provided that, (a) the basis are orthonormal, and (b) the basis and matrix satisfy a standard incoherence condition and an extra joint incoherence condition.

However, because of the block diagonal constraint, the result of [72] cannot be directly applied to derive provable bounds to our problem even if we are given an extra joint incoherence condition. In fact, following this formula we can also change the block diagonal constraint in (3.9) into some more inner product observations, which force the entries outside the block diagonal positions to be zero. But this actually needs to observe even more entries ($O(n^2 n_3^2)$) than the data tensor itself to finish the recovery. On the other hand, we also cannot solve (3.9) slice-wise (which avoids the block diagonal constraint) using matrix completion results, since the constraint $\langle \overline{\boldsymbol{\mathcal{X}}}, \overline{\boldsymbol{\mathcal{E}}_{ijk}} \rangle = \langle \overline{\boldsymbol{\mathcal{M}}}, \overline{\boldsymbol{\mathcal{E}}_{ijk}} \rangle$ is not separable on each frontal slice. As we will see in the next section, in the case of *random tubal sampling*, the tensor completion problem can be separated into individual matrix completion problem on each frontal slice, with random sampling in the Fourier domain.

3.2.2 Tensor Completion with Random Tubal Sampling

Another way to sample a tensor is to perform random or adaptive tubal sampling as considered in [73] for fingerprinting application. Here we will comment on the theoretical guarantees for recovery under random tubal sampling, since [73] did not specifically analyze this. Instead of randomly sampling entries of a third-order tensor $\boldsymbol{\mathcal{M}}$ as in the previous subsection, one can randomly sample tensor tubes along the

third dimension. Then the completion problem becomes,

$$\begin{aligned} &\min_{\mathcal{X}} \ \|\mathcal{X}\|_{TNN} \\ &\text{subject to } \mathcal{X}_{ijk} = \mathcal{M}_{ijk}, (i,j) \in \Omega, k = 1,2,...,n_3 \end{aligned} \tag{3.10}$$

where Ω is the index set of observed tubes. If we take the FFT of $\mathcal{X}$ and $\mathcal{M}$ along the third dimension,

$$\begin{aligned} &\min_{\widehat{\mathfrak{X}}^{(k)}} \ \sum_{k=1}^{n_3} \|\widehat{\mathcal{X}}^{(k)}\|_* \\ &\text{subject to } \widehat{\mathcal{X}}_{ijk} = \widehat{\mathcal{M}}_{ijk}, (i,j) \in \Omega, k = 1,2,...,n_3 \end{aligned} \tag{3.11}$$

then it is easy to see that solving the above optimization problem is equivalent to solving n_3 matrix completion problems in the Fourier domain,

$$\begin{aligned} &\min_{\widehat{\mathfrak{X}}^{(k)}} \ \|\widehat{\mathcal{X}}^{(k)}\|_* \\ &\text{subject to } \widehat{\mathcal{X}}^{(k)}_{ij} = \widehat{\mathcal{M}}^{(k)}_{ij}, (i,j) \in \Omega \end{aligned} \tag{3.12}$$

for $k = 1,2,...,n_3$. Therefore tensor completion problem with tubal sampling is essentially the matrix completion from random samplings of each frontal slice in the Fourier domain. Then we can directly use the result of matrix completion here.

Suppose there are p third-dimensional tubes of $\mathcal{M} \in \mathbb{R}^{n_1 \times n_2 \times n_3}$ sampled according to the Bernoulli model, which means each tube in the tensor is sampled with probability p independent of the other tubes. Then we have the following theorem,

Theorem 3.2.2. *[73] Let $\mathcal{M}$ be an $n_1 \times n_2 \times n_3$ tensor and its reduced t-SVD is given by $\mathcal{M} = \mathcal{U} * \mathcal{S} * \mathcal{V}^\top$ where $\mathcal{U} \in \mathbb{R}^{n_1 \times r \times n_3}$, $\mathcal{S} \in \mathbb{R}^{r \times r \times n_3}$ and $\mathcal{V} \in \mathbb{R}^{n_2 \times r \times n_3}$. Suppose each frontal slice $\widehat{\mathcal{M}}^{(i)}$ satisfies the matrix weak incoherence condition (3.13)*

with parameter $\mu_0 > 0$. Then there exists constants $c_0, c_1, c_2 > 0$ such that if

$$p \geq c_0 \frac{\mu_0 r \log^2(n_1 + n_2)}{min\{n_1, n_2\}},$$

then $\mathfrak{M}$ is the unique minimizer to (3.5) with probability at least $1 - c_1 n_3 (n_1 + n_2)^{-c_2}$.

Remark 3.2.1. ***(A comment on the incoherence conditions for tubal sampling)*** *In Theorem 3.2.2 each slice $\widehat{\mathfrak{M}}^{(k)}$ in the Fourier domain needs to satisfy* matrix weak incoherence condition *[65] with parameter μ_0. That is for all $k \in \{1, 2, ..., n_3\}$,*

$$\begin{aligned} \max_{i=1,2,..,n_1} \|\widehat{\mathcal{U}}^{(k)\top} \mathbf{e}_i\|_2 &\leq \sqrt{\frac{\mu_0 r}{n_1}}, \\ \max_{j=1,2,..,n_2} \|\widehat{\mathcal{V}}^{(k)\top} \mathbf{e}_j\|_2 &\leq \sqrt{\frac{\mu_0 r}{n_2}}, \end{aligned} \tag{3.13}$$

where $\mathbf{e}_i$ denotes the i-th standard basis in $\mathbb{R}^n$. Note that we have $1 \leq \mu_0 \leq \min\{n_1, n_2\}/r$. We now show that the matrix incoherence condition of Equation (3.13) is not equivalent to the tensor incoherence conditions of Equations (3.6). In fact from Equations (3.13) we have,

$$\max_{i \in \{1,2,..,n_1\}} \|\widehat{\mathcal{U}}^{(k)\top} \mathbf{e}_i\|_2 \leq \sqrt{\frac{\mu_0 r}{n_1}} \tag{3.14}$$

$$\implies \max_{i=1,2,..,n_1} \sum_{k=1}^{n_3} \|\widehat{\mathcal{U}}^{(k)\top} \mathbf{e}_i\|_2^2 \leq \frac{n_3 \mu_0 r}{n_1} \tag{3.15}$$

$$\iff \max_{i=1,2,..,n_1} \|\widehat{\mathcal{U}}_i^{\top}\|_{2^*}^2 \leq \frac{n_3 \mu_0 r}{n_1}$$

$$\iff \max_{i=1,2,..,n_1} \left\|\mathcal{U}^{\top} * \vec{\boldsymbol{e}}_i\right\|_{2^*} \leq \sqrt{\frac{\mu_0 r}{n_1}}.$$

Similarly we can get,

$$\max_{j=1,...,n_2} \left\|\mathcal{V}^{\top} * \vec{\boldsymbol{e}}_j\right\|_{2^*} \leq \sqrt{\frac{\mu_0 r}{n_2}} \ ,$$

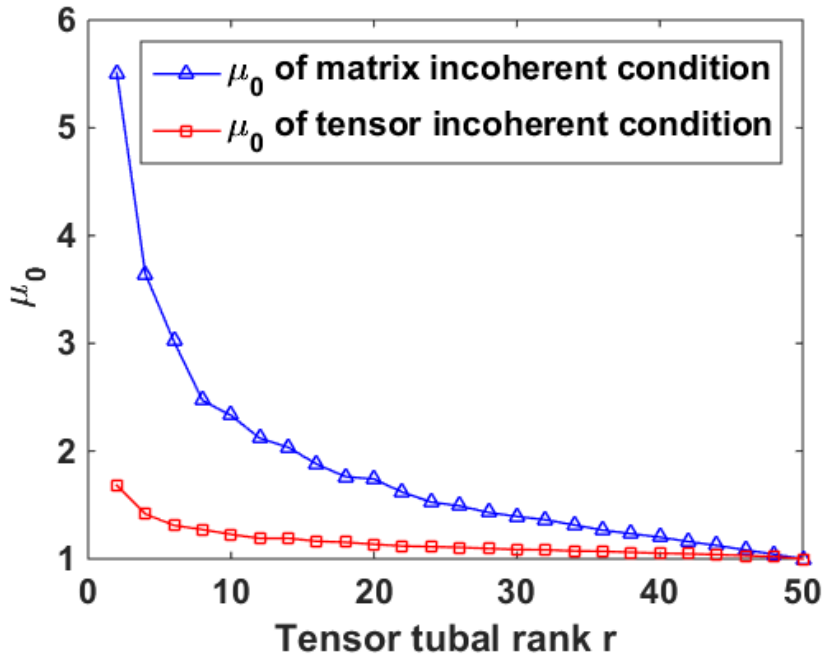


Figure 3.1: Comparison of tensor and matrix incoherent condition on $50 \times 50 \times 20$ tensor.

which is exactly the tensor incoherence condition. Therefore our tensor incoherence condition can be obtained from the matrix incoherence condition, but not vice versa since (3.15) does not imply (3.14).

Figure 3.1 shows a comparison of tensor and matrix incoherence condition. Each time we randomly generate a $50 \times 50 \times 20$ *tensor with different tubal-rank and compute* μ_0 *of both cases. We repeat this process* 20 *times and plot the average* μ_0 *in the figure. As we can tell,* μ_0 *of tensor incoherence condition is indeed lower than that of matrix incoherence condition. In the worst case, when tensor tubal-rank is small, the two incoherence constants can differ by an order of* $\sqrt{n_3}$.

3.3 Main proof

In this section, we provide the proof of Theorem 3.2.1. The main idea is to use convex analysis to derive conditions under which one can verify whether $\boldsymbol{\mathcal{M}}$ is the unique minimum tubal nuclear norm solution to (3.5), and then to explicitly show that such conditions are met with high probability under the conditions of Theorem 3.2.1.

To simplify the notation and without loss of generality we assume $n_1 = n_2 = n$ and do not put any assumption on n_3.

Before continuing, some notations used in the proof should be clarified. A tensor $\boldsymbol{\mathcal{Y}}$ is the subgradient of $\|\cdot\|_{TNN}$ at $\boldsymbol{\mathcal{M}}_0$ (denoted $\boldsymbol{\mathcal{Y}} \in \partial\|\boldsymbol{\mathcal{M}}_0\|_{TNN}$), if for all $\boldsymbol{\mathcal{X}} \in \mathbb{R}^{n\times n\times n_3}$,

$$\|\overline{\boldsymbol{\mathcal{X}}}\|_* \geq \|\overline{\boldsymbol{\mathcal{M}}}_0\|_* + \langle \overline{\boldsymbol{\mathcal{Y}}}, \overline{\boldsymbol{\mathcal{X}}} - \overline{\boldsymbol{\mathcal{M}}}_0 \rangle \,. \tag{3.16}$$

Recall that a matrix $\mathbf{Y}$ is a subgradient of a convex function $f : \mathbb{R}^{n_1\times n_2} \to \mathbb{R}$ at matrix $\mathbf{M}_0$ if

$$f(\mathbf{X}) \geq f(\mathbf{M}_0) + \langle \mathbf{Y}, \mathbf{X} - \mathbf{M}_0 \rangle \,.$$

Moreover, $\mathbf{Y}$ is a subgradient of the nuclear norm at $\mathbf{M}_0$ if and only if $\mathbf{Y}$ is of the form [74]

$$\mathbf{Y} = \mathbf{U}\mathbf{V}^\top + \mathbf{W} \,, \tag{3.17}$$

where $\mathbf{U}\mathbf{S}_0\mathbf{V}^\top = \text{svd}(\mathbf{M}_0)$ is the singular value decomposition of $\mathbf{M}_0$, and $\mathbf{W}$ satisfies

1. $\mathbf{U}^\top\mathbf{W} = 0, \;\; \mathbf{W}\mathbf{V} = 0$

2. $\|\mathbf{W}\| \leq 1$.

Similarly, let $\mathcal{Y} = \mathcal{U} * \mathcal{V}^\top + \mathcal{W}$, where $\mathcal{U} \in \mathbb{R}^{n\times r\times n_3}, \mathcal{S}_0 \in \mathbb{R}^{r\times r\times n_3}, \mathcal{V} \in \mathbb{R}^{n\times r\times n_3}$ is the t-SVD of $\mathcal{M}_0 = \mathcal{U} * \mathcal{S} * \mathcal{V}^\top$, and $\|\mathcal{W}\| \leq 1$. One can verify that such a $\mathcal{Y}$ satisfies (3.16), therefore $\mathcal{Y} \in \partial\|\mathcal{M}_0\|_{TNN}$.

In order to proceed, we introduce the orthogonal decomposition $\mathbb{R}^{n\times n\times n_3} = T \oplus T^\perp$, where T is the linear space spanned by the elements of the form $\mathcal{U}(:,k,:) * \vec{\boldsymbol{x}}^\top$ and $\vec{\boldsymbol{y}} * \mathcal{V}(:,k,:)^\top$, where $\vec{\boldsymbol{x}},\ \vec{\boldsymbol{y}} \in \mathbb{R}^{n\times 1\times n_3}$ are arbitrary tensor columns, $k = 1, 2, ..., r$. Let $T^\perp$ be its orthogonal complement. The orthogonal projections $\mathcal{P}_T$ onto T and $\mathcal{P}_{T^\perp}$ onto $T^\perp$ are given as follows,

$$\mathcal{P}_T(\mathcal{Z}) = \mathcal{U} * \mathcal{U}^\top * \mathcal{Z} + \mathcal{Z} * \mathcal{V} * \mathcal{V}^\top - \mathcal{U} * \mathcal{U}^\top * \mathcal{Z} * \mathcal{V} * \mathcal{V}^\top \tag{3.18}$$

$$\mathcal{P}_{T^\perp}(\mathcal{Z}) = \mathcal{Z} - \mathcal{P}_T(\mathcal{Z}) = (\mathcal{I} - \mathcal{U} * \mathcal{U}^\top) * \mathcal{Z} * (\mathcal{I} - \mathcal{V} * \mathcal{V}^\top) \tag{3.19}$$

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

Define a random variable $\delta_{ijk} = \mathbf{1}_{(i,j,k)\in\Omega}$ where $\mathbf{1}_{(\cdot)}$ is the indicator function. Let $\mathcal{R}_\Omega : \mathbb{R}^{n\times n\times n_3} \to \mathbb{R}^{n\times n\times n_3}$ be a random projection as follows,

$$\mathcal{R}_\Omega(\mathcal{Z}) = \sum_{i,j,k} \frac{1}{p}\delta_{ijk}\mathcal{Z}_{ijk}\vec{\boldsymbol{e}}_i * \mathring{\boldsymbol{e}}_k * \vec{\boldsymbol{e}}_j^\top . \tag{3.20}$$

Similar to the matrix completion case, in the following we will construct a *dual certificate* $\mathcal{Y}$ and show that it is close to the subgradient of $\|\mathcal{M}\|_{TNN}$ under certain conditions. In [58], the authors constructed such a certificate via an infinite series using a construction obtained in the compressed sensing literature [75, 76]. Then each term in the series were analyzed individually using some decoupling inequalities.

The following Proposition 3.3.1 and Lemma 3.3.1 directly support the proof of the main theorem. The proofs of these technical and supporting results are provided in the Appendices A-D.

Proposition 3.3.1. *Tensor* $\boldsymbol{\mathcal{M}}$ *is the unique minimizer to (3.5) if the following conditions hold.*

1. $\|\boldsymbol{\mathcal{P}}_T\boldsymbol{\mathcal{R}}_\Omega\boldsymbol{\mathcal{P}}_T - \boldsymbol{\mathcal{P}}_T\|_{op} \leq \frac{1}{2}$

2. *There exists a tensor* $\boldsymbol{\mathcal{Y}}$ *such that* $\boldsymbol{\mathcal{P}}_\Omega(\boldsymbol{\mathcal{Y}}) = \boldsymbol{\mathcal{Y}}$ *and*

   (a) $\|\boldsymbol{\mathcal{P}}_T(\boldsymbol{\mathcal{Y}}) - \boldsymbol{\mathcal{U}} * \boldsymbol{\mathcal{V}}^\top\|_F \leq \frac{1}{4nn_3^2}$

   (b) $\|\boldsymbol{\mathcal{P}}_{T^\perp}(\boldsymbol{\mathcal{Y}})\| \leq \frac{1}{2}$

Lemma 3.3.1. *Suppose* $\|\boldsymbol{\mathcal{P}}_T\boldsymbol{\mathcal{R}}_\Omega\boldsymbol{\mathcal{P}}_T - \boldsymbol{\mathcal{P}}_T\|_{op} \leq \frac{1}{2}$. *Then for any* $\boldsymbol{\mathcal{Z}}$ *such that* $\boldsymbol{\mathcal{P}}_\Omega(\boldsymbol{\mathcal{Z}}) = 0$, *we have*

$$\frac{1}{2}\|\boldsymbol{\mathcal{P}}_{\mathcal{T}^\perp}(\boldsymbol{\mathcal{Z}})\|_{TNN} > \frac{1}{4nn_3}\|\boldsymbol{\mathcal{P}}_T(\boldsymbol{\mathcal{Z}})\|_F\,. \tag{3.21}$$

Proof of Proposition 3.3.1 The main idea is that we want to prove that for any $\boldsymbol{\mathcal{Z}}$ supported in Ω^c, $\|\boldsymbol{\mathcal{M}} + \boldsymbol{\mathcal{Z}}\|_{TNN} > \|\boldsymbol{\mathcal{M}}\|_{TNN}$. To prove this the following three facts are used.

Fact 3.3.1. $\|\boldsymbol{\mathcal{A}}\|_{TNN} = n_3 \sup_{\|\boldsymbol{\mathcal{B}}\|\leq 1}\langle \boldsymbol{\mathcal{A}}, \boldsymbol{\mathcal{B}}\rangle$, *where* $\boldsymbol{\mathcal{A}}, \boldsymbol{\mathcal{B}} \in \mathbb{R}^{n\times n\times n_3}$. *Specifically, if the t-SVD of* $\boldsymbol{\mathcal{A}}$ *is given by* $\boldsymbol{\mathcal{A}} = \boldsymbol{\mathcal{U}} * \boldsymbol{\mathcal{S}} * \boldsymbol{\mathcal{V}}^\top$, *then let* $\boldsymbol{\mathcal{B}} = \boldsymbol{\mathcal{U}} * \boldsymbol{\mathcal{V}}^\top$. *Obviously* $\|\boldsymbol{\mathcal{B}}\| \leq 1$ *and we have* $n_3\langle \boldsymbol{\mathcal{A}}, \boldsymbol{\mathcal{B}}\rangle = trace(\bar{\boldsymbol{\mathcal{S}}}) = \|\boldsymbol{\mathcal{A}}\|_{TNN}$.

Recall that for matrix case, we have $\|\mathbf{A}\|_* = \sup_{\|\mathbf{B}\|\leq 1}\langle \mathbf{A}, \mathbf{B}\rangle$, where $\mathbf{A}, \mathbf{B} \in \mathbb{R}^{n\times n}$. Then the fact comes from,

$$\begin{aligned}\|\boldsymbol{\mathcal{A}}\|_{TNN} =& \|\overline{\boldsymbol{\mathcal{A}}}\|_* \\ =& \sup_{\|\overline{\boldsymbol{\mathcal{B}}}\|\leq 1} \langle \overline{\boldsymbol{\mathcal{A}}}, \overline{\boldsymbol{\mathcal{B}}}\rangle \\ =& n_3 \sup_{\|\boldsymbol{\mathcal{B}}\|\leq 1} \langle \boldsymbol{\mathcal{A}}, \boldsymbol{\mathcal{B}}\rangle .\end{aligned}$$

Define the t-SVD of $\boldsymbol{\mathcal{P}}_{\boldsymbol{\mathcal{T}}^\perp}(\boldsymbol{\mathcal{Z}})$ to be $\boldsymbol{\mathcal{P}}_{\boldsymbol{\mathcal{T}}^\perp}(\boldsymbol{\mathcal{Z}}) = \boldsymbol{\mathcal{U}}_\perp * \boldsymbol{\mathcal{S}}_\perp * \boldsymbol{\mathcal{V}}_\perp^\top$, where $\boldsymbol{\mathcal{Z}} \in \mathbb{R}^{n\times n\times n_3}$ such that $\boldsymbol{\mathcal{P}}_\Omega(\boldsymbol{\mathcal{Z}}) = 0$. Then use the fact above we have

$$\|\boldsymbol{\mathcal{P}}_{\boldsymbol{\mathcal{T}}^\perp}(\boldsymbol{\mathcal{Z}})\|_{TNN} = n_3\langle \boldsymbol{\mathcal{U}}_\perp * \boldsymbol{\mathcal{V}}_\perp^\top, \boldsymbol{\mathcal{P}}_{\boldsymbol{\mathcal{T}}^\perp}(\boldsymbol{\mathcal{Z}})\rangle . \tag{3.22}$$

Fact 3.3.2. $\|\boldsymbol{\mathcal{M}}\|_{TNN} = n_3\langle \boldsymbol{\mathcal{U}} * \boldsymbol{\mathcal{V}}^\top + \boldsymbol{\mathcal{U}}_\perp * \boldsymbol{\mathcal{V}}_\perp^\top, \boldsymbol{\mathcal{M}}\rangle$

Since $\boldsymbol{\mathcal{P}}_T(\boldsymbol{\mathcal{U}}) = \boldsymbol{\mathcal{U}}$ and $\boldsymbol{\mathcal{P}}_{T^\perp}(\boldsymbol{\mathcal{U}}_\perp) = \boldsymbol{\mathcal{U}}_\perp$, we have $\boldsymbol{\mathcal{U}} * \boldsymbol{\mathcal{U}}_\perp^\top = 0$ and similarly $\boldsymbol{\mathcal{V}} * \boldsymbol{\mathcal{V}}_\perp^\top = 0$ by definition. Then the fact can be verified by the following.

$$\begin{aligned}& n_3\langle \boldsymbol{\mathcal{U}} * \boldsymbol{\mathcal{V}}^\top + \boldsymbol{\mathcal{U}}_\perp * \boldsymbol{\mathcal{V}}_\perp^\top, \boldsymbol{\mathcal{M}}\rangle \\ =& n_3\langle \boldsymbol{\mathcal{U}} * \boldsymbol{\mathcal{V}}^\top + \boldsymbol{\mathcal{U}}_\perp * \boldsymbol{\mathcal{V}}_\perp^\top, \boldsymbol{\mathcal{U}} * \boldsymbol{\mathcal{S}} * \boldsymbol{\mathcal{V}}^\top\rangle \\ =& \text{trace}((\overline{\boldsymbol{\mathcal{U}}}\overline{\boldsymbol{\mathcal{V}}}^\top + \overline{\boldsymbol{\mathcal{U}}}_\perp\overline{\boldsymbol{\mathcal{V}}}_\perp^\top)^\top \overline{\boldsymbol{\mathcal{U}}}\overline{\boldsymbol{\mathcal{S}}}\overline{\boldsymbol{\mathcal{V}}}^\top) \\ =& \text{trace}(\bar{\boldsymbol{\mathcal{S}}}) = \|\boldsymbol{\mathcal{M}}\|_{TNN}\end{aligned}$$

Fact 3.3.3. $\|\boldsymbol{\mathcal{U}} * \boldsymbol{\mathcal{V}}^\top + \boldsymbol{\mathcal{U}}_\perp * \boldsymbol{\mathcal{V}}_\perp^\top\| = 1$

Consider a matrix Q such that

$$Q = \bar{u}\bar{v}^\top + \bar{u}_\perp \bar{v}_\perp^\top = \begin{bmatrix} \bar{u} & \bar{u}_\perp \end{bmatrix} \begin{bmatrix} \bar{v}^\top \\ \bar{v}_\perp^\top \end{bmatrix}$$

Since $\bar{u}^\top \bar{u}_\perp = 0$ and $\bar{v}^\top \bar{v}_\perp = 0$, the above expression is the matrix singular value decomposition of Q, so we have

$$\begin{aligned} &\|u * v^\top + u_\perp * v_\perp^\top\| \\ =&\|\bar{u}\bar{v}^\top + \bar{u}_\perp \bar{v}_\perp^\top\| \\ =&\|Q\| = 1 \end{aligned}$$

Now using the above facts, given any $\mathcal{Z} \in \mathbb{R}^{n\times n\times n_3}$ such that $\mathcal{P}_\Omega(\mathcal{Z}) = 0$, we have

$$
\begin{aligned}
&\|\mathcal{M}+\mathcal{Z}\|_{TNN} \\
\geq& n_3\langle \mathcal{U} * \mathcal{V}^\top + \mathcal{U}_\perp * \mathcal{V}_\perp^\top, \mathcal{M}+\mathcal{Z}\rangle && (3.23)\\
=&\|\mathcal{M}\|_{TNN} + n_3\langle \mathcal{U} * \mathcal{V}^\top + \mathcal{U}_\perp * \mathcal{V}_\perp^\top, \mathcal{Z}\rangle \\
=&\|\mathcal{M}\|_{TNN} + n_3\langle \mathcal{U} * \mathcal{V}^\top, \mathcal{P}_T(\mathcal{Z})\rangle + n_3\langle \mathcal{U}_\perp * \mathcal{V}_\perp^\top, \mathcal{P}_{T^\perp}(\mathcal{Z})\rangle \\
=&\|\mathcal{M}\|_{TNN} + n_3\langle \mathcal{U} * \mathcal{V}^\top, \mathcal{P}_T(\mathcal{Z})\rangle + n_3\langle \mathcal{U}_\perp * \mathcal{V}_\perp^\top, \mathcal{P}_{T^\perp}(\mathcal{Z})\rangle - n_3\langle \mathcal{Y}, \mathcal{Z}\rangle && (3.24)\\
=&\|\mathcal{M}\|_{TNN} + n_3\langle \mathcal{U} * \mathcal{V}^\top - \mathcal{P}_T(\mathcal{Y}), \mathcal{P}_T(\mathcal{Z})\rangle + n_3\langle \mathcal{U}_\perp * \mathcal{V}_\perp^\top - \mathcal{P}_{T^\perp}(\mathcal{Y}), \mathcal{P}_{T^\perp}(\mathcal{Z})\rangle \\
=&\|\mathcal{M}\|_{TNN} + \langle \overline{\mathcal{U}}\overline{\mathcal{V}}^\top - \overline{\mathcal{P}_T(\mathcal{Y})}, \overline{\mathcal{P}_T(\mathcal{Z})}\rangle + \|\mathcal{P}_{T^\perp}(\mathcal{Z})\|_{TNN} - \langle \overline{\mathcal{P}_{T^\perp}(\mathcal{Y})}, \overline{\mathcal{P}_{T^\perp}(\mathcal{Z})}\rangle && (3.25)\\
\geq&\|\mathcal{M}\|_{TNN} - \|\overline{\mathcal{U}}\overline{\mathcal{V}}^\top - \overline{\mathcal{P}_T(\mathcal{Y})}\|_F\|\overline{\mathcal{P}_T(\mathcal{Z})}\|_F + \|\mathcal{P}_{T^\perp}(\mathcal{Z})\|_{TNN} - \|\overline{\mathcal{P}_{T^\perp}(\mathcal{Y})}\|\|\overline{\mathcal{P}_{T^\perp}(\mathcal{Z})}\|_* \\
& && (3.26)\\
=&\|\mathcal{M}\|_{TNN} - n_3\|\mathcal{U} * \mathcal{V}^\top - \mathcal{P}_T(\mathcal{Y})\|_F\|\mathcal{P}_T(\mathcal{Z})\|_F + \|\mathcal{P}_{T^\perp}(\mathcal{Z})\|_{TNN} \\
&- \|\mathcal{P}_{T^\perp}(\mathcal{Y})\|\|\mathcal{P}_{T^\perp}(\mathcal{Z})\|_{TNN} \\
\geq&\|\mathcal{M}\|_{TNN} - \frac{1}{4nn_3}\|\mathcal{P}_T(\mathcal{Z})\|_F + \frac{1}{2}\|\mathcal{P}_{T^\perp}(\mathcal{Z})\|_{TNN} && (3.27)\\
>&\|\mathcal{M}\|_{TNN}
\end{aligned}
$$

where (3.23) uses the Fact 3.3.1; $\mathcal{Y}$ in (3.24) is a tensor dual certificate supported in Ω such that $\mathcal{P}_\Omega(\mathcal{Y}) = \mathcal{Y}$. So it is easy to show $\langle \mathcal{Z}, \mathcal{Y}\rangle = 0$ using the standard basis decomposition; (3.25) uses equation (3.22); (3.26) is based on the following two facts

for any same size matrices $\mathbf{A}$ and $\mathbf{B}$,

$$|\langle \mathbf{A}, \mathbf{B} \rangle| \leq \|\mathbf{A}\|_F \|\mathbf{B}\|_F$$

$$\langle \mathbf{A}, \mathbf{B} \rangle \leq \|\mathbf{A}\| \|\mathbf{B}\|_* ,$$

and (3.27) uses the Condition **2** of Proposition 3.3.1.

