ABSTRACT

LI, XIAOSHAN. Tensor Based Statistical Models with Applications in Neuroimaging Data Analysis. (Under the direction of Dr. Lexin Li and Dr. Hua Zhou.)

In the last few years there has been growing interest in neuroimaging data analysis. Large-scale neuroimaging studies have been collecting images of study individuals, which take the form of two-dimensional, three-dimensional, or higher dimensional arrays, also known as tensors. Addressing scientific questions arising from such data demands new statistical models that take multidimensional arrays as covariates, which further add complications to dimension reduction problems. In this dissertation, we study methods of statistical modeling when covariates are multidimensional arrays. There are three parts in this dissertation.

In the first part (Chapter 2), we propose a family of generalized linear tensor regression models based upon the Tucker decomposition of regression coefficient arrays. Effectively exploiting the low rank structure of tensor covariates brings the ultra-high dimensionality to a manageable level that leads to efficient estimation. We demonstrate the new model could both numerically provide a sound recovery of even high rank signals and asymptotically estimate the best Tucker structure approximation to the full array model. Simulation studies and real data analysis show that the proposed Tucker model outperforms a recently proposed CANDECOMP/PARAFAC (CP) tensor regression model.

In the second part of this dissertation (Chapter 3), we consider situations where individuals have multiple measurements from the same image modality. We develop tensor based linear and logistic regression model under measurement error model framework. We implement the estimation procedure based on conditional score equation, which is a

classical method to handle measurement error model.

In the last part (Chapter 4), we investigate classification problem for neuroimaging data. We focus on support vector machine (SVM) classifier when features come from multidimensional arrays. Efficient algorithm is developed to build separating hyperplane. Simulation studies and real data analysis show our proposed method outperforms vectorized feature based SVM or tensor logistic regression model for binary data.

Tensor Based Statistical Models with Applications in Neuroimaging Data Analysis

by
Xiaoshan Li

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

2014

APPROVED BY:

Dr. Arnab Maity	Dr. Ana-Maria Staicu
Dr. Lexin Li Co-chair of Advisory Committee	Dr. Hua Zhou Co-chair of Advisory Committee

DEDICATION

To my family.

To my Love, Wei Zhong.

BIOGRAPHY

The author was born in Jingzhou, Hubei, China in February 1988, and spent the first seventeen years of his life in the hometown. He graduated from Jingzhou High School in 2005 and attended Nankai University afterwards for his undergraduate study in Mathematics. After obtaining his Bachelor's degree in 2009, he enrolled in the North Carolina State University Depart of Statistics to pursue Ph.D. degree in statistics. His doctoral dissertation research at NCSU is under the direction of Dr. Lexin Li and Dr. Hua Zhou. He earned his Master's degree in statistics in 2011 and is expected to graduate with his Doctorate in Summer, 2014.

ACKNOWLEDGEMENTS

I would like to express my sincerest gratitude and appreciation to my advisors, Dr. Lexin Li and Dr. Hua Zhou, for their immense help in the entire journey of my research career at NCSU. Their insightful instruction, generous support and unending encouragement guided me throughout every step of my research. The experience of working with them has been great, and I have learned a lot from my advisors, not only on specific statistical knowledge, but the professional attitudes and activities in research work as well.

I thank Dr. Arnab Maity for the valuable guidance he provided at the early stage of my research, and for joining my thesis committee. His valuable advice helped expand my horizons in statistics research in the first two years of study at NCSU. I thank all the other members in my thesis committee, Dr. Ana-Maria Staicu and Dr. Kara Peters, for carefully reviewing this thesis and providing valuable suggestions.

I am grateful to all the faculty members in the department for offering a comprehensive collection of courses in statistics, which help lay solid foundations for my research. I would like to thank Dr. Charlie Smith for his warm greetings and snacks when I stay up late in the Lab. Also I would like to thank all the staff for their excellent service to the department, which creates great environment for study.

I also owe my gratitude to my mentors, Jeffery Painter and Alan Menius, at GlaxoSmithKline Pharmaceutical, where I worked as a Graduate Industrial Traineeship, and David Grinder at American Credit Acceptance, where I worked as an internship. Under their instruction, I got access to statistical methods used in real industrial applications.

Thanks to my fellow students and friends at NCSU. Your supports and encouragements have made my life easier and enriched my life in the United States.

Finally, I would like to express my deepest gratitude to my family. To my lovely wife,

Wei Zhong, although in long distance, your love and patience has been accompanying me every moment of my life. To my great parents and grandma, your unconditional support and love is the source of my strength.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

Chapter 1

Introduction

1.1 Overview of Brain Imaging Analysis

Advancing technologies are constantly producing large scale scientific data with complex structures. An important class arises from medical imaging, where the data takes the form of multidimensional array, also known as *tensor*. Notable examples include electroencephalography (EEG, 2D matrix), positron emission tomography (PET, 3D array), anatomical magnetic resonance images (MRI, 3D array), functional magnetic resonance images (fMRI, 4D array), among other image modalities. EEG is the recording of electrical activity along the scalp. It measures voltage fluctuations resulting from ionic current flows within the neurons of the brain (Niedermeyer and da Silva, 2005). EEG data have two dimensions, which consist of electrodes (channels) and time. PET is a functional imaging technique which produces a three-dimensional image of functional processes in the body (Valk, 2003). In neuroimaging application, PET measures indirectly the flow of blood to different parts of the brain by tracking the emissions from radioactively labeled metabolically active chemicals that have been injected into the bloodstream. MRI

of the brain uses magnetic fields and radio waves to create high quality images of brain structures. MRI scan is produced with the signals of hydrogen atoms realigned by the magnetic field and radio waves. MRI is used to visualize the brain, but it does not check the function of the brain. FMRI is a functional neuroimaging procedure using MRI technology that measures brain activity by detecting associated changes in blood flow (Huettel et al., 2004). It learns how a normal, diseased or injured brain is working when a subject alternates between periods of doing a particular task and a control state.

In medical imaging data analysis, a primary goal is to better understand associations between brains and clinical outcomes. Applications include using brain images to diagnose neurodegenerative disorders, to predict onset of neuropsychiatric diseases, and to identify disease relevant brain regions or activity patterns. This family of problems can collectively be formulated as statistical models with clinical outcome as response, and image, or tensor, as predictor. Either regression or classification analyses have been applied to patients with neurological or psychiatric disorders for diagnosis including Alzheimer's disease (AD), mild cognitive impairment (MCI), major depression (MD), and schizophrenia (SCP) (Davatzikos et al., 2005; Klöppel et al., 2008; Costafreda et al., 2009; Stonnington et al., 2010; Zhang et al., 2011; Zhang and Shen, 2012; Dukart et al., 2013; Zhou et al., 2013). Typical data sets employed for the analyses include the Attention Deficit Hyperactivity Disorder Sample Initiative (ADHD, 2013) and the the Alzheimer's Disease Neuroimaging Initiative (ADNI, 2013) database. ADHD consists of over 900 participants from eight imaging centers with both MRI and fMRI images, as well as their clinical information. ADNI accumulates over 3,000 participants with MRI, PET and fMRI to help measure the progression of MCI and early AD.

