UNIVERSITY OF CALIFORNIA, SAN DIEGO

**Research on Tensor Computation and Its Application on Data Science**

A dissertation submitted in partial satisfaction of the
requirements for the degree
Doctor of Philosophy

in

Mathematics

by

Zequn Zheng

Committee in charge:

Professor Jiawang Nie, Chair
Professor Todd Kemp
Professor Julian McAuley
Professor Rayan Saab
Professor Lawrence Saul

2023

The dissertation of Zequn Zheng is approved, and it is acceptable in quality and form for publication on microfilm and electronically.

University of California San Diego

2023

DEDICATION

Dedicated to my family.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ACKNOWLEDGEMENTS

First and foremost, I would like to thank my parents for their unwavering love, encouragement, and support throughout my academic journey. Their sacrifices and belief in my abilities have been a constant source of motivation for me to pursue my passion for applied mathematics and complete this thesis.

I am deeply grateful to my advisor, Prof. Jiawang Nie, for his invaluable guidance, mentorship, and expertise. His continuous support, patience, and encouragement have been instrumental in shaping my research direction and pushing me to excel in my studies. I am truly fortunate to have had the opportunity to work with him.

I would also like to express my gratitude to my coauthor, Li Wang, for her contributions to this research project. Her insights, collaboration, and dedication have enriched the quality of my work and made this thesis more robust and comprehensive.

Finally, I would like to thank my dear friends for their encouragement, and companionship throughout this journey. Their presence, words of encouragement, and shared experiences have made this challenging endeavor more enjoyable and meaningful.

Chapter 2 in full, has been submitted for publication. The thesis author is the coauthor of the preprint "J. Nie, L. Wang, and Z. Zheng. Low Rank Tensor Decompositions And Approximations, 2022. arXiv: 2208.07477."

Chapter 3 in full, is currently being prepared for submission for publication, which is a joint work with Nie, Jiawang and Yang, Zi.

Chapter 4 in full, is a reprint of the material as it appears in *Pacific Journal of Optimization* 2023 [45]. The dissertation author coauthored this paper with Nie, Jiawang and Wang, Li.

## VITA

| | |
|---|---|
| 2017 | BSc (Hons) in Computing Mathematics, City University of Hong Kong |
| 2017-2023 | Graduate Teaching Assistant, University of California, San Diego |
| 2023 | Ph. D. in Mathematics, University of California San Diego |

## PUBLICATIONS

J. Nie, L. Wang, and Z. Zheng. Higher Order Correlation Analysis for Multi-View Learning, *Pacific Journal of Optimization*, 2023.

J. Nie, L. Wang, and Z. Zheng. Low Rank Tensor Decompositions And Approximations, *Submitted*, 2022.

ABSTRACT OF THE DISSERTATION

**Research on Tensor Computation and Its Application on Data Science**

by

Zequn Zheng
Doctor of Philosophy in Mathematics
University of California San Diego, 2023
Professor Jiawang Nie, Chair

Tensors or multidimensional arrays are higher order generalizations of matrices. They are natural structures for expressing data that have inherent higher order structures. Tensor decompositions and Tensor approximations play an important role in learning those hidden structures. They have many applications in machine learning, statistical learning, data science, signal processing, neuroscience, and more.

Canonical Polyadic Decomposition (CPD) is a tensor decomposition that decomposes a tensor to a minimal number of summation of rank 1 tensors. While for a given tensor, Low-Rank Tensor Approximation (LRTA) aims at finding a new one whose rank is small and that is close to the given one.

We study the generating polynomials for computing tensor decompositions and low-rank approximations for given tensors and propose methods that compute tensor decompositions for generic tensors under certain rank conditions. For low-rank tensor approximation, the proposed method guarantees that the constructed tensor is a good enough low-rank approximation if the tensor to be approximated is close enough to a low-rank one. The proof built on perturbation analysis is presented.

When the rank is higher than the second dimension, we are not able to find the common zeros of generating polynomials directly. In this case, we need to use the quadratic equations that we get from those generating polynomials. We show that under certain conditions, we are able to find the tensor decompositions using standard linear algebra operations (i.e., solving linear systems, singular value decompositions, QR decompositions). Numerical examples and some comparisons are presented to show the performance of our algorithm.

Multi-view learning is frequently used in data science. The pairwise correlation maximization is a classical approach for exploring the consensus of multiple views. Since the pairwise correlation is inherent for two views, the extensions to more views can be diversified and the intrinsic interconnections among views are generally lost. To address this issue, we propose to maximize the high-order tensor correlation. This can be formulated as a low-rank approximation problem with the high-order correlation tensor of multi-view data. We propose to use the generating polynomial method to efficiently solve the high-order correlation maximization problem of tensor canonical correlation analysis for multi-view learning. Numerical results on simulated data and two real multi-view data sets demonstrate that our proposed method not only consistently outperforms existing methods but also is efficient for large scale tensors.

# Chapter 1

# Introduction

## 1.1 Tensors and Tensor decompositions

In this section, we review some basics about tensors, tensor decompositions, generating polynomials, and multi-view learning.

Let $m$ and $n_1, \ldots, n_m$ be positive integers. A tensor of order $m$ and dimension $(n_1, \ldots, n_m)$ is an array $\mathcal{F}$ that is indexed by integer tuples $(i_1, \ldots, i_m)$ with $1 \leq i_j \leq n_j$ $(j = 1, \ldots, m)$, denoted by

$$\mathcal{F} = (\mathcal{F}_{i_1,\ldots,i_m})_{1\leq i_1\leq n_1,\ldots,1\leq i_m\leq n_m}. \tag{1.1.1}$$

The space of all tensors with real (or complex) entries is denoted as $\mathbb{R}^{n_1\times\cdots\times n_m}$ (or $\mathbb{C}^{n_1\times\cdots\times n_m}$). $m$ is the order of the tensor $\mathcal{F}$. The first order tensors ($m = 1$) are vectors and the second order tensors ($m = 2$) are matrices. For $m \geq 3$, they are called $m$-order tensors. Let $\mathbb{F}$ be a field (either the real field $\mathbb{R}$ or the complex field $\mathbb{C}$). For vectors $\mathbf{v}_1 \in \mathbb{F}^{n_1}, \ldots, \mathbf{v}_m \in \mathbb{F}^{n_m}$, their outer product $\mathbf{v}_1 \otimes \ldots \otimes \mathbf{v}_m$ is a tensor in $\mathbb{F}^{n_1\times\cdots\times n_m}$ for all $1 \leq i_j \leq n_j$ $(j = 1, \ldots, m)$

$$(\mathbf{v}_1 \otimes \cdots \otimes \mathbf{v}_m)_{i_1,\ldots,i_m} = (\mathbf{v}_1)_{i_1} \cdots (\mathbf{v}_m)_{i_m}. \tag{1.1.2}$$

An outer product like $\mathbf{v}_1 \otimes \ldots \otimes \mathbf{v}_m$ is called a rank-1 tensor. For an $m$-order tensor $\mathcal{F} \in \mathbb{F}^{n_1\times\cdots\times n_m}$, there exist tuples $(\mathbf{v}^{s,1}, \ldots, \mathbf{v}^{s,m})$ $(s = 1, \ldots, r)$ with each $\mathbf{v}^{s,j} \in \mathbb{F}^{n_j}$, and integer number $r$, such that

$$\mathcal{F} = \sum_{s=1}^{r} \mathbf{v}^{s,1} \otimes \cdots \otimes \mathbf{v}^{s,m}. \tag{1.1.3}$$

The smallest such $r$ is called the rank of $\mathcal{F}$ in the field $\mathbb{F}$, denoted as $\text{rank}_{\mathbb{F}}(\mathcal{F})$. If $\text{rank}_{\mathbb{F}}(\mathcal{F}) = r$, (1.1.3) is called a rank-$r$ decomposition in the field $\mathbb{F}$. In the literature, $\text{rank}_{\mathbb{C}}(\mathcal{F})$ is also called the candecomp-parafac (CP) rank of tensor $\mathcal{F}$ [50] and (1.1.3) is called the CP decomposition of tensor $\mathcal{F}$.

For a given tensor $\mathcal{F}$, the tensor decomposition problem is to recover the rank decomposition (1.1.3) which is similar to matrices decompositions. However, tensor decomposition is not necessarily unique up to scaling and ordering. A generic tensor's rank cannot be determined with dimension information only. This means if we want to find its tensor decomposition we need to find its rank first. There are many works on the uniqueness of tensor decompositions [7, 51, 17, 19, 40].

Frequently used methods for tensor decomposition given rank include Alternating Least Square(ALS) and Nonlinear Least Square(NLS). Those methods are optimization-based and use alternative linear least square. For three way tensors, we also have some algebraic methods like generalized eigenvalue decomposition (GEVD) [33, 36]. But those methods require the rank not greater than at least two dimensions(ie. $n_1, n_2$) condition. There are also 'algebraic' methods available like [18, 20, 56]. Those methods apply linear algebra type operations to construct a new tensor then apply GEVD to decompose the new tensor to find the decomposition of the original one. Hence they obtain a better bound compared with GEVD. They can outperform 'optimization-based' methods for some rank reach lower bound of uniqueness condition cases. For more study on tensor Decomposition of tensors with other properties like Symmetric, Hermitian. We refer to [3, 43, 46, 32] .

Tensor decompositions have been applied to many fields. Including signal processing [39, 4, 8],neuroscience[11], chemistry[42], machine learning [25, 1, 63, 28, 24] and so on.

### 1.1.1 Notation

We reserve $m \geq 3$ as order of tensors, $n_1 \geq \ldots \geq n_m$ as dimensions of tensors. The symbol $\mathbb{N}$ (resp., $\mathbb{R}$, $\mathbb{C}$) denotes the set of nonnegative integers (resp., real, complex numbers). Uppercase letters (i.e., $A$) denote matrices, $(A)_{i,j}$ denotes the $(i,j)$th entry of matrix $A$, and Curl letters (i.e., $\mathcal{F}$) denote tensors. For a complex matrix $A$, $A^T$ denotes its transpose, $A^*$ denotes its conjugate transpose.

$null(A)$ denotes the null space of $A$. $col(A)$ denotes the column space of $A$. Bold lower case letters (i.e., $\mathbf{v}$) denote vectors, $(\mathbf{v})_i$ denotes the $i$th entry of vector $\mathbf{v}$, and $diag(\mathbf{v})$ is the $n \times n$ diagonal matrix whose diagonal entries are the elements of the $n$ dimensional vector $\mathbf{v}$. For a decomposition of tensor $\mathcal{F} = \sum_{s=1}^{r} \mathcal{F}_s = \sum_{s=1}^{r} \mathbf{v}^{s,1} \otimes \cdots \otimes \mathbf{v}^{s,m}$, denote $U^{(i)} = [\mathbf{v}^{1,i}, ..., \mathbf{v}^{r,i}],\ i \in \{1, 2, \cdots, m\}$. We call those $U^{(i)} \in \mathbb{C}^{n_i \times r}$ the decomposition matrices of tensor $\mathcal{F}$ and

$$\mathcal{F} = U^{(1)} \circ ... \circ U^{(m)} = \sum_{i=1}^{r} (U^{(1)})_{:,i} \otimes \ldots \otimes (U^{(m)})_{:,i}, \text{ where } U^{(i)} \in \mathbb{C}^{n_i \times r}.$$

For sets $\mathfrak{A}$ and $\mathfrak{B}$, $\mathfrak{A} \sqcup \mathfrak{B}$ is the union of two sets. For matices $A$ and $B$, denote their columns by $\mathbf{a}_1, \ldots, \mathbf{a}_n$ and $\mathbf{b}_1, \ldots, \mathbf{b}_n$. Khatri-Rao product is denoted by $\odot$,

$$A \odot B := \begin{bmatrix} a_{11}\mathbf{b}_1 & \ldots & a_{1n}\mathbf{b}_n \\ \vdots & \ddots & \vdots \\ a_{n1}\mathbf{b}_1 & \ldots & a_{nn}\mathbf{b}_n \end{bmatrix} \quad \text{where } A = \begin{bmatrix} a_{11} & \ldots & a_{1n} \\ \vdots & \ddots & \vdots \\ a_{n1} & \ldots & a_{nn} \end{bmatrix}.$$

Given matrix $M \in \mathbb{C}^{p \times n_t}$ (vector $\mathbf{v} \in \mathbb{C}^{n_t}$), and tensor $\mathcal{F} \in \mathbb{C}^{n_1 \times n_2 \times \ldots \times n_m}$, we define the tensor matrix (vector) product as

$$\hat{\mathcal{F}} = \mathcal{F} \times_t M \in \mathbb{C}^{n_1 \times \ldots \times n_{t-1} \times p \times n_{t+1} \times \ldots \times n_m},$$

$$\text{or } \hat{\mathcal{F}} = \mathcal{F} \times_t \mathbf{v} \in \mathbb{C}^{n_1 \times \ldots \times n_{t-1} \times 1 \times n_{t+1} \times \ldots \times n_m},$$

where $\hat{\mathcal{F}}_{i_1,\ldots,i_{t-1},:,i_{t+1},\ldots,i_m} = M\mathcal{F}_{i_1,\ldots,i_{t-1},:,i_{t+1},\ldots,i_m}$ or $\mathbf{v}^T\mathcal{F}_{i_1,\ldots,i_{t-1},:,i_{t+1},\ldots,i_m}$.

Kruskal rank or k-rank, is the largest number $r$ such that any subset of columns of A has at most $r$ linear independent vectors. When we say a matrix $A \in \mathbb{C}^{a \times b}$ has full Kruskal rank, we are referring to $krank(A) = min(a, b)$.

### 1.1.2 Flattening matrices

We partition the dimensions $n_1, n_2, ..., n_m$ into two disjoint groups $I_1$ and $I_2$ such that we minimize the difference

$$\Big| \prod_{i \in I_1} n_i - \prod_{j \in I_2} n_j \Big|.$$

Up to a permutation, we write that $I_1 = \{n_1, \ldots, n_k\}$, $I_2 = \{n_{k+1}, \ldots, n_m\}$. For convenience, denote that

$$I = \{(\imath_1, \ldots, \imath_k) : 1 \leqslant \imath_j \leqslant n_j, j = 1, \ldots, k\},$$
$$J = \{(\imath_{k+1}, \ldots, \imath_m) : 1 \leqslant \imath_j \leqslant n_j, j = k+1, \ldots, m\}.$$

For a tensor $\mathcal{F} \in \mathbb{C}^{n_1 \times \cdots \times n_m}$, the above partition gives the flattening matrix

$$\mathrm{Flat}(\mathcal{F}) \coloneqq (\mathcal{F}_{\imath,\jmath})_{\imath \in I, \jmath \in J}\,. \tag{1.1.4}$$

This gives the most square flattening matrix for $\mathcal{F}$. Let $\sigma_r$ denote the closure of all rank-$r$ tensors in $\mathbb{C}^{n_1 \times \cdots \times n_m}$, under the Zariski topology (see [12]). The set $\sigma_r$ is an irreducible variety of $\mathbb{C}^{n_1 \times \cdots \times n_m}$. For a given tensor $\mathcal{F} \in \sigma_r$, it is possible that $\mathrm{rank}(\mathcal{F}) > r$. This fact motivates the notion of border rank:

$$\mathrm{rank}_B(\mathcal{F}) = \min\{r : \mathcal{F} \in \sigma_r\}\,, \tag{1.1.5}$$

For every tensor $\mathcal{F} \in \mathbb{C}^{n_1 \times \cdots \times n_m}$, one can show that

$$\mathrm{rank}\,\mathrm{Flat}(\mathcal{F}) \leqslant \mathrm{rank}_B(\mathcal{F}) \leqslant \mathrm{rank}(\mathcal{F}). \tag{1.1.6}$$

A property P is said to hold *generically* on $\sigma_r$ if P holds on a Zariski open subset $T$ of $\sigma_r$. For such a property P, each $u \in T$ is called a generic point. Interestingly, the above three ranks are equal for generic points of $\sigma_r$ for a range of values of $r$.

**Lemma 1.1.1.** *Let $s$ be the smaller dimension of the matrix Flat($\mathcal{F}$). For every $r \leqslant s$, the equalities*

$$\mathrm{rank}\,\mathit{Flat}(\mathcal{F}) \;=\; \mathrm{rank}_B(\mathcal{F}) \;=\; \mathrm{rank}(\mathcal{F}) \tag{1.1.7}$$

*hold for tensors $\mathcal{F}$ in a Zariski open subset of $\sigma_r$.*

*Proof.* Let $\phi_1, \ldots, \phi_\ell$ be the $r \times r$ minors of the matrix

$$\mathrm{Flat}(\sum_{i=1}^{r} x^{i,1} \otimes \cdots \otimes x^{i,m}). \tag{1.1.8}$$

They are homogeneous polynomials in $x^{i,j} (i = 1, \ldots, r, j = 1, \ldots, m)$. Let $x$ denote the tuple $(x^{1,1}, x^{1,2}, \ldots, x^{r,m})$. Define the projective variety in $\mathbb{P}^{r(n_1+\cdots+n_m)-1}$

$$Z = \{x : \phi_1(x) = \cdots = \phi_\ell(x) = 0\}\,. \tag{1.1.9}$$

Then $Y := \mathbb{P}^{r(n_1+\cdots+n_m)-1} \backslash Z$ is a Zariski open subset of full dimension. Consider the polynomial mapping $\pi : Y \to \sigma_r$,

$$\left(x^{1,1}, x^{1,2}, \ldots, x^{r,m}\right) \mapsto \sum_{i=1}^{r} \left(x^{i,1}\right) \otimes \cdots \otimes \left(x^{i,m}\right). \tag{1.1.10}$$

The image $\pi(Y)$ is dense in the irreducible variety $\sigma_r$. So, $\pi(Y)$ contains a Zariski open subset $\mathscr{Y}$ of $\sigma_r$ (see [53]). For each $\mathcal{F} \in \mathscr{Y}$, there exists $u \in Y$ such that $\mathcal{F} = \pi(u)$. Because $u \notin Z$, at least one of $\phi_1(u), \ldots, \phi_\ell(u)$ is nonzero, and hence $\operatorname{rank}\operatorname{Flat}(\mathcal{F}) \geqslant r$. By (1.1.6), we know (1.1.7) holds for all $\mathcal{F} \in \mathscr{Y}$ since $\operatorname{rank}(\mathcal{F}) \leqslant r$. Since $\mathscr{Y}$ is a Zariski open subset of $\sigma_r$, the lemma holds. □

By Lemma 1.1.1, if $r \leqslant s$ and $\mathcal{F}$ is a generic tensor in $\sigma_r$, we can use $\operatorname{rank}\operatorname{Flat}(\mathcal{F})$ to estimate $\operatorname{rank}(\mathcal{F})$. However, for a generic $\mathcal{F} \in \mathbb{C}^{n_1 \times \cdots \times n_m}$ such that $\operatorname{rank}\operatorname{Flat}(\mathcal{F}) = r$, we cannot conclude $\mathcal{F} \in \sigma_r$.

### 1.1.3 Reshaping of tensor decompositions

A tensor $\mathcal{F}$ of order greater than 3 can be reshaped to another tensor $\widehat{\mathcal{F}}$ of order 3. A tensor decomposition of $\widehat{\mathcal{F}}$ can be converted to a decomposition for $\mathcal{F}$ under certain conditions. In the following, we assume a given tensor $\mathcal{F}$ has the decomposition (1.1.3). Suppose the set $\{1, \ldots, m\}$ is partitioned into 3 disjoint subsets

$$\{1, \ldots, m\} \;=\; I_1 \cup I_2 \cup I_3.$$

Let $p_i = |I_i|$ for $i = 1, 2, 3$. For the reshaped vectors

$$\begin{cases} w^{s,1} &=& u^{s,i_1} \boxtimes \cdots \boxtimes u^{s,i_{p_1}} & \text{for } I_1 \,=\, \{i_1, \ldots, i_{p_1}\}, \\ w^{s,2} &=& u^{s,j_1} \boxtimes \cdots \boxtimes u^{s,j_{p_2}} & \text{for } I_2 \,=\, \{j_1, \ldots, j_{p_2}\}, \\ w^{s,3} &=& u^{s,k_1} \boxtimes \cdots \boxtimes u^{s,k_{p_3}} & \text{for } I_3 \,=\, \{k_1, \ldots, k_{p_3}\}, \end{cases} \tag{1.1.11}$$

we get the following tensor decomposition

$$\widehat{\mathcal{F}} \;=\; \sum_{s=1}^{r} w^{s,1} \otimes w^{s,2} \otimes w^{s,3}. \tag{1.1.12}$$

Conversely, for a decomposition like (1.1.12) for $\widehat{\mathcal{F}}$, if all $w^{s,1}, w^{s,2}, w^{s,3}$ can be expressed as rank-1 products as in (1.1.11), then the equation (1.1.12) can be reshaped to a tensor decomposition for $\mathcal{F}$ as in (1.1.3). When the flattened tensor $\widehat{\mathcal{F}}$ satisfies some conditions, the tensor decomposition of $\widehat{\mathcal{F}}$ is unique. For such a case, we can obtain a tensor decomposition for $\mathcal{F}$ through the decomposition (1.1.12). A classical result about the uniqueness is the Kruskal's criterion [31].

**Theorem 1.1.2.** *(Kruskal's Criterion, [31]) Let $\mathcal{F} = U^{(1)} \circ U^{(2)} \circ U^{(3)}$ be a tensor with each $U^{(i)} \in \mathbb{C}^{n_i \times r}$. Let $\kappa_i$ be the Kruskal rank of $U^{(i)}$, for $i = 1, 2, 3$. If*

$$2r + 2 \leqslant \kappa_1 + \kappa_2 + \kappa_3,$$

*then $\mathcal{F}$ has a unique rank-$r$ tensor decomposition.*

The Kruskal's Criterion can be generalized for more range of $r$ as in [7]. Assume the dimension $n_1 \geqslant n_2 \geqslant n_3 \geqslant 2$ and the rank $r$ is such that

$$2r + 2 \leqslant \min(n_1, r) + \min(n_2, r) + \min(n_3, r),$$

or equivalently, for $\delta = n_2 + n_3 - n_1 - 2$, $r$ is such that

$$r \leqslant n_1 + \min\{\frac{1}{2}\delta, \delta\}.$$

If $\mathcal{F}$ is a generic tensor of rank $r$ as above in the space $\mathbb{C}^{n_1 \times n_2 \times n_3}$, then $\mathcal{F}$ has a unique rank-$r$ decomposition. There following is a uniqueness result for reshaped tensor decompositions.

**Theorem 1.1.3.** *(Reshaped Kruskal Criterion, [7, Theorem 4.6]) For the tensor space $\mathbb{C}^{n_1 \times n_2 \times \cdots \times n_m}$ with $m \geqslant 3$, let $I_1 \cup I_2 \cup I_3 = \{1, 2, ..., m\}$ be a union of disjoint sets and let*

$$p_1 = \prod_{i \in I_1} n_i, \quad p_2 = \prod_{j \in I_2} n_j, \quad p_3 = \prod_{k \in I_3} n_k.$$

*Suppose $p_1 \geqslant p_2 \geqslant p_3$ and let $\delta = p_2 + p_3 - p_1 - 2$. Assume*

$$r \leqslant p_1 + \min\{\frac{1}{2}\delta, \delta\}. \tag{1.1.13}$$

*If $\mathcal{F}$ is a generic tensor of rank $r$ in $\mathbb{C}^{n_1 \times n_2 \times \cdots \times n_m}$, then the reshaped tensor $\widehat{\mathcal{F}} \in \mathbb{C}^{p_1 \times p_2 \times p_3}$ as in (1.1.12) has a unique rank-$r$ decomposition.*

## 1.2 Generating Polynomials

This subsection introduces generating polynomials. For the convenience of discussion, in the following, we assume $n_1 \geq n_2 \geq \ldots \geq n_m$ and consider tensors whose rank $r \leq n_1$. We denote indeterminate variables

$$\mathbf{x_1} = (x_{1,2}, ...x_{1,n_1}), \quad \mathbf{x_2} = (x_{2,2}, ...x_{2,n_2}), \quad ..., \quad \mathbf{x_m} = (x_{m,2}, ...x_{m,n_m}).$$

By setting $x_{j,1} = 1$ for $\forall j \in \{1, 2, ..., m\}$, the $(i_1, i_2, ..., i_m)$ element of $\mathcal{F}$ can be viewed as a monomial $x_{1,i_1}x_{2,i_2}...x_{m,i_m}$. Let

$$\mathbb{M} := \left\{x_{1,i_1}...x_{m,i_m} \mid 1 \leq i_j \leq n_j, 1 \leq j \leq m\right\}, \quad \mathcal{M} = \text{span}\{\mathbb{M}\}. \tag{1.2.1}$$

Let $J$ be a subset of $\{1, 2, ..., m\}$, denote

$$J^c := \{1, 2, ..., m\} \backslash J,$$
$$\mathbb{M}_J := \left\{x_{1,i_1}...x_{m,i_m} \mid i_j = 1, \ \ \forall j \in J^c\right\}, \quad \mathcal{M}_J = \text{span}\{\mathbb{M}_J\}. \tag{1.2.2}$$

Each $\mathcal{F} \in \mathbb{C}^{n_1 \times ... \times n_m}$ is indexed by $(i_1, \ldots, i_m)$ with

$$1 \leq i_1 \leq n_1, \ldots, 1 \leq i_m \leq n_m.$$

Since $(i_1, \ldots, i_m)$ is uniquely dertermined by the multi-linear monomial $x_{1,i_1}, \ldots, x_{m,i_m} \in \mathbb{M}$, we can equivalently index $\mathcal{F}$ as

$$\mathcal{F}_{x_{1,i_1}...x_{m,i_m}} := \mathcal{F}_{i_1,...,i_m}. \tag{1.2.3}$$

The tensor in $\mathbb{C}^{n_1,...,n_m}$ can be equivalently indexed by multi-linear monomials in $\mathbb{M}$. For $\mathcal{F} \in \mathbb{C}^{n_1,...,n_m}$, define the operation on polynomial in $\mathcal{M}$ as

$$\langle \sum_{\mu \in \mathbb{M}} c_\mu \mu, \mathcal{F} \rangle := \sum_{\mu \in \mathbb{M}} c_\mu \mathcal{F}_\mu. \tag{1.2.4}$$

In the above, $c_\mu$ is a complex scalar.