Therefore, for any $\mathcal{X} \neq \mathcal{M}$ obeying $\mathcal{P}_\Omega(\mathcal{X} - \mathcal{M}) = 0$, we have $\|\mathcal{X}\|_{TNN} > \|\mathcal{M}\|_{TNN}$, which proves $\mathcal{M}$ is the unique minimizer of (3.5).

Proof of Theorem 3.2.1 As proved in the Appendix A, B and C, if p satisfies (3.7), the Condition 1 and Condition 2 in Proposition 3.3.1 are satisfied with probability at least $1 - c_1((n_1 + n_2)n_3)^{-c_2}$ for some positive constants c_1 and c_2. The proof of Theorem 3.2.1 then follows directly from Proposition 3.3.1, which states that $\mathcal{M}$ is the unique minimizer to (3.5).

3.4 Algorithms For Tensor completion

We will show the case when the tensor data is simply decimated randomly or down sampled in this section. Specifically we consider the problem of data completion from missing entries for multilinear signals. Suppose there is an unknown tensor $\mathcal{M}$ of size $n_1 \times n_2 \times n_3$ which is ***assumed to have a low tubal-rank*** and we are given a subset of entries $\{\mathcal{M}_{ijk} : (i, j, k) \in \Omega\}$ where Ω is an indicator tensor of size $n_1 \times n_2 \times n_3$. Our objective is to recover the entire $\mathcal{M}$. This section develops an algorithm for

addressing this problem via solving the following complexity penalized algorithm:

$$\begin{aligned} \min \quad & \|\boldsymbol{\mathcal{X}}\|_{TNN} \\ \text{subject to} \quad & P_{\boldsymbol{\Omega}}(\boldsymbol{\mathcal{X}}) = P_{\boldsymbol{\Omega}}(\boldsymbol{\mathcal{M}}) \end{aligned} \tag{3.28}$$

where $P_{\boldsymbol{\Omega}}$ is the orthogonal projector onto the span of tensors vanishing outside of $\boldsymbol{\Omega}$. So the $(i,j,k)_{th}$ component of $P_{\boldsymbol{\Omega}}(\boldsymbol{\mathcal{X}})$ is equal to $\boldsymbol{\mathcal{M}}_{ijk}$ if $(i,j,k) \in \boldsymbol{\Omega}$ and zero otherwise. Let $\boldsymbol{\mathcal{Y}}$ be the available (sampled) data: $\boldsymbol{\mathcal{Y}} = P_{\boldsymbol{\Omega}}\boldsymbol{\mathcal{M}}$. Define $\mathcal{G} = \mathcal{F}_3 P_{\boldsymbol{\Omega}} \mathcal{F}_3^{-1}$ where $\mathcal{F}_3$ and $\mathcal{F}_3^{-1}$ are the operators representing the Fourier and inverse Fourier transform along the third dimension of tensors. Then we have $\hat{\boldsymbol{\mathcal{Y}}} = \mathcal{G}(\hat{\boldsymbol{\mathcal{M}}})$ where $\hat{\boldsymbol{\mathcal{Y}}}$ and $\hat{\boldsymbol{\mathcal{M}}}$ are the Fourier transforms of $\boldsymbol{\mathcal{Y}}$ and $\boldsymbol{\mathcal{M}}$ along the third mode.

So (3.28) is equivalent with the following:

$$\begin{aligned} \min \quad & ||\text{blkdiag}(\hat{\boldsymbol{\mathcal{X}}})||_* \\ \text{subject to} \quad & \hat{\boldsymbol{\mathcal{Y}}} = \mathcal{G}(\hat{\boldsymbol{\mathcal{X}}}) \end{aligned} \tag{3.29}$$

where $\hat{\boldsymbol{\mathcal{X}}}$ is the Fourier transform of $\boldsymbol{\mathcal{X}}$ along the third dimension and blkdiag($\hat{\boldsymbol{\mathcal{X}}}$) is defined in (**??**). Noting that $\|\boldsymbol{\mathcal{X}}\|_{TNN} = ||\text{blkdiag}(\hat{\boldsymbol{\mathcal{X}}})||_*$. To solve the optimization problem, one can re-write (3.29) equivalently as follows:

$$\begin{aligned} \min \quad & ||\text{blkdiag}(\hat{\boldsymbol{\mathcal{Z}}})||_* + \mathbf{1}_{\hat{\boldsymbol{\mathcal{Y}}}=\mathcal{G}(\hat{\boldsymbol{\mathcal{X}}})} \\ \text{subject to} \quad & \hat{\boldsymbol{\mathcal{X}}} - \hat{\boldsymbol{\mathcal{Z}}} = 0 \end{aligned} \tag{3.30}$$

where **1** denotes the indicator function. Then using the general framework of Alternating Direction Method of Multipliers(ADMM)[77] we have the following recursion,

$$\begin{aligned}\boldsymbol{\mathcal{X}}^{k+1} &= \arg\min_{\boldsymbol{\mathcal{X}}} \left\{ \mathbf{1}_{\boldsymbol{\mathcal{Y}}=P_\Omega(\boldsymbol{\mathcal{X}})} + \boldsymbol{\mathcal{X}}(:)^\top \boldsymbol{\mathcal{Q}}^k(:) + \frac{1}{2}||\boldsymbol{\mathcal{X}} - \boldsymbol{\mathcal{Z}}^k||_F^2 \right\} \\ &= \arg\min_{\boldsymbol{\mathcal{X}}:\boldsymbol{\mathcal{Y}}=P_\Omega(\boldsymbol{\mathcal{X}})} \left\{ ||\boldsymbol{\mathcal{X}} - (\boldsymbol{\mathcal{Z}}^k - \boldsymbol{\mathcal{Q}}^k)||_F^2 \right\} \qquad (3.31) \\ \hat{\boldsymbol{\mathcal{Z}}}^{k+1} &= \arg\min_{\hat{\boldsymbol{\mathcal{Z}}}} \left\{ \frac{1}{\rho}||\text{blkdiag}(\hat{\boldsymbol{\mathcal{Z}}})||_* + \frac{1}{2}||\hat{\boldsymbol{\mathcal{Z}}} - (\hat{\boldsymbol{\mathcal{X}}}^{k+1} + \hat{\boldsymbol{\mathcal{Q}}}^k)||_F^2 \right\} \qquad (3.32) \\ \boldsymbol{\mathcal{Q}}^{k+1} &= \boldsymbol{\mathcal{Q}}^k + \left(\boldsymbol{\mathcal{X}}^{k+1} - \boldsymbol{\mathcal{Z}}^{k+1}\right) \qquad (3.33)\end{aligned}$$

where Equation (3.31) is least-squares projection onto the constraint and the solution to Equation (3.32) is given by the singular value thresholding[74, 78]. The $\boldsymbol{\mathcal{X}}(:)$ and $\boldsymbol{\mathcal{Q}}^k(:)$ means vectorizing the tensors which is Matlab notation.

Equivalence of the algorithm to iterative singular-tubal shrinkage via convolution

We will now show that the proposed algorithm for tensor completion has a very nice interpretation as an iterative singular tubal shrinkage using a convolution operation between the singular tubes and a tube of threshold vectors.

According to the particular format that (3.32) has, we can break it up into n_3 independent minimization problems. Let $\hat{\boldsymbol{\mathcal{Z}}}^{k+1,(i)}$ denotes the i_{th} frontal slice of $\hat{\boldsymbol{\mathcal{Z}}}^{k+1}$. Similarly define $\hat{\boldsymbol{\mathcal{X}}}^{k+1,(i)}$ and $\hat{\boldsymbol{\mathcal{Q}}}^{k,(i)}$. Then (3.32) can be separated as:

$$\hat{\boldsymbol{\mathcal{Z}}}^{k+1,(i)} = \arg\min_{W} \left\{ \frac{1}{\rho}||W||_* + \frac{1}{2}||W - (\hat{\boldsymbol{\mathcal{X}}}^{k+1,(i)} + \hat{\boldsymbol{\mathcal{Q}}}^{k,(i)})||_F^2 \right\} \qquad (3.34)$$

for $i = 1, 2, ..., n_3$. This means each i_{th} frontal slice of $\hat{\mathfrak{Z}}^{k+1}$ can be calculated through (3.34).

In order to solve (3.34), the following lemma is needed [78].

Lemma 3.4.1. *Consider the singular value decomposition (SVD) of a matrix* $X \in \mathbb{C}^{n_1 \times n_2}$ *of rank* r.

$$X = U\Sigma V^*, \;\; \Sigma = diag(\{\sigma_i\}_{1 \leq i \leq r}), \tag{3.35}$$

where U *and* V *are respectively* $n_1 \times r$ *and* $n_2 \times r$ *unitary matrices with orthonormal columns, and the singular values* σ_i *are real and positive. Then for all* $\tau \geq 0$ *, define the soft-thresholding operator* D_τ *as follows [78] :*

$$D_\tau(X) := UD_\tau(\Sigma)V^*, \;\; D_\tau(\Sigma) = diag\{(\sigma_i - \tau)_+\}, \tag{3.36}$$

where t_+ *is the positive part of* t*, namely,* $t_+ = max(0, t)$*. Then, for each* $\tau \geq 0$ *and* $Y \in C^{n_1 \times n_2}$*, the singular value shrinkage operator (3.36) obeys*

$$D_\tau(Y) = \arg\min_{X \in \mathbb{C}} \left\{ \frac{1}{2}\|X - Y\|_F^2 + \tau\|X\|_* \right\} \tag{3.37}$$

The proof can be found in [78] for the case when the matrix is real valued. However, it can be easily extended to matrices with complex entries using the result on gradients of unitarily invariant norms in [79].

Now note that, for matrices $\hat{\mathfrak{X}}^{k+1,(i)} + \hat{\mathfrak{Q}}^{k,(i)} \in \mathbb{R}^{n_1 \times n_2 \times n_3}, i = 1, 2, ..., n_3$, if $USV^{\mathrm{T}} = (\hat{\mathfrak{X}}^{k+1,(i)} + \hat{\mathfrak{Q}}^{k,(i)})$ is the SVD of $(\hat{\mathfrak{X}}^{k+1,(i)} + \hat{\mathfrak{Q}}^{k,(i)})$, then the solution to (3.34) is $UD_\tau(S)V^{\mathrm{T}}$, where $D_\tau(S) = \mathrm{diag}(S_{i,i} - \tau)_+$ for some positive constant τ and " + "

means keeping the positive part. This is equivalent to multiplying $(1 - \frac{\tau}{S_{i,i}})_+$ to the i_{th} singular value of S. So each frontal slice of $\hat{\mathcal{Z}}^{k+1}$ can be calculated using this shrinkage on each frontal slice of $(\hat{\mathcal{X}}^{k+1} + \hat{\mathcal{Q}}^k)$. Let $\mathcal{U} * \mathcal{S} * \mathcal{V}^{\mathrm{T}} = (\mathcal{X}^{k+1} + \mathcal{Q}^k)$ be the t-SVD of $(\mathcal{X}^{k+1} + \mathcal{Q}^k)$ and $\hat{\mathcal{S}}$ be the Fourier transform of $\mathcal{S}$ along the third dimension. Then each element of the singular tubes of $\hat{\mathcal{Z}}^{k+1}$ is the result of multiplying every entry $\hat{\mathcal{S}}(i,i,j)$ with $(1 - \frac{\tau}{\hat{\mathcal{S}}(i,i,j)})_+$ for some $\tau > 0$. Since this process is carried out in the Fourier domain, in the original domain it is equivalent to convolving each tube $\mathcal{S}(i,i,:)$ of $\mathcal{S}$ with a real valued tubal vector $\vec{\tau}_i$ which is the inverse Fourier transform of the vector $[(1 - \frac{\tau_i(1)}{\hat{\mathcal{S}}(i,i,1)})_+, (1 - \frac{\tau_i(2)}{\hat{\mathcal{S}}(i,i,2)})_+, ..., (1 - \frac{\tau_i(n_3)}{\hat{\mathcal{S}}(i,i,n_3)})_+]$. This operation can be captured by $\mathcal{S} * \mathcal{T}$, where $\mathcal{T}$ is an f-diagonal tensor with i_{th} diagonal tube to be $\vec{\tau}_i$. Then $\mathcal{Z}^{k+1} = \mathcal{U} * (\mathcal{S} * \mathcal{T}) * \mathcal{V}^{\mathrm{T}}$. In summary, the shrinkage operation in the Fourier domain on the singular values of each of the frontal faces is equivalent to performing a *tubal shrinkage* via convolution in the original domain.

3.5 Experiments

3.5.1 Video Completion

For experiments we test 3 algorithms for video data completion from randomly missing entries: TNN minimization of Section 3.4, Low Rank Tensor Completion (LRTC) algorithm in [80], which uses the notion of tensor-n-rank [63], and the nuclear norm

minimization on the vectorized video data using the algorithm in [78]. As an application of the t-SVD to higher order tensor we also show performance on a 4-D color Basketball video data of size $144 \times 256 \times 3 \times 80$, where the first 2 dimensions are the width and heights, third dimension is the RBG mode, 4th dimension is the time(frames).

Figures 3.2 and 3.3 show the results of recovery using the 3 algorithms. Figure 3.4 shows the RSE (dB) plots for sampling rates ranging from 10% to 90% where the sampling rate is defined to be the percentage of pixels which are known. Results from the figures show that the TNN minimization algorithm gives excellent reconstruction over the LRTC and Nuclear norm minimization. Figure 3.5 shows the recovery of 4-D colorful video using the proposed algorithm.

3.5.2 Cellular Data Completion

A wide variety of data for assessing the service quality experienced by their smartphone users are collected from lots of cellular network providers. A big challenge for effective service quality management in operational setup is the presence of missing or unavailable data. Furthermore, the cellular data is inherently multidimensional, i.e. it is a function of several variables such as location, device type, and time. In this section, we show that we can use our algorithm to complete cellular data as well. One thing to notice here is that, we also consider the case of tensor completion under

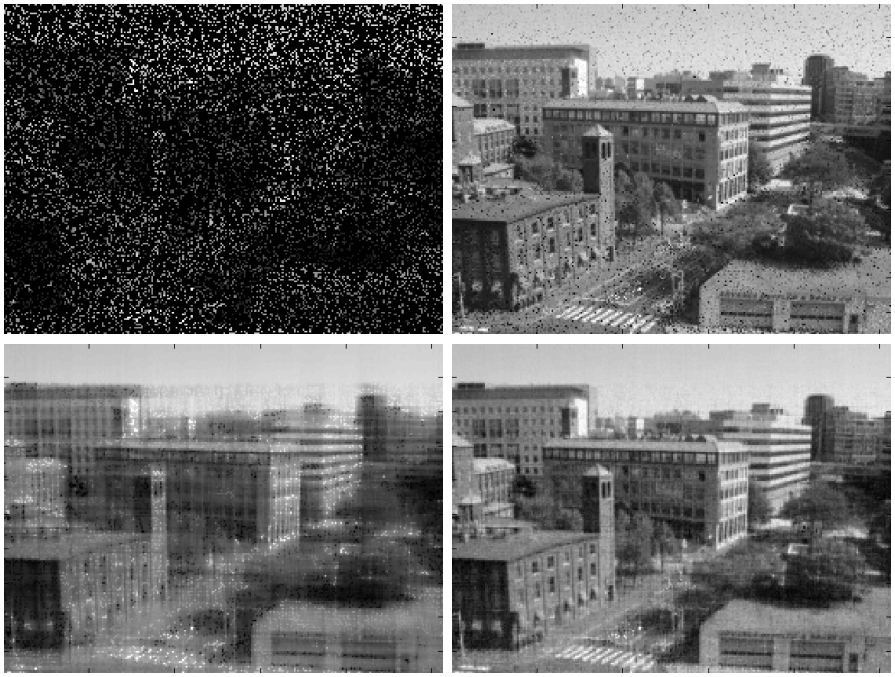


Figure 3.2: Tensor completion results for MERL video. **Upper left**: Sampled video(20%). **Upper right**: Nuclear norm minimization (vectorization and SVD based) result. **Lower left**: LRTC result. **Lower right**: TNN minimization result.

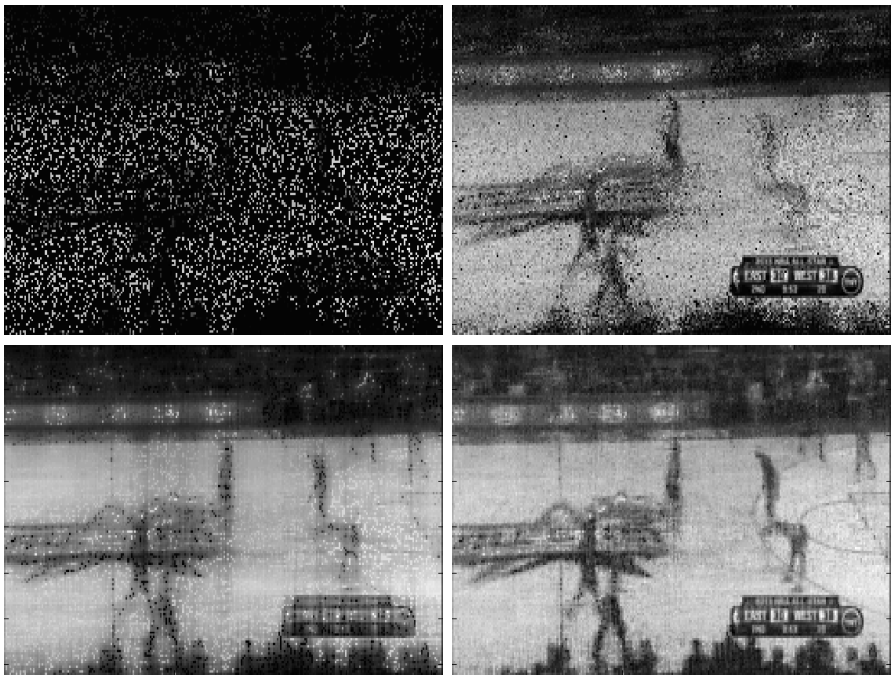


Figure 3.3: Tensor completion results for basketball video. **Upper left**: Sampled video(20%). **Upper right**: Nuclear norm minimization (vectorization and SVD based) result. **Lower left**: LRTC result. **Lower right**: TNN minimization result.

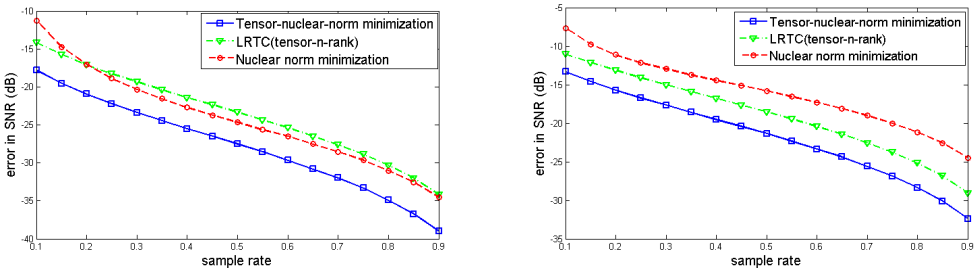


Figure 3.4: RSE (dB) plot against sampling rate **Left**: MERL video. **Right**: Basketball video

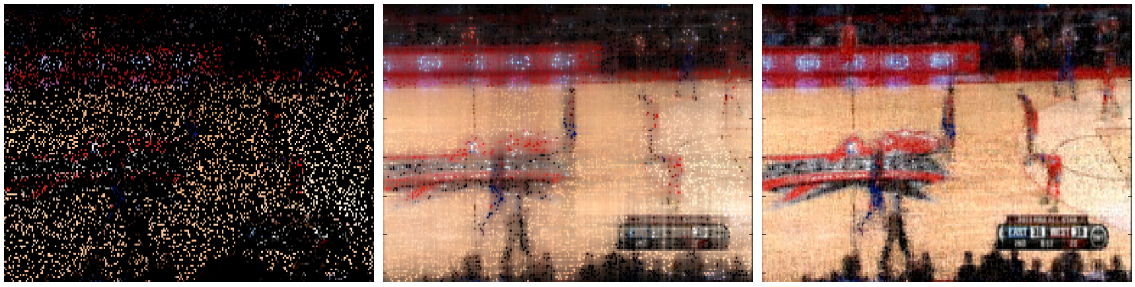


Figure 3.5: Recovery for color basketball video: **Left**: Sampled Video(10%). **Middle**: LRTC recovery. **Right**: Tensor-nuclear-norm minimization recovery

linear constraint :

$$\begin{aligned} &\min_{\mathcal{X}} \|\mathcal{X}\|_{TNN} \\ &\text{subject to } \mathcal{L}(\mathcal{X}) = \mathcal{Y} \ , \end{aligned} \tag{3.38}$$

where $\mathcal{L}$ represents the linear constraint. The algorithm is almost the same under the ADMM framework.

Firstly we consider a three-dimensional tensor consisting of service performance data. This data set is of size $314 \times 360 \times 73$ for 314 RNCs (Radio Network Controllers) and for 360 hours and contains a total of 73 KPIs (Key Performance Indicators). The data contains several KPIs including voice and data accessibility, retainability, RRC, SRB and RAB success rates, paging success rates, uplink and downlink traffic, voice Erlangs or minutes of usage. We sample the tensor elements randomly (independent and equally likely) with probability p to obtain the incomplete tensor, and complete this tensor using our approach. The results is showing in Figure 3.6, we can see that our method is better than LRTC(unfold completion) in this case.

Secondly we consider a four-dimensional tensor extracted from smartphone specific

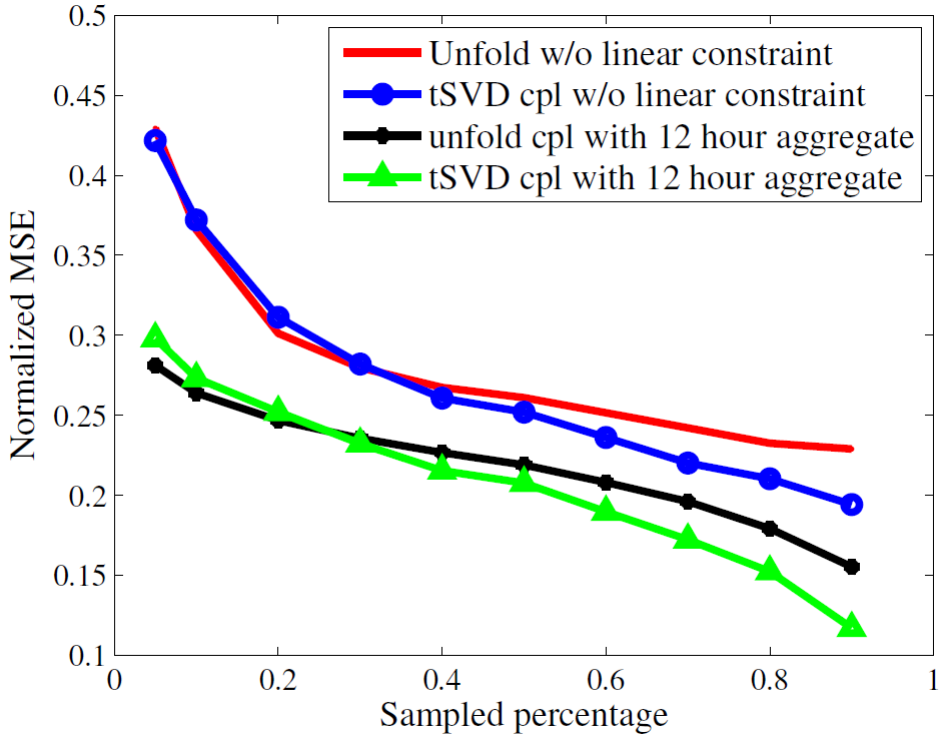


Figure 3.6: Completing the 3D tensor with varying sampled data with/without aggregate linear constraints.

measurements. The size of the tensor is $29 \times 253 \times 97 \times 5$. The first dimension is the number of days (29); 253 is the number of smartphone types; 97 is the number of network locations where the measurements are aggregated; and 5 is the number of KPIs from voice call detail records. This 4D tensor has inherently missing data because of the sparse population of users across certain types of smartphones.

Therefore, the missing data in this case is not random, and has a structure which is given by the available measurements. Our tensor-based completion approach can be used to predict the places where there is no data. We have only 68% entries available in this tensor, and use cross-validation to study the performance of our algorithm. We sample the available data with probability p (choosing each element among the available data randomly with probability p), and check the error on the remaining unsampled available data ($1p$ fraction of 68% data). Figure 3.7 gives the error on the unsampled data as p increases. Based on the accuracy for different unfolds, we find that for this data, unfolding onto the first dimension yields the best results while using the t-SVD approach we see that fixing the tensor orientation as a $29 \times 253 \times 97 \times 5$ tensor (obtained by simply permuting the indices) yields the best results. Figure 3.7 shows the results for the two cases.