A common challenge for analyzing these data is the ultra-high dimensionality and complex structure of the image modalities. Most classical statistical models take vector

as covariate. Simply turning an image array into a long vector causes extremely high dimensionality. For instance, a typical MRI of size 128-by-128-by-128 results in a covariate vector of length $128^3 = 2097152$. Both computability and theoretical guarantee of the classical regression models are severely compromised by this ultra-high dimensionality. More seriously, vectorizing an array destroys the inherent spatial structure of the image array that usually possesses abundant information. To handle this challenge, voxel-based methods are usually visited in the literature. Voxel-based methods take the image data at each voxel as response and the clinical outcomes as well as some clinical variables such as age and gender as predictors, and then generate a statistical parametric map (SPM) of test statistics or p-values across all voxels (Friston et al., 1994). A major limitation of these methods is that they treat each voxel independently, since statistical model is performed at each individual voxel, and thus ignore the inherent spatial structure of array data.

Another typical method is the feature extraction, which extracts a vector of features from images, and then feeds the feature vector into a classical statistical model. The method to extract the feature vector can be subject to priori knowledge (Mckeown et al., 1998; Blankertz et al., 2001; Haxby et al., 2001; Kontos et al., 2003; Cox and Savoy, 2003; Mitchell et al., 2004; LaConte et al., 2005; Shinkareva et al., 2006). Alternatively one first applies unsupervised dimension reduction, often some variant of principal components analysis, to the image array, and then fits a classical statistical model in the reduced dimensional vector space (Martino et al., 2007; Caffo et al., 2010). Both solutions are intuitive and popular, and have enjoyed varying degrees of success. At heart, both transform the problem to a classical vector covariate statistical model. However, there is no consensus on what choice best summarizes a brain image even for a single modality, whereas unsupervised dimension reduction like principal components could re-

sult in information loss since the extracted principal components can be irrelevant to the response.

In this dissertation, we propose a new class of statistical models for array-valued covariate. Specifically, we adopt tensor decomposition to the coefficient array. Using this technique, we are able to fit substantially low dimensional models but utilize the information of the whole array covariate. In our work, tensor decomposition is embedded in a variety of statistical models including generalized linear model, measurement error model and support vector machine.

1.2 Tensor Decomposition Preliminaries

Multidimensional array, also called tensor, plays a central role in this dissertation. In this section we briefly summarize a few results for matrix/array operations. We use the terms multidimensional array and tensor interchangeably.

First we review two matrix products frequently used in this dissertation.

- *Kronecker product*

  Given two matrices $\boldsymbol{A} = [\boldsymbol{a}_1 \dots \boldsymbol{a}_n] \in \mathbb{R}^{m \times n}$ and $\boldsymbol{B} = [\boldsymbol{b}_1 \dots \boldsymbol{b}_q] \in \mathbb{R}^{p \times q}$, the *Kronecker product* is the mp-by-nq matrix

$$\boldsymbol{A} \otimes \boldsymbol{B} = \begin{bmatrix} a_{11}\boldsymbol{B} & a_{12}\boldsymbol{B} & \cdots & a_{1n}\boldsymbol{B} \\ a_{21}\boldsymbol{B} & a_{22}\boldsymbol{B} & \cdots & a_{2n}\boldsymbol{B} \\ \vdots & \ddots & \ddots & \vdots \\ a_{m1}\boldsymbol{B} & a_{m2}\boldsymbol{B} & \cdots & a_{mn}\boldsymbol{B} \end{bmatrix}.$$

- *Khatri-Rao product*

If $\boldsymbol{A}$ and $\boldsymbol{B}$ have the same number of columns $n = q$, then the *Khatri-Rao product* (Rao and Mitra, 1971) is defined as the mp-by-n columnwise Kronecker product $\boldsymbol{A} \odot \boldsymbol{B} = [\boldsymbol{a}_1 \otimes \boldsymbol{b}_1 \ \boldsymbol{a}_1 \otimes \boldsymbol{b}_2 \ \ldots \ \boldsymbol{a}_n \otimes \boldsymbol{b}_n]$. If $n = q = 1$, then $\boldsymbol{A} \odot \boldsymbol{B} = \boldsymbol{A} \otimes \boldsymbol{B}$.

We next review some important operators that transform a tensor into a vector/matrix. The *vec operator* stacks the entries of a D-dimensional tensor $\boldsymbol{B} \in \mathbb{R}^{p_1 \times \cdots \times p_D}$ into a column vector. Specifically, an entry $b_{i_1 \ldots i_D}$ maps to the j-th entry of $\text{vec}(\boldsymbol{B})$ where $j = 1 + \sum_{d=1}^{D} (i_d - 1) \prod_{d'=1}^{d-1} p_{d'}$. For instance, when $D = 2$, the matrix entry at cell (i_1, i_2) maps to position $j = 1 + i_1 - 1 + (i_2 - 1)p_1 = i_1 + (i_2 - 1)p_1$, which is consistent with the more familiar vec operator on a matrix. The *mode-d matricization*, $\boldsymbol{B}_{(d)}$, maps a tensor $\boldsymbol{B}$ into a $p_d \times \prod_{d' \neq d} p_{d'}$ matrix such that the $(i_1, \ldots, i_D)$ element of the array $\boldsymbol{B}$ maps to the (i_d, j) element of the matrix $\boldsymbol{B}_{(d)}$, where $j = 1 + \sum_{d' \neq d} (i_{d'} - 1) \prod_{d'' < d', d'' \neq d} p_{d''}$. When $D = 1$, we observe that $\text{vec}(\boldsymbol{B})$ is the same as vectorizing the mode-1 matricization $\boldsymbol{B}_{(1)}$. The *mode-(d, d') matricization* $\boldsymbol{B}_{(dd')} \in \mathbb{R}^{p_d p_{d'} \times \prod_{d'' \neq d, d'} p_{d''}}$ is defined in a similar fashion (Kolda, 2006). We also introduce an operator that turns vectors into an array. Specifically, an *outer product*, $\boldsymbol{b}_1 \circ \boldsymbol{b}_2 \circ \cdots \circ \boldsymbol{b}_D$, of D vectors $\boldsymbol{b}_d \in \mathbb{R}^{p_d}$, $d = 1, \ldots, D$, is a $p_1 \times \cdots \times p_D$ array with entries $(\boldsymbol{b}_1 \circ \boldsymbol{b}_2 \circ \cdots \circ \boldsymbol{b}_D)_{i_1 \cdots i_D} = \prod_{d=1}^{D} b_{d i_d}$ (Zhou et al., 2013).