**Definition 1.2.1.** *For some $J \subseteq \{1, 2, ..., m\}$ and $\mathcal{F} \in \mathbb{C}^{n_1 \times n_2 \times ... \times n_m}$, we call $p \in \mathcal{M}_J$ a generating polynomial for $\mathcal{F}$ if*

$$\langle pq, \mathcal{F} \rangle = 0 \quad \forall q \in \mathbb{M}_{J^c}. \tag{1.2.5}$$

**Example 1.2.2.** Consider an order-3 rank-2 tensor $\mathcal{F} \in \mathbb{C}^{3\times3\times3}$, where

$$\mathcal{F} = \left[\mathcal{F}_{:,:,1} \,\middle|\, \mathcal{F}_{:,:,2} \,\middle|\, \mathcal{F}_{:,:,3}\right] = \left[\begin{array}{ccc|ccc|ccc} -10 & 48 & 70 & 22 & -16 & -58 & -1 & 44 & 49 \\ -10 & -64 & -50 & -42 & 0 & 78 & -29 & -68 & -19 \\ -5 & 10 & 20 & 3 & -6 & -12 & -4 & 8 & 16 \end{array}\right]. \tag{1.2.6}$$

It has a generating polynomial $p \in \mathcal{M}_{\{1,2\}}$ defined by

$$p := (3x_{1,1} + x_{1,2})(-2x_{2,1} - x_{2,2}).$$

We can check $p$ is a generating polynomial by using definition 1.2.1. Since $p \in \mathcal{M}_{\{1,2\}}$, for all $q \in \mathcal{M}_{\{3\}}$, we want to check $\langle pq, \mathcal{F} \rangle = 0$. We have

$$pq = (3x_{1,1} + x_{1,2})(2x_{2,1} + x_{2,2})x_{3,i_3} \quad \text{for } 1 \leq i_3 \leq n_3,$$

We have $6\mathcal{F}_{1,1,i_3}+3\mathcal{F}_{1,2,i_3}+2\mathcal{F}_{2,1,i_3}+\mathcal{F}_{2,2,i_3} = 0$ for $i_3 = 1, 2, 3$. Therefore $\langle pq, \mathcal{F} \rangle = 0$ for all $q \in \mathcal{M}_{\{3\}}$, $p$ is a generating polynomial.

## 1.3 Low-rank tensor approximation

Mathematically, LRTA is equivalent to solving a nonlinear least square problem. For a given tensor $\mathcal{F} \in \mathbb{F}^{n_1 \times \cdots \times n_m}$, and a given small integer number $r$, LRTA is to find $r$ tuples $\mathbf{v}^{(s)} := (\mathbf{v}^{s,1}, \ldots, \mathbf{v}^{s,m}) \in \mathbb{F}^{n_1} \times \cdots \times \mathbb{F}^{n_m}$ $(s = 1, \ldots, r)$, which gives a minimizer to the following nonlinear least square problem (the norm below is Frobenius norm for tensors)

$$\min_{\mathbf{v}^{(1)} \in \mathbb{F}^{n_1}, \cdots, \mathbf{v}^{(m)} \in \mathbb{F}^{n_m}} \left\| \mathcal{F} - \sum_{s=1}^{r} \mathbf{v}^{s,1} \otimes \cdots \otimes \mathbf{v}^{s,m} \right\|^2. \tag{1.3.1}$$

Different from the matrix case, the best rank-$r$ tensor approximation may not exist [14]. This is because the set of tensors, whose ranks are less than or equal to $r$, may not be closed. A very popularly used method to find approximations for a given tensor is the alternating least squares (ALS) method [9, 30], which is easy to implement, while its convergence property is generally not very satisfying. When $r = 1$, problem (1.3.1) is called the best rank-1 approximation, several other methods were proposed in the past few years, for example, higher order power iterations [13], semidefinite relaxations [44], SVD-based algorithm [23]. For a generic tensor $\mathcal{F}$, the best rank-1 approximation is unique [21]. For $r > 1$, there exist various works on best rank-$r$ approximations, e.g., [9, 50, 54], most of these methods are designed based on ALS, which are easy to implement, but their performance is generally not satisfying, and convergence properties are in general not guaranteed. We refer to [9, 10, 22] for recent research study on LRTA.

## 1.4 Multiview learning

Multi-view learning is a learning paradigm for multi-view data, which has been widely used for a variety of real applications. Specifically, multi-view data contain sets of samples, each of which is depicted by different characteristics. For example, an image can have different feature descriptors such as color, texture and shape. A webpage can contain text and images, as well as hyperlinks to other web pages. Due to the heterogeneous features extracted from each view, multi-view learning becomes a hot topic in the field of machine learning to reduce the heterogeneous gap among multiple views by maximizing the consensus of multiple views in some latent common space. A variety of multi-view learning methods have been proposed in the literature. Among them, canonical correlation analysis (CCA), originally designed for measuring the linear correlation between two sets of variables [27], has been the workhorse for learning a common latent space between two views [61], and it is extended to various different learning scenarios such as more than two views [41, 47], nonlinear [2] and sparse [26] representations. Its usefulness and those of its variants have been well demonstrated in many scientific domains [57]. Correlation as a measurement is originally defined for two sets of variables. The extension from two sets to more than two sets can be diversified, see [47] for twenty combinations of objectives and constraints.

Correlation as a measurement is originally defined for two sets of variables is often used to impose consensus among multiple views such as canonical correlation analysis (CCA) [27] and its variants [61, 57, 41, 47, 2, 26]. Pairwise correlation is a common criterion to capture the intrinsic interconnections of two views [47], but for more than two views, the intrinsic interconnections among all views are lost. To overcome the above issue, high-order tensor correlation methods are proposed by directly modeling the interconnections via a tensor. [41] introduces tensor CCA (TCCA) by maximizing the high-order tensor correlation, which not only generalizes correlation between two views but also explores the high-order correlation for more than two views. However, the maximization of high-order tensor correlation using ALS in [41] is suboptimal.

# Chapter 2

# Low-Rank Tensor Decompositions and Tensor Approximations

Tensor decompositions are closely related to generating polynomials. We are interested in a set of polynomials whose roots imply a tensor decomposition. Suppose the rank $r \leqslant n_1$ is given. For the convenience of notation, denote the label set

$$J \coloneqq \{(i,j,k) : 1 \leqslant i \leqslant r,\ 2 \leqslant j \leqslant m,\ 2 \leqslant k \leqslant n_j\}. \tag{2.0.1}$$

For a matrix $G \in \mathbb{C}^{[r]\times J}$ and a triple $\tau = (i,j,k) \in J$, define the bi-linear polynomial

$$\phi[G,\tau](x) \coloneqq \sum_{\ell=1}^{r} G(\ell,\tau)x_{1,\ell}x_{j,1} - x_{1,i}x_{j,k} \ \in \mathcal{M}_{\{1,j\}}. \tag{2.0.2}$$

The rows of $G$ are labelled by $\ell = 1,2,...,r$ and the columns of $G$ are labelled by $\tau \in J$. We are interested in $G$ such that $\phi[G,\tau]$ is a generating polynomial for a tensor $\mathcal{F} \in \mathbb{C}^{n_1\times n_2\times\ldots\times n_m}$. This requires that

$$\langle \phi[G,\tau]\cdot\mu, \mathcal{F}\rangle = 0 \quad \text{for all } \mu \in \mathbb{M}_{\{1,j\}^c}.$$

The above is equivalent to the equation ($\mathcal{F}$ is labelled as in (1.2.3))

$$\sum_{\ell=1}^{r} G(\ell,\tau)\mathcal{F}_{x_{1,\ell}\cdot\mu} = \mathcal{F}_{x_{1,i}x_{j,k}\cdot\mu}. \tag{2.0.3}$$

**Definition 2.0.1.** *When* (2.0.3) *holds for all $\tau \in J$, the $G$ is called a generating matrix for $\mathcal{F}$.*

For given $G$, $j \in \{2, \ldots, m\}$ and $k \in \{2, \ldots, n_j\}$, we denote the matrix

$$M^{j,k}[G] \coloneqq \begin{bmatrix} G(1,(1,j,k)) & G(2,(1,j,k)) & \ldots & G(r,(1,j,k)) \\ G(1,(2,j,k)) & G(2,(2,j,k)) & \ldots & G(r,(2,j,k)) \\ \vdots & \vdots & \ddots & \vdots \\ G(1,(r,j,k)) & G(2,(r,j,k)) & \ldots & G(r,(r,j,k)) \end{bmatrix}. \tag{2.0.4}$$

For each $(j,k)$, define the matrix/vector

$$\begin{cases} A[\mathcal{F},j] & \coloneqq & \left(\mathcal{F}_{x_{1,\ell}\cdot\mu}\right)_{\mu\in\mathbb{M}_{\{1,j\}^c},1\leqslant\ell\leqslant r}, \\ b[\mathcal{F},j,k] & \coloneqq & \left(\mathcal{F}_{x_{1,\ell}\cdot x_{j,k}\cdot\mu}\right)_{\mu\in\mathbb{M}_{\{1,j\}^c},1\leqslant\ell\leqslant r}. \end{cases} \tag{2.0.5}$$

The equation (2.0.3) is then equivalent to

$$A[\mathcal{F},j](M^{j,k}[G])^T \;=\; b[\mathcal{F},j,k]. \tag{2.0.6}$$

The following is a useful property for the matrices $M^{j,k}[G]$.

**Theorem 2.0.2.** *Suppose $\mathcal{F} = \sum_{s=1}^{r} u^{s,1} \otimes \ldots \otimes u^{s,m}$ for vectors $u^{s,j} \in \mathbb{C}^{n_j}$. If $r \leqslant n_1$, $(u^{s,2})_1 \ldots (u^{s,m})_1 \neq 0$, and the first $r$ rows of the first decomposing matrix*

$$U^{(1)} \coloneqq [u^{1,1} \;\cdots\; u^{r,1}]$$

*are linearly independent, then there exists a $G$ satisfying* (2.0.6) *and satisfying (for all $j \in \{2, \ldots, m\}$, $k \in \{2, \ldots, n_j\}$ and $s = 1, \ldots, r$)*

$$M^{j,k}[G] \cdot (u^{s,1})_{1:r} = (u^{s,j})_k \cdot (u^{s,1})_{1:r}. \tag{2.0.7}$$

## 2.1 low-rank tensor decompositions

Without loss of generality, assume the dimensions are decreasing as $n_1 \geqslant n_2 \geqslant \cdots \geqslant n_m$. We discuss how to compute tensor decomposition for a tensor $\mathcal{F} \in \mathbb{C}^{n_1 \times \cdots \times n_m}$ when the rank $r$ is not bigger than the highest dimension, i.e., $r \leqslant n_1$. As in Theorem 2.0.2, the decomposing vectors $(u^{s,1})_{1:r}$ are common eigenvectors of the matrices $M^{j,k}[G]$, with $(u^{s,j})_k$ being the eigenvalues respectively. This implies that the matrices $M^{j,k}[G]$ are simultaneously diagonalizable. This property can be used to compute tensor decompositions.

Suppose $G$ is a matrix such that (2.0.6) holds and $M^{j,k}[G]$ are simultaneously diagonalizable. That is, there is an invertible matrix $P \in \mathbb{C}^{r \times r}$ such that all the

products $P^{-1}M^{j,k}[G]P$ are diagonal for all $j = 2, \ldots, m$ and for all $k = 2, \ldots, n_j$. Suppose $M^{j,k}[G]$ are diagonalized such that

$$P^{-1}M^{j,k}[G]P \;=\; \mathrm{diag}[\lambda_{j,k,1}, \lambda_{j,k,2}, \ldots, \lambda_{j,k,r}] \tag{2.1.1}$$

with the eigenvalues $\lambda_{j,k,s}$. For each $s = 1, \ldots, r$ and $j = 2, \ldots, m$, denote the vectors

$$u^{s,j} \;\coloneqq\; (1, \lambda_{j,2,s}, \ldots, \lambda_{j,n_j,s}). \tag{2.1.2}$$

When $\mathcal{F}$ is rank-$r$, there exist vectors $u^{1,1}, \ldots, u^{r,1} \in \mathbb{C}^{n_1}$ such that

$$\mathcal{F} = \sum_{s=1}^{r} u^{s,1} \otimes u^{s,2} \otimes \cdots \otimes u^{s,m}. \tag{2.1.3}$$

The vectors $u^{s,1}$ can be found by solving linear equations after $u^{s,j}$ are obtained for $j = 2, \ldots, m$ and $s = 1, \ldots, r$. The existence of vectors $u^{s,1}$ satisfying the tensor decomposition (2.1.3) is shown in the following theorem.

**Theorem 2.1.1.** *Let $\mathcal{F} = V^{(1)} \circ \ldots \circ V^{(m)}$ be a rank-$r$ tensor, for matrices $V^{(i)} \in \mathbb{C}^{n_i \times r}$, such that the first $r$ rows of $V^{(1)}$ are linearly independent. Suppose $G$ is a matrix satisfying* (2.0.6) *and $P \in \mathbb{C}^{r\times r}$ is an invertible matrix such that all matrix products $P^{-1} \cdot M^{j,k}[G] \cdot P$ are simultaneously diagonalized as in* (2.1.1)*. For $j = 2, \ldots, m$ and $s = 1, \ldots, r$, let $u^{s,j}$ be vectors given as in* (2.1.2)*. Then, there must exist vectors $u^{1,1}, \ldots, u^{r,1} \in \mathbb{C}^{n_1}$ such that the tensor decomposition* (2.1.3) *holds.*

*Proof.* Since the matrix $P = \begin{pmatrix} p_1 & \cdots & p_r \end{pmatrix}$ is invertible, there exist scalars $c_1, \ldots, c_r \in \mathbb{C}$ such that

$$\mathcal{F}_{1:r,1,\ldots,1} = c_1 p_1 + c_2 p_2 + \cdots + c_r p_r. \tag{2.1.4}$$

Consider the new tensor

$$\mathcal{H} \;\coloneqq\; \sum_{s=1}^{r} c_s p_s \otimes u^{s,2} \otimes \cdots \otimes u^{s,m}.$$

In the following, we show that $\mathcal{F}_{1:r,:,\ldots,:} = \mathcal{H}$ and there exist vectors $u^{1,1}, \ldots, u^{r,1} \in \mathbb{C}^{n_1}$ satisfying the equation (2.1.3).

By Theorem 2.0.2, one can see that the generating matrix $G$ for $\mathcal{F}$ is also a generating matrix for $\mathcal{H}$, so it holds that

$$\langle \phi[G, \tau]p, \mathcal{F}\rangle = \langle \phi[G, \tau]p, \mathcal{H}\rangle = 0, \quad \text{for all } p \in \mathbb{M}_{\{1,j\}^c}. \tag{2.1.5}$$

Therefore, we have

$$\langle \phi[G,\tau]p, \mathcal{H}-\mathcal{F}\rangle = 0, \quad \text{for all } p \in \mathbb{M}_{\{1,j\}^c}. \tag{2.1.6}$$

By (2.1.4), one can see that

$$(\mathcal{H}-\mathcal{F})_{1:r,1,\dots,1} = 0. \tag{2.1.7}$$

In (2.1.6), for each $\tau = (i,2,k) \in J$ and $p = 1$, we can get

$$(\mathcal{H}-\mathcal{F})_{1:r,:,1,\dots,1} = 0.$$

Similarly, for $\tau = (i,2,k) \in J$ and $p = x_{3,j_3}$, we can get

$$(\mathcal{H}-\mathcal{F})_{1:r,:,:,1,\dots,1} = 0.$$

Continuing this, we can eventually get $\mathcal{H} = \mathcal{F}_{1:r,:,:,\dots,:}$. Since the matrix $(V^{(1)})_{1:r,:}$ is invertible, there exists a matrix $W \in \mathbb{C}^{n_1 \times r}$ such that $V^{(1)} = W(V^{(1)})_{1:r,:}$. Observe that

$$\mathcal{F} = W \times_1 \mathcal{F}_{1:r,:,:,\dots,:} = W \times_1 \mathcal{H}.$$

Let $u^{s,1} = W \cdot (c_s p_s)$ for $s = 1,\dots,r$. Then the tensor decomposition (2.1.3) holds. □

### 2.1.1 An algorithm for computing tensor decompositions

Consider a tensor $\mathcal{F} \in \mathbb{C}^{n_1 \times n_2 \times \cdots \times n_m}$ with a given rank $r$. Recall that the dimensions are ordered such that $n_1 \geqslant n_2 \geqslant \cdots \geqslant n_m$. We discuss how to compute a rank-$r$ tensor decomposition for $\mathcal{F}$. Recall $A[\mathcal{F},j]$, $b[\mathcal{F},j,k]$ as in (2.0.5), for $j > 1$. Note that $A[\mathcal{F},j]$ has the dimension $N_j \times r$, where

$$N_j \coloneqq \frac{n_2 \cdots n_m}{n_j}. \tag{2.1.8}$$

If $r \leqslant N_j$, then the matrices $A[\mathcal{F},j]$ have full column rank for generic cases. For instance, $r \leqslant N_3$ if $m = 3$ and $r \leqslant n_2$. Since $N_2$ is the smallest, we often use the matrices $A[\mathcal{F},j]$ for $j \geqslant 3$. For convenience, denote the label set

$$\Upsilon \coloneqq \{(j,k) : 3 \leqslant j \leqslant m, 2 \leqslant k \leqslant n_j\}. \tag{2.1.9}$$

In the following, we consider the case that $r \leqslant N_3$. For each pair $(j,k) \in \Upsilon$, the linear system (2.0.6) has a unique solution, for which we denote

$$Y^{j,k} \;=\; M^{j,k}[G].$$

For $j = 2$, the equation (2.0.6) may not have a unique solution if $r > N_2$. In the following, we show how to get the tensor decomposition without using the matrices $M^{2,k}[G]$. By Theorem 2.0.2, the matrices $Y^{j,k}$ are simultaneously diagonalizable, that is, there is an invertible matrix $P \in \mathbb{C}^{r\times r}$ such that all products $P^{-1}Y^{j,k}P$ are diagonal for every $(j,k) \in \Upsilon$. Suppose they are diagonalized as

$$P^{-1}Y^{j,k}P \;=\; \mathrm{diag}[\lambda_{j,k,1}, \lambda_{j,k,2}, \ldots, \lambda_{j,k,r}] \tag{2.1.10}$$

with the eigenvalues $\lambda_{j,k,s}$. Write $P$ in the column form

$$P \;=\; \begin{pmatrix} p_1 & \cdots & p_r \end{pmatrix}.$$

For each $s = 1, \ldots, r$ and $j = 3, \ldots, m$, let

$$v^{s,j} \;\coloneqq\; (1, \lambda_{j,2,s}, \ldots, \lambda_{j,n_j,s}). \tag{2.1.11}$$

Suppose $\mathcal{F}$ has a rank-$r$ decomposition

$$\mathcal{F} \;=\; \sum_{s=1}^{r} u^{s,1} \otimes \ldots \otimes u^{s,m}.$$

Under the assumptions of Theorem 2.0.2, the linear system (2.0.6) has a unique solution for each pair $(j,k) \in \Upsilon$. For every $j \in \{3, \ldots, m\}$, there exist scalars $c_{s,j}, c_{s,1}$ such that

$$u^{s,j} \;=\; c_{s,j} v^{s,j}, \quad u^{s,1} \;=\; c_{s,1} p_s.$$

Then, we consider the sub-tensor equation in the vector variables $y_1, \ldots, y_r \in \mathbb{C}^{n_2}$

$$\mathcal{F}_{1:r,:,\ldots,:} \;=\; \sum_{s=1}^{r} p_s \otimes y_s \otimes v^{s,3} \otimes \cdots \otimes v^{s,m}. \tag{2.1.12}$$

There are $rn_2 \cdots n_m$ equations and $rn_2$ unknowns. This overdetermined linear system has solutions such that

$$y_s = c_{s,2} u^{s,2}, \text{ for some } c_{s,2} \in \mathbb{C}.$$

After all $y_s$ are obtained, we solve the linear equation in $z_1, \ldots, z_r \in \mathbb{C}^{n_1 - r}$

$$\mathcal{F}_{r+1:n_1,:,\ldots,:} = \sum_{s=1}^{r} z_s \otimes y_s \otimes v^{s,3} \otimes \cdots \otimes v^{s,m}. \tag{2.1.13}$$

After all $y_s, z_s$ are obtained, we choose the vectors $(s = 1, \ldots, r)$

$$v^{s,1} = \begin{pmatrix} p_s \\ z_s \end{pmatrix}, \quad v^{s,2} = y_s.$$

Then we get the tensor decomposition

$$\mathcal{F} = \sum_{s=1}^{r} v^{s,1} \otimes v^{s,2} \otimes \cdots \otimes v^{s,m}. \tag{2.1.14}$$

Summarizing the above, we get the following algorithm for computing tensor decompositions when $r \leqslant n_1$ and $r \leqslant N_3$. Suppose the dimensions are ordered such that $n_1 \geqslant n_2 \geqslant \cdots \geqslant n_m$.

**Algorithm 2.1.2.** (Rank-$r$ tensor decomposition.)

**Input** A tensor $\mathcal{F} \in \mathbb{C}^{n_1 \times \ldots \times n_m}$ with rank $r \leqslant \min(n_1, N_3)$.

**Step 1** For each pair $(j,k) \in \Upsilon$, solve the matrix equation for the solution $Y^{j,k}$:

$$A[\mathcal{F}, j] Y^{j,k} = b[\mathcal{F}, j, k]. \tag{2.1.15}$$

**Step 2** Choose generic scalars $\xi_{j,k}$. Then compute the eigenvalue decomposition $P^{-1} Y P = D$ for the matrix

$$Y \coloneqq \frac{1}{\sum\limits_{(j,k)\in\Upsilon} \xi_{j,k}} \sum_{(j,k)\in\Upsilon} \xi_{j,k} Y^{j,k}.$$

**Step 3** For $s = 1, \ldots, r$ and $j \geqslant 3$, let $v^{s,j}$ be the vectors as in (2.1.11).

**Step 4** Solve the linear system (2.1.12) for vectors $y_1, \ldots, y_r$.

**Step 5** Solve the linear system (2.1.13) for vectors $z_1, \ldots, z_r$.

**Step 6** For each $s = 1, \ldots, r$, let $v^{s,1} = \begin{pmatrix} p_s \\ z_s \end{pmatrix}$ and $v^{s,2} = y_s$.

**Output** A tensor rank-$r$ decomposition as in (2.1.14).

The correctness of Algorithm 2.1.2 is justified as follows.

**Theorem 2.1.3.** *Suppose* $n_1 \geqslant n_2 \geqslant \cdots \geqslant n_m$ *and* $r \leqslant \min(n_1, N_3)$ *as in* (2.1.8). *For a generic tensor* $\mathcal{F}$ *of rank-*$r$*, Algorithm 2.1.2 produces a rank-*$r$ *tensor decomposition for* $\mathcal{F}$.

*Proof.* This can be implied by Theorem 2.1.1. □

### 2.1.2 Tensor decompositions via reshaping

A tensor $\mathcal{F} \in \mathbb{C}^{n_1 \times \cdots \times n_m}$ can be reshaped as a cubic order tensor $\widehat{\mathcal{F}}$ as in (1.1.12). One can apply Algorithm 2.1.2 to compute the tensor decomposition (1.1.12) for $\widehat{\mathcal{F}}$. If the decomposing vectors $w^{s,1}, w^{s,2}, w^{s,3}$ can be reshaped to rank-1 tensors, then we can convert (1.1.12) to a tensor decomposition for $\mathcal{F}$. This is justified by Theorem 1.1.3, under some assumptions. A benefit of doing this is that we may be able to compute tensor decompositions for the case that

$$N_3 < r \leqslant p_2,$$

with the dimension $p_2$ as in Theorem 1.1.3. This leads to the following algorithm for computing tensor decompositions.

**Algorithm 2.1.4.** (Tensor decompositions via reshaping.) Let $p_1, p_2, p_3$ be dimensions as in Theorem 1.1.3.

**Input** A tensor $\mathcal{F} \in \mathbb{C}^{n_1 \times \cdots \times n_m}$ with rank $r \leqslant p_2$.

**Step 1** Reshape the tensor $\mathcal{F}$ to a cubic tensor $\widehat{\mathcal{F}} \in \mathbb{C}^{p_1 \times p_2 \times p_3}$ as in (1.1.12).

**Step 2** Use Algorithm 2.1.2 to compute the tensor decomposition

$$\widehat{\mathcal{F}} = \sum_{s=1}^{r} w^{s,1} \otimes w^{s,2} \otimes w^{s,3}. \tag{2.1.16}$$

**Step 3** If all $w^{s,1}, w^{s,2}, w^{s,3}$ can be expressed as outer products of rank-1 tensors as in (1.1.11), then output the tensor decomposition as in (1.1.3). If one of $w^{s,1}, w^{s,2}, w^{s,3}$ cannot be expressed as in (1.1.11), then the reshaping does not produce a tensor decomposition for $\mathcal{F}$.

**Output** A tensor decomposition for $\mathcal{F}$ as in (1.1.3).

For Algorithm 2.1.4, we have a similar conclusion like Theorem 2.1.3. For the cleanness, we do not repeat it here.

## 2.2 low-rank Tensor Approximations

When a tensor $\mathcal{F} \in \mathbb{C}^{n_1 \times \ldots \times n_m}$ has the rank bigger than $r$, the linear systems in Algorithm 2.1.2 may not be consistent. However, we can find linear least squares solutions for them. This gives an algorithm for computing low-rank tensor approximations. Recall the label set $\Upsilon$ as in (2.1.9). The following is the algorithm.