3.5.3 Numerical Experiments

To demonstrate our results, we conducted some numerical experiments to recover third order tensors of different sizes and tubal-ranks r from m observed entries. For

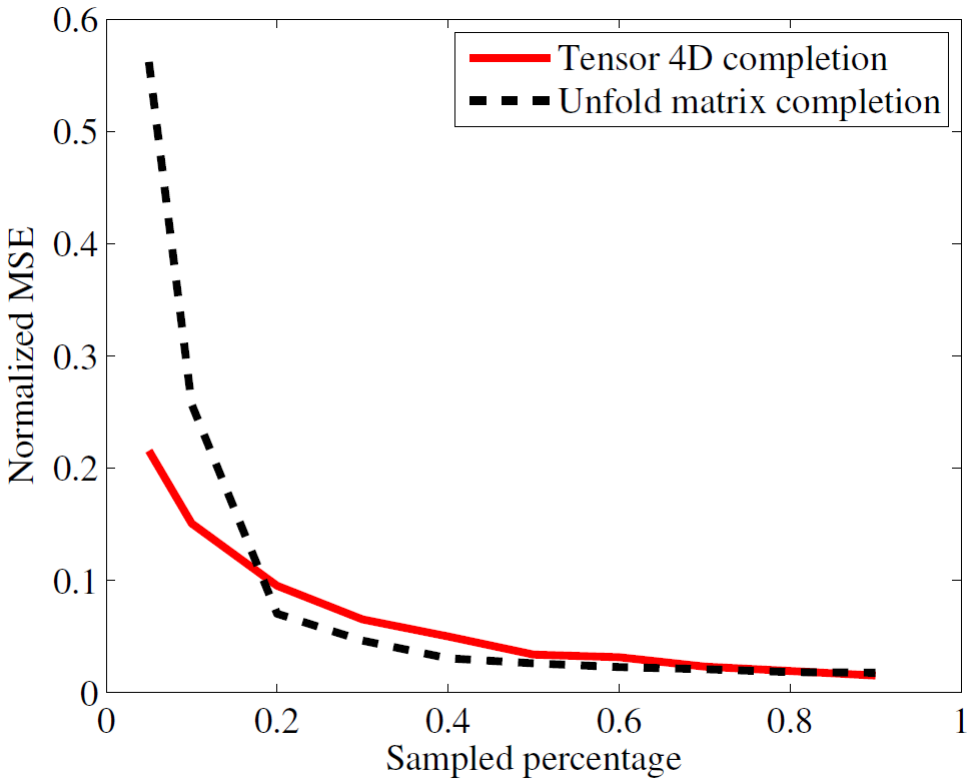
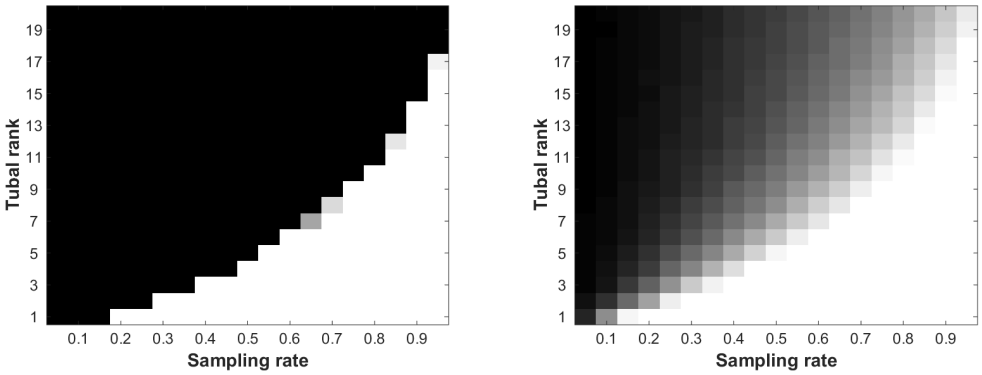
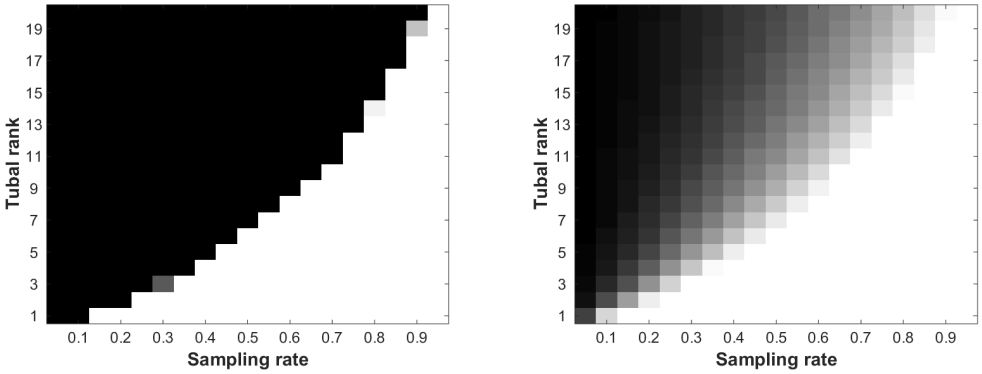


Figure 3.7: Completing the 4D tensor with varying sampled data on the 68% available data.

each episode we generated an $n_1 \times n_2 \times n_3$ random tensor with i.i.d. Gaussian entries, performed the t-SVD of it, kept the first r singular tubes and got $\boldsymbol{\mathcal{M}}$ with tubal rank r. We sampled m entries of $\boldsymbol{\mathcal{M}}$ uniformly at random and try to recover $\boldsymbol{\mathcal{M}}$ using (3.5). We denote the solution of (3.5) by $\boldsymbol{\mathcal{X}}$ and compute the relative square error (RSE): $\|\boldsymbol{\mathcal{X}} - \boldsymbol{\mathcal{M}}\|_F / \|\boldsymbol{\mathcal{M}}\|_F$. If the RSE $\leq 10^{-3}$, then we claim that the recovery is exact. We repeated our experiments 20 times. The results are shown in Figure 3.8. In the left figures, the color of each cell reflects the empirical recovery rate ranging from 0 to 1. White cell means exact recoveries in all experiments; and black cell means all experiments failed. The right figures are the RSE plots of one typical run of the simulation. For each cell, the value reflects the RSE of the recovery under the corresponding sampling rate and tubal rank. Black denotes 1 and white is 0.



(a) $40 \times 40 \times 30$ tensor.



(b) $30 \times 30 \times 20$ tensor.

Figure 3.8: **Recovery of third order tensors from their entries.** In the left figures of both cases, each cell's value reflects the empirical recovery rate. Black denotes failure and white denotes success in recovery in all simulations. In the right figures of both cases, each cell's value is the RSE of the recovery under the corresponding sampling rate and tubal rank. Black denotes 1 and white denotes 0.

Chapter 4

Tensor Robust PCA

In this chapter we consider a "tensor robust principal component analysis" problem of recovering a low tensor-multi rank tensor $\boldsymbol{\mathcal{L}}$ from a sparsely corrupted observation tensor. There are two topics introduced in this chapter, batch tensor Robust PCA and online tensor Robust PCA.

4.1 Batch Tensor Robust PCA

4.1.1 Problem Formulation and Algorithm

Similar to the matrix robust PCA case [81], suppose we have a third-order tensor $\boldsymbol{\mathcal{M}}$ such that it can be decomposed as

$$\boldsymbol{\mathcal{M}} = \boldsymbol{\mathcal{L}} + \boldsymbol{\mathcal{S}} \tag{4.1}$$

where $\boldsymbol{\mathcal{L}}$ has low tensor-multi-rank and $\boldsymbol{\mathcal{S}}$ is sparse tensor. Here we focus on a case where the sparse tensor S is tubewise sparse as shown in Figure 4.1. To resolve the low rank and the sparse components given the observation $\boldsymbol{\mathcal{M}}$ we consider the following optimization problem.

$$\begin{aligned} \min \ & \|\boldsymbol{\mathcal{L}}\|_{TNN} + \lambda \|\boldsymbol{\mathcal{S}}\|_{1,1,2} \\ \text{subject to } & \boldsymbol{\mathcal{M}} = \boldsymbol{\mathcal{L}} + \boldsymbol{\mathcal{S}} \end{aligned} \tag{4.2}$$

where $\lambda > 0$ and the $\|\boldsymbol{\mathcal{S}}\|_{1,1,2}$ for 3-D tensors is defined as $\sum_{i,j} ||\boldsymbol{\mathcal{S}}(i,j,:)||_F$.

An application where this is useful arises in multilinear imaging scenarios where some pixels have heavy noise on them and the task is to automatically locate such pixels and recover the video. Although this may be done by processing each frame but if the noise artifacts and video features are aligned, one needs to both detect the noise and estimate the corrupted video feature.

In order to solve the convex optimization problem of Equation (4.2) we use ADMM. Then we have the following recursion,

$$\boldsymbol{\mathcal{L}}^{k+1} = \arg\min_{\boldsymbol{\mathcal{L}}} \|\boldsymbol{\mathcal{L}}\|_{TNN} + \frac{\rho}{2}\|\boldsymbol{\mathcal{L}} + \boldsymbol{\mathcal{S}}^k - \boldsymbol{\mathcal{M}} + \boldsymbol{\mathcal{W}}^k\|_F^2 \tag{4.3}$$

$$\boldsymbol{\mathcal{S}}^{k+1} = \arg\min_{\boldsymbol{\mathcal{S}}} \lambda\|\boldsymbol{\mathcal{S}}\|_{1,1,2} + \frac{\rho}{2}\|\boldsymbol{\mathcal{L}}^{k+1} + \boldsymbol{\mathcal{S}} - \boldsymbol{\mathcal{M}} + \boldsymbol{\mathcal{W}}^k\|_F^2 \tag{4.4}$$

$$\boldsymbol{\mathcal{W}}^{k+1} = \boldsymbol{\mathcal{W}}^k + \boldsymbol{\mathcal{L}}^{k+1} + \boldsymbol{\mathcal{S}}^{k+1} - \boldsymbol{\mathcal{M}} \tag{4.5}$$

where $\boldsymbol{\mathcal{W}} = \rho\boldsymbol{\mathcal{Y}}$, for some $\rho > 0$. From section 3.4 we already have the solution to (4.3) if we transform this equation into the Fourier domain then the tensor-nuclear-norm of $\boldsymbol{\mathcal{L}}$ will be the nuclear norm of blkdiag($\hat{\boldsymbol{\mathcal{L}}}$). The closed form solution to Equation (4.4)

is given by,

$$\mathcal{S}^{k+1}(i,j,:) = \left(1 - \frac{\lambda}{\rho\|\mathcal{S}^k(i,j,:)\|_F}\right)_+ \mathcal{S}^k(i,j,:) \tag{4.6}$$

where $i = 1, 2, ..., n_3$.

4.1.2 Experimental Results

For experiment we consider a video, which is compressible in the t-SVD. We randomly corrupt video data by corrupting some pixels with heavy additive noise. We want to estimate the locations of such pixels using tensor robust PCA. The video used in this application is the basketball video with randomly chosen sparse pixels tubes along the third dimension. This set-up is especially useful when dealing with old movies where some certain pixel locations are dead for the whole movie. For each selected pixel we add random Gaussian noise on it. Figure (4.1) shows the original video(tensor) and the noise tensor. The size of each frame is 72×128 and the total number of frames is 80. The noisy pixel tubes within every 10 frames are consistent. We use the above ADMM algorithm to separate the original video and the noise. We also perform matrix robust PCA on this noisy video data by vectorizing each frame, saving it as a column vector and then get a $n_1 n_2 \times n_3$ matrix. In this case the choice of λ is $\frac{1}{\sqrt{\max(n_1 n_2, n_3)}}$ [81].

The result of both tensor robust PCA and matrix robust PCA is shown in Figure 4.2. From the results we can see that tensor robust PCA works very well on separating the noisy pixels from the video. However, the matrix robust PCA results

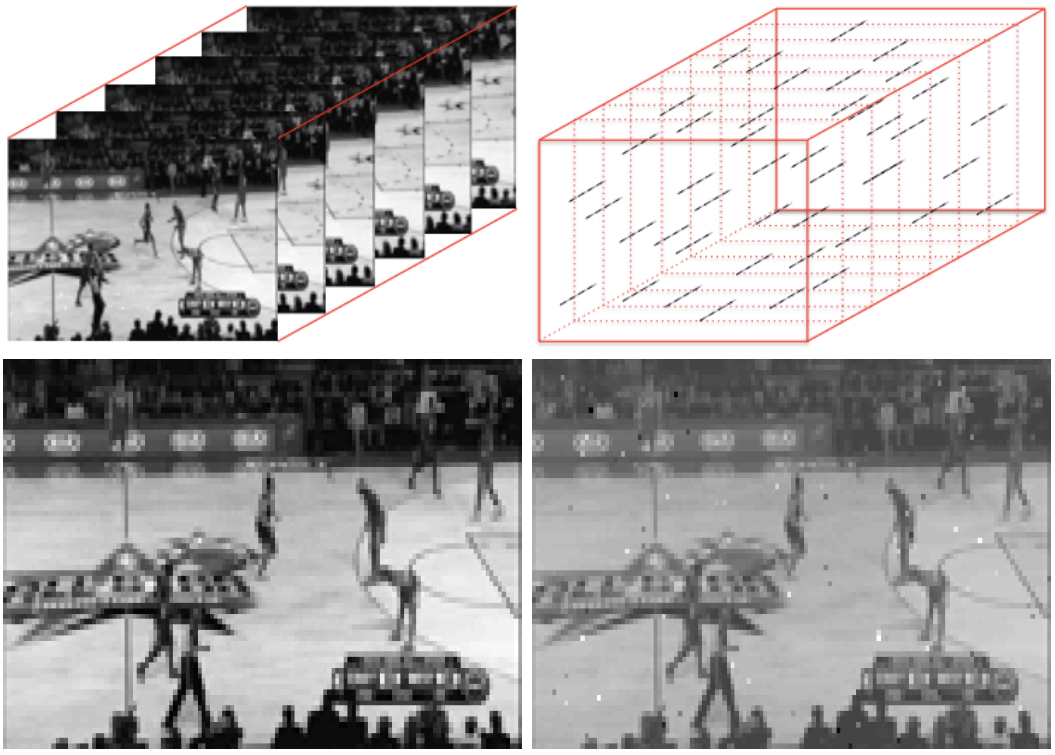


Figure 4.1: **Upper left**: Original video. **Upper right**: Noisy tensor. For 10 consecutive frames the locations of noisy pixels are the same and then selected randomly for the next 10 frames. **Lower left** 21st frame of the original video. **Lower right** 21st frame of the noisy video.

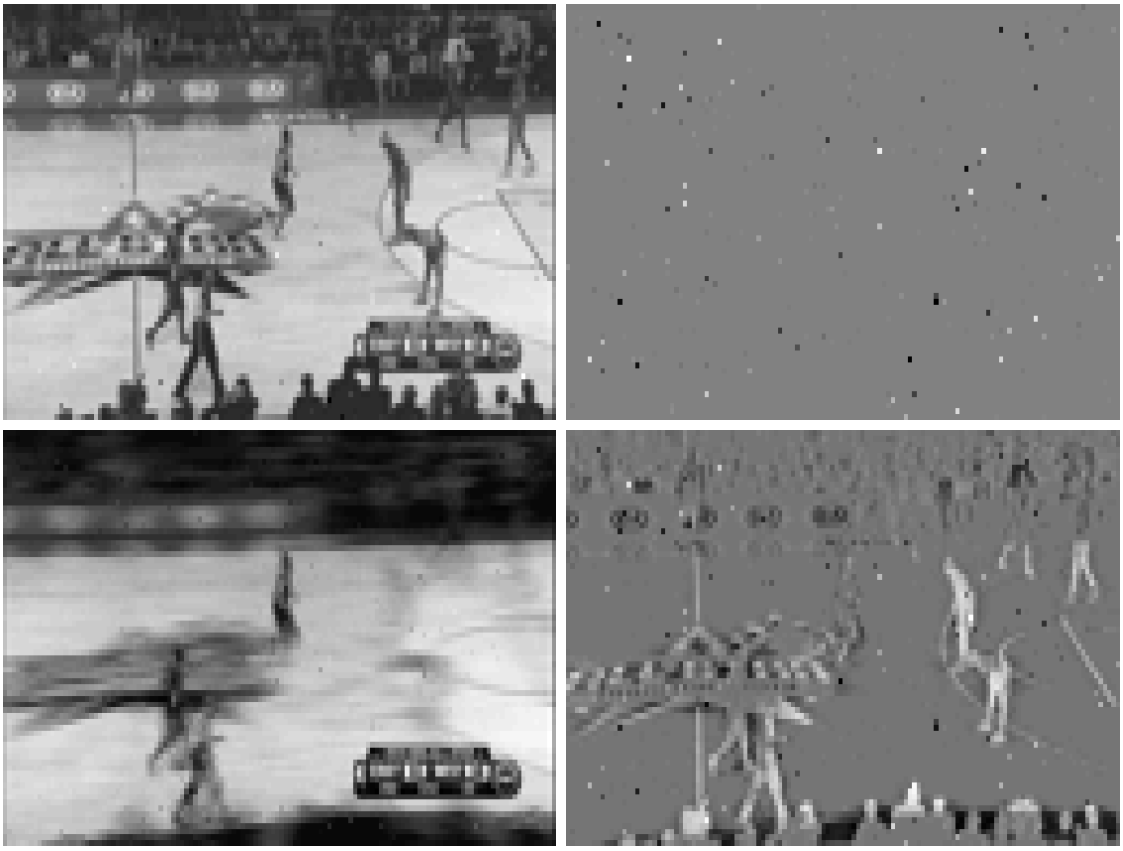


Figure 4.2: (21st frame shown) **Upper Left**: Low tensor multi-rank part recovered from tensor robust PCA. **Upper Right**: Sparse reconstruction from tensor robust PCA. **Lower left**: Low matrix rank part recovered from matrix robust PCA. **Lower right**: Sparse reconstruction from matrix robust PCA.

in an almost fixed blurred background as the low rank part while some structure of the playground, moving people and the noise are recovered as the sparse part.

4.2 Online Tensor Robust PCA

4.2.1 Problem Formulation and Algorithm

Before going to the online version, we start with the a slightly different batch setting. One difference between this section and the previous one is the sparse component. Suppose we have a third-order tensor $\mathcal{Z}$ which can be decomposed as,

$$\mathcal{Z} = \mathcal{X} + \mathcal{E}, \tag{4.7}$$

where $\mathcal{X}$ is a tensor with low tensor tubal rank and $\mathcal{E}$ is a sparse tensor. The problem of recovering $\mathcal{X}$ and $\mathcal{E}$ separately, termed tensor RPCA [22], can be formulated as an optimization problem [1]

$$\min_{\mathcal{X},\mathcal{E}} \frac{1}{2}\|\mathcal{Z} - \mathcal{X} - \mathcal{E}\|_F^2 + \lambda_1\|\mathcal{X}\|_{\text{TNN}} + \lambda_2\|\mathcal{E}\|_1, \tag{4.8}$$

where $\|\mathcal{X}\|_{\text{TNN}} = \sum_{i,j} \widehat{\mathcal{S}}(i,i,j)$ denotes the tensor nuclear norm and it's a convex relaxation of the tensor tubal rank; $\|\mathcal{E}\|_1 = \sum_{i,j,k} |\mathcal{E}(i,j,k)|$; and $\lambda_1, \lambda_2 > 0$.

Now we describe an implementation of tensor robust PCA that operates online. Suppose the 2-D data samples $\mathcal{Z}(:,i,:), i = 1,2,...,T$ are observed sequentially. Our goal is to estimate the spanning basis (principal components) of $\mathcal{X}$ on the fly, and separate the sparse tensor simultaneously. A figure that illustrates this is given in 4.3

In order to proceed we introduce the following lemma with its proof.

[1] In the previous section, we use $\|.\|_{1,1,2}$ norm as the complexity of the "tubal-sparsity" of tensors in the third-dimension. Here we consider a general case of sparsity of $\mathcal{S}$, therefore $\|.\|_1$ norm is utilized.

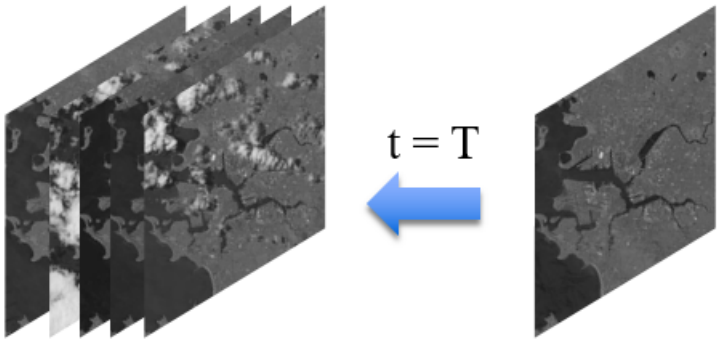


Figure 4.3: Online manner: data samples observed sequentially. Note that here each data sample is an $n_1 \times 1 \times n_3$ lateral slice (tensor column).

Lemma 4.2.1. *For a third-order tensor $\mathcal{X} \in \mathbb{R}^{n_1 \times n_2 \times n_3}$, suppose its tensor tubal rank is upper bounded by r, then we have*

$$\|\mathcal{X}\|_{TNN} = \inf_{\substack{\mathcal{L} \in \mathbb{R}^{n_1 \times r \times n_3} \\ \mathcal{R} \in \mathbb{R}^{n_2 \times r \times n_3}}} \left\{ \frac{n_3}{2} (\|\mathcal{L}\|_F^2 + \|\mathcal{R}\|_F^2) : \mathcal{X} = \mathcal{L} * \mathcal{R}^\top \right\}. \tag{4.9}$$

Proof. Let $\widehat{\mathcal{X}} = \text{fft}(\mathcal{X}, [\,], 3)$, then we have

$$\begin{aligned} \|\mathcal{X}\|_{\text{TNN}} &= \sum_{i=1}^{n_3} \|\widehat{\mathcal{X}}^{(i)}\|_* \\ &= \sum_{i=1}^{n_3} \inf_{\substack{\widehat{\mathcal{L}}^{(i)} \in \mathbb{R}^{n_1 \times r} \\ \widehat{\mathcal{R}}^{(i)} \in \mathbb{R}^{n_2 \times r}}} \left\{ \frac{n_3}{2} \|\widehat{\mathcal{L}}^{(i)}\|_F^2 + \frac{n_3}{2} \|\widehat{\mathcal{R}}^{(i)}\|_F^2 : \right. \\ &\qquad \left. \widehat{\mathcal{X}}^{(i)} = \widehat{\mathcal{L}}^{(i)} \widehat{\mathcal{R}}^{(i)\text{T}} \right\} \end{aligned} \tag{4.10}$$

since tensor tubal rank is always greater or equal to the matrix rank of each frontal slice in the Fourier domain. Note that for each $i = 1, 2, ..., n_3$, $\widehat{\mathcal{L}}^{(i)}$ is independent

from each other, so as $\widehat{\mathcal{R}}^{(i)}$. We can write

$$\begin{aligned}
&\|\mathcal{X}\|_{\text{TNN}} \\
&= \inf_{\substack{\widehat{\mathcal{L}}^{(i)}\in\mathbb{R}^{n_1\times r} \\ \widehat{\mathcal{R}}^{(i)}\in\mathbb{R}^{n_2\times r}}} \Big\{ \frac{1}{2}\sum_{i=1}^{n_3}\|\widehat{\mathcal{L}}^{(i)}\|_F^2 + \frac{1}{2}\sum_{i=1}^{n_3}\|\widehat{\mathcal{R}}^{(i)}\|_F^2 : \\
&\qquad\qquad \widehat{\mathcal{X}}^{(i)} = \widehat{\mathcal{L}}^{(i)}\widehat{\mathcal{R}}^{(i)\text{T}} \Big\} \\
&= \inf_{\substack{\widehat{\mathcal{L}}^{(i)}\in\mathbb{R}^{n_1\times r} \\ \widehat{\mathcal{R}}^{(i)}\in\mathbb{R}^{n_2\times r}}} \Big\{ \frac{1}{2}\|\widehat{\mathcal{L}}\|_F^2 + \frac{1}{2}\|\widehat{\mathcal{R}}\|_F^2 : \widehat{\mathcal{X}}^{(i)} = \widehat{\mathcal{L}}^{(i)}\widehat{\mathcal{R}}^{(i)\text{T}} \Big\} \\
&= \inf_{\substack{\mathcal{L}\in\mathbb{R}^{n_1\times r\times n_3} \\ \mathcal{R}\in\mathbb{R}^{n_2\times r\times n_3}}} \Big\{ \frac{n_3}{2}\|\mathcal{L}\|_F^2 + \frac{n_3}{2}\|\mathcal{R}\|_F^2 : \mathcal{X} = \mathcal{L} * \mathcal{R}^\top \Big\}
\end{aligned}$$

where the last step is because $\|\mathcal{X}\|_F = 1/\sqrt{n_3}\|\widehat{\mathcal{X}}\|_F$ [82]. □

Using the above lemma, we re-write (4.8) as

$$\begin{aligned}
\min_{\mathcal{L},\mathcal{R},\mathcal{E}} \quad & \frac{1}{2}\|\mathcal{Z} - \mathcal{L} * \mathcal{R}^\top - \mathcal{E}\|_F^2 + \frac{n_3\lambda_1}{2}(\|\mathcal{L}\|_F^2 + \|\mathcal{R}\|_F^2) \\
& + \lambda_2\|\mathcal{E}\|_1 \ \text{ s.t. } \ \mathcal{X} = \mathcal{L} * \mathcal{R}^\top,
\end{aligned} \tag{4.11}$$

where $\mathcal{L} \in \mathbb{R}^{n_1\times r\times n_3}$, $\mathcal{R} \in \mathbb{R}^{n_2\times r\times n_3}$. For sequentially observed data $\{\overrightarrow{\mathcal{Z}}_1, \overrightarrow{\mathcal{Z}}_2, ..., \overrightarrow{\mathcal{Z}}_T\} \in \mathbb{R}^{n_1\times 1\times n_3}$, we define the loss function for each sample based on (4.11) as

$$\ell(\overrightarrow{\mathcal{Z}}_i, \mathcal{L}) = \min_{\overrightarrow{\mathcal{R}},\overrightarrow{\mathcal{E}}} \frac{1}{2}\|\overrightarrow{\mathcal{Z}}_i - \mathcal{L} * \overrightarrow{\mathcal{R}}^\top - \overrightarrow{\mathcal{E}}\|_F^2 + \frac{n_3\lambda}{2}\|\overrightarrow{\mathcal{R}}\|_F^2 + \lambda_2\|\overrightarrow{\mathcal{E}}\|_1. \tag{4.12}$$

To solve the online tensor RPCA problem, we summarize our online tensor RPCA approach in Algorithm 2. For simplicity, we use $\widehat{\mathcal{A}}$ to denote $\text{fft}(\mathcal{A}, [\,], 3)$, and $\overline{\mathcal{A}} \in \mathbb{R}^{n_1n_3\times n_2n_3}$ to denote the block diagonal matrix of $\widehat{\mathcal{A}}$. In Matlab notation, it's defined by

$$\overline{\mathcal{A}} = \text{blkdiag}(\widehat{\mathcal{A}}^{(1)}, \widehat{\mathcal{A}}^{(2)}, \cdots, \widehat{\mathcal{A}}^{(n_3)}). \tag{4.13}$$

The key idea of our online tensor RPCA algorithm is that at each time round t, we minimizes the loss function over $\overrightarrow{\mathcal{Z}}_t$ given the previous estimation $\mathcal{L}_{t-1}$, to get the optimal $\overrightarrow{\mathcal{R}}_t$ and $\overrightarrow{\mathcal{E}}_t$. Then we alternately use the latest estimated components to update the spanning basis $\mathcal{L}_t$ via minimizing the cumulative loss.

Specifically, $\overrightarrow{\mathcal{R}}_t$ and $\overrightarrow{\mathcal{E}}_t$ are optimized in Step 3 with detailed description in Algorithm 3. In the data projection step in Algorithm 3, $S_\lambda[\cdot]$ is a soft-thresholding operator defined by [83],

$$S_\lambda[x] = \begin{cases} x-\lambda, & \text{if } x > \lambda \\ x+\lambda, & \text{if } x < -\lambda \\ 0, & \text{otherwise.} \end{cases} \tag{4.14}$$

To update $\mathcal{L}_t$, we have

$$\begin{aligned}
\mathcal{L}_t &= \underset{\mathcal{L}}{\operatorname{argmin}} \sum_{i=1}^{t} \Big(\frac{1}{2} \|\overrightarrow{\mathcal{Z}}_i - \mathcal{L} * \overrightarrow{\mathcal{R}}_i^\top - \overrightarrow{\mathcal{E}}_i\|_F^2 \\
&\qquad + \frac{n_3\lambda_1}{2}\|\overrightarrow{\mathcal{R}}_i\|_F^2 + \lambda_2 \|\overrightarrow{\mathcal{E}}_i\|_1 \Big) + \frac{n_3\lambda_1}{2}\|\mathcal{L}\|_F^2 \\
&= \underset{\mathcal{L}}{\operatorname{argmin}} \frac{1}{2}\|\mathcal{Z}_t - \mathcal{E}_t - \mathcal{L} * \mathcal{R}_t^\top\|_F^2 + \frac{n_3\lambda_1}{2}\|\mathcal{L}\|_F^2 \\
&= \underset{\mathcal{L}}{\operatorname{argmin}} \frac{1}{2}\|\overline{\mathcal{Z}}_t - \overline{\mathcal{E}}_t - \overline{\mathcal{L}}\overline{\mathcal{R}}_t^\top\|_F^2 + \frac{n_3\lambda_1}{2}\|\overline{\mathcal{L}}\|_F^2 \\
&= \underset{\mathcal{L}}{\operatorname{argmin}} \frac{1}{2}\operatorname{tr}\big((\overline{\mathcal{Z}}_t - \overline{\mathcal{E}}_t - \overline{\mathcal{L}}\overline{\mathcal{R}}_t^\top)^\top(\overline{\mathcal{Z}}_t - \overline{\mathcal{E}}_t - \overline{\mathcal{L}}\overline{\mathcal{R}}_t^\top)\big) \\
&\qquad + \frac{n_3\lambda_1}{2}\operatorname{tr}(\overline{\mathcal{L}}^\top\overline{\mathcal{L}}) \\
&= \underset{\mathcal{L}}{\operatorname{argmin}} \frac{1}{2}\operatorname{tr}\big(\overline{\mathcal{L}}(\overline{\mathcal{R}}_t^\top\overline{\mathcal{R}}_t + n_3\lambda_1 I)\overline{\mathcal{L}}^\top\big) - \operatorname{tr}(\overline{\mathcal{L}}^\top(\overline{\mathcal{Z}}_t - \overline{\mathcal{E}}_t)\overline{\mathcal{R}}_t).
\end{aligned}$$

Algorithm 2 Online Tensor Robust PCA

Input :Sequentially observed data $\mathcal{Z}_T = \{\overrightarrow{\mathcal{Z}}_1, ..., \overrightarrow{\mathcal{Z}}_T\} \in \mathbb{R}^{n_1 \times 1 \times n_3}$. $\lambda_1, \lambda_2 > 0$, number of rounds T.
Initial: $\mathcal{L}_0 \in \mathbb{R}^{n_1 \times r \times n_3}, \overrightarrow{\mathcal{R}}_0 \in \mathbb{R}^{r \times 1 \times n_3}, \overrightarrow{\mathcal{E}}_0 \in \mathbb{R}^{n_1 \times 1 \times n_3}$.