We then introduce a concept that plays a key role in this dissertation. We say an array $\boldsymbol{B} \in \mathbb{R}^{p_1 \times \cdots \times p_D}$ admits a *rank-R CANDECOMP/PARAFAC (CP) decomposition* if

$$\boldsymbol{B} = \sum_{r=1}^{R} \boldsymbol{\beta}_1^{(r)} \circ \cdots \circ \boldsymbol{\beta}_D^{(r)}, \tag{1.1}$$

where $\boldsymbol{\beta}_d^{(r)} \in \mathbb{R}^{p_d}, d = 1, \ldots, D, r = 1, \ldots, R$, are all column vectors, and $\boldsymbol{B}$ cannot be written as a sum of less than R outer products. For convenience, the decomposition is often represented by a shorthand, $\boldsymbol{B} = [\![\boldsymbol{B}_1, \ldots, \boldsymbol{B}_D]\!]$, where $\boldsymbol{B}_d = [\boldsymbol{\beta}_d^{(1)}, \ldots, \boldsymbol{\beta}_d^{(R)}] \in$

$\mathbb{R}^{p_d \times R}$, $d = 1, \ldots, D$ (Kolda, 2006; Kolda and Bader, 2009). The use of CP decomposition is often accompanied with the following result to relate the mode-d matricization and the vec operator of an array to its rank-R decomposition.

Lemma 1. *If a tensor* $\boldsymbol{B} \in \mathbb{R}^{p_1 \times \cdots \times p_D}$ *admits a rank-R decomposition (1.1), then*

$$\boldsymbol{B}_{(d)} = \boldsymbol{B}_d(\boldsymbol{B}_D \odot \cdots \odot \boldsymbol{B}_{d+1} \odot \boldsymbol{B}_{d-1} \odot \cdots \odot \boldsymbol{B}_1)^\intercal \quad \textit{and} \quad \operatorname{vec} \boldsymbol{B} = (\boldsymbol{B}_D \odot \cdots \odot \boldsymbol{B}_1)\mathbf{1}_R.$$

Based on CP decomposition (1.1) and Lemma 1, Zhou et al. (2013) proposed a class of generalized linear tensor regression model. With a low rank CP decomposition, ultra-high dimensionality of generalized linear model (GLM) is reduced to a manageable level under the setup in Zhou et al. (2013).

1.3 Thesis Organization

In the last ten years, interest in tensor decompositions has expanded to many fields including signal processing, numerical linear algebra, computer vision and more. Extensive references can be found in the review paper (Kolda and Bader, 2009). In this dissertation, a comprehensive collection of works are presented that integrate tensor decomposition within different supervised statistical learning paradigms. The rest of this dissertation is divided into three chapters. In Chapter 2, we introduce Tucker tensor decomposition, or higher-order singular value decomposition (HOSVD) (Kolda and Bader, 2009) under GLM framework. Compared with CP tensor regression model (Zhou et al., 2013), the proposed Tucker tensor regression model enjoys two major advantages. On one hand, the Tucker tensor regression model shares the advantages of the CP tensor regression model, in that it effectively exploits the special structure of the tensor data, it substantially

reduces the dimensionality to enable efficient model estimation, and it provides a sound low rank approximation to a potentially high rank signal. On the other hand, Tucker tensor regression offers a much more flexible modeling framework than CP regression, as it allows distinct order along each dimension. When the orders are all identical, it includes the CP model as a special case. This flexibility leads to several improvements that are particularly useful for neuroimaging analysis. Both simulation studies and real data analysis demonstrate the flexibility of the proposed Tucker tensor regression model over CP tensor regression model.

In Chapter 3, we consider situations where measurement error is present for multiple measurements of tensor covariate. In many areas of statistical analysis, some covariates are not directly observable for some reason. Instead, they can only be measured indirectly and imprecisely, which results in data with measurement error. These situations are not rare in image data analysis, either. Although there are a variety of methods to solve measurement error models (Carroll et al., 2006), they treat covariate as vector and the dimensionality is low. In this chapter we review advantages and disadvantages of the methods and focus on implementing conditional score method (Stefanski and Carroll, 1987) in high dimensional tensor covariate setting. Specifically, we mainly focus on using CP tensor decomposition in the linear and logistic models with measurement error. The proposed method leads to reduced bias of estimates of the linear coefficients. Numerical studies demonstrate that the proposed method outperforms the CP tensor regression model (Zhou et al., 2013) as if there is no measurement error in terms of reducing bias of estimates.

In Chapter 4, we focus on classification problem based on neuroimaing data. In the last few years there has been growing interest in using machine learning classifiers for analyzing neuroimaging data. Typical problems include pattern recognition such as decoding

stimuli, mental states, or behaviors, and discriminating healthy individuals and patients with a wide range of neurological and psychiatric disorders. Support vector machine (SVM) is usually employed as the classifier in the literature for its stable performance in high dimensional data space (Vapnik, 1996; Pereira et al., 2009). To be consistent with the literature in neuroimage analysis, we use 'features' as independent variables or predictors in this chapter. Classical studies for neuroimaging data always turn an image into a long vector at first (Pereira et al., 2009), and then feature selection is used to extract the most discriminant features. Most feature selection methods in the literature are either voxel-based method or region of interest (ROI) based method. In voxel based method, voxels, or features are treated individually and ranked according to their discriminant power, and then top ranked voxels are selected for training classifier. In ROI based method, priori information is present to help identify the ROIs. Then features in the ROIs are selected for training classifier. In this chapter, we propose a tensor linear SVM method, which not only keeps the spatial structure of an image instead of turning it into a vector, but also uses all voxels to train classifier. Simulation studies and real data analysis are presented to show the usefulness of the proposed method.

