ABSTRACT

MIN, EUN JEONG. Generalized Eigenvalue Problems: Algorithms and Applications. (Under the direction of Dr. Hua Zhou.)

Nowadays we encounter massive data from various fields, and there are growing demands for efficient methods to transform data into the useful information. Especially, most data from various research areas contain observations of more than two variables, which we call "multivariate data". Many statistical methods have been developed to extract useful information under the category "multivariate data analysis". As dimensionality of the data gets bigger with the development of technologies, many statisticians are striving to keep pace with this change by proposing updated methods that are appropriate for those data. There are at least two characteristics of the recent data. One is high dimensionality, the data gets bigger and bigger. In many cases the number of variables is much larger than the sample size. The other one is the complexity. Multi-dimensional array data, known as a tensor, is emerging. The imaging data is the representative one. Only a few statistical methods exist to handle these data. In this dissertation, we study some tools to deal with the data with these characteristics.

In Chapter 2, we suggest a unifying approach for many constrained multivariate analysis methods. It is well known that many multivariate analysis methods are based on the generalized eigenvalue problem, which can be formulated as an optimization problem. Building on this relationship, we can solve constrained multivariate analysis problem using existing optimization algorithms. We introduced five classes of optimization algorithms to solve the constrained generalized eigenvalue problem and tested their effectiveness on some sparsity or nonnegativity constrained statistical problems.

In Chapter 3, we propose novel Canonical Correlation Analysis (CCA) methods for

two multi-dimensional array data sets. There is little literature about multivariate analysis of tensor data, especially there are no methods to conduct CCA on tensor data without destroying its own structure. Our models are based on a decomposition of the tensor parameter that enables us to handle the problem with manageable number of parameters. Also, by imposing parsimonious structure on the covariance matrix, we achieved better statistical efficiency. Simulation studies are conducted to assess performance of our proposed models. The results show that our methods exhibit excellent performance.

Generalized Eigenvalue Problems: Algorithms and Applications

by
Eun Jeong Min

A dissertation submitted to the Graduate Faculty of
North Carolina State University
in partial fulfillment of the
requirements for the Degree of
Doctor of Philosophy

Statistics

Raleigh, North Carolina

2015

APPROVED BY:

Dr. Brian Reich	Dr. Eric Laber
Dr. Yichao Wu	Dr. Hua Zhou Chair of Advisory Committee

DEDICATION

To my beloved family.

BIOGRAPHY

Eun Jeong Min is from Seoul, the capital of South Korea. She earned her Bachelor degree in statistics and economics in Ewha Womans University in Seoul. She continued to study statistics in the Master program at Ewha Womans University under the direction of Dr. Jongwoo Song. Her thesis for master degree mainly concerns with EM algorithm and accelerated EM. In the year of 2010, she enrolled in the Ph.D program in the department of statistics at North Carolina State University. Her research interests include modern statistics, high dimensional data analysis, tensor data analysis, statistical computing, and optimization algorithm.

ACKNOWLEDGEMENTS

I would like to express my sincerest gratitude and appreciation to my advisor Dr. Hua Zhou, for his immense help and concerns about me. His intuitive instructions and generous support was my driving force in the PhD research. It was a tremendous fortune to me that I could watch and learn his profound insight and professionalism for the last three years. Time that I could spent with him in NCSU will be a nourishment for my future academic achievement.

I also would like to express my dearest gratitude and appreciation to my mentor Dr. Hoon Hong and Pauline Hong, for their endless love, help and concerns about me. Their continued concerns were great motive power for me to finish my research. Especially extremely valuable advice from Dr. Hoon Hong will be a spiritual nourishment for the future research.

I am grateful to Drs. Yichao Wu, Brian Reich and Eric Laber for being committee members for my PhD study. Their insight was a great help to my PhD research. I am grateful to Dr. Huixia Judy Wang. For the first two years in NCSU, she was a great guide to my study. Her valuable advice and warm words were great help to me, and the fuel for my study. Also, I thank Dr. Charlie Smith. For almost two years, I got his warm concerns and greetings while I worked as his teaching assistant. Also, I would like to thank Drs. Kim Weems and Howard Bondell for their great help as a graduate program director.

Thanks to my friends in our department. Great time we had together made me happy whenever I feel hard and tired. Their support and encouragement would be a great memory forever. Also thank to friends in my homeland. Their continued contacts and phone calls were a great pleasure during my PhD study.

I would like to express my deepest gratitude to my dearest family in my homeland.

Their sacrificial devotion and prays towards me are beyond price; I know I am a beloved person.

Finally, I know everything was possible because there was His plan towards me. I appreciate everything He gave me, my Father. Thank you God for blessing me much more than I deserve.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

Chapter 1

Introduction

1.1 Multivariate Data Analysis

Researchers gather data to answer scientific questions. Most data sets collected in various research areas contain observations of more than two variables. Analysis methods on those multivariate data are highly demanded. To fulfill those demands, statisticians have developed various methodologies to analyze the multivariate data. Multivariate analysis can be thought as a general term to call those methodologies. Multivariate analysis deals with the data that includes simultaneous measurements on more than two variables to get useful information or answers of the scientific inquiries.

The scientific questions can take many forms. Some researchers require an effective data dimension reduction so that one may understand the structure of the data easily without much loss of information. This kind of questions come from the scientific areas with huge datasets, such as genetics, neuroscience, pattern recognition, feature selection, and imaging analysis. One representative example is the Nuclear Magnetic Resonance (NMR) data, which is often studied in chemistry and metabolomics (Manganas et al.,

2007; Maletić-Savatić et al., 2008). NMR spectroscopy is the technique that produces a spectrum of a mixture of metabolites in biological samples, resulting high-dimensional data.. For example, the data from Manganas et al. (2007), which is studied in Allen and Maletić-Savatić (2011), has more than 2000 variables from 27 samples. By studying those NMR data, one can determine what kinds of metabolites are contained in the understand biological patterns. Principal Components Analysis (PCA) proposed by Hotelling (1936) is the one of the most widely used dimensional reduction methods. Linear Discriminant Analysis (LDA) and Factor Analysis are also dimension reduction methods. Recently, sparse PCA are also produced to further reduce the dimensionality and aid interpretation Zou et al. (2006); Shen and Huang (2008); Johnstone and Lu (2009).

Another question of interest to researchers is the relationship between several variables. This problem arises in many areas such as social science, chemometrics, biology, and image genetics. For example, Joyner et al. (2009) study relationship between 4 brain size measures and 11 single nucleotide polymorphisms (SNPs). For a high dimensional example, Stein et al. (2010) study the relationship between 31,622 voxels and 448,293 SNPs. Canonical Correlation Analysis proposed by Hotelling (1936) is the one of the popular methods to find the relationship between two sets of variables. Also, grouping similar variables or objects may be the main task of researchers. Logistic Regression, Support Vector Machine (Vapnik and Vapnik, 1998), Discriminant Analysis, and many other clustering or classification methods are proposed for this purpose.

As discussed above, various multivariate methods exist to address different questions. Interestingly several analysis methods such as PCA, CCA, and MLR share same structure. They can be written as the generalized eigenvalue problem. Consider a square matrix $\mathbf{A} \in \mathbb{R}^{n \times n}$. Eigenvalue problem seeks the eigen vector $\mathbf{x} \in \mathbb{R}^n$ and eigen value $\lambda \in \mathbb{R}$ that

satisfies

$$\mathbf{A}\mathbf{x} = \lambda\mathbf{x}.$$

The generalized eigen problem considers two matrices, $\mathbf{A} \in \mathbb{R}^{n\times n}$ and $\mathbf{B} \in \mathbb{R}^{n\times n}$. The goal is to find the generalized eigen vector $\mathbf{x} \in \mathbb{R}^n$ and generalized eigen value $\lambda \in \mathbb{R}$ that satisfies

$$\mathbf{A}\mathbf{x} = \lambda\mathbf{B}\mathbf{x}. \tag{1.1}$$

If the matrix $\mathbf{B}$ is the identity matrix, then the generalized eigenvalue problem is reduced to the eigenvalue problem. This means many multivariate methods can be solved by a unifying algorithm for the generalized eigenvalue problem such as the QZ algorithm (Golub and van Van Loan, 1996). But in many applications, we want to find solutions that satisfy certain constraints. In this thesis, we investigate a unifying approach for solving constrained multivariate analysis.

In Chapter 2, we propose a unifying approach for constrained multivariate analysis methods which is based on the generalized eigenvalue formulation. There are several methods that deal with constrained multivariate analysis such as sparse PCA or sparse CCA, for specific methods under specific constraint. There is no unifying approach to solve various constrained problems yet. We conduct several simulations to show that our approach can be a good way to solve the constrained multivariate problems. Specifically, we consider non-negativity and sparsity constraints in simulations, which are widely used in many applications.

1.2 Tensor Data Analysis

Beyond high dimensionality, data with more than two dimensions have been gathered. Multi-dimensional array, also called a tensor, frequently occurs in many areas such as computer vision, gene expression, engineering, and neuroimaging.

Consider brain imaging data as an example. Electroencephalography (EEG) measures voltage fluctuations of ionic current flows within the neurons of the brain (Niedermeyer and da Silva, 2005). Voltages are recorded by a number of electrodes over a short period at high frequencies. Therefore the data is two dimensional, taking form a matrix. Positron emission tomography (PET) produces a three-dimensional image of functional activities in the body (Valk, 2003). Magnetic resonance images (MRI), which is used to visualize the brain, produces three dimensional data as well. And a neuroimaging procedure using MRI technology, known as functional magnetic resonance images (fMRI), measures brain activity by detecting changes in blood flow (Huettel et al., 2004). This technology produces more than 10^2 MRI images for each subject.

There have been few methods in literature for handling tensor data, in contrast to the enormous literature on high-dimensional or even ultra high dimensional vector data. One approach is to simply vectorize tensor data and then apply existing methods. But this approach has at least two drawbacks. First, simply vectorizing tensor data yields extremely high dimensional data.. For example, MRI data typically has $128 \times 128 \times 128$ voxels and produces more than 2 million variables if we vectorize them. Even worse, fMRI produces 10^2 to 10^3 MRI images for each subject, which is almost impossible to analyze after vectorization. The second problem is even more serious; vectorization looses the spatial information contained in its own structure.

Recently several methods to deal with tensor data without destroying its own struc-

ture have been proposed. In the regression context, matrix regression (Zhou and Li, 2014), CP tensor regression that utilizes CP decomposition of tensor (Zhou et al., 2013), Tucker tensor regression that utilizes Tucker tensor decomposition (Li et al., 2013), and tensor partition regression model (TPRM) (Miranda et al., 2015) have been proposed. These methods utilize tensor decompositions to achieve significant reduction in the number of parameters while preserving the array structure of the data. More recently, Li and Zhang (2015) proposed parsimonious tensor response regression for the data containing vector predictive variables and tensor response variables. To achieve the sparsity of the coefficient, they impose existence of irrelevant linear combination of variables instead of irrelevant individual variables.

Following these, several more approaches are proposed. Tensor sliced inverse regression Ding and Cook (2015b) and higher-order sliced inverse regressions (Ding and Cook, 2015a) deal with tensor dimension reduction problem. Generalized estimating equation for tensor covariates (Zhang et al., 2014), logistic regression for classification of tensor data (Tan et al., 2013), tensor linear support vector machines (Li, 2014) are other proposed statistical analysis methods for tensor data.

Following this stream, we propose novel approaches to conduct CCA and sparse CCA on two tensor data sets in the Chapter 3. Specifically, we achieve significant reduction in the number of parameters using tensor decomposition, like several proposed regression methods introduced above. Also our method preserves the multidimensional structure in the data. And by adopting a parsimonious structures on the covariance matrices, we accomplish both statistical and computational efficiency in the estimation procedure.

1.3 Thesis Organization

The rest of this dissertation is organized in two chapters. In Chapter 2, a unifying approach for finding constrained solution of multivariate analysis is proposed.

There are some proposed algorithms to get a sparse solution of PCA (Zou et al., 2006; Shen and Huang, 2008; Witten et al., 2009; Journée et al., 2010; Allen and Maletić-Savatić, 2011). And three algorithms proposed by Waaijenborg et al.; Parkhomenko et al.; Witten et al. are most widely known methods to get a sparse solution of CCA. However, these methods are specific to the method and constraint and do not generalized to other constraints rather than sparsity.

We focus on the fact that several multivariate analysis such as PCA and CCA can be represented in one simple form, the generalized eigenvalue problem. And the generalized eigen value problem can be written as an optimization problem of a function, namely the Rayleigh quotient. Based on these relationships, we propose to utilize existing constrained optimization algorithms to solve the problem. Thus, we can freely add constraints to any multivariate problem that can be represented as the generalized eigenvalue problem and solve those problems using one numerical algorithm. In specific, we investigate five numerical optimization algorithms and focus on utilizing those algorithms to constrained multivariate analysis problem. In the simulation study, CCA has been considered with non-negativity and sparsity constraint to show the usefulness of this approach. Numerical studies in Chapter 2 also form the basis of sparse tensor CCA in Chapter 3.

In Chapter 3, we consider CCA problem with two tensor data sets. Since data is not an ordinary vector any more, we develop new models that can deal with array structure. We first review the ordinary CCA problem, and sparse CCA proposed by Witten et al. (2009); Chi et al. (2013). Notations and some basic results in tensor linear algebra are

then introduced.

Several tensor CCA methods are proposed next. First model we impose a tensor decomposition structure on coefficients, namely the CP decomposition (Kolda, 2006; Kolda and Bader, 2009), to achieve the reduction in the number of parameters without destroying the structure of the data. Then we propose a more parsimonious model by imposing a separable covariance structure proposed by (Hoff et al., 2011) on covariance matrices of two data sets. Finally we propose a sparse version for both models. Simulation studies are presented to show the usefulness of proposed models.

Chapter 2

Constrained Generalized Eigenvalue Problem

2.1 Introduction

Multivariate analysis techniques such as principal component analysis (PCA), canonical correlation analysis (CCA), partial least squares (PLS), and multivariate linear regression (MLR) are frequently used in various fields such as chemometrics (Höskuldsson, 1988), neuroimaging (Friman et al., 2001, 2003), bioinformatics (Allen and Maletić-Savatić, 2011), and genetics (Chi et al., 2013). It is well known that these multivariate analysis problems can be reformulated as a generalized eigenvalue problem (GEP). This relationship enables us to solve these multivariate statistical problems via algorithms that solve GEP. There exists a well-established algorithm for solving the generalized eigenvalue problem, known as QZ algorithm (Golub and van Van Loan, 1996). But if we add constraints on the problem, we cannot use QZ algorithm any more. New methods to solve those constrained multivariate problems are required. In real world applications,

constraints are frequently added for easier interpretation or better reflection of the data structure. Thus the need for the new methods to solve those constrained problems has increased.

For certain methods and constraints, there exist several proposed approaches to handle the problem. In the audiovisual source separation problem that uses CCA, the non-negativity constraint is required to interpret the result as energy signals (Wold et al., 2001; Sigg et al., 2007). In this context, algorithm for the non-negative CCA has been proposed (Sigg et al., 2007). On the other hand, sparsity is another representative constraint required when we deal with the high dimensional data. We seek a sparse solution that helps us to understand and interpret the underlying phenomena better. There are many proposed algorithms for PCA/CCA with sparsity constraint (Zou et al., 2006; Sigg et al., 2007; Sigg and Buhmann, 2008; Waaijenborg et al., 2008; Kim and Cipolla, 2009; Witten et al., 2009; Hardoon and Shawe-Taylor, 2011). In some applications, both non-negativity and sparsity constraints are simultaneously added (Hoyer, 2004; Zass and Shashua, 2006; Allen and Maletić-Savatić, 2011). Besides these, other constraints could be adopted for the better results or applications.

However, existing techniques are specific to individual multivariate problem with specific constraints, such as sparse PCA or non-negative CCA. We propose a unifying approach to solve the constrained multivariate analysis problems. In an effort to develop this approach, we first focus on the relationship that many popular multivariate statistical problems can be recast as GEP. And then we take the GEP as an optimization problem with a certain objective function, known as the generalized Rayleigh quotient (GRQ) (Borga et al., 1997). Based on these two relationships, we can treat a multivariate statistical analysis as a numerical optimization problem. For constrained estimation, we deal with constrained optimization. This viewpoint enables us to exploit existing numeri-

cal optimization methods to most constrained multivariate problem instead of developing specific algorithm for specific constrained problem.

For constrained optimization, we consider gradient descent, coordinate descent, and alternating direction method of multipliers. In the simulation study, we compare the performance of these algorithms under two most common constraints: non-negativity and sparsity.

This chapter is structured as follows. In Section 2.2, the problem is formally formulated and the relation to multivariate statistical analysis is explained. In Section 2.3, five algorithms for optimization and several algorithms for projection and acceleration are described. In Section 2.4, performance of algorithms is assessed under various situations. Lastly, Section 2.5 contains conclusions.

2.2 Problem

In this section, we provide a precise statement of the problem that will be studied in this chapter. Our approach is based on the relationship between multivariate analysis, the generalized eigenvalue problem, and the optimization problem of the Rayleigh Quotient function. We will describe those relationship and then give a problem statement. First we define the generalized eigenvalue problem.

Problem 1 (Generalized Eigenvalue Problem (GEP))**.** *For two matrices* $\mathbf{A} \in \mathbb{R}^{p\times q}$ *and* $\mathbf{B} \in \mathbb{R}^{p\times q}$*, find a vector* $\mathbf{w} \in \mathbb{R}^q$ *and a constant* $\lambda \in \mathbb{R}$ *that satisfy*

$$\mathbf{A}\mathbf{w} = \lambda\mathbf{B}\mathbf{w}. \tag{2.1}$$

GEP frequently occurs mathematics, physics, and statistics. It is well known that

	$\mathbf{A}$	$\mathbf{B}$	$\mathbf{w}$
PCA	$\mathbf{X}^\intercal\mathbf{X}$	$\mathbf{I}$	$\mathbf{w}$
CCA	$\begin{bmatrix} \mathbf{0} & \mathbf{X}^\intercal\mathbf{Y} \\ \mathbf{Y}^\intercal\mathbf{X} & \mathbf{0} \end{bmatrix}$	$\begin{bmatrix} \mathbf{X}^\intercal\mathbf{X} & \mathbf{0} \\ \mathbf{0} & \mathbf{Y}^\intercal\mathbf{Y} \end{bmatrix}$	$\begin{bmatrix} \mathbf{w_X} \\ \mathbf{w_Y} \end{bmatrix}$
PLS	$\begin{bmatrix} \mathbf{0} & \mathbf{X}^\intercal\mathbf{Y} \\ \mathbf{Y}^\intercal\mathbf{X} & \mathbf{0} \end{bmatrix}$	$\begin{bmatrix} \mathbf{I} & \mathbf{0} \\ \mathbf{0} & \mathbf{I} \end{bmatrix}$	$\begin{bmatrix} \mathbf{w_X} \\ \mathbf{w_Y} \end{bmatrix}$
MLR	$\begin{bmatrix} \mathbf{0} & \mathbf{X}^\intercal\mathbf{Y} \\ \mathbf{Y}^\intercal\mathbf{X} & \mathbf{0} \end{bmatrix}$	$\begin{bmatrix} \mathbf{X}^\intercal\mathbf{X} & \mathbf{0} \\ \mathbf{0} & \mathbf{I} \end{bmatrix}$	$\begin{bmatrix} \mathbf{w_X} \\ \mathbf{w_Y} \end{bmatrix}$

Table 2.1: Reformulation from multivariate problems to GEP. $\mathbf{X}$ and $\mathbf{Y}$ represent observed datasets from same objects. Since PCA deals only one data set, its expression only contains $\mathbf{X}$.

many multivariate statistical methods can be recast as GEP. Above Table 2.1 shows the reformulation from several multivariate analysis problems to GEP.

On the other hand, GEP relates to the function called generalized Rayleigh quotient. The definition of the generalized Rayleigh quotient is as follows.

Definition 1 (Generalized Rayleigh Quotient (GRQ))**.** *Let* $\mathbf{A} \in \mathbb{R}^{n\times n}$ *be symmetric and* $\mathbf{B} \in \mathbb{R}^{n\times n}$ *be symmetric and positive definite. The* Generalized Rayleigh Quotient $R(\mathbf{w})$ *of* $\mathbf{A}$ *and* $\mathbf{B}$ *is the function in* $\mathbf{w} \in \mathbb{R}^n$ *given by*

$$R(\mathbf{w}) = \frac{\mathbf{w}^\intercal\mathbf{A}\mathbf{w}}{\mathbf{w}^\intercal\mathbf{B}\mathbf{w}}. \tag{2.2}$$

Finding a vector $\mathbf{w}$ satisfying the equation (2.1) is the same problem of the optimization of GRQ (De Bie et al., 2005). Without loss of generality, optimizing the equation (2.2)

can be reformulated as the constrained optimization problem of the form:

$$\max_{\mathbf{w}} \ \mathbf{w}^\top \mathbf{A}\mathbf{w} \quad \text{s.t.} \quad \mathbf{w}^\top \mathbf{B}\mathbf{w} = 1.$$

Using the Lagrangian $L(\mathbf{w})$ with the Lagrange multiplier λ, we can reformulate the problem as follows.

$$\max_{\mathbf{w}} L(\mathbf{w}) = \max_{\mathbf{w}} \mathbf{w}^\top \mathbf{A}\mathbf{w} - \lambda \mathbf{w}^\top \mathbf{B}\mathbf{w}.$$

Taking the first derivative and equating that to zero leads to the GEP (2.1).

Remark 1. *The generalized eigenvalue problem (GEP) is closely related to optimizing the generalized Rayleigh Quotient (GRQ)* (2.2)*. The GEP finds a pair* $(\lambda, \mathbf{w})$ *that satisfies the equation* $\mathbf{A}\mathbf{w} = \lambda \mathbf{B}\mathbf{w}$*. Seeking the smallest (or largest) value of* λ *and its paired vector* $\mathbf{w}$ *is equivalent to minimizing (or maximizing)* $R(\mathbf{w})$*.*

Thus, multivariate analysis can be viewed as the optimization of the GRQ with appropriately reformulated matrices from observed datasets. This means we can conduct various multivariate analysis via solving a numerical optimization problem. If we consider a constrained multivariate analysis, we solve constrained numerical optimization problem.

Problem 2 (Constrained Optimization of Generalized Rayleigh Quotient (COGRQ))**.** *Consider two matrices* $\mathbf{A} \in \mathbb{R}^{n \times n}$ *which is symmetric and* $\mathbf{B} \in \mathbb{R}^{n \times n}$ *which is symmetric and positive definite. Find* $\mathbf{w}$ *such that*

$$\mathbf{w} = \operatorname*{argmin}_{\mathbf{w} \in \Delta} R(\mathbf{w}) \tag{2.3}$$

where Δ *is a constrained space.*

Eventually a constrained multivariate analysis is equivalent to calculating two matrices $\mathbf{A}$ and $\mathbf{B}$ from datasets $\mathbf{X}$ and $\mathbf{Y}$ following the Table 2.1, pluging in them in the Problem 2, and solving the problem. This approach enables us to utilize existing numerical optimization techniques instead of developing a new algorithm for each multivariate analysis problem with a specific constraint. Moreover, we can conduct various multivariate analysis under constraints using one numerical algorithm, which is a great advantage.

In this chapter, we will consider two constraints: non-negativity and sparsity. Definitions of constrained spaces are

$$\begin{aligned} \Delta_{nonneg} &= \{w_1 + \cdots + w_n = 1, w_1, \ldots, w_n \geq 0\} \\ \Delta_{sparse} &= \{\mathbf{w} \in \mathbb{R}^n : |w_1| + \cdots + |w_n| \leq \gamma,\ w_1^2 + \cdots + w_n^2 = \lambda\}, \end{aligned}$$

where γ and λ are pre-determined penalty parameters. The constraint set Δ_{nonneg} is used when w_i are intrinsically non-negative in the underlying application. The condition that $w_1 + \cdots + w_n = 1$ is introduced to make the solution identifiable. The constraint set Δ_{sparse} is used when data exhibits high dimensionality so that sparse solution is preferred.

2.3 Algorithms

In this section, we investigate various numerical algorithms for solving constrained optimization of the Generalized Rayleigh Quotient as stated in Problem 2.

We will consider three types of numerical algorithms. The first one is gradient descent typed algorithm. Gradient descent method is the one of the most widely used methods for optimization. Since the convergence of gradient descent can be slow, many variants

have been proposed to overcome this drawback. However, recent demand for the high dimensional data brings the original gradient descent method back to the attention for its simple and intuitive nature. We will consider three kinds of gradient descent based algorithms: accelerated projected gradient, accelerated proximal gradient, and accelerated projected gradient for nonconvex objective function.

The second one is the coordinate descent algorithm. Coordinate descent algorithm is also the one of the most widely used methods, because it is simple and intuitive.

The third one is the alternating direction method of multipliers (ADMM). ADMM algorithm is based on dual decomposition and augmented Lagrangian method, so it enjoys benefits from both methods. Boyd et al. (2011) demonstrates that ADMM effectively solve many constrained or regularized optimization problems arising from big data.

In summary, we will investigate the following five numerical optimization algorithms:

1. Accelerated Projected Gradient Method with Exact Line Search (APGE)
2. Accelerated Proximal Gradient Method with Backtracking Line Search (APGB)
3. Accelerated Projected Gradient Method for NonConvex Objective Function (APGNC)
4. Coordinate Descent Method (CD)
5. Alternating Direction Method of Multipliers (ADMM)

Each of these algorithms needs projecting a point to a constraint set Δ. We describe projection algorithms for two type of constraints sets:

6. Projection to Δ_{nonneg}
7. Projection to Δ_{sparse}

Pseudo code for each algorithm is provided in the Appendix A.

2.3.1 Accelerated Projected Gradient Method with Exact Line Search (APGE)

Let $\mathbf{w}^{(t)}$ be a current iterate and $\mathbf{w}^{(t+1)}$ be the next iterate. And let $\nabla R(\mathbf{w})$ denote the gradient of the objective function $R(\cdot)$ at the point $\mathbf{w}$,

$$\nabla R(\mathbf{w}) = \frac{2}{\mathbf{w}^\intercal \mathbf{B}\mathbf{w}}[\mathbf{A}\mathbf{w} - R(\mathbf{w})\mathbf{B}\mathbf{w}]. \tag{2.4}$$

From the current iterate $\mathbf{w}^{(t)}$, gradient descent moves along the direction of $-\nabla R(\mathbf{w}^{(t)})$ with stepsize s such that next objective value $R(\mathbf{w}^{(t+1)})$ is less than $R(\mathbf{w}^{(t)})$.

We can perform the gradient descent with exact line search for the stepsize s, an idea dating back to Hestenes and Karush (1951a,b). Define $\mathbf{u} = \mathbf{w}^{(t)}$ and $\mathbf{v} = \nabla R(\mathbf{w}^{(t)}) = \frac{2}{\mathbf{w}^{(t)\intercal}\mathbf{B}\mathbf{w}^{(t)}}[\mathbf{A} - R(\mathbf{w}^{(t)})\mathbf{B}]\mathbf{w}^{(t)}$. We search along the line $c \mapsto \mathbf{u} + c\mathbf{v}$ emanating from $\mathbf{u}$. Then the Rayleigh quotient

$$R(\mathbf{u} + c\mathbf{v}) = \frac{(\mathbf{u} + c\mathbf{v})^\intercal \mathbf{A}(\mathbf{u} + c\mathbf{v})}{(\mathbf{u} + c\mathbf{v})^\intercal \mathbf{B}(\mathbf{u} + c\mathbf{v})}$$

reduces to a ratio of two quadratics in c. The optimal points are found by setting the derivative

$$2\frac{(\mathbf{v}^\intercal \mathbf{A}\mathbf{u} + c\mathbf{v}^\intercal \mathbf{A}\mathbf{v})(\mathbf{u} + c\mathbf{v})^\intercal \mathbf{B}(\mathbf{u} + c\mathbf{v}) - (\mathbf{u} + c\mathbf{v})^\intercal \mathbf{A}(\mathbf{u} + c\mathbf{v})(\mathbf{v}^\intercal \mathbf{B}\mathbf{u} + c\mathbf{v}^\intercal \mathbf{B}\mathbf{v})}{[(\mathbf{u} + c\mathbf{v})^\intercal \mathbf{B}(\mathbf{u} + c\mathbf{v})]^2}$$

with respect to c to zero and solving for c. Conveniently the coefficients of c^3 in the

numerator of this rational function cancel. This leaves a quadratic with coefficients

$$\begin{aligned}
&(\mathbf{v}^\intercal \mathbf{A}\mathbf{v})(\mathbf{u}^\intercal \mathbf{B}\mathbf{v}) - (\mathbf{u}^\intercal \mathbf{A}\mathbf{v})(\mathbf{v}^\intercal \mathbf{B}\mathbf{v}) \\
&(\mathbf{v}^\intercal \mathbf{A}\mathbf{v})(\mathbf{u}^\intercal \mathbf{B}\mathbf{u}) - (\mathbf{u}^\intercal \mathbf{A}\mathbf{u})(\mathbf{v}^\intercal \mathbf{B}\mathbf{v}) \\
&(\mathbf{u}^\intercal \mathbf{A}\mathbf{v})(\mathbf{u}^\intercal \mathbf{B}\mathbf{u}) - (\mathbf{u}^\intercal \mathbf{A}\mathbf{u})(\mathbf{u}^\intercal \mathbf{B}\mathbf{v})
\end{aligned} \tag{2.5}$$

for c^2, c^1, and c^0 respectively. These coefficients of the powers of c can be evaluated by matrix-vector and inner product operations alone. No matrix-matrix operations are needed. One of the root yields the steepest descent. The unrestricted steepest descent iterate may violate the constraint and has to be projected back to Δ, the constrained space. Projected gradient descent method iterates the two step process (steepest descent and projection to the constrained space) until convergence.

It may suffer from slow convergence but can be readily accelerated by the Nesterov method (Nesterov, 1983, 2004). The Nesterov algorithm accelerates ordinary gradient descent by making extrapolation based on the previous two iterates $\mathbf{w}^{(t-1)}$ and $\mathbf{w}^{(t)}$. Without extrapolation, Nesterov method collapses to a gradient method with the slow non-asymptotic convergence rate of $O(t^{-1})$ rather than $O(t^{-2})$. Remarkably, the Nesterov method requires essentially the same computational cost per iteration as the unaccelerated gradient method. The overall algorithm is summarized in Algorithm 1.