1: **for** $t = 1, 2, ..., T$ **do**
2: Reveal data sample $\overrightarrow{\mathcal{Z}}_t$.
3: Project the new sample(See Algorithm 3):

$$\{\overrightarrow{\mathcal{R}}_t, \overrightarrow{\mathcal{E}}_t\} = \underset{\substack{\overrightarrow{\mathcal{R}} \in \mathbb{R}^{r \times 1 \times n_3} \\ \overrightarrow{\mathcal{E}} \in \mathbb{R}^{n_1 \times 1 \times n_3}}}{\operatorname{argmin}} \frac{1}{2} \| \overrightarrow{\mathcal{Z}}_t - \mathcal{L}_{t-1} * \overrightarrow{\mathcal{R}} - \overrightarrow{\mathcal{E}} \|_F^2 + \frac{\lambda_1}{2} \| \overrightarrow{\mathcal{R}} \|_F^2 + \lambda_2 \| \overrightarrow{\mathcal{E}} \|_1$$

4: $\mathcal{A}_t = \mathcal{A}_{t-1} + \overrightarrow{\mathcal{R}}_t * \overrightarrow{\mathcal{R}}_t^\top$, $\mathcal{B}_t = \mathcal{B}_{t-1} + (\overrightarrow{\mathcal{Z}}_t - \overrightarrow{\mathcal{E}}_t) * \overrightarrow{\mathcal{R}}_t^\top$. Then take the Fourier transform of $\mathcal{A}_t, \mathcal{B}_t$, and let the block diagonal form be $\overline{\mathcal{A}}_t, \overline{\mathcal{B}}_t$, as in (4.13).
5: Compute $\mathcal{L}_t$ using $\mathcal{L}_{t-1}$ as the starting point(See Algorithm 4):

$$\overline{\mathcal{L}}_t \triangleq \underset{\overline{\mathcal{L}}}{\operatorname{argmin}} \frac{1}{2} \operatorname{tr}[\overline{\mathcal{L}}^\top (\overline{\mathcal{A}}_t + n_3 \lambda_1 I) \overline{\mathcal{L}}] - \operatorname{tr}(\overline{\mathcal{L}}^\top \overline{\mathcal{B}}_t).$$

6: Organize the block diagonal $\overline{\mathcal{L}}_t$ into the tensor form $\widehat{\mathcal{L}}_t$ and let $\mathcal{L}_t = \texttt{ifft}(\widehat{\mathcal{L}}_t, [\,], 3)$.
7: **end for**
8: $\mathcal{R}_T(t, :, :) = \overrightarrow{\mathcal{R}}_t^\top, \mathcal{E}_T(:, t, :) = \overrightarrow{\mathcal{E}}_t, t = 1, 2, ..., T$.
9: **Return** $\mathcal{X}_T = \mathcal{L}_T * \mathcal{R}_T^\top$ and $\mathcal{E}_T$.

Output: Low tubal rank tensor $\mathcal{X}_T$ and sparse tensor $\mathcal{E}_T$.

Let $\mathcal{A}_t = \mathcal{A}_{t-1} + \overrightarrow{\mathcal{R}}_t * \overrightarrow{\mathcal{R}}_t^\top$ and $\mathcal{B}_t = \mathcal{B}_{t-1} + (\overrightarrow{\mathcal{Z}}_t - \overrightarrow{\mathcal{E}}_t) * \overrightarrow{\mathcal{R}}_t^\top$, where $\overrightarrow{\mathcal{R}}_t \in \mathbb{R}^{r \times 1 \times n_3}, \overrightarrow{\mathcal{E}}_t \in \mathbb{R}^{n_1 \times 1 \times n_3}$, as indicated in Step 4 of Algorithm 2. We update $\mathcal{A}_t, \mathcal{B}_t$ each time new data comes and save the updated ones such that we can update the spanning basis $\mathcal{L}$ in the Fourier domain with block-coordinate descent [84], as indicated in Step 5 of Algorithm 2 with details in Algorithm 4. Note that our algorithm needs a prior information about estimated upper bound of the rank of the overall data samples.

Algorithm 3 Projecting data samples

Input: $\mathcal{L}_{t-1} \in \mathbb{R}^{n_1 \times r \times n_3}$, $\overrightarrow{\mathcal{Z}}_t \in \mathbb{R}^{n_1 \times 1 \times n_3}$, $\lambda_1, \lambda_2 > 0$.
Initial: $\overrightarrow{\mathcal{E}}_t = 0$, $\hat{\boldsymbol{r}}_t \in \mathbb{C}^{r \times 1 \times n_3}$.

$\hat{\boldsymbol{z}}_t = \texttt{fft}(\overrightarrow{\mathcal{Z}}_t, [\,], 3)$, $\widehat{\mathcal{L}}_{t-1} = \texttt{fft}(\mathcal{L}_{t-1}, [\,], 3)$.
while not converge **do**
  for $i = 1, 2, ..., n_3$ **do**
    $\hat{\boldsymbol{r}}_t^{(i)} = ((\widehat{\mathcal{L}}_{t-1}^{(i)})^\top \widehat{\mathcal{L}}_{t-1}^{(i)} + \lambda_1 I)^{-1} (\widehat{\mathcal{L}}_{t-1}^{(i)})^\top (\hat{\boldsymbol{z}}_t^{(i)} - \hat{\boldsymbol{e}}_t^{(i)})$.
  end for
  $\overrightarrow{\mathcal{R}}_t = \texttt{ifft}(\hat{\boldsymbol{r}}_t, [\,], 3)$.
  $\overrightarrow{\mathcal{E}}_t = S_{\lambda_2}[\overrightarrow{\mathcal{Z}}_t - \mathcal{L}_{t-1}^\top * \overrightarrow{\mathcal{R}}_t]$.
end while
Output: $\overrightarrow{\mathcal{R}}_t$ and $\overrightarrow{\mathcal{E}}_t$.

Algorithm 4 Update spanning tensor basis

Input: $\mathcal{L}_{t-1} \in \mathbb{R}^{n_1 \times n_2 \times n_3}$,$\mathcal{A}_t \in \mathbb{R}^{r \times r \times n_3}$,$\mathcal{B}_t \in \mathbb{R}^{n_1 \times r \times n_3}$

$\mathcal{L}_t = \mathcal{L}_{t-1}$, $\widehat{\mathcal{L}}_t = \texttt{fft}(\mathcal{L}_t, [\,], 3)$.
$\widehat{\mathcal{B}}_t = \texttt{fft}(\mathcal{B}_t, [\,], 3)$.
$\mathcal{C}_t = \mathcal{A}_t + \lambda_1 \mathcal{I}$, $\widehat{\mathcal{C}}_t = \texttt{fft}(\mathcal{C}_t, [\,], 3)$.
for $j = 1, 2, ..., r$ **do**
  for $k = 1, 2, ..., n_3$ **do**
    $\widehat{\mathcal{L}}_t(:, j, k) = \frac{\widehat{\mathcal{B}}_t(:,j,k) - \widehat{\mathcal{L}}_t(:,:,k) * \widehat{\mathcal{C}}_t(:,j,k)}{\widehat{\mathcal{C}}_t(j,j,k)} + \widehat{\mathcal{L}}_t(:, j, k)$.
  end for
end for
Output: $\mathcal{L}_t = \texttt{ifft}(\widehat{\mathcal{L}}_t, [\,], 3)$.

As regarding to the storage needed, for the batch tensor robust PCA all the data samples up to time T, *i.e.*, the total number of entries in $\{\mathcal{Z}_i\}_{i=1}^T$, are required. Therefore the storage requirement for the batch tensor robust PCA is $n_1 n_3 T$. While for online tensor robust PCA, we need to save $\mathcal{L}_{t-1} \in \mathbb{R}^{n_1 \times r \times n_3}$, $\mathcal{R}_T \in \mathbb{R}^{T \times r \times n_3}$ ($\mathcal{A}_T$ can be computed through $\mathcal{R}_T$), $\mathcal{B}_T \in \mathbb{R}^{n_1 \times r \times n_3}$, and the total storage requirement is $n_3 r T + n_1 n_3 r$, which is much smaller than that of the batch tensor robust PCA when

$r << T$.

4.2.2 Experimental Results

We consider the cloud removal problem on satellite images. A total of 24 cloud contaminated images captured by Landsat 7 ETM+ and Landsat 8 OLI near Harz, Germany over a period of time are used in our numerical experiments. Each image is of size 598×1070. The image backgrounds change slightly and the variability is mainly caused by the clouds and their shadows. Two example images are shown in Fig. 4.4. Considering the fact that the images are captured sequentially with restricted onboard storage, we use tensor online RPCA to perform cloud removal process such that clear images can be generated. Each time we receive an image from the satellite of this area, we directly estimate the spanning basis without saving all the past images but only the latest estimation and the newly collected data.

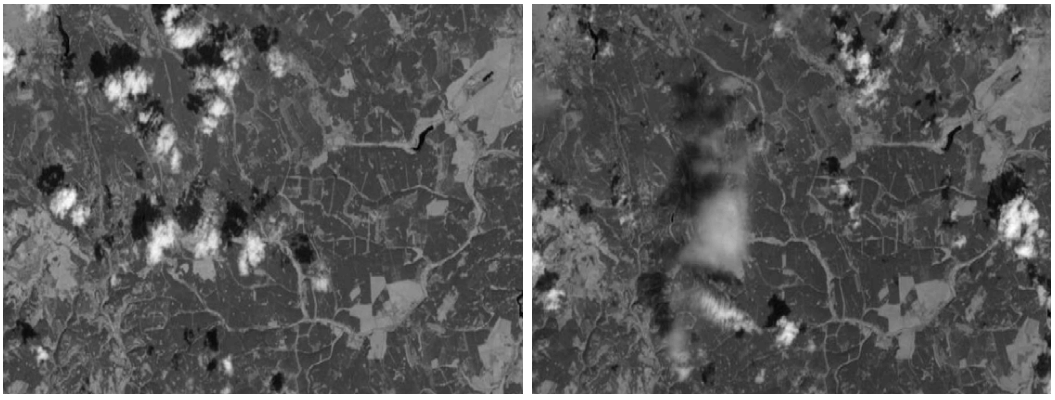


Figure 4.4: Example images of the same location taken by Landsat 7 ETM+ and Landsat 8 OLI on different days. A total of 24 images are used in our experiments.

To perform the proposed algorithm, images are normalized to have intensity of

$[0, 1]$ and reshaped as a lateral slice $\vec{z}_i$ of size $598 \times 1 \times 1070$. In the experiments we set $r = 3$ as a upper bound of the tensor tubal rank since the background is barely changing without clouds and shadows and $\lambda_1 = \lambda_2 = 1/\sqrt{598}$. The results are shown in Fig. 4.5, in which the top row and the bottom row represents the low tubal rank parts and the sparse parts respectively. It is clear that the cloud and shadow contaminated areas are well recovered using our online tensor RPCA method.

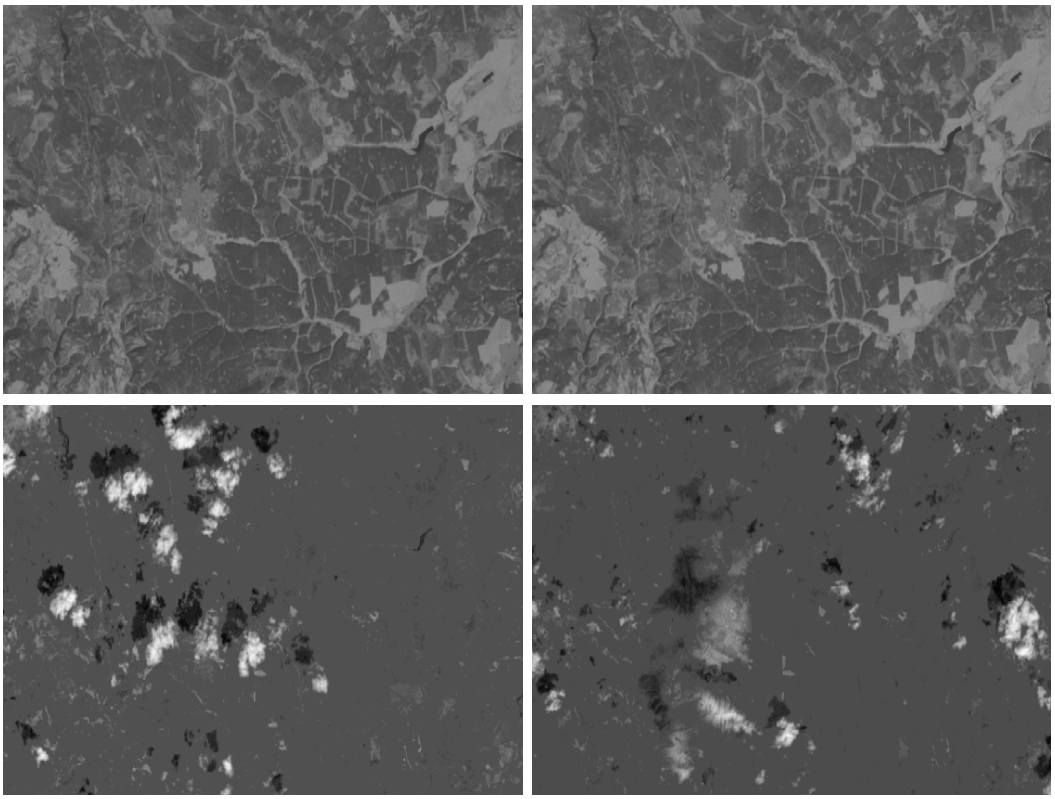


Figure 4.5: Tensor online Robust PCA results on the images shown in Fig. 4.4. The top row shows the low tubal rank components and the bottom row shows the sparse components.

In order to further evaluate our algorithm, we compare the tensor online robust PCA to the matrix online robust PCA [84] and the Grasta streaming [85]. Here we

synthesize a total number of 500 cloud-free images of the same location but different time as ground truth tensor $\mathcal{G}$. The images are of size 100×100 and intensity normalized to $[0, 1]$. We randomly add zero-mean Gaussian noise with standard deviation 10 to sparse locations of the images and use them as our input data samples $\mathcal{Z}$. The sparsity level varies from 0.02 to 0.2. We run over all the possible ranks using the tensor online robust PCA and the matrix online robust PCA, then choose the ranks which give the best performance on the two algorithms. In tensor online robust PCA, $\lambda_1 = \lambda_2 = 1/\sqrt{100} = 0.1$. For matrix online robust PCA and the Grasta streaming, each time the data image is vectorized into a long vector as input. At each round t we compute the relative square error up to time t defined as

$$\mathbf{RSE}(t) = \|\mathcal{X}_t - \mathcal{G}_t\|_F / \|\mathcal{G}_t\|_F, \tag{4.15}$$

where $\mathcal{X}_t$ and $\mathcal{G}_t$ are the low rank component and ground truth we get up to time t, respectively. The performance is compared in Fig. 4.6. We can tell that our tensor online RPCA has a smaller relative square error comparing to the matrix online robust PCA and Grasta streaming on all sparsity levels.

Figure 4.7 shows the convergence speed of the three algorithms when the sparsity level of the Gaussian noise is equal to 0.2. Our tensor online robust PCA converges the fastest among the three, which is one of the most important things in online algorithms. Consequently, when the total number of data samples are small, our proposed algorithm is more efficient and accurate than the other two methods.

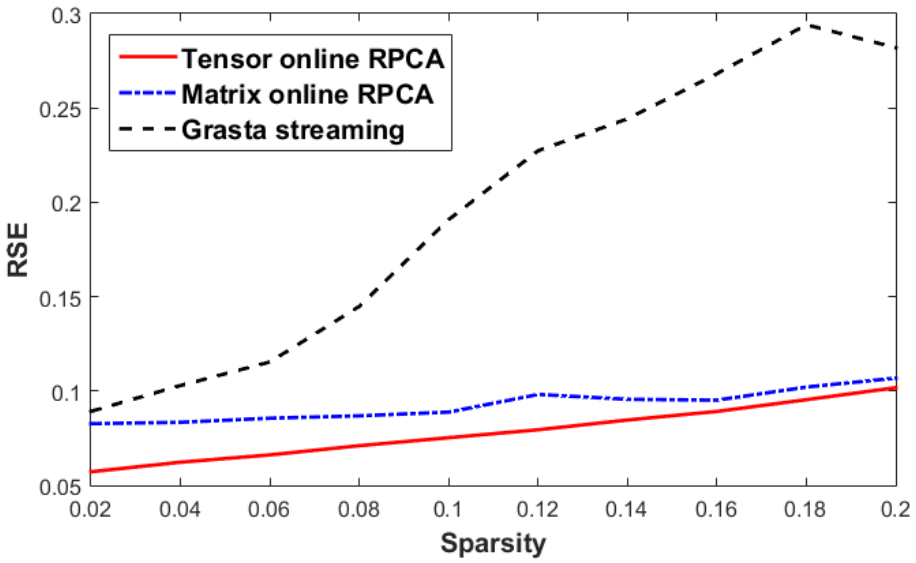


Figure 4.6: Comparison of tensor online robust PCA, matrix online robust PCA, and Grasta streaming version.

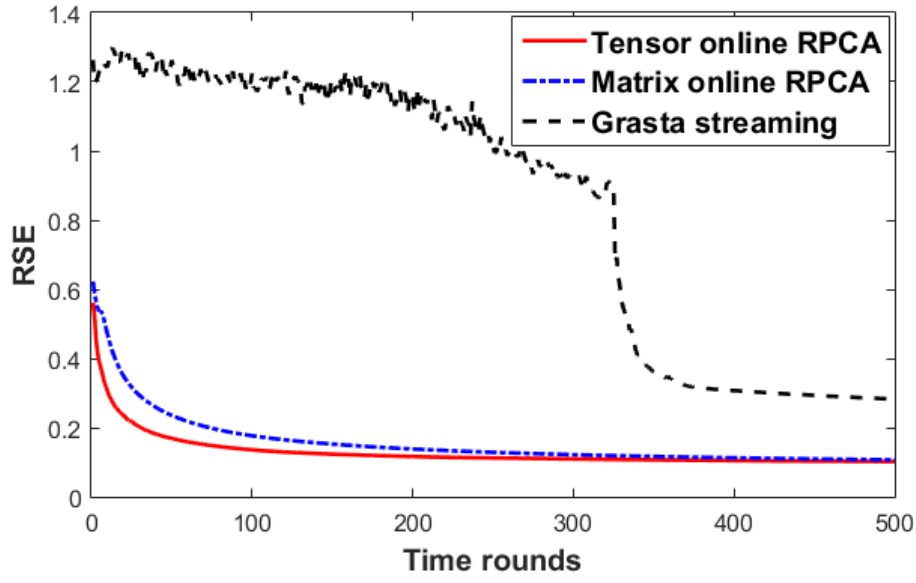


Figure 4.7: Convergence speed comparison when the sparsity of Gaussian noise is 0.2.

Chapter 5

Tensor Dictionary Learning

5.1 Introduction

Sparsity driven signal processing has been widely used in many areas across computer vision and image analysis, such as image restoration and classification [86, 87, 88]. The main principle driving the gains is the idea of sparse coding, i.e. the underlying signal is compactly represented by a few large coefficients in the overcomplete dictionary, while the noise and the sampling process are incohrent. Since the performance heavily relies on the chosen dictionary, a lot of dictionary learning algorithms are developed to obtain dictionaries that are more adapted to the signal than the predefined ones, such as wavelet and DCT. In [86], Aharon *et al.* proposed an algorithm called K-SVD, which efficiently learns an overcomplete dictionary from a set of training signals. The method of optimal directions (MOD)[89] shares the same effective sparse coding

principle for dictionary learning as K-SVD. The discriminative K-SVD algorithm (D-KSVD) proposed in [90] improved the K-SVD method by unifying the dictionary and classifier learning processes. In [91], the authors efficiently accelerated the K-SVD algorithm and reduced its memory consumption using a batch orthogonal matching pursuit method.

When the signal is not limited to two dimensional signals, traditional methods generally embed the high dimensional data into a vector space by vectorizing the data points; therefore the conventional matrix based approaches can still be used. This kind of vectorization, however, will break the original multidimensional structure of the signal and reduce the reliability of post processing. To this end, some dictionary learning techniques have been explored based on different tensor decompositions such as CP decomposition [41, 92], Tucker Decomposition[93, 94, 95] and tensor-SVD[96]. In [41], the authors developed an algorithm called K-CPD which learns high order dictionaries based on the CP decomposition. [93] proposed a tensor dictionary learning algorithm based on the Tucker model with sparsity constraints over its core tensor, and applied gradient descent algorithm to learn overcomplete dictionaries along each mode of the tensor (see [97] for definition of tensor modes). Peng et al. The authors of [94] presented a tensor dictionary learning algorithm based on Tucker model with Group-block-sparsity constraint on the core tensor with good performance. In [96], the authors considered the problem of tomographic reconstruction using priors in the form of a dictionary learned based on t-SVD, in which the

dictionary learning problem is presented as non-negative tensor factorization problem with sparsity constraints.

In this chapter, we present a new multidimensional dictionary learning approach based on a notion of tensor-SVD proposed in [1, 2, 3]. Essentially the t-SVD is based on an operator theoretic interpretation of the 3rd order tensors [1], as linear operators over the set of 2-D matrices. This framework has recently been used for dictionary learning for 2-D images in [96], but the authors there employ a different algorithm and the problem considered is tomographic image reconstruction. Moreover we will also consider the problem of filling in missing data by sparse coding using the learned dictionary.

5.2 Problem Formulation

In this section, we introduce our tensor dictionary learning model and the related algorithm.

5.2.1 t-linear Combination of Tensor Dictionaries and Coefficients

As in the matrix case, given an overcomplete dictionary $D \in \mathbb{R}^{n \times K}$ which contains K prototype signal-atoms for columns, a signal $y \in \mathbb{R}^n$ can be represented as a linear

combination of columns of D

$$y = Dx, \tag{5.1}$$

where $x \in \mathbb{R}^K$ is called the representation coefficient vector of y. This set up could be easily extended to 3rd order tensors using the framework outlined in the previous section. Given K tensor columns (or dictionary atoms) $\vec{\mathcal{D}}_k \in \mathbb{R}^{n_1 \times 1 \times n_3}$, we represent a tensor signal $\vec{\mathcal{X}} \in \mathbb{R}^{n_1 \times 1 \times n_3}$ using the ***t-linear combination***[3] of the given tensor dictionaries as follows,

$$\vec{\mathcal{X}} = \sum_{k=1}^{K} \vec{\mathcal{D}}_k * \vec{c}_k = \mathcal{D} * \vec{\mathcal{C}}, \tag{5.2}$$

where $\{\vec{c}_k\}_{k=1}^K$ are tubes of size $1 \times 1 \times n_3$; $\vec{\mathcal{C}} \in \mathbb{R}^{K \times 1 \times n_3}$ is called coefficient tensor obtained by aligning all the $\vec{c}_k$. $\mathcal{D} = \{\vec{\mathcal{D}}_1, \vec{\mathcal{D}}_2, ..., \vec{\mathcal{D}}_K\} \in \mathbb{R}^{n_1 \times K \times n_3}$ is the tensor dictionary. The representation (5.2) may either be exact or approximate satisfying

$$\|\vec{\mathcal{X}} - \mathcal{D} * \vec{\mathcal{C}}\| \leq \epsilon, \tag{5.3}$$

for some $\epsilon > 0$. When $K > n$, we say the tensor dictionary $\mathcal{D}$ is overcomplete.

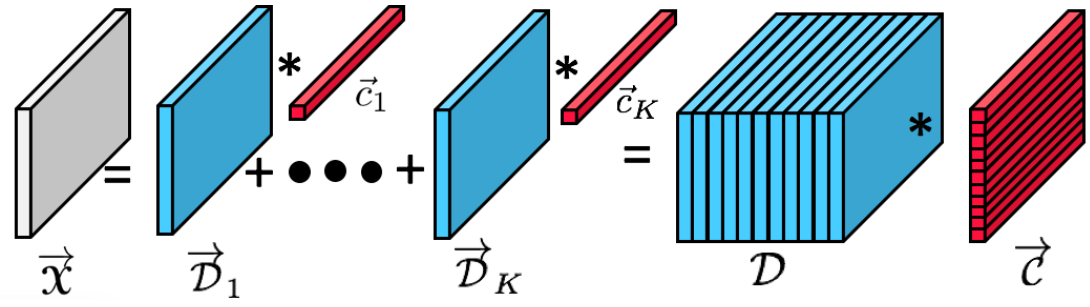


Figure 5.1: A tensor signal represented by a t-linear combination of K tensor dictionary atoms.

5.2.2 From Matrix to Tensor Dictionary Learning

Given an overcomplete dictionary $D \in \mathbb{R}^{n\times K}$ with $K > n$, if D is full rank, there are infinite number of solutions to the representation problem (5.1); therefore in order to constrain the solution set, one common approach is to enforce sparsity. As in classic dictionary learning model which was first designed for the purpose of reconstruction, one adaptively learns an overcomplete dictionary using the training data, which leads to the best possible representation of the data with sparsity constraints. Specifically, given training data $\{y_i\}_{i=1}^n \in \mathbb{R}^d$ where d is the dimensionality and n is the total number of training data used, dictionary learning methods aim at finding an overcomplete dictionary $D \in \mathbb{R}^{d\times K}$ with $K > d$, and a coefficient matrix $X = [x_1, x_2, ..., x_n] \in \mathbb{R}^{K\times n}$ by the following optimization problem,

$$\begin{aligned} &\min_{D,X} \ \sum_{i=1}^{n} \|y_i - Dx_i\|_F^2 \\ &\text{subject to} \quad \|x_i\|_q \leq T, \ \ i = 1, 2, ..., n, \end{aligned} \tag{5.4}$$

where $\|\cdot\|_q, q \geq 1$ is the ℓ_q norm which represents different sparsity regularization.

Using t-SVD structure discussed in the previous section, we generalize this dictionary learning model to higher dimensional cases. Given training data as tensor columns $\{\vec{\boldsymbol{\mathcal{Y}}}_i\}_{i=1}^n \in \mathbb{R}^{d\times 1\times n_3}$, we want to find a dictionary $\boldsymbol{\mathcal{D}} \in \mathbb{R}^{n\times K\times n_3}$ with $K > n$, and "***tubal sparse***" tensor coefficients $\{\vec{\boldsymbol{\mathcal{X}}}_i\}_{i=1}^n \in \mathbb{R}^{K\times 1\times n_3}$ to represent the training data using t-product. The tubal sparsity of a tensor column is defined in Chapter 3 as follows.