Chapter 2

Tucker Tensor Regression and Neuroimaging Analysis

2.1 Introduction

In Chapter 1 we have introduced that many brain imaging applications can be formulated as a statistical model with image, or tensor, as predictor. In a recent work, Zhou et al. (2013) proposed a class of generalized linear tensor regression models with tensor predictor. Specifically, for a response variable Y, a vector predictor $\boldsymbol{Z} \in \mathbb{R}^{p_0}$ and a D-dimensional tensor predictor $\boldsymbol{X} \in \mathbb{R}^{p_1 \times \cdots \times p_D}$, the response is assumed to belong to an exponential family where the linear systematic part is of the form,

$$g(\mu) = \boldsymbol{\gamma}^{\mathsf{T}} \boldsymbol{Z} + \langle \boldsymbol{B}, \boldsymbol{X} \rangle. \tag{2.1}$$

Here $g(\cdot)$ is a strictly increasing link function, $\mu = E(Y|\boldsymbol{X}, \boldsymbol{Z})$, $\boldsymbol{\gamma} \in \mathbb{R}^{p_0}$ is the regular regression coefficient vector, $\boldsymbol{B} \in \mathbb{R}^{p_1 \times \cdots \times p_D}$ is the coefficient array that captures

the effects of tensor covariate $\boldsymbol{X}$, and the inner product between two arrays is defined as $\langle \boldsymbol{B}, \boldsymbol{X} \rangle = \text{vec}(\boldsymbol{B})^{\intercal}\text{vec}(\boldsymbol{X}) = \sum_{i_1,\ldots,i_D} \beta_{i_1\ldots i_D} x_{i_1\ldots i_D}$. This model, without further simplification, is prohibitive given its gigantic dimensionality: $p_0 + \prod_{d=1}^{D} p_d$. Motivated by CP tensor decomposition, Zhou et al. (2013) introduced a low rank structure on the coefficient array $\boldsymbol{B}$. That is, $\boldsymbol{B}$ is assumed to follow a rank-R CP decomposition (1.1). Combining (2.1) and (1.1) yields generalized linear tensor regression models of Zhou et al. (2013), where the dimensionality decreases to the scale of $p_0 + R \times \sum_{d=1}^{D} p_d$. Under this setup, ultra-high dimensionality of (2.1) is reduced to a manageable level, which in turn results in efficient estimation and prediction. For instance, for a regression with 128-by-128-by-128 MRI image and 5 usual covariates, the dimensionality is reduced from the order of $2,097,157 = 5 + 128^3$ to $389 = 5 + 128 \times 3$ for a rank-1 model, and to $1,157 = 5 + 3 \times 128 \times 3$ for a rank-3 model. Zhou et al. (2013) showed that this low rank tensor model could provide a sound recovery of even high rank signals.

In the tensor literature, there has been an important development parallel to CP decomposition, which is called Tucker decomposition, or higher-order singular value decomposition (HOSVD) (Kolda and Bader, 2009). In this chapter, we propose a class of *Tucker tensor regression models*. To differentiate, we call the models of Zhou et al. (2013) *CP tensor regression models*. Specifically, we continue to adopt the model (2.1), but assume that the coefficient array $\boldsymbol{B}$ follows a Tucker decomposition,

$$\boldsymbol{B} = \sum_{r_1=1}^{R_1} \cdots \sum_{r_D=1}^{R_D} g_{r_1,\ldots,r_D} \boldsymbol{\beta}_1^{(r_1)} \circ \cdots \circ \boldsymbol{\beta}_D^{(r_D)}, \tag{2.2}$$

where $\boldsymbol{\beta}_d^{(r_d)} \in \mathbb{R}^{p_d}$ are all column vectors, $d = 1, \ldots, D, r_d = 1, \ldots, R_d$, and $g_{r_1,\ldots,r_D}$ are constants. It is often abbreviated as $\boldsymbol{B} = [\![\boldsymbol{G}; \boldsymbol{B}_1, \ldots, \boldsymbol{B}_D]\!]$, where $\boldsymbol{G} \in \mathbb{R}^{R_1 \times \cdots \times R_D}$ is a D-dimensional *core tensor* with entries $(\boldsymbol{G})_{r_1\ldots r_D} = g_{r_1,\ldots,r_D}$, and $\boldsymbol{B}_d \in \mathbb{R}^{p_d \times R_d}$ are the factor

matrices. $\boldsymbol{B}_d$'s are usually orthogonal and can be thought of as the *principal components* in each dimension (and thus the name, HOSVD). The number of parameters of a Tucker tensor model is in the order of $p_0 + \sum_{d=1}^{D} R_d \times p_d$. Comparing the two decompositions (1.1) and (2.2), the key difference is that CP fixes the number of basis vectors R along each dimension of $\boldsymbol{B}$ so that all $\boldsymbol{B}_d$'s have the *same* number of columns (ranks). In contrast, Tucker allows the number R_d to differ along different dimensions and $\boldsymbol{B}_d$'s could have *different* ranks.

This difference between the two decompositions seems minor; however, in the context of tensor regression modeling and neuroimging analysis, it has profound implications, and such implications essentially motivate the work in this chapter. On one hand, the Tucker tensor regression model shares the advantages of the CP tensor regression model, in that it effectively exploits the special structure of the tensor data, it substantially reduces the dimensionality to enable efficient model estimation, and it provides a sound low rank approximation to a potentially high rank signal. On the other hand, Tucker tensor regression offers a much more *flexible* modeling framework than CP regression, as it allows distinct order along each dimension. When the orders are all identical, it includes the CP model as a special case. This flexibility leads to several improvements that are particularly useful for neuroimaging analysis. First, a Tucker model could be more parsimonious than a CP model thanks to the flexibility of different orders. For instance, suppose a 3D signal $\boldsymbol{B} \in \mathbb{R}^{16\times16\times16}$ admits a Tucker decomposition (2.2) with $R_1 = R_2 = 2$ and $R_3 = 5$. It can only be recovered by a CP decomposition with $R = 5$, costing 230 parameters. In contrast, the Tucker model is more parsimonious with only 131 parameters. This reduction of free parameters is valuable for medical imaging studies, as the number of subjects is often limited. Second, the freedom in the choice of different orders is useful when the tensor data is skewed in dimensions, which is common

in neuroimaging data. For instance, in EEG, the two dimensions consist of electrodes (channels) and time, and the number of sampling time points usually far exceeds the number of channels. Third, even when all tensor modes have comparable sizes, the Tucker formulation explicitly models the interactions between factor matrices $\boldsymbol{B}_d$'s, and as such allows a finer grid search within a larger model space, which in turn may explain more trait variance. Finally, as we will show in Section 2.3, there exists a duality regarding the Tucker tensor model. Thanks to this duality, a Tucker tensor decomposition naturally lends itself to a principled way of imaging data downsizing, which, given the often limited sample size, again plays a practically very useful role in neuroimaging analysis.