2.3.2 Accelerated Proximal Gradient Method with Backtracking Line Search (APGB)

Consider the following problem,

Problem 3. *Consider two matrices* $\mathbf{A} \in \mathbb{R}^{n\times n}$ *which is symmetric and* $\mathbf{B} \in \mathbb{R}^{n\times n}$ *which*

is symmetric and positive definite. Find $\mathbf{w}$ *that*

$$\textit{minimize}\ \ R^{\dagger}(\mathbf{w}) = R(\mathbf{w}) + \mathbf{I}_{\Delta}(\mathbf{w}) = \frac{\mathbf{w}^{\intercal}\mathbf{A}\mathbf{w}}{\mathbf{w}^{\intercal}\mathbf{B}\mathbf{w}} + \mathbf{I}_{\Delta}(\mathbf{w}), \tag{2.6}$$

where $\mathbf{I}_{\Delta}(\mathbf{w}) = \begin{cases} 0 & \mathbf{w} \in \Delta \\ \infty & \mathbf{w} \notin \Delta \end{cases}$ *and* Δ *is the constrained space.*

We can easily notice that above problem is same as the Problem 2. $R^{\dagger}(\mathbf{w})$ is non-differentiable due to $\mathbf{I}_{\Delta}$, making it impossible to apply the gradient descent algorithm. Proximal gradient method can be considered as a remedy when the objective function is non-differentiable like in (2.6). At each iteration, we minimize the sum of $\mathbf{I}_{\Delta}(\mathbf{w})$ and the first order approximation of $R(\mathbf{w})$ at some given point $\boldsymbol{\zeta}$

$$Q(\mathbf{w}, \boldsymbol{\zeta}) = R(\boldsymbol{\zeta}) + \langle \mathbf{w} - \boldsymbol{\zeta}, \nabla R(\boldsymbol{\zeta}) \rangle + \frac{1}{2s}\|\mathbf{w} - \boldsymbol{\zeta}\|^2 + \mathbf{I}_{\Delta}(\mathbf{w}), \tag{2.7}$$

instead of the sum of $\mathbf{I}_{\Delta}(\mathbf{w})$ and $R(\mathbf{w})$. Through several lines of calculations it can be shown that the next iterate $\mathbf{w}^{(t+1)}$ satisfying (2.7) is

$$\mathbf{w}^{(t+1)} = \underset{\mathbf{w}}{\operatorname{argmin}} \left\{ \mathbf{I}_{\Delta}(\mathbf{w}) + \frac{1}{2s^{(t+1)}}\|\mathbf{w} - (\mathbf{w}^{(t)} - s^{(t+1)}\nabla R(\mathbf{w}^{(t)}))\|^2 \right\}. \tag{2.8}$$

See details in Beck and Teboulle (2009). Expression (2.8) has two parts. Second term in the expression, $\mathbf{w}^{(t)} - s^{(t+1)}\nabla R(\mathbf{w}^{(t)})$, is identical with gradient descent procedure. First term, $\mathbf{I}_{\Delta}$ leads us to project the point $\mathbf{w}^{(t)} - s^{(t+1)}\nabla R(\mathbf{w}^{(t)})$ to the constrained set. Thus finding $\mathbf{w}^{(t+1)}$ satisfying (2.8) is same as conducting gradient descent on $R(\cdot)$ at current iterate and then project that point to the constrained space.

For the stepsize selection, we employ backtracking line search rule. It starts with $s^{(t)}$ and keeps reducing the stepsize by multiplying a pre-determined positive constant $\eta < 1$

until $R^{\dagger}(\mathbf{w}^{(t+1)})$ becomes smaller than $Q(\mathbf{w}^{(t+1)}, \mathbf{w}^{(t)})$.

Like APGE, Nesterov acceleration procedure can be applied to accelerate the algorithm. The overall algorithm is summarized as Algorithm 2 in Appendix A.

2.3.3 Accelerated Projected Gradient Method for NonConvex Objective Function (APGNC)

Accelerated projected gradient method for nonconvex objective function (APGNC) is recently proposed by Ghadimi and Lan (2013) to improve the convergence properties of existing gradient descent based methods. APGE and APGB guarantee global convergence and optimal convergence rate only if the objective function is convex. However, if the objective function is not convex, we cannot guarantee that found solution has the global minimum function value. Even worse, the convergence result becomes weakened relative to the convex case. APGNC is devised to overcome the convergence issue. It guarantees the optimal convergence rate even when the problem is nonconvex.

Consider following composite problem for variable $\boldsymbol{w} \in \mathbb{R}^n$

$$\begin{aligned} &\text{minimize} \quad \Psi(\mathbf{w}) + \chi(\mathbf{w}) \\ &\text{where} \quad \begin{cases} \Psi(\mathbf{w}) = f(\mathbf{w}) + h(\mathbf{w}) \\ f \in \mathcal{C}^{1,1}_{L_f}(\mathbb{R}^n), \text{ possibly nonconvex} \\ h \in \mathcal{C}^{1,1}_{L_h}(\mathbb{R}^n), \text{ possibly convex} \\ \chi \text{ is a simple convecx function with bounded domain.} \end{cases} \end{aligned} \tag{2.9}$$

Notice that substituting $\Psi(\mathbf{w})$ as $R(\mathbf{w})$ and $\chi(\mathbf{w})$ as $\mathbf{I}_{\Delta}(\mathbf{w})$ and setting $h(\cdot)$ to zero results in the Problem 2.6. Thus we can utilize APGNC for solving the constrained eigenvalue problem 2. APGNC algorithm keeps iterating following three steps of updating

procedures,

$$\begin{aligned}
&\mathbf{w}_{md}^{(t)} \leftarrow (1-\alpha^{(t)})\mathbf{w}_{ag}^{(t-1)} + \alpha^{(t)}\mathbf{w}^{(t-1)} \\
&\mathbf{w}^{(t)} \leftarrow \mathrm{argmin}_{\mathbf{w}}\{\tfrac{1}{\lambda^{(t)}}\|\mathbf{w} - (\mathbf{w}^{(t-1)} - \lambda^{(t)}\nabla\Psi(\mathbf{w}_{md}^{(t)}))\|^2 + \chi(\mathbf{w})\} \\
&\mathbf{w}_{ag}^{(t)} \leftarrow \mathrm{argmin}_{\mathbf{w}}\{\tfrac{1}{\beta^{(t)}}\|\mathbf{w} - (\mathbf{w}_{md}^{(t)} - \beta^{(t)}\nabla\Psi(\mathbf{w}_{md}^{(t)}))\|^2 + \chi(\mathbf{w})\}.
\end{aligned} \tag{2.10}$$

APGNC has two more variables $\mathbf{w}_{ag}^{(t)}$, $\mathbf{w}_{md}^{(t)}$, which are updated iteratively together with $\mathbf{w}^{(t)}$. We can notice that first step for updating $\mathbf{w}_{md}^{(t)}$ resembles Nesterov acceleration, since this step extrapolates two points. Like APGB, the second and third steps have an indicator function that projects to the constrained space Δ.

It is important to choose appropriate parameters used in each steps, α, λ and β in APGNC. Authors suggest the following values for the ideal result under the assumption that we know L_R, the Lipschitz constant of $R(\cdot)$,

$$\alpha^{(t)} = 2/(t+1), \quad \beta^{(t)} = 1/2L_R, \quad \lambda^{(t)} \in [\beta, (1+\alpha^{(t)}/4)\beta]. \tag{2.11}$$

The overall algorithm is summarized as Algorithm 3 in Appendix A.

2.3.4 Coordinate Descent Method (CD)

Coordinate descent algorithm alternately updates w_i subject to the constraint. When updating w_i with other coordinates fixed, the scalar objective function is a quadratic function over another quadratic function. The extrema occurs either at a stationary point or the boundary. Setting partial derivative with respect to w_i, i.e., the i-th component of

the gradient vector (2.4), to zero yields a scalar quadratic function with coefficients

$$
\begin{aligned}
& a_{ii}\left(\sum_{j\neq i} b_{ij}w_j\right) - b_{ii}\left(\sum_{j\neq i} a_{ij}w_j\right), \\
& a_{ii}\left(\sum_{j,j'\neq i} b_{jj'}w_j w_{j'}\right) - b_{ii}\left(\sum_{j,j'\neq i} a_{jj'}w_j w_{j'}\right), \\
& \left(\sum_{j\neq i} a_{ij}w_j\right)\left(\sum_{j,j'\neq i} b_{jj'}w_j w_{j'}\right) - \left(\sum_{j\neq i} b_{ij}w_j\right)\left(\sum_{j,j'\neq i} a_{jj'}w_j w_{j'}\right)
\end{aligned}
\tag{2.12}
$$

for w_i^2, w_i^1, and w_i^0 respectively. Iteration superscripts on w_j and $w_{j'}$ are omitted for simplicity in the above display. Denote the real roots (if any) of above quadratic equation (2.13) by r_1 and r_2. Coordinate w_i is updated by whichever among the possible candidates gives the smallest objective value $R(\mathbf{w})$. Possible candidates are not the raw value of r_1, r_2 from the quadratic equation with coefficients (2.12), determined by considering constrained space and its boundary. For example, if we think non-negativity constraint, there are four possible candidates we need to consider 0, $\min(\max(r_1, 1), -1)$, $\min(\max(r_2, 1), -1)$, and 1. After finishing the cycle for updating all coordinates, check whether $\mathbf{w}$ remains in the constraint set and project the point if it places outside of the constraint set. The overall algorithm is summarized as Algorithm 4 in Appendix A.

2.3.5 Alternating Direction Method of Multipliers (ADMM)

ADMM is originally proposed by Gabay and Mercier (1976).

Problem 4 (Problem of ADMM)**.** *Given matrices* $\mathbf{A} \in \mathbb{R}^{p\times n}$, $\mathbf{B} \in \mathbb{R}^{p\times m}$ *and a vector*

$\mathbf{c} \in \mathbb{R}^p$. *Find* $\mathbf{w} \in \mathbb{R}^n$ *and* $\mathbf{z} \in \mathbb{R}^m$ *such that*

$$\begin{aligned} &\textit{minimize} \quad f(\mathbf{w}) + g(\mathbf{z}) \\ &\textit{subject to} \quad \boldsymbol{A}\mathbf{w} + \boldsymbol{B}\mathbf{z} = \mathbf{c}. \end{aligned} \tag{2.13}$$

ADMM algorithm keeps iterating following three steps,

$$\begin{aligned} \mathbf{w}^{(t+1)} &= \underset{\mathbf{w}}{\operatorname{argmin}}\, L_\rho(\mathbf{w}, \mathbf{z}^{(t)}, \boldsymbol{u}^{(t)}) \\ \mathbf{z}^{(t+1)} &= \underset{\mathbf{z}}{\operatorname{argmin}}\, L_\rho(\mathbf{w}^{(t+1)}, \mathbf{z}, \boldsymbol{u}^{(t)}) \\ \boldsymbol{u}^{(t+1)} &= \boldsymbol{u}^k + \rho(\boldsymbol{A}^{(t+1)} + \boldsymbol{B}\mathbf{z}^{(t+1)} - \mathbf{c}), \end{aligned}$$

with $L_\rho(\mathbf{w}, \mathbf{z}, \boldsymbol{u}) = f(\mathbf{w}) + g(\mathbf{z}) + \boldsymbol{u}^\top(\boldsymbol{A}\mathbf{w} + \boldsymbol{B}\mathbf{z} - \mathbf{c}) + (\rho/2)\|\boldsymbol{A}\mathbf{w} + \boldsymbol{B}\mathbf{z} - \mathbf{c}\|_2^2$ and $\rho > 0$. Problem 2 can be reformulated in the form of the Problem 4 as

$$\begin{aligned} &\text{minimize} \quad R(\mathbf{w}) + g(\mathbf{z}) = \tfrac{\mathbf{w}^\top \boldsymbol{A}\mathbf{w}}{\mathbf{w}^\top \boldsymbol{B}\mathbf{w}} + I_\Delta(\mathbf{z}) \\ &\text{subject to} \quad \mathbf{w} - \mathbf{z} = \mathbf{0} \\ &\text{where} \qquad I_\Delta(\mathbf{z}) = \begin{cases} 0 & \mathbf{z} \in \Delta \\ \infty & \mathbf{z} \notin \Delta \end{cases} \\ &\qquad\qquad \Delta : \text{constrained space.} \end{aligned} \tag{2.14}$$

Thus we can utilize ADMM algorithm to solve Problem 2. The ADMM algorithm updates

$\mathbf{w}, \mathbf{z}$ and $\mathbf{u}$ alternately

$$\begin{aligned}
\mathbf{w}^{(t+1)} &\leftarrow \underset{\mathbf{w}}{\operatorname{argmin}} \left\{ R(\mathbf{w}) + \frac{1}{2\lambda} \|\mathbf{w} - \mathbf{z}^{(t)} + \mathbf{u}^{(t)}\|_2^2 \right\} && (2.15) \\
\mathbf{z}^{(t+1)} &\leftarrow P_{\Delta}\left(\mathbf{w}^{(t+1)} + \mathbf{u}^{(t)}\right) && (2.16) \\
\mathbf{u}^{(t+1)} &\leftarrow \mathbf{u}^{(t)} + \mathbf{w}^{(t+1)} - \mathbf{z}^{(t+1)}. && (2.17)
\end{aligned}$$

We can implement the first step (2.15) using any reasonable methods. In this paper we choose APGB introduced in Section 2.3.2. The overall algorithm is summarized as Algorithm 5 in Appendix A.

2.3.6 Projection to Δ_{nonneg}

Recall that

$$\Delta_{nonneg} = \{w_1 + \cdots + w_n = 1, w_1, \ldots, w_n \geq 0\}.$$

The projection to Δ_{nonneg} can be efficiently carried out by employing Michelot algorithm (Michelot, 1986). Notice that any points in Δ_{nonneg} satisfy two conditions. Michelot algorithm alternatively projects the point onto the constrained space, the first constraint is required or the second one is required. Then it removes coordinates satisfying both constraints from the consideration of the next iterate. Michelot algorithm converges in at most n iterations and often much sooner because every iteration reduces the dimension n by at least one. The overall algorithm is summarized as Algorithm 6 in Appendix A.

2.3.7 Projection to Δ_{sparse}

Recall that

$$\Delta_{sparse} = \{\boldsymbol{w} \in \mathbb{R}^n : |w_1| + \cdots + |w_n| \leq \gamma,\ w_1^2 + \cdots + w_n^2 = \lambda\}.$$

It is trivial that if $\lambda \geq \sqrt{2}\gamma$, this projection problem is reduced to a projection to l_2 ball. Projection to l_2 ball can be done by rescaling, as described in Algorithm 10 in the Appendix A. In a similar way, if $\lambda \leq \gamma$, then this projection is reduced to a projection to l_1 ball. Since sign of the projected point should have same sign of $\mathbf{w}$, we only consider the absolute value of w_i, $i = 1, \ldots, n$. Projection of $|\mathbf{w}|$ to l_1 ball can be done by Michelot's algorithm when $\gamma = 1$. If γ is bigger than 1, we can use the projection to the simplex $\Delta_{simplex} = \{\boldsymbol{w} \in \mathbb{R}^n : \sum w_i = \gamma,\ w_i \geq 0,\ i = 1, \ldots, n\}$. This projection algorithm is summarized in Algorithm 8. After projection of $|\mathbf{w}|$ we can find the projected point of $\mathbf{w}$ onto l_1 ball by restoring their sign. Overall projection to l_1 ball is summarized in Algorithm 9. Lastly, if $\gamma < \lambda < \sqrt{2}\gamma$, then we can use Dykstra algorithm (Boyle and Dykstra, 1986) summarized in Algorithm 11. This algorithm projects the point to the intersection of convex sets. The overall algorithm is summarized in Algorithm 7.

2.4 Experimental Results

We conduct three simulations with synthetic data under one of two constraints, non-negativity or sparsity. We simulated generic and CCA problem, but note that other multivariate analysis methods that can be transformed to the generalized eigenvalue problem are subject to our algorithms as well.

The choice of a starting point is an important issue for the optimization problem. In

practice the projected generalized eigenvector to constrained space corresponding to the smallest generalized eigenvalue supplies a good starting point $\boldsymbol{x}^{(0)}$. In all three simulations, we use this as a starting point.

First two simulations are conducted under non-negativity constraint and the last one under the sparsity constraint. In each of following subsections, synthetic data description will be introduced, then the results and discussion will be followed.

2.4.1 Non-negativity Constrained Problem for Inputs with Known Exact Solutions

The goal of the first simulation is to show that our approach is useful even though numerical algorithms generally cannot guarantee to find the global maximum if the objective function is nonconvex. To show the accuracy of algorithms, we will compare the analytic solution and the results from five numerical algorithms introduced in previous section.

For some specific $\mathbf{A}$ and $\mathbf{B}$ matrices, there is a known analytic solution of Problem 2 with non-negativity constraint. Consider a sequence of n quantiles $\tau_i = \frac{i}{n+1}$, $i = 1, \ldots, n$. F and f be the distribution function and density function in order. In the weighted composite quantile regression context, the estimator of the model has asymptotic covariance $\frac{\mathbf{w}^\top \mathbf{A}\mathbf{w}}{\mathbf{w}^\top \mathbf{B}\mathbf{w}} \boldsymbol{\Sigma}_{\mathbf{X}}^{-1}$ where $\mathbf{w}$ is a weight vector for n quantiles, $\mathbf{X}$ is covariate, $\Sigma_{\mathbf{X}}$ is covariance matrix of $\mathbf{X}$, and $\mathbf{A}$, $\mathbf{B}$ is

$$\begin{aligned} \mathbf{A}_{i,j} &= \min(\tau_i, \tau_j) - \tau_i \tau_j \\ \mathbf{B}_{i,j} &= f(F^{-1}(\tau_i)) f(F^{-1}(\tau_j)). \end{aligned} \tag{2.18}$$

Zhao and Xiao (2011) have shown that under some assumptions on the error distri-

bution f and F, there exists a closed form solution $\mathbf{w}^* \in \Delta_{nonneg}$. We compared this analytic solution with the results from our five suggested algorithms in terms of runtime and difference between analytic solution and optimized value. For the simulation, we considered $n = 3, \ldots, 140$ which means dimension of $\mathbf{A}$ and $\mathbf{B}$, and the standard normal distribution is used as the error distribution. Tolerance value for stopping criteria was 10^{-4}, and maximum iterations limit was set as $10,000$. For backtracking line search parameter value $\eta = 0.5$ is used, while penalty parameter $\rho = 2$ is used in ADMM algorithm. We have three parameters in APGNC, and the Lipschitz constant L_R is required to get an ideal result on APGNC, which is actually unknown. We used a rough guess of L_R according to the value of each dimension n. At last, we simulate 50 replicates for each n so that we calculate mean and standard deviation of time consumed.

Following two figures show the results. Figure 2.1 shows the mean and standard deviation of the runtime for each dimension. In this figure, we first notice that all algorithms run fast. For the biggest dimension $n = 140$, we can get the solution within 30 seconds at maximum. All algorithms except APGNC show fast speed.

Figure 2.2 shows differences between analytic solution and results from each of the five algorithms. Except APGNC, all other four algorithms find exact solution, the difference between the analytic solution and results from algorithms are less than 10^{-5}. APGNC show less accuracy relative to other four algorithms.

Note that we used a roughly guessed parameter values for β and λ due to the lack of information about L_R. Thus there is a possibility to get better result on APGNC in terms of the speed and accuracy, if we find better guessed value for L_R. Except APGNC, all four algorithms give us accurate results within very short time.

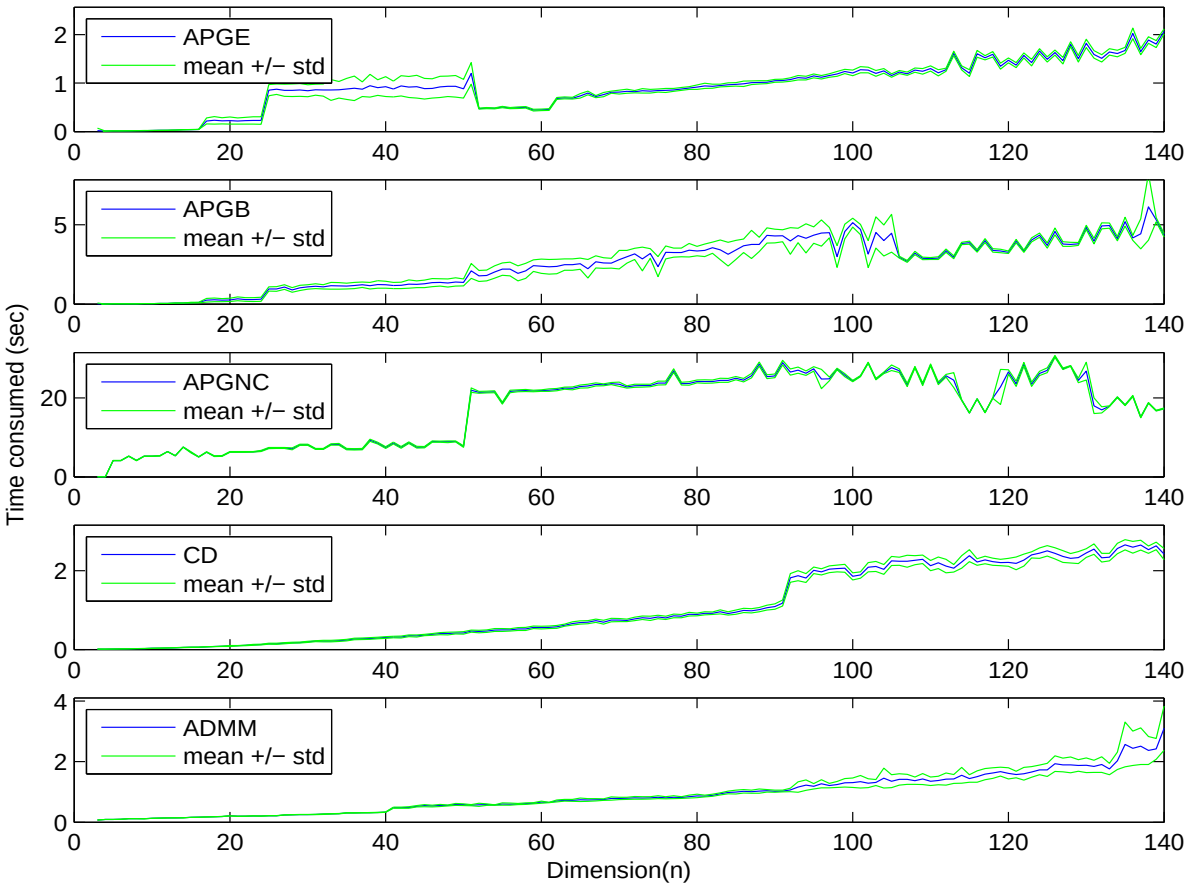


Figure 2.1: Time consumed for getting result from five algorithms.

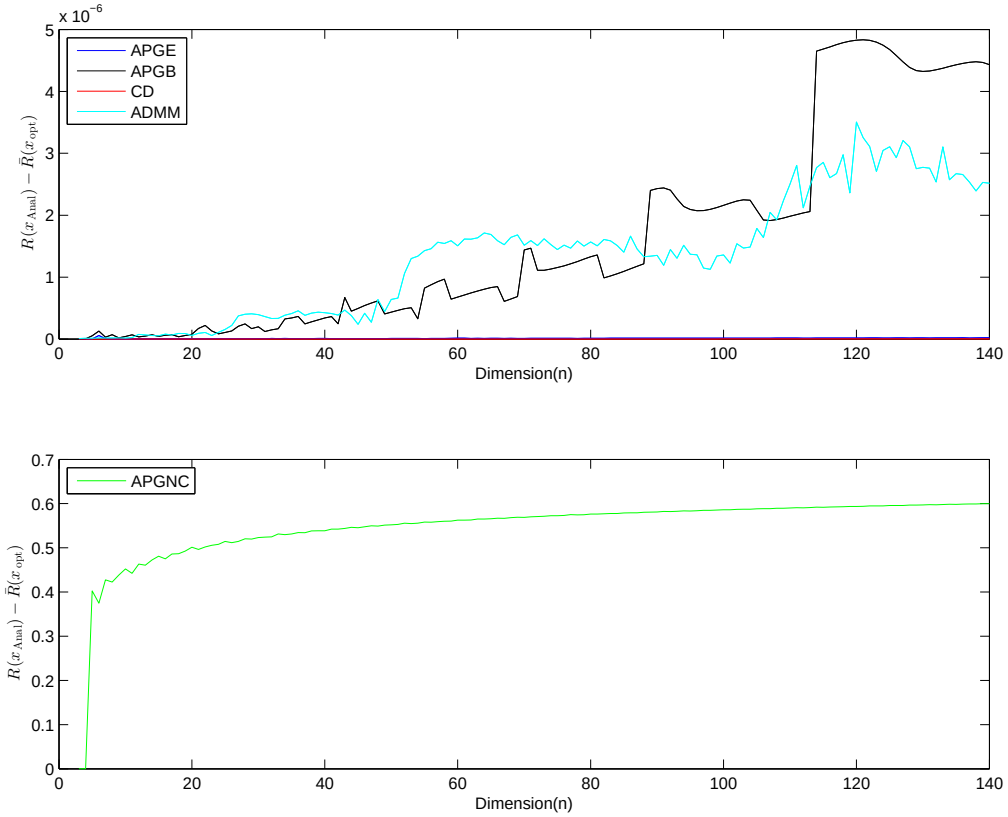


Figure 2.2: Plots for comparison between analytical solution and optimal solution from four algorithms.

2.4.2 Non-negativity Constrained Problem for Randomly Generated Inputs

The analytic solution is known for the specific case. We do not have an exact expression of the solution most of the time. To assess the performance of five algorithms, we generated random matrices $\mathbf{A} \in \mathbb{R}^{n \times n}$ and $\mathbf{B} \in \mathbb{R}^{n \times n}$

$$
\begin{aligned}
A^{\dagger}_{i,j} &= \begin{cases} \text{generated from } N(0,1) & i \geq j \\ A^{\dagger}_{j,i} & \text{otherwise} \end{cases} \\
\mathbf{A} &= \mathbf{A}^{\dagger} + \boldsymbol{\epsilon} \\
B^{\dagger}_{i,j} &\sim N(0,1) \\
\mathbf{B} &= \mathbf{B}^{\dagger\intercal}\mathbf{B}^{\dagger} + \boldsymbol{\epsilon},
\end{aligned}
\tag{2.19}
$$

where $\epsilon_{i,j} \sim N(0, 0.3^2)$. We considered dimensions $n =$ 3, 20, 50, 70, and 100. For each dimension, we simulated 500 replicates and recorded runtime, the numbers of iterations, and final objective values. Same values as the first simulation in Section 2.4.1 has been used for η, maximum iteration limit, tolerance, ρ, and L_R.

Simulation results are summarized as Figure 2.3. The first row of the Figure 2.3 shows the box plots of final objective values from each algorithms. There were not much differences of final objective value between algorithms except APGNC. Especially APGE, APGB, and CD show almost identical distribution. This implies that four algorithms find local minimum well. Efficiency can be compared in following rows of the Figure 2.3. APGB, and CD algorithms seem to be a good choice for solving this problem, since they cost the least time to calculate from low to high dimensional problem while they converge within not many iterations. APGE can also be a good choice, since it gives a solution within very short time, but it reaches maximum iteration limits to get that solution.

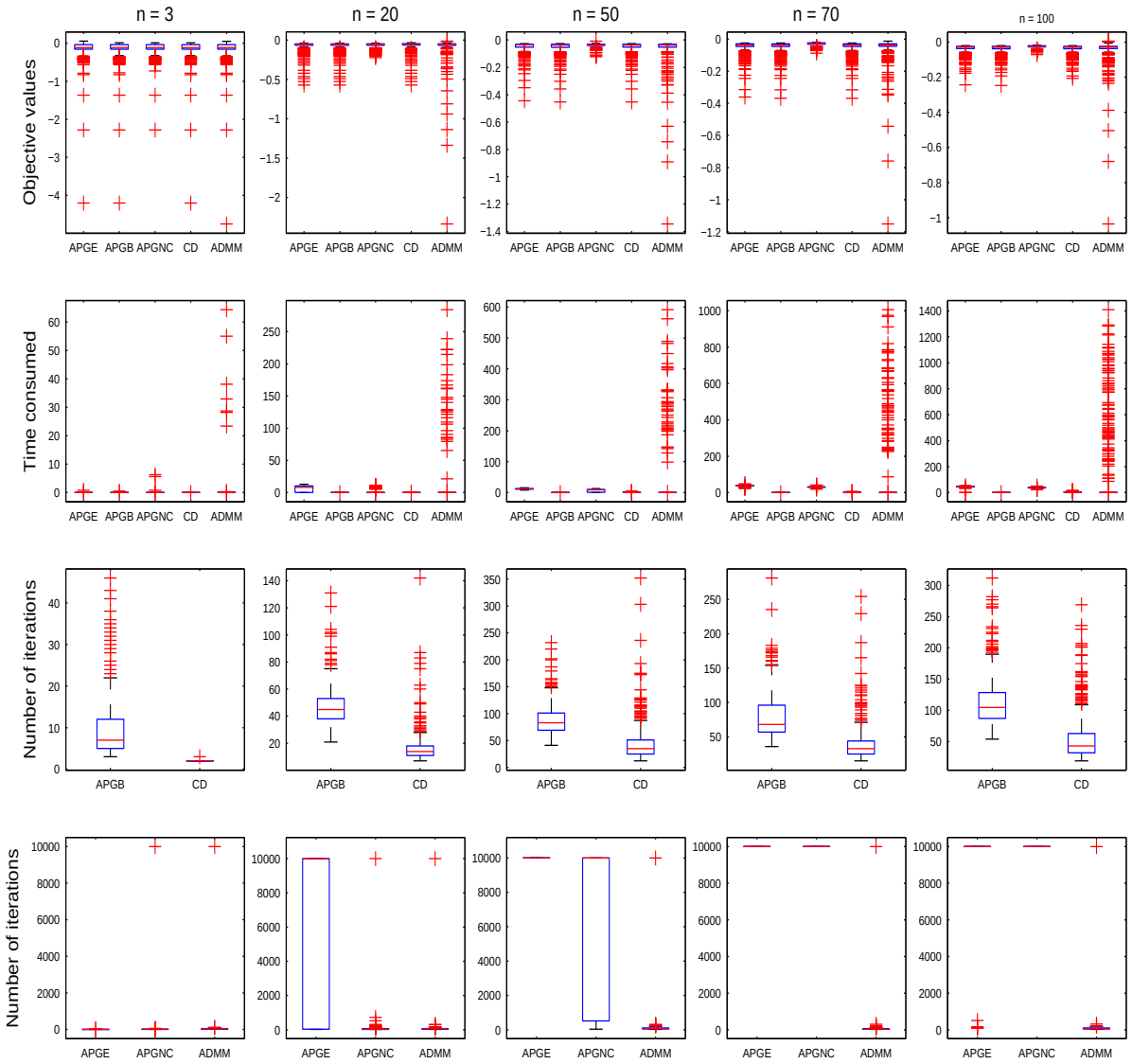


Figure 2.3: Boxplot of final optimized value, time consumed (in sec), and total number of iterations under $n = (3, 20, 50, 70, 100)$ for five algorithms. For the number of iterations, we draw separate boxplots for APGE, APGNC, and ADMM in the fourth row.

2.4.3 Sparsity Constrained Problem for Randomly Generated CCA Inputs

In the third simulation, we compare four algorithms, APGE, APGB, ADMM and CD for solving the sparsity constrained CCA problem. Assume that we have data matrices $\mathbf{X}$ and $\mathbf{Y}$. Then our problem is

$$\text{minimize} \quad -\frac{\mathbf{w}^\top \mathbf{A}\mathbf{w}}{\mathbf{w}^\top \mathbf{B}\mathbf{w}} \quad \text{subject to} \quad \begin{cases} \|\boldsymbol{\alpha}\|_1 \le \gamma_1, & \|\boldsymbol{\beta}\|_1 \le \gamma_2 \\ \|\boldsymbol{\alpha}\|_2 \le 1, & \|\boldsymbol{\beta}\|_2 \le 1 \end{cases}$$

where $\mathbf{w} = (\boldsymbol{\alpha}^\top, \boldsymbol{\beta}^\top)^\top$, $\boldsymbol{\alpha}$ is the canonical coefficient for $\mathbf{X}$, $\boldsymbol{\beta}$ is the canonical coefficient for $\mathbf{Y}$, γ_1, γ_2 are pre-specified penalty parameters, and $\mathbf{A}, \mathbf{B}$ are matrices calculated from $\mathbf{X}, \mathbf{Y}$ as described in the Table 2.1. Note that we put a negative sign in front of the objective function, since the goal of the CCA is finding a pair of canonical vectors that maximizes the canonical correlation. We can find a pair of sparse canonical coefficients $(\boldsymbol{\alpha}^*, \boldsymbol{\beta}^*)$ that maximizes the canonical correlation $\text{Corr}(\boldsymbol{\alpha}^\top \mathbf{X}, \boldsymbol{\beta}^\top \mathbf{Y})$.