Definition 5.2.1. ***(tensor tubal sparsity)*** *Given a tensor column* $\vec{\mathcal{X}}$*, the tensor tubal sparsity* $\|\cdot\|_{TS}$ *is defined as the number of non-zero tubes of* $\vec{\mathcal{X}}$ *in the third dimension.*

Then we can construct our dictionary learning model:

$$\begin{aligned} &\min_{\mathcal{D},\vec{\mathcal{X}}_i} \sum_{i=1}^{n} \|\vec{\mathcal{Y}}_i - \mathcal{D} * \vec{\mathcal{X}}_i\|_F^2 \\ &\text{subject to } \|\vec{\mathcal{X}}_i\|_{\text{TS}} \leq T, \;\; i = 1, 2, ..., n, \end{aligned} \tag{5.5}$$

or equivalently,

$$\begin{aligned} &\min_{\mathcal{D},\mathcal{X}} \|\mathcal{Y} - \mathcal{D} * \mathcal{X}\|_F^2 \\ &\text{subject to } \|\mathcal{X}\|_{\text{TS}} \leq T_0, \end{aligned} \tag{5.6}$$

where $\mathcal{Y} = \left[\vec{\mathcal{Y}}_1, \vec{\mathcal{Y}}_2, ..., \vec{\mathcal{Y}}_n\right] \in \mathbb{R}^{d\times n\times n_3}$ and $\mathcal{X} = \left[\vec{\mathcal{X}}_1, \vec{\mathcal{X}}_2, ..., \vec{\mathcal{X}}_n\right] \in \mathbb{R}^{K\times n\times n_3}$. Figure 5.2 illustrates the tensor sparse coding model. Note that if the jth tube of $\vec{\mathcal{X}}_i(j, 1, :)$ is zero, then it means that the jth dictionary $\mathcal{D}(:, j, :)$ is not being used in the representation of $\vec{\mathcal{Y}}_i$.

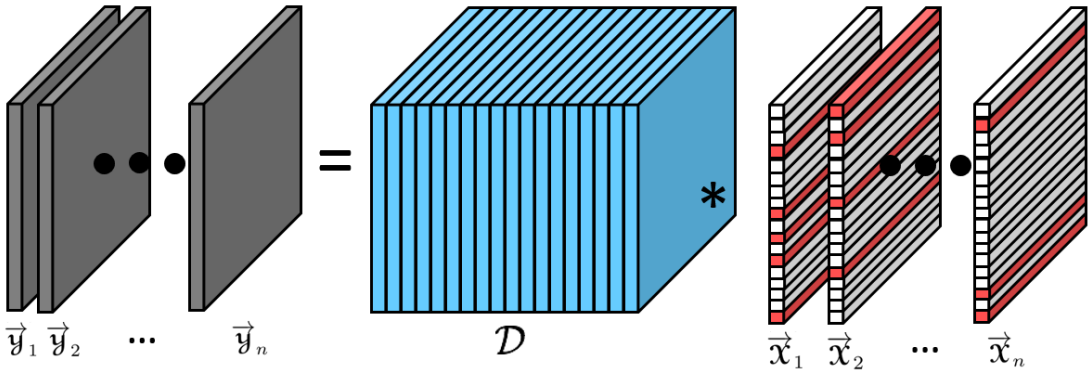


Figure 5.2: Data in the form of tensor columns represented by the t-product of tensor dictioanry and tubal-sparse coefficient tensors. The red tubes in the coefficient tensors stand for the non-zero tubes and white ones are zero tubes.

5.2.3 K-TSVD

We now discuss our tensor dictionary learning model in detail. Our model is called K-TSVD since it is a general extension from classic K-SVD to high dimensional tensor based on t-SVD. Similarly to K-SVD algorithm, K-TSVD also consists of two stages: the tensor sparse coding stage and the tensor dictionary update stage. First let's consider the sparse coding stage where the tensor dictionary $\mathcal{D}$ is fixed. So we need to solve

$$\begin{aligned} \min_{\mathcal{X}} \ & \|\mathcal{Y} - \mathcal{D} * \mathcal{X}\|_F^2 \\ \text{subject to } & \|\mathcal{X}\|_{\text{TS}} \leq T_0, \end{aligned} \tag{5.7}$$

or alternatively we can work on an equivalent form,

$$\min_{\vec{\mathcal{X}}_i} \|\mathcal{Y} - \mathcal{D} * \mathcal{X}\|_F^2 + \lambda \|\mathcal{X}\|_{\text{TS}}, \tag{5.8}$$

for some positive λ. Since the sparsity measure is computational intractable in both matrix and tensor cases, we use the $\|\cdot\|_{1,1,2}$ norm instead as a convex relaxation for the tubal sparsity, where the $\|\cdot\|_{1,1,2}$ norm of a 3rd order tensor $\mathcal{X}$ is defined as

$$\|\mathcal{X}\|_{1,1,2} = \sum_{i,j} \|\mathcal{X}(i,j,:)\|_F$$

If we regard a third dimensional tube $\vec{x} \in \mathbb{R}^{1\times 1\times n_3}$ as a $n_3 \times 1$ column vector, then the $\ell_{1,1,2}$ norm of $\mathcal{X}$ is just the summation of ℓ_2 norm of all such tubes along the third dimension in $\mathcal{X}$.

Replacing the tubal sparsity with the $\ell_{1,1,2}$ norm, the problem becomes

$$\min_{\mathcal{X}} \|\mathcal{Y} - \mathcal{D} * \mathcal{X}\|_F^2 + \lambda \|\mathcal{X}\|_{1,1,2} \tag{5.9}$$

Using the block-diagonal form, (5.9) can be equivalently reformulated in Fourier domain as

$$\min_{\overline{\mathcal{X}}} \|\overline{\mathcal{Y}} - \overline{\mathcal{D}}\overline{\mathcal{X}}\|_F^2 + \lambda\sqrt{n_3}\|\widehat{\mathcal{X}}\|_{1,1,2}$$

where the $\sqrt{n_3}$ factor comes from the fact that $\|\mathcal{X}\|_F = \|\widehat{\mathcal{X}}\|_F/\sqrt{n_3}$. Use the general framework of Alternating Direction Method of Multipliers (ADMM) [77], we can solve this optimization problem recursively with the following algorithm:

$$\overline{\mathcal{X}}_{k+1} = \arg\min_{\overline{\mathcal{X}}} \|\overline{\mathcal{Y}} - \overline{\mathcal{D}}\overline{\mathcal{X}}\|_F^2 + \mathrm{tr}(\overline{\mathcal{Q}}_k^\top \overline{\mathcal{X}}) + \frac{\rho}{2}\|\overline{\mathcal{X}} - \overline{\mathcal{Z}}_k\|_F^2 \tag{5.10}$$

$$\mathcal{Z}_{k+1} = \arg\min_{\mathcal{Z}} \|\mathcal{Z}\|_{1,1,2} + \frac{\rho}{2\lambda}\|\mathcal{X}_{k+1} + \frac{1}{\rho}\mathcal{Q}_k - \mathcal{Z}\|_F^2 \tag{5.11}$$

$$\mathcal{Q}_{k+1} = \mathcal{Q}_k + \rho(\mathcal{X}_{k+1} - \mathcal{Z}_{k+1}) \tag{5.12}$$

where $\mathcal{Q} \in \mathbb{R}^{n_1 \times n_2 \times n_3}$ is a helper tensor which comes from the framework of ADMM and could be initialized as a zero tensor, $\rho > 0$. (5.10) is essentially a least square minimization problem and we can separately solve it in each frontal slice of $\widehat{\mathcal{X}}$ (or equivalently, each diagonal block of $\overline{\mathcal{X}}$). Let $\mathcal{C}_{k+1} = \mathcal{X}_{k+1} + \mathcal{Q}_k/\rho$, the update of (5.11) is given by

$$\begin{gathered}\mathcal{Z}_{k+1}(i,j,:) = \left(1 - \frac{\lambda}{\rho\|\mathcal{C}_k(i,j,:)\|_F}\right)_+ \mathcal{C}(i,j,:) \\ \forall i = 1,2,...,K,\ j = 1,2,...,n\end{gathered} \tag{5.13}$$

where $(\cdot)_+ = \max(0,\cdot)$.

The second stage of our tensor dictionary learning model is dictionary update. Given fixed $\mathcal{D}$ and $\mathcal{X}$, suppose we only want to update the k-th element of $\mathcal{D}$, we

can decompose the error term as follows,

$$\begin{aligned}
&\|\mathcal{Y} - \mathcal{D} * \mathcal{X}\|_F^2 \\
=&\left\|\mathcal{Y} - \sum_{j=1}^{K} \overrightarrow{\mathcal{D}}_j * \mathcal{X}(j,:,:)\right\|_F^2 \\
=&\left\|\left(\mathcal{Y} - \sum_{j\neq k} \overrightarrow{\mathcal{D}}_j * \mathcal{X}(j,:,:)\right) - \overrightarrow{\mathcal{D}}_k * \mathcal{X}(k,:,:)\right\|_F^2 \\
=&\|\mathcal{E}_k - \overrightarrow{\mathcal{D}}_k * \mathcal{X}(k,:,:)\|_F^2 \\
=&\|\mathcal{E}_k - \mathcal{D}(:,k,:) * \mathcal{X}(k,:,:)\|_F^2
\end{aligned}$$

$\mathcal{E}_k$ here stands for the representation error when the k-th atom $\mathcal{D}(:,k,:)$ is removed from the dictionary. The next step is to find $\mathcal{D}(:,k,:) * \mathcal{X}(k,:,:)$ which best approximates $\mathcal{E}_k$, so that the error term is minimized. This is essentially to compute the best tubal rank-1 approximation using Lemma 2.0.1. Since we need to maintain the tubal sparsity of $\mathcal{X}$ and avoid to fully fill $\mathcal{X}(k,:,:)$, let $w_k = \{i|\mathcal{X}(k,i,:) \neq 0, i = 1,2,...,n\}$ be the set of indices where data $\mathcal{Y}$ uses tensor dictionary $\mathcal{D}(:,k,:)$ and restrict $\mathcal{E}_k$ by choosing the tensor columns corresponding to w_k to obtain $\mathcal{R}_k$: $\mathcal{R}(:,i,:) = \mathcal{E}(:,w_k(i),:), i = 1,2,...,|w_k|$. From Lemma 2.0.1, we apply t-SVD on $\mathcal{R}_k$ to get $\mathcal{U}, \mathcal{S}$ and $\mathcal{V}$, and take the first tensor column of $\mathcal{U}$ to update $\mathcal{D}(:,k,:)$, use $\mathcal{S}(1,1,:) * \mathcal{V}(:,1,:)^\top$ to renovate the coefficient tensors which use the k-th dictionary. To accelerate the algorithm we only compute the approximate rank-1 SVDs in Fourier domain when we compute t-SVD of $\mathcal{R}$. The complete algorithm is presented in Algorithm 5. Note that the optimization problem in Step 1 is convex and we use ADMM [77] with global convergence guarantees. Similarly to K-SVD, the dictionary update stage is

only guaranteed to converge to a local minimum theoretically. However, in practice the output dictionary of K-TSVD performs very well as we will see in the next section.

Algorithm 5 K-TSVD

Input : Observed tensor data $\mathcal{Y} = \{\overrightarrow{\mathcal{Y}}_i\}_{i=1}^{n_2} \in \mathbb{R}^{n_1 \times n_2 \times n_3}$, $\lambda > 0$.
Initialize: Dictionary $\mathcal{D}_0 \in \mathbb{R}^{n_1 \times K \times n_3}$
Repeat until convergence:

1: Compute the sparse coefficient tensor using (5.10)-(5.12):

$$\mathcal{X} = \arg\min_{\mathcal{X}} \ \|\mathcal{Y} - \mathcal{D} * \mathcal{X}\|_F^2 + \lambda \|\mathcal{X}\|_{1,1,2}$$

2: **for** $k = 1, 2, ..., K$ **do**
3: Let $w_k = \{i | \mathcal{X}(k, i, :) \neq 0\}$ be the set of indices where data $\mathcal{Y}$ uses dictionary $\mathcal{D}(:, k, :)$.
4: Compute $\mathcal{E}_k = \mathcal{Y} - \sum_{j \neq k} \mathcal{D}(:, j, :) * \mathcal{X}(j, :, :)^\top$, which is the over all error without using the k-th dictionary atom $\mathcal{D}(:, k, :)$.
5: Restrict $\mathcal{E}_k$ by choosing only the tensor columns corresponding to w_k and obtain $\mathcal{R}_k$:

$$\mathcal{R}(:, i, :) = \mathcal{E}(:, w_k(i), :) \tag{5.14}$$

for $i = 1, 2, ..., |w_k|$.
6: Compute the t-SVD of $\mathcal{R}_k$:

$$\mathcal{R}_k = \mathcal{U} * \mathcal{S} * \mathcal{V}^\top.$$

7: Update $\mathcal{D}(:, k, :) = \mathcal{U}(:, 1, :)$.
8: Update $\mathcal{X}(k, w_k, :) = \mathcal{S}(1, 1, :) * \mathcal{V}(:, 1, :)^\top$.
9: **end for**

Output: Trained tensor dictionary $\mathcal{D}$.

5.3 Experiment Results

5.3.1 Filling Missing Pixels in Tensors

In this section we consider the application of filling missing pixels in third order tensors. Suppose that we are given a video with dead pixels, where the dead pixels mean pixel values are deleted or missing on some fixed positions of each frame. Specifically, let Ω indicate the set of indices of the remaining pixels and $\mathcal{M}$ be the data tensor, then $\mathcal{M}(i,j,:) = 0$ for all $(i,j) \notin \Omega$. Our goal is to recover such tensors with missing pixels. Suppose $\mathcal{D}$ is the learned overcomplete dictionary on the training data, define P_Ω as an orthogonal projector such that $P_\Omega(\mathcal{M})(i,j,:) = \mathcal{M}(i,j,:)$, if $(i,j) \in \Omega$ and 0 otherwise. Then for each patch $\overrightarrow{\mathcal{M}}_k$ in the test data, the reconstruction of this patch is $\mathcal{D} * \overrightarrow{\mathcal{C}}_k$, where $\overrightarrow{\mathcal{C}}_k$ is the solution to

$$\min_{\overrightarrow{\mathcal{C}}_k} \|P_\Omega(\overrightarrow{\mathcal{M}}_k) - P_\Omega(\mathcal{D} * \overrightarrow{\mathcal{C}}_k)\|_F^2 + \lambda \|\overrightarrow{\mathcal{C}}_k\|_{1,1,2} \tag{5.15}$$

which can be solved in the same manner as (5.9).

We utilized a basketball video here to apply K-TSVD algorithm and reconstruct $\mathcal{M}$ from missing pixels. There are 40 frames in the video and the resolution of each frame is 144×256. To learn the overcomplete dictionary using K-TSVD, we randomly took 9000 overlapping block patches of size $8 \times 8 \times 10$ from the first 30 frames, saved them as tensor columns of size $64 \times 1 \times 10$, and obtained our training data $\mathcal{Y}$ of total size $64 \times 9000 \times 10$. All these patches were used to train a tensor dictionary with $K = 256$ atoms. The last 10 frames of the video were used for testing. We took the

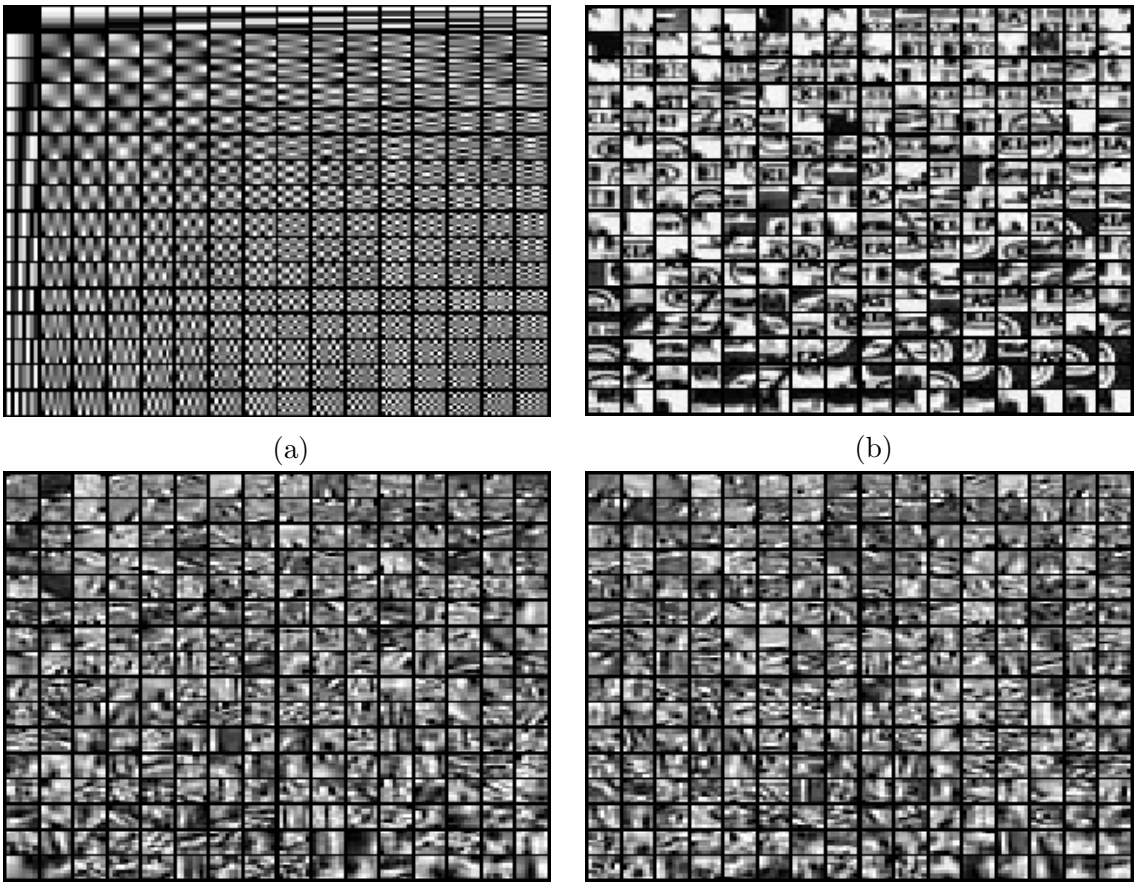
total 576 disjoint $8 \times 8 \times 10$ blocks in the last 10 frames, saved each block into a tensor column, and obtained our training data of size $64 \times 576 \times 10$.

We investigated the performance of K-TSVD by comparing it with K-SVD and pre-fixed overcomplete DCT dictionary. In K-SVD, in order to have a fair comparison, for each test frame we also randomly trained 10000 block patches of size 8×8 in the first 30 frames. We visualize an example of the overcomplete DCT dictioanry, the K-SVD learned dictionary and the K-TSVD learned dictionary in Figure 5.3. One frame with 50% and 70% missing pixels and its reconstructions are shown in Figure 5.4. As one can see the reconstruction based on K-TSVD learned dictionary has a better quality. Figure 5.4(e) shows the reconstruction error (RE) comparison of those three approaches, where the error is computed via $\text{RE} = \sqrt{\|\boldsymbol{\mathcal{X}} - \boldsymbol{\mathcal{X}}_{\text{rec}}\|_F^2/N}$, N is the total number of pixels in the data. We can see that when the percentage of missing pixels is small, all three methods perform equally well. With more missing pixels, K-TSVD gives better performance over the other two methods.

5.3.2 Multispectral Image and Video Denoising

In order to further test the proposed method, we applied our algorithm on multi-spectral/hyperspectral images and video data denoising. In the first experiment the multispectral data was from the **Columbia datasets** [1], each dataset contains 31 real-world images of size 512×512 and is collected from 400nm to 700nm at $10nm$

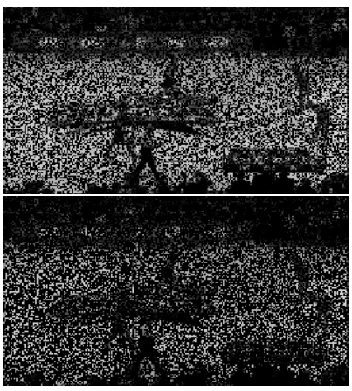
[1] `http://www1.cs.columbia.edu/CAVE/databases/multispectral/`



(a) (b)

(c) (d)

Figure 5.3: (a) The overcomplete DCT dictionary. (b) Dictionary learned on the first frame of the basketball video using K-SVD. (c) The first frontal slice $\boldsymbol{\mathcal{D}}(:,:,1)$ of the learned dictionary of the tensor. (d) The 3rd frontal slice $\boldsymbol{\mathcal{D}}(:,:,3)$ of the learned dictionary of the tensor.

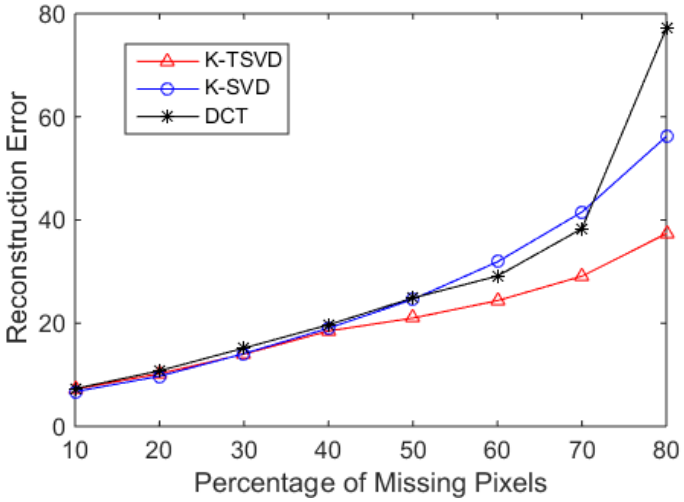


(a) Pixels missing

(b) DCT recovery

(c) K-SVD recovery

(d) K-TSVD recovery



(e) Reconstruction error comparison.

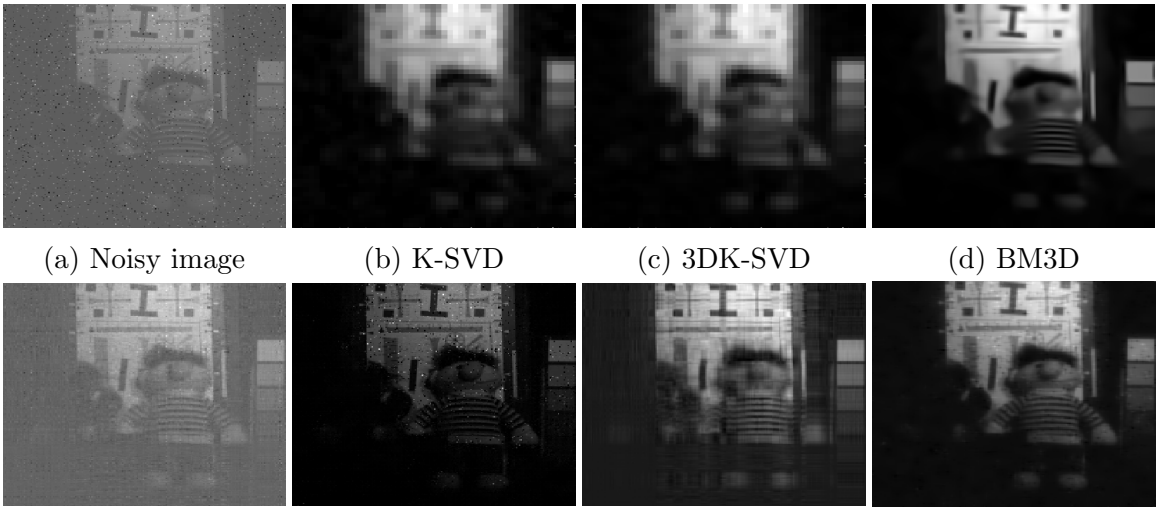
Figure 5.4: The reconstruction result from missing pixels on the basketball video. The different rows are for 50% and 70% of missing pixels respectively.

steps. In our experiment we resized each image into size of 205×205, and took images of the last 10 bands to accelerate the speed of training tensor dictionaries. Therefore the total size of the tensor data we used here is $205 \times 205 \times 10$. Further work is required to fully deploy the algorithm in large-scale high order tensor applications.

For the noise model we consider the fixed-location defects without knowing the noisy positions, which commonly exists in video and multispectral images. On image of each bandwidth, some fixed pixel locations are corrupted with very high noise and our task is to recover the image. Specifically in our experiment we picked a sparse number of pixel locations and added Gaussian noise on these positions of each image. Let Ω indicate the set of noisy pixel locations, then what we did was for each $(i, j) \in \Omega$, $k = 1, 2, ..., 10$, $\widehat{\mathcal{Y}}(i, j, k) = \mathcal{Y}(i, j, k) + w_{ijk}$, where $\mathcal{Y}$ is the clean tensor and $w_{ijk} \sim \mathcal{N}(0, \sigma)$ is the additive Gaussian noise.

To train the data and learn the dictionaries, similar to what we did in the previous experiment, we randomly took 10000 overlapping patches of size $8 \times 8 \times 10$ from the noisy tensor data, which is about a quarter of all the overlapping patches in the data. For a fair comparison, in K-SVD we also randomly select 10000 overlapping patches of size 8×8 within each noisy image.

The denoising process of our method includes a tensor sparse coding stage based on the learned tensor dictionary. We extracted each $8 \times 8 \times 10$ patch in the noisy multispectral images and solved the tensor sparse coding problem (5.9) to obtain the denoised patch. Following a similar idea in [98], we averaged all the denoised patches



(a) Noisy image (b) K-SVD (c) 3DK-SVD (d) BM3D

(e) LRTA (f) DNMDL (g) PARAFAC (h) K-TSVD

Figure 5.5: Denoised image at the 610nm band of chart and stuffed toy. The sparsity of the noisy pixels is 10% and the locations of noisy pixels are consistent on image of each band. The additive noise is Gaussian with $\sigma = 100$.

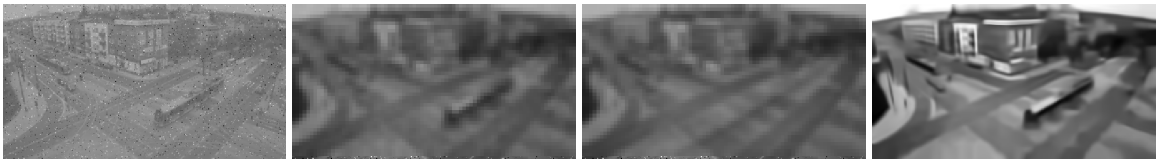
with some relaxation obtained by averaging with the original noisy data then got our denoised tensor.

To test the performance of our method, we compared our K-TSVD to these methods: K-SVD (band-wise)[86, 98] 3D K-SVD [98], BM3D (band-wise) [99], LRTA[100], DNMDL[94] and PARAFAC[101]. The result with $\sigma = 100$ and the sparsity of noisy pixels equaling 10% is shown in Figure 5.5. The detailed PSNR comparison on different noise levels of these methods is in Table 5.1. We can see that our algorithm has a better performance over the other competing methods on most cases.

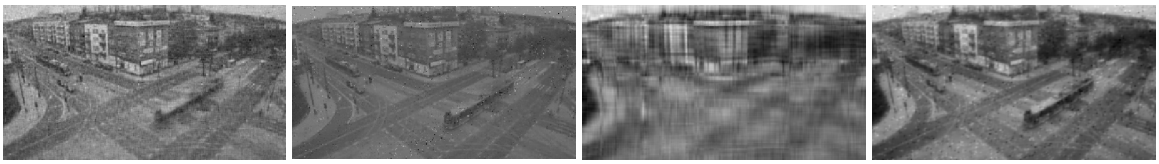
We also applied K-TSVD algorithm on video denoising. The video that we used here was footage from a still camera view of a traffic intersection [102]. The resolution of each frame is 175×328, and we performed our method on every 10 frames. Figure 5.6 shows one frame of the denoising result with sparsity = 10% and noise

Table 5.1: PSNR(dB) of chart and stuffed toy images.