For these reasons, we feel it important to develop a complete methodology of Tucker tensor regression and its associated theory. The resulting Tucker tensor model carries a number of useful features. It performs dimension reduction through low rank tensor decomposition but in a supervised fashion, and as such avoids potential information loss in regression. It works for general array-valued image modalities and/or any combination of them, and for various types of responses, including continuous, binary, and count data. Besides, an efficient and highly scalable algorithm has been developed for the associated maximum likelihood estimation. This scalability is important considering the massive scale of imaging data. In addition, regularization has been studied in conjunction with the proposed model, yielding a collection of regularized Tucker tensor models, and particularly one that encourages sparsity of the core tensor to facilitate model selection among the defined Tucker model space.

Recently there have been some increasing interests in matrix/tensor decomposition and their applications in brain imaging studies (Crainiceanu et al., 2011; Allen et al., 2011; Hoff, 2011; Aston and Kirch, 2012). Nevertheless, this work is distinct in that we concentrate on a regression framework with scalar response and tensor valued covariates.

In contrast, Crainiceanu et al. (2011) and Allen et al. (2011) studied unsupervised decomposition, Hoff (2011) considered model-based decomposition, whereas Aston and Kirch (2012) focused on change point distribution estimation. The most closely related work to the proposed method is Zhou et al. (2013); however, we feel our work is *not* a simple extension of theirs. First of all, considering the complex nature of tensor, the development of the Tucker model estimation as well as its asymptotics is far from a trivial extension of the CP model of Zhou et al. (2013). Moreover, we offer a detailed comparison, both analytically (in Section 2.4.4) and numerically (in Sections 2.8.4 and 2.8.5), of the CP and Tucker decompositions in the context of regression with imaging/tensor covariates. We believe this comparison is crucial for an adequate comprehension of tensor regression models and supervised tensor decomposition in general.

2.2 Overview

In this chapter, we present a generalized linear model framework with tensor covariate. We impose a Tucker structure on the array coefficient, which not only substantially reduces the dimensionality, but also helps utilize information of the tensor covariate as a whole. We develop a highly scalable and efficient algorithm for maximum likelihood estimation. We also study statistical theory for the proposed model including score and information, identifiability, consistency and asymptotic normality. In addition, regularized Tucker tensor regression is investigated for stabilizing the estimate. As a competitor to a recent work (Zhou et al., 2013) based on CP decomposition, our proposed model offers a much more flexible modeling framework than CP regression, as it allows a distinct order along each dimension. We use both analytical arguments and numerical study to demonstrate this property.

The rest of the chapter is organized as follows. Section 2.3 begins with a brief review of applications of Tucker decomposition. Section 2.4 reviews of some results on Tucker tensor decomposition, and then presents the Tucker tensor regression model. Section 2.5 develops an efficient algorithm for maximum likelihood estimation. Section 2.6 derives inferential tools such as score, Fisher information, identifiability, consistency, and asymptotic normality. Section 2.7 investigates regularization method for the Tucker regression. Section 2.8 presents extensive numerical results. Section 2.9 concludes with some discussions and points to future extensions. All technical proofs are delegated to the Appendix A.

2.3 Literature Review – Application of Tucker

Tucker decomposition, or HOSVD, can be considered as an extension of the matrix singular value decomposition to multidimensional array. Applications of Tucker decomposition are extensive. De Lathauwer and Vandewalle (2004) and Muti and Bourennane (2005) considered the Tucker decomposition for signal processing. Tucker decomposition is also widely used to construct image approximations and compression in image processing studies (Vasilescu, 2002; Vasilescu and Terzopoulos, 2002; Wang and Ahuja, 2003; Vlasic et al., 2005). In data mining, Savas and Eldén (2007) applied Tucker decomposition to the problem of identifying handwriting digits. Sun et al. (2005) utilized Tucker decomposition for web site click-through data analysis. Liu et al. (2005) and Sun et al. (2006) conducted text mining based on Tucker decomposition.

2.4 Model

2.4.1 Tucker Tensor Decomposition Preliminaries

A *tensor* is a multidimensional array. *Fibers* of a tensor are the higher order analogue of matrix rows and columns. A fiber is defined by fixing every index but one. For instance, a matrix column is a mode-1 fiber and a matrix row is a mode-2 fiber; third-order tensors have column, row, and tube fibers, respectively. We then define the *mode-d multiplication* of the tensor $\boldsymbol{B}$ with a matrix $\boldsymbol{U} \in \mathbb{R}^{p_d \times q}$, denoted by $\boldsymbol{B} \times_d \boldsymbol{U} \in \mathbb{R}^{p_1 \times \cdots \times q \times \cdots \times p_D}$, as the multiplication of the mode-d fibers of $\boldsymbol{B}$ by $\boldsymbol{U}$. In other words, the mode-d matricization of $\boldsymbol{B} \times_d \boldsymbol{U}$ is $\boldsymbol{U}\boldsymbol{B}_{(d)}$.

We also review two properties of a tensor $\boldsymbol{B}$ that admits a Tucker decomposition (2.2). The mode-d matricization of $\boldsymbol{B}$ can be expresses as

$$\boldsymbol{B}_{(d)} = \boldsymbol{B}_d \boldsymbol{G}_{(d)} (\boldsymbol{B}_D \otimes \cdots \otimes \boldsymbol{B}_{d+1} \otimes \boldsymbol{B}_{d-1} \otimes \cdots \otimes \boldsymbol{B}_1)^\intercal,$$

where $\otimes$ denotes the Kronecker product of matrices. If applying the vec operator to $\boldsymbol{B}$, then

$$\text{vec}(\boldsymbol{B}) = \text{vec}(\boldsymbol{B}_{(1)}) = \text{vec}(\boldsymbol{B}_1 \boldsymbol{G}_{(1)} (\boldsymbol{B}_D \otimes \cdots \otimes \boldsymbol{B}_2)^\intercal) = (\boldsymbol{B}_D \otimes \cdots \otimes \boldsymbol{B}_1)\text{vec}(\boldsymbol{G}).$$

These two properties are useful for our subsequent Tucker regression development.