**Algorithm 2.2.1.** (Rank-$r$ tensor approximation.)

**Input** A tensor $\mathcal{F} \in \mathbb{C}^{n_1 \times n_2 \times \ldots \times n_m}$ and a rank $r \leqslant \min(n_1, N_3)$.

**Step 1** For each pair $(j,k) \in \Upsilon$, solve the linear least squares problem

$$\min_{Y^{j,k} \in \mathbb{C}^{r \times r}} \quad \left\| A[\mathcal{F}, j](Y^{j,k})^T - b[\mathcal{F}, j, k] \right\|^2. \tag{2.2.1}$$

Let $\hat{Y}^{j,k}$ be an optimizer.

**Step 2** Choose generic scalars $\xi_{j,k}$ and let

$$\hat{Y}[\xi] \;=\; \frac{1}{\sum\limits_{(j,k)\in\Upsilon} \xi_{j,k}} \sum_{(j,k)\in\Upsilon} \xi_{j,k} \hat{Y}^{j,k}.$$

Compute the eigenvalue decomposition $\hat{P}^{-1}\hat{Y}[\xi]\hat{P} = \Lambda$ with $\hat{P} = \begin{pmatrix} \hat{p}_1 & \cdots & \hat{p}_r \end{pmatrix}$ is invertible and $\Lambda$ is diagonal.

**Step 3** For each pair $(j,k) \in \Upsilon$, select the diagonal entries

$$\operatorname{diag}[\hat{\lambda}_{j,k,1}\ \hat{\lambda}_{j,k,2}\ \ldots\ \hat{\lambda}_{j,k,r}] \;=\; \operatorname{diag}(\hat{P}^{-1}\hat{Y}^{j,k}\hat{P}).$$

For each $s = 1, \ldots, r$ and $j = 3, \ldots, m$, let

$$\hat{v}^{s,j} = (1, \hat{\lambda}_{j,2,2}, \ldots, \hat{\lambda}_{j,n_j,s}).$$

**Step 4** Let $(\hat{y}_1, \ldots, \hat{y}_r)$ be an optimizer for the following least squares:

$$\min_{(y_1,\ldots,y_r)} \quad \left\| \mathcal{F}_{1:r,:,\ldots,:} - \sum_{s=1}^{r} \hat{p}_s \otimes y_s \otimes \hat{v}^{s,3} \otimes \cdots \otimes \hat{v}^{s,m} \right\|^2. \tag{2.2.2}$$

**Step 5** Let $(\hat{z}_1, \ldots, \hat{z}_r)$ be an optimizer for the following least squares:

$$\min_{(z_1,\ldots,z_r)} \quad \left\| \mathcal{F}_{r+1:n_1,:,\ldots,:} - \sum_{s=1}^{r} z_s \otimes \hat{y}_s \otimes \hat{v}^{s,3} \otimes \cdots \otimes \hat{v}^{s,m} \right\|^2. \tag{2.2.3}$$

**Step 6** Let $\hat{v}^{s,1} = \begin{pmatrix} \hat{p}_s \\ \hat{z}_s \end{pmatrix}$ and $\hat{v}^{s,2} = \hat{y}_s$ for each $s = 1, \ldots, r$.

**Output** A rank-$r$ approximation tensor

$$\mathcal{X}^{gp} := \sum_{s=1}^{r} \hat{v}^{s,1} \otimes \hat{v}^{s,2} \otimes ... \otimes \hat{v}^{s,m}. \tag{2.2.4}$$

If $\mathcal{F}$ is sufficiently close to a rank-$r$ tensor, then $\mathcal{X}^{gp}$ is expected to be a good rank-$r$ approximation. Mathematically, the tensor $\mathcal{X}^{gp}$ produced by Algorithm 2.2.1 may not be a best rank-$r$ approximation. However, in computational practice, we can use (2.2.4) as a starting point to solve the nonlinear least squares optimization

$$\min_{(u^{s,1},\ldots,u^{s,m})} \left\| \mathcal{F} - \sum_{s=1}^{r} u^{s,1} \otimes u^{s,2} \otimes ... \otimes u^{s,m} \right\|^2. \tag{2.2.5}$$

to improve the approximation quality. Let $\mathcal{X}^{opt}$ be a rank-$r$ approximation tensor

$$\mathcal{X}^{opt} := \sum_{s=1}^{r} u^{s,1} \otimes u^{s,2} \otimes ... \otimes u^{s,m} \tag{2.2.6}$$

which is an optimizer to (2.2.5) obtained by nonlinear optimization methods with $\mathcal{X}^{opt}$ as the initial point.

### 2.2.1 Approximation error analysis

Suppose the tensor $\mathcal{F}$ has a best (or nearly best) rank-$r$ approximation

$$\mathcal{X}^{bs} := \sum_{s=1}^{r} (x^{s,1}) \otimes (x^{s,2}) \otimes ... \otimes (x^{s,m}). \tag{2.2.7}$$

Let $\mathcal{E}$ be the tensor such that

$$\mathcal{F} = \mathcal{X}^{bs} + \mathcal{E}. \tag{2.2.8}$$

We analyze the approximation performance of $\mathcal{X}^{gp}$ when the distance $\epsilon = \|\mathcal{E}\|$ is small. For a generating matrix $G$ and a generic $\xi = (\xi_{j,k})_{(j,k)\in\Upsilon}$, denote that

$$M[\xi, G] := \frac{1}{\sum\limits_{(j,k)\in\Upsilon} \xi_{j,k}} \sum_{(j,k)\in\Upsilon} \xi_{j,k} M^{j,k}[G]. \tag{2.2.9}$$

Recall the $A[\mathcal{F}, j]$, $b[\mathcal{F}, j, k]$ as in (2.0.5). Note that

$$\begin{aligned} A[\mathcal{F}, j] &= A[\mathcal{X}^{bs}, j] + A[\mathcal{E}, j], \\ b[\mathcal{F}, j, k] &= b[\mathcal{X}^{bs}, j, k] + b[\mathcal{E}, j, k]. \end{aligned} \tag{2.2.10}$$

Suppose $(x^{s,j})_1 \neq 0$ for $j = 2, \ldots, m$.

Up to a scaling, we can further assume that

$$\left(\mathbf{u}^{bs}_{s,2}\right)_1 = \cdots = \left(\mathbf{u}^{bs}_{s,m}\right)_1 = 1. \tag{2.2.11}$$

**Theorem 2.2.2.** *Let $\mathcal{X}^{gp}$ be produced by Algorithm 2.2.1. Let $\mathcal{F}, \mathcal{X}^{bs}, \mathcal{X}^{opt}, \mathcal{E}, x^{s,j}, \xi_{j,k}$ be as above. Assume the following conditions hold:*

*(i) The subvectors $(x^{1,1})_{1:r}, \ldots, (x^{r,1})_{1:r}$ are linearly independent.*

*(ii) All matrices $A[\mathcal{F}, j]$ and $A[\mathcal{X}^{bs}, j]$ $(3 \leqslant j \leqslant m)$ have full column rank.*

*(iii) The first entry $(x^{s,j})_1 \neq 0$ for all $j = 2, \ldots, m$.*

*(iv) The following scalars are pairwisely distinct*

$$\sum_{(j,k)\in\Upsilon} \xi_{j,k}(x^{1,j})_k, ..., \sum_{(j,k)\in\Upsilon} \xi_{j,k}(x^{r,j})_k. \tag{2.2.12}$$

*If the distance $\epsilon = \|\mathcal{F} - \mathcal{X}^{bs}\|$ is sufficiently small, then*

$$\|\mathcal{X}^{bs} - \mathcal{X}^{gp}\| = O(\epsilon) \quad \text{and} \quad \|\mathcal{F} - \mathcal{X}^{gp}\| = O(\epsilon). \tag{2.2.13}$$

*where the constants in the above $O(\cdot)$ only depend on $\mathcal{F}$ and $\xi$.*

*Proof.* By conditions (i) and (iii) and by Theorem 2.0.2, there exists a generating matrix $G^{bs}$ for $\mathcal{X}^{bs}$ such that

$$A[\mathcal{X}^{bs}, j](M^{j,k}[G^{bs}])^T = b[\mathcal{X}^{bs}, j, k] \tag{2.2.14}$$

for all $j \in \{2, \ldots, m\}$ and $k \in \{2, \ldots, n_j\}$. Note that $Y^{j,k}$ is the least squares solution to (2.2.1), so for each $(j, k) \in \Upsilon$,

$$Y^{j,k} = A[\mathcal{F}, j]^\dagger \cdot b[\mathcal{F}, j, k], \quad M^{j,k}[G^{bs}_0] = A[\mathcal{X}^{bs}, j]^\dagger \cdot b[\mathcal{X}^{bs}, j, k].$$

(The super script $^\dagger$ denotes the Pseudo-inverse of a matrix.) By (2.2.8), for $j = 2, \ldots, m$, we have

$$\begin{aligned} \left\|A[\mathcal{F}, j] - A[\mathcal{X}^{bs}, j]\right\|_F &\leqslant \left\|\mathcal{F} - \mathcal{X}^{bs}\right\| \leqslant \epsilon, \\ \left\|b[\mathcal{F}, j, k] - b[\mathcal{X}^{bs}, j, k]\right\|_F &\leqslant \left\|\mathcal{F} - \mathcal{X}^{bs}\right\| \leqslant \epsilon. \end{aligned} \tag{2.2.15}$$

Hence, by the condition (ii), if $\epsilon > 0$ is small enough, we have

$$\left\| Y^{j,k} - M^{j,k}[G^{bs}] \right\| = O(\epsilon). \tag{2.2.16}$$

for all $(j,k) \in \Upsilon$. This follows from perturbation analysis for linear least squares (see [15, Theorem 3.4]).

By (2.2.7) and Theorem 2.0.2, for $s = 1, \ldots, r$ and $(j,k) \in \Upsilon$, it holds that

$$M^{j,k}[G^{bs}] \left(x^{s,1}\right)_{1:r} \quad = \quad \left(x^{s,j}\right)_k \left(x^{s,1}\right)_{1:r}.$$

This means that each $(x^{s,1})_{1:r}$ is an eigenvector of $M^{j,k}[G^{bs}]$, associated to the eigenvalue $(x^{s,j})_k$, for each $s = 1, \ldots, r$. The matrices $M^{j,k}[G^{bs}]$ are simultaneously diagonalizable, by the condition (i). So $M[\xi, G^{bs}]$ is also diagonalizable. Note the eigenvalues of $M[\xi, G^{bs}]$ are the sums in (2.2.12). They are distinct from each other, by the condition (iv). When $\epsilon > 0$ is small enough, $M[\xi, G^{bs}]$ also has distinct eigenvalues. Write that

$$Q = \begin{pmatrix} (x^{1,1})_{1:r} & \cdots & (x^{r,1})_{1:r} \end{pmatrix}.$$

Note that $Q^{-1} M[\xi, G^{bs}] Q = D$ is an eigenvalue decomposition. Up to a scaling on $\hat{P}$ in algorithm 2.2.1, it holds that

$$\|\hat{p}_s - x^{s,1}\|_2 = O(\epsilon), \quad \|D - \Lambda\|_F = O(\epsilon). \tag{2.2.17}$$

We refer to [6] for the perturbation bounds in (2.2.17). The constants in the above $O(\cdot)$ eventually only depend on $\mathcal{F}, \xi$.

Note that $(\hat{y}_s, \ldots, \hat{y}_r)$ is the least squares solution to (2.2.2) and

$$\mathcal{X}^{bs}_{1:r,:,\ldots,:} = \sum_{s=1}^{r} x^{s,1} \otimes x^{s,2} \otimes x^{s,3} \otimes \cdots \otimes x^{s,m}. \tag{2.2.18}$$

Due to perturbation analysis of linear least squares, we also have

$$\|\hat{y}_s - x^{s,2}\|_2 = O(\epsilon). \tag{2.2.19}$$

Note that the subvectors $(x^{s,1})_{r+1:n_1}$ satisfy the equation

$$\mathcal{X}^{bs}_{r+1:n_1,:,\ldots,:} = \sum_{s=1}^{r} (x^{s,1})_{r+1:n_1} \otimes x^{s,2} \otimes \cdots \otimes x^{s,m}. \tag{2.2.20}$$

Recall that $(\hat{z}_1, \ldots, \hat{z}_r)$ is the least squares solution to (2.2.3). Due to perturbation analysis of linear least squares, we further have the error bound

$$\|(x^{s,1})_{r+1:n_1} - \hat{z}_s\|_2 = O(\epsilon). \tag{2.2.21}$$

Summarizing the above, we eventually get $\|\mathcal{X}^{gp} - \mathcal{X}^{bs}\| = O(\epsilon)$, so

$$\|\mathcal{F} - \mathcal{X}^{gp}\| \leqslant \left\|\mathcal{F} - \mathcal{X}^{bs}\right\| + \left\|\mathcal{X}^{bs} - \mathcal{X}^{gp}\right\| = O(\epsilon).$$

The constant for the above $O(\cdot)$ eventually only depends on $\mathcal{F}$, $\xi$. □

### 2.2.2 Reshaping for low-rank approximations

Similar to tensor decompositions, the reshaping trick as in Section 2.1.2 can also be used for computing low-rank tensor approximations. For $m > 3$, a tensor $\mathcal{F} \in \mathbb{C}^{n_1 \times n_2 \times \cdots \times n_m}$ can be reshaped as a cubic tensor $\widehat{\mathcal{F}} \in \mathbb{C}^{p_1 \times p_2 \times p_3}$ as in (1.1.12). Similarly, Algorithm 2.2.1 can be used to compute low-rank tensor approximations. Suppose the computed rank-$r$ approximating tensor for $\widehat{\mathcal{F}}$ is

$$\widehat{\mathcal{X}}^{gp} := \sum_{s=1}^{r} \hat{w}^{s,1} \otimes \hat{w}^{s,2} \otimes \hat{w}^{s,3}. \tag{2.2.22}$$

Typically, the decomposing vectors $\hat{w}^{s,1}, \hat{w}^{s,2}, \hat{w}^{s,3}$ may not be reshaped to rank-1 tensors. Suppose the reshaping is such that $I_1 \cup I_2 \cup I_3 = \{1, 2, ..., m\}$ is a union of disjoint label sets and the reshaped dimensions are

$$p_1 = \prod_{i \in I_1} n_i, \quad p_2 = \prod_{i \in I_2} n_i, \quad p_3 = \prod_{i \in I_3} n_i.$$

Let $m_i = |I_i|$ for $i = 1, 2, 3$. By the reshaping, the vectors $\hat{w}^{s,i}$ can be reshaped back to a tensor $\hat{W}^{s,i}$ of order $m_i$, for each $i = 1, 2, 3$. If $m_i = 1$, $\hat{W}^{s,i}$ is a vector. If $m_i = 2$, we can find a best rank-1 matrix approximation for $\hat{W}^{s,i}$. If $m_i \geqslant 3$, we can apply Algorithm 2.2.1 with $r = 1$ to get a rank-1 approximation for $\hat{W}^{s,i}$. In application, we are mostly interested in reshaping such that all $m_i \leqslant 2$. Finally, this produces a rank-$r$ approximation for $\mathcal{F}$.

The following is a low-rank tensor approximation algorithm via reshaping tensors.

**Algorithm 2.2.3.** (low-rank tensor approximations via reshaping.)

**Input** A tensor $\mathcal{F} \in \mathbb{C}^{n_1 \times n_2 \times \ldots \times n_m}$ and a rank $r$.

**Step 1** Reshape $\mathcal{F}$ to a cubic order tensor $\widehat{\mathcal{F}} \in \mathbb{C}^{p_1 \times p_2 \times p_3}$.

**Step 2** Use Algorithm 2.2.1 to compute a rank-$r$ approximating tensor $\widehat{\mathcal{X}}^{gp}$ as in (2.2.22) for $\widehat{\mathcal{F}}$.

**Step 3** For each $i = 1, 2, 3$, reshape each vector $\hat{w}^{s,i}$ back to a tensor $\widehat{W}^{s,i}$ of order $m_i$ as above.

**Step 4** For each $i = 1, 2, 3$, compute a rank-1 approximating tensor $\widehat{X}^{s,i}$ for $\widehat{W}^{s,i}$ of order $m_i$ as above.

**Output** Reshape the sum $\sum_{s=1}^{r} \widehat{X}^{s,1} \otimes \widehat{X}^{s,2} \otimes \widehat{X}^{s,3}$ to a tensor in $\mathbb{C}^{n_1 \times n_2 \times \ldots \times n_m}$, which is a rank-$r$ approximation for $\mathcal{F}$.

We can do a similar approximation analysis for Algorithm 2.2.3 as for Theorem 2.2.2. For the cleanness, we do not repeat that.

## 2.3 Numerical Experiments

In this section, we apply Algorithms 2.1.2 and 2.2.1 to compute tensor decompositions and low-rank tensor approximations. We implement these algorithms in `MATLAB` 2020b on a workstation with Ubuntu 20.04.2 LTS, Intel® Xeon(R) Gold 6248R CPU @ 3.00GHz and memory 1TB. For computing low-rank tensor approximations, we use the function *cpd_nls* provided in `Tensorlab 3.0` [58] to solve the nonlinear least squares optimization (2.2.5). The $\mathcal{X}^{gp}$ denotes the approximating tensor returned by Algorithm 2.2.1 and $\mathcal{X}^{opt}$ denotes the approximating tensor obtained by solving (2.2.5), with $\mathcal{X}^{gp}$ as the initial point. In our numerical experiments, if the rank $r$ is unknown, we use the most square flattening matrix to estimate $r$ as in (1.1.4) and Lemma 1.1.1.

**Example 2.3.1.** Consider the tensor $\mathcal{F} \in \mathbb{C}^{4\times4\times3}$ whose slices $\mathcal{F}_{:,:,1}, \mathcal{F}_{:,:,2}, \mathcal{F}_{:,:,3}$ are respectively

$$\left[\begin{array}{cccc|cccc|cccc} 27 & 25 & 35 & 42 & 44 & 32 & 52 & 56 & 42 & 26 & 48 & 45 \\ 48 & 68 & 80 & 80 & 68 & 76 & 100 & 96 & 64 & 60 & 88 & 76 \\ 26 & 24 & 34 & 40 & 42 & 30 & 50 & 52 & 47 & 27 & 53 & 47 \\ 33 & 41 & 49 & 66 & 46 & 46 & 62 & 76 & 45 & 37 & 57 & 60 \end{array}\right].$$

By Lemma 1.1.1, the estimated rank is $r = 4$.

Applying Algorithm 2.1.2 with $r = 4$, we get the rank-4 decomposition $\mathcal{F} = U^{(1)} \circ U^{(2)} \circ U^{(3)}$, with

$$U^{(1)} = \begin{bmatrix} 8 & 6 & 4 & 9 \\ 8 & 12 & 16 & 12 \\ 4 & 6 & 4 & 12 \\ 4 & 12 & 8 & 9 \end{bmatrix}, \; U^{(2)} = \begin{bmatrix} 1 & 1 & 1 & 1 \\ \frac{1}{2} & 1 & 3 & \frac{1}{3} \\ 1 & 1 & 3 & 1 \\ 1 & 4 & 1 & \frac{2}{3} \end{bmatrix}, \; U^{(3)} = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 2 & 1 & 1 & 2 \\ 1 & \frac{2}{3} & \frac{3}{4} & 3 \end{bmatrix}.$$

**Example 2.3.2.** Consider the tensor in $\mathbb{C}^{5\times4\times3\times3}$

$$\mathcal{F} = V^{(1)} \circ V^{(2)} \circ V^{(3)} \circ V^{(4)},$$

where the matrices $V^{(i)}$ are

$$V^{(1)} = \begin{bmatrix} 10 & 5 & -9 & -5 & 7 \\ 8 & 6 & -3 & -9 & 7 \\ -9 & -1 & 7 & -3 & -1 \\ 9 & -7 & -8 & 8 & -5 \\ -1 & 10 & 7 & -3 & 10 \end{bmatrix}, \quad V^{(2)} = \begin{bmatrix} -1 & 9 & -8 & 8 & 2 \\ 0 & -1 & -4 & 6 & 8 \\ 7 & -7 & -2 & 2 & 10 \\ 2 & 10 & -3 & -1 & -3 \end{bmatrix},$$

$$V^{(3)} = \begin{bmatrix} 5 & 2 & -2 & -7 & 3 \\ 9 & -3 & -7 & 7 & -2 \\ 0 & -10 & 10 & 6 & 10 \end{bmatrix}, \quad V^{(4)} = \begin{bmatrix} 8 & 2 & -7 & 10 & -5 \\ 4 & -8 & 4 & -6 & -10 \\ 5 & 0 & 7 & -1 & -2 \end{bmatrix}.$$

By Lemma 1.1.1, the estimated rank $r = 5$.

Applying Algorithm 2.1.2 with $r = 5$, we get the rank-5 tensor decomposition $\mathcal{F} = U^{(1)} \circ U^{(2)} \circ U^{(3)} \circ U^{(4)}$, where the computed matrices $U^{(i)}$ are

$$U^{(1)} = \begin{bmatrix} -400 & 180 & 1008 & 2800 & -210 \\ -320 & 216 & 336 & 5040 & -210 \\ 360 & -36 & -784 & 1680 & 30 \\ -360 & -252 & 896 & -4480 & 150 \\ 40 & 360 & -784 & 1680 & -300 \end{bmatrix}, \quad U^{(2)} = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 \\ 0 & -\frac{1}{9} & \frac{1}{2} & \frac{3}{4} & 4 \\ -7 & -\frac{7}{9} & \frac{1}{4} & \frac{1}{4} & 5 \\ -2 & \frac{10}{9} & \frac{3}{8} & -\frac{1}{8} & -\frac{3}{2} \end{bmatrix},$$

$$U^{(3)} = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 \\ \frac{9}{5} & -\frac{3}{2} & \frac{7}{2} & -1 & -\frac{2}{3} \\ 0 & -5 & -5 & -\frac{6}{7} & \frac{10}{3} \end{bmatrix}, \quad U^{(4)} = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 \\ \frac{1}{2} & -4 & -\frac{4}{7} & -\frac{3}{5} & 2 \\ \frac{5}{8} & 0 & -1 & -\frac{1}{10} & \frac{2}{5} \end{bmatrix}.$$

**Example 2.3.3.** Consider the tensor $\mathcal{F} \in \mathbb{C}^{5\times5\times4}$ such that

$$\mathcal{F}_{i_1,i_2,i_3} = i_1 + \frac{i_2}{2} + \frac{i_3}{3} + \sqrt{i_1^2 + i_2^2 + i_3^2}$$

for all $i_1, i_2, i_3$ in the corresponding range. The 5 biggest singular values of the flattening matrix $\text{Flat}(\mathcal{F})$ are

$$109.7393, \quad 5.2500, \quad 0.1068, \quad 8.325 \times 10^{-3}, \quad 3.401 \times 10^{-4}.$$

Applying Algorithm 2.2.1 with rank $r = 2, 3, 4, 5$, we get the approximation errors

| $r$ | 2 | 3 | 4 | 5 |
|---|---|---|---|---|
| $\|\mathcal{F} - \mathcal{X}^{gp}\|$ | $5.1237 \times 10^{-1}$ | $6.8647 \times 10^{-2}$ | $1.0558 \times 10^{-2}$ | $9.9449 \times 10^{-3}$ |
| $\|\mathcal{F} - \mathcal{X}^{opt}\|$ | $1.5410 \times 10^{-1}$ | $1.3754 \times 10^{-2}$ | $2.6625 \times 10^{-3}$ | $4.9002 \times 10^{-4}$ |

For the case $r = 3$, the computed approximating tensor by Algorithm 2.2.1 and by solving (2.2.5) is $U^{(1)} \circ U^{(2)} \circ U^{(3)}$, with

$$U^{(1)} = \begin{bmatrix} -0.4973 & -7.6813 & 11.7465 \\ -0.2525 & -6.9651 & 12.4970 \\ -0.0872 & -6.0497 & 13.2858 \\ -0.0132 & -5.0521 & 14.1423 \\ -0.0010 & -4.0469 & 15.0771 \end{bmatrix}, \quad U^{(2)} = \begin{bmatrix} 1.0000 & 1.0000 & 1.0000 \\ 0.5058 & 0.9211 & 1.0306 \\ 0.1713 & 0.8167 & 1.0649 \\ 0.0262 & 0.7003 & 1.1042 \\ 0.0136 & 0.5807 & 1.1490 \end{bmatrix},$$

$$U^{(3)} = \begin{bmatrix} 1.0000 & 1.0000 & 1.0000 \\ 0.5075 & 0.9289 & 1.0216 \\ 0.1756 & 0.8323 & 1.0469 \\ 0.0399 & 0.7231 & 1.0771 \end{bmatrix}.$$

**Example 2.3.4.** Consider the tensor $\mathcal{F} \in \mathbb{C}^{6\times6\times6\times5\times4}$ such that

$$\mathcal{F}_{i_1,i_2,i_3,i_4,i_5} = \ arctan(i_1 + 2i_2 + 3i_3 + 4i_4 + 5i_5),$$

for all $i_1, i_2, i_3, i_4, i_5$ in the corresponding range. The 5 biggest singular values of the flattening matrix $\text{Flat}(\mathcal{F})$ are

$$101.71, \quad 7.7529 \times 10^{-2}, \quad 2.2870 \times 10^{-3}, \quad 7.2294 \times 10^{-5}, \quad 2.0633 \times 10^{-6}.$$

Applying Algorithm 2.2.1 with rank $r = 2, 3, 4, 5$, we get the approximation errors as follows:

| $r$ | 2 | 3 | 4 | 5 |
|---|---|---|---|---|
| $\|\mathcal{F} - \mathcal{X}^{gp}\|$ | $9.8148 \times 10^{-3}$ | $3.1987 \times 10^{-3}$ | $5.7945 \times 10^{-3}$ | $1.0121 \times 10^{-5}$ |
| $\|\mathcal{F} - \mathcal{X}^{opt}\|$ | $5.3111 \times 10^{-3}$ | $2.2623 \times 10^{-4}$ | $3.0889 \times 10^{-5}$ | $1.7523 \times 10^{-6}$ |