Sparsity Noise level	5% 100	10% 100	15% 100	10% 150	10% 200
Noisy image	20.96	18.18	16.35	14.75	12.10
K-SVD	22.73	22.60	22.49	22.38	22.00
3DK-SVD	22.61	22.53	22.47	22.41	22.20
BM3D	26.95	26.62	26.36	25.23	24.29
LRTA	23.54	26.84	26.65	23.90	22.03
DNMDL	24.07	23.73	25.16	17.89	16.83
PARAFAC	27.07	26.86	26.72	26.13	25.24
KTSVD	**27.19**	**26.98**	**26.79**	**26.18**	**25.44**



(a) Noisy image (b) K-SVD (c) 3DK-SVD (d) BM3D



(e) LRTA (f) DNMDL (g) PARAFAC (h) K-TSVD

Figure 5.6: Video denoising result. The sparsity is 10% and $\sigma = 100$.

level 100. As one can see in this experiment K-TSVD perform very well.

Chapter 6

Conclusions and Future Work

In this thesis, we firstly considered the problem of recovering third-order tensors under random sampling, based on the t-SVD framework. Using the notion of tensor tubal rank, we showed that under the certain tensor incoherence conditions, we could exactly recover a third-order tensor with low tubal-rank, and establish a theoretical bound for exact completion when using a convex optimization algorithm for recovery. Moreover, we presented the algorithms for recovery using ADMM. As applications we considered the problem of video and cellular data completion, and showed significant performance gains compared to the existing methods.

Secondly we considered batch and online tensor robust PCA methods, and proposed algorithms on them. Applications includes image denoising and fusing cloud-contaminated satellite images. We compared our methods to the state-of-the-art approaches and showed superiority in both convergence speed and performance.

We proposed a new tensor dictionary learning method in the end. Our K-TSVD method learned a multidimensional dictionary directly of the data tensor via t-SVD. The K-TSVD could be regarded as a general extension of K-SVD from matrix (2-D tensors) to higher-order tensors. We showed that the K-TSVD methods outperformed some classic methods in video/multispectral image completion and denoising.

A lot remains to be done in the future. For instance, the theoretical bound of exact recovery of tensor completion could be generalized to tensors with order higher than 3, in which the tensor tubal rank needs to be re-defined. In the batch and online robust PCA, more applications could be considered, such as object tracking and data completion. The tensor dictionary learning could also be generalized to higher-order tensor cases, where the sparse coding is orientation depended. In summary, there is a lot to explore in the area of tensor analysis, and I will continue working on them.

Chapter 7

Appendix

In this appendix, we provide all the detailed proofs to the tensor completion theory of Chapter 2 Section 2.3.

7.1 Proof of Proposition 3.3.1 Condition 1

The following theorem is first developed in [59], and it will be used frequently in this section.

Theorem 7.1.1. ***(Noncommutative Bernstein Inequality)***

Let $\mathbf{X}_1, \mathbf{X}_2, ..., \mathbf{X}_L$ *be independent zero-mean random matrices of dimension* $d_1 \times d_2$. *Suppose*

$$\rho_k^2 = \max\{\|\mathbb{E}[\mathbf{X}_k \mathbf{X}_k^\top]\|, \|\mathbb{E}[\mathbf{X}_k^\top \mathbf{X}_k]\|\}$$

and

$$\|\mathbf{X}_k\| \leq M$$

almost surely for all k. Then for any $\tau > 0$,

$$\mathbb{P}\left[\left\|\sum_{k=1}^{L}\mathbf{X}_k\right\| > \tau\right] \leq (d_1+d_2)\exp\left(\frac{-\tau^2/2}{\sum_{k=1}^{L}\rho_k^2 + M\tau/3}\right) \tag{7.1}$$

This theorem is a corollary of a Chernoff bound for finite dimension operators developed by [103]. An extension of this theorem[104] states that if

$$\max\left\{\left\|\sum_{k=1}^{L}\mathbf{X}_k\mathbf{X}_k^\top\right\|, \left\|\sum_{k=1}^{L}\mathbf{X}_k^\top\mathbf{X}_k\right\|\right\} \leq \sigma^2 \tag{7.2}$$

and let

$$\tau = \sqrt{4c\sigma^2\log(d_1+d_2)} + cM\log(d_1+d_2)$$

for any $c > 0$. Then (7.1) becomes

$$\mathbb{P}\left[\left\|\sum_{k=1}^{L}\mathbf{X}_k\right\| \geq \tau\right] \leq (d_1+d_2)^{-(c-1)} \tag{7.3}$$

The following fact is used frequently in this section.

Fact 7.1.1. $\left\|\boldsymbol{\mathcal{P}}_T(\vec{\boldsymbol{e}}_i * \mathring{\boldsymbol{e}}_k * \vec{\boldsymbol{e}}_j^\top)\right\|_F^2 \leq \frac{2\mu_0 r}{n}$

Proof of Fact 7.1.1. Following the definition of $\boldsymbol{\mathcal{P}}_T$ in (3.18), we have

$$\begin{aligned}
&\boldsymbol{\mathcal{P}}_T(\vec{\boldsymbol{e}}_i * \mathring{\boldsymbol{e}}_k * \vec{\boldsymbol{e}}_j^\top) \\
=&\boldsymbol{\mathcal{U}} * \boldsymbol{\mathcal{U}}^\top * \vec{\boldsymbol{e}}_i * \mathring{\boldsymbol{e}}_k * \vec{\boldsymbol{e}}_j^\top + \vec{\boldsymbol{e}}_i * \mathring{\boldsymbol{e}}_k * \vec{\boldsymbol{e}}_j^\top * \boldsymbol{\mathcal{V}} * \boldsymbol{\mathcal{V}}^\top - \boldsymbol{\mathcal{U}} * \boldsymbol{\mathcal{U}}^\top * \vec{\boldsymbol{e}}_i * \mathring{\boldsymbol{e}}_k * \vec{\boldsymbol{e}}_j^\top * \boldsymbol{\mathcal{V}} * \boldsymbol{\mathcal{V}}^\top
\end{aligned}$$

This gives

$$\begin{aligned}
&\left\|\boldsymbol{\mathcal{P}}_T(\vec{\boldsymbol{e}}_i * \mathring{\boldsymbol{e}}_k * \vec{\boldsymbol{e}}_j^\top)\right\|_F^2 \\
=&\langle \boldsymbol{\mathcal{P}}_T(\vec{\boldsymbol{e}}_i * \mathring{\boldsymbol{e}}_k * \vec{\boldsymbol{e}}_j^\top), \vec{\boldsymbol{e}}_i * \mathring{\boldsymbol{e}}_k * \vec{\boldsymbol{e}}_j^\top \rangle \\
=&\|\boldsymbol{\mathcal{U}} * \boldsymbol{\mathcal{U}}^\top * \vec{\boldsymbol{e}}_i\|_F^2 + \|\boldsymbol{\mathcal{V}} * \boldsymbol{\mathcal{V}}^\top * \vec{\boldsymbol{e}}_j\|_F^2 - \|\boldsymbol{\mathcal{U}} * \boldsymbol{\mathcal{U}}^\top * \vec{\boldsymbol{e}}_i\|_F^2 \|\boldsymbol{\mathcal{V}} * \boldsymbol{\mathcal{V}}^\top * \vec{\boldsymbol{e}}_j\|_F^2 \\
\leq&\mu_0 r \frac{n_1 + n_2}{n_1 n_2} \\
\leq&\frac{2\mu_0 r}{n}
\end{aligned}$$

where the first inequality comes from the tensor incoherence condition (3.6).

Proof of Proposition 3.3.1 **Condition** (1).

Proof. First note that

$$\mathbb{E}[\boldsymbol{\mathcal{P}}_T \boldsymbol{\mathcal{R}}_\Omega \boldsymbol{\mathcal{P}}_T] = \boldsymbol{\mathcal{P}}_T (\mathbb{E}\boldsymbol{\mathcal{R}}_\Omega) \boldsymbol{\mathcal{P}}_T = \boldsymbol{\mathcal{P}}_T \ , \tag{7.4}$$

which gives

$$\mathbb{E}[\boldsymbol{\mathcal{P}}_T \boldsymbol{\mathcal{R}}_\Omega \boldsymbol{\mathcal{P}}_T - \boldsymbol{\mathcal{P}}_T] = 0$$

and

$$\mathbb{E}[\overline{\boldsymbol{\mathcal{P}}_T}\,\overline{\boldsymbol{\mathcal{R}}_\Omega}\,\overline{\boldsymbol{\mathcal{P}}_T} - \overline{\boldsymbol{\mathcal{P}}_T}] = 0 \tag{7.5}$$

Our goal is to prove the operator $\overline{\boldsymbol{\mathcal{P}}_T}\,\overline{\boldsymbol{\mathcal{R}}_\Omega}\,\overline{\boldsymbol{\mathcal{P}}_T}$ is not far away from its expected value $\overline{\boldsymbol{\mathcal{P}}_T}$ in the spectral norm using the Noncommutative Bernstein Inequality.

Given any tensor $\mathfrak{Z}$ of size $n \times n \times n_3$, we can decompose $\boldsymbol{\mathcal{P}}_T(\mathfrak{Z})$ as the following

$$\begin{aligned}
&\boldsymbol{\mathcal{P}}_T(\mathfrak{Z}) \\
=&\sum_{i,j,k} \langle \boldsymbol{\mathcal{P}}_T(\mathfrak{Z}), \vec{\boldsymbol{e}}_i * \mathring{\boldsymbol{e}}_k * \vec{\boldsymbol{e}}_j^\top \rangle \vec{\boldsymbol{e}}_i * \mathring{\boldsymbol{e}}_k * \vec{\boldsymbol{e}}_j^\top \\
=&\sum_{i,j,k} \langle \mathfrak{Z}, \boldsymbol{\mathcal{P}}_T(\vec{\boldsymbol{e}}_i * \mathring{\boldsymbol{e}}_k * \vec{\boldsymbol{e}}_j^\top) \rangle \vec{\boldsymbol{e}}_i * \mathring{\boldsymbol{e}}_k * \vec{\boldsymbol{e}}_j^\top
\end{aligned}$$

This gives

$$\boldsymbol{\mathcal{R}}_\Omega \boldsymbol{\mathcal{P}}_T(\mathfrak{Z}) = \sum_{i,j,k} \frac{1}{p} \delta_{ijk} \langle \mathfrak{Z}, \boldsymbol{\mathcal{P}}_T(\vec{\boldsymbol{e}}_i * \mathring{\boldsymbol{e}}_k * \vec{\boldsymbol{e}}_j^\top) \rangle \vec{\boldsymbol{e}}_i * \mathring{\boldsymbol{e}}_k * \vec{\boldsymbol{e}}_j^\top$$

and

$$\boldsymbol{\mathcal{P}}_T \boldsymbol{\mathcal{R}}_\Omega \boldsymbol{\mathcal{P}}_T(\mathfrak{Z}) = \sum_{i,j,k} \frac{1}{p} \delta_{ijk} \langle \mathfrak{Z}, \boldsymbol{\mathcal{P}}_T(\vec{\boldsymbol{e}}_i * \mathring{\boldsymbol{e}}_k * \vec{\boldsymbol{e}}_j^\top) \rangle \boldsymbol{\mathcal{P}}_T(\vec{\boldsymbol{e}}_i * \mathring{\boldsymbol{e}}_k * \vec{\boldsymbol{e}}_j^\top)$$

which implies

$$\overline{\boldsymbol{\mathcal{P}}_T} \overline{\boldsymbol{\mathcal{R}}_\Omega} \overline{\boldsymbol{\mathcal{P}}_T}(\overline{\mathfrak{Z}}) = \sum_{i,j,k} \frac{1}{p} \delta_{ijk} \langle \mathfrak{Z}, \boldsymbol{\mathcal{P}}_T(\vec{\boldsymbol{e}}_i * \mathring{\boldsymbol{e}}_k * \vec{\boldsymbol{e}}_j^\top) \rangle \overline{\boldsymbol{\mathcal{P}}_T}(\overline{\vec{\boldsymbol{e}}_i * \mathring{\boldsymbol{e}}_k * \vec{\boldsymbol{e}}_j^\top})$$

Define operator $\boldsymbol{\mathcal{T}}_{ijk}$ which maps $\mathfrak{Z}$ to $\frac{1}{p} \delta_{ijk} \langle \mathfrak{Z}, \boldsymbol{\mathcal{P}}_T(\vec{\boldsymbol{e}}_i * \mathring{\boldsymbol{e}}_k * \vec{\boldsymbol{e}}_j^\top) \rangle \boldsymbol{\mathcal{P}}_T(\vec{\boldsymbol{e}}_i * \mathring{\boldsymbol{e}}_k * \vec{\boldsymbol{e}}_j^\top)$. Observe that $\|\boldsymbol{\mathcal{T}}_{ijk}\|_{op} = \|\overline{\boldsymbol{\mathcal{T}}_{ijk}}\| = \frac{1}{p} \|\boldsymbol{\mathcal{P}}_T(\vec{\boldsymbol{e}}_i * \mathring{\boldsymbol{e}}_k * \vec{\boldsymbol{e}}_j^\top)\|_F^2$ and $\|\boldsymbol{\mathcal{P}}_T\|_{op} = \|\overline{\boldsymbol{\mathcal{P}}_T}\| \leq 1$. Then we have

$$\begin{aligned}
&\|\boldsymbol{\mathcal{T}}_{ijk} - \frac{1}{n^2 n_3} \boldsymbol{\mathcal{P}}_T\|_{op} \\
=&\|\overline{\boldsymbol{\mathcal{T}}_{ijk}} - \frac{1}{n^2 n_3} \overline{\boldsymbol{\mathcal{P}}_T}\| \\
\leq& \max\left\{ \frac{1}{p} \|\boldsymbol{\mathcal{P}}_T(\vec{\boldsymbol{e}}_i * \mathring{\boldsymbol{e}}_k * \vec{\boldsymbol{e}}_j^\top)\|_F^2, \frac{1}{n^2 n_3} \right\} \\
\leq& \frac{2\mu_0 r}{np}
\end{aligned}$$

where the first inequality uses the fact that if A and B are positive semidefinite matrices, then $\|A - B\| \leq \max\{\|A\|, \|B\|\}$.

On the other hand, from the definition of $\boldsymbol{\mathcal{T}}_{ijk}$ we have $\mathbb{E}[\boldsymbol{\mathcal{T}}_{ijk}] = \frac{1}{n^2 n_3}\boldsymbol{\mathcal{P}}_T$. So

$$
\begin{aligned}
&\left\|\mathbb{E}[(\overline{\boldsymbol{\mathcal{T}}_{ijk}} - \frac{1}{n^2 n_3}\overline{\boldsymbol{\mathcal{P}}}_T)^2]\right\| \\
=&\left\|\mathbb{E}[(\boldsymbol{\mathcal{T}}_{ijk} - \frac{1}{n^2 n_3}\boldsymbol{\mathcal{P}}_T)^2]\right\|_{op} \\
=&\left\|\mathbb{E}[\frac{1}{p}\|\boldsymbol{\mathcal{P}}_T(\vec{\boldsymbol{e}}_i * \mathring{\boldsymbol{e}}_k * \vec{\boldsymbol{e}}_j^{\top})\|_F^2 \boldsymbol{\mathcal{T}}_{ijk}] - \frac{2}{n^2 n_3}\boldsymbol{\mathcal{P}}_T\mathbb{E}[\boldsymbol{\mathcal{T}}_{ijk}] + \frac{1}{n^4 n_3^2}\boldsymbol{\mathcal{P}}_T\right\|_{op} \\
=&\left\|\frac{1}{p}\|\boldsymbol{\mathcal{P}}_T(\vec{\boldsymbol{e}}_i * \mathring{\boldsymbol{e}}_k * \vec{\boldsymbol{e}}_j^{\top})\|_F^2 \frac{1}{n^2 n_3}\boldsymbol{\mathcal{P}}_T - \frac{1}{n^4 n_3^2}\boldsymbol{\mathcal{P}}_T\right\|_{op} \\
<&\left(\frac{1}{p}\frac{2\mu_0 r}{n}\frac{1}{n^2 n_3}\right)\|\boldsymbol{\mathcal{P}}_T\|_{op} \\
\leq&\frac{2\mu_0 r}{n^3 n_3 p}
\end{aligned}
$$

Now let

$$\tau = \sqrt{\frac{14\beta\mu_0 r\log(nn_3)}{3np}} \leq \frac{1}{2}$$

with some constant $\beta > 1$. The inequality holds given p satisfying (3.7) with c_0 large enough. Use **Theorem** 7.1.1 we have

$$
\begin{aligned}
&\mathbb{P}\left[\|\boldsymbol{\mathcal{P}}_T\boldsymbol{\mathcal{R}}_\Omega\boldsymbol{\mathcal{P}}_T - \boldsymbol{\mathcal{P}}_T\|_{op} > \tau\right] \\
=&\mathbb{P}\left[\|\overline{\boldsymbol{\mathcal{P}}_T}\overline{\boldsymbol{\mathcal{R}}_\Omega}\overline{\boldsymbol{\mathcal{P}}_T} - \overline{\boldsymbol{\mathcal{P}}_T}\| > \tau\right] \\
=&\mathbb{P}\left[\left\|\sum_{i,j,k}\left(\overline{\boldsymbol{\mathcal{T}}_{ijk}} - \frac{1}{n^2 n_3}\overline{\boldsymbol{\mathcal{P}}_T}\right)\right\| > \tau\right] \\
\leq&2nn_3\exp\left(\frac{-\frac{7}{3}\frac{\beta\mu_0 r\log(nn_3)}{np}}{\frac{2\mu_0 r}{np} + \frac{2\mu_0 r}{np}\frac{1}{6}}\right) \\
\leq&2(nn_3)^{1-\beta}
\end{aligned}
$$

Therefore we can get

$$\begin{aligned}
&\mathbb{P}\left[\|\boldsymbol{\mathcal{P}}_T\boldsymbol{\mathcal{R}}_\Omega\boldsymbol{\mathcal{P}}_T-\boldsymbol{\mathcal{P}}_T\|_{op}\leq\frac{1}{2}\right]\\
\geq&\mathbb{P}\left[\|\boldsymbol{\mathcal{P}}_T\boldsymbol{\mathcal{R}}_\Omega\boldsymbol{\mathcal{P}}_T-\boldsymbol{\mathcal{P}}_T\|_{op}\leq\tau\right]\\
\geq&1-2(nn_3)^{1-\beta}\ ,
\end{aligned}$$

which finishes the proof. □

7.2 Proof of Lemma 3.3.1

Proof. Given any $\boldsymbol{\mathcal{Z}}$ such that $\boldsymbol{\mathcal{P}}_\Omega(\boldsymbol{\mathcal{Z}})=0$ and $\|\boldsymbol{\mathcal{P}}_T\boldsymbol{\mathcal{R}}_\Omega\boldsymbol{\mathcal{P}}_T-\boldsymbol{\mathcal{P}}_T\|_{op}\geq 1/2$, we have

$$\langle\bar{\boldsymbol{\mathcal{Z}}},\overline{\boldsymbol{\mathcal{P}}_T\boldsymbol{\mathcal{R}}_\Omega\boldsymbol{\mathcal{P}}_T(\boldsymbol{\mathcal{Z}})}-\overline{\boldsymbol{\mathcal{P}}_T(\boldsymbol{\mathcal{Z}})}\rangle\geq-\frac{1}{2}\|\bar{\boldsymbol{\mathcal{Z}}}\|_F$$

which gives

$$\langle\boldsymbol{\mathcal{Z}},\boldsymbol{\mathcal{P}}_T\boldsymbol{\mathcal{R}}_\Omega\boldsymbol{\mathcal{P}}_T(\boldsymbol{\mathcal{Z}})-\boldsymbol{\mathcal{P}}_T(\boldsymbol{\mathcal{Z}})\rangle\geq-\frac{1}{2}\|\boldsymbol{\mathcal{Z}}\|_F$$

Note that

$$\begin{aligned}
&\langle\boldsymbol{\mathcal{Z}},\boldsymbol{\mathcal{P}}_T\boldsymbol{\mathcal{R}}_\Omega\boldsymbol{\mathcal{P}}_T(\boldsymbol{\mathcal{Z}})\rangle\\
=&\frac{1}{\sqrt{n_3}}\langle\bar{\boldsymbol{\mathcal{Z}}},\overline{\boldsymbol{\mathcal{P}}_T\boldsymbol{\mathcal{R}}_\Omega\boldsymbol{\mathcal{P}}_T(\boldsymbol{\mathcal{Z}})}\rangle\\
=&\frac{1}{\sqrt{n_3}}\|\overline{\boldsymbol{\mathcal{R}}_\Omega\boldsymbol{\mathcal{P}}_T(\boldsymbol{\mathcal{Z}})}\|_F^2\\
=&\sqrt{n_3}\|\boldsymbol{\mathcal{R}}_\Omega\boldsymbol{\mathcal{P}}_T(\boldsymbol{\mathcal{Z}})\|_F^2\\
=&\sqrt{n_3}\|\boldsymbol{\mathcal{R}}_\Omega(\boldsymbol{\mathcal{Z}}-\boldsymbol{\mathcal{P}}_{T^\perp}(\boldsymbol{\mathcal{Z}}))\|_F^2\\
=&\sqrt{n_3}\|\boldsymbol{\mathcal{R}}_\Omega\boldsymbol{\mathcal{P}}_{T^\perp}(\boldsymbol{\mathcal{Z}})\|_F^2\\
\leq&\frac{\sqrt{n_3}}{p^2}\|\boldsymbol{\mathcal{P}}_{T^\perp}(\boldsymbol{\mathcal{Z}})\|_F^2
\end{aligned}$$

Thus

$$\begin{aligned}
&\|\boldsymbol{\mathcal{P}}_{T^\perp}(\boldsymbol{\mathcal{Z}})\|_F^2 \\
\geq&\frac{p^2}{\sqrt{n_3}}\langle \boldsymbol{\mathcal{Z}}, \boldsymbol{\mathcal{P}}_T\boldsymbol{\mathcal{R}}_\Omega\boldsymbol{\mathcal{P}}_T(\boldsymbol{\mathcal{Z}})\rangle \\
\geq&\frac{p^2}{\sqrt{n_3}}\left(-\frac{1}{2}\|\boldsymbol{\mathcal{Z}}\|_F + \langle \boldsymbol{\mathcal{Z}}, \boldsymbol{\mathcal{P}}_T(\boldsymbol{\mathcal{Z}})\rangle\right) \\
=&\frac{p^2}{\sqrt{n_3}}\left(\frac{1}{\sqrt{n_3}}\langle \overline{\boldsymbol{\mathcal{Z}}}, \overline{\boldsymbol{\mathcal{P}}_T(\boldsymbol{\mathcal{Z}})}\rangle - \frac{1}{2}\|\boldsymbol{\mathcal{Z}}\|_F\right) \\
=&\frac{p^2}{n_3}\|\overline{\boldsymbol{\mathcal{P}}_T(\boldsymbol{\mathcal{Z}})}\|_F^2 - \frac{p^2}{2\sqrt{n_3}}\|\boldsymbol{\mathcal{Z}}\|_F \\
\geq&(p^2 - \frac{p^2}{2\sqrt{n_3}})\|\boldsymbol{\mathcal{P}}_T(\boldsymbol{\mathcal{Z}})\|_F^2 - \frac{p^2}{2\sqrt{n_3}}\|\boldsymbol{\mathcal{P}}_{T^\perp}(\boldsymbol{\mathcal{Z}})\|_F^2
\end{aligned}$$

Then we have

$$\begin{aligned}
&\|\boldsymbol{\mathcal{P}}_{T^\perp}(\boldsymbol{\mathcal{Z}})\|_F^2 \\
\geq&p^2\frac{2\sqrt{n_3}-1}{2\sqrt{n_3}+p^2}\|\boldsymbol{\mathcal{P}}_T(\boldsymbol{\mathcal{Z}})\|_F^2 \\
\geq&\frac{1}{4n^2n_3^3}\|\boldsymbol{\mathcal{P}}_T(\boldsymbol{\mathcal{Z}})\|_F^2
\end{aligned}$$

It follows that

$$\begin{aligned}
&\|\boldsymbol{\mathcal{P}}_{T^\perp}(\boldsymbol{\mathcal{Z}})\|_{TNN} \\
=&\|\overline{\boldsymbol{\mathcal{P}}_{T^\perp}(\boldsymbol{\mathcal{Z}})}\|_* \\
\geq&\|\overline{\boldsymbol{\mathcal{P}}_{T^\perp}(\boldsymbol{\mathcal{Z}})}\|_F \\
\geq&\sqrt{n_3}\|\boldsymbol{\mathcal{P}}_{T^\perp}(\boldsymbol{\mathcal{Z}})\|_F \\
\geq&\frac{1}{2nn_3}\|\boldsymbol{\mathcal{P}}_{T^\perp}(\boldsymbol{\mathcal{Z}})\|_F
\end{aligned}$$

which finishes the proof. □

7.3 Proof of Proposition 3.3.1 Condition 2

Three more lemmas will be introduced in this sections and their proofs are provided in the Appendix D.

We define $\ell_{\infty,2^*}$ norm for tensors to return the largest ℓ_{2^*} norm of the tensor row or tensor column of a third-order tensor.

$$\|\mathcal{Z}\|_{\infty,2*} := \max\left\{\max_i \sqrt{\sum_{b,k} \mathcal{Z}_{ibk}^2}, \max_j \sqrt{\sum_{a,k} \mathcal{Z}_{ajk}^2}\right\}$$

The following lemma states that $\boldsymbol{\mathcal{R}}_\Omega(\mathcal{Z})$ is closed to $\mathcal{Z}$ in tensor spectral norm. The difference is bounded with ℓ_∞ norm and $\ell_{\infty,2^*}$ norm.

Lemma 7.3.1. *If p satisfies the condition in Theorem 3.2.1, and $\mathcal{Z} \in \mathbb{R}^{n\times n\times n_3}$. Then for any constant $c > 0$, we have*

$$\|\boldsymbol{\mathcal{R}}_\Omega(\mathcal{Z}) - \mathcal{Z}\|_{op} \le c\left(\frac{\log(nn_3)}{p}\|\mathcal{Z}\|_\infty + \sqrt{\frac{\log(nn_3)}{p}}\|\mathcal{Z}\|_{\infty,2^*}\right) \tag{7.6}$$

holds with probability at least $1 - (2nn_3)^{-(c-1)}$.

The lemma below bounds the $\ell_{\infty,2^*}$ distance between the terms $\boldsymbol{\mathcal{P}}_T\boldsymbol{\mathcal{R}}_\Omega(\mathcal{Z})$ and $\boldsymbol{\mathcal{P}}_T(\mathcal{Z})$.

Lemma 7.3.2. *If p satisfies the condition in Theorem 3.2.1 for some c_2 sufficiently large, and $\mathcal{Z} \in \mathbb{R}^{n\times n\times n_3}$. Then*

$$\|(\boldsymbol{\mathcal{P}}_T\boldsymbol{\mathcal{R}}_\Omega(\mathcal{Z}) - \boldsymbol{\mathcal{P}}_T(\mathcal{Z})\|_{\infty,2^*} \le \frac{1}{2}\|\mathcal{Z}\|_{\infty,2^*} + \frac{1}{2}\sqrt{\frac{n}{\mu_0 r}}\|\mathcal{Z}\|_\infty$$

with probability at least $1 - (2n^2n_3)^{-(c_2-1)}$.

Lemma 7.3.3. *If p satisfies the condition in Theorem 3.2.1 for some c_3 sufficiently large, and $\mathcal{Z} \in \mathbb{R}^{n\times n\times n_3}$. Then*

$$\|(\mathcal{P}_T\mathcal{R}_\Omega\mathcal{P}_T - \mathcal{P}_T)\mathcal{Z}\|_\infty \leq \frac{1}{2}\|\mathcal{Z}\|_\infty \tag{7.7}$$

with probability at least $1 - 2n^{-(c_3-2)}n_3^{-(c_3-1)}$.