2.4.2 Tucker Regression Model

We elaborate on the Tucker tensor regression model introduced in Section 2.1. We assume that Y belongs to an exponential family with probability mass function or density

(McCullagh and Nelder, 1983),

$$p(y_i|\theta_i, \phi) = \exp\left\{\frac{y_i\theta_i - b(\theta_i)}{a(\phi)} + c(y_i, \phi)\right\} \tag{2.3}$$

with the first two moments $E(Y_i) = \mu_i = b'(\theta_i)$ and $\text{Var}(Y_i) = \sigma_i^2 = b''(\theta_i)a_i(\phi)$. θ and $\phi > 0$ are, respectively, called the natural and dispersion parameters. We assume the systematic part of GLM is of the form

$$g(\mu) = \eta = \boldsymbol{\gamma}^\top \boldsymbol{Z} + \langle \sum_{r_1=1}^{R_1} \cdots \sum_{r_D=1}^{R_D} g_{r_1,\ldots,r_D} \boldsymbol{\beta}_1^{(r_1)} \circ \cdots \circ \boldsymbol{\beta}_D^{(r_D)}, \boldsymbol{X} \rangle. \tag{2.4}$$

That is, we impose a Tucker structure on the array coefficient $\boldsymbol{B}$. We make a few remarks. First, we consider the problem of estimating the core tensor $\boldsymbol{G}$ and factor matrices $\boldsymbol{B}_d$ simultaneously given the response Y and covariates $\boldsymbol{X}$ and $\boldsymbol{Z}$. This can be viewed as a *supervised* version of the classical unsupervised Tucker decomposition. It is also a supervised version of principal components analysis for higher-order multidimensional array. Unlike a two-stage solution that first performs principal components analysis and then fits a regression model, the basis (principal components) $\boldsymbol{B}_d$ in our models are estimated under the guidance (supervision) of the response variable. Second, the CP model of Zhou et al. (2013) corresponds to a special case of the Tucker model (2.4) with $g_{r_1,\ldots,r_D} = 1_{\{r_1=\cdots=r_D\}}$ and $R_1 = \ldots = R_D = R$. In other words, the CP model is a specific Tucker model with a super-diagonal core tensor $\boldsymbol{G}$. The CP model has a rank at most R while the general Tucker model can have a rank as high as R^D. We will further compare the two model sizes in Section 2.4.4.

2.4.3 Duality and Tensor Basis Pursuit

Next we investigate a duality regarding the inner product between a general tensor and a tensor that admits a Tucker decomposition.

Lemma 2 (Duality)**.** *Suppose a tensor* $\boldsymbol{B} \in \mathbb{R}^{p_1 \times \cdots \times p_D}$ *admits Tucker decomposition* $\boldsymbol{B} = [\![\boldsymbol{G}; \boldsymbol{B}_1, \ldots, \boldsymbol{B}_D]\!]$*. Then, for any tensor* $\boldsymbol{X} \in \mathbb{R}^{p_1 \times \cdots \times p_D}$*,* $\langle \boldsymbol{B}, \boldsymbol{X} \rangle = \langle \boldsymbol{G}, \tilde{\boldsymbol{X}} \rangle$*, where* $\tilde{\boldsymbol{X}}$ *admits a Tucker decomposition* $\tilde{\boldsymbol{X}} = [\![\boldsymbol{X}; \boldsymbol{B}_1^\mathsf{T}, \ldots, \boldsymbol{B}_D^\mathsf{T}]\!]$*.*

This duality gives some important insights to the Tucker tensor regression model. First, if we consider $\boldsymbol{B}_d \in \mathbb{R}^{p_d \times R_d}$ as fixed and known basis matrices, then Lemma 2 says fitting the Tucker tensor regression model (2.4) is equivalent to fitting a tensor regression model in $\boldsymbol{G}$ with the *transformed* data $\tilde{\boldsymbol{X}} = [\![\boldsymbol{X}; \boldsymbol{B}_1^\mathsf{T}, \ldots, \boldsymbol{B}_D^\mathsf{T}]\!] \in \mathbb{R}^{R_1 \times \cdots \times R_D}$. When $R_d \ll p_d$, the transformed data $\tilde{\boldsymbol{X}}$ effectively *downsize* the original data. We will further illustrate this downsizing feature in the real data analysis in Section 2.8.5. Second, in applications where the numbers of basis vectors R_d are unknown, we can utilize possibly over-complete basis matrices $\boldsymbol{B}_d$ such that $R_d \geq p_d$, and then estimate $\boldsymbol{G}$ with sparsity regularizations. This leads to a tensor version of the classical basis pursuit problem (Chen et al., 2001). Take fMRI data as an example. We can adopt the wavelet basis for the three image dimensions and the Fourier basis for the time dimension. Regularization on $\boldsymbol{G}$ can be achieved by either imposing a low rank decomposition (CP or Tucker) on $\boldsymbol{G}$ (hard thresholding) or penalized regression (soft thresholding). We will investigate Tucker regression regularization in details in Section 2.7.

2.4.4 Model Size: Tucker vs CP

In this section we investigate the size of the Tucker tensor model. Comparison with the size of the CP tensor model helps gain better understanding of both models. In addition,

it provides a base for data adaptive selection of appropriate orders in a Tucker model.

First we quickly review the number of free parameters p_{C} for a CP model $\boldsymbol{B} = [\![\boldsymbol{B}_1, \ldots, \boldsymbol{B}_d]\!]$, with $\boldsymbol{B}_d \in \mathbb{R}^{p_d \times R}$. For $D = 2$, $p_{\mathrm{C}} = R(p_1 + p_2) - R^2$, and for $D > 2$, $p_{\mathrm{C}} = R(\sum_{d=1}^{D} p_d - D + 1)$. For $D = 2$, the term $-R^2$ adjusts for the nonsingular transformation indeterminancy for model indenfiability; for $D > 2$, the term $R(-D + 1)$ adjusts for the scaling indeterminancy in the CP decomposition. See Zhou et al. (2013) for more details. Following similar arguments, we obtain that the number of free parameters p_{T} in a Tucker model $\boldsymbol{B} = [\![\boldsymbol{G}; \boldsymbol{B}_1, \ldots, \boldsymbol{B}_d]\!]$, with $\boldsymbol{G} \in \mathbb{R}^{R_1 \times \cdots \times R_d}$ and $\boldsymbol{B}_d \in \mathbb{R}^{p_d \times R_d}$, is

$$p_{\mathrm{T}} = \sum_{d=1}^{D} p_d R_d + \prod_{d=1}^{D} R_d - \sum_{d=1}^{D} R_d^2,$$

for any D. Here the term -$\sum_{d=1}^{D} R_d^2$ adjusts for the non-singular transformation indeterminancy in the Tucker decomposition. We summarize these results in Table 2.1.

Next we compare the two model sizes (degrees of freedom) under an additional assumption that $R_1 = \cdots = R_d = R$. The difference becomes:

$$p_{\mathrm{T}} - p_{\mathrm{C}} = \begin{cases} 0 & \text{when } D = 2, \\ R(R-1)(R-2) & \text{when } D = 3, \\ R(R^3 - 4R + 3) & \text{when } D = 4, \\ R(R^{D-1} - DR + D - 1) & \text{when } D > 4. \end{cases}$$

Based on this formula, when $D = 2$, the Tucker model is essentially the same as the CP model. When $D = 3$, Tucker has the same number of parameters as CP for $R = 1$ or $R = 2$, but costs $R(R-1)(R-2)$ more parameters for $R > 2$. When $D > 3$, Tucker and CP are the same for $R = 1$, but Tucker costs substantially more parameters than CP for

Table 2.1: Number of free parameters in Tucker and CP models.