For the case $r = 3$, the computed approximating tensor by Algorithm 2.2.1 and by solving (2.2.5) is $U^{(1)} \circ U^{(2)} \circ U^{(3)} \circ U^{(4)} \circ U^{(5)}$, with

$$U^{(1)} = \begin{bmatrix} -0.0134 & -0.0347 & 1.5524 \\ -0.0112 & -0.0329 & 1.5525 \\ -0.0094 & -0.0312 & 1.5526 \\ -0.0079 & -0.0295 & 1.5527 \\ -0.0066 & -0.0280 & 1.5528 \\ -0.0056 & -0.0265 & 1.5529 \end{bmatrix}, \quad U^{(2)} = \begin{bmatrix} 1.0000 & 1.0000 & 1.0000 \\ 0.7011 & 0.8992 & 1.0001 \\ 0.4939 & 0.8080 & 1.0003 \\ 0.3485 & 0.7260 & 1.0004 \\ 0.2459 & 0.6523 & 1.0006 \\ 0.1734 & 0.5861 & 1.0007 \end{bmatrix},$$

$$U^{(3)} = \begin{bmatrix} 1.0000 & 1.0000 & 1.0000 \\ 0.5886 & 0.8523 & 1.0002 \\ 0.3490 & 0.7258 & 1.0004 \\ 0.2064 & 0.6183 & 1.0006 \\ 0.1214 & 0.5269 & 1.0008 \\ 0.0715 & 0.4489 & 1.0011 \end{bmatrix}, \quad U^{(4)} = \begin{bmatrix} 1.0000 & 1.0000 & 1.0000 \\ 0.4949 & 0.8078 & 1.0003 \\ 0.2463 & 0.6521 & 1.0006 \\ 0.1211 & 0.5269 & 1.0008 \\ 0.0596 & 0.4256 & 1.0011 \end{bmatrix},$$

$$U^{(5)} = \begin{bmatrix} 1.0000 & 1.0000 & 1.0000 \\ 0.4161 & 0.7656 & 1.0003 \\ 0.1730 & 0.5862 & 1.0007 \\ 0.0711 & 0.4489 & 1.0011 \end{bmatrix}.$$

**Example 2.3.5.** As in Theorem 2.2.2, we have shown that if the tensor to be approximated is sufficiently close to a rank-$r$ tensor, then the computed rank-$r$ approximation $\mathcal{X}^{gp}$ is quasi-optimal. It can be further improved to a better approximation $\mathcal{X}^{opt}$ by solving the nonlinear optimization (2.2.5). In this example, we explore the numerical performance of Algorithms 2.2.1 and 2.2.3 for computing low-rank tensor approximations. For the given dimensions $n_1, \ldots, n_m$, we generate the tensor

$$\mathcal{R} = \sum_{s=1}^{r} u^{s,1} \otimes u^{s,2} \otimes \cdots \otimes u^{s,m},$$

where each $u^{s,j} \in \mathbb{C}^{n_j}$ is a complex vector whose real and imaginary parts are generated randomly, obeying the Gaussian distribution. We perturb $\mathcal{R}$ by another tensor $\mathcal{E}$, whose entries are also generated with the Gaussian distribution. We scale the perturbing tensor $\mathcal{E}$ to have a desired norm $\epsilon$. The tensor $\mathcal{F}$ is then generated as

$$\mathcal{F} = \mathcal{R} + \mathcal{E}.$$

We choose $\epsilon$ to be one of $10^{-2}, 10^{-4}, 10^{-6}$, and use the relative errors

$$\rho\_\text{gp} = \frac{\|\mathcal{F} - \mathcal{X}^{gp}\|}{\|\mathcal{E}\|}, \quad \rho\_\text{opt} = \frac{\|\mathcal{F} - \mathcal{X}^{opt}\|}{\|\mathcal{E}\|}$$

to measure the approximation quality of $\mathcal{X}^{\text{gp}}$ , $\mathcal{X}^{\text{opt}}$ respectively. For each case of $(n_1, \ldots, n_m), r$ and $\epsilon$, we generate 10 random instances of $\mathcal{R}, \mathcal{F}, \mathcal{E}$. For the case $(n_1, \ldots, n_m) = (20, 20, 20, 20, 10)$, Algorithm 2.2.3 is used to compute $\mathcal{X}^{gp}$. All other cases are solved by Algorithm 2.2.1. The computational results are reported in Tables 2.1. For each case of $(n_1, \ldots, n_m)$ and $r$, we also list the median of above relative errors and the average CPU time (in seconds). The t_gp and t_opt denote the average CPU time (in seconds) for Algorithms 2.2.1/2.2.3 and for solving (2.2.5) respectively.

In the following, we give a comparison with the generalized eigenvalue decomposition (GEVD) method, which is a classical one for computing tensor decompositions when the rank $r \leqslant n_2$. We refer to [37, 52] for the work about the GEVD method. Consider a cubic order tensor $\mathcal{F} \in \mathbb{C}^{n_1 \times n_2 \times n_3}$ with $n_1 \geqslant n_2 \geqslant n_3$. Suppose $\mathcal{F} = U^{(1)} \circ U^{(2)} \circ U^{(3)}$ is a rank-$r$ decomposition and $r \leqslant n_2$. Assume its first and second decomposing matrices $U^{(1)}, U^{(2)}$ have full column ranks and the third decomposing matrix $U^{(3)}$ does not have colinear columns. Denote the slice matrices

$$F_1 \coloneqq \mathcal{F}_{1:r,1:r,1}, \quad F_2 \coloneqq \mathcal{F}_{1:r,1:r,2}. \tag{2.3.1}$$

One can show that

$$F_1 = U^{(1)}_{1:r,:} \cdot \text{diag}(U^{(3)}_{1,:}) \cdot (U^{(2)}_{1:r,:})^T, \quad F_2 = U^{(1)}_{1:r,:} \cdot \text{diag}(U^{(3)}_{2,:}) \cdot (U^{(2)}_{1:r,:})^T. \tag{2.3.2}$$

This implies that the columns of $(U^{(1)}_{1:r,r})^{-T}$ are generalized eigenvectors of the matrix pair $(F_1^T, F_2^T)$. Consider the transformed tensor

$$\hat{\mathcal{F}} = (U^{(1)}_{1:r,r})^{-1} \times_1 \mathcal{F}_{1:r,:,:}, \tag{2.3.3}$$

For each $s = 1, \ldots, r$, the slice $\hat{\mathcal{F}}_{s,:,:} = U^{(2)}_{:,s} \cdot (U^{(3)}_{:,s})^T$ is a rank-1 matrix. The matrices $U^{(2)}$, $U^{(3)}$ can be obtained by computing rank-1 decompositions for the slices $\hat{\mathcal{F}}_{s,:,:}$. After this is done, we can solve the linear system

$$U^{(1)} \circ U^{(2)} \circ U^{(3)} = \mathcal{F} \tag{2.3.4}$$

to get the matrix $U^{(1)}$. The following is the GEVD method for computing cubic order tensor decompositions when the rank $r \leqslant n_2$.

Table 2.1: Computational performance of Algorithms 2.2.1 and 2.2.3 and of nonlinear optimization (2.2.5).

| $r$ | $\epsilon$ | $\rho$_gp | t_gp | $\rho$_opt | t_opt | $r$ | $\epsilon$ | $\rho$_gp | t_gp | $\rho$_opt | t_opt |
|---|---|---|---|---|---|---|---|---|---|---|---|
| $(n_1,n_2,n_3)$=(50,50,50) | | | | | | $(n_1,n_2,n_3)$=(60, 50, 40) | | | | | |
| 10 | $10^{-2}$ | 1.63 | 0.08 | 0.99 | 1.57 | 15 | $10^{-2}$ | 17.49 | 0.19 | 0.99 | 2.17 |
| | $10^{-4}$ | 6.32 | 0.10 | 0.99 | 1.16 | | $10^{-4}$ | 10.80 | 0.15 | 0.99 | 1.36 |
| | $10^{-6}$ | 3.84 | 0.09 | 0.99 | 0.83 | | $10^{-6}$ | 5.16 | 0.20 | 0.99 | 1.10 |
| 20 | $10^{-2}$ | 25.83 | 0.29 | 0.99 | 2.99 | 30 | $10^{-2}$ | 28.70 | 0.40 | 0.98 | 6.95 |
| | $10^{-4}$ | 5.41 | 0.28 | 0.99 | 1.99 | | $10^{-4}$ | 15.77 | 0.37 | 0.98 | 3.61 |
| | $10^{-6}$ | 30.41 | 0.29 | 0.99 | 1.49 | | $10^{-6}$ | 50.96 | 0.37 | 0.98 | 2.27 |
| 30 | $10^{-2}$ | 27.91 | 0.50 | 0.98 | 7.08 | 45 | $10^{-2}$ | 35.48 | 0.61 | 0.97 | 25.73 |
| | $10^{-4}$ | 213.82 | 0.43 | 0.98 | 3.73 | | $10^{-4}$ | 35.03 | 0.63 | 0.97 | 8.08 |
| | $10^{-6}$ | 17.97 | 0.47 | 0.98 | 2.20 | | $10^{-6}$ | 34.67 | 0.61 | 0.97 | 5.69 |
| $(n_1,n_2,n_3)$=(100,100,100) | | | | | | $(n_1,n_2,n_3)$=(150,150,150) | | | | | |
| 20 | $10^{-2}$ | 11.21 | 0.86 | 1.00 | 6.36 | 30 | $10^{-2}$ | 8.59 | 2.92 | 1.00 | 17.17 |
| | $10^{-4}$ | 3.48 | 0.85 | 1.00 | 4.24 | | $10^{-4}$ | 3.18 | 3.05 | 1.00 | 11.20 |
| | $10^{-6}$ | 3.88 | 0.83 | 1.00 | 3.20 | | $10^{-6}$ | 4.24 | 3.42 | 1.00 | 11.75 |
| 40 | $10^{-2}$ | 24.17 | 1.76 | 0.99 | 17.80 | 60 | $10^{-2}$ | 49.80 | 6.04 | 1.00 | 87.31 |
| | $10^{-4}$ | 11.60 | 1.65 | 0.99 | 11.02 | | $10^{-4}$ | 13.77 | 5.89 | 1.00 | 24.96 |
| | $10^{-6}$ | 11.09 | 1.61 | 0.99 | 7.97 | | $10^{-6}$ | 17.49 | 6.07 | 1.00 | 18.81 |
| 60 | $10^{-2}$ | 18.71 | 3.40 | 0.99 | 28.16 | 90 | $10^{-2}$ | 29.44 | 10.64 | 0.99 | 98.78 |
| | $10^{-4}$ | 26.28 | 3.41 | 0.99 | 17.25 | | $10^{-4}$ | 152.49 | 10.53 | 0.99 | 43.58 |
| | $10^{-6}$ | 19.12 | 3.49 | 0.99 | 13.14 | | $10^{-6}$ | 17.01 | 10.06 | 0.99 | 26.98 |
| $(n_1,n_2,n_3,n_4)$=(20,20,20,20,10) | | | | | | $(n_1,n_2,n_3,n_4)$=(60,50,40,30) | | | | | |
| 24 | $10^{-2}$ | 37.93 | 0.88 | 1.00 | 45.56 | 20 | $10^{-2}$ | 31.42 | 2.78 | 1.00 | 31.16 |
| | $10^{-4}$ | 9.10 | 0.92 | 1.00 | 15.86 | | $10^{-4}$ | 1.17 | 2.76 | 1.00 | 9.39 |
| | $10^{-6}$ | 715.16 | 0.91 | 1.00 | 15.63 | | $10^{-6}$ | 4.14 | 2.79 | 1.00 | 9.48 |
| 48 | $10^{-2}$ | 166.00 | 1.95 | 1.00 | 270.56 | 40 | $10^{-2}$ | 6.99 | 7.52 | 1.00 | 31.81 |
| | $10^{-4}$ | 161.62 | 1.93 | 1.00 | 40.63 | | $10^{-4}$ | 2.58 | 7.32 | 1.00 | 20.07 |
| | $10^{-6}$ | 52.01 | 1.93 | 1.00 | 21.71 | | $10^{-6}$ | 2.49 | 7.22 | 1.00 | 20.22 |
| 72 | $10^{-2}$ | 73.70 | 3.10 | 1.00 | 102.90 | 60 | $10^{-2}$ | 11.48 | 9.83 | 1.00 | 48.08 |
| | $10^{-4}$ | 113.13 | 3.06 | 1.00 | 70.13 | | $10^{-4}$ | 6.38 | 9.80 | 1.00 | 38.97 |
| | $10^{-6}$ | 34.28 | 3.03 | 1.00 | 36.72 | | $10^{-6}$ | 16.35 | 9.76 | 1.00 | 30.38 |

**Algorithm 2.3.6.** (The GEVD method.)

**Input** A tensor $\mathcal{F} \in \mathbb{C}^{n_1 \times n_2 \times n_3}$ with the rank $r \leqslant n_2$.

**1.** Formulate the tensor $\hat{\mathcal{F}}$ as in (2.3.3).

**2.** For $s = 1, \ldots, r$, compute $U^{(2)}_{:,s}$, $U^{(3)}_{:,s}$ from the rank-1 decomposition of the matrix $\hat{\mathcal{F}}_{s,:,:}$.

**3.** Solve the linear system (2.3.4) to get $U^{(1)}$.

**Output** The decomposing matrices $U^{(1)}, U^{(2)}, U^{(3)}$.

We compare the performance of Algorithm 2.1.2 and Algorithm 2.3.6 for randomly generated tensors with the rank $r \leqslant n_2$. We generate $\mathcal{F} = U^{(1)} \circ U^{(2)} \circ U^{(3)}$ such that each $U^{(i)} \in \mathbb{C}^{n_i \times r}$. The entries of $U^{(i)}$ are randomly generated complex numbers. Their real and imaginary parts are randomly generated, obeying the Gaussian distribution. For each case of $(n_1, ..., n_m)$ and $r$, we generate 20 random instances of $\mathcal{F}$. Algorithm 2.3.6 is implemented by the function *cpd_gevd* in the software `Tensorlab`. All the tensor decompositions are computed correctly by both methods. The average CPU time (in seconds) for Algorithm 2.1.2 is denoted as `time-gp`, while the average CPU time for the GEVD method is denoted as `time-gevd`. The computational results are reported in Table 2.2. The numerical experiments show that Algorithm 2.1.2 is more computationally efficient than Algorithm 2.3.6.

Table 2.2: A comparison for the performance of Algorithms 2.1.2 and 2.3.6.

| $(n_1,n_2,n_3)$ | $r$ | `time-gevd` | `time-gp` |
|---|---|---|---|
| (40,30,30) | 30 | 0.91 | 0.29 |
| (50,50,50) | 50 | 4.77 | 0.85 |
| (100,100,100) | 80 | 12.17 | 5.54 |
| (150,150,150) | 100 | 79.85 | 13.30 |
| (200,200,200) | 120 | 161.83 | 25.71 |
| (250,250,250) | 140 | 285.03 | 55.71 |
| (300,300,300) | 100 | 306.64 | 61.38 |
| (400,400,400) | 180 | 934.15 | 271.21 |
| (500,500,500) | 200 | 1688.98 | 539.75 |

## 2.4 Conclusions

.

This Chapter gives computational methods for computing low-rank tensor decompositions and approximations. The proposed methods are based on generating polynomials. For a generic tensor of rank $r \leqslant \min(n_1, N_3)$, its tensor decomposition can be obtained by Algorithm 2.1.2 . Under some general assumptions, we show that if a tensor is sufficiently close to a low-rank one, then the low-rank approximating tensor produced by Algorithm 2.2.1 is quasi-optimal. Numerical experiments are presented to show the efficiency of the proposed methods.

Chapter 2 in full, has been submitted for publication. The thesis author is the coauthor of the preprint "J. Nie, L. Wang, and Z. Zheng. Low Rank Tensor Decompositions And Approximations, 2022. arXiv: 2208.07477."

# Chapter 3

# Find the Generating Matrices and Tensor Decompositions

In Algorithm 2.1.2, the most important step is to find the generating matrix $G$. It requires to solve the linear system

$$A[\mathcal{F}, j](M^{j,k}[G])^T = B(\mathcal{F}, j, k). \tag{3.0.1}$$

When the linear system (3.0.1) has a unique solution, the solution must be the generating matrix we are looking for. However, finding $G$ will be much harder if the system is underdetermined. In such case, we will solve for a $G$ such that matrices $M^{j,k}[G]$'s are commuting.

For the convenience of discussion, we only consider order-3 tensors in the section. Let $\mathcal{F} \in \mathbb{C}^{n_1 \times n_2 \times n_3}$ be a rank-$r$ tensor with the rank decomposition

$$\mathcal{F} = \sum_{s=1}^{r} \mathbf{u}^{s,1} \otimes \mathbf{u}^{s,2} \otimes \mathbf{u}^{s,m}.$$

Recall that we assume $n_1 \geq n_2 \geq n_3$ throughout this chapter.

## 3.1 Case: $n_1 \geq r > n_2$

When $r > n_2$, the linear systems in (3.0.1) are singular for every $j, k$. Thus, we are not able to obtain $M^{j,k}[G]$ by solving (3.0.1). In such case, we have to solve for a matrix $G$ such that matrices $M^{j,k}[G]$'s are simultaneously diagonalizable. Generally, if $M^{j,k}[G]$'s commute, then they are simultaneously diagonalizable. It

enforces the following quadratic equations

$$M^{j_1,k_1}[G]M^{j_2,k_2}[G] = M^{j_2,k_2}[G]M^{j_1,k_1}[G]. \tag{3.1.1}$$

There exist algebraic and numerical algorithms [12, 55] to solve quadratic equations [16, 29, 62]. But, those algorithms are generally much more difficult and expensive than solving linear systems. We observe that there still exist linear relations inside the quadratic equations (3.1.1). With additional linear equations in (3.1.1), we are still able to obtain matrices $\{M^{3,k}[G]\}_{k=2}^{n_3}$ by solely solving linear systems.

In following discussion, $M^{3,k}[G]$ is abbreviated as $M^{3,k}$. We denote $F_k := \mathcal{F}_{1:r,:,k} \in \mathbb{C}^{r\times n_2}$. The linear systems in (3.0.1) for $j = 3$ can be rewritten as

$$M^{3,k}F_1 = F_k.$$

There exists a matrix $C \in \mathbb{C}^{r\times(r-n_2)}$ such that $F := [F_1, C] \in \mathbb{C}^{r\times r}$ is nonsingular. Let $P_k := M^{3,k}C \in \mathbb{C}^{r\times(r-n_2)}$, then we have

$$M^{3,k}F = [F_k, P_k],$$

which is equivalent to

$$M^{3,k} = [F_k, P_k]F^{-1}.$$

Thus, commuting equations in (3.1.1) for $j_1 = j_2 = 3$ are

$$\begin{aligned} & M^{3,i}M^{3,j} - M^{3,j}M^{3,i} \\ = \;& [F_i, P_i]F^{-1}[F_j, P_j]F^{-1} - [F_j, P_j]F^{-1}[F_i, P_i]F^{-1} \\ = \;& 0. \end{aligned}$$

Since $F^{-1}$ is nonsingluar, we have

$$\begin{aligned} & [F_i, P_i]F^{-1}[F_j, P_j] - [F_j, P_j]F^{-1}[F_i, P_i] \\ = \;& [F_i, P_i][F^{-1}F_j, F^{-1}P_j] - [F_j, P_j][F^{-1}F_i, F^{-1}P_i] \\ = \;& 0. \end{aligned}$$

It directly implies $[F_i, P_i]F^{-1}F_j - [F_j, P_j]F^{-1}F_i = 0$. By writing $F^{-1}F_k = \begin{pmatrix} (F_k^1)^T \\ (F_k^2)^T \end{pmatrix}$, we further have

$$P_i(F_j^2)^T - P_j(F_i^2)^T = F_jF_i^1 - F_iF_j^1. \tag{3.1.2}$$

Thus, equations (3.1.2) can be written as the linear system

$$\mathscr{A}(\mathcal{F})X = \mathscr{B}(\hat{\mathcal{F}}), \tag{3.1.3}$$

where

$$\begin{aligned}
\mathscr{A}(\mathcal{F}) &:= \begin{pmatrix}
F_3^2 & -F_2^2 & 0 & \cdots & 0 \\
F_4^2 & 0 & -F_2^2 & \cdots & 0 \\
\vdots & \vdots & \vdots & \ddots & \vdots \\
-F_{n_3}^2 & 0 & 0 & \cdots & -F_2^2 \\
0 & F_4^2 & -F_3^2 & \cdots & 0 \\
\vdots & \vdots & \vdots & \ddots & \vdots
\end{pmatrix} \in \mathbb{C}^{\frac{n_2(n_3-1)(n_3-2)}{2}\times(r-n_2)(n_3-1)}, \\
X &:= \begin{pmatrix} P_2^T \\ P_3^T \\ \vdots \\ P_{n_3}^T \end{pmatrix}, \quad \mathscr{B}(\hat{\mathcal{F}}) := \begin{pmatrix} (F_2F_3^1 - F_3F_2^1)^T \\ (F_2F_4^1 - F_4F_2^1)^T \\ \vdots \\ (F_{n_3-1}F_{n_3}^1 - F_{n_3}F_{n_3-1}^1)^T \end{pmatrix}.
\end{aligned} \tag{3.1.4}$$

Suppose that the matrix $\mathscr{A}(\mathcal{F})$ has full column rank, then the system (3.1.3) has the unique solution $P_i$'s. Thus, the matrices $M^{3,k}$'s are uniquely determined by

$$M^{3,k} = [F_k, P_k]F^{-1}. \tag{3.1.5}$$

Finally, we apply Algorithm 2.1.2 to construct the decomposition of $\mathcal{F}$. The whole algorithm is summarized in Algorithm 3.1.1.

**Algorithm 3.1.1.** Case $n_1 \geq r > n_2$.
**Input:** The tensor $\mathcal{F} \in \mathbb{C}^{n_1\times n_2\times n_3}$ with rank $n_2 < r \leq \min(n_1, \lfloor \frac{n_2}{2} \rfloor n_3)$.

Step 1 Solve the linear system (3.1.3) to get matrices $\{P_k\}_{k=2}^{n_3}$ and obtain $\{M^{3,k}[G]\}_{k=2}^{n_3}$ through (3.1.5).

Step 2 Apply Algorithm 2.1.2 with matrices $\{M^{3,k}[G]\}_{k=2}^{n_3}$ to find decomposition of $\mathcal{F}$.

**Output:** The decomposition of $\mathcal{F}$ obtained from Algorithm 2.1.2.

In Algorithm 3.1.1, we assume that the matrix $\mathscr{A}(\mathcal{F})$ has full column rank when $r \leq \lfloor \frac{n_2}{2} \rfloor n_3$. We can show that this assumption is satisfied generically.

**Theorem 3.1.2.** *Let $\mathcal{F} \in \mathbb{C}^{n_1 \times n_2 \times n_3}$ be a rank-r tensor with the rank decomposition*

$$\mathcal{F} = \sum_{s=1}^{r} u^{s,1} \otimes u^{s,2} \otimes u^{s,m},$$

*where $n_1 \geq r > n_2 \geq n_3$ and $r \leq \lfloor \frac{n_2}{2} \rfloor n_3$, then the matrix $\mathscr{A}(\mathcal{F})$ has full column rank for generic vectors $\{u^{s,1}, u^{s,2}, u^{s,3}\}_{s=1}^{r}$.*

*Proof.* By the construction of the matrix $\mathscr{A}(\mathcal{F})$, we know each entry of $\mathscr{A}(\mathcal{F})$ is a rational polynomial in terms of vectors $\{u^{s,1}, u^{s,2}, u^{s,3}\}_{s=1}^{r}$. Thus, it suffices to prove that there exist some vectors $\{u^{s,1}, u^{s,2}, u^{s,3}\}_{s=1}^{r}$ such that the corresponding $\mathscr{A}(\mathcal{F})$ has full column rank. For convenience, we denote $U^i = [u^{1,i}, \ldots, u^{r,i}]$. We choose the vectors such that $(t = r - n_2)$

$$U^1_{1:r,:} = I_r,\ U^2 = [I_{n_2}, v_1, \ldots, v_t],\ U^3_{1,:} = e.$$

It holds that $F_k = (\mathcal{F})_{1:r,:,k} = D_k (U^2)^T$, where $D_k = \mathrm{diag}(U^3_{k,:})$. Then we have $D_1 = I_r$ and hence $F_1 = (U^2)^T$. We denote that $F^{-1} = \begin{pmatrix} Z^T \\ S^T \end{pmatrix}$ for $S \in \mathbb{R}^{r \times (r-n_2)}$. It holds that

$$F^{-1} F_k = \begin{pmatrix} Z^T \\ S^T \end{pmatrix} F_k = \begin{pmatrix} Z^T D_k (U^2)^T \\ S^T D_k (U^2)^T \end{pmatrix} = \begin{pmatrix} F_k^1 \\ F_k^2 \end{pmatrix}.$$

Thus, we have $(F_k^2)^T = U^2 D_k S$. Furthermore,

$$F^{-1} F = \begin{pmatrix} Z^T \\ S^T \end{pmatrix} [(U^2)^T, C] = I_r \Rightarrow S^T (U^2)^T = U^2 S = 0.$$

Thus, the columns of $S$ belong to the null space of $U^2$. The spaces $col(S)$ and $null(U^2)$ both have the dimension $r - n_2$, so $col(S) = null(B)$. The null space $null(U^2)$ is spanned by

$$\bar{S} := null(U^2) = \begin{pmatrix} v_1 & \cdots & v_t \\ -e_1 & \cdots & -e_t \end{pmatrix}.$$

There must exist a nonsingluar matrix $W$ such that $\bar{S} = SW$. The matrix $I_{(r-n_2)(n_3-1)} \otimes W$ is nonsingluar, so $\mathscr{A}(\mathcal{F})$ has full column rank if and only if

$\mathscr{A}(\mathcal{F})I_{(r-n_2)(n_3-1)} \otimes W$ has full column rank. Thus, it suffices to consider

$$\mathscr{F} := \begin{pmatrix} U^2D_3\bar{S} & -U^2D_2\bar{S} & 0 & \cdots & 0 \\ U^2D_4\bar{S} & 0 & -U^2D_2\bar{S} & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ U^2D_{n_3}\bar{S} & 0 & 0 & \cdots & -U^2D_2\bar{S} \\ 0 & U^2D_4\bar{S} & -U^2D_3\bar{S} & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \end{pmatrix}.$$

We observe that

$$\begin{aligned} U^2D_k\bar{S} &= [I_{n_2}, v_1, v_2, \ldots, v_t]D_k \begin{pmatrix} v_1 & \cdots & v_t \\ -e_1 & \cdots & -e_t \end{pmatrix} \\ &= [(D_k)_{1:n_2,1:n_2}, (D_k)_{n_2+1}v_1, (P_k)_{n_2+2}v_2, \ldots, (P_k)_{n_2+t}v_t] \begin{pmatrix} v_1 & \cdots & v_t \\ -e_1 & \cdots & -e_t \end{pmatrix} \\ &= [(D_k)_{1:n_2,1:n_2}v_1 - (D_k)_{n_2+1}v_1, \cdots, (D_k)_{1:n_2,1:n_2}v_t - (D_k)_{n_2+t}v_t]. \end{aligned}$$

Next, we will prove existence of vectors $\{u^{s,1}, u^{s,2}, u^{s,3}\}_{s=1}^r$ by induction.