Proof of Proposition 3.3.1 Condition 2

Proof. **(a)** We will first construct a tensor dual certificate $\mathcal{Y}$ and then show it satisfies both conditions here. We will use an approach called Golfing Scheme introduced by Gross [72] and we will follow the idea in [59][65] where the strategy is to construct $\mathcal{Y}$ iteratively. Let Ω be a union of smaller sets Ω_t such that $\Omega = \cup_{t=1}^{t_0}\Omega_t$ where $t_0 = 20\log(nn_3)$. For each t, we assume

$$\mathbb{P}\left[(i,j,k) \in \Omega_t\right] = q := 1 - (1-p)^{1/t}$$

and it is easy to verify that it's equivalent to our original Ω. Define $\mathcal{R}_{\Omega_t}$ similarly to $\mathcal{R}_\Omega$ as follows

$$\mathcal{R}_{\Omega_t}(\mathcal{Z}) = \sum_{i,j,k}\frac{1}{q}\mathbf{1}_{(i,j,k)\in\Omega_t}\mathcal{Z}_{ijk}\vec{\boldsymbol{e}}_i * \mathring{\boldsymbol{e}}_k * \vec{\boldsymbol{e}}_j^\top$$

set $\mathcal{W}_0 = 0$ and for $t = 1, 2, ..., t_0$,

$$\mathcal{W}_t = \mathcal{W}_{t-1} + \mathcal{R}_{\Omega_t}\mathcal{P}_T\left(\mathcal{U} * \mathcal{V}^\top - \mathcal{P}_T(\mathcal{W}_{t-1})\right) \tag{7.8}$$

and tensor $\mathcal{Y} = \mathcal{W}_{t_0}$. By this construction we can see $\mathcal{P}_\Omega(\mathcal{Y}) = \mathcal{Y}$.

For $t = 0, 1, ..., t_0$, set $\mathcal{D}_t = \mathcal{U} * \mathcal{V}^\top - \mathcal{P}_T(\mathcal{W}_t)$. Then we have $\mathcal{D}_0 = \mathcal{U} * \mathcal{V}^\top$ and

$$\mathcal{D}_t = (\mathcal{P}_T - \mathcal{P}_T\mathcal{R}_{\Omega_t}\mathcal{P}_T)(\mathcal{D}_{t-1}) \tag{7.9}$$

Note that Ω_t is independent of $\boldsymbol{\mathcal{D}}_t$, which implies

$$\|\boldsymbol{\mathcal{D}}_t\|_F \leq \|\boldsymbol{\mathcal{P}}_T - \boldsymbol{\mathcal{P}}_T\boldsymbol{\mathcal{R}}_{\Omega_t}\boldsymbol{\mathcal{P}}_T\| \, \|\boldsymbol{\mathcal{D}}_{t-1}\|_F \leq \frac{1}{2}\|\boldsymbol{\mathcal{D}}_{t-1}\|_F$$

since $q \geq p/t_0 \geq c'\mu_0 r \log(nn_3)/n$, we have

$$\begin{aligned}
&\left\|\boldsymbol{\mathcal{P}}_T(\boldsymbol{\mathcal{Y}}) - \boldsymbol{\mathcal{U}} * \boldsymbol{\mathcal{V}}^\top\right\|_F \\
=&\|\boldsymbol{\mathcal{D}}_{t_0}\|_F \\
\leq&\left(\frac{1}{2}\right)^{t_0}\left\|\boldsymbol{\mathcal{U}} * \boldsymbol{\mathcal{V}}^\top\right\|_F \\
\leq&\frac{1}{4(nn_3)^2}\sqrt{r} \\
\leq&\frac{1}{4nn_3^2}
\end{aligned}$$

holds with probability at least $1 - c'(2nn_3)^{-c''}$ by the union bound, for some large enough constants $c', c'' > 0$.

(b) From (7.8) we know that $\boldsymbol{\mathcal{Y}} = \boldsymbol{\mathcal{W}}_{t_0} = \sum_{t=1}^{t_0}\left(\boldsymbol{\mathcal{R}}_{\Omega_t}\boldsymbol{\mathcal{P}}_T\right)\left(\boldsymbol{\mathcal{D}}_{t-1}\right)$, so use **Lemma** 7.3.1 we obtain for some constant $c > 0$,

$$\begin{aligned}
&\|\boldsymbol{\mathcal{P}}_{T^\perp}(\boldsymbol{\mathcal{Y}})\|_{op} \\
\leq&\sum_{t=1}^{t_0}\|\boldsymbol{\mathcal{P}}_{T^\perp}\left(\boldsymbol{\mathcal{R}}_{\Omega_t}\boldsymbol{\mathcal{P}}_T\right)\left(\boldsymbol{\mathcal{D}}_{t-1}\right)\|_{op} \\
\leq&\sum_{t=1}^{t_0}\|\left(\boldsymbol{\mathcal{R}}_{\Omega_t} - \boldsymbol{\mathcal{I}}\right)\boldsymbol{\mathcal{P}}_T(\boldsymbol{\mathcal{D}}_{t-1})\|_{op} \\
\leq&c\sum_{t=1}^{t_0}\left(\frac{\log(nn_3)}{q}\|\boldsymbol{\mathcal{D}}_{t-1}\|_\infty + \sqrt{\frac{\log(nn_3)}{q}}\|\boldsymbol{\mathcal{D}}_{t-1}\|_{\infty,2^*}\right) \\
\leq&\frac{c}{\sqrt{c_0}}\sum_{t=1}^{t_0}\left(\frac{n}{\mu_0 r}\|\boldsymbol{\mathcal{D}}_{t-1}\|_\infty + \sqrt{\frac{n}{\mu_0 r}}\|\boldsymbol{\mathcal{D}}_{t-1}\|_{\infty,2^*}\right)
\end{aligned}$$

where we could bound term $\|\boldsymbol{\mathcal{D}}_{t-1}\|_\infty$ using **Lemma** 7.3.3 as follows,

$$\begin{aligned}
&\|\boldsymbol{\mathcal{D}}_{t-1}\|_\infty \\
=&\|(\boldsymbol{\mathcal{P}}_T - \boldsymbol{\mathcal{P}}_T\boldsymbol{\mathcal{R}}_{\Omega_{t-1}}\boldsymbol{\mathcal{P}}_T)...(\boldsymbol{\mathcal{P}}_T - \boldsymbol{\mathcal{P}}_T\boldsymbol{\mathcal{R}}_{\Omega_1}\boldsymbol{\mathcal{P}}_T)(\boldsymbol{\mathcal{D}}_0)\|_\infty \\
\leq&\left(\frac{1}{2}\right)^{t-1}\|\boldsymbol{\mathcal{U}} * \boldsymbol{\mathcal{V}}^\top\|_\infty
\end{aligned} \tag{7.10}$$

and $\|\boldsymbol{\mathcal{D}}_{t-1}\|_{\infty,2^*}$ is bounded using **Lemma** 7.3.2 and (7.9)(7.10),

$$\begin{aligned}
&\|\boldsymbol{\mathcal{D}}_{t-1}\|_{\infty,2^*} \\
=&\left\|(\boldsymbol{\mathcal{P}}_T - \boldsymbol{\mathcal{P}}_T\boldsymbol{\mathcal{R}}_{\Omega_{k-1}}\boldsymbol{\mathcal{P}}_T)(\boldsymbol{\mathcal{D}}_{t-2})\right\|_{\infty,2^*} \\
\leq&\frac{1}{2}\left\|\boldsymbol{\mathcal{D}}_{t-2}\right\|_{\infty,2^*} + \frac{1}{2}\sqrt{\frac{n}{\mu_0 r}}\left\|\boldsymbol{\mathcal{D}}_{t-2}\right\|_\infty \\
\leq&\frac{1}{2}\left(\frac{1}{2}\left\|\boldsymbol{\mathcal{D}}_{t-3}\right\|_{\infty,2^*} + \frac{1}{2}\sqrt{\frac{n}{\mu_0 r}}\left\|\boldsymbol{\mathcal{D}}_{t-3}\right\|_\infty\right) + \frac{1}{2}\sqrt{\frac{n}{\mu_0 r}}\left\|\boldsymbol{\mathcal{D}}_{t-2}\right\|_\infty \\
\leq&t\left(\frac{1}{2}\right)^{t-1}\sqrt{\frac{n}{\mu_0 r}}\left\|\boldsymbol{\mathcal{U}} * \boldsymbol{\mathcal{V}}^\top\right\|_\infty + \left(\frac{1}{2}\right)^{t-1}\left\|\boldsymbol{\mathcal{U}} * \boldsymbol{\mathcal{V}}^\top\right\|_{\infty,2^*}
\end{aligned}$$

So we get

$$\begin{aligned}
&\|\boldsymbol{\mathcal{P}}_{T^\perp}(\boldsymbol{\mathcal{Y}})\|_{op} \\
\leq&\frac{c}{\sqrt{c_0}}\frac{n}{\mu_0 r}\|\boldsymbol{\mathcal{U}} * \boldsymbol{\mathcal{V}}^\top\|_\infty\sum_{t=1}^{t_0}(t+1)\left(\frac{1}{2}\right)^{t-1} + \frac{c}{\sqrt{c_0}}\sqrt{\frac{n}{\mu_0 r}}\|\boldsymbol{\mathcal{U}} * \boldsymbol{\mathcal{V}}^\top\|_{\infty,2^*}\sum_{t=1}^{t_0}\left(\frac{1}{2}\right)^{t-1} \\
\leq&\frac{6c}{\sqrt{c_0}}\frac{n}{\mu_0 r}\|\boldsymbol{\mathcal{U}} * \boldsymbol{\mathcal{V}}^\top\|_\infty + \frac{2c}{\sqrt{c_0}}\sqrt{\frac{n}{\mu_0 r}}\left\|\boldsymbol{\mathcal{U}} * \boldsymbol{\mathcal{V}}^\top\right\|_{\infty,2^*}
\end{aligned}$$

holds with probability at least $1 - c'(2nn_3)^{-c''}$ by the union bound for some large enough constants $c', c'' > 0$.

Now let's bound $\|\boldsymbol{\mathcal{U}} * \boldsymbol{\mathcal{V}}^\top\|_\infty$. We have

$$\begin{aligned}
&\|\boldsymbol{\mathcal{U}} * \boldsymbol{\mathcal{V}}^\top\|_\infty \\
=&\max_{i,j,k} \left(\boldsymbol{\mathcal{U}}(i,:,:) * \boldsymbol{\mathcal{V}}^\top(:,j,:)\right)_k \\
=&\max_{i,j} \|\boldsymbol{\mathcal{U}}(i,:,:) * \boldsymbol{\mathcal{V}}^\top(:,j,:)\|_\infty
\end{aligned}$$

Note the fact that for two tensor tubes $\mathring{\boldsymbol{x}}, \mathring{\boldsymbol{y}} \in \mathbb{R}^{1\times 1\times n_3}$, use the Cauchy-Schwartz inequality we get

$$\|\mathring{\boldsymbol{x}} * \mathring{\boldsymbol{y}}\|_\infty \leq \|\mathring{\boldsymbol{x}}\|_{2^*} \|\mathring{\boldsymbol{y}}\|_{2^*}$$

Then let $\mathring{\boldsymbol{u}}_t = \boldsymbol{\mathcal{U}}(i,t,:), \mathring{\boldsymbol{v}}_t^\top = \boldsymbol{\mathcal{V}}^\top(t,j,:)$, we can further write $\|\boldsymbol{\mathcal{U}} * \boldsymbol{\mathcal{V}}^\top\|_\infty$ as follows

$$\begin{aligned}
&\|\boldsymbol{\mathcal{U}} * \boldsymbol{\mathcal{V}}^\top\|_\infty \\
=&\max_{i,j} \left\| \sum_{t=1}^{r} \mathring{\boldsymbol{u}}_t * \mathring{\boldsymbol{v}}_t^\top \right\|_\infty \\
\leq&\max_{i,j} \sum_{t=1}^{r} \|\mathring{\boldsymbol{u}}_t * \mathring{\boldsymbol{v}}_t^\top\|_\infty \\
\leq&\max_{i,j} \sum_{t=1}^{r} \|\mathring{\boldsymbol{u}}_t\|_{2^*} \|\mathring{\boldsymbol{v}}_t^\top\|_{2^*} \\
\leq&\max_{i,j} \sum_{t=1}^{r} \frac{1}{2} \left(\|\mathring{\boldsymbol{u}}_t\|_{2^*}^2 + \|\mathring{\boldsymbol{v}}_t^\top\|_{2^*}^2\right) \\
=&\max_{i,j} \left\{ \frac{1}{2} \|\vec{\boldsymbol{e}}_i^\top * \boldsymbol{\mathcal{U}}\|_{2^*}^2 + \frac{1}{2} \|\boldsymbol{\mathcal{V}}^\top * \vec{\boldsymbol{e}}_j\|_{2^*}^2 \right\} \\
\leq&\frac{\mu_0 r}{n}
\end{aligned}$$

by the standard incoherence condition. We also have

$$\begin{aligned}
&\|\boldsymbol{\mathcal{U}} * \boldsymbol{\mathcal{V}}^\top\|_{\infty,2^*} \\
=&\max_{i,j}\left\{\|\boldsymbol{\mathcal{U}} * \boldsymbol{\mathcal{V}}^\top * \vec{\boldsymbol{e}}_i\|_{2^*}, \|\vec{\boldsymbol{e}}_j^\top * \boldsymbol{\mathcal{U}} * \boldsymbol{\mathcal{V}}^\top\|_{2^*}\right\} \\
\leq&\sqrt{\frac{\mu_0 r}{n}}
\end{aligned}$$

and thus

$$\|\boldsymbol{\mathcal{P}}_{T^\perp}(\boldsymbol{\mathcal{Y}})\| \leq \frac{8c}{\sqrt{c_0}} \leq \frac{1}{2}$$

given c_0 large enough.

□

7.4 Proofs of supporting Lemmas

Proof of Lemma C.1

Proof. Let

$$\begin{aligned}
&\boldsymbol{\mathcal{R}}_\Omega(\boldsymbol{\mathcal{Z}}) - \boldsymbol{\mathcal{Z}} \\
=&\sum_{i,j,k} \boldsymbol{\mathcal{C}}_{(ijk)} \\
=&\sum_{i,j,k}\left(\frac{1}{p}\delta_{ijk} - 1\right)\boldsymbol{\mathcal{Z}}_{ijk}\vec{\boldsymbol{e}}_i * \mathring{\boldsymbol{e}}_k * \vec{\boldsymbol{e}}_j^\top
\end{aligned}$$

where $\boldsymbol{\mathcal{C}}_{(ijk)}$ are independent tensors. Then we have

$$\overline{\boldsymbol{\mathcal{C}}_{(ijk)}} = \sum_{i,j,k}\left(\frac{1}{p}\delta_{ijk} - 1\right)\boldsymbol{\mathcal{Z}}_{ijk}\overline{\vec{\boldsymbol{e}}_i}\overline{\mathring{\boldsymbol{e}}_k}\overline{\vec{\boldsymbol{e}}_j}^\top$$

Notice that $\mathbb{E}\left[\overline{\boldsymbol{\mathcal{C}}_{(ijk)}}\right] = 0$ and $\|\overline{\boldsymbol{\mathcal{C}}_{(ijk)}}\| \leq \frac{1}{p}\|\boldsymbol{\mathcal{Z}}\|_\infty$. Moreover,

$$\begin{aligned}
&\left\|\mathbb{E}\left[\sum_{i,j,k} \overline{\boldsymbol{\mathcal{C}}_{(ijk)}}^\top \overline{\boldsymbol{\mathcal{C}}_{(ijk)}}\right]\right\| \\
=&\left\|\mathbb{E}\left[\sum_{i,j,k} \boldsymbol{\mathcal{C}}_{(ijk)}^\top \boldsymbol{\mathcal{C}}_{(ijk)}\right]\right\|_{op} \\
=&\left\|\sum_{i,j,k} \boldsymbol{\mathcal{Z}}_{ijk}^2 \vec{\boldsymbol{e}}_j * \vec{\boldsymbol{e}}_j^\top \mathbb{E}\left(\frac{1}{p}\delta_{ijk} - 1\right)^2\right\|_{op} \\
=&\left\|\frac{1-p}{p}\sum_{i,j,k} \boldsymbol{\mathcal{Z}}_{ijk}^2 \vec{\boldsymbol{e}}_j * \vec{\boldsymbol{e}}_j^\top\right\|_{op}
\end{aligned}$$

since $\vec{\boldsymbol{e}}_j * \vec{\boldsymbol{e}}_j^\top$ will return a zero tensor except for $(j,j,1)$th entry equaling 1, we have

$$\begin{aligned}
&\left\|\mathbb{E}\left[\sum_{i,j,k} \overline{\boldsymbol{\mathcal{C}}_{(ijk)}}^\top \overline{\boldsymbol{\mathcal{C}}_{(ijk)}}\right]\right\| \\
=&\frac{1-p}{p}\max_j \left|\sum_{i,k} \boldsymbol{\mathcal{Z}}_{ijk}\right| \\
\leq&\frac{1}{p}\|\boldsymbol{\mathcal{Z}}\|_{\infty,2^*}^2
\end{aligned}$$

And $\left\|\mathbb{E}\left[\sum_{i,j,k} \overline{\boldsymbol{\mathcal{C}}_{(ijk)}}\overline{\boldsymbol{\mathcal{C}}_{(ijk)}}^\top\right]\right\|$ is bounded similarly. Then use the extension of **Theorem A.1**, for any $c' > 0$ we have

$$\begin{aligned}
&\|\boldsymbol{\mathcal{R}}_\Omega(\boldsymbol{\mathcal{Z}}) - \boldsymbol{\mathcal{Z}}\|_{op} \\
=&\|\overline{\boldsymbol{\mathcal{R}}_\Omega(\boldsymbol{\mathcal{Z}})} - \overline{\boldsymbol{\mathcal{Z}}}\| \\
=&\left\|\sum_{i,j,k} \overline{\boldsymbol{\mathcal{C}}_{(ijk)}}\right\| \\
\leq&\sqrt{\frac{4c'}{p}\|\boldsymbol{\mathcal{Z}}\|_{\infty,2^*}^2 \log(2nn_3)} + \frac{c'}{p}\|\boldsymbol{\mathcal{Z}}\|_\infty \log(2nn_3) \\
\leq&c\left(\frac{\log(nn_3)}{p}\|\boldsymbol{\mathcal{Z}}\|_\infty + \sqrt{\frac{\log(nn_3)}{p}}\|\boldsymbol{\mathcal{Z}}\|_{\infty,2^*}\right)
\end{aligned}$$

holds with probability at lease $1 - (2nn_3)^{-(c-1)}$ for any $c \geq \max\{c', 2\sqrt{c'}\}$. □

Proof of Lemma C.2

Proof. Consider any bth tensor column of $\boldsymbol{\mathcal{P}}_T\boldsymbol{\mathcal{R}}_\Omega(\boldsymbol{\mathcal{Z}}) - \boldsymbol{\mathcal{P}}_T(\boldsymbol{\mathcal{Z}})$:

$$\begin{aligned}
&(\boldsymbol{\mathcal{P}}_T\boldsymbol{\mathcal{R}}_\Omega(\boldsymbol{\mathcal{Z}}) - \boldsymbol{\mathcal{P}}_T(\boldsymbol{\mathcal{Z}})) * \vec{\boldsymbol{e}}_b \\
=&\sum_{i,j,k}(\frac{1}{p}\delta_{ijk} - 1)\boldsymbol{\mathcal{Z}}_{ijk}\boldsymbol{\mathcal{P}}_T(\vec{\boldsymbol{e}}_i * \mathring{\boldsymbol{e}}_k * \vec{\boldsymbol{e}}_j^\top) * \vec{\boldsymbol{e}}_b = \sum_{i,j,k}\vec{\boldsymbol{a}}_{ijk}
\end{aligned}$$

where $\vec{\boldsymbol{a}}_{ijk} \in \mathbb{R}^{n\times 1\times n_3}$ are zero-mean independent tensor columns. Let $\tilde{a}_{ijk} \in \mathbb{R}^{nn_3\times 1}$ be the vectorized column vector of $\vec{\boldsymbol{a}}_{ijk}$. Then the ℓ_2 norm of the vector $\tilde{a}_{ijk}$ is bounded by the following

$$\begin{aligned}
&\|\tilde{a}_{ijk}\| \\
=&\|\vec{\boldsymbol{a}}_{ijk}\|_{2^*} \\
\le&\frac{1-p}{p}\boldsymbol{\mathcal{Z}}_{ijk}\left\|\boldsymbol{\mathcal{P}}_T(\vec{\boldsymbol{e}}_i * \mathring{\boldsymbol{e}}_k * \vec{\boldsymbol{e}}_j^\top) * \vec{\boldsymbol{e}}_b\right\|_{2^*} \\
\le&\frac{1}{p}\sqrt{\frac{2\mu_0 r}{n}}\|\boldsymbol{\mathcal{Z}}\|_\infty \\
\le&\frac{1}{c_0\log(nn_3)}\sqrt{\frac{2n}{\mu_0 r}}\|\boldsymbol{\mathcal{Z}}\|_\infty
\end{aligned}$$

for some constant $c_0 > 0$ given p satisfying (3.7). We also have

$$\begin{aligned}
&\left|\mathbb{E}\left[\sum_{i,j,k}\tilde{a}_{ijk}^\top\tilde{a}_{ijk}\right]\right| \\
=&\mathbb{E}\left[\sum_{i,j,k}\|\vec{\boldsymbol{a}}_{ijk}\|_{2^*}^2\right] \\
=&\frac{1-p}{p}\sum_{i,j,k}\boldsymbol{\mathcal{Z}}_{ijk}^2\left\|\boldsymbol{\mathcal{P}}_T(\vec{\boldsymbol{e}}_i * \mathring{\boldsymbol{e}}_k * \vec{\boldsymbol{e}}_j^\top) * \vec{\boldsymbol{e}}_b\right\|_{2^*}^2
\end{aligned}$$

Use the definition of $\boldsymbol{\mathcal{P}}_T$ and the incoherent condition, we can write

$$\begin{aligned}
&\left\|\boldsymbol{\mathcal{P}}_T(\vec{\boldsymbol{e}}_i * \mathring{\boldsymbol{e}}_k * \vec{\boldsymbol{e}}_j^\top) * \vec{\boldsymbol{e}}_b\right\|_{2^*} \\
=&\left\|(\boldsymbol{\mathcal{U}} * \boldsymbol{\mathcal{U}}^\top * \vec{\boldsymbol{e}}_i * \mathring{\boldsymbol{e}}_k) * \mathring{\boldsymbol{e}}_j^\top * \vec{\boldsymbol{e}}_b + (\boldsymbol{\mathcal{I}} - \boldsymbol{\mathcal{U}} * \boldsymbol{\mathcal{U}}^\top) * \vec{\boldsymbol{e}}_i * \mathring{\boldsymbol{e}}_k * \mathring{\boldsymbol{e}}_j^\top * \boldsymbol{\mathcal{V}} * \boldsymbol{\mathcal{V}}^\top * \vec{\boldsymbol{e}}_b\right\|_{2^*} \\
\leq&\sqrt{\frac{\mu_0 r}{n}}\left\|\vec{\boldsymbol{e}}_j^\top * \vec{\boldsymbol{e}}_b\right\|_{2^*} + \left\|(\boldsymbol{\mathcal{I}} - \boldsymbol{\mathcal{U}} * \boldsymbol{\mathcal{U}}^\top) * \vec{\boldsymbol{e}}_i * \vec{\boldsymbol{e}}_k\right\|\left\|\vec{\boldsymbol{e}}_j^\top * \boldsymbol{\mathcal{V}} * \boldsymbol{\mathcal{V}}^\top * \vec{\boldsymbol{e}}_b\right\|_{2^*} \\
\leq&\sqrt{\frac{\mu_0 r}{n}}\left\|\vec{\boldsymbol{e}}_j^\top * \vec{\boldsymbol{e}}_b\right\|_{2^*} + \left\|\vec{\boldsymbol{e}}_j^\top * \boldsymbol{\mathcal{V}} * \boldsymbol{\mathcal{V}}^\top * \vec{\boldsymbol{e}}_b\right\|_{2^*}
\end{aligned}$$

where $\boldsymbol{\mathcal{I}}$ is the identity tensor. Thus,

$$\begin{aligned}
&\left|\mathbb{E}\left[\sum_{i,j,k} \tilde{a}_{ijk}^\top \tilde{a}_{ijk}\right]\right| \\
\leq&\frac{2}{p}\sum_{ijk} \boldsymbol{\mathcal{Z}}_{ijk}^2 \frac{\mu_0 r}{n}\left\|\vec{\boldsymbol{e}}_j^\top * \vec{\boldsymbol{e}}_b\right\|_{2^*}^2 + \frac{2}{p}\sum_{ijk} \boldsymbol{\mathcal{Z}}_{ijk}^2 \left\|\vec{\boldsymbol{e}}_j^\top * \boldsymbol{\mathcal{V}} * \boldsymbol{\mathcal{V}}^\top * \vec{\boldsymbol{e}}_b\right\|_{2^*}^2 \\
=&\frac{2\mu_0 r}{pn}\sum_{i,k} \boldsymbol{\mathcal{Z}}_{ibk}^2 + \frac{2}{p}\sum_j \left\|\vec{\boldsymbol{e}}_j^\top * \boldsymbol{\mathcal{V}} * \boldsymbol{\mathcal{V}}^\top * \vec{\boldsymbol{e}}_b\right\|_{2^*}^2 \sum_{i,k} \boldsymbol{\mathcal{Z}}_{ijk}^2 \qquad (7.11) \\
\leq&\frac{2\mu_0 r}{pn}\left\|\boldsymbol{\mathcal{Z}}\right\|_{\infty,2^*}^2 + \frac{2}{p}\left\|\boldsymbol{\mathcal{V}} * \boldsymbol{\mathcal{V}}^\top * \vec{\boldsymbol{e}}_b\right\|_{2^*}^2 \left\|\boldsymbol{\mathcal{Z}}\right\|_{\infty,2^*}^2 \\
\leq&\frac{4\mu_0 r}{pn}\|\boldsymbol{\mathcal{Z}}\|_{\infty,2^*}^2 \\
\leq&\frac{4}{c_0 \log(nn_3)} \qquad (7.12)
\end{aligned}$$

where (7.11) is because $\vec{\boldsymbol{e}}_j^\top * \vec{\boldsymbol{e}}_b = 0$ if $j \neq b$. In the same fashion $\left|\mathbb{E}\left[\sum_{i,j,k} \tilde{a}_{ijk}\tilde{a}_{ijk}^\top\right]\right|$ is bounded by the exact same quantity. Since the spectral norm of the vector $\tilde{a}_{ijk}$ is equal to its ℓ_2 norm, then use the extension of **Theorem** 7.1.1 we have for any

$c_1 > 0$, we have

$$\begin{aligned}
&\|(\boldsymbol{\mathcal{P}}_T\boldsymbol{\mathcal{R}}_\Omega(\boldsymbol{\mathcal{Z}}) - \boldsymbol{\mathcal{P}}_T(\boldsymbol{\mathcal{Z}})) * \vec{\boldsymbol{e}}_b\|_{2^*} \\
=&\left\|\sum_{i,j,k} \vec{\boldsymbol{a}}_{ijk}\right\|_{2^*} \\
=&\left\|\sum_{i,j,k} \tilde{a}_{ijk}\right\| \\
\leq&\sqrt{4c_1\sigma^2\log(nn_3)} + c_1 M\log(nn_3) \\
\leq&\frac{1}{2}\|\boldsymbol{\mathcal{Z}}\|_{\infty,2^*} + \frac{1}{2}\sqrt{\frac{n}{\mu_0 r}}\|\boldsymbol{\mathcal{Z}}\|_\infty
\end{aligned}$$

holds with probability at least $1-(nn_3)^{-(c_2-1)}$ for c_2 large enough.