	CP	Tucker
$D = 2$	$R(p_1 + p_2) - R^2$	$p_1R_1 + p_2R_2 + R_1R_2 - R_1^2 - R_2^2$
$D > 2$	$R(\sum_d p_d - D + 1)$	$\sum_d p_dR_d + \prod_d R_d - \sum_d R_d^2$

$R > 2$. For instance, when $D = 4$ and $R = 3$, Tucker model takes 54 more parameters than the CP model. However, one should bear in mind that the above discussion assumes $R_1 = \cdots = R_d = R$. In reality, Tucker could require *less* free parameters than CP, as shown in the illustrative example given in Section 1, since Tucker is more flexible and allows different order R_d along each dimension.

Figure 2.1 shows an example with $D = 3$ dimensional array covariates. Half of the true signal (brain activity map) $\boldsymbol{B}$ is displayed in the left panel, which is by no means a low rank signal. Suppose 3D images $\boldsymbol{X}_i$ are taken on $n = 1,000$ subjects. We simulate image traits $\boldsymbol{X}_i$ from independent standard normals and quantitative traits Y_i from independent normals with mean $\langle \boldsymbol{X}_i, \boldsymbol{B} \rangle$ and unit variance. Given the limited sample size, the hope is to infer a reasonable low rank approximation to the activity map from the 3D image covariates. The right panel displays the model deviance versus the degrees of freedom of a series of CP and Tucker model estimates. The CP model is estimated at ranks $R = 1, \ldots, 5$. The Tucker model is fitted at orders $(R_1, R_2, R_3) = (1, 1, 1)$, $(2, 2, 2)$, $(3, 3, 3)$, $(4, 4, 3)$, $(4, 4, 4)$, $(5, 4, 4)$, $(5, 5, 4)$, and $(5, 5, 5)$. We see from the plot that, under the same number of free parameters, the Tucker model could generally achieve a better model fit with a smaller deviance. (Note that the deviance is in the log scale, so a small discrepancy between the two lines translates to a large value of difference in deviance.)

The explicit model size formula of the Tucker model is also useful for choosing appropriate orders R_d's along each direction given data. This can be treated as a model

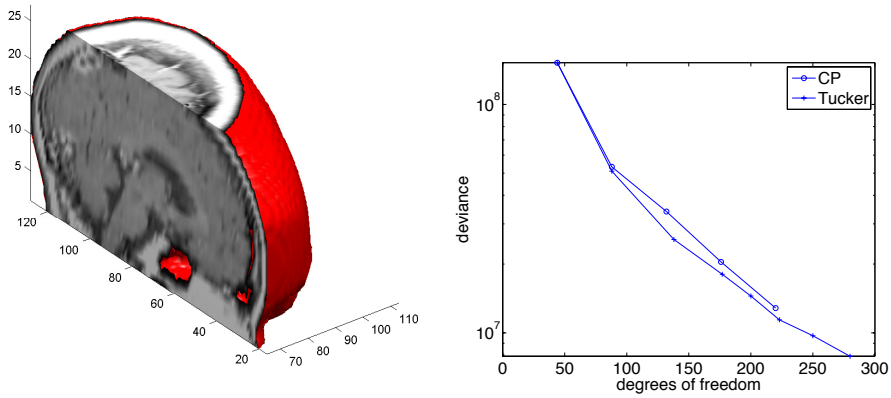

Figure 2.1: Left: half of the true signal array $\boldsymbol{B}$. Right: Deviances of CP regression estimates at $R = 1, \ldots, 5$, and Tucker regression estimates at orders $(R_1, R_2, R_3) = (1,1,1)$, $(2,2,2)$, $(3,3,3)$, $(4,4,3)$, $(4,4,4)$, $(5,4,4)$, $(5,5,4)$, and $(5,5,5)$. The sample size is $n = 1000$.

selection problem, and we can employ a typical model selection criterion, e.g., Bayesian information criterion (BIC). It is of the form: $-2\log\ell + \log(n)p_e$, where ℓ is the log-likelihood, and $p_e = p_{\mathrm{T}}$ is the effective number of parameters of the Tucker model as given in Table 2.1. We will illustrate this BIC criterion in the numerical Section 2.8.1, and will discuss some heuristic guidelines of selecting orders in Section 2.8.5.

2.5 Estimation

We pursue the maximum likelihood estimation (MLE) for the Tucker tensor regression model and develop a scalable estimation algorithm in this section. The key observation is that, although the systematic part (2.4) is not linear in $\boldsymbol{G}$ and $\boldsymbol{B}_d$ *jointly*, it is linear in them *separately*. This naturally suggests a block relaxation algorithm, which updates

each factor matrix $\boldsymbol{B}_d$ and the core tensor $\boldsymbol{G}$ *alternately*.

The algorithm consists of two core steps. First, when updating $\boldsymbol{B}_d \in \mathbb{R}^{p_d \times R_d}$ with the rest $\boldsymbol{B}_{d'}$'s and $\boldsymbol{G}$ fixed , we rewrite the array inner product in (2.4) as

$$\begin{aligned}
\langle \boldsymbol{B}, \boldsymbol{X} \rangle &= \langle \boldsymbol{B}_{(d)}, \boldsymbol{X}_{(d)} \rangle \\
&= \langle \boldsymbol{B}_d \boldsymbol{G}_{(d)} (\boldsymbol{B}_D \otimes \cdots \otimes \boldsymbol{B}_{d+1} \otimes \boldsymbol{B}_{d-1} \otimes \cdots \otimes \boldsymbol{B}_1)^\intercal, \boldsymbol{X}_{(d)} \rangle \\
&= \langle \boldsymbol{B}_d, \boldsymbol{X}_{(d)} (\boldsymbol{B}_D \otimes \cdots \otimes \boldsymbol{B}_{d+1} \otimes \boldsymbol{B}_{d-1} \otimes \cdots \otimes \boldsymbol{B}_1) \boldsymbol{G}_{(d)}^\intercal \rangle.
\end{aligned}$$