We first generated the data $\mathbf{X} \in \mathbb{R}^{n \times p}$ and $\mathbf{Y} \in \mathbb{R}^{n \times q}$,

$$\mathbf{X} = \mathbf{u}\mathbf{v}_1^\top + \boldsymbol{\epsilon}_1, \quad \mathbf{Y} = \mathbf{u}\mathbf{v}_2^\top + \boldsymbol{\epsilon}_2, \tag{2.20}$$

$$\mathbf{u}_i \sim N(0, 0.5^2), \quad \boldsymbol{\epsilon}_{1,i,j} \sim N(0, 0.3^2), \quad \boldsymbol{\epsilon}_{2,i,k} \sim N(0, 0.3^2),$$
$$i = 1, \ldots, n, \qquad j = 1, \ldots, p, \qquad k = 1, \ldots, q.$$

Sparse vectors $\boldsymbol{v}_1$ and $\boldsymbol{v}_2$ in equation (2.20) used under each combination are presented in Table 2.2. Four combinations of n and p are considered in this simulation. In order to compare how well algorithms work when the dimension of variables is much bigger than sample size, our design includes such combinations. Simulations are conducted under each

combination of n and p with 200 repetitions. Parameter values for η, maximum number of

	$p = 20$	$p = 100$
$n = 50$	$\boldsymbol{v}_1 = [\mathbf{1}_5^\top \ -\mathbf{1}_5^\top \ \mathbf{0}_{10}^\top]^\top$ $\boldsymbol{v}_2 = [\mathbf{0}_{10}^\top \ \mathbf{1}_5^\top \ -\mathbf{1}_5^\top]^\top$	$\boldsymbol{v}_1 = [\mathbf{1}_{10}^\top \ -\mathbf{1}_{10}^\top \ \mathbf{0}_{80}^\top]^\top$ $\boldsymbol{v}_2 = [\mathbf{0}_{80}^\top \ \mathbf{1}_{10}^\top \ -\mathbf{1}_{10}^\top]^\top$
	$p = 50$	$p = 300$
$n = 200$	$\boldsymbol{v}_1 = [\mathbf{1}_{10}^\top \ -\mathbf{1}_{10}^\top \ \mathbf{0}_{30}^\top]^\top$ $\boldsymbol{v}_2 = [\mathbf{0}_{30}^\top \ \mathbf{1}_{10}^\top \ -\mathbf{1}_{10}^\top]^\top$	$\boldsymbol{v}_1 = [\mathbf{1}_{10}^\top \ -\mathbf{1}_{10}^\top \ \mathbf{2}_{30}^\top \ \mathbf{0}_{250}]^\top$ $\boldsymbol{v}_2 = [\mathbf{0}_{250}^\top \ \mathbf{1}_{10}^\top \ \mathbf{2}_{30}^\top \ -\mathbf{1}_{10}^\top]^\top$

Table 2.2: True values of $\boldsymbol{v}_1$ and $\boldsymbol{v}_2$ for generating synthetic data under each combination of n and p.

iterations, tolerance, and ρ were set to the same values in other simulations. And we need two more, sparsity parameters γ_1 and γ_2 that impose sparsity on $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$ respectively. We set same value for both γ_1 and γ_2, (1, 2.5, 4, 5) has been used for each combination of n and p. Simulation results are summarized as boxplots in the Table 2.3. We can make several observations from results in the Table 2.3. First, APGE and APGB find coefficient sets that result in high objective values. Their results distributed near 1, which means the optimized canonical coefficients give us a high canonical correlation. Moreover, both algorithms do not take much time relative to the other two methods. Especially, APGB shows the fastest speed with good objective values. CD algorithm shows good performance only if number of parameters is less than the sample size. Though it takes more time relative to APGE and APGB, still it gives us good objective values when $p \leq n$. ADMM seems to be a bad choice for solving sparse CCA. It is a known behavior of ADMM that it shows slow convergence for high accuracy but it shows fast convergence

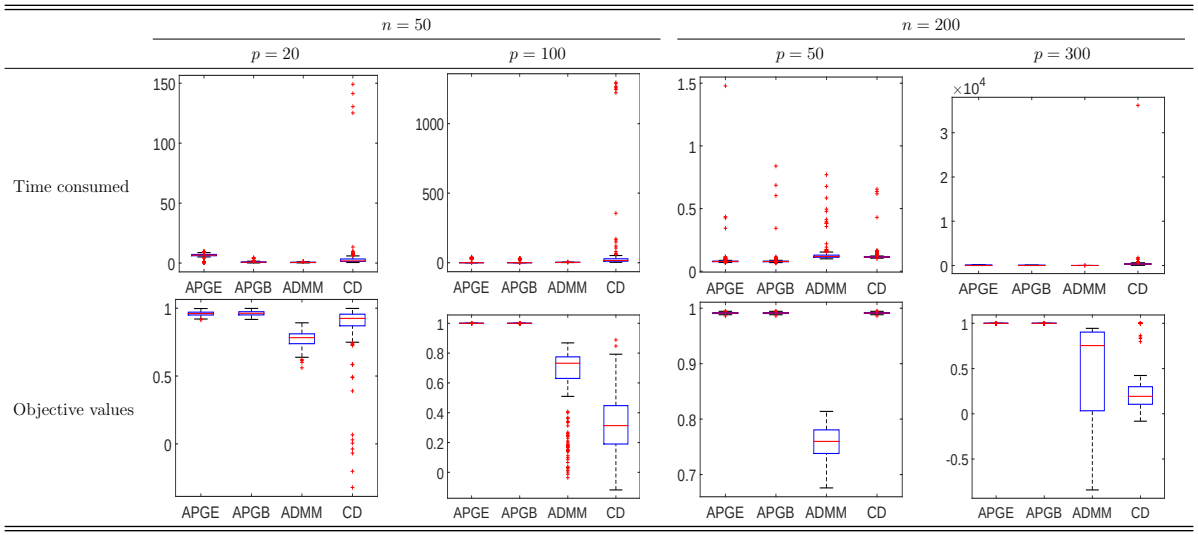


Table 2.3: Boxplots for time consumed and optimized canonical correlation values are displayed for each combination of n and p.

for moderate accuracy. In this context the results of ADMM can be understood since it takes short time while its final objective values are low in general. Thus there is a possibility that we can get a better result using ADMM, since we only considered one value for the parameter ρ.

2.5 Discussion

In this chapter, we proposed a unifying approach to solve constrained multivariate analysis using existing numerical algorithm. This approach can be available due to the relationship that multivariate analysis including PCA, CCA, PLS and MLR can be written as an optimization problem of the generalized Rayleigh quotient. Instead of developing new algorithm for getting constrained solution from each analysis, we can utilize existing numerical algorithms. As we can see in the Section 2.4, this approach works quiet well.

We employ five numerical algorithms some of which are well known.

The great advantages of this approach are three points. First, we can easily add conditions we want, not only the sparsity but any other condition like non-negativity as we saw in Section 2.4. Second, we can employ well established numerical algorithms instead of developing a new algorithm specific to a certain analysis with a certain constraint. Third, we can conduct several analysis stated above by implementing one algorithm.

There are limitations on this approach too. Rayleigh quotient which is the objective function of this approach is nonconvex function This allows guarantee only for the local convergence. Thus, multiple starting points including our suggested one are recommended. Also employed numerical algorithm may require other parameters such as tunning parameters for convergence. For example, ACNPG algorithm in this chapter has two parameters β and λ which is decided based on the unknown value, Lipschitz constant L_R. It would be a good scenario if we know the best value of parameters or suggestions at least. In this chapter, we used a suggested value for those parameter but still there is limitation since we do not know the Lipschitz constant L_R, we conduct our simulations on heuristic toy simulation that gave us a rough guess for L_R. It would be a good approach to choose parameters by conducting cross validations on the parameters, or a intuitive value from the specific knowledge about the dataset.

Chapter 3

Tensor Canonical Correlation Analysis

3.1 Introduction

Canonical Correlation Analysis (CCA), first proposed by Hotelling (1936), is one of the most widely used multivariate statistical analysis techniques to study the relationship between two multivariate data sets. CCA tries to find a set of coefficients such that the correlation between the linear combinations of each data set is maximal. Demands for the analysis of high dimensional data has increased tremendously due to advancement of technology. For example, the relationship between gene networks and DNA-markers is important for understanding complex diseases such as human cancers (Waaijenborg et al., 2008). High dimensionality of gene expression data causes trouble to apply CCA. Traditional CCA does not work any more when the number of variables is far bigger than sample size since sample covariance matrices become singular.

Many sparse estimation methods have been proposed for an intuitive and simple inter-

pretation despite the huge dimension of input data (Waaijenborg et al., 2008; Parkhomenko et al., 2009; Witten et al., 2009; Lykou and Whittaker, 2010). They adopt various regularization methods to shrink the coefficients and conduct variable selection simultaneously. In Chalise and Fridley (2012), several sparse CCA methods are compared to see their difference and performance.

High dimensional data may have huge number of variables but it is still a vector-valued data which can be treated via existing methods. But now we face multi-dimensional array data, also known as a "tensor" data, which is more than just high dimensional. Imaging data is the representative one. Take anatomical magnetic resonance images (MRI, 3D array), and functional magnetic resonance images (fMRI, 4D array) as an example. They possess a multi-dimensional structure. MRI data typically has $2,095,152 = 128 \times 128 \times 128$ observations for one subject. Even more, fMRI typically consists of more than 10^2 MRI images for each subjects.

The first approach that we can try is vectorizing array and applying existing sparse CCA methods. This approach has at least two issues. First is the number of parameters. For example, vectorizing MRI data produces more than two million variables. The second is that it ignores the structure of the data, which may cause loss of spatial information. In this chapter, we propose a new CCA approach that can deal with the tensor data (TCCA). To accomplish this, we utilize the tensor decomposition that significantly downsizes model complexity. Moreover, we extend our tensor CCA to tensor sparse CCA (TSCCA), which gives us a sparse solution. We will use the numerical methods developed in Chapter 2 to solve TSCCA.

The rest of this chapter is organized as follows. In next section, we will review two existing sparse CCA proposed by Witten et al. (2009) and Chi et al. (2013). Section 3.3 will introduce some notations and basic operation of multi-dimensional array. Section 3.4

will develop our proposed models. Efficient algorithms to estimate parameters in those model and discussion about identifiability issue will be followed. In Section 3.5, several numerical simulation results will be described. Section 3.6 concludes this chapter with discussion of some possible future extensions.

3.2 Literature Review

In this section, we summarized two SCCA methods. Sparse CCA proposed by Witten et al. (2009) is the first one that we will review in this section. Based on Witten et al. (2009), adjusted method has been proposed by Chi et al. (2013), which we will review next.

3.2.1 PMD (Witten et al., 2009)

Penalized Matrix Decomposition (PMD) method has been proposed by Witten et al. (2009). PMD approximates a matrix $\boldsymbol{X}$ by $\hat{\boldsymbol{X}} = \sum_{k=1}^{K} d_k \mathbf{u}_k \mathbf{v}_k$ where $d_k, \mathbf{u}_k$ and $\mathbf{v}_k$ minimize the Frobenious norm of $\boldsymbol{X} - \hat{\boldsymbol{X}}$ subject to constraints on $\mathbf{u}_k$ and $\mathbf{v}_k$. As a result, we do a regularized version of singular value decomposition. The penalized matrix decomposition problem for $K = 1$ case can be summarized as follows.

Problem 5. *Let $\boldsymbol{X} \in \mathbb{R}^{n \times p}$ be a real valued matrix with rank $r \leq min\{n, p\}$. Find d and vectors $\boldsymbol{u}, \boldsymbol{v}$ that*

$$\begin{aligned} \text{minimize}_{d,\boldsymbol{u},\boldsymbol{v}} \quad & \frac{1}{2}\|\boldsymbol{X} - d\boldsymbol{u}\boldsymbol{v}^{\mathsf{T}}\|_F^2 \\ \text{subject to} \quad & \|\boldsymbol{u}\|_2^2 = 1,\ \|\boldsymbol{v}\|_2^2 = 1,\ P_1(\boldsymbol{u}) \leq c_1,\ P_2(\boldsymbol{v}) \leq c_2,\ d \geq 0 \end{aligned}$$

where P_1 and P_2 are convex penalty functions, and c_1 and c_2 are constants.

They showed that the optimal $\mathbf{u}$ and $\mathbf{v}$ also solves the following problem.

Problem 6 (rank-1 PMD)**.** *Let* $\boldsymbol{X} \in \mathbb{R}^{n\times p}$ *be a real valued matrix with rank* $r \leq min\{n, p\}$*. Find vectors* $\boldsymbol{u}, \boldsymbol{v}$ *that*

$$\begin{aligned} &\text{maximize}_{\boldsymbol{u},\boldsymbol{v}} \quad \boldsymbol{u}^\intercal \mathbf{X} \boldsymbol{v} \\ &\quad \text{subject to} \quad \|\boldsymbol{u}\|_2^2 = 1,\ \|\boldsymbol{v}\|_2^2 = 1,\ P_1(\boldsymbol{u}) \leq c_1,\ P_2(\boldsymbol{v}) \leq c_2, \end{aligned}$$

where P_1 *and* P_2 *are convex penalty functions, and* c_1 *and* c_2 *are constants. The value of* d *solving the Problem 5 is* $\mathbf{u}^\intercal \mathbf{X} \mathbf{v}$*.*

Witten et al. (2009) also proposed an algorithm to solve above Problem 6.

It is possible to apply this PMD method for estimating sparse canonical coefficients of two datasets on CCA. Let $\boldsymbol{X} \in \mathbb{R}^{n\times p}$ and $\boldsymbol{Y} \in \mathbb{R}^{n\times q}$ be data matrices collected on n subjects. Original CCA proposed by Hotelling (1936) looks for two coefficient vectors $\boldsymbol{u}$ and $\boldsymbol{v}$ that maximizes $Corr(\boldsymbol{X}\boldsymbol{u}, \boldsymbol{Y}\boldsymbol{v})$, the correlation between two linear combination of each datasets. That problem can be written as

$$\text{maximize}_{\boldsymbol{u},\boldsymbol{v}}\ \boldsymbol{u}^\intercal \boldsymbol{X}^\intercal \boldsymbol{Y} \boldsymbol{v} \quad \text{subject to } \boldsymbol{u}^\intercal \boldsymbol{X}^\intercal \boldsymbol{X} \boldsymbol{u} \leq 1,\ , \boldsymbol{v}^\intercal \boldsymbol{Y}^\intercal \boldsymbol{Y} \boldsymbol{v} \leq 1 \tag{3.1}$$

If we assume that covariance matrices of $\boldsymbol{X}$ and $\boldsymbol{Y}$ are identity and add l_1 penalty on $\boldsymbol{u}$ and $\boldsymbol{v}$, then the problem becomes the following one.

Problem 7 (SCCA (Witten et al., 2009))**.**

$$\begin{aligned} &\text{maximize}_{\boldsymbol{u},\boldsymbol{v}} \quad \boldsymbol{u}^\intercal \boldsymbol{X}^\intercal \boldsymbol{Y} \boldsymbol{v} \\ &\quad \text{subject to} \quad \boldsymbol{u}^\intercal \boldsymbol{u} \leq 1,\ \boldsymbol{v}^\intercal \boldsymbol{v} \leq 1,\ \|\boldsymbol{u}\|_1 \leq c_1, \|\boldsymbol{v}\|_1 \leq c_2, \end{aligned}$$

which can be solved using PMD algorithm.

3.2.2 Sparse Canonical Correlation Analysis (Chi et al., 2013)

Above proposed SCCA by Witten et al. (2009) can be improved if we relax the assumption on covariance matrices. Chi et al. (2013) proposed adjusted version of the method of Witten et al. (2009), which relaxes the identity covariance matrices assumption. They started on following basic SCCA model, which is slightly different from the model of Witten et al. (2009).

Problem 8 (SCCA)**.** *Let* $\boldsymbol{X} \in \mathbb{R}^{n\times p}$ *and* $\boldsymbol{Y} \in \mathbb{R}^{n\times q}$ *be data matrices gathered from* n *subjects. Find vectors* $\mathbf{u}$ *and* $\mathbf{v}$ *that*

$$\begin{aligned} \text{maximize} \quad & \mathbf{u}^\intercal \mathbf{X}^\intercal \mathbf{Y}\mathbf{v} - \lambda\|\mathbf{u}\|_1 - \gamma\|\mathbf{v}\|_1 \\ \text{subject to} \quad & \|\mathbf{u}\|_2 \le 1,\ \|\mathbf{v}\|_2 \le 1. \end{aligned}$$

The minor difference with the Problem 7 is that sparsity penalties are added instead of sparsity constraints. If we consider $\mathbf{v}$ fixed, it is a convex problem in terms of $\mathbf{u}$ and vice versa. Thus we can solve Problem 8 by block relaxation method. The update $\mathbf{u}^*$ for $\mathbf{u}$ is given by

$$\begin{aligned} \hat{\mathbf{u}} &= \underset{\mathbf{u}}{\operatorname{argmin}} \frac{1}{2}\|\mathbf{Y}^\intercal\mathbf{X}\mathbf{v} - \mathbf{u}\|_2^2 + \lambda\|\mathbf{u}\|_1 \\ \mathbf{u}^* &= \begin{cases} \hat{\mathbf{u}}/\|\mathbf{u}\|_2 & \text{if } \|\mathbf{u}\|_2 \ge 0 \\ \mathbf{0} & \text{o.w} \end{cases} \end{aligned} \tag{3.2}$$

Same procedure applies to update of $\mathbf{v}$.

While Witten et al. (2009) made an identity covariance matrix assumption, Chi et al.

(2013) suggested to use adjusted covariance estimates, the problem statement is as following.

Problem 9 (SCCA(Chi et al., 2013)). *Let* $\mathbf{X} \in \mathbb{R}^{n\times p}$ *and* $\mathbf{Y} \in \mathbb{R}^{n\times q}$ *be data matrices collected on* n *subjects. Find vectors* $\mathbf{u}$ *and* $\mathbf{v}$ *that*

$$\begin{aligned} \text{maximize} \quad & \mathbf{u}^\intercal \mathbf{X}^\intercal \mathbf{Y} \mathbf{v} - \lambda \|\mathbf{u}\|_1 - \gamma \|\mathbf{v}\|_1 \\ \text{subject to} \quad & \mathbf{u}^\intercal \mathbf{\Sigma}_x \mathbf{u} \leq 1,\, \mathbf{v}^\intercal \mathbf{\Sigma}_y \mathbf{v} \leq 1, \end{aligned}$$

where λ *and* γ *are given constants,* $\mathbf{\Sigma}_x$ *and* $\mathbf{\Sigma}_y$ *are estimated covariance matrices of* $\mathbf{X}$ *and* $\mathbf{Y}$.

Possible issue is that the sample covariance matrices become singular when the number of variables is bigger than sample size. Thus for better performance of the solution of Problem 9, we need good estimators of covariance matrices $\mathbf{\Sigma}_x$ and $\mathbf{\Sigma}_y$. There are many proposed methods to adjust sample covariance matrix (Ledoit and Wolf, 2004; Ledoit et al., 2012; Chi and Lange, 2014). With the adjusted covariance estimates $\tilde{\mathbf{\Sigma}}_x$ and $\tilde{\mathbf{\Sigma}}_y$, we can solve the above problem via iterative algorithm similar to PMD. With fixed $\mathbf{v}$, $\mathbf{u}^*$, the update of $\mathbf{u}$, is given as follows.

$$\begin{aligned} \hat{\mathbf{u}} &= \underset{\mathbf{u}}{\operatorname{argmin}} \ \frac{1}{2} \|\tilde{\mathbf{\Sigma}}_x^{-1/2} \mathbf{X}^\intercal \mathbf{Y} \mathbf{v} - \tilde{\mathbf{\Sigma}}_x^{1/2} \mathbf{u}\|_2^2 + \lambda \|\mathbf{u}\|_1, \\ \mathbf{u}^* &= \begin{cases} \frac{\hat{\mathbf{u}}}{\|\tilde{\mathbf{\Sigma}}_x^{1/2} \hat{\mathbf{u}}\|_2} & \text{if } \|\tilde{\mathbf{\Sigma}}_x \mathbf{u}\|_2 > 0 \\ \mathbf{0} & \text{o.w.} \end{cases} \end{aligned} \tag{3.3}$$

Similarly, we can update $\mathbf{v}$.

3.3 Notation and Basic Operations of Array

Before developing models, we review basic notations and operations of multidimensional arrays in this section. More notations and operations of a tensor including all introduced in this section can be found in Kolda and Bader (2009).

We call the dimension of the tensor the mode, also known as the order. For example, the mode/order of a tensor $\mathcal{X} \in \mathbb{R}^{p_1 \times p_2 \times \cdots p_D}$ is D. For mode-3 tensor $\mathcal{X}$, x_{ijk} indicates the (i, j, k)-th element of $\mathcal{X}$. Fibers of tensor $\mathcal{X}$ denote vectors made by fixing every indices except one. This term can be thought as an extension of matrix rows or columns to the tensor. For example, if we fix the second and third indices of a tensor $\mathcal{X} \in \mathbb{R}^{p_1 \times p_2 \times p_3}$ except the first, we get $p_2 \times p_3$ fibers, which is also called column fibers. In the same manner, we can get $p_1 \times p_3$ number of row fibers and $p_1 \times p_2$ fibers when we fix the third index, which is called tube fibers.

Next, we introduce a operator that has a vector input and a tensor output. We define an outer product $\mathbf{b}_1 \circ \cdots \circ \mathbf{b}_D$ of D vectors $\mathbf{b}_d \in \mathbb{R}^{p_d}$. The result of this outer product is $p_1 \times \cdots \times p_D$ tensor of which entries are calculated as $(\mathbf{a}_1 \circ \cdots \circ \mathbf{a}_D)_{i_1,\ldots,i_D} = \prod_{d=1}^{D} a_{di_d}$.

Next, we introduce three matrix products that are used in this chapter. First, we denote the *Kronecker product* of two matrices $\mathbf{A} \in \mathbb{R}^{m \times n}$ and $\mathbf{B} \in \mathbb{R}^{p \times q}$ as

$$\mathbf{A} \otimes \mathbf{B} = \begin{bmatrix} a_{11}\mathbf{B} & a_{12}\mathbf{B} & \ldots & a_{1n}\mathbf{B} \\ a_{21}\mathbf{B} & a_{22}\mathbf{B} & \ldots & a_{2n}\mathbf{B} \\ \vdots & \vdots & \ddots & \vdots \\ a_{m1}\mathbf{B} & a_{m2}\mathbf{B} & \ldots & a_{mn}\mathbf{B} \end{bmatrix}$$
$$= \begin{bmatrix} \mathbf{a}_1 \otimes \mathbf{b}_1 & \mathbf{a}_1 \otimes \mathbf{b}_2 & \cdots & \mathbf{a}_n \otimes \mathbf{b}_{q-1} & \mathbf{a}_n \otimes \mathbf{b}_q \end{bmatrix} \in \mathbb{R}^{mp \times nq}.$$

When $n = q$, the *Khatri-Rao* product of two matrices $\mathbf{A} \in \mathbb{R}^{m \times n}$ and $\mathbf{B} \in \mathbb{R}^{p \times n}$, denoted as $\mathbf{A} * \mathbf{B}$, is defined as follows.

$$\mathbf{A} * \mathbf{B} = [\ \mathbf{a}_1 \otimes \mathbf{b}_1 \quad \mathbf{a}_2 \otimes \mathbf{b}_2 \quad \cdots \quad \mathbf{a}_n \otimes \mathbf{b}_n\ ].$$

The resulted matrix has dimension $mp \times n$. If $\mathbf{A}$ and $\mathbf{B}$ are vectors, then their *Khatri-Rao product* is identical with *Kronecker product*.

If $\mathbf{A}$ and $\mathbf{B}$ have same size, i.e., if $m = p$ and $n = q$, then the *Hadamard product* is defined by,

$$\mathbf{A} \odot \mathbf{B} = \begin{bmatrix} a_{11}b_{11} & \dots & a_{1n}b_{1n} \\ \vdots & \ddots & \vdots \\ a_{m1}b_{m1} & \dots & a_{mn}b_{mn} \end{bmatrix},$$

which is the elementwise product. Hadamard product commutes, so we will use $\odot_i \mathbf{A}_i$ to denote

$$\mathbf{A}_1 \odot \cdots \odot \mathbf{A}_D = \mathbf{A}_{\pi(1)} \odot \cdots \odot \mathbf{A}_{\pi(D)},$$

where π is any permutation.

We also introduce several basic operators of tensor that transform a tensor into vector or matrix. The *vec* operator stacks column fibers of a tensor. Consider a tensor $\mathcal{X} \in \mathbb{R}^{2 \times 2 \times 2}$ as follows,

$$\mathbf{X}_{..1} = \begin{bmatrix} 1 & 3 \\ 2 & 4 \end{bmatrix}, \qquad \mathbf{X}_{..2} = \begin{bmatrix} 5 & 7 \\ 6 & 8 \end{bmatrix}.$$

Then column fibers of $\mathcal{X}$ are

$$\begin{bmatrix} 1 \\ 2 \end{bmatrix}, \begin{bmatrix} 3 \\ 4 \end{bmatrix}, \begin{bmatrix} 5 \\ 6 \end{bmatrix}, \begin{bmatrix} 7 \\ 8 \end{bmatrix}.$$

Thus by stacking above four vectors we can get $\text{vec}\,\mathcal{X} = [1\ 2\ 3\ 4\ 5\ 6\ 7\ 8]^\intercal$. One more operator that we will use constantly is *matricization*, which is also known as *unfolding* or *flattening*. This operator makes a tensor into a matrix by restructuring. When we reshape a tensor, reshaping order matters since we get different matrices by different orders. We specifically call *mode-n matricization* if we take *mode-n* fibers as columns and we denote $\mathbf{X}_{(n)}$. Consider the toy example above, $2 \times 2 \times 2$ tensor $\mathcal{X}$. *Mode-1* fibers of $\mathcal{X}$ are column fibers listed above, thus *mode-1 matricization* $\mathbf{X}_{(1)}$ is defined as

$$\mathbf{X}_{(1)} = \begin{bmatrix} 1 & 3 & 5 & 7 \\ 2 & 4 & 6 & 8 \end{bmatrix}.$$

Likewise, we can find $\mathbf{X}_{(2)}$ and $\mathbf{X}_{(3)}$

$$\mathbf{X}_{(2)} = \begin{bmatrix} 1 & 2 & 5 & 6 \\ 3 & 4 & 7 & 8 \end{bmatrix}, \qquad \mathbf{X}_{(3)} = \begin{bmatrix} 1 & 2 & 3 & 4 \\ 5 & 6 & 7 & 8 \end{bmatrix}.$$

For two tensors $\mathcal{A}$ and $\mathcal{X}$ with same dimension and mode, we define inner product of them $\langle \mathcal{A}, \mathcal{X} \rangle$ as a sum of elementwise product, $\langle \mathcal{A}, \mathcal{X} \rangle = \sum_{i_1,\ldots,i_D} a_{i_1,\ldots,i_D} x_{i_1,\ldots,i_D}$.

Lastly, we introduce several identities that are useful in the rest of this chapter.

Lemma 1. *For vectors* $\mathbf{u}$ *and* $\mathbf{v}$*, and matrices* $\mathbf{A}, \mathbf{B}, \mathbf{C}$ *and* $\mathbf{D}$ *with compatible dimen-*

sions,

$$\begin{aligned}
\operatorname{vec}(\mathbf{u}\mathbf{v}^{\mathsf{T}}) &= \mathbf{v} \otimes \mathbf{u} \\
\operatorname{vec}(\mathbf{ABC}) &= (\mathbf{C}^{\mathsf{T}} \otimes \mathbf{A})\operatorname{vec}\mathbf{B} \\
(\mathbf{A} \otimes \mathbf{B})^{\mathsf{T}} &= \mathbf{A}^{\mathsf{T}} \otimes \mathbf{B}^{\mathsf{T}} \\
(\mathbf{A} \otimes \mathbf{B})(\mathbf{C} \otimes \mathbf{D}) &= (\mathbf{AC}) \otimes (\mathbf{BD}) \\
(\mathbf{A} \otimes \mathbf{B})(\mathbf{C} * \mathbf{D}) &= (\mathbf{AC}) * (\mathbf{BD}).
\end{aligned}$$

Proof is given in Appendix B.1.

3.4 Model

In this section, we will develop the model for canonical correlation analysis of two tensor data sets that enjoys a significant reduction on the number of parameters while the spatial information is still preserved. To achieve this goal, we adopt a tensor decomposition, which enables us to reconstruct a tensor with fewer parameters than the number of all elements of a tensor. The model that vectorizes data requires huge number of parameters to be estimated which can be a big problem itself. For example if we consider vectorizing the two data which have $100 \times 100 \times 100$ dimension each, we have 2×10^6 parameters to estimates. Instead of the vectorization, we impose a certain structure on two coefficient tensors so that we can deal with a problem with less number of parameters. Furthermore, we impose a certain structure on the covariance matrices of $\mathcal{X}$ and $\mathcal{Y}$ so that we can handle the covariance matrices much easier.

Detailed explanations will be given in following subsections. First, in the Section 3.4.1, We will explain the CANDECOP/PARAFAC (CP) decomposition. Then we will develop the tensor canonical correlation model (TCCA) which imposes a decomposition structure

on two coefficient tensors. Based on this, we will extend the scope so that we can get a sparse solution from the problem, namely tensor sparse CCA (TSCCA). In the Section 3.4.3, we will develop a more parsimonious model based on TCCA by imposing so called separable covariance structure on the covariance matrices of two data sets (TCCA_SEP). TSCCA_SEP, a sparse version of the TCCA_SEP, will follows. Identifiability issue caused by CP decomposition will be treated in Section 3.4.4.

3.4.1 Tensor Decomposition

First, we introduce a key concept of our proposed models, the CANDECOP/PARAFAC (CP) decomposition of a tensor.

Definition 2. *A tensor $\mathcal{X} \in \mathbb{R}^{p_1 \times \cdots \times p_D}$ admits a rank-R CANDECOP/PARAFAC (CP) decomposition if*

$$\mathcal{X} = \sum_{r=1}^{R} \mathbf{x}_1^{(r)} \circ \cdots \circ \mathbf{x}_D^{(r)}, \tag{3.4}$$

where $\mathbf{x}_d^{(r)} \in \mathbb{R}^{p_d}$, $d = 1, \ldots, D$, $r = 1, \ldots, R$ are all column vectors and $\mathcal{X}$ cannot be written as the sum of outer products with the rank lower than R.

This relationship can be also represented as $\mathcal{X} = \left[\!\left[\mathbf{X}^{(1)}, \ldots, \mathbf{X}^{(D)}\right]\!\right]$, where $\mathbf{X}^{(d)} = [\mathbf{x}_d^{(1)}, \ldots, \mathbf{x}_d^{(R)}] \in \mathbb{R}^{p_d \times R}$ following the notation introduced in Kolda (2006); Kolda and Bader (2009). Related to this decomposition, we will utilize following lemma for estimating parameters of our models.

Lemma 2. *If a tensor* $\mathcal{X} \in \mathbb{R}^{p_1 \times \cdots \times p_D}$ *admits a rank-R CP decomposition* (3.4)*, then*

$$\begin{aligned}
\operatorname{vec} \mathcal{X} &= (\mathbf{X}^{(D)} * \cdots * \mathbf{X}^{(1)})\mathbf{1}_R \\
\mathbf{X}_{(d)} &= \mathbf{X}^{(d)}(\mathbf{X}^{(D)} * \cdots * \mathbf{X}^{(d+1)} * \mathbf{X}^{(d-1)} * \cdots * \mathbf{X}^{(1)})^\intercal \\
\operatorname{vec} \mathbf{X}_{(d)} &= [(\mathbf{X}^{(D)} * \cdots * \mathbf{X}^{(d+1)} * \mathbf{X}^{(d-1)} * \cdots * \mathbf{X}^{(1)}) \otimes \mathbf{I}_{p_d})]\operatorname{vec} \mathbf{X}^{(d)}.
\end{aligned}$$

For the proof of first two equalities, see Zhou et al. (2013). The last one comes from the second equality of Lemma 1. This Lemma 2 enables us to reformulate rank-R decomposition of the tensor to the mode-d matricization or vectorization of the tensor (Zhou et al., 2013).