**Base case:** We first show the base case that $n_2 = 2, r = \frac{n_2n_3}{2} = n_3$. Let

$$U^3 = \begin{pmatrix} e & 1 \\ I_{r-1} & e_1 + e_2 \end{pmatrix}.$$

By direct computation, we find that ($t = r - n_2 = r - 2$)

$$U^2D_2\bar{S} = \begin{pmatrix} (v_1)_1 & \cdots & (v_{t-1})_1 & 2(v_t)_1 \\ 0 & \cdots & 0 & (v_t)_2 \end{pmatrix},$$

$$U^2D_3\bar{S} = \begin{pmatrix} 0 & \cdots & 0 & (v_t)_1 \\ (v_1)_2 & \cdots & (v_{t-1})_2 & 2(v_t)_2 \end{pmatrix},$$

$$(U^2D_k\bar{S})_{:,j} = \begin{cases} v_{k-3} & j = k-3 \\ 0 & j \neq k-3 \end{cases} \quad \text{for } k \geq 4.$$

The corresponding $\mathscr{F}$ has full column rank if and only if the following system only has the singular solution

$$U^2D_i\bar{S}p_j - U^2D_j\bar{S}p_i = 0, 2 \leq i < j \leq n_3, \tag{3.1.6}$$

where $p_k \in \mathbb{R}^t$. When $4 \leq i < j \leq n_3$, we have

$$U^2D_i\bar{S}p_j - U^2D_j\bar{S}p_i = (p_j)_{i-3}v_{i-3} - (p_i)_{j-3}v_{j-3} = 0 \Rightarrow (p_j)_{i-3} = (p_i)_{j-3} = 0.$$

Therefore, when $4 \leq i \leq n_3$, $(p_i)_k = 0$ for $1 \leq k \leq t-1, k \neq i-3$. Then, for $i \geq 4$, it holds that

$$U^2 D_2 \bar{S} p_i - U^2 D_i \bar{S} p_2 = (p_i)_{i-3} \begin{pmatrix} (v_{i-3})_1 \\ 0 \end{pmatrix} + (p_i)_t \begin{pmatrix} 2(v_t)_1 \\ (v_t)_2 \end{pmatrix} - (p_2)_{i-3} v_{i-3} = 0, \tag{3.1.7}$$

$$U^2 D_3 \bar{S} p_i - U^2 D_i \bar{S} p_3 = (p_i)_{i-3} \begin{pmatrix} 0 \\ (v_{i-3})_2 \end{pmatrix} + (p_i)_t \begin{pmatrix} (v_t)_1 \\ 2(v_t)_2 \end{pmatrix} - (p_3)_{i-3} v_{i-3} = 0. \tag{3.1.8}$$

By adding above two equations, we get

$$((p_i)_{i-3} - (p_2)_{i-3} - (p_3)_{i-3}) v_{i-3} + 3(p_i)_t v_t = 0 \Rightarrow (p_i)_t = 0,\ 4 \leq i \leq n_3.$$

Plugging in $(p_i)_t = 0$ back into (3.1.7) and (3.1.8), we will have $(p_i)_{i-3} = (p_2)_{i-3} = (p_3)_{i-3} = 0$ for $4 \leq i \leq n_3$. Then,

$$U^2 D_3 \bar{S} p_2 - U^2 D_2 \bar{S} p_3 = (p_2)_t \begin{pmatrix} (v_t)_1 \\ 2(v_t)_2 \end{pmatrix} + (p_3)_t \begin{pmatrix} 2(v_t)_1 \\ (v_t)_2 \end{pmatrix} = 0 \Rightarrow (p_2)_t = (p_3)_t = 0.$$

Summarizing everything above, we know (3.1.6) only has the singular solution and it implies that $\mathscr{F}$ has full column rank for $n_2 = 2, r = \frac{n_2 n_3}{2} = n_3$.

**Inductive step:** Suppose that $\mathcal{F}_1 \in \mathbb{C}^{n_1 \times n_2 \times n_3}$ has rank $r_1 = \frac{n_2 n_3}{2} \leq n_1$ with decomposing matrices $U_1^2, U_1^3$ such that $\mathscr{F}_1$ has full column rank and $\mathcal{F}_2 \in \mathbb{C}^{n_1 \times m_2 \times n_3}$ has rank $r_2 = \frac{m_2 n_3}{2} \leq n_1$ with decomposing matrices $U_2^2, U_2^3$ such that $\mathscr{F}_2$ has full column rank. For the dimension $(n_1, n_2 + m_2, n_3)$ with rank $r = \frac{n_2 n_3}{2} + \frac{m_2 n_3}{2} \leq n_1$, we consider

$$U^2 = \begin{pmatrix} U_1^2 & 0 \\ 0 & U_2^2 \end{pmatrix}, U^3 = \begin{pmatrix} U_1^3 & U_2^3 \end{pmatrix}.$$

$\bar{S}$ can be chosen as

$$\bar{S} = \begin{pmatrix} \bar{S}_1 & 0 \\ 0 & \bar{S}_2 \end{pmatrix}.$$

Then, we will have

$$U^2 D_k \bar{S} = \begin{pmatrix} U_1^2 & 0 \\ 0 & U_2^2 \end{pmatrix} \begin{pmatrix} (D_k)_1 & 0 \\ 0 & (D_k)_2 \end{pmatrix} \begin{pmatrix} \bar{S}_1 & 0 \\ 0 & \bar{S}_2 \end{pmatrix} = \begin{pmatrix} U_1^2 (D_k)_1 \bar{S}_1 & 0 \\ 0 & U_2^2 (D_k)_2 \bar{S}_2 \end{pmatrix}.$$

Therefore, the corresponding matrix $\bar{\mathscr{F}}$ can be rearranged as

$$\begin{pmatrix} \mathscr{F}_1 & 0 \\ 0 & \mathscr{F}_2 \end{pmatrix}.$$

$\mathscr{F}_1, \mathscr{F}_2$ both have full column rank, so $\mathscr{F}$ also has full column rank.

Combining the base case and inductive step, we have the existence for $r \leq \min(n_1, \frac{n_2 n_3}{2})$ when $n_2$ is even. The odd $n_2$ can be reduced to $n_2 - 1$. Therefore, our conclusion holds for $r \leq \min(n_1, \lfloor \frac{n_2}{2} \rfloor n_3)$.

□

**Theorem 3.1.3.** *Suppose that* $\mathcal{F} := \sum_{s=1}^{r} u^{s,1} \otimes u^{s,2} \otimes u^{s,m} \in \mathbb{C}^{n_1 \times n_2 \times n_3}$ *with rank* $\min(n_1, \lfloor \frac{n_2}{2} \rfloor n_3) \geq r > n_2 \geq n_3$*, then Algorithm 3.1.1 can find the decomposition of* $\mathcal{F}$ *for generic* $\{u^{s,1}, u^{s,2}, u^{s,3}\}_{s=1}^{r}$.

*Proof.* The tensor $\mathcal{F}$ satisfies all assumptions of Theorem 3.1.2, so the matrix $\mathscr{A}(\mathcal{F})$ generically has full column rank. Thus, the linear system (3.1.3) has a unique solution and the matrices $M^{3,k}$'s can be recovered by (3.1.5). The conclusion will then follow Theorem 2.1.1. □

## 3.2 An extension to the case $r > n_1$

The condition $r \leq n_1$ is necessary for constructing the generating polynomials and generating matrices. Thus, the previous generating matrix based algorithms cannot be directly applied to the tensor $\mathcal{F} = U^1 \circ U^2 \circ U^3 \in \mathbb{C}^{n_1 \times n_2 \times n_3}$ with rank $r > n_1 \geq n_2 \geq n_3$. In this case, we can first construct a new tensor $\hat{\mathcal{F}} \in \mathbb{C}^{r \times n_2 \times n_3}$ such that $\hat{\mathcal{F}} := \hat{U}^1 \circ U^2 \circ U^3$ and $\hat{U}^1 := \begin{pmatrix} U^1 \\ \tilde{U}^1 \end{pmatrix}$ for some matrix $\tilde{U}^1$. Then construct a new tensor $\mathcal{T} \in \mathbb{C}^{r \times r \times n_3}$ such that $\mathcal{T} := \hat{U}^1 \circ \hat{U}^2 \circ U^3$ and $\hat{U}^2 := \begin{pmatrix} U^2 \\ \tilde{U}^2 \end{pmatrix}$ for some matrix $\tilde{U}^2$. After the new tensor $\mathcal{T}$ is constructed, previous algorithms can be applied to find the decomposition of $\mathcal{T}$ and subsequently the decomposition of $\mathcal{F}$.

The new tensor $\hat{\mathcal{F}}$ should have the decomposition $\hat{\mathcal{F}} := \hat{U}^1 \circ U^2 \circ U^3$, so it holds that

$$\hat{\mathcal{F}}_{1:n_1,:,:} = \mathcal{F}.$$

Let $\hat{F}_i := \hat{\mathcal{F}}_{:,:,i} \in \mathbb{C}^{r\times n_2}, F_k := \mathcal{F}_{:,:,k}$ for $i = 1,\dots,n_3$ and $D_k := \mathrm{diag}(U^3_{k,:})$, for $k = 1,...,n_3$. We may choose a matrix $C \in \mathbb{C}^{n_1\times(n_1-n_2)}$ such that $\begin{pmatrix} F_1 & C \end{pmatrix}$ is nonsingular. Denote $E := \begin{pmatrix} F_1 & C \end{pmatrix}^{-1}$ and $E^2 := E_{1+n_2:n_1,:}$. Consider the matrix $[(E^2F_2U^2D_2)^T, (U^2)^T)] \in \mathbb{C}^{r\times n_1}$. Since $r > n_1$, the matrix $\tilde{U}^1 \in \mathbb{C}^{(r-n_1)\times r}$ can be chosen such that

$$\tilde{U}^1[(E^2F_2U^2D_2)^T, (U^2)^T)] = 0. \tag{3.2.1}$$

We denote $C_i = \hat{\mathcal{F}}_{n_1+1:r,:,i} \in \mathbb{C}^{(r-n_1)\times r}, i = 1,\dots,n_3$. In the following, we will discuss how to find all matrices $\{C_i\}_{i=1}^{n_3}$ to construct such $\tilde{\mathcal{F}}$. By (3.2.1), it holds that $C_1 = \tilde{U}^1(U^2)^T D_1 = \tilde{U}^1(U^2)^T = 0$. We require that $\hat{\mathcal{F}}_{1:n_1,:,:} = \mathcal{F}$, so

$$\hat{F}_k = \begin{pmatrix} F_k \\ C_k \end{pmatrix}, \ k = 1,\dots,n_3.$$

Suppose the tensor $\hat{\mathcal{F}}$ satisfies the condition of Theorem 2.0.2, there exists a generating matrix $\hat{G}$ of $\hat{\mathcal{F}}$ such that

$$M^{3,k}[\hat{G}]\hat{F}_1 = \hat{F}_k, \text{ for } k = 2,\dots,n_3.$$

There exists a matrix $\hat{C} \in \mathbb{C}^{r\times(r-n_2)}$ such that $\hat{F} := [\hat{F}_1, \hat{C}] \in \mathbb{C}^{r\times r}$ is nonsingular. The matrix $\hat{C}$ can be

$$\hat{C} = \begin{pmatrix} C & 0 \\ 0 & I_{r-n_1} \end{pmatrix}. \tag{3.2.2}$$

We denote $P_k := M^{3,k}[\hat{G}]\hat{C}$. Then, we have that

$$M^{3,k}[\hat{G}]\hat{F} = \begin{pmatrix} \hat{F}_k & P_k \end{pmatrix} \Leftrightarrow M^{3,k}[\hat{G}] = \begin{pmatrix} \hat{F}_k & P_k \end{pmatrix} \hat{F}^{-1}. \tag{3.2.3}$$

The commuting equations are

$$\begin{aligned}
&M^{3,i}[\hat{G}]M^{3,j}[\hat{G}] - M^{3,j}[\hat{G}]M^{3,i}[\hat{G}] \\
=&\begin{pmatrix} \hat{F}_i & P_i \end{pmatrix} \hat{F}^{-1} \begin{pmatrix} \hat{F}_j & P_j \end{pmatrix} \hat{F}^{-1} - \begin{pmatrix} \hat{F}_j & P_j \end{pmatrix} \hat{F}^{-1} \begin{pmatrix} \hat{F}_i & P_i \end{pmatrix} \hat{F}^{-1} \\
=&(\begin{pmatrix} \hat{F}_i & P_i \end{pmatrix} \hat{F}^{-1} \begin{pmatrix} \hat{F}_j & P_j \end{pmatrix} - \begin{pmatrix} \hat{F}_j & P_j \end{pmatrix} \hat{F}^{-1} \begin{pmatrix} \hat{F}_i & P_i \end{pmatrix})\hat{F}^{-1} \\
=&0.
\end{aligned}$$

Since $\hat{F}$ is nonsingular, the above equation is equivalent to

$$\begin{pmatrix} \hat{F}_i & P_i \end{pmatrix} \hat{F}^{-1} \begin{pmatrix} \hat{F}_j & P_j \end{pmatrix} - \begin{pmatrix} \hat{F}_j & P_j \end{pmatrix} \hat{F}^{-1} \begin{pmatrix} \hat{F}_i & P_i \end{pmatrix} = 0. \tag{3.2.4}$$

It implies

$$\begin{pmatrix} \hat{F}_i & P_i \end{pmatrix} \hat{F}^{-1} \hat{F}_j - \begin{pmatrix} \hat{F}_j & P_j \end{pmatrix} \hat{F}^{-1} \hat{F}_i = 0. \tag{3.2.5}$$

By writing $P_i = \begin{pmatrix} P_i^1 \\ P_i^2 \end{pmatrix}$ for $P_i^1 \in \mathbb{C}^{n_1 \times (r-n_2)}, P_i^2 \in \mathbb{C}^{(r-n_1)\times(r-n_2)}$, we have

$$\begin{pmatrix} F_i & P_i^1 \\ C_i & P_i^2 \end{pmatrix} \hat{F}^{-1} \begin{pmatrix} F_j \\ C_j \end{pmatrix} - \begin{pmatrix} F_j & P_j^1 \\ C_j & P_j^2 \end{pmatrix} \hat{F}^{-1} \begin{pmatrix} F_i \\ C_i \end{pmatrix} = 0. \tag{3.2.6}$$

By the construction of $\hat{F}$, we know

$$\hat{F}^{-1} = \begin{pmatrix} E & 0 \\ 0 & I_{r-n_1} \end{pmatrix}. \tag{3.2.7}$$

Consider the first $n_1$ rows of above equation, we have

$$\begin{aligned} &\begin{pmatrix} F_i & (P_i^1)_{:,1:(n_1-n_2)} \end{pmatrix} EF_j + (P_i^1)_{:,1+n_1-n_2:r-n_2} C_j \\ -&\begin{pmatrix} F_j & (P_j^1)_{:,1:(n_1-n_2)} \end{pmatrix} EF_i + (P_j^1)_{:,1:(n_1-n_2)} C_i = 0 \end{aligned}. \tag{3.2.8}$$

The rest rows of this equation are

$$\begin{aligned} &\begin{pmatrix} C_i & (P_i^2)_{:,1:(n_1-n_2)} \end{pmatrix} EF_j + (P_i^2)_{:,1+n_1-n_2:r-n_2} C_j, \\ -&\begin{pmatrix} C_j & (P_j^2)_{:,1:(n_1-n_2)} \end{pmatrix} EF_i + (P_j^2)_{:,1:(n_1-n_2)} C_i = 0. \end{aligned} \tag{3.2.9}$$

We write $E := \begin{pmatrix} E^1 \\ E^2 \end{pmatrix}$ for $E^1 \in \mathbb{C}^{n_2 \times n_1}, E^2 \in \mathbb{C}^{(n_1-n_2)\times n_1}$. After rearranging (3.2.8), we get

$$F_j E^1 F_i - F_i E^1 F_j = P_i^1 \begin{pmatrix} E^2 F_j \\ C_j \end{pmatrix} - P_j^1 \begin{pmatrix} E^2 F_i \\ C_i \end{pmatrix}. \tag{3.2.10}$$

Let $V_{i,j}$ be the row space of $F_j E^1 F_i - F_i E^1 F_j$ for $i \neq j \in \{2, ..., n_3\}$ and $W_s := (\cap_{j=2}^{s-1} V_{s,j}) \cap (\cap_{j=s+1}^{n_3} V_{s,j})$. We prove in Proposition 3.2.2 that $W_s$ is the row space of $\begin{pmatrix} E^2 F_s \\ C_s \end{pmatrix}$ when $r \leq \frac{(3n_3-8)n_2}{2n_3-5}$. Let $B_s \in \mathbb{R}^{(r-n_1)\times n_2}$ be a matrix whose rows form an orthogonal basis of the space $W_s \cap null(E^2 F_s)$. For each $s$, there exists a unique nonsingular matrix $T_s \in \mathbb{R}^{(r-n_2)\times(r-n_2)}$ such that

$$\begin{pmatrix} E^2 F_s \\ C_s \end{pmatrix} = T_s \begin{pmatrix} E^2 F_s \\ B_s \end{pmatrix}. \tag{3.2.11}$$

By equations (3.2.11), the equation (3.2.10) is equivalent to

$$F_j E^1 F_i - F_i E^1 F_j = P_i^1 T_j \begin{pmatrix} E^2 F_j \\ B_j \end{pmatrix} - P_j^1 T_i \begin{pmatrix} E^2 F_i \\ B_i \end{pmatrix}. \tag{3.2.12}$$

For convenience, we denote $X_{ij} = P_i^1 T_j$, $Y_{ij} = P_j^1 T_i$. The equation (3.2.12) is linear in $X_{ij}, Y_{ij}$, so we can solve for $X_{ij}, Y_{ij}$ by (3.2.12).

The matrix $C_2$ can be written as $C_2 = \tilde{U}^1 D_2 (U^2)^T$. By (3.2.1), we have

$$E^2 F_2 C_2^T = E^2 F_2 U^2 D_2 (\tilde{U}^1)^T = 0.$$

That implies row space of $C_2$ equal row space of $B_2$ so there exists a nonsingular matrix $H \in \mathbb{C}^{(r-n_1)\times(r-n_1)}$ such that $B_2 = HC_2$. The new matrix $\tilde{U}^{new} = H\tilde{U}^1$ also satisfies the condition (3.2.1). Thus, there exists some $\tilde{U}^1$ such that $H = I_{r-n_1}$. For such $\tilde{U}^1$, it holds that $C_2 = B_2$ and hence $T_2 = I_{r-n_2}$ by (3.2.11). Consequently, $X_{i2} = P_i^1 T_2 = P_i^1$. Afterwards, we have

$$X_{ij} = X_{i2} T_j = X_{ij},\ j = 3, \ldots, n_3. \tag{3.2.13}$$

Therefore, we can obtain $\{T_j\}_{j=3}^{n_3}$ by solving (3.2.13). Finally, the matrices $\{C_s\}_{s=2}^{n_3}$ are computed directly by (3.2.11). The new tensor $\hat{\mathcal{F}} = \hat{U}^1 \circ U^2 \circ U^3 \in \mathbb{C}^{r\times n_2 \times n_3}$ with rank $r$ is such that

$$\hat{\mathcal{F}}_{1:n_1,:,:} = \mathcal{F},\ \hat{\mathcal{F}}_{n_1+1:r,:,s} = C_s,\ s = 1, \ldots, n_3. \tag{3.2.14}$$

After we get $C_i$s, we will be able to get $P_i$s by solving

$$\begin{pmatrix} F_j \\ C_j \end{pmatrix} E^1 F_i - \begin{pmatrix} F_i \\ C_i \end{pmatrix} E^1 F_j = \begin{pmatrix} P_i & P_j \end{pmatrix} \begin{pmatrix} E^2 F_j \\ C_j \\ E^2 F_i \\ C_i \end{pmatrix} \tag{3.2.15}$$

Under the assumption in Proposition 3.2.2, $\begin{pmatrix} E^2 F_j \\ C_j \\ E^2 F_i \\ C_i \end{pmatrix}$ is a full rank matrix $\in$ $\mathbb{C}^{2(r-n_2)\times n_2}$ where $2(r-n_2) \leq n_2$. Therefore (3.2.15) is an overdetermined linear system, it must have a unique solution. Consider the tensor

$$\mathcal{T} = \left(\begin{array}{cc|cc|c|cc} \hat{F} & \hat{C} & \hat{F}_2 & P_2 & \ldots & \hat{F}_{n_3} & P_{n_3} \end{array}\right) \in \mathbb{C}^{r\times r\times n_3} \tag{3.2.16}$$

as in (3.2.3). It is easy to check

$$\mathcal{T}_{1:n_1,1:n_2,:} = \mathcal{F} \text{ and } \mathcal{T} = \hat{U}^1 \circ \hat{U}^2 \circ U^3,$$

for some $\hat{U}^2 = \begin{pmatrix} U^2 & \tilde{U}^2 \end{pmatrix}^T$ and

$$\tilde{U}^2 = ((\hat{U}^1 D_1)^{-1} \hat{C})^T.$$

We can apply Algorithm 2.1.2 on $\mathcal{T}$ to find the decomposition of $\mathcal{F}$.

All steps of the algorithm are summarized in the following.

**Algorithm 3.2.1.** Case $r > n_1$.

**Input:** The tensor $\mathcal{F} \in \mathbb{C}^{n_1 \times n_2 \times n_3}$ with rank $\frac{(3n_3-8)n_2}{2n_3-5} \geq r > n_1$.

Step 1 Select the matrix $\hat{C}$ as in (3.2.2) such that $\hat{F} = [\hat{F}_1, \hat{C}]$ is nonsingluar.

Step 2 Let $E$ be the matrix as in (3.2.7) and $E^1 = E_{1:n_2,:}, E^2 = E_{1+n_2:n_1,:}$. Let $W_s := (\cap_{j=2}^{s-1} V_{s,j}) \cap (\cap_{j=s+1}^{n_3} V_{s,j})$ where $V_{ij}$ is the row space of $F_j E^1 F_i - F_i E^1 F_j$ and $B_s$ be the matrix whose rows form an orthogonal basis of the space $W_s \cap null(E^2 F_s)$.

Step 2 Solve equations (3.2.12) to get matrices $X_{ij} = P_i^1 T_j, Y_{ij} = P_j^1 T_i$ and then solve (3.2.13) to get matrices $T_s, s = 2, \ldots, n_3$. Matrices $C_s, s = 2, \ldots, n_3$ are obtained from (3.2.11).

Step 3 Solve (3.2.15) for $P_s, s = 2, \ldots, n_3$

Step 4 Let tensor $\mathcal{T}$ be the tensor as in (3.2.16). Apply the Algorithm 2.1.2 to find the decomposition of $\mathcal{T} = \hat{U}^1 \circ \hat{U}^2 \circ U^3$.

**Output:** The decomposition of $\mathcal{F}$ is

$$\mathcal{F} = U^1 \circ U^2 \circ U^3, \tag{3.2.17}$$

where $U^1 = \hat{U^1}_{1:n_1,:}$, $U^2 = \hat{U^2}_{1:n_2,:}$.

The construction of $\hat{\mathcal{F}}$ and $\mathcal{T}$ requires that $W_s$ in Step 2 of Algorithm 3.2.1 is the row space of the matrix $\begin{pmatrix} E^2 F_s \\ C_s \end{pmatrix}$. We prove in the following proposition that this requirement is true under mild conditions.

**Proposition 3.2.2.** Suppose that $n_1 < r \leq \frac{(3n_3-8)n_2}{2n_3-5}$, the matrix $[P_i^1, P_j^1]$ has full column rank for $2 \leq i < j \leq n_3$, and the matrix

$$\left((E^2F_2)^T, C_2^T, \ldots, (E^2F_{n_3})^T, C_{n_3}^T\right) \tag{3.2.18}$$

has full Kruskal rank, then the space $W_s = (\cap_{j=2}^{s-1} V_{s,j}) \cap (\cap_{j=s+1}^{n_3} V_{s,j})$ is the row space of $\begin{pmatrix} E^2F_s \\ C_s \end{pmatrix}$ for $s = 2, \ldots, n_3$.