We can also do the same to the tensor rows $\vec{\boldsymbol{e}}_a^\top * (\boldsymbol{\mathcal{P}}_T\boldsymbol{\mathcal{R}}_\Omega(\boldsymbol{\mathcal{Z}}) - \boldsymbol{\mathcal{P}}_T(\boldsymbol{\mathcal{Z}}))$ and get the same bound. Then using a union bound over all the tensor columns and tensor rows, the result holds with probability at least $1 - 2n^2 n_3^{-(c_2-1)}$. With c_2 large enough, the probability goes to zero. Done. □

Proof of Lemma C.3

Proof. Observe that

$$\boldsymbol{\mathcal{P}}_T\boldsymbol{\mathcal{R}}_\Omega\boldsymbol{\mathcal{P}}_T(\boldsymbol{\mathcal{Z}}) = \sum_{i,j,k}\frac{1}{p}\delta_{ijk}\boldsymbol{\mathcal{Z}}_{ijk}\boldsymbol{\mathcal{P}}_T(\vec{\boldsymbol{e}}_i * \mathring{\boldsymbol{e}}_k * \vec{\boldsymbol{e}}_j^\top)$$

so we have that any (a,b,c)th entry of $\boldsymbol{\mathcal{P}}_T\boldsymbol{\mathcal{R}}_\Omega\boldsymbol{\mathcal{P}}_T(\boldsymbol{\mathcal{Z}}) - \boldsymbol{\mathcal{P}}_T(\boldsymbol{\mathcal{Z}})$ is given by

$$\begin{aligned}
&\langle \boldsymbol{\mathcal{P}}_T\boldsymbol{\mathcal{R}}_\Omega\boldsymbol{\mathcal{P}}_T(\boldsymbol{\mathcal{Z}}) - \boldsymbol{\mathcal{P}}_T(\boldsymbol{\mathcal{Z}}), \vec{\boldsymbol{e}}_a * \mathring{\boldsymbol{e}}_c * \vec{\boldsymbol{e}}_b^\top\rangle \\
=&\sum_{i,j,k}\left(\frac{\delta_{ijk}}{p} - 1\right)\boldsymbol{\mathcal{Z}}_{ijk}\langle \boldsymbol{\mathcal{P}}_T(\vec{\boldsymbol{e}}_i * \mathring{\boldsymbol{e}}_k * \vec{\boldsymbol{e}}_j^\top), \vec{\boldsymbol{e}}_a * \mathring{\boldsymbol{e}}_c * \vec{\boldsymbol{e}}_b^\top\rangle \\
:=&\sum_{i,j,k}\boldsymbol{\mathcal{H}}_{ijk,abc}
\end{aligned}$$

It is easy to observe that

$$\begin{aligned}
&|\mathcal{H}_{ijk,abc}| \\
\leq&\frac{1}{p}\|\mathcal{Z}\|_\infty\|\mathcal{P}_T(\vec{\boldsymbol{e}}_i * \mathring{\boldsymbol{e}}_k * \vec{\boldsymbol{e}}_j^\top)\|_F\|\mathcal{P}_T(\vec{\boldsymbol{e}}_a * \mathring{\boldsymbol{e}}_c * \vec{\boldsymbol{e}}_b^\top)\|_F \\
\leq&\frac{2\mu_0 r}{np}\|\mathcal{Z}\|_\infty
\end{aligned}$$

We also have

$$\begin{aligned}
&\left|\mathbb{E}\left[\sum_{i,j,k}\mathcal{H}_{ijk,abc}^2\right]\right| \\
=&\frac{1-p}{p}\|\mathcal{Z}\|_\infty^2\sum_{i,j,k}\left|\langle\mathcal{P}_T(\vec{\boldsymbol{e}}_i * \mathring{\boldsymbol{e}}_k * \vec{\boldsymbol{e}}_j^\top), \vec{\boldsymbol{e}}_a * \mathring{\boldsymbol{e}}_c * \vec{\boldsymbol{e}}_b^\top\rangle\right|^2 \\
=&\frac{1-p}{p}\|\mathcal{Z}\|_\infty^2\left\|\mathcal{P}_T(\vec{\boldsymbol{e}}_a * \mathring{\boldsymbol{e}}_c * \vec{\boldsymbol{e}}_b^\top)\right\|_F \\
\leq&\frac{2\mu_0 r}{np}\|\mathcal{Z}\|_\infty^2
\end{aligned}$$

Then use **Theorem A.1**, we have

$$\begin{aligned}
&\mathbb{P}\left[(\mathcal{P}_T\mathcal{R}_\Omega\mathcal{P}_T(\mathcal{Z}) - \mathcal{P}_T(\mathcal{Z}))_{abc} \geq \frac{1}{2}\|\mathcal{Z}\|_\infty\right] \\
\leq&2\exp\left(\frac{-\|\mathcal{Z}\|_\infty^2/4}{\frac{2\mu_0 r}{np}\|\mathcal{Z}\|_\infty^2 + \frac{\mu_0 r}{3np}\|\mathcal{Z}\|_\infty^2}\right) \\
\leq&2(nn_3)^{-c_3}
\end{aligned}$$

for some $c_3 = 3c_0/28$ large enough, given p satisfying (24). Then using the union bound on every (a, b, c)th entry we have $\|(\mathcal{P}_T\mathcal{R}_\Omega\mathcal{P}_T - \mathcal{P}_T)(\mathcal{Z})\|_\infty \leq \frac{1}{2}\|\mathcal{Z}\|_\infty$ holds with probability at least $1 - 2n^{-(c_3-2)}n_3^{-(c_3-1)}$.

□

Bibliography

[1] K. Braman, "Third-order tensors as linear operators on a space of matrices," *Linear Algebra and its Applications*, vol. 433, no. 7, pp. 1241 – 1253, 2010.

[2] M. Kilmer, K. Braman, N. Hao, and R. Hoover, "Third-order tensors as operators on matrices: A theoretical and computational framework with applications in imaging," *SIAM Journal on Matrix Analysis and Applications*, vol. 34, no. 1, pp. 148–172, 2013.

[3] M. E. Kilmer and C. D. Martin, "Factorization strategies for third-order tensors," *Linear Algebra and its Applications*, vol. 435, no. 3, pp. 641 – 658, 2011, special Issue: Dedication to Pete Stewart on the occasion of his 70th birthday.

[4] D. F. Gleich, C. Greif, and J. M. Varah, "The power and arnoldi methods in an algebra of circulants," *Numerical Linear Algebra with Applications*, vol. 20, no. 5, pp. 809–831, 2013.

[5] F. L. Hitchcock, "The expression of a tensor or a polyadic as a sum of products," *J. Math. Phys*, vol. 6, no. 1, pp. 164–189, 1927.

[6] Hitchcock, "Multiple invariants and generalized rank of a p-way matrix or tensor," *J. Math. Phys*, vol. 7, no. 1, pp. 39–79, 1927.

[7] L. Tucker, "Implications of factor analysis of three-way matrices for measurement of change," *Problems in measuring change*, pp. 122–137, 1963.

[8] Tucker, "The extension of factor analysis to three-dimensional matrices," *Contributions to mathematical psychology*, pp. 109–127, 1964.

[9] L. R. Tucker, "Some mathematical notes on three-mode factor analysis," *Psychometrika*, pp. 279–311, 1966.

[10] J. D. Caroll and J. J. Chang, "Analysis of individual differences in multidimensional scaling via n-way generalization of Eckart–Young decomposition," *Psychometrika*, vol. 35, pp. 283–319, 1970.

[11] R. A. Harshman, "Foundations of the PARAFAC procedure: models and conditions for an explanatory multimodal factor analysis," *UCLA Working Papers in Phonetics*, vol. 16, pp. 1–84, 1970.

[12] B. Chen, A. Petropulu, and L. de Lathauwer, "Blind identification of convolutive MIMO systems with 3 sources and 2 sensors," *EURASIP Journal on Applied Signal Processing*, vol. 5, pp. 487–496, 2002.

[13] P. Comon, "Tensor decomposition: state of the art and applications," in *IMA Conf. Math. in Sig. Proc., Warwick, UK*, 2000.

[14] L. de Lathauwer, J. Castaing, and J. Cardoso, "Fourth-order cumulant-based blind identification of underdetermined mixtures," *IEEE Trans. on Signal Processing*, vol. 55, no. 6, pp. 2965–2973, 2007.

[15] L. de Lathauwer, B. de Moor, and J. Vandewalle, "A multilinear singular value decomposition," *SIAM J. Matrix Anal. Appl.*, vol. 21, pp. 1253–1278, 2000.

[16] T. G. Kolda, "Orthogonal tensor decompositions," *SIAM J. Matrix Anal. Appl.*, vol. 23, no. 1, pp. 243–255, 2001.

[17] T. Zhang and G. H. Golub, "Rank-one approximation to high-order tensors," *SIAM J. Matrix Anal. Appl.*, vol. 23, pp. 534–550, 2001.

[18] E. Acar, S. Camtepe, M. Krishnamoorthy, and B. Yener, "Modeling and multiway analysis of chatroom tensors," *ISI*, vol. 3495, pp. 256–268, 2005.

[19] J. Sun, S. Papadimitriou, and P. Yu, "Window-based tensor analysis on high-dimensional and multi-aspect streams," *Proc. ICDM2006*, 2006.

[20] J. Sun, S. Papadimitriou, and P. S. Yu, "Window-based tensor analysis on high-dimensional and multi-aspect streams," in *Sixth International Conference on Data Mining (ICDM'06)*, Dec 2006, pp. 1076–1080.

[21] M. Vasilescu and D. Terzopoulos, "Multilinear analysis of image ensembles: Tensorfaces," *Lecture Notes in Computer Science*, pp. 447–460, 2002.

[22] 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 2014 IEEE Conference on Computer Vision and Pattern Recognition*, ser. CVPR '14, 2014, pp. 3842–3849.

[23] M. A. O. Vasilescu and D. Terzopoulos, "Multilinear subspace analysis of image ensembles," in *2003 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2003. Proceedings.*, vol. 2, June 2003, pp. II–93–9 vol.2.

[24] D. Vlasic, M. Brand, H. Pfister, and J. Popović, "Face transfer with multilinear models," *ACM Trans. Graph.*, vol. 24, no. 3, pp. 426–433, Jul. 2005. [Online]. Available: http://doi.acm.org/10.1145/1073204.1073209

[25] O. Semerci, N. Hao, M. E. Kilmer, and E. L. Miller, "Tensor-based formulation and nuclear norm regularization for multi-energy computed tomography," *CoRR*, vol. abs/1307.5348, 2013.

[26] N. Hao, M. E. Kilmer, K. Braman, and R. C. Hoover, "Facial recognition using tensor-tensor decompositions," *SIAM Journal on Imaging Sciences*, vol. 6, no. 1, pp. 437–463, 2013. [Online]. Available: http://dx.doi.org/10.1137/110842570

[27] L. Grasedyck, "Hierarchical singular value decomposition of tensors," *SIAM Journal on Matrix Analysis and Applications*, vol. 31, no. 4, pp. 2029–2054, 2010.

[28] I. V. Oseledets, "Tensor-train decomposition," *SIAM Journal on Scientific Computing*, vol. 33, no. 5, pp. 2295–2317, 2011.

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

[30] J. M. F. ten Berge, "Kruskal's polynomial for 2x2x2 arrays and a generalization to 2xnxn arrays," *Psychometrika*, vol. 56, no. 4, pp. 631–636, 1991. [Online]. Available: http://dx.doi.org/10.1007/BF02294495

[31] J. Hastad, "Tensor rank is NP–complete," *J. of Algorithms*, vol. 11, no. 4, pp. 644–654, 1990.

[32] T. G. Kolda, "Multilinear operators for higher-order decompositions," Tech. Rep., 2006.

[33] C. J. Appellof and E. R. Davidson, "Strategies for analyzing data from video fluorometric monitoring of liquid chromatographic effluents," *Analytical Chemistry*, vol. 53, no. 13, pp. 2053–2056, 1981.

[34] F. Miwakeichi, E. Martinez-Montes, P. Valdes-Sosa, N. Nishiyama, H. Mizuhara, and Y. Yamaguchi, "Decomposing EEG data into space–time–frequency components using parallel factor analysis," *NeuroImage*, vol. 22, no. 3, pp. 1035–1045, 2004.

[35] N. Sidiropoulos, R. Bro, and G. Giannakis, "Parallel factor analysis in sensor array processing," *IEEE transactions on Signal Processing*, vol. 48, no. 8, pp. 2377–2388, 2000.

[36] E. Acar, S. A. Çamtepe, and B. Yener, *Collective Sampling and Analysis of High Order Tensors for Chatroom Communications*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2006, pp. 213–224. [Online]. Available: http://dx.doi.org/10.1007/11760146_19

[37] A. Shashua and A. Levin, "Linear image coding for regression and classification using the tensor-rank principle," in *Proceedings of the 2001 IEEE Computer Society Conference on Computer Vision and Pattern Recognition. CVPR 2001*, vol. 1, 2001, pp. I–42–I–49 vol.1.

[38] R. Furukawa, H. Kawasaki, K. Ikeuchi, and M. Sakauchi, "Appearance based object modeling using texture database: Acquisition, compression and rendering," in *Eurographics Workshop on Rendering*, P. Debevec and S. Gibson, Eds. The Eurographics Association, 2002.

[39] L. Karlsson, D. Kressner, and A. Uschmajew, "Parallel algorithms for tensor completion in the CP format," Sep. 2014.

[40] P. Jain and S. Oh, "Provable tensor factorization with missing data," in *Advances in Neural Information Processing Systems 27*, Z. Ghahramani,

M. Welling, C. Cortes, N. Lawrence, and K. Weinberger, Eds. Curran Associates, Inc., 2014, pp. 1431–1439.

[41] G. Duan, H. Wang, Z. Liu, J. Deng, and Y. Chen, "K-CPD: learning of overcomplete dictionaries for tensor sparse coding," in *Proceedings of the 21st International Conference on Pattern Recognition, ICPR 2012, Tsukuba, Japan, November 11-15, 2012*, 2012, pp. 493–496. [Online]. Available: http://ieeexplore.ieee.org/xpl/freeabs_all.jsp?arnumber=6460179

[42] L. De Lathauwer and J. Vandewalle, "Dimensionality reduction in higher-order signal processing and rank-$(R_1, R_2, ..., R_N)$ reduction in multilinear algebra," *Linear Algebra Appl.*, vol. 391, pp. 31–55, 2004.

[43] D. Muti and S. Bourennane, "Multidimensional filtering based on a tensor approach," *Signal Processing*, vol. 85, no. 12, pp. 2338–2353, 2005.

[44] M. A. O. Vasilescu and D. Terzopoulos, "Multilinear image analysis for facial recognition," in *Object recognition supported by user interaction for service robots*, vol. 2, 2002, pp. 511–514 vol.2.

[45] M. A. O. Vasilescu, "Human motion signatures: analysis, synthesis, recognition," in *Object recognition supported by user interaction for service robots*, vol. 3, 2002, pp. 456–460 vol.3.

[46] H. Wang and N. Ahuja, "Facial expression decomposition," in *Proceedings Ninth IEEE International Conference on Computer Vision*, Oct 2003, pp. 958–965 vol.2.

[47] H. Wang and N.Ahuja, "Compact representation of multidimensional data using tensor rank-one decomposition," in *Proceedings of the 17th International Conference on Pattern Recognition, 2004. ICPR 2004.*, vol. 1, Aug 2004, pp. 44–47 Vol.1.

[48] E. Kernfeld, S. Aeron, and M. E. Kilmer, "Clustering multi-way data: a novel algebraic approach," *CoRR*, vol. abs/1412.7056, 2014.

[49] E. Kernfeld, M. Kilmer, and S. Aeron, "Tensor-tensor products with invertible linear transforms," *Linear Algebra and its Applications*, vol. 485, pp. 545–570, 2015.

[50] N. Hao, "Moving from matrix to tensor-based analysis and algorithms for applications in imaging science and beyond," Ph.D. dissertation, Medford, MA, USA, 2014.

[51] W. Hackbusch, *Tensor Spaces and Numerical Tensor Calculus*, ser. Springer Series in Computational Mathematics. Springer, 2012, vol. 42.

[52] N. Boumal and P.-A. Absil, "RTRMC: A Riemannian trust-region method for low-rank matrix completion," in *Advances in Neural Information Processing Systems 24 (NIPS)*, 2011, pp. 406–414.

[53] N. Linial, E. London, and Y. Rabinovich, "The geometry of graphs and some of its algorithmic applications," *Combinatorica*, vol. 15, no. 2, pp. 215–245, 1995.

[54] G. Obozinski, B. Taskar, and M. I. Jordan, "Joint covariate selection and joint subspace selection for multiple classification problems," *Statistics and Computing*, vol. 20, no. 2, pp. 231–252, 2010.

[55] T. Kolda and J. Sun, "Scalable tensor decompositions for multi-aspect data mining," in *Data Mining, 2008. ICDM '08. Eighth IEEE International Conference on*, Dec 2008, pp. 363–372.

[56] J. Sun, S. Papadimitriou, C. Lin, N. Cao, S. Liu, and W. Qian, "Multivis: Content-based social network exploration through multi-way visual analysis," in *Proceedings of the SIAM International Conference on Data Mining, SDM 2009, April 30 - May 2, 2009, Sparks, Nevada, USA*, 2009, pp. 1064–1075.

[57] J. Liu, P. Musialski, P. Wonka, and J. Ye, "Tensor completion for estimating missing values in visual data," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 35, no. 1, pp. 208–220, Jan 2013.

[58] E. J. Candès and B. Recht, "Exact matrix completion via convex optimization," *Commun. ACM*, vol. 55, no. 6, pp. 111–119, 2012.

[59] B. Recht, "A simpler approach to matrix completion," *Journal of Machine Learning Research*, vol. 12, pp. 3413–3430, 2011.

[60] V. D. Silva and L.-H. Kim, "Tensor rank and the ill-posedness of the best low-rank approximation problem," *SIAM J. Matrix Anal. Appl.*, vol. 30, no. 3, 2008.

[61] L. Grasedyck, "Hierarchical singular value decomposition of tensors," *SIAM J. Matrix Anal. Appl.*, vol. 31, no. 4, pp. 2029–2054, May 2010.

[62] B. Huang, C. Mu, D. Goldfarb, and J. Wright, "Provable low-rank tensor recovery," *Optimization-Online*, p. 4252, 2014.

[63] S. Gandy, B. Recht, and I. Yamada, "Tensor completion and low-n-rank tensor recovery via convex optimization," *Inverse Problems*, vol. 27, no. 2, p. 025010, 2011.

[64] C. Navasca, M. Opperman, T. Penderghest, and C. Tamon, "Tensors as module homomorphisms over group rings," *ArXiv e-prints*, May 2010.

[65] Y. Chen, "Incoherence-optimal matrix completion," *CoRR*, vol. abs/1310.0154, 2013.

[66] C. Mu, B. Huang, J. Wright, and D. Goldfarb, "Square deal: Lower bounds and improved relaxations for tensor recovery," in *Proceedings of the 31th International Conference on Machine Learning, ICML 2014, Beijing, China, 21-26 June 2014*, 2014, pp. 73–81.

[67] A. Krishnamurthy and A. Singh, "Low-rank matrix and tensor completion via adaptive sampling." in *NIPS*, C. J. C. Burges, L. Bottou, Z. Ghahramani, and K. Q. Weinberger, Eds., 2013, pp. 836–844.

[68] J. Liu, P. Musialski, P. Wonka, and J. Ye, "Tensor completion for estimating missing values in visual data," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 35, no. 1, pp. 208–220, 2013.

[69] B. Romera-Paredes and M. Pontil, "A new convex relaxation for tensor completion," in *Advances in Neural Information Processing Systems 26: 27th Annual Conference on Neural Information Processing Systems 2013. Proceedings of a meeting held December 5-8, 2013, Lake Tahoe, Nevada, United States.*, 2013, pp. 2967–2975.

[70] E. J. Candès and T. Tao, "The power of convex relaxation: near-optimal matrix completion," *IEEE Transactions on Information Theory*, vol. 56, no. 5, pp. 2053–2080, 2010.

[71] D. Gross and V. Nesme, "Note on sampling without replacing from a finite collection of matrices," *CoRR*, vol. abs/1001.2738, 2010.

[72] D. Gross, "Recovering low-rank matrices from few coefficients in any basis," *IEEE Transactions on Information Theory*, vol. 57, no. 3, pp. 1548–1566, 2011.

[73] Xiao-Yang Liu and Vaneet Aggarwal and Xiaodong Wang and Shuchin Aeron and Min-You Wu, "Adaptive Sampling of RF Fingerprints for Fine-Grained Indoor Localization," December 2015, Accepted to IEEE Transactions on Mobile Computing.

[74] G. Watson, "Characterization of the subdifferential of some matrix norms," *Linear Algebra and its Applications*, vol. 170, pp. 33 – 45, 1992.

[75] E. J. Candes, J. Romberg, and T. Tao, "Robust uncertainty principles: exact signal reconstruction from highly incomplete frequency information," *IEEE Transactions on Information Theory*, vol. 52, no. 2, pp. 489–509, Feb 2006.

[76] J. J. Fuchs, "On sparse representations in arbitrary redundant bases," *IEEE Trans. Inf. Theor.*, vol. 50, no. 6, pp. 1341–1344, Jun. 2004. [Online]. Available: http://dx.doi.org/10.1109/TIT.2004.828141

[77] S. Boyd, N. Parikh, E. Chu, B. Peleato, and J. Eckstein, "Distributed Optimization and Statistical Learning via the Alternating Direction Method of Multipliers," *Foundations and Trends® in Machine Learning*, vol. 3, no. 1, pp. 1–122, 2011.

[78] J. F. Cai, E. J. Candès, and Z. Shen, "A singular value thresholding algorithm for matrix completion," *SIAM Journal on Optimization*, vol. 20, no. 4, pp. 1956–1982, 2010.

[79] A. S. Lewis, "The convex analysis of unitarily invariant matrix functions," *Journal of Convex Analysis Volume 2 (1995)*, vol. 2, no. 1/2, pp. 173–183, 1995.

[80] J. Liu, P. Musialski, P. Wonka, and J. Ye, "Tensor completion for estimating missing values in visual data," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 35, no. 1, pp. 208–220, 2013.

[81] E. J. Candès, X. Li, Y. Ma, and J. Wright, "Robust principal component analysis?" *J. ACM*, vol. 58, no. 3, pp. 11:1–11:37, Jun. 2011. [Online]. Available: http://doi.acm.org/10.1145/1970392.1970395

[82] Z. Zhang and S. Aeron, "Exact tensor completion using t-svd," *CoRR*, vol. abs/1502.04689, 2015. [Online]. Available: http://arxiv.org/abs/1502.04689

[83] E. T. Hale, W. Yin, and Y. Zhang, "Fixed-point continuation for l_1-minimization: Methodology and convergence," *SIAM Journal on Optimization*, vol. 19, no. 3, pp. 1107–1130, 2008. [Online]. Available: http://dx.doi.org/10.1137/070698920

[84] J. Feng, H. Xu, and S. Yan, "Online robust PCA via stochastic optimization," in *Advances in Neural Information Processing Systems 26*. Curran Associates, Inc., 2013, pp. 404–412. [Online]. Available: http://papers.nips.cc/paper/5131-online-robust-pca-via-stochastic-optimization.pdf

[85] J. He, L. Balzano, and A. Szlam, "Incremental gradient on the grassmannian for online foreground and background separation in subsampled video," in *IEEE*

Conference on Computer Vision and Pattern Recognition, 2012, pp. 1568–1575. [Online]. Available: http://dx.doi.org/10.1109/CVPR.2012.6247848

[86] M. Aharon, M. Elad, and A. Bruckstein, "K-svd: An algorithm for designing overcomplete dictionaries for sparse representation," *Trans. Sig. Proc.*, vol. 54, no. 11, pp. 4311–4322, Nov. 2006. [Online]. Available: http://dx.doi.org/10.1109/TSP.2006.881199

[87] J. Yang, Z. Wang, Z. Lin, S. Cohen, and T. Huang, "Coupled dictionary training for image super-resolution," *Image Processing, IEEE Transactions on*, vol. 21, no. 8, pp. 3467–3478, Aug 2012.

[88] I. Ramirez, P. Sprechmann, and G. Sapiro, "Classification and clustering via dictionary learning with structured incoherence and shared features," in *Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on*, June 2010, pp. 3501–3508.

[89] K. Engan, S. O. Aase, and J. H. Husy, "Multi-frame compression: theory and design." *Signal Processing*, vol. 80, no. 10, pp. 2121–2140, 2000. [Online]. Available: http://dblp.uni-trier.de/db/journals/sigpro/sigpro80.html#EnganAH00

[90] Q. Zhang and B. Li, "Discriminative k-svd for dictionary learning in face recognition." in *CVPR*. IEEE Computer Society, 2010, pp. 2691–2698. [Online]. Available: http://dblp.uni-trier.de/db/conf/cvpr/cvpr2010.html#ZhangL10

[91] R. Rubinstein, M. Zibulevsky, and M. Elad, "Efficient implementation of the k-svd algorithm using batch orthogonal matching pursuit," 2008.

[92] F. Huang and A. Anandkumar, "Convolutional dictionary learning through tensor factorization," *CoRR*, vol. abs/1506.03509, 2015. [Online]. Available: http://arxiv.org/abs/1506.03509

[93] S. Zubair and W. Wang, "Tensor dictionary learning with sparse tucker decomposition," in *Digital Signal Processing (DSP), 2013 18th International Conference on*, July 2013, pp. 1–6.

[94] Y. Peng, D. Meng, Z. Xu, C. Gao, Y. Yang, and B. Zhang, "Decomposable nonlocal tensor dictionary learning for multispectral image denoising," in *Proceedings of the 2014 IEEE Conference on Computer Vision and Pattern Recognition*, ser. CVPR '14. Washington, DC, USA: IEEE Computer Society, 2014, pp. 2949–2956. [Online]. Available: http://dx.doi.org/10.1109/CVPR.2014.377

[95] Y. Fu, J. Gao, Y. Sun, and X. Hong, "Joint multiple dictionary learning for tensor sparse coding," in *2014 International Joint Conference on Neural Networks, IJCNN 2014, Beijing, China, July 6-11, 2014*, 2014, pp. 2957–2964. [Online]. Available: http://dx.doi.org/10.1109/IJCNN.2014.6889490

[96] S. Soltani, M. E. Kilmer, and P. C. Hansen, "A tensor-based dictionary learning approach to tomographic image reconstruction," *CoRR*, vol. abs/1506.04954,

2015. [Online]. Available: http://arxiv.org/abs/1506.04954

[97] L. R. Tucker, "Some mathematical notes on three-mode factor analysis," *Psychometrika*, vol. 31, pp. 279–311, 1966c.

[98] M. Elad and M. Aharon, "Image denoising via sparse and redundant representations over learned dictionaries," *Trans. Img. Proc.*, vol. 15, no. 12, pp. 3736–3745, Dec. 2006. [Online]. Available: http://dx.doi.org/10.1109/TIP.2006.881969

[99] K. Dabov, A. Foi, V. Katkovnik, and K. Egiazarian, "Image denoising by sparse 3d transform-domain collaborative filtering," *IEEE TRANS. IMAGE PROCESS*, vol. 16, no. 8, p. 2080, 2007.

[100] N. Renard, S. Bourennane, and J. Blanc-Talon, "Denoising and dimensionality reduction using multilinear tools for hyperspectral images," in *IEEE Trans. Geoscience and Remote Sensing*, 2008. [Online]. Available: http://dx.doi.org/10.1109/ICASSP.2008.4517867

[101] X. Liu, S. Bourennane, and C. Fossati, "Denoising of hyperspectral images using the PARAFAC model and statistical performance analysis," *IEEE T. Geoscience and Remote Sensing*, vol. 50, no. 10, pp. 3717–3724, 2012. [Online]. Available: http://dx.doi.org/10.1109/TGRS.2012.2187063

[102] N. Goyette, P.-M. Jodoin, F. Porikli, J. Konrad, and P. Ishwar, "Changedetection.net: A new change detection benchmark dataset." in

CVPR Workshops. IEEE, 2012, pp. 1–8. [Online]. Available: http://dblp.uni-trier.de/db/conf/cvpr/cvprw2012.html#GoyetteJPKI12

[103] R. Ahlswede and A. Winter, "Strong converse for identification via quantum channels," *IEEE Transactions on Information Theory*, vol. 48, no. 3, pp. 569–579, 2002.

[104] J. A. Tropp, "User-friendly tail bounds for sums of random matrices," *Found. Comput. Math.*, vol. 12, no. 4, pp. 389–434, Aug. 2012.