3.4.2 Tensor Canonical Correlation Analysis

Consider a pair of random variables $(\mathcal{X}, \mathcal{Y})$, where $\mathcal{X} \in \mathbb{R}^{I_1 \times \cdots \times I_{d_x}}$ and $\mathcal{Y} \in \mathbb{R}^{J_1 \times \cdots \times J_{d_y}}$. The goal of the CCA is to find the coefficient tensors $\mathcal{A} \in \mathbb{R}^{I_1 \times \cdots \times I_{d_x}}$ and $\mathcal{B} \in \mathbb{R}^{J_1 \times \cdots \times J_{d_y}}$, which maximize the correlation between each linear combination of $\mathcal{X}$ and $\mathcal{Y}$ as follows.

$$\begin{aligned}
& \operatorname{Corr}(\langle \mathcal{X}, \mathcal{A} \rangle, \langle \mathcal{Y}, \mathcal{B} \rangle) \\
= & \frac{\operatorname{Cov}(\langle \mathcal{X}, \mathcal{A} \rangle, \langle \mathcal{Y}, \mathcal{B} \rangle)}{\sqrt{\operatorname{Var}(\langle \mathcal{X}, \mathcal{A} \rangle)\operatorname{Var}(\langle \mathcal{Y}, \mathcal{B} \rangle)}} \\
= & \frac{\operatorname{Cov}(\langle \operatorname{vec} \mathcal{X}, \operatorname{vec} \mathcal{A} \rangle, \langle \operatorname{vec} \mathcal{Y}, \operatorname{vec} \mathcal{B} \rangle)}{\sqrt{\operatorname{Var}(\langle \operatorname{vec} \mathcal{X}, \operatorname{vec} \mathcal{A} \rangle)\operatorname{Var}(\langle \operatorname{vec} \mathcal{Y}, \operatorname{vec} \mathcal{B} \rangle)}} \\
= & \frac{(\operatorname{vec} \mathcal{A})^\intercal \operatorname{Cov}(\operatorname{vec} \mathcal{X}, \operatorname{vec} \mathcal{Y}) \operatorname{vec} \mathcal{B}}{\sqrt{(\operatorname{vec} \mathcal{A})^\intercal \operatorname{Var}(\operatorname{vec} \mathcal{X}) \operatorname{vec} \mathcal{A} (\operatorname{vec} \mathcal{B})^\intercal \operatorname{Var}(\operatorname{vec} \mathcal{Y}) \operatorname{vec} \mathcal{B}}} \\
= & \frac{(\operatorname{vec} \mathcal{A})^\intercal \mathbf{\Sigma}_{\operatorname{vec} \mathcal{X}, \operatorname{vec} \mathcal{B}} (\operatorname{vec} \mathcal{B})}{\sqrt{(\operatorname{vec} \mathcal{A})^\intercal \mathbf{\Sigma}_{\operatorname{vec} \mathcal{X}} (\operatorname{vec} \mathcal{A}) (\operatorname{vec} \mathcal{B})^\intercal \mathbf{\Sigma}_{\operatorname{vec} \mathcal{Y}} (\operatorname{vec} \mathcal{B})}},
\end{aligned} \tag{3.5}$$

where $\mathcal{A} \in \mathbb{R}^{I_1 \times \cdots \times I_{d_x}}$ and $\mathcal{B} \in \mathbb{R}^{J_1 \times \cdots \times J_{d_y}}$, $\mathbf{\Sigma}_{\mathrm{vec}\,\mathcal{X},\mathrm{vec}\,\mathcal{Y}} = \mathrm{Cov}(\mathrm{vec}\,\mathcal{X}, \mathrm{vec}\,\mathcal{Y})$, $\mathbf{\Sigma}_{\mathrm{vec}\,\mathcal{X}} = \mathrm{Var}(\mathrm{vec}\,\mathcal{X})$, and $\mathbf{\Sigma}_{\mathrm{vec}\,\mathcal{Y}} = \mathrm{Var}(\mathrm{vec}\,\mathcal{Y})$. Thus finding $\mathcal{A}$ and $\mathcal{B}$ that maximize the correlation described in above equation (3.5) can be considered as following optimization problem.

Problem 10 (Vectorized TCCA)**.** *For two random tensors $\mathcal{X} \in \mathbb{R}^{I_1 \times \cdots \times I_{d_x}}$ and $\mathcal{Y} \in \mathbb{R}^{J_1 \times \cdots \times J_{d_y}}$, find two coefficient tensors $\mathcal{A} \in \mathbb{R}^{I_1 \times \cdots \times I_{d_x}}$ and $\mathcal{B} \in \mathbb{R}^{J_1 \times \cdots \times J_{d_y}}$ that*

$$\begin{aligned} &\text{maximize} \;\; (\mathrm{vec}\,\mathcal{A})^\intercal \mathbf{\Sigma}_{\mathrm{vec}\,\mathcal{X},\mathrm{vec}\,\mathcal{Y}} (\mathrm{vec}\,\mathcal{B}) \\ &\text{subject to} \;\; (\mathrm{vec}\,\mathcal{A})^\intercal \mathbf{\Sigma}_{\mathrm{vec}\,\mathcal{X}} (\mathrm{vec}\,\mathcal{A}) = 1, (\mathrm{vec}\,\mathcal{B})^\intercal \mathbf{\Sigma}_{\mathrm{vec}\,\mathcal{Y}} (\mathrm{vec}\,\mathcal{B}) = 1 \end{aligned}$$

This can be thought as the ordinary CCA since the model uses vectorized datasets and coefficients. Thus it can be solved as a generalized eigenvalue problem.

And if we want to get a sparse solution, l_1 constraints/penalties can be added on the problem.

Problem 11 (Vectorized TSCCA)**.** *For two random tensors $\mathcal{X} \in \mathbb{R}^{I_1 \times \cdots \times I_{d_x}}$ and $\mathcal{Y} \in \mathbb{R}^{J_1 \times \cdots \times J_{d_y}}$ and constants c_1 and c_2, find two coefficient tensors $\mathcal{A} \in \mathbb{R}^{I_1 \times \cdots \times I_{d_x}}$ and $\mathcal{B} \in \mathbb{R}^{J_1 \times \cdots \times J_{d_y}}$ such that*

$$\begin{aligned} &\text{maximize} \;\; (\mathrm{vec}\,\mathcal{A})^\intercal \mathbf{\Sigma}_{\mathrm{vec}\,\mathcal{X},\mathrm{vec}\,\mathcal{Y}} (\mathrm{vec}\,\mathcal{B}) \\ &\text{subject to} \;\; (\mathrm{vec}\,\mathcal{A})^\intercal \mathbf{\Sigma}_{\mathrm{vec}\,\mathcal{X}} (\mathrm{vec}\,\mathcal{A}) = 1, \; (\mathrm{vec}\,\mathcal{B})^\intercal \mathbf{\Sigma}_{\mathrm{vec}\,\mathcal{Y}} (\mathrm{vec}\,\mathcal{B}) = 1 \\ &\qquad\qquad \|\mathrm{vec}\,\mathcal{A}\|_1 \le c_1, \; \|\mathrm{vec}\,\mathcal{B}\|_1 \le c_2 \end{aligned}$$

or for constants λ and γ,

$$\begin{aligned} &\text{maximize} \;\; (\mathrm{vec}\,\mathcal{A})^\intercal \mathbf{\Sigma}_{\mathrm{vec}\,\mathcal{X},\mathrm{vec}\,\mathcal{Y}} (\mathrm{vec}\,\mathcal{B}) - \lambda \|\mathrm{vec}\,\mathcal{A}\|_1 - \gamma \|\mathrm{vec}\,\mathcal{B}\|_1 \\ &\text{subject to} \;\; (\mathrm{vec}\,\mathcal{A})^\intercal \mathbf{\Sigma}_{\mathrm{vec}\,\mathcal{X}} (\mathrm{vec}\,\mathcal{A}) = 1, \; (\mathrm{vec}\,\mathcal{B})^\intercal \mathbf{\Sigma}_{\mathrm{vec}\,\mathcal{Y}} (\mathrm{vec}\,\mathcal{B}) = 1 \end{aligned}$$

Two SCCA methods introduced in Section 3.2 can be used to solve above Problem 11. Or it is also a good way to utilize the approach that we suggested in Chapter 2, which optimizes the problem using existing well known numerical algorithms.

However, there exist two drawbacks of this model. First, even if we have moderately sized tensor data, the number of parameters can be formidable. For example, consider 128×128 image data. Then the coefficient for this data has $128^2 = 16,384$ parameters. Moreover, vectorization of the data may causes a loss of important spatial information contained in the data structure.

To solve these two issues, we consider a parametrization of coefficient tensors $\mathcal{A}$ and $\mathcal{B}$ in terms of CP decomposition introduced in Definition 2. Let us assume that $\mathcal{A}$ and $\mathcal{B}$ admit CP decomposition as follows.

$$\begin{aligned} \mathcal{A} &= \left[\!\left[\mathbf{A}^{(1)}, \ldots, \mathbf{A}^{(d_x)} \right]\!\right], \qquad \mathbf{A}^{(m)} \in \mathbb{R}^{I_m \times R_x}, \quad , m = 1, \ldots, d_x \\ \mathcal{B} &= \left[\!\left[\mathbf{B}^{(1)}, \ldots, \mathbf{B}^{(d_y)} \right]\!\right], \qquad \mathbf{B}^{(n)} \in \mathbb{R}^{J_n \times R_y}, \quad , n = 1, \ldots, d_y, \end{aligned} \tag{3.6}$$

where R_x and R_y are ranks of each coefficient tensors. Adopting decomposition structures in coefficient tensors causes a significant reduction on the number of parameters in the model, from $\prod_{m=1}^{dx} I_m + \prod_{n=1}^{dy} J_n$ to $R_x \sum_{m=1}^{dx} I_m + R_y \sum_{n=1}^{dy} J_n$. Also we can preserve the spatial information since we can avoid the vectorization. Under this parameterization, our problem can be represented in a much simpler form.

Proposition 1. *Let* $\mathcal{X} \in \mathbb{R}^{I_1 \times \cdots \times I_{d_x}}$ *and* $\mathcal{Y} \in \mathbb{R}^{J_1 \times \cdots \times J_{d_y}}$ *be two random tensors and* $\mathcal{A} = \left[\!\left[\mathbf{A}^{(1)}, \ldots, \mathbf{A}^{(d_x)} \right]\!\right] \in \mathbb{R}^{I_1 \times \cdots \times I_{d_x}}$ *and* $\mathcal{B} = \left[\!\left[\mathbf{B}^{(1)}, \ldots, \mathbf{B}^{(d_y)} \right]\!\right] \in \mathbb{R}^{J_1 \times \cdots \times J_{d_y}}$ *be two*

constant tensors. Define

$$\begin{aligned}\mathbf{A}^{(-m)} &= \mathbf{A}^{(d_x)} * \cdots * \mathbf{A}^{(m+1)} * \mathbf{A}^{(m-1)} * \cdots * \mathbf{A}^{(1)} \\ \mathbf{B}^{(-n)} &= \mathbf{B}^{(d_y)} * \cdots * \mathbf{B}^{(n+1)} * \mathbf{B}^{(n-1)} * \cdots * \mathbf{B}^{(1)}.\end{aligned}$$

Then,

$$\begin{aligned}\mathrm{Var}(\langle \mathcal{X}, \mathcal{A} \rangle) &= (\mathrm{vec}\, \mathbf{A}^{(m)})^\top (\mathbf{A}^{(-m)\top} \otimes \mathbf{I}_{I_m}) \Sigma_{\mathrm{vec}\, \mathbf{X}_{(m)}} (\mathbf{A}^{(-m)} \otimes \mathbf{I}_{I_m}) \mathrm{vec}\, \mathbf{A}^{(m)} \\ \mathrm{Var}(\langle \mathcal{Y}, \mathcal{B} \rangle) &= (\mathrm{vec}\, \mathbf{B}^{(n)})^\top (\mathbf{B}^{(-n)\top} \otimes \mathbf{I}_{J_n}) \Sigma_{\mathrm{vec}\, \mathbf{Y}_{(n)}} (\mathbf{B}^{(-n)} \otimes \mathbf{I}_{J_n}) \mathrm{vec}\, \mathbf{B}^{(n)} \\ \mathrm{Cov}(\langle \mathcal{X}, \mathcal{A} \rangle, \langle \mathcal{Y}, \mathcal{B} \rangle) &= (\mathrm{vec}\, \mathbf{A}^{(m)})^\top (\mathbf{A}^{(-m)\top} \otimes \mathbf{I}_{I_m}) \Sigma_{\mathrm{vec}\, \mathbf{X}_{(m)} \mathrm{vec}\, \mathbf{Y}_{(n)}} (\mathbf{B}^{(-n)} \otimes \mathbf{I}_{J_n}) \mathrm{vec}\, \mathbf{B}^{(n)}\end{aligned}$$

where $\Sigma_{\mathrm{vec}\, \mathbf{X}_{(m)} \mathrm{vec}\, \mathbf{Y}_{(n)}} = \mathrm{Cov}(\mathrm{vec}\, \mathbf{X}_{(m)}, \mathrm{vec}\, \mathbf{Y}_{(n)})$, $\Sigma_{\mathrm{vec}\, \mathbf{X}_{(m)}} = \mathrm{Var}(\mathrm{vec}\, \mathbf{X}_{(m)})$, *and* $\Sigma_{\mathrm{vec}\, \mathbf{Y}_{(n)}} = \mathrm{Var}(\mathrm{vec}\, \mathbf{Y}_{(n)})$ *for* $m \in \{1, \ldots, d_x\}$ *and* $n \in \{1, \ldots, d_y\}$.

Proof is given in Appendix B.2. By Proposition 1 the objective function for tensor CCA (TCCA) can be written as the following optimization problem.

Problem 12 (TCCA)**.** *For two random tensors* $\mathcal{X} \in \mathbb{R}^{I_1 \times \cdots \times I_{d_x}}$ *and* $\mathcal{Y} \in \mathbb{R}^{J_1 \times \cdots \times J_{d_y}}$*, find two coefficient tensors which admit CP decomposition with rank* R_x *and* R_y *respectively as* $\mathcal{A} = \left[\!\left[\mathbf{A}^{(1)}, \ldots, \mathbf{A}^{(d_x)} \right]\!\right] \in \mathbb{R}^{I_1 \times \cdots \times I_{d_x}}$ *and* $\mathcal{B} = \left[\!\left[\mathbf{B}^{(1)}, \ldots, \mathbf{B}^{(d_y)} \right]\!\right] \in \mathbb{R}^{J_1 \times \cdots \times J_{d_y}}$ *such that*

$$\begin{aligned}\text{maximize} \quad & (\mathrm{vec}\, \mathbf{A}^{(m)})^\top (\mathbf{A}^{(-m)\top} \otimes \mathbf{I}_{I_m}) \boldsymbol{\Sigma}_{\mathrm{vec}\, \mathbf{X}_{(m)}, \mathrm{vec}\, \mathbf{Y}_{(n)}} (\mathbf{B}^{(-n)\top} \otimes \mathbf{I}_{J_n}) (\mathrm{vec}\, \mathbf{B}^{(n)}) \\ \text{subject to} \quad & (\mathrm{vec}\, \mathbf{A}^{(m)})^\top (\mathbf{A}^{(-m)\top} \otimes \mathbf{I}_{I_m}) \boldsymbol{\Sigma}_{\mathrm{vec}\, \mathbf{X}_{(m)}} (\mathbf{A}^{(-m)} \otimes \mathbf{I}_{I_m}) \mathrm{vec}\, \mathbf{A}^{(m)} = 1 \\ & (\mathrm{vec}\, \mathbf{B}^{(n)})^\top (\mathbf{B}^{(-n)\top} \otimes \mathbf{I}_{J_n}) \boldsymbol{\Sigma}_{\mathrm{vec}\, \mathbf{Y}_{(n)}} (\mathbf{B}^{(-n)} \otimes \mathbf{I}_{J_n}) \mathrm{vec}\, \mathbf{B}^{(n)} = 1,\end{aligned}$$

for $m \in \{1, \ldots, d_x\}$ *and* $n \in \{1, \ldots, d_y\}$.

We can observe that this optimization problem can be separated according to a pair $(\mathrm{vec}\, \mathbf{A}^{(m)}, \mathrm{vec}\, \mathbf{B}^{(n)})$. Thus we can update a block of two parameter vectors alternately

instead of updating all parameters of tensor coefficient at a time like as the Problem 10. To appreciate the advantage of this estimation procedure, $\mathbf{\Sigma}_{\text{vec}\,\mathbf{X}_{(m)}}$ is a permuted version of $\mathbf{\Sigma}_{\text{vec}\,\mathcal{X}}$, of size $(\prod_{m=1}^{dx} I_m) \times (\prod_{n=1}^{dy} J_n)$. But in practice the sample size N is much less than that $(\prod_{m=1}^{dx} I_m) \times (\prod_{n=1}^{dy} J_n)$. Therefore the sample covariance matrix $\mathbf{\Sigma}_{\text{vec}\,\mathbf{X}_{(m)}}$ is singular. Fortunately in the estimation procedure, we work on the "compressed" covariance matrix

$$(\mathbf{A}^{(-m)\top} \otimes \mathbf{I}_{I_m}) \mathbf{\Sigma}_{\text{vec}\,\mathbf{X}_{(m)}} (\mathbf{A}^{(-m)} \otimes \mathbf{I}_{I_m}) \in \mathbb{R}^{I_m R_x \times I_m R_x},$$

of which is likely to be full rank when $N \geq I_m R_x$.

This problem can be solved via the generalized eigenvalue decomposition. Above is a population model. For considering the sample model, if we have N observed sample data, then we can estimate $\hat{\mathbf{\Sigma}}_{\text{vec}\,\mathbf{X}_{(m)}}, \hat{\mathbf{\Sigma}}_{\text{vec}\,\mathbf{Y}_{(n)}}$ and $\hat{\mathbf{\Sigma}}_{\text{vec}\,\mathbf{X}_{(m)},\text{vec}\,\mathbf{Y}_{(n)}}$. We plugs in those values and iterates algorithm to get a solution. Pseudo code is presented as in the Algorithm 12 in the Appendix B.

We can extend this model to sparsity-constrained problem. If we consider to add l_1 constraint in each parameter vectors, we can get a sparse solution. Since we update a block of parameters, not all parameters, we consider to add sparsity constraints in each updating procedure, constrain the l_1 norm of each mode-m and mode-n parameter vectors of $\mathcal{A}$ and $\mathcal{B}$ as follows.

Problem 13 (TSCCA)**.** *For two random tensors $\mathcal{X} \in \mathbb{R}^{I_1 \times \cdots \times I_{d_x}}$ and $\mathcal{Y} \in \mathbb{R}^{J_1 \times \cdots \times J_{d_y}}$ and two constants λ and β, find two coefficient tensors which admit CP decomposition with rank R_x and R_y respectively as $\mathcal{A} = \left[\!\left[\mathbf{A}^{(1)}, \ldots, \mathbf{A}^{(d_x)} \right]\!\right] \in \mathbb{R}^{I_1 \times \cdots \times I_{d_x}}$ and $\mathcal{B} =$*

$\left[\!\left[\mathbf{B}^{(1)}, \ldots, \mathbf{B}^{(d_y)}\right]\!\right] \in \mathbb{R}^{J_1 \times \cdots \times J_{d_y}}$ *such that*

$$
\begin{aligned}
\text{maximize} \quad & (\operatorname{vec} \mathbf{A}^{(m)})^\top (\mathbf{A}^{(-m)\top} \otimes \mathbf{I}_{I_m}) \boldsymbol{\Sigma}_{\operatorname{vec} \mathbf{X}_{(m)}, \operatorname{vec} \mathbf{Y}_{(n)}} (\mathbf{B}^{(-n)\top} \otimes \mathbf{I}_{J_n}) (\operatorname{vec} \mathbf{B}^{(n)}) \\
\text{subject to} \quad & (\operatorname{vec} \mathbf{A}^{(m)})^\top (\mathbf{A}^{(-m)\top} \otimes \mathbf{I}_{I_m}) \boldsymbol{\Sigma}_{\operatorname{vec} \mathbf{X}_{(m)}} (\mathbf{A}^{(-m)} \otimes \mathbf{I}_{I_m}) \operatorname{vec} \mathbf{A}^{(m)} = 1 \\
& (\operatorname{vec} \mathbf{B}^{(n)})^\top (\mathbf{B}^{(-n)\top} \otimes \mathbf{I}_{J_n}) \boldsymbol{\Sigma}_{\operatorname{vec} \mathbf{Y}_{(n)}} (\mathbf{B}^{(-n)} \otimes \mathbf{I}_{J_n}) \operatorname{vec} \mathbf{B}^{(n)} = 1 \\
& \|\operatorname{vec} \mathbf{A}^{(m)}\|_1 \leq \lambda \\
& \|\operatorname{vec} \mathbf{B}^{(n)}\|_1 \leq \gamma,
\end{aligned}
$$

for $m \in \{1, \ldots, d_x\}$ *and* $n \in \{1, \ldots, d_y\}$.

Since we have more conditions, this problem cannot be solved via the generalized eigenvalue decomposition. This complicates the problem, but we can apply our optimization approach suggested in the Chapter 2, which is the exact method for the Problem 13. Same as TCCA, if we consider sample version, we can use a sample estimator $\hat{\boldsymbol{\Sigma}}_{\operatorname{vec} \mathbf{X}_{(m)}}$, $\hat{\boldsymbol{\Sigma}}_{\operatorname{vec} \mathbf{Y}_{(n)}}$ and $\hat{\boldsymbol{\Sigma}}_{\operatorname{vec} \mathbf{X}_{(m)}, \operatorname{vec} \mathbf{Y}_{(n)}}$ to get an estimated sparse coefficient tensors. Pseudo code is suggested in the Algorithm 13 in the Appendix B.

3.4.3 Tensor Canonical Correlation Analysis with Separable Covariance Structure

Hoff et al. (2011) proposed the array normal distribution with the separable covariance structure. Consider a random tensor $\mathcal{X} \in \mathbb{R}^{p_1 \times p_2}$. Separable covariance model assumes that $\operatorname{Cov}(\operatorname{vec} \mathcal{X}) = \boldsymbol{\Sigma}_2 \otimes \boldsymbol{\Sigma}_1$, where $\boldsymbol{\Sigma}_1$ and $\boldsymbol{\Sigma}_2$ represent covariances among the rows and columns of $\mathcal{X}$ respectively. This model is a good approach if we want a parsimonious structure on $\operatorname{Cov}(\operatorname{vec} \mathcal{X})$ since sample covariance can be unstable or unavailable due to its huge size relative to the sample size. We adopt this structure on our TCCA and TSCCA model to gain stability of the model by reducing the estimation variance. Con-

sider same problem setting as TCCA and TSCCA, a pair of random variables $(\mathcal{X}, \mathcal{Y})$, where $\mathcal{X} \in \mathbb{R}^{I_1 \times \cdots \times I_{d_x}}$ and $\mathcal{Y} \in \mathbb{R}^{J_1 \times \cdots \times J_{d_y}}$. Our goal is to find the coefficient tensors $\mathcal{A} = \left[\!\left[\mathbf{A}^{(1)}, \ldots, \mathbf{A}^{(d_x)} \right]\!\right] \in \mathbb{R}^{I_1 \times \cdots \times I_{d_x}}$ and $\mathcal{B} = \left[\!\left[\mathbf{B}^{(1)}, \ldots, \mathbf{B}^{(d_y)} \right]\!\right] \in \mathbb{R}^{J_1 \times \cdots \times J_{d_y}}$ which admit rank R_x and R_y CP decomposition respectively such that the correlation between each linear combination of $\mathcal{X}$ and $\mathcal{Y}$ is maximal.

Now, let us assume the population covariance model,

$$\begin{aligned} \operatorname{Var}(\operatorname{vec} \mathcal{X}) &= \mathbf{\Sigma}_{\mathbf{X},d_x} \otimes \cdots \otimes \mathbf{\Sigma}_{\mathbf{X},1}, \quad \mathbf{\Sigma}_{\mathbf{X},m} \in \mathbb{R}^{I_m \times I_m}, \quad m = 1, \ldots, d_x \\ \operatorname{Var}(\operatorname{vec} \mathcal{Y}) &= \mathbf{\Sigma}_{\mathbf{Y},d_y} \otimes \cdots \otimes \mathbf{\Sigma}_{\mathbf{Y},1}, \quad \mathbf{\Sigma}_{\mathbf{Y},n} \in \mathbb{R}^{J_n \times J_n}, \quad n = 1, \ldots, d_y. \end{aligned} \tag{3.7}$$

As a result, the covariance matrix of $(\operatorname{vec} \mathcal{X}^\intercal,\ \operatorname{vec} \mathcal{Y}^\intercal)^\intercal$ can be written as follows

$$\operatorname{Var} \begin{bmatrix} \operatorname{vec} \mathcal{X} \\ \operatorname{vec} \mathcal{Y} \end{bmatrix} = \begin{bmatrix} \mathbf{\Sigma}_{\mathbf{X},d_x} \otimes \cdots \otimes \mathbf{\Sigma}_{\mathbf{X},1} & \mathbf{\Sigma}_{\mathbf{X},\mathbf{Y}} \\ \mathbf{\Sigma}_{\mathbf{Y},\mathbf{X}} & \mathbf{\Sigma}_{\mathbf{Y},d_y} \otimes \cdots \otimes \mathbf{\Sigma}_{\mathbf{Y},1} \end{bmatrix}. \tag{3.8}$$

Intuitively $\mathbf{\Sigma}_{\mathbf{X},m}$ summarizes the covariance along mode-m fibers of tensor $\mathcal{X}$, and $\mathbf{\Sigma}_{\mathbf{Y},n}$ summarizes the covariance along mode-n fibers of tensor $\mathcal{Y}$. We inspect how the separable covariance structure, together with the CP structure on $\mathcal{A}$ and $\mathcal{B}$, simplifies the variance terms in the TCCA model, Problem 12. Following result represents the variances in presence of the separable covariance structure (3.7) and plays a key role in our estimation algorithm.

Proposition 2. *Let $\mathcal{X} \in \mathbb{R}^{I_1 \times \cdots \times I_{d_x}}$ and $\mathcal{Y} \in \mathbb{R}^{J_1 \times \cdots \times J_{d_y}}$ be two random tensors with the separable covariance structure* (3.7) *and $\mathcal{A} = \left[\!\left[\mathbf{A}^{(1)}, \ldots, \mathbf{A}^{(d_x)} \right]\!\right] \in \mathbb{R}^{I_1 \times \cdots \times I_{d_x}}$ and*

$\mathcal{B} = \left[\!\left[\mathbf{B}^{(1)}, \ldots, \mathbf{B}^{(d_y)}\right]\!\right] \in \mathbb{R}^{J_1 \times \cdots \times J_{d_y}}$ *be two constant tensors. Define*

$$\begin{aligned}\mathbf{R}_{\mathbf{X},-m} &= \odot_{m' \neq m}(\mathbf{A}^{(m')\intercal}\boldsymbol{\Sigma}_{\mathbf{X},m'}\mathbf{A}^{(m')}) \\ \mathbf{R}_{\mathbf{Y},-n} &= \odot_{n' \neq n}(\mathbf{B}^{(n')\intercal}\boldsymbol{\Sigma}_{\mathbf{Y},n'}\mathbf{B}^{(n')}).\end{aligned}$$

Then

$$\begin{aligned}\mathrm{Var}(\langle \mathcal{X}, \mathcal{A} \rangle) &= (\mathrm{vec}\,\mathbf{A}^{(m)})^\intercal(\mathbf{R}_{\mathbf{X},-m} \otimes \boldsymbol{\Sigma}_{\mathbf{X},m})(\mathrm{vec}\,\mathbf{A}^{(m)}) \\ \mathrm{Var}(\langle \mathcal{Y}, \mathcal{B} \rangle) &= (\mathrm{vec}\,\mathbf{B}^{(n)})^\intercal(\mathbf{R}_{\mathbf{Y},-n} \otimes \boldsymbol{\Sigma}_{\mathbf{Y},n})(\mathrm{vec}\,\mathbf{B}^{(n)}),\end{aligned}$$

for $m \in \{1, \ldots, d_x\}$ *and* $n \in \{1, \ldots, d_y\}$.

Proof is given in Appendix B.3. By Proposition 2, the objective function of tensor CCA with the separable covariance structure (TCCA_SEP) can be written as a following optimization problem.

Problem 14 (TCCA_SEP)**.** *For two random tensors* $\mathcal{X} \in \mathbb{R}^{I_1 \times \cdots \times I_{d_x}}$ *and* $\mathcal{Y} \in \mathbb{R}^{J_1 \times \cdots \times J_{d_y}}$ *which admit the separable covariance structure* (3.7)*, find two coefficient tensors which admit CP decomposition with rank* R_x *and* R_y *respectively as* $\mathcal{A} = \left[\!\left[\mathbf{A}^{(1)}, \ldots, \mathbf{A}^{(d_x)}\right]\!\right] \in \mathbb{R}^{I_1 \times \cdots \times I_{d_x}}$ *and* $\mathcal{B} = \left[\!\left[\mathbf{B}^{(1)}, \ldots, \mathbf{B}^{(d_y)}\right]\!\right] \in \mathbb{R}^{J_1 \times \cdots \times J_{d_y}}$ *such that*

$$\begin{aligned}\text{maximize} \quad & (\mathrm{vec}\,\mathbf{A}^{(m)})^\intercal(\mathbf{A}^{(-m)\intercal} \otimes \mathbf{I}_{I_m})\boldsymbol{\Sigma}_{\mathrm{vec}\,\mathbf{X}_{(m)},\mathrm{vec}\,\mathbf{Y}_{(n)}}(\mathbf{B}^{(-n)\intercal} \otimes \mathbf{I}_{J_n})(\mathrm{vec}\,\mathbf{B}^{(n)}) \\ \text{subject to} \quad & (\mathrm{vec}\,\mathbf{A}^{(m)})^\intercal(\mathbf{R}_{\mathbf{X},-m} \otimes \boldsymbol{\Sigma}_{\mathbf{X},m})(\mathrm{vec}\,\mathbf{A}^{(m)}) = 1 \\ & (\mathrm{vec}\,\mathbf{B}^{(n)})^\intercal(\mathbf{R}_{\mathbf{Y},-n} \otimes \boldsymbol{\Sigma}_{\mathbf{Y},n})(\mathrm{vec}\,\mathbf{B}^{(n)}) = 1,\end{aligned}$$

for $m \in \{1, \ldots, d_x\}$ *and* $n \in \{1, \ldots, d_y\}$.

We can observe that this optimization problem can be also separated according to a pair $(\mathrm{vec}\,\mathbf{A}^{(m)}, \mathrm{vec}\,\mathbf{B}^{(n)})$. Thus we can use same scheme, block relaxation method, to update parameter values. The difference with TCCA in Problem 12 is that Proposition 2

simplifies the calculation of variances $\mathrm{Var}(\langle \mathcal{X}, \mathcal{A} \rangle)$ and $\mathrm{Var}(\langle \mathcal{Y}, \mathcal{B} \rangle)$. This difference causes less computation on calculating covariance matrices compared to TCCA model. When we calculate "compressed" covariance matrix of $\mathcal{X}$ for updating $\mathbf{A}^{(m)}$ in TCCA, it takes $(R_x \prod_{m=1}^{d_x} I_m)^2$ flops. In the meantime, it takes $(R_x I_m)^2$ flops for calculating $\mathbf{R}_{\mathbf{X},-m} \otimes \mathbf{\Sigma}_{\mathbf{X},m}$ of TCCA_SEP model for updating $\mathbf{A}^{(m)}$.