*Proof.* Since the matrix $[P_i^1, P_j^1]$ has full column rank, there exists a nonsingluar matrix $R$ such that

$$R[P_i^1, P_j^1] = \begin{pmatrix} I_{r-n_2} & 0 \\ 0 & I_{r-n_2} \\ 0 & 0 \end{pmatrix}.$$

Row operations do not change the row space, so

$$V_{i,j} = span\{(E^2F_i)^T, (E^2F_j)^T, C_i^T, C_j^T\}.$$

In the following, we will only prove the conclusion for $s = 2$. All other values of $s$ can be proved similarly. Let $S_k := \cap_{j=3}^{k} V_{2,j}$, then $W_s = S_{n_3}$. It is clear that $col[(E^2F_2)^T, C_2^T] \subset S_k$ and $S_k \subset V_{2,3}$ for $k = 3, \ldots, n_3$. Since the matrix in (3.2.18) has full Kruskal rank, it holds that

$$\begin{aligned} \dim S_k \cup V_{2,k+1} &= \min(n_2, \dim S_k + \dim V_{2,k+1} - (r - n_2)) \\ &= \min(n_2, \dim S_k + (r - n_2)). \end{aligned}$$

We also have

$$\dim S_{k+1} = \dim S_k \cap V_{2,k+1} = \dim S_k + \dim V_{2,k+1} - \dim S_k \cup V_{2,k+1}.$$

We assume by contradiction that for all $k = 3, \ldots, n_3 - 1$, it holds $\dim S_k > n_2 - (r - n_2) = 2n_2 - r$, then

$$\begin{aligned} \dim S_{k+1} &= \dim S_k \cap V_{2,k+1} \\ &= \dim S_k + \dim V_{2,k+1} - \dim S_k \cup V_{2,k+1} \\ &= \dim S_k + 2r - 3n_2. \end{aligned}$$

Therefore,

$$\begin{aligned} \dim S_{n_3-1} &= 2(r - n_2) + (n_3 - 4)(2r - 3n_2) \\ &= (2n_3 - 5)r - (3n_3 - 8)n_2 + 2n_2 - r \\ &\leq 2n_2 - r \end{aligned}$$

It contradicts that the assumption that $\dim S_k > 2n_2 - r$ for $k = 3, \ldots, n_3 - 1$. Thus, there must exist some $3 \leq t \leq n_3 - 1$ such that $\dim S_t \leq 2n_2 - r$. For such $t$, $\dim S_t \cup V_{2,t+1} = \dim S_t + (r - n_2)$ and

$$\dim S_{t+1} = \dim S_t + \dim V_{2,t+1} - \dim S_t \cup V_{2,t+1} = r - n_2 = \dim col[(E^2F_2)^T, C_2^T].$$

Since $col[(E^2F_2)^T, C_2^T] \subset S_k$ for $k = 3, \ldots, n_3$ and $S_{n_3} \subset S_{t+1}$, we conclude $W_2 = S_{n_3} = col[(E^2F_2)^T, C_2^T]$. □

Before we show the condition above is generic, we first prove the following lemma.

**Lemma 3.2.3.** *For some positive integers $m_1, m_2, m_3$. For a matrix $B \in \mathbb{C}^{m_2 \times m_3}$ with $krank(B) = r$ and a matrix $A \in \mathbb{C}^{m_1 \times m_2}$ with full rank, $krank(AB) \geq r - dim(null(A) \cap col(B))$.*

*Proof.* Since $krank(B) = r$, there does not exist a nonzero $\alpha = \begin{pmatrix} \alpha_1 & ... & \alpha_r \end{pmatrix}^T \in \mathbb{C}^r$ such that $\alpha_1 B_{:,i_1} + ... + \alpha_r B_{:,i_r} = 0$ for any $\{i_1, ..., i_r\} \subset \{1, ..., m_3\}$. If $p \neq 0$, let $p := dim(null(A) \cap col(B))$, and $v_1, ..., v_p$ be a basis for $null(A) \cap col(B)$ that satisfies

$$krank(\begin{pmatrix} B & v_1 & \ldots & v_p \end{pmatrix}) = r. \tag{3.2.19}$$

It is easy to check such a basis always exists. We prove by contradiction, assuming $krank(AB) < r - dim(null(A) \cap col(B)) = r - p$. Hence, there exists a nonzero $\alpha = \begin{pmatrix} \alpha_1 & ... & \alpha_{r-p} \end{pmatrix}^T \in \mathbb{C}^{r-p}$ such that $\alpha_1 (AB)_{:,i_1} + ... + \alpha_{r-p} (AB)_{:,i_{r-p}} = 0$ for some $\{i_1, ..., i_{r-p}\} \subset \{1, ..., m_3\}$. We have

$$\begin{aligned} \alpha_1 A(B_{:,i_1}) + ... + \alpha_{r-p} A(B_{:,i_{r-p}}) =& 0 \\ \implies A(\alpha_1 B_{:,i_1} + ... + \alpha_{r-p} B_{:,i_{r-p}}) =& 0. \end{aligned}$$

Because $krank(B) = r > r - p$, $\alpha_1 B_{:,i_1} + ... + \alpha_{r-p} B_{:,i_{r-p}} \neq 0$, we have $\alpha_1 B_{:,i_1} + ... + \alpha_{r-p} B_{:,i_{r-p}} \in null(A) \cap col(B)$. That implies

$$\alpha_1 B_{:,i_1} + ... + \alpha_{r-p} B_{:,i_{r-p}} + \alpha_{r-p+1} v_1 + ... + \alpha_r v_p = 0$$

for some $\alpha_{r-p+1}, ..., \alpha_r$. That contradicts the (3.2.19). The case $p = 0$ will be similar. □

We also need a lemma for the Kruskal rank of matrices' Khatri-Rao product.

**Lemma 3.2.4** (Lemma 12 in [7]). *Let $m, n_1, ..., n_m$ be positive integers. For generic $U^1, ..., U^m \in \mathbb{C}^{n_1 \times r}, ..., \mathbb{C}^{n_m \times r}$ respectively, $U^1 \odot, ..., \odot U^m$ has full Kruskal rank.*

**Theorem 3.2.5.** *Suppose that $\mathcal{F} := U^1 \circ U^2 \circ U^3 \in \mathbb{C}^{n_1 \times n_2 \times n_3}$ with rank $n_1 < r \leq \frac{(3n_3-8)n_2}{2n_3-5}$, Let the matrix $[P_i^1, P_j^1]$ and $\left((E^2F_2)^T, C_2^T, \ldots, (E^2F_{n_3})^T, C_{n_3}^T\right)$ for $2 \leq i < j \leq n_3$ as in Proposition 3.2.2. In generic condition, matrix $[P_i^1, P_j^1]$ has full column rank, and the matrix*

$$\left((E^2F_2)^T, C_2^T, \ldots, (E^2F_{n_3})^T, C_{n_3}^T\right)$$

*has full Kruskal rank. Algorithm 3.2.1 finds the decomposition of $\mathcal{F}$.*

*Proof.* We start this proof with the $n_1 = n_2$ case. In this case, all the $E^2F_s$ for $s = 2, ..., n_3$ will not exist, if we can show both the matrices

$$\left(C_2^T, \ldots, C_{n_3}^T\right) \tag{3.2.20}$$

and

$$\left(P_2^1, \ldots, P_{n_3}^1\right) \tag{3.2.21}$$

have full Kruskal rank for a generic condition, then we are able to prove the theorem. Consider the tensor

$$\mathcal{T} = \left(\hat{F} \quad \hat{C} \;\middle|\; \hat{F}_2 \quad P_2 \;\middle|\; ... \;\middle|\; \hat{F}_{n_3} \quad P_{n_3}\right) \in \mathbb{C}^{r \times r \times n_3} \tag{3.2.22}$$

as in (3.2.3). It will have decomposition matrices

$$\mathcal{T} = \begin{pmatrix} U^1 \\ \tilde{U}^1 \end{pmatrix} \circ \begin{pmatrix} U^2 \\ \tilde{U}^2 \end{pmatrix} \circ U^3. \tag{3.2.23}$$

We know

$$C_i = \tilde{U}^1 D_i (U^2)^T \quad P_i^1 = U^1 D_2 (\tilde{U}^2)^T \quad \text{for } i \in \{2, ..., n_3\} \tag{3.2.24}$$

and

$$P_i = M^{3,i}[\hat{G}]\hat{C}. \tag{3.2.25}$$

For our tensor $\mathcal{F}$, by (3.2.25), $P_i^1$s are fully determined by $M^{3,i}[\hat{G}]$s which are fully determined by $C_i$s. $C_i$s are fully determined by $\tilde{U}^1$. But we only require (3.2.1), and there is still some freedom in choosing $\tilde{U}^1$.

To fully determine (3.2.20) and (3.2.21), we denote $b_i \in \mathbb{C}^r$ for $i \in \{1, ..., r\}$ be $r$ random vectors. Let $b_{i_1}...b_{i_{r-n_2}}$ be the first $r-n_2$ vectors that make the matrix

$$\begin{pmatrix} (U^2)^T & b_{i_1} & \dots & b_{i_{r-n_2}} \end{pmatrix} \tag{3.2.26}$$

be of rank $r$. We let

$$(\tilde{U}^1)^T_{j,:} = b_{i+j} - \begin{pmatrix} U^2 \\ b_{i_1}^T \\ \vdots \\ b_{i_{j-1}}^T \end{pmatrix}^T \begin{pmatrix} U^2 \\ b_{i_1}^T \\ \vdots \\ b_{i_{j-1}}^T \end{pmatrix} b_{i+j}. \tag{3.2.27}$$

Then it is easy to check this $\tilde{U}^1$ will satisfy (3.2.1) and is fully determined by $\mathcal{F}$ and $b_i$s. Therefore, (3.2.20) and (3.2.21) are fully determined.

Now each entry of (3.2.20) and (3.2.21) is a rational polynomial in term of randomly generated vectors $b_i$s and vectors $\{u^{s,1}, u^{s,2}, u^{s,3}\}_{s=1}^r$. Thus, it suffices to prove that there exist some vectors $b_i$s and $\{u^{s,1}, u^{s,2}, u^{s,3}\}_{s=1}^r$ such that the corresponding (3.2.20) and (3.2.21) have full Kruskal rank.

By (3.2.24), we have

$$\left(C_2^T, \dots, C_{n_3}^T\right) = U^2((U^3)_{2:n_3,:} \odot \tilde{U}^1)^T, \quad \left(P_2^1, \dots, P_{n_3}^1\right) = U^1((U^3)_{2:n_3,:} \odot \tilde{U}^2)^T.$$

Let $B_1 \in \mathbb{C}^{(r-n_1)\times r}$, $B_2 \in \mathbb{C}^{n_3 \times r}$ be two matrices with

(1) $(B_2)_{1,i} \neq 0$ for $i \in \{1, 2, ..., r\}$,

(2) $B_1 \odot B_2$ has full Kruskal rank,

(3) $(B_1^+)^T \odot B_2$ has full Kruskal rank.

Such two matrices must exist, because $(B_2)_{1,i} \neq 0$ is a generic condition, (2) is also a generic condition by Lemma 3.2.4. Since the Moore-Penrose inverse is unique, (3) is also a generic condition.

Let

$$\tilde{U}^1 = B_1 \odot (B_2)_{1,:}, \quad \tilde{U}^2 = (B_1^+)^T \odot (B_2)_{1,:}, \tag{3.2.28}$$

$$U^3 = B_2 \odot \begin{pmatrix} \frac{1}{(B_2)_{1,1}} & \dots & \frac{1}{(B_2)_{1,r}} \end{pmatrix}. \tag{3.2.29}$$

Since $(B_1^+)^T$ and $B_1$ have the same row space, there must exist some full rank $U^1 = U^2 \in \mathbb{C}^{n_1 \times r}$, such that

$$\tilde{U}^1 (U^2)^T = 0. \tag{3.2.30}$$

We have

$$krank(U^3 \odot \tilde{U}^1) = krank(U^3 \odot \tilde{U}^2) = min(r, n_3(r - n_1)), \tag{3.2.31}$$

and

$$krank((U^3)_{2:n_3,:} \odot \tilde{U}^1) = krank((U^3)_{2:n_3,:} \odot \tilde{U}^2) = min(r, (n_3 - 1)(r - n_1)). \tag{3.2.32}$$

Denote $B = ((U^3)_{2:n_3,:} \odot \tilde{U}^1)^T$ and $A = U^2$. Since $(U^3)_{1,:} \odot \tilde{U}^1 = \tilde{U}^1$ and $col((\tilde{U}^1)^T) = null(U^2)$, then

$$dim(null(A) \cap col(B)) = dim(null(A)) + dim(col(B)) - dim(null(A) \cup col(B)) \tag{3.2.33}$$

$$= r - n_1 + min(r, (n_3 - 1)(r - n_1)) - min(r, n_3(r - n_1)). \tag{3.2.34}$$

$n_3(r - n_1) - (n_3 - 1)(r - n_1) = r - n_1$, therefore, there are 4 possible conditions:

(1) $(n_3 - 1)(r - n_1) \geq r > n_1$,

(2) $n_3(r - n_1) \geq r > (n_3 - 1)(r - n_1) \geq n_1$,

(3) $r > n_3(r - n_1) \geq n_1 > (n_3 - 1)(r - n_1)$,

(4) $r > n_1 > n_3(r - n_1)$.

Under the condition (1), $krank(B) = r$, $A$ is a full rank matrix with $dim(null(A) \cap col(B)) = r - n_1$. Using the Lemma 3.2.3, we have $krank(AB) = r - (r - n_1) = n_1$, $\left(C_2^T, \ldots, C_{n_3}^T\right)$ has full Kruskal rank.

Under the condition (2), $krank(B) = (n_3 - 1)(r - n_1)$, $A$ is a full rank matrix with $dim(null(A) \cap col(B)) = (n_3 - 1)(r - n_1) - n_1$. Using the Lemma 3.2.3, we have $krank(AB) \geq (n_3 - 1)(r - n_1) - ((n_3 - 1)(r - n_1) - n_1) = n_1$, $\left(C_2^T, \ldots, C_{n_3}^T\right)$ has full krank.

Under the condition (3) and (4), $krank(B) = (n_3 - 1)(r - n_1)$, $A$ is a full rank matrix with $dim(null(A) \cap col(B)) = 0$. Using the Lemma 3.2.3, we have $krank(AB) \geq (n_3 - 1)(r - n_1)$, $\left(C_2^T, \ldots, C_{n_3}^T\right)$ has full Kruskal rank.

From (3.2.28), similar as $\left(C_2^T, \ldots, C_{n_3}^T\right)$, $\left(P_2^1, \ldots, P_{n_3}^1\right)$ also has full Kruskal rank. Therefore, based on Proposition 3.2.2, Algorithm 3.2.1 will be able to find its tensor decomposition.

For the $n_1 \neq n_2$ case, similarly as above, each entry of

$$\left((E^2F_2)^T, C_2^T, \ldots, (E^2F_{n_3})^T, C_{n_3}^T\right) \tag{3.2.35}$$

is still a rational polynomial in terms of randomly generated vectors $b_i$s and vectors $\{u^{s,1}, u^{s,2}, u^{s,3}\}_{s=1}^r$. Thus, it suffices to prove that there exist some vectors $b_i$s and $\{u^{s,1}, u^{s,2}, u^{s,3}\}_{s=1}^r$ such that the corresponding (3.2.35) and (3.2.21) have full Kruskal rank. From the $n_1 = n_2$ case, since the rank requirement $r \leq \frac{(3n_3-8)n_2}{2n_3-5}$ is independent with $n_1$, for some tensor $\mathcal{F} \in \mathbb{C}^{n_2 \times n_2 \times n_3}$, there exist a tensor $\hat{\mathcal{F}} \in \mathbb{C}^{n_1 \times n_2 \times n_3}$ such that

$$\hat{\mathcal{F}} = \begin{pmatrix} U^1 \\ \tilde{U}^1 \end{pmatrix} \circ U^2 \circ U^3. \tag{3.2.36}$$

This gives a choice for $\{u^{s,1}, u^{s,2}, u^{s,3}\}_{s=1}^r$. after choose the correct $b_i$s, we will have (3.2.35) for $\hat{\mathcal{F}}$ same as (3.2.20) for $\hat{F}$ and (3.2.21) for $\hat{\mathcal{F}}$ same as (3.2.21) for $\hat{F}$. They both have full Kruskal rank. Therefore, we are able to apply Algorithm 3.2.1 to find its tensor decomposition.

□

**Example 3.2.6.** Let $\mathcal{F} \in \mathbb{C}^{4\times4\times4}$ be a tensor with decomposition matrices

$$U^{(1)} = \begin{pmatrix} 1 & 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 1 & 1 \end{pmatrix} U^{(2)} = \begin{pmatrix} 1 & 1 & 1 & 1 & 1 \\ 0.2 & 0.4 & 0.6 & 0.8 & 1 \\ 0.2^2 & 0.4^2 & 0.6^2 & 0.8^2 & 1 \\ 0.2^3 & 0.4^3 & 0.6^3 & 0.8^3 & 1 \end{pmatrix}$$

$$U^{(3)} = \begin{pmatrix} 1 & 1 & 1 & 1 & 1 \\ 1.2 & 1.4 & 1.6 & 1.8 & 2 \\ 1.2^2 & 1.4^2 & 1.6^2 & 1.8^2 & 2^2 \\ 1.2^3 & 1.4^3 & 1.6^3 & 1.8^3 & 2^3 \end{pmatrix}.$$

It is easy to check those decomposition matrices are of full k-rank. Using Algorithm 3.2.1, we get the constructed tensor $\mathcal{T}$ is

$$\mathcal{T}(:,:,1) = \begin{pmatrix} 2.0000 & 1.2000 & 1.0400 & 1.0080 & 0 \\ 2.0000 & 1.4000 & 1.1600 & 1.0640 & 0 \\ 2.0000 & 1.6000 & 1.3600 & 1.2160 & 0 \\ 2.0000 & 1.8000 & 1.6400 & 1.5120 & 0 \\ 0 & 0 & 0 & 0 & 1.0000 \end{pmatrix},$$

$$\mathcal{T}(:,:,2) = \begin{pmatrix} 3.2000 & 2.2400 & 2.0480 & 2.0096 & 0.0154 \\ 3.4000 & 2.5600 & 2.2240 & 2.0896 & 0.0115 \\ 3.6000 & 2.9600 & 2.5760 & 2.3456 & 0.0077 \\ 3.8000 & 3.4400 & 3.1520 & 2.9216 & 0.0038 \\ -0.0000 & -0.0000 & 0.0000 & 1.0000 & 2.0000 \end{pmatrix},$$

$$\mathcal{T}(:,:,3) = \begin{pmatrix} 5.4400 & 4.2880 & 4.0576 & 4.0115 & 0.0492 \\ 5.9600 & 4.7840 & 4.3136 & 4.1254 & 0.0392 \\ 6.5600 & 5.5360 & 4.9216 & 4.5530 & 0.0276 \\ 7.2400 & 6.5920 & 6.0736 & 5.6589 & 0.0146 \\ -0.0000 & -0.0000 & 1.0000 & 5.0000 & 4.0000 \end{pmatrix},$$

$$\mathcal{T}(:,:,4) = \begin{pmatrix} 9.7280 & 8.3456 & 8.0691 & 8.0138 & 0.1204 \\ 10.7440 & 9.0976 & 8.4390 & 8.1756 & 0.1009 \\ 12.0960 & 10.4576 & 9.4746 & 8.8847 & 0.0750 \\ 13.8320 & 12.6656 & 11.7325 & 10.9860 & 0.0416 \\ -0.0000 & 1.0000 & 6.0000 & 17.6000 & 8.0000 \end{pmatrix}.$$

Apply Algorithm 2.1.2 on $\mathcal{T}$, we get a tensor decomposition with error $1.5782e-10$.

## 3.3 Numerical Experiments

In this section, we present numerical experiments for tensor decomposition and compare it with nonlinear least square method in `Tensorlab`[58]. The experiment is implemented on `MATLAB` R2018b with a Mircosoft Surface Laptop i5-8250U CPU @ 1.60GHz 1.80GHz RAM 8G. Tensors are randomly generated by its decomposition matrix using Matlab's *randn*() function. Its decomposition matrix's entries follow normal distribution.

### 3.3.1 Case 2

This subsection explores the performance of Algorithm 3.1.1.

**Example 3.3.1.** Consider the tensor $\mathcal{F} \in \mathbb{C}^{4\times3\times3}$ such that

$$\mathcal{F}_{i_1,i_2,i_3} = e^{i_1-i_2+i_3} + (\frac{i_1 i_3}{i_2})^{\frac{1}{2}} + \frac{i_1 i_3}{i_2} + (\frac{i_1 i_3}{i_2})^2,$$

for all $i_1, i_2, i_3$ in the corresponding range. The 4 biggest singular values of the flattening matrix $\text{Flat}(\mathcal{F}, 1)$ are

$$720.29, \quad 6.3723, \quad 1.6688, \quad 1.2767 \times 10^{-2}.$$

Applying Algorithm 3.1.1 with rank $r = 2, 3, 4$, we get the approximation errors as follows:

| $r$ | 2 | 3 | 4 |
|---|---|---|---|
| $\|\mathcal{F} - output\|$ | 12.411 | 2.3448 | $2.0520 \times 10^{-12}$ |

For the case $r = 4$, the computed approximating tensor by Algorithm 3.1.1 is $U^{(1)} \circ U^{(2)} \circ U^{(3)}$, with

$$U^{(1)} = \begin{pmatrix} 1 & 1 & 1 & e \\ 4 & \sqrt{2} & 2 & e^2 \\ 9 & \sqrt{3} & 3 & e^3 \\ 16 & \sqrt{4} & 4 & e^4 \end{pmatrix}, U^{(2)} = \begin{pmatrix} 1 & 1 & 1 & 1 \\ \frac{1}{4} & \frac{1}{\sqrt{2}} & \frac{1}{2} & e^{-1} \\ \frac{1}{9} & \frac{1}{\sqrt{3}} & \frac{1}{3} & e^{-2} \\ \frac{1}{16} & \frac{1}{\sqrt{4}} & \frac{1}{4} & e^{-3} \end{pmatrix},$$
$$U^{(3)} = \begin{pmatrix} 1 & 1 & 1 & 1 \\ 4 & \sqrt{2} & 2 & e \\ 9 & \sqrt{3} & 3 & e^2 \end{pmatrix}.$$

**Example 3.3.2.** In this example we explore the performance of Algorithm 3.1.1 for random tensors. We generate $\mathcal{F}$ by decomposition matrices $U^{(i)} \in \mathbb{C}^{n_i \times r}$ with $i = 1, \ldots, n_m$, whose entries are complex numbers. Each entry's real and imaginary parts are randomly drawn i.i.d. from normal distributions with mean $i \in 1, 2, ...m$ and standard deviation 1. Such tensors are expected to have rank $r$. We apply Algorithm 3.1.1 and classic nonlinear least square method with tensor $\mathcal{F}$ and $r$ as input. For each pair $(n_1, \ldots, n_m)$ in Table 3.1 and Table 3.2, we generate 50 instances of random tensors in $\mathbb{C}^{n_1 \times \ldots \times n_m}$. For each pair $(n_1, \ldots, n_m)$, we report the average time (in seconds) consumed by computation and the average error in

frobenius norm over 50 random instances for generating polynomial method(gp) and Nonlinear least square(nls)/Normal Form method(nf) in [56]. The computational results are summarized in Table 3.1 and 3.2. For such tensors, Nonlinear least squares failed to get its decomposition with frobenius error within $10^{-4}$, while our method is faster compared with Normal Form method and can get their rank decompositions efficiently.

Table 3.1: Computational results for rank decompositions of random tensors. Time (in seconds) is the average of the consumed time.

| $(n_1, \ldots, n_m)$ | r | $time_{nls}$ | $error_{nls}$ | $time_{gp}$ | $error_{gp}$ |
|---|---|---|---|---|---|
| (10,7,3) | 9 | 0.083 | NaN | 0.001 | 2.360e-11 |
| (11,9,4) | 11 | 1.324 | NaN | 0.001 | 1.317e-10 |
| (30,15,6) | 30 | 3.876 | NaN | 0.004 | 1.209e-08 |
| (50,20,10) | 50 | 6.178 | NaN | 0.0336 | 1.464e-07 |
| (70,25,10) | 70 | 10.269 | NaN | 0.0758 | 1.230e-07 |
| (90,30,20) | 90 | 17.55 | NaN | 3.333 | 1.953e-07 |
| (120,50,20) | 120 | 30.9 | NaN | 7.392 | 7.813e-07 |
| (150,70,20) | 150 | 50.048 | NaN | 14.162 | 1.485e-06 |

Table 3.2: Computational results for rank decompositions of random tensors for Normal Forms method (denoted by nf) and our method. Time (in seconds) is the average of the consumed time.

| $(n_1, \ldots, n_m)$ | r | $time_{nf}$ | $error_{nf}$ | $time_{gp}$ | $error_{gp}$ |
|---|---|---|---|---|---|
| (10,7,3) | 9 | 0.42986 | 3.5429e-11 | 0.0043221 | 2.8905e-11 |
| (11,9,4) | 11 | 0.85329 | 9.4432e-11 | 0.0011281 | 1.2177e-10 |
| (30,15,6) | 30 | 14.311 | 1.4285e-09 | 0.0098707 | 9.6275e-09 |
| (50,20,10) | 50 | 129.91 | 4.7615e-08 | 0.060555 | 7.742e-08 |
| (70,25,10) | 70 | 329.91 | 1.1512e-08 | 0.12794 | 2.4733e-07 |

**Example 3.3.3.** In this example, we explore the numerical performance of Algorithms 3.1.1 when the tensor is not but close to a rank r tensor. For the given dimensions $n_1, \ldots, n_m$, we generate the tensor

$$\mathcal{R} = \sum_{s=1}^{r} \mathbf{u}^{s,1} \otimes \mathbf{u}^{s,2} \otimes \cdots \otimes \mathbf{u}^{s,m},$$

where each $\mathbf{u}^{s,j} \in \mathbb{C}^{n_j}$ is a complex vector whose real and imaginary parts are generated randomly, obeying the Gaussian distribution with mean $s$. We perturb $\mathcal{R}$ by another tensor $\mathcal{E}$, whose entries are generated with the Gaussian distribution with mean 0 and standard derivation 1. Then we normalize the perturbing tensor

$\mathcal{E}$ to have a desired norm $\epsilon$. The tensor $\mathcal{F}$ is then generated as

$$\mathcal{F} = \mathcal{R} + \mathcal{E}.$$

Denote $\mathcal{X}^{gp}$ be the solution of algorithm 3.1.1, we choose $\epsilon$ to be one of $10^{-3}, 10^{-6}, 10^{-9}$, and use the relative errors

$$\rho = \frac{\|\mathcal{F} - \mathcal{X}^{gp}\|}{\|\mathcal{E}\|},$$

to measure the approximation quality of the approximating tensors $\mathcal{X}^{\text{gp}}$ . For each case of $(n_1, \ldots, n_m), r$ and $\epsilon$, we generate 20 random instances of $\mathcal{R}, \mathcal{F}, \mathcal{E}$. The computational results are reported in Tables 3.3. For each case of $(n_1, \ldots, n_m)$ and $r$, we list the mean of above relative errors.