This problem also can be solved via the generalized eigenvalue decomposition. Since above model is a population model, we need to estimate $\hat{\mathbf{\Sigma}}_{\mathbf{X},m}$ and $\hat{\mathbf{\Sigma}}_{\mathbf{Y},n}$ for $m = 1, \ldots, d_x$, $n = 1, \ldots, d_y$ and $\hat{\mathbf{\Sigma}}_{\mathbf{XY}}$ using observed data. $\hat{\mathbf{\Sigma}}_{\mathbf{XY}}$ is an unstructured one, so we will use sample estimates. For $\hat{\mathbf{\Sigma}}_{\mathbf{X},m}$ and $\hat{\mathbf{\Sigma}}_{\mathbf{Y},n}$, there are few proposed estimating methods. Hoff et al. (2011) suggested two estimators, one is the ML estimator and the other is the Bayes estimator. Also Gerard and Hoff (2015) proposed uniformly minimum risk equivariant estimator (UMREE) via a generalized Bayes procedure. However, those estimators performs well under the assumption that data come from the array normal distribution with separable covariance structure. But in our problem, we do not assume any condition about the population. Thus we suggest to use other estimators. For developing new estimators, first we need following lemma.

Lemma 3. *If a random tensor* $\mathcal{X} \in \mathbb{R}^{p_1 \times \cdots \times p_D}$ *has mean zero and separable covariance* $\mathrm{Var}(\mathrm{vec}\, \mathcal{X}) = \mathbf{\Sigma}_D \otimes \cdots \otimes \mathbf{\Sigma}_1$, *then*

$$\begin{aligned} E(\mathbf{X}_{(d)} \mathbf{X}_{(d)}^{\mathsf{T}}) &= (\prod_{d' \neq d} \mathrm{tr} \mathbf{\Sigma}_{d'}) \mathbf{\Sigma}_d \\ E(\|\mathrm{vec}\, \mathcal{X}\|_2^2) &= \prod_{d=1}^{D} \mathrm{tr}(\mathbf{\Sigma}_d). \end{aligned}$$

Proof is given in the Appendix B.4. Based on N iid observations of $\mathcal{X}_i$ and $\mathcal{Y}_i$, con-

sistent estimators of $\prod_{m=1}^{d_x} \mathrm{tr}(\mathbf{\Sigma}_{\mathbf{X},m})$ and $\prod_{n=1}^{d_y} \mathrm{tr}(\mathbf{\Sigma}_{\mathbf{Y},n})$ are following respectively.

$$\begin{aligned} \hat{r}_x &= \frac{1}{N}\sum_{i=1}^{N} \|\mathrm{vec}\,(\mathcal{X}_i - \bar{\mathcal{X}})\|_2^2 \\ \hat{r}_y &= \frac{1}{N}\sum_{i=1}^{N} \|\mathrm{vec}\,(\mathcal{Y}_i - \bar{\mathcal{Y}})\|_2^2, \end{aligned} \tag{3.9}$$

where $\bar{\mathcal{X}}$ and $\bar{\mathcal{Y}}$ are mean of $\mathcal{X}$ and $\mathcal{Y}$ respectively. By Lemma 3 and using above estimators, we propose estimators for covariance matrices as follows

$$\begin{aligned} \hat{\mathbf{\Sigma}}_{\mathbf{X},m} &= \frac{1}{N\hat{r}_x^{(d_x-1)/d_x}}\sum_{i=1}^{N}(\mathbf{X}_{(m)i} - \bar{\mathbf{X}}_{(m)})(\mathbf{X}_{(m)i} - \bar{\mathbf{X}}_{(m)})^\intercal, \qquad m = 1,\ldots,d_x \\ \hat{\mathbf{\Sigma}}_{\mathbf{Y},n} &= \frac{1}{N\hat{r}_y^{(d_y-1)/d_y}}\sum_{i=1}^{N}(\mathbf{Y}_{(n)i} - \bar{\mathbf{Y}}_{(n)})(\mathbf{Y}_{(n)i} - \bar{\mathbf{Y}}_{(n)})^\intercal, \qquad n = 1,\ldots,d_y \\ \hat{\mathbf{\Sigma}}_{\mathbf{X},\mathbf{Y}} &= \frac{1}{N}\sum_{i=1}^{N}(\mathrm{vec}\,\mathcal{X}_i - \mathrm{vec}\,\bar{\mathcal{X}})(\mathrm{vec}\,\mathcal{Y}_i - \mathrm{vec}\,\bar{\mathcal{Y}})^\intercal \end{aligned} \tag{3.10}$$

where $\mathbf{X}_{(m)i}$ is the mode-m matricization of $\mathcal{X}_i$ and $\mathbf{Y}_{(n)i}$ is the mode-n matricization of $\mathcal{Y}_i$, $i = 1,\ldots,N$.

The separable covariance structure is not identifiable in individual $\mathbf{\Sigma}_{\mathbf{X},m}$ due to scaling indeterminacy. Therefore $\mathbf{\Sigma}_{\mathbf{X},m}$ cannot be consistently estimated. However, via Slutsky theorem,

$$\begin{aligned} & \hat{\mathbf{\Sigma}}_{\mathbf{X},d_x} \otimes \cdots \otimes \hat{\mathbf{\Sigma}}_{\mathbf{X},1} \\ \to\ & \frac{(\prod_m \mathrm{tr}(\mathbf{\Sigma}_{\mathbf{X},m}))^{d_x-1}}{(\prod_m \mathrm{tr}(\mathbf{\Sigma}_{\mathbf{X},m}))^{d_x-1}} \mathbf{\Sigma}_{\mathbf{X},d_x} \otimes \cdots \otimes \mathbf{\Sigma}_{\mathbf{X},1} \\ =\ & \mathbf{\Sigma}_{\mathbf{X},d_x} \otimes \cdots \otimes \mathbf{\Sigma}_{\mathbf{X},1} \end{aligned}$$

consistently estimates $\mathrm{Var}(\mathrm{vec}\ \boldsymbol{X})$, which is the term that matters in CCA.

By plugging in those values 3.10 and iterating the algorithm, we can get a solution. Pseudo code is presented as the Algorithm 14 in the Appendix B.

We can extend this model to sparsity-constrained problem version same as we extend TCCA to TSCCA. If we consider to add l_1 constraint in each parameter vectors, we can get a sparse solution. The problem set up for TSCCA_SEP is as follows.

Problem 15 (TSCCA_SEP)**.** *For two random tensors* $\mathcal{X} \in \mathbb{R}^{I_1 \times \cdots \times I_{d_x}}$ *and* $\mathcal{Y} \in \mathbb{R}^{J_1 \times \cdots \times J_{d_y}}$ *and two constants* λ *and* β*, find two coefficient tensors which admit CP decomposition with rank* R_x *and* R_y *respectively as* $\mathcal{A} = \left[\!\left[\mathbf{A}^{(1)}, \ldots, \mathbf{A}^{(d_x)}\right]\!\right] \in \mathbb{R}^{I_1 \times \cdots \times I_{d_x}}$ *and* $\mathcal{B} = \left[\!\left[\mathbf{B}^{(1)}, \ldots, \mathbf{B}^{(d_y)}\right]\!\right] \in \mathbb{R}^{J_1 \times \cdots \times J_{d_y}}$ *such that*

$$
\begin{aligned}
\text{maximize} \quad & (\operatorname{vec} \mathbf{A}^{(m)})^\intercal (\mathbf{A}^{(-m)\intercal} \otimes \mathbf{I}_{I_m}) \Sigma_{\operatorname{vec} \mathbf{X}_{(m)}, \operatorname{vec} \mathbf{Y}_{(n)}} (\mathbf{B}^{(-n)\intercal} \otimes \mathbf{I}_{J_n}) (\operatorname{vec} \mathbf{B}^{(n)}) \\
\text{subject to} \quad & (\operatorname{vec} \mathbf{A}^{(m)})^\intercal (\mathbf{R}_{\mathbf{X},-m} \otimes \mathbf{\Sigma}_{\mathbf{X},m}) (\operatorname{vec} \mathbf{A}^{(m)}) = 1 \\
& (\operatorname{vec} \mathbf{B}^{(n)})^\intercal (\mathbf{R}_{\mathbf{Y},-n} \otimes \mathbf{\Sigma}_{\mathbf{Y},n}) (\operatorname{vec} \mathbf{B}^{(n)}) = 1 \\
& \|\operatorname{vec} \mathbf{A}^{(m)}\|_1 \leq \lambda \\
& \|\operatorname{vec} \mathbf{B}^{(n)}\|_1 \leq \gamma,
\end{aligned}
$$

for $m \in \{1, \ldots, d_x\}$ *and* $n \in \{1, \ldots, d_y\}$.

Though l_1 constraints complicate problem, we can apply our optimization approach suggested in Chapter 2, which is exactly Problem 15. Pseudo code is suggested in the Algorithm 15 in the Appendix B.

3.4.4 Identifiability

In this section, we will tackle the identifiability issue in the parameter estimation. This problem has been dealt in Zhou et al. (2013, Section 4.2), which is identical with the problem in our model. We will explain those results in this section for the convenience.

Identifiability issue arises when we use tensor decomposition. Consider a rank-R decomposition of a tensor $\mathcal{X} = \left[\!\left[\mathbf{X}^{(1)}, \ldots, \mathbf{X}^{(D)}\right]\!\right] \in \mathbb{R}^{p_1 \times \cdots p_D}$. There are two reasons to cause this problem: indeterminacy of $\mathcal{X}$ and the non-uniqueness of the decomposition.

Consider n diagonal matrices $\Lambda_1, \ldots, \Lambda_D$, where $\Lambda_i \in \mathbb{R}^{R \times R}$, and let denote λ_{ir} means rth diagonal element of Λ_i. Then it is easily shown that $\mathcal{X}$ can be written as $\left[\!\left[\mathbf{X}^{(1)}\Lambda_1, \ldots, \mathbf{X}^{(D)}\Lambda_D\right]\!\right]$ for any matrices Λ_i, $i = 1, \ldots, D$, such that $\prod_i \lambda_{ir} = 1$ for $r = 1, \ldots, R$. Similarly, $\mathcal{X}$ is not identifiable against to permutation. $\mathcal{X}$ can be written as $\left[\!\left[\mathbf{X}^{(1)}\Pi, \ldots, \mathbf{X}^{(D)}\Pi\right]\!\right]$ for any permutation matrix $\Pi \in \mathbb{R}^{R \times R}$.

To resolve this indeterminacy, constrained parameterization is required. For scaling issue, we fix values in the first row of $\mathbf{X}^{(i)}$, $i = 1, \ldots, D-1$ as one. This will resolve the scaling indeterminacy issue. Also, to dealing with the permutation indeterminacy, we assume that the first row of each $\mathbf{X}^{(i)}$, $i = 1, \ldots, D-1$ are distinct and arranged in descending order, such as $x_{i1}^{(1)} > \ldots > x_{i1}^{(R)}$. The resulting parameter space is

$$\{(\mathbf{X}^{(1)}, \ldots, \mathbf{X}^{(D)}) : x_{i1}^{(r)} = 1, \text{for}\, i = 1, \ldots, D,\ r = 1, \ldots, R, \text{and}\, x_{i1}^{(1)} > \ldots > x_{i1}^{(R)}\},$$

which is open and convex. Our suggestion in this chapter is arbitrary, and there may be many other choice of arrays that is excluded in our suggested parameter space. In some applications, more sensible choice may be available.

The second reason that cause the identifiability issue is the possible non-uniqueness of the decomposition. There are several results that we can check the uniqueness of the decomposition results, which are well introduced in Kolda and Bader (2009, Section 3.2).

Proposition 3. *Consider a rank-R decomposition of D-dimensional array $\mathcal{X}$.*

1. *(De Lathauwer, 2006, $D = 3$ case) If $R(R-1) \leq p_1(p_1-1)p_2(p_2-1)/2$, this decomposition is unique for almost all such arrays except on a set of Lebesgue*

measure zero.

2. *(De Lathauwer, 2006, $D = 4$ case) For four dimensional case, if $R \leq p_4$ and $R(R-1) \leq p_1p_2p_3(3p_1p_2p_3-p_1p_2-p_1p_3-p_2p_3-p_1-p_2-p_3+3)/4$, this decomposition is unique for almost all such arrays except on a set of Lebesgue measure zero.*

3. *(Liu and Sidiropoulos, 2001, Necessary Condition for D-way tensor) If this decomposition is unique up to scaling and permutation,* $\min_{d=1,\ldots,D}(\prod_{d' \neq d} rank(\mathbf{X}_{(d')})) \geq R$.

4. *(Sidiropoulos and Bro, 2000, Sufficient Condition for D-way tensor) If* $\sum_{d=1}^{D} k_{\mathbf{X}_{(d)}} \geq 2R + (D-1)$, *where* $k_{\mathbf{A}}$ *is the k-rank of matrix* $\mathbf{A}$, *this decomposition is unique up tp scaling and permutation.*

3.5 Numerical Results

In this section, we describe simulation results conducted under various conditions to see the performance of our TCCA and TCCA_SEP models and their sparsity-constrained models. Our gaol is to show that our proposed models can recover various signals even if we adopt a row rank decomposition structure. Also we show that sparsity-constrained models perform well, and get more clearly recovered signals.

3.5.1 Probabilistic Model for CCA

For the assessment of performance of our models, we need to generate data with prespecified canonical correlation coefficients and canonical vectors. We can generate such data from the probabilistic CCA model. Bach and Jordan (2005) first introduced a probabilistic CCA model, which interpret the relationship between two data sets as

originating the hidden common variable. Let desired D canonical vectors $(\mathbf{A}_D, \mathbf{B}_D)$ and canonical correlations $\boldsymbol{\rho}_d = (\rho_1, \ldots, \rho_D)$. Our goal is to generate two datasets $\mathbf{X}$ and $\mathbf{Y}$ such that correlation between their linear combinations $\mathbf{a}_d^\top \mathbf{X}$ and $\mathbf{b}_d^\top \mathbf{Y}$ is ρ_d, $d = 1, \ldots, D$. Consider a random variable $\mathbf{z}$ of which each element follows standard normal and assume that two random vectors $\mathbf{x}$ and $\mathbf{y}$ have conditional distribution given $\mathbf{z}$. We will find the joint distribution of $\mathbf{x}$ and $\mathbf{y}$ which has our desired properties as mentioned above. Let $\boldsymbol{\Sigma}_\mathbf{x}$ and $\boldsymbol{\Sigma}_\mathbf{y}$ be two covariance matrices satisfying $\mathbf{A}_D^\top \boldsymbol{\Sigma}_\mathbf{x} \mathbf{A}_D = \mathbf{B}_D^\top \boldsymbol{\Sigma}_\mathbf{y} \mathbf{B}_D = \mathbf{I}_D$. And define two linear transformations

$$\begin{aligned} \mathbf{P}_\mathbf{X} &= \boldsymbol{\Sigma}_\mathbf{X} \mathbf{A}_D \mathbf{M}_\mathbf{x} \\ \mathbf{P}_\mathbf{Y} &= \boldsymbol{\Sigma}_\mathbf{Y} \mathbf{B}_D \mathbf{M}_\mathbf{y}, \end{aligned} \tag{3.11}$$

where $\mathbf{M}_\mathbf{x}, \mathbf{M}_\mathbf{y} \in \mathbb{R}^{D \times D}$ are arbitrary matrices such that $\mathbf{M}_\mathbf{x} \mathbf{M}_\mathbf{y}^\top = \mathrm{diag}(\boldsymbol{\rho}_D)$ and their spectral norms are less than 1. We consider the latent factor model

$$\begin{aligned} \mathbf{z} &\sim N(\mathbf{0}, \mathbf{I}_D) \\ \mathbf{x}|\mathbf{z} = \boldsymbol{z} &\sim N(\mathbf{P}_\mathbf{x} \boldsymbol{z} + \boldsymbol{\mu}_\mathbf{x}, \boldsymbol{\Sigma}_\mathbf{x} - \mathbf{P}_\mathbf{x} \mathbf{P}_\mathbf{x}^\top) \\ \mathbf{y}|\mathbf{z} = \boldsymbol{z} &\sim N(\mathbf{P}_\mathbf{y} \boldsymbol{z} + \boldsymbol{\mu}_\mathbf{y}, \boldsymbol{\Sigma}_\mathbf{y} - \mathbf{P}_\mathbf{y} \mathbf{P}_\mathbf{y}^\top). \end{aligned} \tag{3.12}$$

Then we can derive the joint distribution of $\mathbf{x}$ and $\mathbf{y}$,

$$\begin{bmatrix} \mathbf{x} \\ \mathbf{y} \end{bmatrix} \sim N\left(\begin{bmatrix} \boldsymbol{\mu}_\mathbf{x} \\ \boldsymbol{\mu}_\mathbf{y} \end{bmatrix}, \begin{bmatrix} \boldsymbol{\Sigma}_\mathbf{x} & \boldsymbol{\Sigma}_\mathbf{x} \mathbf{A}_D \mathrm{diag}(\boldsymbol{\rho}_D) \mathbf{B}_D^\top \boldsymbol{\Sigma}_\mathbf{y} \\ \boldsymbol{\Sigma}_\mathbf{y} \mathbf{B}_D \mathrm{diag}(\boldsymbol{\rho}_D) \mathbf{A}_D^\top \boldsymbol{\Sigma}_\mathbf{x} & \boldsymbol{\Sigma}_\mathbf{y} \end{bmatrix} \right). \tag{3.13}$$

We can verify that $\mathbf{x}$ and $\mathbf{y}$ have our desired covariance matrices and correlations. As we can see in the remark 2.1, CCA eventually finds a canonical correlation and canonical

coefficients which satisfy

$$\begin{bmatrix} \mathbf{0} & \mathbf{\Sigma_{xy}} \\ \mathbf{\Sigma_{yx}} & \mathbf{0} \end{bmatrix} \begin{bmatrix} \mathbf{A}_D \\ \mathbf{B}_D \end{bmatrix} = \begin{bmatrix} \mathbf{\Sigma_x} & \mathbf{0} \\ \mathbf{0} & \mathbf{\Sigma_y} \end{bmatrix} \begin{bmatrix} \mathbf{A}_D \\ \mathbf{B}_D \end{bmatrix} \mathrm{diag}(\boldsymbol{\rho}_D). \tag{3.14}$$

Our latent factor model satisfies above equation (3.14),

$$\begin{aligned} & \begin{bmatrix} \mathbf{0} & \mathbf{\Sigma_x} \mathbf{A}_D \mathrm{diag}(\boldsymbol{\rho}_D) \mathbf{B}_D^\top \mathbf{\Sigma_y} \\ \mathbf{\Sigma_y} \mathbf{B}_D \mathrm{diag}(\boldsymbol{\rho}_D) \mathbf{A}_D^\top \mathbf{\Sigma_x} & \mathbf{0} \end{bmatrix} \begin{bmatrix} \mathbf{A}_D \\ \mathbf{B}_D \end{bmatrix} \\ = & \begin{bmatrix} \mathbf{\Sigma_x} \mathbf{A}_D \mathrm{diag}(\boldsymbol{\rho}_D) \mathbf{B}_D^\top \mathbf{\Sigma_y} \mathbf{B}_D \\ \mathbf{\Sigma_y} \mathbf{B}_D \mathrm{diag}(\boldsymbol{\rho}_D) \mathbf{A}_D^\top \mathbf{\Sigma_x} \mathbf{A}_D \end{bmatrix} \\ = & \begin{bmatrix} \mathbf{\Sigma_x} \mathbf{A}_D \mathrm{diag}(\boldsymbol{\rho}_D) \\ \mathbf{\Sigma_y} \mathbf{B}_D \mathrm{diag}(\boldsymbol{\rho}_D) \end{bmatrix} \\ = & \begin{bmatrix} \mathbf{\Sigma_x} & \mathbf{0} \\ \mathbf{0} & \mathbf{\Sigma_y} \end{bmatrix} \begin{bmatrix} \mathbf{A}_D \\ \mathbf{B}_D \end{bmatrix} \mathrm{diag}(\boldsymbol{\rho}_D) \end{aligned} \tag{3.15}$$

where the second equality is from the assumption that $\mathbf{A}_D^\top \mathbf{\Sigma_x} \mathbf{A}_D = \mathbf{B}_D^\top \mathbf{\Sigma_y} \mathbf{B}_D = \mathbf{I}_D$.

Now, we discuss how to construct $\mathbf{\Sigma_x}$ and $\mathbf{\Sigma_y}$ so that we can get all parameters in the joint distribution of $\mathbf{x}$ and $\mathbf{y}$. Let

$$\mathbf{A}_D \;=\; \begin{bmatrix} \mathbf{Q_x} \; \widetilde{\mathbf{Q}}_\mathbf{x} \end{bmatrix} \begin{bmatrix} \mathbf{R_x} \\ \mathbf{0} \end{bmatrix}$$

be the full QR decomposition of $\mathbf{A}_d$. Then

$$\mathbf{\Sigma_x} \;=\; \mathbf{Q_x} \mathbf{R_x}^{-\top} \mathbf{R_x}^{-1} \mathbf{Q_x}^\top + \widetilde{\mathbf{Q}}_\mathbf{x} \widetilde{\mathbf{\Sigma}}_\mathbf{x} \widetilde{\mathbf{Q}}_\mathbf{x}^\top$$

satisfies $\mathbf{A}_D^\top \boldsymbol{\Sigma}_\mathbf{x} \mathbf{A}_D = \mathbf{I}_d$ for arbitrary positive semidefinite $\widetilde{\boldsymbol{\Sigma}}_\mathbf{x}$. Similarly let Let

$$\mathbf{B}_D \;=\; \left[\mathbf{Q}_\mathbf{y}, \widetilde{\mathbf{Q}}_\mathbf{y}\right] \begin{bmatrix} \mathbf{R}_\mathbf{y} \\ \mathbf{0} \end{bmatrix}$$

be the full QR decomposition of $\mathbf{B}_D$. Then

$$\boldsymbol{\Sigma}_\mathbf{y} \;=\; \mathbf{Q}_\mathbf{y} \mathbf{R}_\mathbf{y}^{-\top} \mathbf{R}_\mathbf{y}^{-1} \mathbf{Q}_\mathbf{y}^\top + \widetilde{\mathbf{Q}}_\mathbf{y} \widetilde{\boldsymbol{\Sigma}}_\mathbf{y} \widetilde{\mathbf{Q}}_\mathbf{y}^\top$$

satisfies $\mathbf{B}_D^\top \boldsymbol{\Sigma}_\mathbf{y} \mathbf{B}_D = \mathbf{I}_d$ for arbitrary positive semidefinite $\widetilde{\boldsymbol{\Sigma}}_\mathbf{y}$. We can think $\widetilde{\boldsymbol{\Sigma}}_\mathbf{x}$ and $\widetilde{\boldsymbol{\Sigma}}_\mathbf{y}$ as the free parameters that adjust noise level in $\mathbf{x}$ and and $\mathbf{y}$ respectively. In this notation, the joint distribution of $(\mathbf{x}, \mathbf{y})$ is

$$\begin{bmatrix} \mathbf{x} \\ \mathbf{y} \end{bmatrix} \;\sim\; N\left(\begin{bmatrix} \boldsymbol{\mu}_\mathbf{x} \\ \boldsymbol{\mu}_\mathbf{y} \end{bmatrix}, \begin{bmatrix} \mathbf{Q}_\mathbf{x} \mathbf{R}_\mathbf{x}^{-\top} \mathbf{R}_\mathbf{x}^{-1} \mathbf{Q}_\mathbf{x}^\top + \widetilde{\mathbf{Q}}_\mathbf{x} \widetilde{\boldsymbol{\Sigma}}_\mathbf{x} \widetilde{\mathbf{Q}}_\mathbf{x}^\top & \mathbf{Q}_\mathbf{x} \mathbf{R}_\mathbf{x}^{-\top} \mathrm{diag}(\boldsymbol{\rho}_D) \mathbf{R}_\mathbf{y}^{-1} \mathbf{Q}_\mathbf{y}^\top \\ \mathbf{Q}_\mathbf{y} \mathbf{R}_\mathbf{y}^{-\top} \mathrm{diag}(\boldsymbol{\rho}_D) \mathbf{R}_\mathbf{x}^{-1} \mathbf{Q}_\mathbf{x}^\top & \mathbf{Q}_\mathbf{y} \mathbf{R}_\mathbf{y}^{-\top} \mathbf{R}_\mathbf{y}^{-1} \mathbf{Q}_\mathbf{y}^\top + \widetilde{\mathbf{Q}}_\mathbf{y} \widetilde{\boldsymbol{\Sigma}}_\mathbf{y} \widetilde{\mathbf{Q}}_\mathbf{y}^\top \end{bmatrix} \right). \tag{3.16}$$

and the generative model is

$$\begin{aligned} \mathbf{z} &\sim N(\mathbf{0}, \mathbf{I}_D) \\ \mathbf{x}|\mathbf{z} = \boldsymbol{z} &\sim N(\mathbf{Q}_\mathbf{x} \mathbf{R}_\mathbf{x}^{-\top} \mathbf{M}_\mathbf{x} \boldsymbol{z} + \boldsymbol{\mu}_\mathbf{x}, \mathbf{Q}_\mathbf{x} \mathbf{R}_\mathbf{x}^{-\top} (\mathbf{I}_D - \mathbf{M}_\mathbf{x} \mathbf{M}_\mathbf{x}^\top) \mathbf{R}_\mathbf{x}^{-1} \mathbf{Q}_\mathbf{x}^\top + \widetilde{\mathbf{Q}}_\mathbf{x} \widetilde{\boldsymbol{\Sigma}}_\mathbf{x} \widetilde{\mathbf{Q}}_\mathbf{x}^\top) \\ \mathbf{y}|\mathbf{z} = \boldsymbol{z} &\sim N(\mathbf{Q}_\mathbf{y} \mathbf{R}_\mathbf{y}^{-\top} \mathbf{M}_\mathbf{y} \boldsymbol{z} + \boldsymbol{\mu}_\mathbf{y}, \mathbf{Q}_\mathbf{y} \mathbf{R}_\mathbf{y}^{-\top} (\mathbf{I}_D - \mathbf{M}_\mathbf{y} \mathbf{M}_\mathbf{y}^\top) \mathbf{R}_\mathbf{y}^{-1} \mathbf{Q}_\mathbf{y}^\top + \widetilde{\mathbf{Q}}_\mathbf{y} \widetilde{\boldsymbol{\Sigma}}_\mathbf{y} \widetilde{\mathbf{Q}}_\mathbf{y}^\top). \end{aligned}$$

3.5.2 Simulation Study 1 : TCCA and TSCCA

To show the performance of TCCA and TSCCA models in various situation, we conduct several simulations. First, we consider an ordinary vector-valued data. We set a sparse signal $\mathbf{a}$ and $\mathbf{b}$, which have five nonzero elements among 100 coordinates each.

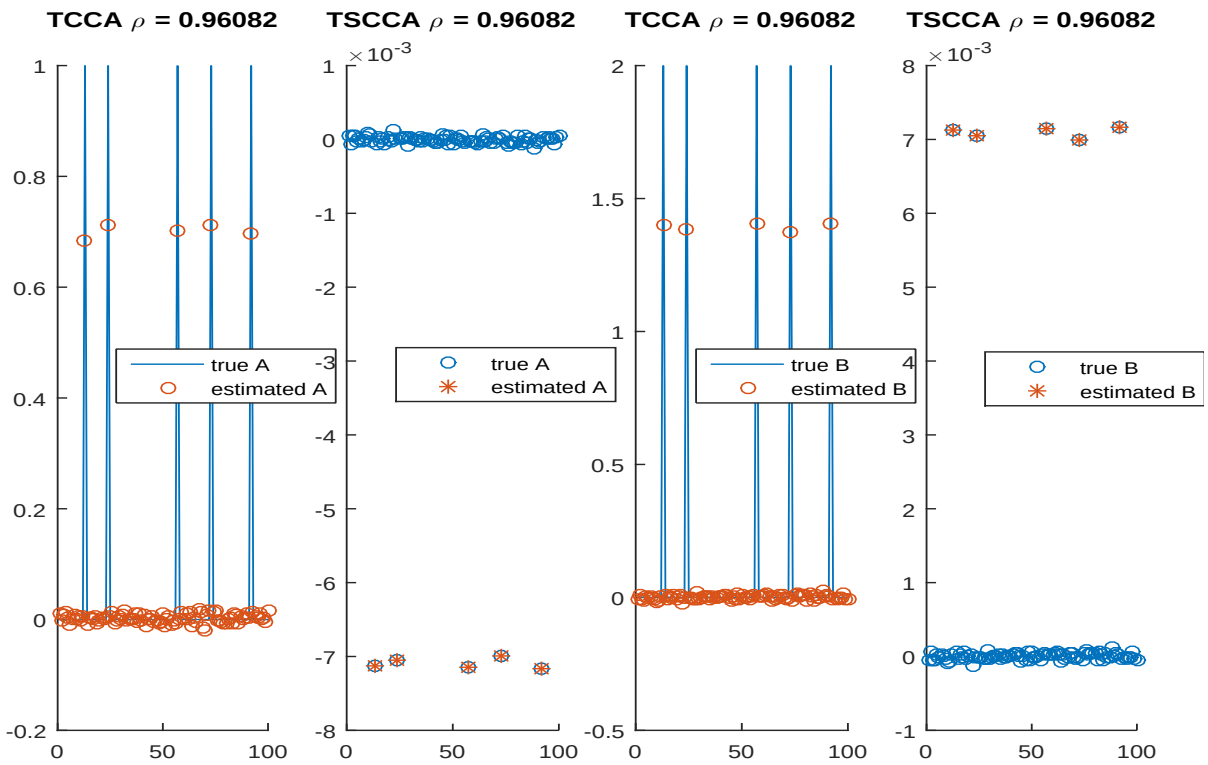


Figure 3.1: Result of TCCA and TSCCA conducted on two mode-1 data sets. Left two panels shows estimated coefficients for $\mathbf{X}$, while right two panels shows estimated coefficients for $\mathbf{Y}$.

Also we set a true canonical correlation between two data sets as 0.95. Under these true parameter values, we generated data sets $\mathbf{x}$ and $\mathbf{y}$ following the generative probabilistic CCA model introduced in previous section. We conduct analysis for those generated data set, $\mathbf{X} \in \mathbb{R}^{100\times 1000}$ and $\mathbf{Y} \in \mathbb{R}^{100\times 1000}$ each have 1000 observations.

For penalty parameters of TSCCA, $\lambda = 0.2$ and $\gamma = 1$ have been used. Figure 3.1 shows us the result. Left two panels show true coefficients $\mathbf{a}$ in blue color and estimated coefficients $\hat{\mathbf{a}}$ in red color of two models. Right two panels show true coefficients $\mathbf{b}$ in blue color and estimated coefficients $\hat{\mathbf{b}}$ in red color of two models. TSCCA result only shows estimated nonzero elements in each figures. We can observe that both models found

nonzero informational signal well with highly estimated canonical correlation values.

We conduct the second simulation to see the performance when our TCCA/TSCCA models deal with array-valued data sets, which is our original problem. We generated two data sets, one is vector data, but the other is now a matrix, mode-2, which is a tensor data. A vector data $\mathbf{x}$ has dimension $\mathbb{R}^{100}$ and a matrix data $\mathbf{Y}$ has dimension $\mathbb{R}^{64\times 64}$, and we generated 1000 observations from the joint distribution of them described in previous chapter. Thus our data set are mode-2 data $\mathbf{X} \in \mathbb{R}^{100\times 1000}$ and mode-3 data $\mathcal{Y} \in \mathbb{R}^{64\times 64\times 1000}$. Pre-defined true canonical correlation is 0.95 as same as the vector data case, and we used same true coefficients for $\mathbf{X}$. For the dataset y, we consider five type of image signals, square, cross, circle, triangle, and butterfly. We consider three values of the coefficient ranks to estimate, from 1 to 3 for the coefficient tensor of $\mathcal{Y}$. In last, we choose penalty parameter pairs based on the cross validation method. For 625 combinatorial pairs of λ and γ from the set $\{0.2, 0.4, \ldots, 5.0\}$, we conduct 5-fold cross validation. Exceptionally, the dataset which has true signal of $\mathcal{Y}$ as a triangle or butterfly image, we conducted 5-fold CV on the penalty parameter ranges $(0.2, 0.4, \ldots, 10)$, which has 1250 possible combination of λ and γ. Based on 5-fold CV, we select the parameter pair of (λ, γ) which maximizes following measure

$$\triangle_{choice} = \sum_{k=1}^{K} c(|\hat{\rho}_{(-k)}| + |\hat{\rho}_{(k)}|) + SR_{\mathbf{a}} + SR_{\mathbf{B}}$$

where
- c is the weight for correlation
- $|\hat{\rho}_{(-k)}|$ is the optimized canonical correlation using training data
- $|\hat{\rho}_{(-k)}|$ is the optimized canonical correlation using test data
- $SR_{\mathbf{a}}$ is $\frac{\#\text{ of nonzero elements}}{(100\times R_x)}$
- $SR_{\mathbf{B}}$ is $\frac{\#\text{ of nonzero elements}}{(2\times 64\times R_y)}$,

where $R_x = 1$ and $R_y \in \{1, 2, 3\}$ are the rank of two coefficient tensors $\mathcal{A}$ and $\mathcal{B}$ respectively. This measure chooses the parameter pairs such that the estimated correlation is high and the estimated coefficients are sparse. Using the weight scalar c, we can give different weights to principles for the choice of parameter pairs, between better recovery ($c > 1$) or sparser solution ($c < 1$). In our simulations, $c = 1$ when we deal with square, cross, and circle image. In other signals like triangle and butterfly, we used $c = 2$.

For conducting TSCCA, we adopt APGB algorithm in Section 2.3.2. In the simulation study of the Section 2.4.3, we conclude that for the sparsity constraint, APGB, APGE, and ADMM work fast and give a good result. Following those simulation results, we choose APGB for TSCCA with backtracking line search parameter $\eta = 0.2$. APGB shows very fast calculation. But it is a numerical algorithm which is affected a lot by the initial starting point. Thus, after we made a choice of penalty parameters we ran algorithms several times to get a good results.

Simulation results are listed in Tables 3.1, 3.2, 3.3, 3.4, and 3.5. From following five tables, we can make several observations. First, TCCA recovers the signal pretty well, as we can see in tables. Especially, the square image can be recovered enough with rank-1 for coefficients of $\mathcal{B}$, it is rather overfitted in the higher rank settings. TSCCA even gives us clearer image, with much less noise. For the cross image, rank-2 looks to be enough, especially TSCCA gives us almost clear image. Circle, triangle, and butterfly images seems to be not enough for rank-3, but still we can get a fair amount of information about the image when we see a rank-3 coefficient fitted results. And same as square and cross examples, TSCCA removes noise so that we can get better recovered images.

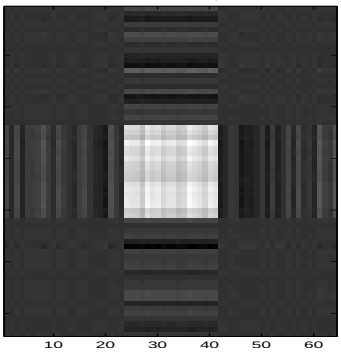
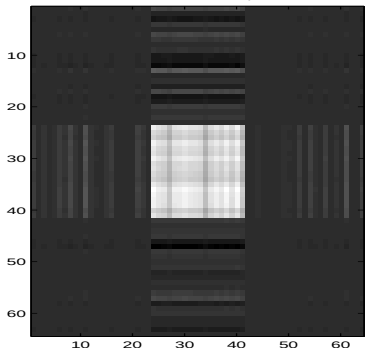
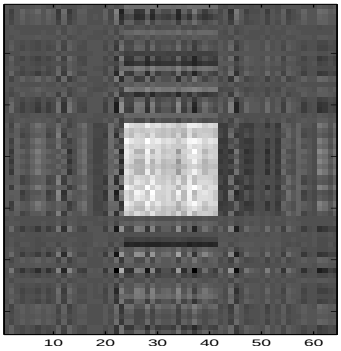
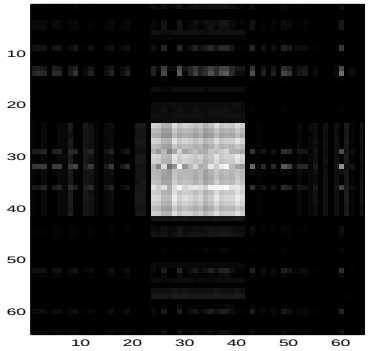
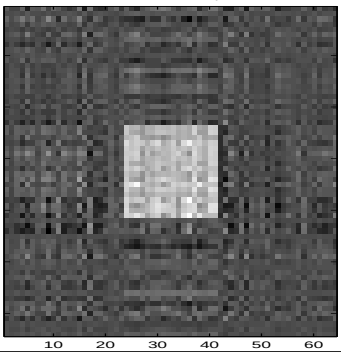
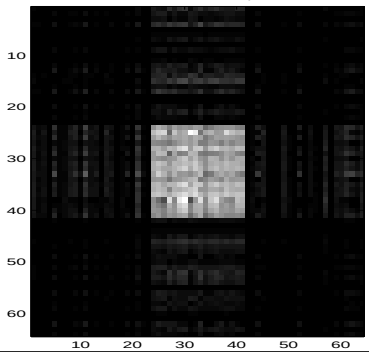
True Image	$(\text{rank}_{\mathbf{a}}, \text{rank}_{\mathbf{B}})$	Estimated image		(λ, γ) for TSCCA
	$(1,1)$			(1.2, 4.6)
	$(1,2)$			(4.8, 2.6)
	$(1,3)$			(1.6, 2.0)

Table 3.1: Results of TCCA and TSCCA for the square image. First column shows a true image, and the rest of 6 images are recovered image by TCCA and TSCCA. Among estimated 6 images, left three images are recovered by TCCA, while right three are by TSCCA. We can observe that rank-1 structured coefficient $\mathbf{B}$ catches the signal well enough, higher rank shows overfitting. Also we can observe that image from TSCCA is much clearer than the one from the TCCA.

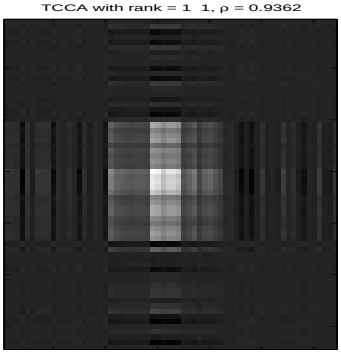
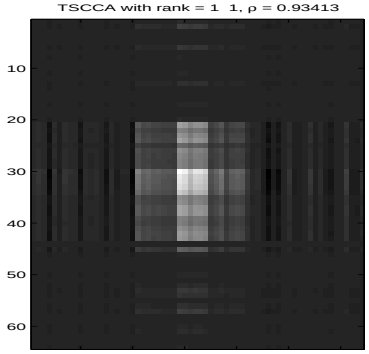
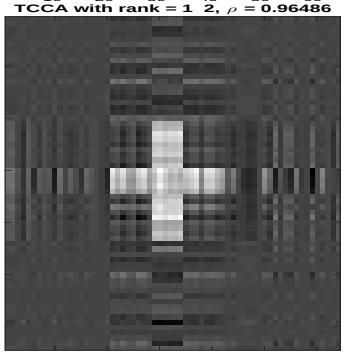
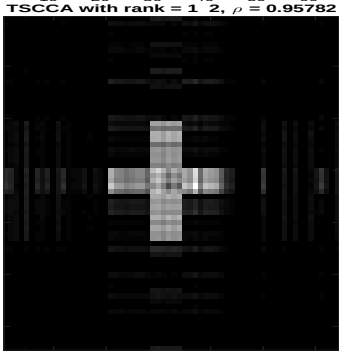
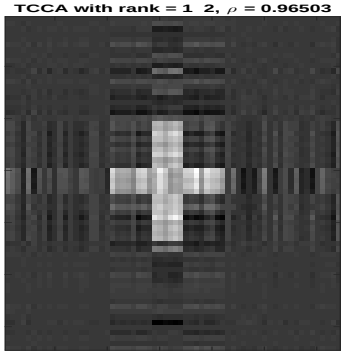
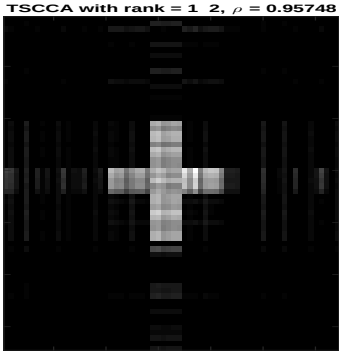
True Image	$(\text{rank}_{\mathbf{a}}, \text{rank}_{\mathbf{B}})$	Estimated image		(λ, γ) for TSCCA
	$(1,1)$			(1.2, 4.6)
	$(1,2)$			(4.8, 2.6)
	$(1,3)$			(1.6, 2.0)

Table 3.2: Results of TCCA and TSCCA for the cross image. First column shows the true image, and the rest of 6 images are recovered images by TCCA and TSCCA. Among estimated 6 images, left three images are recovered by TCCA, while right three are by TSCCA. We can observe that higher rank structure imposed model shows better signal recovery. Also, we can observe that image from TSCCA is much clearer than the one from the TCCA.

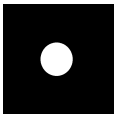
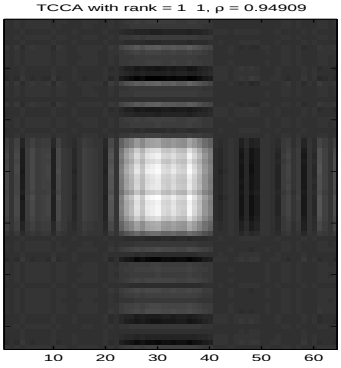
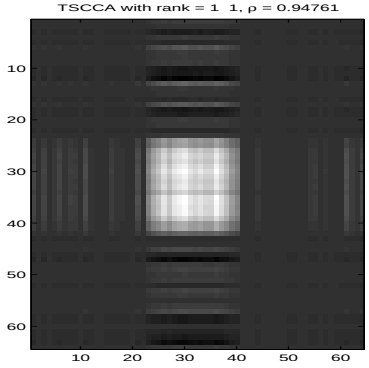
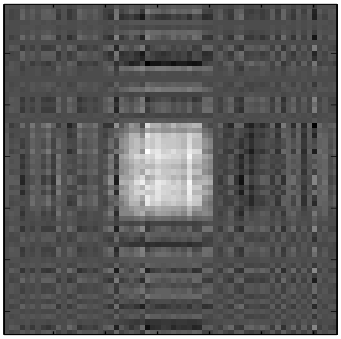
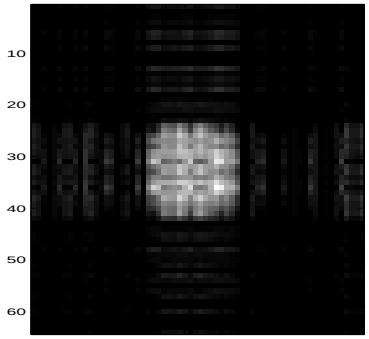
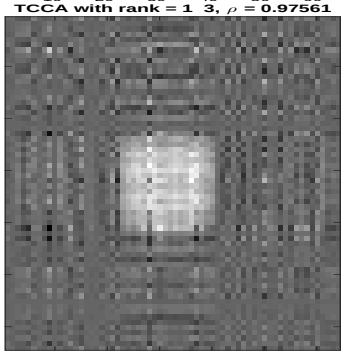
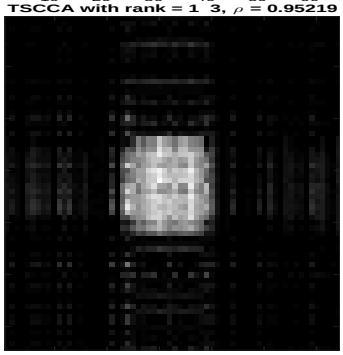
True Image	$(\text{rank}_{\mathbf{a}}, \text{rank}_{\mathbf{B}})$	Estimated image		(λ, γ) for TSCCA
	$(1,1)$			(1.2, 4.6)
	$(1,2)$			(4.8, 2.6)
	$(1,3)$			(1.6, 2.0)

Table 3.3: Results of TCCA and TSCCA for the circle image. First column shows the true image, and the rest of 6 images are recovered images by TCCA and TSCCA. Among estimated 6 images, left three images are recovered by TCCA, while right three are by TSCCA. We can observe that higher rank structure imposed model shows better signal recovery. Also, we can observe that image from TSCCA is much clearer than the one from the TCCA.

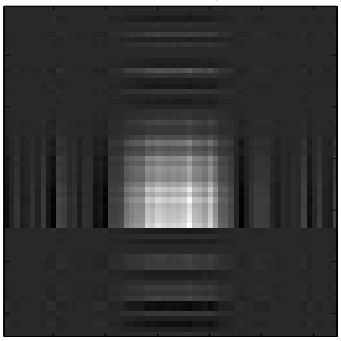
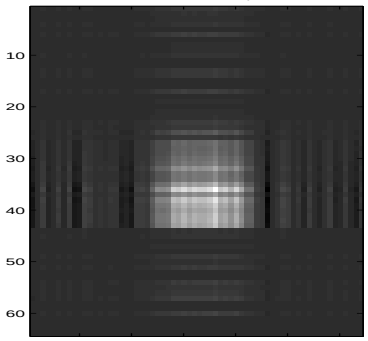
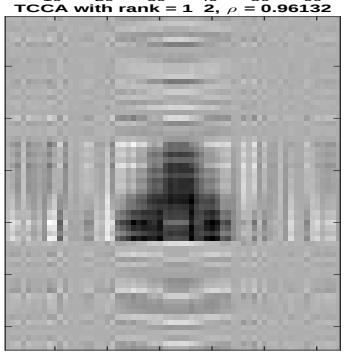
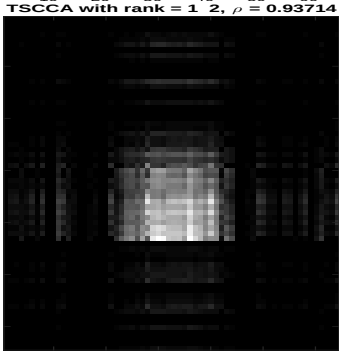
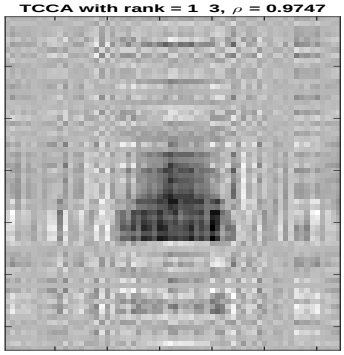
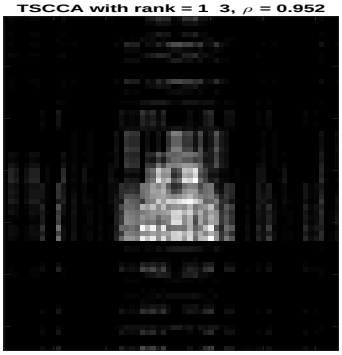
True Image	$(\text{rank}_{\mathbf{a}}, \text{rank}_{\mathbf{B}})$	Estimated image		(λ, γ) for TSCCA
	$(1,1)$			(1.8, 4.2)
	$(1,2)$			(8.0, 5.4)
	$(1,3)$			(7.0, 5.6)

Table 3.4: Results of TCCA and TSCCA for the triangle image. First column shows the true image, and the rest of 6 images are recovered images by TCCA and TSCCA. Among estimated 6 images, left three images are recovered by TCCA, while right three are by TSCCA. We can observe that higher rank structure imposed model shows better signal recovery, but we may need higher rank to get better recovery of the triangle signal. Similarly with previous examples, we can observe that image from TSCCA is much clearer than the one from the TCCA.

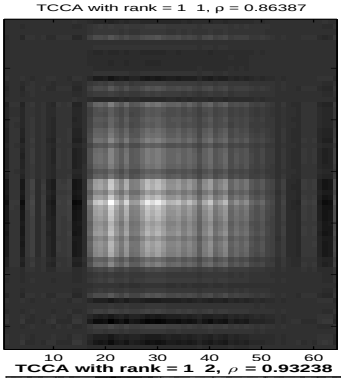
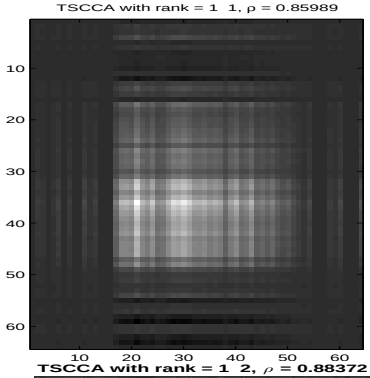
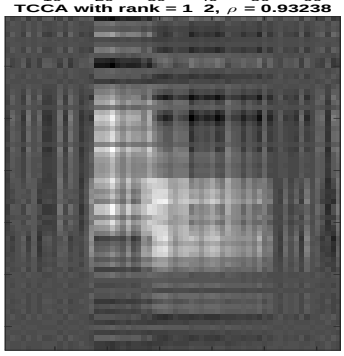
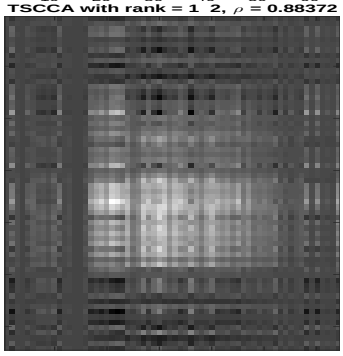
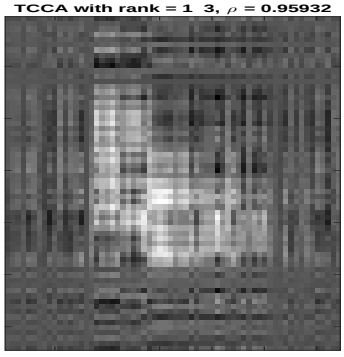
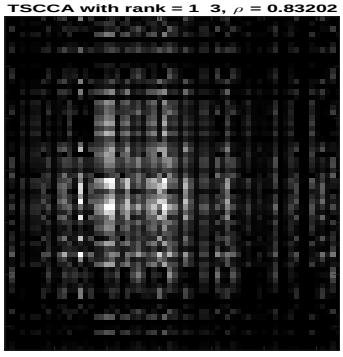
True Image	$(\text{rank}_{\mathbf{a}}, \text{rank}_{\mathbf{B}})$	Estimated image	(λ, γ) for TSCCA
	$(1,1)$	 	(1.2, 4.6)
	$(1,2)$	 	(4.8, 2.6)
	$(1,3)$	 	(1.6, 2.0)

Table 3.5: Results of TCCA and TSCCA for the butterfly image. First column shows the true image, and the rest of 6 images are recovered images by TCCA and TSCCA. Among estimated 6 images, left three images are recovered by TCCA, while right three are by TSCCA. We can observe that higher rank structure imposed model shows better signal recovery, but we may need higher rank to get better recovery of the triangle signal. Similarly with previous examples, we can observe that image from TSCCA is much clearer than the one from the TCCA.

3.5.3 Simulation Study 2 : TCCA_SEP and TSCCA_SEP

We conduct the second simulation to assess the performance of TCCA_SEP and TSCCA_SEP model. We generated data using the same scheme as the first simulation, we use same value of true canonical correlation coefficient, and same two canonical coefficients. From those true value, we generate vector data $\mathbf{X} \in \mathbb{R}^{100 \times 1000}$ and matrix data $\mathcal{Y} \in \mathbb{R}^{64 \times 64 \times 1000}$ which possess 1000 samples. For true $\mathcal{Y}$ signal, we used five images same as the first simulation, square, cross, circle, triangle, and butterfly image. We consider three values of the coefficient ranks in the model, from 1 to 3 for the coefficient tensor of $\mathcal{Y}$. In last, we choose penalty parameter pairs based on several toy simulations, used value of the pair of parameters in each experiments are shown in the table. Results are displayed in next five Tables 3.6,3.7,3.8, 3.9 and 3.10

We can make several observations from following results. First, TCCA_SEP and TSCCA_SEP model works well. Square, cross, and circle images recovered very well, and TSCCA_SEP gives us even more clearer image. Even more, square and cross images are almost all recovered with low rank. For circle, triangle and butterfly images, it seems to require a higher rank for better recovery of the image. But still we can get a fair amount of information from the results. Important thing to remember is that we generated data from probabilistic model, not an array normal distribution which has separable covariance structure. In other words, data does not follow the population assumption of models. However our model still recovers signal very well, which is very impressive result. Secondly, even though we make a choice for the penalty parameters based on toy simulations, they work great. In most cases, TSCCA_SEP gives us clearer image than the TCCA_SEP. Exceptionally, TSCCA_SEP failed to recover butterfly image in rank-3 of $\mathcal{Y}$, it seems that higher values of penalty parameters is required.

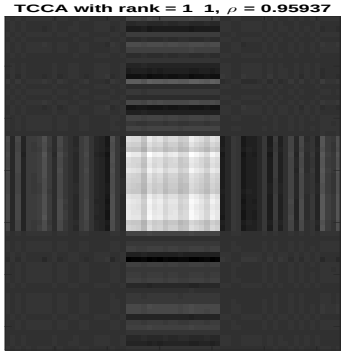
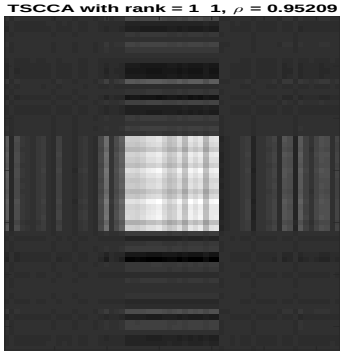
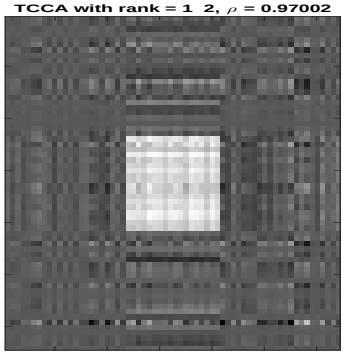
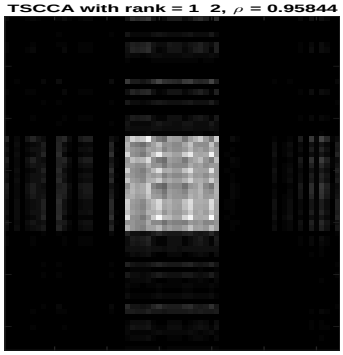
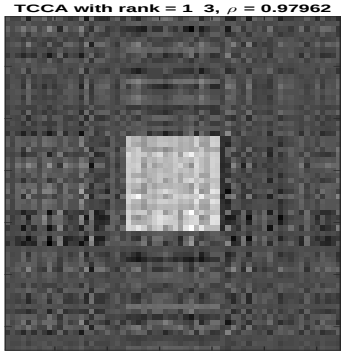
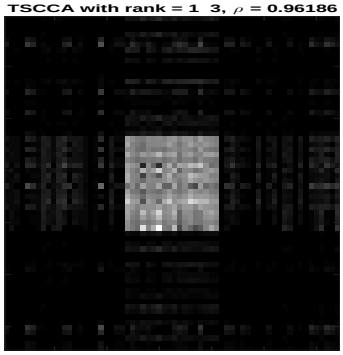
True Image	$(\text{rank}_{\mathbf{a}}, \text{rank}_{\mathbf{B}})$	Estimated image		(λ, γ) for TSCCA
	$(1,1)$			(0.5, 85)
	$(1,2)$			(2.0, , 2.0)
	$(1,3)$			(4.5, 4.5)

Table 3.6: Results of TCCA_SEP and TSCCA_SEP for the square image. First column shows a true image, and the rest of 6 images are recovered image by TCCA_SEP and TSCCA_SEP. Among estimated 6 images, left three images are recovered by TCCA_-SEP, while right three are by TSCCA_SEP. Same as TCCA and TSCCA, we can observe that rank-1 structured coefficient $\mathbf{B}$ catches the signal well enough, higher rank shows overfitting. Also, we can observe that image from TSCCA_SEP is much clearer than the one from the TCCA_SEP.

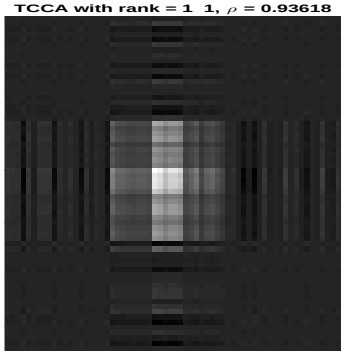
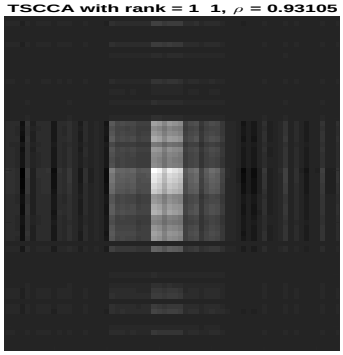
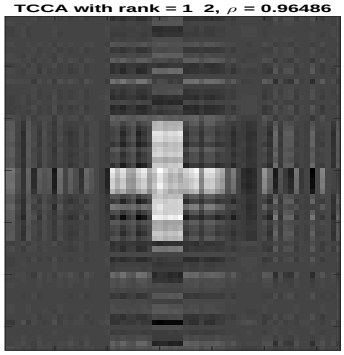
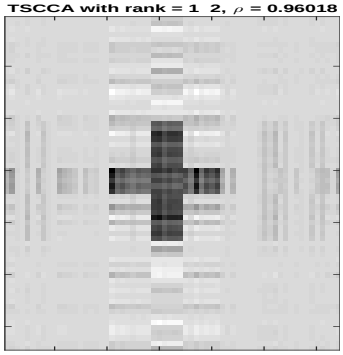
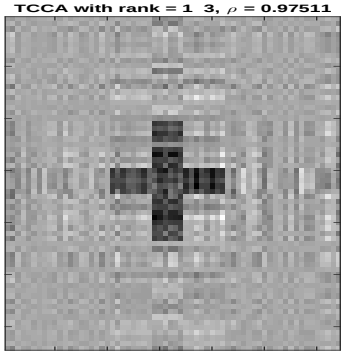
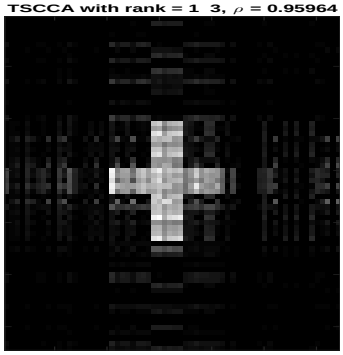
True Image	$(\text{rank}_{\mathbf{a}}, \text{rank}_{\mathbf{B}})$	Estimated image		(λ, γ) for TSCCA
	$(1,1)$			(1.5, 7.0)
	$(1,2)$			(2.0, 2.0)
	$(1,3)$			(4.5, 4.5)

Table 3.7: Results of TCCA_SEP and TSCCA_SEP for the cross image. First column shows a true image, and the rest of 6 images are recovered image by TCCA_SEP and TSCCA_SEP. Among estimated 6 images, left three images are recovered by TCCA_SEP, while right three are by TSCCA_SEP. It seems to be enough with rank-2 of the coefficient $\mathcal{B}$, we can see that the model with rank-3 of $\mathcal{B}$ gives us a overfitted result. Also we can observe that image from TSCCA_SEP is much clearer than the one from the TCCA_SEP.

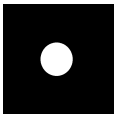
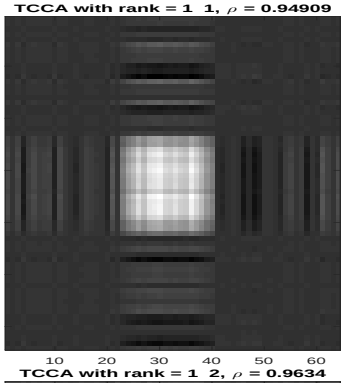
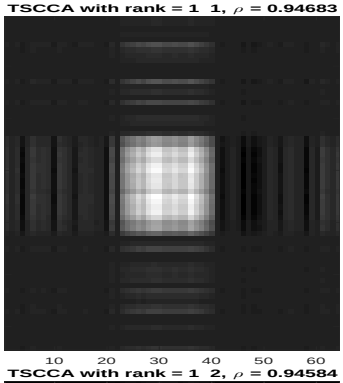
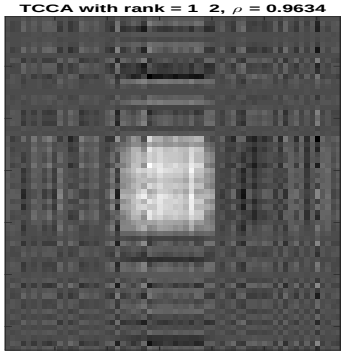
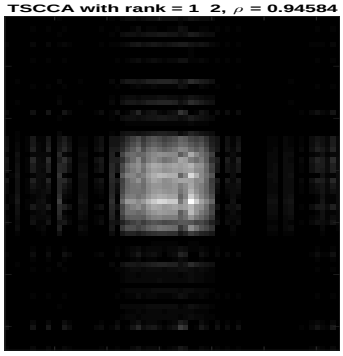
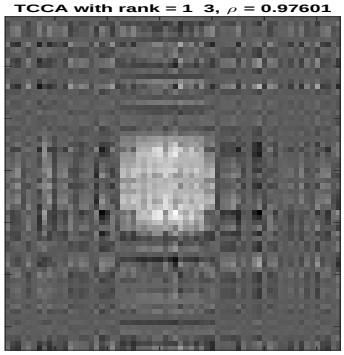
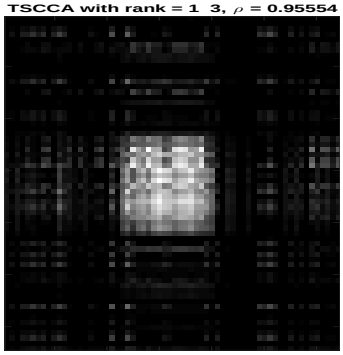
True Image	$(\text{rank}_{\mathbf{a}}, \text{rank}_{\mathbf{B}})$	Estimated image		(λ, γ) for TSCCA
	$(1,1)$			(5.5, 8.0)
	$(1,2)$			(2.0, 2.0)
	$(1,3)$			(4.5, 4.5)

Table 3.8: Results of TCCA_SEP and TSCCA_SEP for the circle image. First column shows a true image, and the rest of 6 images are recovered image by TCCA_SEP and TSCCA_SEP. Among estimated 6 images, left three images are recovered by TCCA_SEP, while right three are by TSCCA_SEP. It seems that higher rank of $\mathcal{B}$ is required for better recovery. But still We can observe that image from TSCCA_SEP is much clearer than the one from the TCCA_SEP.