Table 3.3: Computational results for rank decompositions of random tensors with error $\epsilon$ in Case 2. The mean of relative errors are shown in the table.

| $(n_1, \ldots, n_m)$ | r | $\epsilon = 10^{-3}$ | $\epsilon = 10^{-6}$ | $\epsilon = 10^{-9}$ |
|---|---|---|---|---|
| (9,7,4) | 9 | 21.594 | 17.479 | 16.683 |
| (11,9,4) | 11 | 36.618 | 36.166 | 49.859 |
| (30,15,6) | 30 | 167.51 | 158.89 | 96.773 |
| (50,20,10) | 50 | 63.669 | 59.869 | 60.7 |
| (70,25,10) | 70 | 86.099 | 90.508 | 98.937 |
| (90,30,20) | 90 | 72.643 | 69.27 | 164.49 |
| (120,50,20) | 120 | 78.191 | 77.034 | 302.22 |
| (150,70,20) | 150 | 127.06 | 127.11 | 521.32 |

### 3.3.2 Case 3

This subsection explores the performance of Algorithm 3.2.1.

**Example 3.3.4.** In this example, we explore the performance of Algorithm 3.2.1 for random tensors. We generate $\mathcal{F}$ by decomposition matrices $U^{(i)} \in \mathbb{C}^{n_i \times r}$ with $i = 1, \ldots, m$, whose entries are complex numbers. Each entry's real and imaginary parts are randomly drawn i.i.d. from the normal distributions with mean $i$ and standard deviation 1. Such tensors are expected to have rank $r$. We apply Algorithm 3.2.1 and classic nonlinear least square method (*cpd_nls* provided in the software `Tensorlab`) with tensor $\mathcal{F}$ and $r$ as input. For each pair $(n_1, \ldots, n_m)$ in Table 3.4, we generate 50 instances of random tensors in $\mathbb{C}^{n_1 \times \cdots \times n_m}$. For each pair $(n_1, \ldots, n_m)$, we report the average time (in seconds) consumed by computation and the average error in frobenius norm over 50 random instances for generating

polynomial method(gp) and nonlinear least square(nls). The computational results are summarized in Table 3.4. For such tensors, Nonlinear least squares failed to get its decomposition with frobenius error within $10^{-4}$, while our method can get their rank decompositions efficiently.

Table 3.4: Computational results for rank decompositions of random tensors in Case 3. Time (in seconds) is the average of the consumed time. Error is the average error in frobenius norm.

| $(n_1, \ldots, n_m)$ | r | $time_{nls}$ | $error_{nls}$ | $time_{gp}$ | $error_{gp}$ |
|---|---|---|---|---|---|
| (9,8,6) | 11 | 0.544 | NaN | 0.00223 | 3.261e-10 |
| (12,10,10) | 14 | 0.657 | NaN | 0.00997 | 1.849e-09 |
| (21,20,10) | 29 | 2.817 | NaN | 0.0229 | 2.644e-07 |
| (32,30,10) | 40 | 4.026 | NaN | 0.0352 | 7.435e-08 |
| (42,40,20) | 55 | 6.006 | NaN | 0.241 | 4.968e-07 |
| (52,50,30) | 70 | 10.392 | NaN | 0.767 | 1.392e-06 |
| (64,60,30) | 88 | 14.617 | NaN | 0.995 | 1.752e-05 |
| (90,70,30) | 100 | 19.821 | NaN | 1.566 | 9.775e-06 |

**Example 3.3.5.** In this example, we explore the numerical performance of Algorithms 3.2.1 when the tensor is not but close to a rank r tensor. For the given dimensions $n_1, \ldots, n_m$, we generate the tensor

$$\mathcal{R} = \sum_{s=1}^{r} \mathbf{u}^{s,1} \otimes \mathbf{u}^{s,2} \otimes \cdots \otimes \mathbf{u}^{s,m},$$

where each $\mathbf{u}^{s,j} \in \mathbb{C}^{n_j}$ is a complex vector whose real and imaginary parts are generated randomly, obeying the Gaussian distribution with mean $s$. We perturb $\mathcal{R}$ by another tensor $\mathcal{E}$, whose entries are generated with the Gaussian distribution with mean 0 and standard derivation 1. Then we normalize the perturbing tensor $\mathcal{E}$ to have a desired norm $\epsilon$. The tensor $\mathcal{F}$ is then generated as

$$\mathcal{F} = \mathcal{R} + \mathcal{E}.$$

Denote $\mathcal{X}^{gp}$ be the solution of algorithm 3.2.1, we choose $\epsilon$ to be one of $10^{-3}, 10^{-6}, 10^{-9}$, and use the relative errors

$$\rho = \frac{\|\mathcal{F} - \mathcal{X}^{gp}\|}{\|\mathcal{E}\|},$$

to measure the approximation quality of the approximating tensors $\mathcal{X}^{\text{gp}}$ . For each case of $(n_1, \ldots, n_m), r$ and $\epsilon$, we generate 20 random instances of $\mathcal{R}, \mathcal{F}, \mathcal{E}$. The computational results are reported in Tables 3.5. For each case of $(n_1, \ldots, n_m)$ and $r$, we list the mean of above relative errors.

Table 3.5: Computational results for rank decompositions of random tensors with error $\epsilon$ in Case 3. The mean of relative errors are shown in the table.

| $(n_1, \ldots, n_m)$ | r | $\epsilon = 10^{-3}$ | $\epsilon = 10^{-6}$ | $\epsilon = 10^{-9}$ |
|---|---|---|---|---|
| (9,8,6) | 10 | 68.754 | 110.9 | 82.53 |
| (12,12,10) | 14 | 62.944 | 50.438 | 62.281 |
| (21,20,10) | 29 | 2248.8 | 4047.4 | 3273.2 |
| (32,30,10) | 40 | 385.94 | 330.26 | 466.02 |
| (42,40,20) | 55 | 681.21 | 734.26 | 745.05 |
| (52,50,30) | 70 | 807.05 | 807.79 | 813.89 |
| (64,60,30) | 88 | 4951.7 | 6049.3 | 6041.6 |
| (90,70,30) | 100 | 5088.5 | 5015.5 | 5328.3 |

## 3.4 Conclusions

This Chapter gives a novel algorithm for the general nonsymmetric tensor decomposition problem. Under some generic conditions, we prove that the exact tensor decomposition can be found by the proposed algorithm using linear algebra operations only. Numerical examples successfully demonstrate the robustness and efficiency of our algorithm.

Chapter 3 in full, is currently being prepared for submission for publication, which is a joint work with Nie, Jiawang and Yang, Zi.

# Chapter 4

# Tensor Canonical Correlation Analysis

Multi-view learning has attracted much attention in recent years in the community of machine learning due to its increasing availability of multi-view data sets in a wide range of scientific domains such as computer vision, computational biology, and medical imaging analysis. A multi-view data set consists of multiple objects. Each object is measured from different aspects, e.g., an image can be described by color, texture, and shapes; A movie clip contains audio, text and image. Each aspect is called a view, which can be complementary or supplementary to other views. Multi-view data contain much more information than each single view, but it simultaneously causes the difficulty of learning models due to the heterogeneous gap among multiple views. Multi-view learning is proposed to reduce the heterogeneous gap by enforcing the consistency among multiple views.

## 4.1 TCCA as a Tensor Approximation Problem

Let $\{(\mathbf{y}_{i,1}, \ldots, \mathbf{y}_{i,m})\}_{i=1}^{N}$ be a multi-view data set, with $m$ views and $N$ points. The vector $\mathbf{y}_{i,j} \in \mathbb{R}^{n_j}$ is the $i$th data point of the view $j$ residing in the $n_j$-dimensional space. We are looking for a $r$-dimensional latent space $\mathbb{R}^r$ such that each $\mathbf{y}_{i,j}$ is projected to $\mathbf{z}_{i,j} \in \mathbb{R}^r$. The projection for the $j$th view can be represented by a matrix $P_j$, that is, $\mathbf{z}_{i,j} = P_j^{\mathrm{T}} \mathbf{y}_{i,j}$. The higher order canonical correlation

$\rho$ of $m$ views is the quantity

$$\rho \coloneqq \sum_{i=1}^{N} \sum_{s=1}^{r} \prod_{j=1}^{m} (\mathbf{z}_{i,j})_s. \tag{4.1.1}$$

The tensor canonical correlation analysis aims to find optimal projection matrices $P_1, \ldots, P_m$ that maximize $\rho$. When $m = 2$, $\rho$ reduces to the trace of sample cross-correlation, which is used in the classical canonical correlation analysis. When $m \geq 3$, $\rho$ generalizes CCA for capturing higher order correlations, which is inherently different from the sum of pairwise correlations [47].

The connection of $\rho$ as in (4.1.1) to a tensor can be built based on the $t$-mode product of a tensor obtained from the input data with projection matrices $P_1, \ldots, P_m$. The tensor of the input data is the $m$th order tensor of dimension $n_1 \times \cdots \times n_m$

$$\mathcal{C} \coloneqq \sum_{i=1}^{N} \mathbf{y}_{i,1} \otimes \cdots \otimes \mathbf{y}_{i,m}. \tag{4.1.2}$$

Write that $P_j = [\mathbf{p}^{1,j}, \ldots, \mathbf{p}^{r,j}]$, where $\mathbf{p}^{s,j} \in \mathbb{R}^{n_j}$ is the $s$th column of $P_j$. The higher order canonical correlation $\rho$ can be written as

$$\rho = \sum_{s=1}^{r} (\mathbf{p}^{s,1})^T \times_1 \cdots (\mathbf{p}^{s,m})^T \times_m \mathcal{C}. \tag{4.1.3}$$

People often pose the uncorrelation constraints for projected points in the latent common space

$$\frac{1}{N} \sum_{i=1}^{N} \mathbf{z}_{i,j} \mathbf{z}_{i,j}^{\mathrm{T}} = P_j^{\mathrm{T}} C_j P_j = I_r, j = 1, \ldots, m, \tag{4.1.4}$$

where the $j$th view matrix

$$C_j \coloneqq \frac{1}{N} \sum_{i=1}^{N} \mathbf{y}_{i,j} \mathbf{y}_{i,j}^{\mathrm{T}}$$

Denote the vectors and tensor

$$\mathbf{u}^{s,j} \coloneqq C_j^{\frac{1}{2}} \mathbf{p}^{s,j}, \quad \mathbf{p}^{s,j} \coloneqq C_j^{-\frac{1}{2}} \mathbf{u}^{s,j}, \tag{4.1.5}$$

$$\mathcal{M} \coloneqq C_1^{-\frac{1}{2}} \times_1 \cdots C_m^{-\frac{1}{2}} \times_m \mathcal{C}. \tag{4.1.6}$$

Then, we get the tensor correlation maximization problem

$$\begin{cases} \max\limits_{\mathbf{u}^{s,j}} & \sum_{s=1}^{r}(\mathbf{u}^{s,1})^T \times_1 \cdots (\mathbf{u}^{s,m})^T \times_m \mathcal{M} \\ \text{s.t.} & \|\mathbf{u}^{s,j}\|_2 = 1,\ s = 1, \dots r,\ j = 1, \dots, m, \\ & (\mathbf{u}^{s,j})^{\mathrm{T}}\mathbf{u}^{s',j} = 0 \quad \text{for all } s \neq s'. \end{cases} \tag{4.1.7}$$

The above is equivalent to the rank-$r$ tensor approximation problem

$$\begin{cases} \min\limits_{\mathbf{u}^{s,j},\lambda_s} & \left\| \mathcal{M} - \sum_{s=1}^{r} \lambda_s \cdot \mathbf{u}^{s,1} \otimes \cdots \otimes \mathbf{u}^{s,m} \right\|^2, \\ \text{s.t.} & \|\mathbf{u}^{s,j}\|_2 = 1, s = 1, \dots r,\ j = 1, \dots, m, \\ & (\mathbf{u}^{s,j})^{\mathrm{T}}\mathbf{u}^{s',j} = 0 \quad \text{for all } s \neq s'. \end{cases} \tag{4.1.8}$$

The optimization (4.1.8) requires to compute the best rank-$r$ orthogonal tensor approximation. This is typically a computationally hard task. Generally, the orthogonality constraints in (4.1.7) is hard to be enforced, because the rank decomposition and the orthogonal decomposition are usually not achievable simultaneously [34]. For better performance in computational practice, people often relax the orthogonality constraints (see [41]) and then solve the following relaxation of (4.1.8):

$$\begin{cases} \min\limits_{\mathbf{u}^{s,j},\lambda_s} & \left\| \mathcal{M} - \sum_{s=1}^{r} \lambda_s \cdot \mathbf{u}^{s,1} \otimes \cdots \otimes \mathbf{u}^{s,m} \right\|^2, \\ \text{s.t.} & \|\mathbf{u}^{s,j}\|_2 = 1, s = 1, \dots r,\ j = 1, \dots, m. \end{cases} \tag{4.1.9}$$

After the vectors $\mathbf{u}^{s,j}$ are obtained by solving (4.1.9), the projection matrices $P_j$ can be chosen such that $\mathbf{p}^{s,j} = C_r^{-\frac{1}{2}}\mathbf{u}^{s,j}$. We would like to remark that when $\mathcal{M}$ is sufficiently close to a rank-$r$ orthogonal tensor, the optimizer of (4.1.9) is expected to be close to a rank-$r$ orthogonal tensor.

## 4.2 The Algorithm for TCCA

For the given multi-view data set $\{(\mathbf{y}_{i,1}, \dots, \mathbf{y}_{i,m})\}_{i=1}^{n}$, we can formulate the tensor $\mathcal{M}$ as in (4.1.6). Then compute an approximating tensor for $\mathcal{M}$ and use it to get the projection matrices $P_j$.

We use the method described in Section 2.2 to compute a rank-$r$ approximation for $\mathcal{M}$. Suppose the rank $r \leq n_1$. By (2.0.5), the equation (2.0.3) is equivalent to

$$A[\mathcal{M}, j](M^{j,k}[G])^T = B[\mathcal{M}, j, k]. \tag{4.2.1}$$

Due to noises, the linear equation (4.2.1) may be overdetermined or even inconsistent. Therefore, we look for a matrix $G$ that satisfies (4.2.1) as much as possible. This can be done by solving linear least squares. Let $G^{ls}$ be a least square solution to

$$\min_{G\in\mathbb{C}^{[r]\times J}} \sum_{\tau=(i,j,k)\in J} \left\| A[\mathcal{M},j]M^{j,k}[G]^T - B[\mathcal{M},j,k] \right\|^2. \tag{4.2.2}$$

After $G^{ls}$ is obtained, select generic scalars $\xi_{j,k} \in \mathbb{R}$ obeying the standard normal distribution and let

$$M[\xi, G^{ls}] := \sum_{(1,j,k)\in J} \xi_{j,k} M^{j,k}[G^{ls}]. \tag{4.2.3}$$

We can compute its Schur Decomposition as

$$Q^* M[\xi, G^{ls}] Q = T, \tag{4.2.4}$$

where $Q = [\mathbf{q}_1, \ldots, \mathbf{q}_r]$ is unitary and $T$ is upper triangular. For $s = 1, ..., r$, $j = 2, ..., m$, let

$$\mathbf{v}^{s,j} := (1, \mathbf{q}_s^* M^{j,2}[G^{ls}]\mathbf{q}_s, \ldots, \mathbf{q}_s^* M^{j,n_j}[G^{ls}]\mathbf{q}_s). \tag{4.2.5}$$

If the noises are big, it may have complex eigenvalue pairs, $\mathbf{q}_s$ maybe complex and the above vectors $\mathbf{v}^{s,j}$ maybe complex. In computational practice, we can choose the real part to get a real tensor approximation. Denote the real part of $\mathbf{v}^{s,j}$ by $\hat{\mathbf{v}}^{s,j}_{real}$. After they are obtained, we solve the linear least squares problem

$$\min_{\mathbf{z}_1,\ldots,\mathbf{z}_r\in\mathbb{R}^{n_1}} \left\| \sum_{s=1}^{r} \mathbf{z}_s \otimes \mathbf{v}^{s,2}_{real} \otimes \mathbf{v}^{s,3}_{real} \otimes ... \otimes \mathbf{v}^{s,m}_{real} - \mathcal{M} \right\|^2. \tag{4.2.6}$$

Let $(\mathbf{v}^{1,1}, \mathbf{v}^{2,1}, ..., \mathbf{v}^{r,1})$ be optimal ones for the least squares problem (4.2.6). Then we consider the tensor

$$\mathcal{X}^{gp} := \sum_{s=1}^{r} \mathbf{v}^{s,1} \otimes \mathbf{v}^{s,2}_{real} \otimes ... \otimes \mathbf{v}^{s,m}_{real}. \tag{4.2.7}$$

It can be used as an initial point for solving the nonlinear optimization

$$\min_{\mathbf{u}^{s,j}\in\mathbb{R}^{n_j}} \left\| \sum_{s=1}^{r} \mathbf{u}^{s,1} \otimes \mathbf{u}^{s,2} \otimes \cdots \otimes \mathbf{u}^{s,m} - \mathcal{M} \right\|^2. \tag{4.2.8}$$

By solving (4.2.8), one can improve the quality of the rank-$r$ approximating tensor $\mathcal{X}^{opt}$. Finally, we get the projection matrices $P_1, \ldots, P_m$ as in (4.1.5).

The above can be summarized as the following algorithm.

**Algorithm 4.2.1.** (A generating polynomial method for TCCA)

Input: a multi-view data set $\{(\mathbf{y}_{i,1}, \ldots, \mathbf{y}_{i,m})\}_{i=1}^{N}$ and an approximating rank $r \leq n_1$.

Step 1. Generate tensor $\mathcal{M} \in \mathbb{R}^{n_1 \times \cdots n_m}$ as in (4.1.6).

Step 2. Solve the linear least squares (4.2.2) of tensor $\mathcal{M}$ for an optimizer $G^{ls}$.

Step 3. Choose generic $\xi_{j,k} \in \mathbb{R}$ obeying the standard normal distribution and formulate $M[\xi, G^{ls}]$ as in (4.2.3). Compute the Schur Decomposition (4.2.4).

Step 4. For $s \in 1, ..., r$ and $j \in 2, ..., m$, compute $\mathbf{v}^{s,j}$ as in (4.2.5) and keep its real part only ($\mathbf{v}^{s,j} = \text{real}(\mathbf{v}^{s,j})$). Solve (4.2.6) for optimal solution $(\mathbf{v}^{1,1}, \mathbf{v}^{2,1}, ..., \mathbf{v}^{r,1})$.

Step 5. Compute an improved solution $\mathbf{u}^{s,j}$ as in

$$\min_{\mathbf{u}^{s,j} \in \mathbb{R}^{n_j}} \left\| \sum_{s=1}^{r} \mathbf{u}^{s,1} \otimes \mathbf{u}^{s,2} \otimes ... \otimes \mathbf{u}^{s,m} - \mathcal{F} \right\|^2.$$

Output: The matrices $P_j, \ldots, P_m$ as in (4.1.5).

When $\mathcal{F}$ is a rank-$r$ tensor, Algorithm 4.2.1 should give a rank-$r$ decomposition for $\mathcal{F}$. When $\mathcal{F}$ is close to a rank-$r$ tensor, Algorithm 4.2.1 is expected to give a good rank-$r$ approximation. An interesting future work is to study the stability analysis.

## 4.3 Numerical Experiments on Real Data

We implement our proposed Algorithm 3 in `MATLAB` and run numerical experiments in `MATLAB` 2020b on a workstation with Ubuntu 20.04.2 LTS, Intel® Xeon(R) Gold 6248R CPU @ 3.00GHz and memory 1TB. We evaluate our algorithm for multi-view feature extraction by comparing it with baseline methods on two real data sets.

### Data description and experimental setup

Two image data sets are used in this experiment: Caltech101-7 [38] and Scene15 [35], we apply six feature descriptors to extract features of views including centrist [60], gist [49], lbp [48], histogram of oriented gradient (hog), color

histogram (ch), and sift-spm [35]. Note that Scene15 consists of gray images, so ch is not used. The statistics of the two multi-view data sets are summarized in Table 4.1.

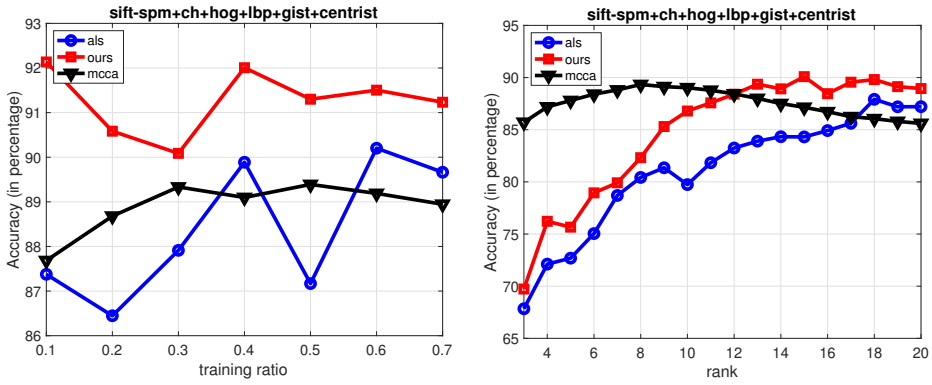
Table 4.1: Data sets used in the experiments.

| Data set | samples | class | centrist | gist | lbp | hog | ch | sift-spm |
|---|---|---|---|---|---|---|---|---|
| Caltech101-7 | 1474 | 7 | 254 | 512 | 1180 | 1008 | 64 | 1000 |
| Scene15 | 4310 | 15 | 254 | 512 | 531 | 360 | - | 1000 |

As our main focus is on data sets with more than two views, our proposed algorithm is evaluated by comparing with multiset CCA (mcca) [59] and tensor CCA using ALS (als) [41]. For each data, we first apply PCA to each view to reduce the input dimension to 20 so that the constructed tensors can be properly handled by tensor-based methods. And then, we split the data into training and testing sets with a predefined training ratio. All compared methods are run on the training data to get the projection matrix of each view for a given dimension of the common space (or rank). To report the testing accuracy, we apply the learned projection matrix to both training and testing sets of each view, concatenate the projected features of all views, train linear support vector classifier (SVC) [5] on training data and evaluate on testing data. The classification accuracy is used as the evaluation metric. The regularization parameter of the linear SVC is tuned in $\{0.01, 0.1, 1, 10, 100\}$. The experiments of compared methods on each data set are repeated ten times with randomly sampled training and testing sets, and the mean accuracy with standard deviation on the ten experiments are reported for compared methods.

**Experiments on Caltech101-7**

We tested the overall performance of three compared methods on Caltech101-7 with all combinations of more than two views. For six views, there are 42 combinations in total. This experiment is conducted on 30% training data and 70% testing data by running three compared methods on each combination separately with the size of the common space (or rank) varied from 3 to 20. The experiment is repeated 10 times on randomly splits drawn from the input data, and the mean accuracies with standard deviations of three compared methods are reported in Table 4.2.




(a) (b)

Figure 4.1: Sensitivity analysis of compared methods with six views on data Caltech101-7. (a) varying the size of common space on 30% training data; (b) varying the training ratios over common spaces from 3 to 20.

From Table 4.2, we have the following observations: (i) als outperforms mcca on three views, but underperforms mcca for more than three views; (ii) Our method outperforms both als and mcca consistently over all 42 combinations. These results imply that tensor-based methods can outperform mcca, when a good tensor approximation solver like our proposed algorithm is applied.

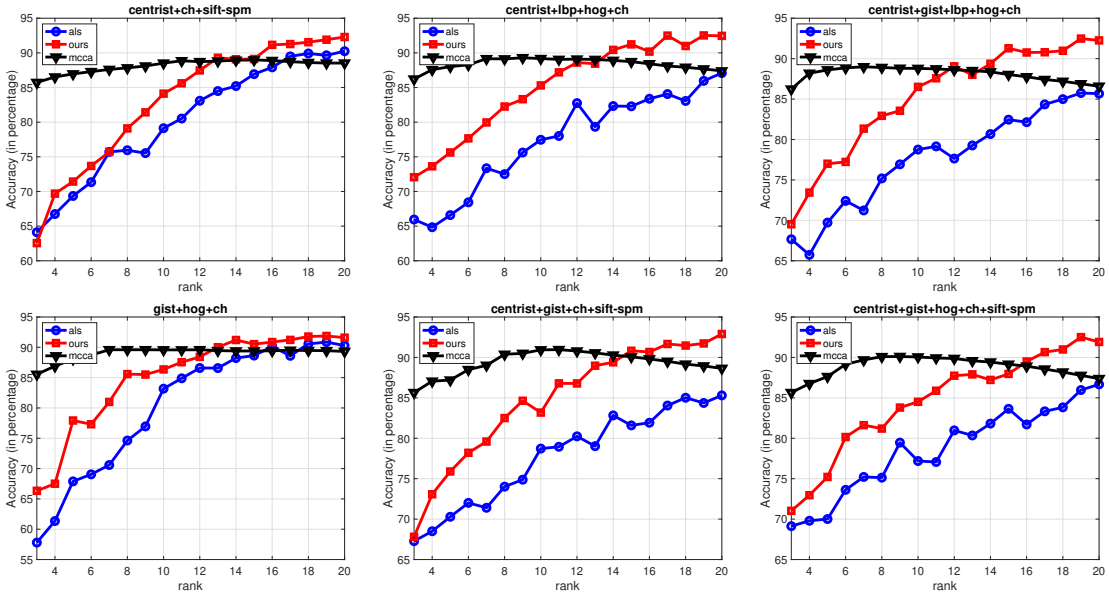



Figure 4.2: Experimental results of compared methods on Caltech101-7 with three, four and five views over 30% training data.