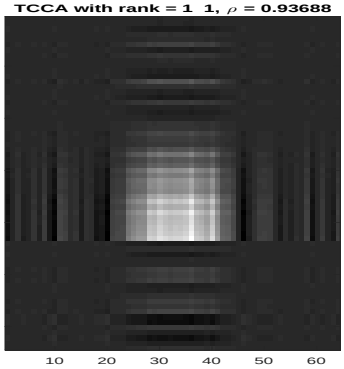
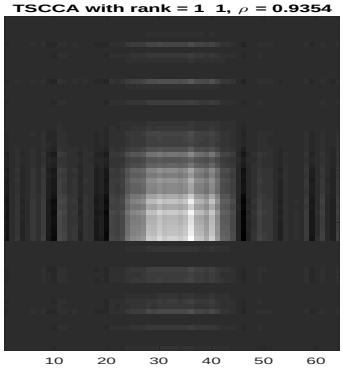
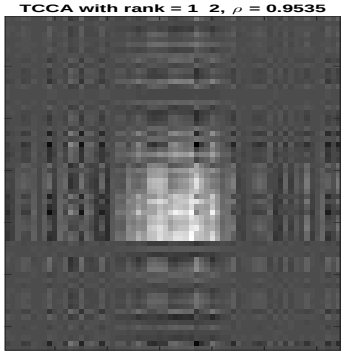
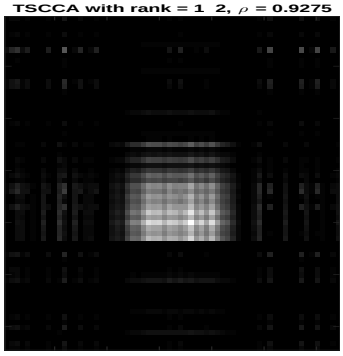
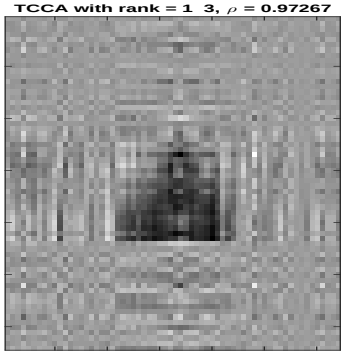
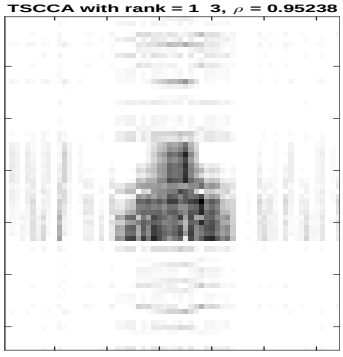
True Image	$(\text{rank}_{\mathbf{a}}, \text{rank}_{\mathbf{B}})$	Estimated image		(λ, γ) for TSCCA
	$(1,1)$			(5.5, 9.0)
	$(1,2)$			(2.0, 2.0)
	$(1,3)$			(4.5, 4.5)

Table 3.9: Results of TCCA_SEP and TSCCA_SEP for the triangle image. First column shows a true image, and the rest of 6 images are recovered image by TCCA_SEP and TSCCA_SEP. Among estimated 6 images, left three images are recovered by TCCA_SEP, while right three are by TSCCA_SEP. We can see that higher ranked mode gives better recovered results. It seems that rank bigger than 3 may required for better recovered image. And we can observe again that image from TSCCA_SEP is much clearer than the one from the TCCA_SEP.

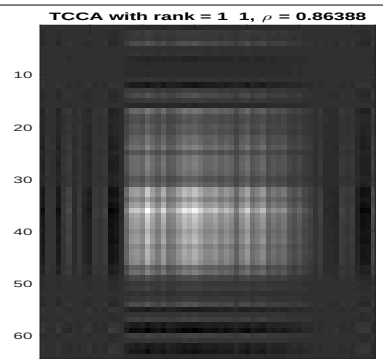
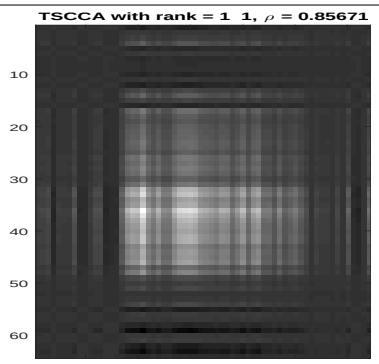
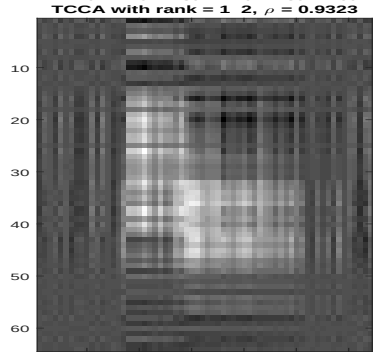
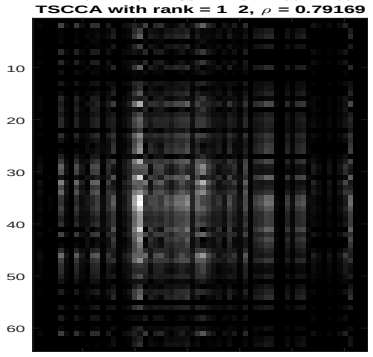
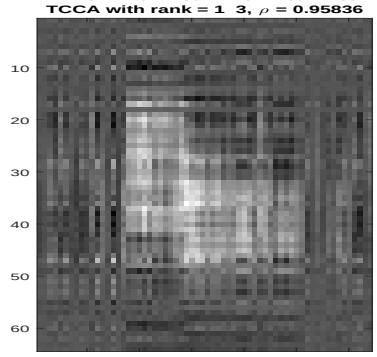
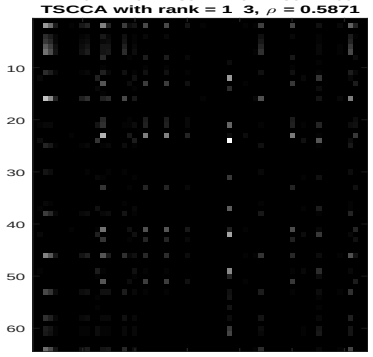
True Image	$(\text{rank}_{\mathbf{a}}, \text{rank}_{\mathbf{B}})$	Estimated image		(λ, γ) for TSCCA
	$(1,1)$			(1.2, 4.6)
	$(1,2)$			(4.8, 2.6)
	$(1,3)$			(1.6, 2.0)

Table 3.10: Results of TCCA_SEP and TSCCA_SEP for the butterfly image. First column shows a true image, and the rest of 6 images are recovered image by TCCA_SEP and TSCCA_SEP. Among estimated 6 images, left three images are recovered by TCCA_SEP, while right three are by TSCCA_SEP. We can see that higher ranked mode gives better recovered results. It seems that rank bigger than 3 may required for better recovered image. Exceptionally, we failed to recover the image under the model with rank-3 of $\mathcal{B}$ of TSCCA_SEP. It seems that we may try higher ranks or bigger penalty parameters to get a good results. But in general we can observe again that image from TSCCA_SEP is much clearer than the one from the TCCA_SEP.

3.6 Discussion

In this Chapter, we develop a novel method to find a set of coefficients that maximizes linear relationship between each linear combination of two data sets, not only for an ordinary vector-valued data but also for a multi-dimensional data array, so called tensor. Vectorizing data and applying ordinary CCA or SCCA methods require formidable number of parameters, which causes a big problem. Even though we apply SCCA, still its dimensionality is too huge, if we consider a big data such as fMRI data. Moreover, spatial information that data contains in its own structure can be lost, which is a big loss in the view of researchers.

To address these issues, we propose a new framework for tensor CCA. We impose a low rank structure on coefficients of $\mathcal{X}$ and $\mathcal{Y}$ so that we can significantly reduce the number of parameters while we can preserve its array structure. Moreover, our method allows us to update blocks of parameters alternately one at a time, which is the great advantage for us especially in terms of computational cost and time. And for the last, our models avoid to deal with singular sample covariance estimates. Instead we work on the "compressed covariance matrices", which is likely to be a full rank. It may be a hard problem if we have to deal with the sample covariance matrices directly especially we have small sample size, since estimated sample covariance matrices may be singular. However, our methods alleviate these issues.

Furthermore, imposition of separable covariance structure improves statistical estimation efficiency and reduce computation burden. Simulation results show that though the generative model does not follow the separable covariance structure, model still works well.

On top of TCCA and TCCA_SEP models, we also proposed sparse version of these

models by adding l_1 constraints on the parameter blocks, and solved them by using numerical methods developed in Chapter 2. Sparsity is the frequently preferred or required constraint, but it is hard to solve l_1 constrained problem due to its non-differentiability. However adopting numerical optimization algorithm, we can easily add sparsity constraint and get a solution, which works well as we saw in the simulation results.

Still we have more consideration on this problem. We put the separable covariance matrix structure on covariance matrices of each data set, but not in cross covariance matrices. Thus, our future research will concentrate on developing the model which has separable cross covariance structure and sparsity constrained model of that. More broadly, our work is done based on the generalized eigenvalue problem (GEP) framework, which implies that other multivariate analysis methods which can be transformed to GEP can be extended for the tensor data set using the same framework of this Chapter. Thus, our future research will also focus on the extension of this model to other multivariate data analysis techniques.

REFERENCES

Allen, G. I. and Maletić-Savatić, M. (2011). Sparse non-negative generalized PCA with applications to metabolomics. *Bioinformatics*, 27(21):3029–3035.

Bach, F. R. and Jordan, M. I. (2005). A probabilistic interpretation of canonical correlation analysis.

Beck, A. and Teboulle, M. (2009). Gradient-based algorithms with applications to signal recovery. *Convex Optimization in Signal Processing and Communications*.

Borga, M., Landelius, T., and Knutsson, H. (1997). A unified approach to PCA, PLS, MLR and CCA.

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

Boyle, J. P. and Dykstra, R. L. (1986). A method for finding projections onto the intersection of convex sets in hilbert spaces. In *Advances in order restricted statistical inference*, pages 28–47. Springer.

Chalise, P. and Fridley, B. L. (2012). Comparison of penalty functions for sparse canonical correlation analysis. *Computational statistics & data analysis*, 56(2):245–254.

Chi, E. C., Allen, G. I., Zhou, H., Kohannim, O., Lange, K., and Thompson, P. M. (2013). Imaging genetics via sparse canonical correlation analysis. In *Biomedical Imaging (ISBI), 2013 IEEE 10th International Symposium on*, pages 740–743. IEEE.

Chi, E. C. and Lange, K. (2014). Stable estimation of a covariance matrix guided by nuclear norm penalties. *Computational statistics & data analysis*, 80:117–128.

De Bie, T., Cristianini, N., and Rosipal, R. (2005). Eigenproblems in pattern recognition. In *Handbook of Geometric Computing*, pages 129–167. Springer.

De Lathauwer, L. (2006). A link between the canonical decomposition in multilinear algebra and simultaneous matrix diagonalization. *SIAM Journal on Matrix Analysis and Applications*, 28(3):642–666.

Ding, S. and Cook, R. D. (2015a). Higher-order sliced inverse regressions. *Wiley Interdisciplinary Reviews: Computational Statistics*.

Ding, S. and Cook, R. D. (2015b). Tensor sliced inverse regression. *Journal of Multivariate Analysis*, 133:216–231.

Friman, O., Borga, M., Lundberg, P., and Knutsson, H. (2003). Adaptive analysis of fMRI data. *NeuroImage*, 19(3):837–845.

Friman, O., Cedefamn, J., Lundberg, P., Borga, M., and Knutsson, H. (2001). Detection of neural activity in functional mri using canonical correlation analysis. *Magnetic Resonance in Medicine*, 45(2):323–330.

Gabay, D. and Mercier, B. (1976). A dual algorithm for the solution of nonlinear variational problems via finite element approximation. *Computers & Mathematics with Applications*, 2(1):17–40.

Gerard, D. and Hoff, P. (2015). Equivariant minimax dominators of the MLE in the array normal model. *Journal of multivariate analysis*, 137:32–49.

Ghadimi, S. and Lan, G. (2013). Accelerated gradient methods for nonconvex nonlinear and stochastic programming. *arXiv preprint arXiv:1310.3787*.

Golub, G. H. and van Van Loan, C. F. (1996). *Matrix computations (Johns Hopkins studies in mathematical sciences)*. The Johns Hopkins University Press.

Hardoon, D. R. and Shawe-Taylor, J. (2011). Sparse canonical correlation analysis. *Machine Learning*, 83(3):331–353.

Hestenes, M. R. and Karush, W. (1951a). A method of gradients for the calculation of the characteristic roots and vectors of a real symmetric matrix. *Journal of Research of the National Bureau of Standards*, Vol. 47(No.1):45–61.

Hestenes, M. R. and Karush, W. (1951b). Solutions of $ax = \lambda bx$. *Journal of Research of the National Bureau of Standards*, Vol. 47:471–478.

Hoff, P. D. et al. (2011). Separable covariance arrays via the tucker product, with applications to multivariate relational data. *Bayesian Analysis*, 6(2):179–196.

Höskuldsson, A. (1988). PLS regression methods. *Journal of chemometrics*, 2(3):211–228.

Hotelling, H. (1936). Relations between two sets of variates. *Biometrika*, pages 321–377.

Hoyer, P. O. (2004). Non-negative matrix factorization with sparseness constraints. *The Journal of Machine Learning Research*, 5:1457–1469.

Huettel, S. A., Song, A. W., and McCarthy, G. (2004). *Functional magnetic resonance imaging*, volume 1. Sinauer Associates Sunderland.

Johnstone, I. M. and Lu, A. Y. (2009). On consistency and sparsity for principal components analysis in high dimensions. *Journal of the American Statistical Association*, 104(486).

Journée, M., Nesterov, Y., Richtárik, P., and Sepulchre, R. (2010). Generalized power method for sparse principal component analysis. *The Journal of Machine Learning Research*, 11:517–553.

Joyner, A. H., Bloss, C. S., Bakken, T. E., Rimol, L. M., Melle, I., Agartz, I., Djurovic, S., Topol, E. J., Schork, N. J., Andreassen, O. A., et al. (2009). A common mecp2 haplotype associates with reduced cortical surface area in humans in two independent populations. *Proceedings of the National Academy of Sciences*, 106(36):15483–15488.

Kim, T.-K. and Cipolla, R. (2009). Canonical correlation analysis of video volume tensors for action categorization and detection. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 31(8):1415–1428.

Kolda, T. G. (2006). *Multilinear operators for higher-order decompositions*. United States. Department of Energy.

Kolda, T. G. and Bader, B. W. (2009). Tensor decompositions and applications. *SIAM review*, 51(3):455–500.

Laub, A. J. (2005). *Matrix analysis for scientists and engineers*. Siam.

Ledoit, O. and Wolf, M. (2004). A well-conditioned estimator for large-dimensional covariance matrices. *Journal of multivariate analysis*, 88(2):365–411.

Ledoit, O., Wolf, M., et al. (2012). Nonlinear shrinkage estimation of large-dimensional covariance matrices. *The Annals of Statistics*, 40(2):1024–1060.

Li, L. and Zhang, X. (2015). Parsimonious tensor response regression. *arXiv preprint arXiv:1501.07815*.

Li, X. (2014). Tensor based statistical models with applications in neuroimaging data analysis.

Li, X., Zhou, H., and Li, L. (2013). Tucker tensor regression and neuroimaging analysis. *arXiv preprint arXiv:1304.5637*.

Liu, X. and Sidiropoulos, N. D. (2001). Cramer-rao lower bounds for low-rank decomposition of multidimensional arrays. *Signal Processing, IEEE Transactions on*, 49(9):2074–2086.

Lykou, A. and Whittaker, J. (2010). Sparse CCA using a lasso with positivity constraints. *Computational Statistics & Data Analysis*, 54(12):3144–3157.

Maletić-Savatić, M., Vingara, L., Manganas, L., Li, Y., Zhang, S., Sierra, A., Hazel, R., Smith, D., Wagshul, M., Henn, F., et al. (2008). Metabolomics of neural progenitor cells: a novel approach to biomarker discovery. In *Cold Spring Harbor symposia on quantitative biology*, volume 73, pages 389–401. Cold Spring Harbor Laboratory Press.

Manganas, L. N., Zhang, X., Li, Y., Hazel, R. D., Smith, S. D., Wagshul, M. E., Henn, F., Benveniste, H., Djurić, P. M., Enikolopov, G., et al. (2007). Magnetic resonance spectroscopy identifies neural progenitor cells in the live human brain. *Science*, 318(5852):980–985.

Michelot, C. (1986). A finite algorithm for finding the projection of a point onto the canonical simplex of $\mathbb{R}^n$. *Journal of Optimixation Theory and Applicaions*, 50(1):195–200.

Miranda, M., Zhu, H., and Ibrahim, J. G. (2015). TPRM: Tensor partition regression models with applications in imaging biomarker detection. *arXiv preprint arXiv:1505.05482*.

Nesterov, Y. (2004). Introductory lectures on convex optimization, volume 87 of applied optimization.

Nesterov, Y. E. (1983). A method of solving a concave programming problem with convergence rate $\mathcal{O}(1/k^2)$. *Soviet Mathematics Doklady*, 27(2):372–376.

Niedermeyer, E. and da Silva, F. L. (2005). *Electroencephalography: basic principles, clinical applications, and related fields*. Lippincott Williams & Wilkins.

Parkhomenko, E., Tritchler, D., and Beyene, J. (2009). Sparse canonical correlation analysis with application to genomic data integration. *Statistical Applications in Genetics and Molecular Biology*, 8(1):1–34.

Shen, H. and Huang, J. Z. (2008). Sparse principal component analysis via regularized low rank matrix approximation. *Journal of multivariate analysis*, 99(6):1015–1034.

Sidiropoulos, N. D. and Bro, R. (2000). On the uniqueness of multilinear decomposition of n-way arrays. *Journal of chemometrics*, 14(3):229–239.

Sigg, C., Fischer, B., Ommer, B., Roth, V., and Buhmann, J. (2007). Nonnegative CCA for audiovisual source separation. In *Machine Learning for Signal Processing, 2007 IEEE Workshop on*, pages 253–258. IEEE.

Sigg, C. D. and Buhmann, J. M. (2008). Expectation-maximization for sparse and non-negative PCA. In *Proceedings of the 25th international conference on Machine learning*, pages 960–967. ACM.

Stein, J. L., Hua, X., Lee, S., Ho, A. J., Leow, A. D., Toga, A. W., Saykin, A. J., Shen, L., Foroud, T., Pankratz, N., et al. (2010). Voxelwise genome-wide association study (vgwas). *Neuroimage*, 53(3):1160–1174.

Tan, X., Zhang, Y., Tang, S., Shao, J., Wu, F., and Zhuang, Y. (2013). Logistic tensor regression for classification. In *Intelligent Science and Intelligent Data Engineering*, pages 573–581. Springer.

Valk, P. E. (2003). *Positron emission tomography: basic sciences*. Springer Science & Business Media.

Vapnik, V. N. and Vapnik, V. (1998). *Statistical learning theory*, volume 1. Wiley New York.

Waaijenborg, S., Verselewel de Witt Hamer, P. C., and Zwinderman, A. H. (2008). Quantifying the association between gene expressions and DNA-markers by penalized canonical correlation analysis. *Statistical Applications in Genetics and Molecular Biology*, 7(1).

Witten, D. M., Tibshirani, R., and Hastie, T. (2009). A penalized matrix decomposition, with applications to sparse principal components and canonical correlation analysis. *Biostatistics*, 10(3):515–534.

Wold, S., Sjöström, M., and Eriksson, L. (2001). PLS-regression: a basic tool of chemometrics. *Chemometrics and Intelligent Laboratory Systems*, 58(2):109 – 130. {PLS} Methods.

Zass, R. and Shashua, A. (2006). Nonnegative sparse PCA. In *Advances in Neural Information Processing Systems*, pages 1561–1568.

Zhang, X., Li, L., Zhou, H., Shen, D., et al. (2014). Tensor generalized estimating equations for longitudinal imaging analysis. *arXiv preprint arXiv:1412.6592*.

Zhao, Z. and Xiao, Z. (2011). Efficient regressions via optimally combining quantile information. `http://www.economics.illinois.edu/seminars/econometrics/documents/CQR09302011.pdf`.

Zhou, H. and Li, L. (2014). Regularized matrix regression. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 76(2):463–483.

Zhou, H., Li, L., and Zhu, H. (2013). Tensor regression with applications in neuroimaging data analysis. *Journal of the American Statistical Association*, 108(502):540–552.

Zou, H., Hastie, T., and Tibshirani, R. (2006). Sparse principal component analysis. *Journal of computational and graphical statistics*, 15(2):265–286.

APPENDICES

Appendix A

Algorithms for CGE

A.1 Accelerated Projected Gradient Method with Exact Line Search (APGE)

Algorithm 1: Accelerated Projected Gradient Method with Exact Line Search (APGE)

Initialize $\mathbf{w}^{(0)} = \mathbf{w}^{(1)} \in \Delta,\ \alpha^{(0)} = 0,\ \alpha^{(1)} = 1$
repeat
  $\mathbf{u} \leftarrow \mathbf{w}^{(t)} + \left(\frac{\alpha^{(t-1)}-1}{\alpha^{(t)}}\right)(\mathbf{w}^{(t)} - \mathbf{w}^{(t-1)})$ ▷ `Extrapolation`
  $\mathbf{v} \leftarrow 2[\mathbf{A} - R(\mathbf{u})\mathbf{B}]\mathbf{u}/(\mathbf{u}^{\intercal}\mathbf{B}\mathbf{u})$
  $c_1, c_2 \leftarrow$ roots of a quadratic with coefficients (2.5) ▷ `Exact line search`
  $\mathbf{w}_{\text{temp},1} \leftarrow P_{\Delta}(\mathbf{u} + c_1\mathbf{v})$ ▷ `Projection`
  $\mathbf{w}_{\text{temp},2} \leftarrow P_{\Delta}(\mathbf{u} + c_2\mathbf{v})$ ▷ `Projection`
  if $R(\mathbf{w}_{temp,1}) < R(\mathbf{w}_{temp,1})$ **then**
    $\mathbf{w}^{(t+1)} \leftarrow \mathbf{w}_{\text{temp},1}$
  else
    $\mathbf{w}^{(t+1)} \leftarrow \mathbf{w}_{\text{temp},2}$
  end
  $\alpha^{(t+1)} \leftarrow (1 + \sqrt{1 + (2\alpha^{(t)})^2})/2$
until *objective value converges*

A.2 Accelerated Proximal Gradient Method with Backtracking Line Search (APGB)

Algorithm 2: Accelerated Proximal Gradient Method with Backtracking Line Search (APGB)

Initialize $\mathbf{w}^{(0)} = \mathbf{w}^{(1)} \in \Delta,\ s^{(0)} > 0,\ 0 < \eta < 1,\ \alpha^{(0)} = 0,\ \alpha^{(1)} = 1$
repeat
  $\mathbf{u} \leftarrow \mathbf{w}^{(t)} + \left(\frac{\alpha^{(t-1)}-1}{\alpha^{(t)}}\right)(\mathbf{w}^{(t)} - \mathbf{w}^{(t-1)})$ ▷ `Extrapolation`
  $\mathbf{v} \leftarrow (2[\mathbf{A} - R(\mathbf{u})\mathbf{B}]\mathbf{u}) / (\mathbf{u}^{\intercal}\mathbf{B}\mathbf{u})$
  repeat
    $\mathbf{w}^{(t)}_{\text{temp}} \leftarrow P_{\Delta}(\mathbf{u} - s^{(t)}\mathbf{v})$ ▷ `Projection`
    $Q \leftarrow R(\mathbf{u}) + \langle \mathbf{w}^{(t)}_{\text{temp}} - \mathbf{u}, \mathbf{v} \rangle + (1/2s^{(t)})\|\mathbf{w}^{(t)}_{\text{temp}} - \mathbf{u}\|^2$
    $s^{(t)} \leftarrow \eta s^{(t)}$ ▷ `Backtracking rule`
    $\mathbf{w}^{(t+1)} \leftarrow \mathbf{w}^{(t)}_{\text{temp}}$
  until $R(\mathbf{w}^{(t)}_{temp}) \leq Q$
  $s^{(t+1)} \leftarrow s^{(t)}$
  $\alpha^{(t+1)} \leftarrow (1 + \sqrt{1 + (2\alpha^{(t)})^2})/2$
until *objective value converges*

A.3 Accelerated Projected Gradient Descent Method for Nonconvex function (APGNC)

Algorithm 3: Accelerated Projected Gradient Descent Method for Nonconvex function (APGNC)

Initialize $\mathbf{w}^{(0)} = \mathbf{w}_{ag}^{(0)} = \mathbf{w}_0 \in \Delta,\ \mathbf{u}^{(0)} = 0,\ L > 0,\ \lambda^{(0)} > 0,\ \beta = 1/2L$
repeat
  $\alpha^{(t+1)} \leftarrow 2/(t+1)$
  $\lambda^{(t+1)} \leftarrow (\alpha^{(t+1)}/4 + 1)\beta$
  $\mathbf{w}_{md}^{(t+1)} \leftarrow (1 - \alpha^{(t+1)})\mathbf{w}_{ag}^{(t)} + \alpha^{(t+1)}\mathbf{w}^{(t)}$ ▷ Extrapolation
  $\mathbf{v} \leftarrow 2(\mathbf{A} - R(\mathbf{w}_{md}^{(t+1)})\mathbf{B})\mathbf{w}_{md}^{(t+1)}/(\mathbf{w}_{md}^{(t+1)\intercal}\mathbf{B}\mathbf{w}_{md}^{(t+1)})$ ▷ Descent direction
  $\mathbf{w}^{(t+1)} \leftarrow P_\Delta(\mathbf{w}^{(t)} - \lambda^{(t+1)}\mathbf{v})$ ▷ Projection
  $\mathbf{w}_{ag}^{(t+1)} \leftarrow P_\Delta(\mathbf{w}_{md}^{(t+1)} - \beta\mathbf{v})$ ▷ Projection
until *Objective value converges*

A.4 Coordinate Descent Method (CD)

Algorithm 4: Coordinate Descent Method (CD)

Initialize $\mathbf{w}^{(0)}$
repeat
  for $i = 1, \ldots, n$ **do** ▷ Coordinate descent
    $r_1, r_2 \leftarrow$ real roots of a quadratic with coefficients(2.12)
    $w_i^{(t+1)} \leftarrow \text{argmin}_{p \in \Delta^*} R(w_1^{(t+1)}, \ldots, w_{i-1}^{(t+1)}, p, w_{i+1}^{(t)}, \ldots, w_n^{(t)})$
  end
  $\mathbf{w}^{(t+1)} \leftarrow P_\Delta(\mathbf{w}^{(t+1)})$ ▷ Projection
until *objective value converges*

A.5 Alternating Direction Method of Multipliers (ADMM)

Algorithm 5: Alternating Direction Method of Multipliers (ADMM)

Initialize $\mathbf{w}^{(0)} = \boldsymbol{z}^{(0)} = \mathbf{w}_0 \in \Delta,\ \mathbf{u}^{(0)} = 0,\ 0 < \eta < 1,\ \rho > 0$
repeat ▷ APGE for $\mathbf{w}$ update
  $s \leftarrow 1$
  $\alpha^{(0)} \leftarrow 0$
  $\alpha^{(1)} \leftarrow 1$
  $\mathbf{p}^{(0)} = \mathbf{p}^{(1)} \leftarrow \mathbf{w}^{(t)}$
  repeat
    $\mathbf{q} \leftarrow \mathbf{p}^{(k)} + (\frac{\alpha^{(k-1)}-1}{\alpha^{(k)}})(\mathbf{p}^{(k)} - \mathbf{p}^{(k-1)})$ ▷ Extrapolation
    $\mathbf{d} \leftarrow \frac{2}{\mathbf{q}^{(k)\intercal}\mathbf{B}\mathbf{q}^{(k)}} \left(\mathbf{A} - R(\mathbf{q}^{(k)})\mathbf{B}\right)\mathbf{q}^{(k)} + (1/\rho)\left(\mathbf{q}^{(k)} - (\mathbf{z}^{(t)} - \mathbf{u}^{(t)})\right)$
    repeat
      $\mathbf{p}_{\text{temp}} \leftarrow \mathbf{q} - s\mathbf{d}$
      $F_{\text{old}} \leftarrow R(\mathbf{q}) + (1/2\rho)\|\mathbf{q} - (\mathbf{z}^{(t)} - \mathbf{u}^{(t)})\|_2^2 + \langle \mathbf{q} - \mathbf{p}_{\text{temp}}, \mathbf{d}\rangle + (1/2s)\|\mathbf{q} - \mathbf{p}_{\text{temp}}\|_2^2$
      $F_{\text{new}} \leftarrow R(\mathbf{p}_{\text{temp}}) + (1/2\rho)\|\mathbf{p}_{\text{temp}} - (\mathbf{z}^{(t)} - \mathbf{u}^{(t)})\|_2^2$
      $s \leftarrow s\eta$ ▷ Backtracking line search
    until $F_{new} \leq F_{old}$
    $\mathbf{p}^{(k+1)} \leftarrow \mathbf{p}_{\text{temp}}$
    $\alpha^{(k+1)} \leftarrow (1 + \sqrt{1 + (2\alpha^{(k)})^2})/2$
  until *Objective value converges*
  $\mathbf{w}^{(t+1)} \leftarrow \mathbf{p}^{(k+1)}$ ▷ Update $\mathbf{w}$
  $\mathbf{z}^{(t+1)} \leftarrow P_{\Delta}(\mathbf{w}^{(t+1)} + \mathbf{u}^{(t)})$ ▷ Update $\mathbf{z}$ with projection
  $\mathbf{u}^{(t+1)} \leftarrow \mathbf{u}^{(t)} + \mathbf{w}^{(t+1)} - \mathbf{z}^{(t+1)}$ ▷ Update $\mathbf{u}$
until *Objective value converges*

A.6 (Michelot) Projection to Δ_{nonneg}

Algorithm 6: (Michelot) Projection to $\Delta_{nonneg} = \{w_1 + \cdots + w_n = 1,\ w_1, \ldots, w_n \geq 0\}$

repeat
  Project $\mathbf{w}$ onto the hyperplane $H = \{\mathbf{y} : \sum_i y_i = 1\}$: $\mathbf{w} \leftarrow \mathbf{w} - \frac{\sum_i w_i - 1}{n}\mathbf{1}_n$
  For $i = 1, \ldots, n$, if some $w_i < 0$, then set $w_i = 0$ and eliminate w_i from further consideration
until $w_i \geq 0$ *for all* i

A.7 Projection to Δ_{sparse}

Algorithm 7: Projection to $\Delta_{sparse} = \{\mathbf{w} \in \mathbb{R}^n : |w_1| + \cdots + |w_n| \leq \gamma,\ \sqrt{w_1^2 + \cdots + w_n^2} = \lambda\}$

if $\gamma \leq \lambda$ **then**
  $\mathbf{w} \leftarrow P_{L_1}(\mathbf{w})$ ▷ Use Algorithm 9 for $P_{\Delta_{L_1}}$
else if $\gamma \geq \sqrt{2}\lambda$ **then**
  $\mathbf{w} \leftarrow P_{L_2}(\mathbf{w})$ ▷ Use Algorithm 10 for $P_{\Delta_{L_2}}$
else
  $\mathbf{w} \leftarrow P_{L_1 \cap L_2}(\mathbf{w})$ ▷ Use Algorithm 11 for $P_{\Delta_{L_1 \cap L_2}}$
end

A.8 Projection to $\Delta_{simplex}$

Algorithm 8: Projection to $\Delta_{simplex} = \{\mathbf{w} \in \mathbb{R}^n : \sum w_i = \gamma,\ w_i \geq 0,\ i = 1, \ldots, n\}$

$n \leftarrow length(\mathbf{w})$
$\mathbf{z} \leftarrow \text{sort}(\mathbf{w})$
$\Sigma_z \leftarrow 0$
$\delta \leftarrow 0$
for $i = 1 : n$ **do**
    $\Sigma_z \leftarrow \Sigma_z + z_i$
    $\delta \leftarrow (\Sigma_z - \gamma)/i$
    if $i < n$ *and* $\delta < z_i$ *and* $\delta \geq z_{i+1}$ **then**
        break
    end
    $\mathbf{w} \leftarrow (\mathbf{w} - \delta)_+$
end

A.9 Projection to Δ_{L_1}

Algorithm 9: Projection to $\Delta_{L_1} = \{\mathbf{w} \in \mathbb{R}^n : \|\mathbf{w} - \mathbf{c}\|_1 \leq \gamma, \mathbf{c} \in \mathbb{R}^n :$ constant vector$\}$