We further investigate the impact of compared methods in terms of the varied ranks and the training ratios. In Figure 4.1(a), the mean accuracy of testing data

Table 4.2: Mean accuracy and standard deviation of three compared methods on 42 data sets generated from Caltech101-7 over 10 random splits with 30% training data and rank 20.

| views | als | ours | mcca |
|---|---|---|---|
| centrist+gist+lbp | 95.23 ± 0.69 | **95.30 ± 0.73** | 90.74 ± 0.88 |
| centrist+gist+hog | 95.26 ± 0.58 | **95.32 ± 0.46** | 90.28 ± 0.96 |
| centrist+gist+ch | 92.47 ± 2.21 | **93.86 ± 0.96** | 90.66 ± 1.25 |
| centrist+gist+sift-spm | 95.56 ± 0.75 | **95.93 ± 0.39** | 92.59 ± 0.84 |
| centrist+lbp+hog | 94.95 ± 0.63 | **95.19 ± 0.65** | 90.09 ± 0.75 |
| centrist+lbp+ch | 92.63 ± 0.56 | **92.97 ± 0.65** | 90.06 ± 1.04 |
| centrist+lbp+sift-spm | 94.82 ± 0.75 | **95.15 ± 0.71** | 91.59 ± 0.71 |
| centrist+hog+ch | 91.29 ± 1.14 | **92.62 ± 1.07** | 89.79 ± 0.68 |
| centrist+hog+sift-spm | 93.46 ± 1.34 | **93.86 ± 1.27** | 91.15 ± 0.55 |
| centrist+ch+sift-spm | 90.24 ± 1.83 | **92.29 ± 0.94** | 88.99 ± 0.33 |
| gist+lbp+hog | 95.49 ± 0.92 | **95.63 ± 0.82** | 90.25 ± 0.85 |
| gist+lbp+ch | 93.04 ± 1.11 | **93.99 ± 0.60** | 90.93 ± 1.29 |
| gist+lbp+sift-spm | 95.71 ± 1.16 | **96.02 ± 0.60** | 92.41 ± 0.92 |
| gist+hog+ch | 90.86 ± 1.74 | **91.87 ± 1.92** | 89.60 ± 0.74 |
| gist+hog+sift-spm | 93.02 ± 0.58 | **93.05 ± 0.52** | 91.03 ± 0.47 |
| gist+ch+sift-spm | 90.14 ± 2.19 | **92.73 ± 1.09** | 90.25 ± 0.86 |
| lbp+hog+ch | 91.65 ± 1.51 | **92.98 ± 1.12** | 89.88 ± 0.67 |
| lbp+hog+sift-spm | 93.18 ± 0.77 | **94.16 ± 1.10** | 91.21 ± 0.50 |
| lbp+ch+sift-spm | 90.48 ± 1.61 | **92.33 ± 1.10** | 89.09 ± 0.46 |
| hog+ch+sift-spm | 88.35 ± 3.82 | **91.93 ± 0.94** | 90.07 ± 0.87 |
| centrist+gist+lbp+hog | 92.14 ± 3.41 | **95.12 ± 1.03** | 89.14 ± 0.70 |
| centrist+gist+lbp+ch | 89.25 ± 2.64 | **92.41 ± 1.93** | 90.17 ± 0.87 |
| centrist+gist+lbp+sift-spm | 90.05 ± 3.10 | **94.60 ± 1.25** | 90.64 ± 0.88 |
| centrist+gist+hog+ch | 84.91 ± 4.33 | **93.01 ± 2.05** | 89.63 ± 0.53 |
| centrist+gist+hog+sift-spm | 88.32 ± 4.01 | **92.94 ± 0.81** | 90.42 ± 0.46 |
| centrist+gist+ch+sift-spm | 85.30 ± 3.53 | **92.90 ± 1.94** | 90.97 ± 0.63 |
| centrist+lbp+hog+ch | 87.09 ± 2.72 | **92.52 ± 1.78** | 89.29 ± 0.55 |
| centrist+lbp+hog+sift-spm | 87.00 ± 5.27 | **93.30 ± 2.36** | 89.82 ± 0.58 |
| centrist+lbp+ch+sift-spm | 84.64 ± 3.74 | **92.21 ± 1.96** | 89.21 ± 0.90 |
| centrist+hog+ch+sift-spm | 84.65 ± 3.19 | **92.31 ± 1.33** | 90.02 ± 0.51 |
| gist+lbp+hog+ch | 86.50 ± 6.35 | **92.76 ± 1.41** | 89.11 ± 0.38 |
| gist+lbp+hog+sift-spm | 87.96 ± 2.63 | **93.86 ± 1.21** | 89.96 ± 0.47 |
| gist+lbp+ch+sift-spm | 85.25 ± 3.50 | **91.93 ± 1.98** | 90.83 ± 0.60 |
| gist+hog+ch+sift-spm | 81.57 ± 5.98 | **90.82 ± 2.06** | 90.53 ± 0.54 |
| lbp+hog+ch+sift-spm | 83.20 ± 4.93 | **91.59 ± 1.65** | 89.89 ± 0.68 |
| gist+lbp+hog+ch+sift-spm | 84.38 ± 3.13 | **91.32 ± 2.46** | 89.76 ± 0.61 |
| centrist+lbp+hog+ch+sift-spm | 86.11 ± 3.49 | **91.80 ± 2.32** | 89.48 ± 0.73 |
| centrist+gist+hog+ch+sift-spm | 86.67 ± 4.70 | **92.51 ± 1.11** | 90.12 ± 0.60 |
| centrist+gist+lbp+ch+sift-spm | 88.70 ± 3.73 | **93.07 ± 1.35** | 89.96 ± 0.87 |
| centrist+gist+lbp+hog+sift-spm | 85.31 ± 4.66 | **94.17 ± 1.37** | 89.25 ± 0.47 |
| centrist+gist+lbp+hog+ch | 85.75 ± 4.66 | **92.48 ± 2.10** | 89.00 ± 0.49 |
| sift-spm+ch+hog+lbp+gist+centrist | 87.91 ± 2.98 | **90.09 ± 2.95** | 89.33 ± 0.65 |

obtained by three compared methods varies when the training ratio increases from 10% to 70%. Due to the complexity of the tensor approximation problem, both als and our method show larger fluctuations than that of mcca when the training ratio increases. However, our method consistently outperforms both als and mcca over all tested training ratios. In Figure 4.1(b), we show the mean accuracy of compared methods on 10 random splits with 30% training data by varying rank from 3 to 20 on six views. In addition, we show the mean accuracy of compared methods on varied ranks on 30% with respect to combinations of different views in Figure 4.2. All these results demonstrate a similar trend with respect to testing accuracy when the rank increases from 3 to 20: mcca shows better performance on small ranks, but our method outperforms both mcca and als on large ranks, and overall our method obtains the best performance over all tested ranks. From Figure 4.1, we can see that our model on 30% training data shows the worst results comparing to other training ratios. This implies that the results in Figure 4.1 show the worst results of our method, which still outperforms other two methods as shown in Figure 4.2.

**Experiments on Scene15**

The experiments same as in section 4.3 are performed on data Scene15. In Table 4.3, tensor-based methods including both als and ours outperform mcca on all view combinations. Our method outperforms als on three and four views, while it is competitive to als on five views. Figure 4.3 demonstrates the sensitivity of compared methods by varying the rank and the training ratios. On data Scene15, the tensor-based methods are consistently better than mcca over all tested training ratios, while our method outperforms als on large ranks and is competitive on small ranks. These results are consistent with the observations on Caltech101-7 in section 4.3, especially on relatively large ranks.

Table 4.3: Mean accuracy and standard deviation of two compared methods on 16 data sets generated from Scene15 over 10 random splits with 30% training data and rank 20.

| views | als | ours | mcca |
|---|---|---|---|
| centrist+gist+lbp | 65.91 ± 1.45 | **66.66 ± 1.12** | 57.01 ± 0.81 |
| centrist+gist+hog | 67.21 ± 1.15 | **67.73 ± 1.08** | 58.47 ± 0.88 |
| centrist+gist+sift-spm | 70.40 ± 1.48 | **72.68 ± 1.70** | 64.34 ± 1.65 |
| centrist+lbp+hog | 60.29 ± 1.80 | **62.32 ± 1.56** | 53.63 ± 0.72 |
| centrist+lbp+sift-spm | 60.30 ± 1.83 | **65.43 ± 1.45** | 58.10 ± 1.46 |
| centrist+hog+sift-spm | 63.05 ± 1.98 | **67.45 ± 1.24** | 58.11 ± 1.24 |
| gist+lbp+hog | 61.08 ± 1.33 | **62.86 ± 1.20** | 54.20 ± 0.74 |
| gist+lbp+sift-spm | 65.82 ± 1.68 | **68.88 ± 1.30** | 59.23 ± 1.42 |
| gist+hog+sift-spm | 63.12 ± 3.94 | **67.13 ± 2.28** | 54.71 ± 1.76 |
| lbp+hog+sift-spm | 57.08 ± 2.54 | **60.58 ± 1.32** | 51.01 ± 2.05 |
| centrist+gist+lbp+hog | 62.33 ± 2.21 | **63.83 ± 2.11** | 51.07 ± 0.72 |
| centrist+gist+lbp+sift-spm | 61.96 ± 3.02 | **65.34 ± 1.62** | 55.56 ± 1.08 |
| centrist+gist+hog+sift-spm | 62.41 ± 3.08 | **63.67 ± 3.70** | 58.69 ± 1.19 |
| centrist+lbp+hog+sift-spm | 54.63 ± 3.26 | **57.98 ± 2.37** | 51.77 ± 1.57 |
| gist+lbp+hog+sift-spm | 58.93 ± 3.18 | **61.28 ± 2.26** | 50.05 ± 1.15 |
| centrist+gist+lbp+hog+sift-spm | **60.72 ± 2.87** | 60.35 ± 3.33 | 49.11 ± 1.49 |

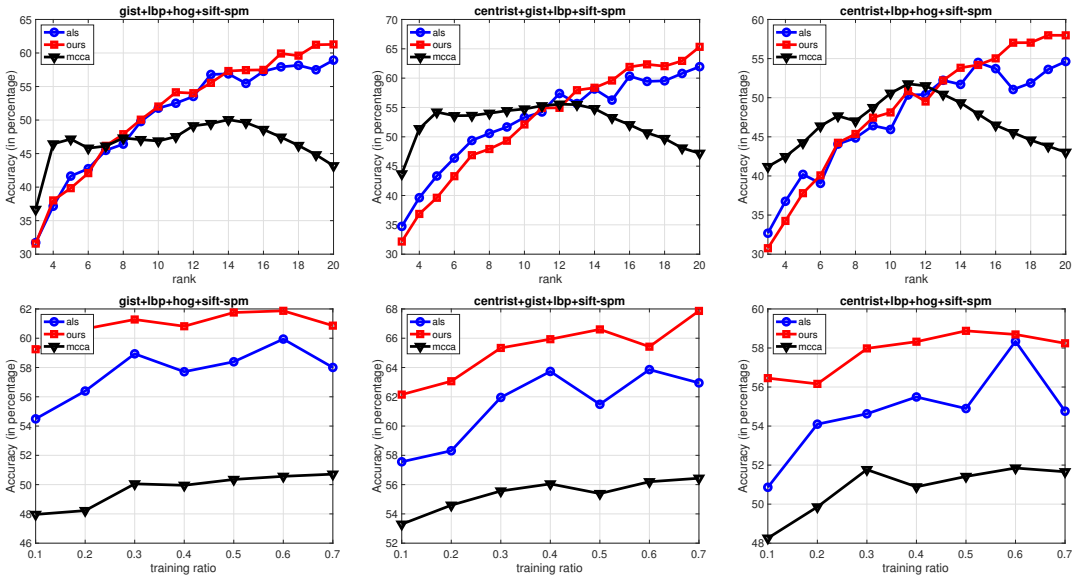



Figure 4.3: Sensitivity analysis of compared methods with four views on data Scene15. (top row) varying the size of common space on 30% training data; (bottom row) varying the training ratios over common spaces from 3 to 20.

## 4.4 Conclusion

In this chapter, we propose to use the so-called generating polynomials [43] for computing approximations for given tensors. First, we estimate generating polynomials by solving linear least squares. Second, we find their approximately common zeros, which can be done by computing Schur decompositions. Third, we construct a approximation from those zeros, by solving linear least squares. Our main conclusion is that if the tensor to be approximated is close enough to a one, then the constructed tensor is a good enough approximation. The proof is built on perturbation analysis of linear least squares and Schur decompositions. The proposed methods can also be applied to compute approximations efficiently, especially for large scale tensors.

The proposed approximation method using generating polynomials is applied to solve the problem of tensor canonical correlation analysis (TCCA). First, we reformulate TCCA as the approximation problem over the high-order correlation tensor of multi-view input data. Second, the real part of the solution obtained by our proposed approximation method is proved to be a good initial solution to existing TCCA methods. Finally, we evaluate our TCCA algorithm on two real data sets for multi-view feature extraction by comparing it with baselines. Experimental results show that our TCCA method consistently outperforms TCCA with ALS and the pairwise correlation approach in terms of different number of views.

Chapter 4 in full, is a reprint of the material as it appears in *Pacific Journal of Optimization* 2023 [45]. The dissertation author coauthored this paper with Nie, Jiawang and Wang, Li.

# Bibliography


[1] Anima Anandkumar, Rong Ge, Daniel Hsu, Sham M. Kakade, and Matus Telgarsky. Tensor decompositions for learning latent variable models, 2014.

[2] Galen Andrew, Raman Arora, Jeff Bilmes, and Karen Livescu. Deep canonical correlation analysis. In *International conference on machine learning*, pages 1247–1255. PMLR, 2013.

[3] Aditya Bhaskara, Moses Charikar, Ankur Moitra, and Aravindan Vijayaraghavan. Smoothed analysis of tensor decompositions, 2014.

[4] Zehong Cao, Yu-Cheng Chang, Mukesh Prasad, M. Tanveer, and Chin-Teng Lin. Tensor decomposition for eeg signals retrieval. *2019 IEEE International Conference on Systems, Man and Cybernetics (SMC)*, Oct 2019.

[5] Chih-Chung Chang and Chih-Jen Lin. Libsvm: a library for support vector machines. *ACM transactions on intelligent systems and technology (TIST)*, 2(3):1–27, 2011.

[6] Françoise Chatelin. *Eigenvalues of Matrices*. Society for Industrial and Applied Mathematics, Philadelphia, PA, 2012.

[7] Luca Chiantini, Giorgio Ottaviani, and Nick Vannieuwenhoven. Effective criteria for specific identifiability of tensors and forms. *SIAM Journal on Matrix Analysis and Applications*, 38(2):656–681, Jan 2017.

[8] Andrzej Cichocki, Danilo Mandic, Lieven De Lathauwer, Guoxu Zhou, Qibin Zhao, Cesar Caiafa, and HUY ANH PHAN. Tensor decompositions for signal processing applications: From two-way to multiway component analysis. *IEEE Signal Processing Magazine*, 32(2):145–163, Mar 2015.

[9] Pierre Comon. Tensor decompositions, state of the art and applications. *arXiv preprint arXiv:0905.0454*, 2009.

[10] Pierre Comon and Lek-Heng Lim. Sparse representations and low-rank tensor approximation. 2011.

[11] Fengyu Cong, Qiu-Hua Lin, Li-Dan Kuang, Xiao-Feng Gong, Piia Astikainen, and Tapani Ristaniemi. Tensor decomposition of eeg signals: A brief review. *Journal of Neuroscience Methods*, 248:59 – 69, 2015.

[12] David Cox, John Little, and Donal O'Shea. *Ideals, Varieties, and Algorithms. An Introduction to Computational Algebraic Geometry and Commutative Algebra*. Springer Cham, 2013.

[13] Lieven De Lathauwer, Bart De Moor, and Joos Vandewalle. On the best rank-1 and rank-(r 1, r 2,..., rn) approximation of higher-order tensors. *SIAM journal on Matrix Analysis and Applications*, 21(4):1324–1342, 2000.

[14] Vin De Silva and Lek-Heng Lim. Tensor rank and the ill-posedness of the best low-rank approximation problem. *SIAM Journal on Matrix Analysis and Applications*, 30(3):1084–1127, 2008.

[15] James W. Demmel. *Applied Numerical Linear Algebra*. SIAM, January 1997.

[16] J. E. Dennis and Robert B. Schnabel. *Numerical Methods for Unconstrained Optimization and Nonlinear Equations*. Society for Industrial and Applied Mathematics, 1996.

[17] Ignat Domanov and Lieven De Lathauwer. On the uniqueness of the canonical polyadic decomposition of third-order tensors—part i: Basic results and uniqueness of one factor matrix. *SIAM Journal on Matrix Analysis and Applications*, 34(3):855–875, 2013.

[18] Ignat Domanov and Lieven De Lathauwer. Canonical polyadic decomposition of third-order tensors: Reduction to generalized eigenvalue decomposition. *SIAM Journal on Matrix Analysis and Applications*, 35(2):636–660, Jan 2014.

[19] Ignat Domanov and Lieven De Lathauwer. Generic uniqueness conditions for the canonical polyadic decomposition and indscal. *SIAM Journal on Matrix Analysis and Applications*, 36(4):1567–1589, 2015.

[20] Ignat Domanov and Lieven De Lathauwer. Canonical polyadic decomposition of third-order tensors: relaxed uniqueness conditions and algebraic algorithm, 2016.

[21] Shmuel Friedland and Giorgio Ottaviani. The number of singular vector tuples and uniqueness of best rank-one approximation of tensors. *Foundations of Computational Mathematics*, 14(6):1209–1242, 2014.

[22] Lars Grasedyck, Daniel Kressner, and Christine Tobler. A literature survey of low-rank tensor approximation techniques. *GAMM-Mitteilungen*, 36(1):53–78, 2013.

[23] Yu Guan, Moody T Chu, and Delin Chu. Convergence analysis of an svd-based algorithm for the best rank-1 tensor approximation. *Linear Algebra and its Applications*, 555:53–69, 2018.

[24] Bingni Guo, Jiawang Nie, and Zi Yang. Learning diagonal gaussian mixture models and incomplete tensor decompositions, 2021.

[25] Alexandros Haliassos, Kriton Konstantinidis, and Danilo P. Mandic. Supervised learning for non-sequential data with the canonical polyadic decomposition, 2020.

[26] David R Hardoon and John Shawe-Taylor. Sparse canonical correlation analysis. *Machine Learning*, 83(3):331–353, 2011.

[27] Hotelling Harold. Relations between two sets of variates. *Biometrika*, 28(3/4):321–377, 1936.

[28] Nikos Kargas and Nicholas D. Sidiropoulos. Supervised learning and canonical decomposition of multivariate functions. *IEEE Transactions on Signal Processing*, 69:1097–1107, 2021.

[29] C. T. Kelley. *Iterative Methods for Linear and Nonlinear Equations*. Society for Industrial and Applied Mathematics, 1995.

[30] Tamara G Kolda and Brett W Bader. Tensor decompositions and applications. *SIAM review*, 51(3):455–500, 2009.

[31] Joseph B. Kruskal. Three-way arrays: rank and uniqueness of trilinear decompositions, with application to arithmetic complexity and statistics. *Linear Algebra and its Applications*, 18(2):95 – 138, 1977.

[32] Brett W. Larsen and Tamara G. Kolda. Practical leverage-based sampling for low-rank tensor decomposition, 2020.

[33] Lieven Lathauwer. A link between the canonical decomposition in multilinear algebra and simultaneous matrix diagonalization. *SIAM J. Matrix Analysis Applications*, 28:642–666, 01 2006.

[34] Lieven Lathauwer and Bart De Moor. A multi-linear singular value decomposition. *Society for Industrial and Applied Mathematics*, 21:1253–1278, 03 2000.

[35] Svetlana Lazebnik, Cordelia Schmid, and Jean Ponce. Beyond bags of features: Spatial pyramid matching for recognizing natural scene categories. In *2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'06)*, volume 2, pages 2169–2178. IEEE, 2006.

[36] S. E. Leurgans, R. T. Ross, and R. B. Abel. A decomposition for three-way arrays. *SIAM Journal on Matrix Analysis and Applications*, 14(4):1064–1083, 1993.

[37] S. E. Leurgans, R. T. Ross, and R. B. Abel. A decomposition for three-way arrays. *SIAM Journal on Matrix Analysis and Applications*, 14(4):1064–1083, 1993.

[38] Fei-Fei Li, Rob Fergus, and Pietro Perona. Learning generative visual models from few training examples: An incremental bayesian approach tested on 101 object categories. *Computer vision and Image understanding*, 106(1):59–70, 2007.

[39] Lek-Heng Lim and Pierre Comon. Multiarray signal processing: Tensor decomposition meets compressed sensing. *Comptes Rendus Mécanique*, 338(6):311–320, 2010.

[40] Benjamin Lovitz and Fedor Petrov. A generalization of kruskal's theorem on tensor decomposition, 2021.

[41] Yong Luo, Dacheng Tao, Kotagiri Ramamohanarao, Chao Xu, and Yonggang Wen. Tensor canonical correlation analysis for multi-view dimension reduction. *IEEE transactions on Knowledge and Data Engineering*, 27(11):3111–3124, 2015.

[42] Kathleen R. Murphy, Colin A. Stedmon, Daniel Graeber, and Rasmus Bro. Fluorescence spectroscopy and multi-way techniques. parafac. *Anal. Methods*, 5:6557–6566, 2013.

[43] Jiawang Nie. Generating polynomials and symmetric tensor decompositions, 2015.

[44] Jiawang Nie and Li Wang. Semidefinite relaxations for best rank-1 tensor approximations. *SIAM Journal on Matrix Analysis and Applications*, 35(3):1155–1179, 2014.

[45] Jiawang Nie, Li Wang, and Zequn Zheng. Higher order correlation analysis for multi-view learning, 2022.

[46] Jiawang Nie and Zi Yang. Hermitian tensor decompositions. *SIAM Journal on Matrix Analysis and Applications*, 41(3):1115–1144, 2020.

[47] Allan Aasbjerg Nielsen. Multiset canonical correlations analysis and multispectral, truly multitemporal remote sensing data. *IEEE transactions on image processing*, 11(3):293–305, 2002.

[48] Timo Ojala, Matti Pietikäinen, and Topi Mäenpää. Multiresolution gray-scale and rotation invariant texture classification with local binary patterns. *IEEE Transactions on Pattern Analysis & Machine Intelligence*, (7):971–987, 2002.

[49] Aude Oliva and Antonio Torralba. Modeling the shape of the scene: A holistic representation of the spatial envelope. *International journal of computer vision*, 42(3):145–175, 2001.

[50] Anh-Huy Phan, Petr Tichavskỳ, and Andrzej Cichocki. Fast alternating ls algorithms for high order candecomp/parafac tensor factorizations. *IEEE Transactions on Signal Processing*, 61(19):4834–4846, 2013.

[51] John A. Rhodes. A concise proof of kruskal's theorem on tensor decomposition. *Linear Algebra and its Applications*, 432(7):1818–1824, 2010.

[52] Eugenio Sanchez and Bruce R. Kowalski. Tensorial resolution: A direct trilinear decomposition. *Journal of Chemometrics*, 4, 1990.

[53] Igor Schafarevich. Basic algebraic geometry i—varieties in projective space, 1988.

[54] Laurent Sorber, Marc Van Barel, and Lieven De Lathauwer. Optimization-based algorithms for tensor decompositions: Canonical polyadic decomposition, decomposition in rank-(l_r,l_r,1) terms, and a new generalization. *SIAM Journal on Optimization*, 23(2):695–720, 2013.

[55] Bernd Sturmfels. Solving systems of polynomial equations. 2002.

[56] Simon Telen and Nick Vannieuwenhoven. A normal form algorithm for tensor rank decomposition. *ACM Trans. Math. Softw.*, 48(4), dec 2022.

[57] Viivi Uurtio, João M Monteiro, Jaz Kandola, John Shawe-Taylor, Delmiro Fernandez-Reyes, and Juho Rousu. A tutorial on canonical correlation methods. *ACM Computing Surveys (CSUR)*, 50(6):1–33, 2017.

[58] N. Vervliet, O. Debals, and L. De Lathauwer. Tensorlab 3.0 — numerical optimization strategies for large-scale constrained and coupled matrix/tensor factorization. In *2016 50th Asilomar Conference on Signals, Systems and Computers*, pages 1733–1738, 2016.

[59] Javier Vía, Ignacio Santamaría, and Jesús Pérez. A learning algorithm for adaptive canonical correlation analysis of several data sets. *Neural Networks*, 20(1):139–152, 2007.

[60] Jianixn Wu and James M Rehg. Where am i: Place instance and category recognition using spatial pact. In *2008 Ieee Conference on Computer Vision and Pattern Recognition*, pages 1–8. IEEE, 2008.

[61] Xinghao Yang, Liu Weifeng, Wei Liu, and Dacheng Tao. A survey on canonical correlation analysis. *IEEE Transactions on Knowledge and Data Engineering*, 2019.

[62] Ya-xiang Yuan. Recent advances in numerical methods for nonlinear equations and nonlinear least squares. *Numerical Algebra, Control and Optimization*, 1, 03 2011.

[63] Qingchen Zhang, Laurence T. Yang, Zhikui Chen, and Peng Li. An improved deep computation model based on canonical polyadic decomposition. *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, 48(10):1657–1666, 2018.