$\mathbf{w}_{temp} \leftarrow \mathbf{w} - \mathbf{c}$
$\text{sgn} \leftarrow sign(\mathbf{w})$
if $\|\mathbf{w}_{temp}\|_1 \leq \gamma$ **then**
    $\mathbf{w} \leftarrow \mathbf{w}$
else
    $\mathbf{w}_{temp} \leftarrow P_{\Delta_{simplex}}(\mathbf{w}_{temp})$ ▷ `use Algorith 8 for` $P_{\Delta_{simplex}}$
    $\mathbf{w} \leftarrow \text{sgn} \cdot \mathbf{w}_{temp} + \mathbf{c}$
end

A.10 Projection to Δ_{L_2}

Algorithm 10: Projection to $\Delta_{L_2} = \{\mathbf{w} \in \mathbb{R}^n : \|\mathbf{w} - c\|_2 \leq \lambda, \mathbf{c} \in \mathbb{R}^n : \text{constant vector}\}$

if $\|\mathbf{w} - \mathbf{c}\|_2 \leq \lambda$ **then**
  $\mathbf{w} \leftarrow \mathbf{w}$
end
else
  $\mathbf{w} \leftarrow \mathbf{c} + \lambda(\mathbf{w} - \mathbf{c})/(\|\mathbf{w} - \mathbf{c}\|_2)$
end

A.11 (Dykstra) Projection to the intersection of r closed convex sets $C_0, \ldots, C_{r-1}$

Algorithm 11: (Dykstra) Projection to the intersection of r closed convex sets $C_0, \ldots, C_{r-1}$

Initialize $\mathbf{w}^{(-1)} = \mathbf{w}_0$, $\mathbf{e}^{(-r)} = \ldots = \mathbf{e}^{(-1)} = \mathbf{0}$
repeat
  $\mathbf{w}^{(t)} \leftarrow P_{C_{t \bmod r}}(\mathbf{w}^{(t-1)} + \mathbf{e}^{(t-r)})$ ▷ t `mod` r `: remainder of` t/r
  $\mathbf{e}^{(t)} \leftarrow \mathbf{w}^{(t-1)} + \mathbf{e}^{(t-r)} - \mathbf{w}^{(t)}$
until *Objective value converges*

Appendix B

Proofs and Algorithms for Tensor Canonical Correlation Analysis

B.1 Proof for Lemma 1

Proof. For the first equality, let $\mathbf{u} \in \mathbb{R}^m$ and $\mathbf{v} \in \mathbb{R}^n$. Then

$$\begin{aligned}
\operatorname{vec}(\mathbf{u}\mathbf{v}^\intercal) &= \operatorname{vec}\left(\begin{bmatrix} u_1v_1 & \cdots & u_1v_n \\ \vdots & \ddots & \vdots \\ u_mv_1 & \cdots & u_mv_n \end{bmatrix}\right) \\
&= \operatorname{vec}\left(\begin{bmatrix} v_1\mathbf{u} & \cdots & v_n\mathbf{u} \end{bmatrix}\right) \\
&= \begin{bmatrix} v_1\mathbf{u} \\ \vdots \\ v_n\mathbf{u} \end{bmatrix} \\
&= \mathbf{v} \otimes \mathbf{u}
\end{aligned}$$

For the second, third, and fourth equality, see Laub (2005, Theorem 13.26, 13.4, 13.3). For the last equality, let $\mathbf{A}$ has m rows and $\mathbf{C}, \mathbf{D}$ have p columns. Then

$$
\begin{aligned}
(\mathbf{A} \otimes \mathbf{B})(\mathbf{C} * \mathbf{D}) &= \begin{bmatrix} \mathbf{A}_{1\cdot} \otimes \mathbf{B} \\ \vdots \\ \mathbf{A}_{m\cdot} \otimes \mathbf{B} \end{bmatrix} \begin{bmatrix} \mathbf{C}_{\cdot 1} \otimes \mathbf{D}_{\cdot 1} & \cdots & \mathbf{C}_{\cdot p} \otimes \mathbf{D}_{\cdot p} \end{bmatrix} \\
&= \begin{bmatrix} (\mathbf{A}_{1\cdot} \otimes \mathbf{B})(\mathbf{C}_{\cdot 1} \otimes \mathbf{D}_{\cdot 1}) & \cdots & (\mathbf{A}_{1\cdot} \otimes \mathbf{B})(\mathbf{C}_{\cdot p} \otimes \mathbf{D}_{\cdot p}) \\ \vdots & \ddots & \vdots \\ (\mathbf{A}_{m\cdot} \otimes \mathbf{B})(\mathbf{C}_{\cdot 1} \otimes \mathbf{D}_{\cdot 1}) & \cdots & (\mathbf{A}_{m\cdot} \otimes \mathbf{B})(\mathbf{C}_{\cdot p} \otimes \mathbf{D}_{\cdot p}) \end{bmatrix} \\
&= \begin{bmatrix} (\mathbf{A}_{1\cdot}\mathbf{C}_{\cdot 1}) \otimes (\mathbf{B}\mathbf{D}_{\cdot 1}) & \cdots & (\mathbf{A}_{1\cdot}\mathbf{C}_{\cdot p}) \otimes (\mathbf{B}\mathbf{D}_{\cdot p}) \\ \vdots & \ddots & \vdots \\ (\mathbf{A}_{m\cdot}\mathbf{C}_{\cdot 1}) \otimes (\mathbf{B}\mathbf{D}_{\cdot 1}) & \cdots & (\mathbf{A}_{m\cdot}\mathbf{C}_{\cdot p}) \otimes (\mathbf{B}\mathbf{D}_{\cdot p}) \end{bmatrix} \\
&= \begin{bmatrix} \mathbf{A}\mathbf{C}_{\cdot 1} \otimes \mathbf{B}\mathbf{D}_{\cdot 1} & \cdots & \mathbf{A}\mathbf{C}_{\cdot p} \otimes \mathbf{B}\mathbf{D}_{\cdot p} \end{bmatrix} \\
&= \begin{bmatrix} (\mathbf{A}\mathbf{C})_{\cdot 1} \otimes (\mathbf{B}\mathbf{D})_{\cdot 1} & \cdots & (\mathbf{A}\mathbf{C})_{\cdot p} \otimes (\mathbf{B}\mathbf{D})_{\cdot p} \end{bmatrix} \\
&= \mathbf{A}\mathbf{C} * \mathbf{B}\mathbf{D}.
\end{aligned}
$$

□

B.2 Proof for Proposition 1

Proof. Consider derivation of $\mathrm{Cov}(\langle \mathcal{X}, \mathcal{A} \rangle, \langle \mathcal{Y}, \mathcal{B} \rangle)$.

$$
\begin{aligned}
& \mathrm{Cov}(\langle \mathcal{X}, \mathcal{A} \rangle, \langle \mathcal{Y}, \mathcal{B} \rangle) \\
= \; & \mathrm{Cov}(\langle \mathbf{X}_{(m)}, \mathbf{A}^{(m)} (\mathbf{A}^{(d_x)} * \cdots * \mathbf{A}^{(m+1)} * \mathbf{A}^{(m-1)} * \cdots * \mathbf{A}^{(1)}) \rangle, \\
& \qquad \langle \mathbf{Y}_{(n)}, \mathbf{B}^{(n)} (\mathbf{B}^{(d_y)} * \cdots * \mathbf{B}^{(n+1)} * \mathbf{B}^{(n-1)} * \cdots * \mathbf{B}^{(1)}) \rangle) \\
= \; & \mathrm{Cov}(\langle \mathbf{X}_{(m)} (\mathbf{A}^{(d_x)} * \cdots * \mathbf{A}^{(m+1)} * \mathbf{A}^{(m-1)} * \cdots * \mathbf{A}^{(1)}), \mathbf{A}^{(m)} \rangle, \\
& \qquad \langle \mathbf{Y}_{(n)} (\mathbf{B}^{(d_y)} * \cdots * \mathbf{B}^{(n+1)} * \mathbf{B}^{(n-1)} * \cdots * \mathbf{B}^{(1)}), \mathbf{B}^{(n)} \rangle) \\
= \; & \mathrm{Cov}(\langle \mathbf{X}_{(m)} \mathbf{A}^{(-m)}, \mathbf{A}^{(m)} \rangle, \langle \mathbf{Y}_{(n)} \mathbf{B}^{(-n)}, \mathbf{B}^{(n)} \rangle) \\
= \; & \mathrm{Cov}(\langle \mathrm{vec}\,(\mathbf{X}_{(m)} \mathbf{A}^{(-m)}), \mathrm{vec}\,\mathbf{A}^{(m)} \rangle, \; \langle \mathrm{vec}\,(\mathbf{Y}_{(n)} \mathbf{B}^{(-n)}), \mathrm{vec}\,\mathbf{B}^{(n)} \rangle) \\
= \; & (\mathrm{vec}\,\mathbf{A}^{(m)})^{\intercal} \mathrm{Cov}(\mathrm{vec}\,(\mathbf{X}_{(m)} \mathbf{A}^{(-m)}), \; \mathrm{vec}\,(\mathbf{Y}_{(n)} \mathbf{B}^{(-n)})) \mathrm{vec}\,\mathbf{B}^{(n)} \\
= \; & (\mathrm{vec}\,\mathbf{A}^{(m)})^{\intercal} \mathrm{Cov}((\mathbf{A}^{(-m)} \otimes \mathbf{I}_{I_m}) \mathrm{vec}\,\mathbf{X}_{(m)}, \; (\mathbf{B}^{(-n)} \otimes \mathbf{I}_{J_n}) \mathrm{vec}\,\mathbf{Y}_{(n)}) \mathrm{vec}\,\mathbf{B}^{(n)} \\
= \; & (\mathrm{vec}\,\mathbf{A}^{(m)})^{\intercal} (\mathbf{A}^{(-m)\intercal} \otimes \mathbf{I}_{I_m}) \mathrm{Cov}(\mathrm{vec}\,\mathbf{X}_{(m)}, \; \mathrm{vec}\,\mathbf{Y}_{(n)}) (\mathbf{B}^{(-n)} \otimes \mathbf{I}_{J_n}) \mathrm{vec}\,\mathbf{B}^{(n)} \\
= \; & (\mathrm{vec}\,\mathbf{A}^{(m)})^{\intercal} (\mathbf{A}^{(-m)\intercal} \otimes \mathbf{I}_{I_m}) \Sigma_{\mathrm{vec}\,\mathbf{X}_{(m)} \mathrm{vec}\,\mathbf{Y}_{(n)}} (\mathbf{B}^{(-n)} \otimes \mathbf{I}_{J_n}) \mathrm{vec}\,\mathbf{B}^{(n)}
\end{aligned}
$$

Derivation of $\mathrm{Var}(\langle \mathcal{X}, \mathcal{A} \rangle)$ and $\mathrm{Var}(\langle \mathcal{Y}, \mathcal{B} \rangle)$ can be done in similar way. □

B.3 Proof for Proposition 2

Proof. Consider $\mathrm{Var}(\langle \mathcal{X}, \mathcal{A} \rangle)$,

$$
\begin{aligned}
& \mathrm{Var}(\langle \mathcal{X}, \mathcal{A} \rangle) \\
= \; & (\mathrm{vec}\, \mathcal{A})^{\intercal} \boldsymbol{\Sigma}_{\mathrm{vec}\, \mathcal{X}} (\mathrm{vec}\, \mathcal{A}) \\
= \; & \mathbf{1}_{R_x}^{\intercal} (\mathbf{A}^{(d_x)} * \cdots * \mathbf{A}^{(1)})^{\intercal} (\boldsymbol{\Sigma}_{\mathbf{X},d_x} \otimes \cdots \otimes \boldsymbol{\Sigma}_{\mathbf{X},1}) (\mathbf{A}^{(d_x)} * \cdots * \mathbf{A}^{(1)}) \mathbf{1}_{R_x} \\
= \; & \mathbf{1}_{R_x}^{\intercal} [(\mathbf{A}^{(d_x)\intercal} \boldsymbol{\Sigma}_{\mathbf{X},d_x} \mathbf{A}^{(d_x)}) \odot \cdots \odot (\mathbf{A}^{(1)\intercal} \boldsymbol{\Sigma}_{\mathbf{X},1} \mathbf{A}^{(1)})] \mathbf{1}_{R_x} \\
= \; & \mathbf{1}_{R_x}^{\intercal} [(\mathbf{A}^{(m)\intercal} \boldsymbol{\Sigma}_{\mathbf{X},m} \mathbf{A}^{(m)}) \odot \mathbf{R}_{\mathbf{X},-m}] \mathbf{1}_{R_x} \\
= \; & \mathrm{tr}((\mathbf{A}^{(m)\intercal} \boldsymbol{\Sigma}_{\mathbf{X},m} \mathbf{A}^{(m)}) \mathbf{R}_{\mathbf{X},-m}) \\
= \; & \langle \boldsymbol{\Sigma}_{\mathbf{X},m} \mathbf{A}^{(m)}, \mathbf{A}^{(m)} \mathbf{R}_{\mathbf{X},-m} \rangle \\
= \; & \langle \mathrm{vec}\, (\boldsymbol{\Sigma}_{\mathbf{X},m} \mathbf{A}^{(m)}), \mathrm{vec}\, (\mathbf{A}^{(m)} \mathbf{R}_{\mathbf{X},-m}) \rangle \\
= \; & (\mathrm{vec}\, \mathbf{A}^{(m)})^{\intercal} (\mathbf{I}_{R_x} \otimes \boldsymbol{\Sigma}_{\mathbf{X},m}) (\mathbf{R}_{\mathbf{X},-m} \otimes \mathbf{I}_{I_m}) (\mathrm{vec}\, \mathbf{A}^{(m)}) \\
= \; & (\mathrm{vec}\, \mathbf{A}^{(m)})^{\intercal} (\mathbf{R}_{\mathbf{X},-m} \otimes \boldsymbol{\Sigma}_{\mathbf{X},m}) (\mathrm{vec}\, \mathbf{A}^{(m)})
\end{aligned}
$$

for any $m \in 1, \ldots, d_x$, where

$$
\begin{aligned}
& \mathbf{R}_{\mathbf{X},-m} \\
= \; & (\mathbf{A}^{(d_x)\intercal} \boldsymbol{\Sigma}_{\mathbf{X},d_x} \mathbf{A}^{(d_x)}) \odot \cdots \odot (\mathbf{A}^{(m+1)\intercal} \boldsymbol{\Sigma}_{\mathbf{X},m+1} \mathbf{A}^{(m+1)}) \\
& \qquad \odot (\mathbf{A}^{(m-1)\intercal} \boldsymbol{\Sigma}_{\mathbf{X},m-1} \mathbf{A}^{(m-1)}) \odot \cdots \odot (\mathbf{A}^{(1)\intercal} \boldsymbol{\Sigma}_{\mathbf{X},1} \mathbf{A}^{(1)}) \\
= \; & \odot_{m' \neq m} (\mathbf{A}^{(m')\intercal} \boldsymbol{\Sigma}_{\mathbf{X},m'} \mathbf{A}^{(m')})
\end{aligned}
$$

Similarly, it can be shown that

$$\begin{aligned} & \mathrm{Var}(\langle \mathcal{Y}, \mathcal{B} \rangle) \\ = \; & (\mathrm{vec}\, \mathbf{B}^{(n)})^\intercal (\mathbf{R}_{\mathbf{Y},-n} \otimes \mathbf{\Sigma}_{\mathbf{Y},n})(\mathrm{vec}\, \mathbf{B}^{(n)}) \end{aligned}$$

for any $n \in 1, \ldots, d_y$, where $\mathbf{R}_{\mathbf{Y},-n} = \odot_{n' \neq n}(\mathbf{B}^{(n')\intercal} \mathbf{\Sigma}_{\mathbf{B},n'} \mathbf{B}^{(n')})$ □

B.4 Proof for Lemma 3

Proof. For the first identity, see Hoff et al. (2011, Proposition 2.1).

For the second identity,

$$\begin{aligned} & E(\|\mathrm{vec}\, \mathcal{X}\|_2^2) \\ = \; & \mathrm{tr}(\mathrm{Var}(\mathrm{vec}\, \mathcal{X})) \\ = \; & \mathrm{tr}(\mathbf{\Sigma}_D \otimes \cdots \otimes \mathbf{\Sigma}_1) \\ = \; & \prod_d \mathrm{tr}(\mathbf{\Sigma}_d) \end{aligned}$$

□

B.5 Estimation Algorithm of TCCA

Algorithm 12: Parameter estimation of TCCA model

Initialize $\mathbf{A}^{(1)}, \ldots, \mathbf{A}^{(d_x)}, \mathbf{B}^{(1)}, \ldots, \mathbf{B}^{(d_y)}$

repeat

$m \leftarrow ((t-1) \mod dx) + 1$

$n \leftarrow ((t-1) \mod dy) + 1$

$\mathbf{A}_{(t)}^{(-m)} \leftarrow \mathbf{A}^{(d_x)} * \cdots * \mathbf{A}^{(m+1)} * \mathbf{A}^{(m-1)} * \cdots * \mathbf{A}^{(1)}$

$\mathbf{B}_{(t)}^{(-n)} \leftarrow \mathbf{B}^{(d_y)} * \cdots * \mathbf{B}^{(n+1)} * \mathbf{B}^{(n-1)} * \cdots * \mathbf{B}^{(1)}$

$\mathbf{C}_{(t)}^{\mathbf{XY}} \leftarrow (\mathbf{A}^{(-m)\top} \otimes \mathbf{I}_{I_m}) \hat{\mathbf{\Sigma}}_{\operatorname{vec} \mathbf{X}_{(m)}, \operatorname{vec} \mathbf{Y}_{(n)}} (\mathbf{B}^{(-n)\top} \otimes \mathbf{I}_{J_n})$

$\mathbf{C}_{(t)}^{\mathbf{X}} \leftarrow (\mathbf{A}^{(-m)\top} \otimes \mathbf{I}_{I_m}) \hat{\mathbf{\Sigma}}_{\operatorname{vec} \mathbf{X}_{(m)}} (\mathbf{A}^{(-m)} \otimes \mathbf{I}_{I_m})$

$\mathbf{C}_{(t)}^{\mathbf{Y}} \leftarrow (\mathbf{B}^{(-n)\top} \otimes \mathbf{I}_{J_n}) \hat{\mathbf{\Sigma}}_{\operatorname{vec} \mathbf{Y}_{(n)}} (\mathbf{B}^{(-n)} \otimes \mathbf{I}_{J_n})$

Solve following generalized eigenvalue decomposition,

$$\begin{bmatrix} \mathbf{0} & \mathbf{C}_{(t)}^{\mathbf{XY}} \\ \mathbf{C}_{(t)}^{\mathbf{XY}\top} & \mathbf{0} \end{bmatrix} \begin{bmatrix} \operatorname{vec} \mathbf{A}_{(t+1)}^{(m)} \\ \operatorname{vec} \mathbf{B}_{(t+1)}^{(n)} \end{bmatrix} = \lambda_{(t)} \begin{bmatrix} \mathbf{C}_{(t)}^{\mathbf{X}} & \mathbf{0} \\ \mathbf{0} & \mathbf{C}_{(t)}^{\mathbf{Y}} \end{bmatrix} \begin{bmatrix} \operatorname{vec} \mathbf{A}_{(t+1)}^{(m)} \\ \operatorname{vec} \mathbf{B}_{(t+1)}^{(n)} \end{bmatrix}$$

until *objective value converges*

B.6 Estimation Algorithm of TSCCA

Algorithm 13: Parameter estimation of TSCCA model

Initialize $\mathbf{A}^{(1)}, \ldots, \mathbf{A}^{(d_x)}, \mathbf{B}^{(1)}, \ldots, \mathbf{B}^{(d_y)}$

repeat

  $m \leftarrow ((t-1) \mod dx) + 1$

  $n \leftarrow ((t-1) \mod dy) + 1$

  $\mathbf{A}_{(t)}^{(-m)} \leftarrow \mathbf{A}^{(d_x)} * \cdots * \mathbf{A}^{(m+1)} * \mathbf{A}^{(m-1)} * \cdots * \mathbf{A}^{(1)}$

  $\mathbf{B}_{(t)}^{(-n)} \leftarrow \mathbf{B}^{(d_y)} * \cdots * \mathbf{B}^{(n+1)} * \mathbf{B}^{(n-1)} * \cdots * \mathbf{B}^{(1)}$

  $\mathbf{C}_{(t)}^{\mathbf{XY}} \leftarrow (\mathbf{A}^{(-m)\top} \otimes \mathbf{I}_{I_m}) \hat{\mathbf{\Sigma}}_{\operatorname{vec} \mathbf{X}_{(m)}, \operatorname{vec} \mathbf{Y}_{(n)}} (\mathbf{B}^{(-n)\top} \otimes \mathbf{I}_{J_n})$

  $\mathbf{C}_{(t)}^{\mathbf{X}} \leftarrow (\mathbf{A}^{(-m)\top} \otimes \mathbf{I}_{I_m}) \hat{\mathbf{\Sigma}}_{\operatorname{vec} \mathbf{X}_{(m)}} (\mathbf{A}^{(-m)} \otimes \mathbf{I}_{I_m})$

  $\mathbf{C}_{(t)}^{\mathbf{Y}} \leftarrow (\mathbf{B}^{(-n)\top} \otimes \mathbf{I}_{J_n}) \hat{\mathbf{\Sigma}}_{\operatorname{vec} \mathbf{Y}_{(n)}} (\mathbf{B}^{(-n)} \otimes \mathbf{I}_{J_n})$

  $\mathbf{V}_{(t)} \leftarrow \begin{bmatrix} \mathbf{0} & \mathbf{C}_{(t)}^{\mathbf{XY}} \\ \mathbf{C}_{(t)}^{\mathbf{XY}\top} & \mathbf{0} \end{bmatrix}$

  $\mathbf{W}_{(t)} \leftarrow \begin{bmatrix} \mathbf{C}_{(t)}^{\mathbf{X}} & \mathbf{0} \\ \mathbf{0} & \mathbf{C}_{(t)}^{\mathbf{Y}} \end{bmatrix}$

  Use CGE algorithm with two matrices input $\mathbf{V}_{(t)}$ and $\mathbf{W}_{(t)}$

  Divide solutions from previous step to $\begin{bmatrix} \operatorname{vec} \mathbf{A}_{(t+1)}^{(m)} \\ \operatorname{vec} \mathbf{B}_{(t+1)}^{(n)} \end{bmatrix}$ with compatible dimensions each

until *objective value converges*

B.7 Estimation Algorithm of TCCA_SEP

Algorithm 14: Parameter estimation of TCCA_SEP model

Initialize $\mathbf{A}^{(1)}, \ldots, \mathbf{A}^{(d_x)}, \mathbf{B}^{(1)}, \ldots, \mathbf{B}^{(d_y)}$

$\mathbf{R}_{\mathbf{X},(0)} \leftarrow \odot_m (\mathbf{A}^{(m)\intercal} \hat{\mathbf{\Sigma}}_{\mathbf{X},m} \mathbf{A}^{(m)})$

$\mathbf{R}_{\mathbf{Y},(0)} \leftarrow \odot_n (\mathbf{B}^{(n)\intercal} \hat{\mathbf{\Sigma}}_{\mathbf{Y},n} \mathbf{B}^{(n)})$

repeat

$m \leftarrow ((t-1) \mod dx) + 1$

$n \leftarrow ((t-1) \mod dy) + 1$

$\mathbf{R}_{\mathbf{X},-m,(t)} \leftarrow \mathbf{R}_{\mathbf{X},(t)} \oslash (\mathbf{A}_{(t)}^{(m)\intercal} \hat{\mathbf{\Sigma}}_{\mathbf{X},m} \mathbf{A}_{(t)}^{(m)})$

$\mathbf{R}_{\mathbf{Y},-n,(t)} \leftarrow \mathbf{R}_{\mathbf{Y},(t)} \oslash (\mathbf{B}_{(t)}^{(n)\intercal} \hat{\mathbf{\Sigma}}_{\mathbf{Y},n} \mathbf{B}_{(t)}^{(n)})$

$\mathbf{C}_{(t)}^{\mathbf{X}} \leftarrow \mathbf{R}_{\mathbf{X},-m,(t)} \otimes \hat{\mathbf{\Sigma}}_{\mathbf{X},m}$

$\mathbf{C}_{(t)}^{\mathbf{Y}} \leftarrow \mathbf{R}_{\mathbf{Y},-n,(t)} \otimes \hat{\mathbf{\Sigma}}_{\mathbf{Y},n}$

$\mathbf{A}_{(t)}^{(-m)} \leftarrow \mathbf{A}_{(t)}^{(d_x)} * \cdots * \mathbf{A}_{(t)}^{(m+1)} * \mathbf{A}_{(t)}^{(m-1)} * \cdots * \mathbf{A}_{(t)}^{(1)}$

$\mathbf{B}_{(t)}^{(-n)} \leftarrow \mathbf{B}_{(t)}^{(d_y)} * \cdots * \mathbf{B}_{(t)}^{(n+1)} * \mathbf{B}_{(t)}^{(n-1)} * \cdots * \mathbf{B}_{(t)}^{(1)}$

$\mathbf{C}_{(t)}^{\mathbf{XY}} \leftarrow (\mathbf{A}_{(t)}^{(-m)\intercal} \otimes \mathbf{I}_{I_m}) \hat{\mathbf{\Sigma}}_{\operatorname{vec} \mathbf{X}_{(m)}, \operatorname{vec} \mathbf{Y}_{(n)}} (\mathbf{B}_{(t)}^{(-n)\intercal} \otimes \mathbf{I}_{J_n})$

Solve following generalized eigenvalue decomposition,

$$\begin{bmatrix} \mathbf{0} & \mathbf{C}_{(t)}^{\mathbf{XY}} \\ \mathbf{C}_{(t)}^{\mathbf{XY}\intercal} & \mathbf{0} \end{bmatrix} \begin{bmatrix} \operatorname{vec} \mathbf{A}_{(t+1)}^{(m)} \\ \operatorname{vec} \mathbf{B}_{(t+1)}^{(n)} \end{bmatrix} = \lambda_{(t)} \begin{bmatrix} \mathbf{C}_{(t)}^{\mathbf{X}} & \mathbf{0} \\ \mathbf{0} & \mathbf{C}_{(t)}^{\mathbf{Y}} \end{bmatrix} \begin{bmatrix} \operatorname{vec} \mathbf{A}_{(t+1)}^{(m)} \\ \operatorname{vec} \mathbf{B}_{(t+1)}^{(n)} \end{bmatrix}$$

$\mathbf{R}_{\mathbf{X},(t+1)} \leftarrow \mathbf{R}_{\mathbf{X},-m,(t)} \odot ({\mathbf{A}_{(t+1)}^{(m)}}^{\intercal} \hat{\mathbf{\Sigma}}_{\mathbf{X},m} \mathbf{A}_{(t+1)}^{(m)})$

$\mathbf{R}_{\mathbf{Y},(t+1)} \leftarrow \mathbf{R}_{\mathbf{Y},-n,(t)} \odot ({\mathbf{B}_{(t+1)}^{(n)}}^{\intercal} \hat{\mathbf{\Sigma}}_{\mathbf{Y},n} \mathbf{B}_{(t+1)}^{(n)})$

until *objective value converges*

B.8 Estimation Algorithm of TSCCA_SEP

Algorithm 15: Parameter estimation of TSCCA_SEP model

Initialize $\mathbf{A}^{(1)}, \ldots, \mathbf{A}^{(d_x)}, \mathbf{B}^{(1)}, \ldots, \mathbf{B}^{(d_y)}$

$\mathbf{R}_{\mathbf{X},(0)} \leftarrow \odot_m(\mathbf{A}^{(m)\intercal}\hat{\mathbf{\Sigma}}_{\mathbf{X},m}\mathbf{A}^{(m)})$

$\mathbf{R}_{\mathbf{Y},(0)} \leftarrow \odot_n(\mathbf{B}^{(n)\intercal}\hat{\mathbf{\Sigma}}_{\mathbf{Y},n}\mathbf{B}^{(n)})$

repeat

  $m \leftarrow ((t-1) \mod dx) + 1$

  $n \leftarrow ((t-1) \mod dy) + 1$

  $\mathbf{R}_{\mathbf{X},-m,(t)} \leftarrow \mathbf{R}_{\mathbf{X},(t)} \oslash (\mathbf{A}_{(t)}^{(m)\intercal}\hat{\mathbf{\Sigma}}_{\mathbf{X},m}\mathbf{A}_{(t)}^{(m)})$

  $\mathbf{R}_{\mathbf{Y},-n,(t)} \leftarrow \mathbf{R}_{\mathbf{Y},(t)} \oslash (\mathbf{B}_{(t)}^{(n)\intercal}\hat{\mathbf{\Sigma}}_{\mathbf{Y},n}\mathbf{B}_{(t)}^{(n)})$

  $\mathbf{C}_{(t)}^{\mathbf{X}} \leftarrow \mathbf{R}_{\mathbf{X},-m,(t)} \otimes \hat{\mathbf{\Sigma}}_{\mathbf{X},m}$

  $\mathbf{C}_{(t)}^{\mathbf{Y}} \leftarrow \mathbf{R}_{\mathbf{Y},-n,(t)} \otimes \hat{\mathbf{\Sigma}}_{\mathbf{Y},n}$

  $\mathbf{A}_{(t)}^{(-m)} \leftarrow \mathbf{A}_{(t)}^{(d_x)} * \cdots * \mathbf{A}_{(t)}^{(m+1)} * \mathbf{A}_{(t)}^{(m-1)} * \cdots * \mathbf{A}_{(t)}^{(1)}$

  $\mathbf{B}_{(t)}^{(-n)} \leftarrow \mathbf{B}_{(t)}^{(d_y)} * \cdots * \mathbf{B}_{(t)}^{(n+1)} * \mathbf{B}_{(t)}^{(n-1)} * \cdots * \mathbf{B}_{(t)}^{(1)}$

  $\mathbf{C}_{(t)}^{\mathbf{XY}} \leftarrow (\mathbf{A}_{(t)}^{(-m)\intercal} \otimes \mathbf{I}_{I_m})\hat{\mathbf{\Sigma}}_{\mathrm{vec}\,\mathbf{X}_{(m)},\mathrm{vec}\,\mathbf{Y}_{(n)}}(\mathbf{B}_{(t)}^{(-n)\intercal} \otimes \mathbf{I}_{J_n})$

  $\mathbf{V}_{(t)} \leftarrow \begin{bmatrix} \mathbf{0} & \mathbf{C}_{(t)}^{\mathbf{XY}} \\ \mathbf{C}_{(t)}^{\mathbf{XY}\intercal} & \mathbf{0} \end{bmatrix}$

  $\mathbf{W}_{(t)} \leftarrow \begin{bmatrix} \mathbf{C}_{(t)}^{\mathbf{X}} & \mathbf{0} \\ \mathbf{0} & \mathbf{C}_{(t)}^{\mathbf{Y}} \end{bmatrix}$

  Use CGE algorithm with two matrices input $\mathbf{V}_{(t)}$ and $\mathbf{W}_{(t)}$

  Divide solutions from previous step to $\begin{bmatrix} \mathrm{vec}\,\mathbf{A}_{(t+1)}^{(m)} \\ \mathrm{vec}\,\mathbf{B}_{(t+1)}^{(n)} \end{bmatrix}$ with compatible dimensions each

  $\mathbf{R}_{\mathbf{X},(t+1)} \leftarrow \mathbf{R}_{\mathbf{X},-m,(t)} \odot (\mathbf{A}_{(t+1)}^{(m)\intercal}\hat{\mathbf{\Sigma}}_{\mathbf{X},m}\mathbf{A}_{(t+1)}^{(m)})$

  $\mathbf{R}_{\mathbf{Y},(t+1)} \leftarrow \mathbf{R}_{\mathbf{Y},-n,(t)} \odot (\mathbf{B}_{(t+1)}^{(n)\intercal}\hat{\mathbf{\Sigma}}_{\mathbf{Y},n}\mathbf{B}_{(t+1)}^{(n)})$

until *objective value converges*