Then the problem turns into a GLM regression with $\boldsymbol{B}_d$ as the "parameter" and the term $\boldsymbol{X}_{(d)} (\boldsymbol{B}_D \otimes \cdots \otimes \boldsymbol{B}_{d+1} \otimes \boldsymbol{B}_{d-1} \otimes \cdots \otimes \boldsymbol{B}_1) \boldsymbol{G}_{(d)}^\intercal$ as the "predictor". It is a low dimensional GLM with only $p_d R_d$ parameters and thus is easy to solve. Second, when updating $\boldsymbol{G} \in \mathbb{R}^{R_1 \times \cdots \times R_D}$ with all $\boldsymbol{B}_d$'s fixed,

$$\begin{aligned}
\langle \boldsymbol{B}, \boldsymbol{X} \rangle &= \langle \mathrm{vec}(\boldsymbol{B}), \mathrm{vec}(\boldsymbol{X}) \rangle \\
&= \langle (\boldsymbol{B}_D \otimes \cdots \otimes \boldsymbol{B}_1) \mathrm{vec}(\boldsymbol{G}), \mathrm{vec}(\boldsymbol{X}) \rangle \\
&= \langle \mathrm{vec}(\boldsymbol{G}), (\boldsymbol{B}_D \otimes \cdots \otimes \boldsymbol{B}_1)^\intercal \mathrm{vec}(\boldsymbol{X}) \rangle.
\end{aligned}$$

This implies a GLM regression with $\mathrm{vec}(\boldsymbol{G})$ as the "parameter" and the term $(\boldsymbol{B}_D \otimes \cdots \otimes \boldsymbol{B}_1)^\intercal \mathrm{vec}(\boldsymbol{X})$ as the "predictor". Again this is a low dimensional regression problem with $\prod_d R_d$ parameters. For completeness, we summarize the above alternating estimation procedure in Algorithm 1. The orthogonality between the columns of factor matrices $\boldsymbol{B}_d$ is not enforced as in unsupervised HOSVD (Kilmer and Martin, 2004), because our primary goal is approximating tensor signal instead of finding the principal components along each mode.

Next we study the convergence properties of the proposed algorithm. As the block

Algorithm 1 Block relaxation algorithm for fitting the Tucker tensor regression.

Initialize: $\boldsymbol{\gamma}^{(0)} = \text{argmax}_{\boldsymbol{\gamma}}\, \ell(\boldsymbol{\gamma}, \mathbf{0}, \ldots, \mathbf{0})$, $\boldsymbol{B}_d^{(0)} \in \mathbb{R}^{p_d \times R_d}$ a random matrix for $d = 1, \ldots, D$, and $\boldsymbol{G}^{(0)} \in \mathbb{R}^{R_1 \times \cdots \times R_D}$ a random matrix.
repeat
  for $d = 1, \ldots, D$ **do**
    $\boldsymbol{B}_d^{(t+1)} = \text{argmax}_{\boldsymbol{B}_d}\, \ell(\boldsymbol{\gamma}^{(t)}, \boldsymbol{B}_1^{(t+1)}, \ldots, \boldsymbol{B}_{d-1}^{(t+1)}, \boldsymbol{B}_d, \boldsymbol{B}_{d+1}^{(t)}, \ldots, \boldsymbol{B}_D^{(t)}, \boldsymbol{G}^{(t)})$
  end for
  $\boldsymbol{G}^{(t+1)} = \text{argmax}_{\boldsymbol{G}}\, \ell(\boldsymbol{\gamma}^{(t)}, \boldsymbol{B}_1^{(t+1)}, \ldots, \boldsymbol{B}_D^{(t+1)}, \boldsymbol{G})$
  $\boldsymbol{\gamma}^{(t+1)} = \text{argmax}_{\boldsymbol{\gamma}}\, \ell(\boldsymbol{\gamma}, \boldsymbol{B}_1^{(t+1)}, \ldots, \boldsymbol{B}_D^{(t+1)}, \boldsymbol{G}^{(t+1)})$
until $\ell(\boldsymbol{\theta}^{(t+1)}) - \ell(\boldsymbol{\theta}^{(t)}) < \epsilon$

relaxation algorithm monotonically increases the objective value, the stopping criterion is well-defined and the convergence properties of iterates follow from the standard theory for monotone algorithms (de Leeuw, 1994; Lange, 2010). The proof of next result is given in the Appendix.

Proposition 1. *Assume (i) the log-likelihood function ℓ is continuous, coercive, i.e., the set $\{\boldsymbol{\theta} : \ell(\boldsymbol{\theta}) \geq \ell(\boldsymbol{\theta}^{(0)})\}$ is compact, and bounded above, (ii) the objective function in each block update of Algorithm 1 is strictly concave, and (iii) the set of stationary points (modulo nonsingular transformation indeterminancy) of $\ell(\boldsymbol{\gamma}, \boldsymbol{G}, \boldsymbol{B}_1, \ldots, \boldsymbol{B}_D)$ are isolated. We have the following results.*

1. *(Global Convergence) The sequence $\boldsymbol{\theta}^{(t)} = (\boldsymbol{\gamma}^{(t)}, \boldsymbol{G}^{(t)}, \boldsymbol{B}_1^{(t)}, \ldots, \boldsymbol{B}_D^{(t)})$ generated by Algorithm 1 converges to a stationary point of $\ell(\boldsymbol{\gamma}, \boldsymbol{G}, \boldsymbol{B}_1, \ldots, \boldsymbol{B}_D)$.*

2. *(Local Convergence) Let $\boldsymbol{\theta}^{(\infty)} = (\boldsymbol{\gamma}^{(\infty)}, \boldsymbol{G}^{(\infty)}, \boldsymbol{B}_1^{(\infty)}, \ldots, \boldsymbol{B}_D^{(\infty)})$ be a strict local maximum of ℓ. The iterates generated by Algorithm 1 are locally attracted to $\boldsymbol{\theta}^{(\infty)}$ for $\boldsymbol{\theta}^{(0)}$ sufficiently close to $\boldsymbol{\theta}^{(\infty)}$.*

2.6 Statistical Theory

In this section we study the usual large n asymptotics of the proposed Tucker tensor regression. Regularization is treated in the next section for the small or moderate n cases. For simplicity, we drop the classical covariate $\boldsymbol{Z}$ in this section, but all the results can be straightforwardly extended to include $\boldsymbol{Z}$. We also remark that, although the usually limited sample size of neuroimging studies makes the large n asymptotics seem irrelevant, we still believe such an asymptotic investigation important, for several reasons. First, when the sample size n is considerably larger than the effective number of parameters p_{T}, the asymptotic study tells us that the model is consistently estimating the best Tucker structure approximation to the full array model in the sense of Kullback-Liebler distance. Second, the explicit formula for score and information are not only useful for asymptotic theory but also for computation, while the identifiability issue has to be properly dealt with for the given model. Finally, the regular asymptotics can be of practical relevance, for instance, can be useful in a likelihood ratio type test in a replication study.

2.6.1 Score and Information

We first derive the score and information for the tensor regression model, which are essential for statistical estimation and inference. The following standard calculus notations are used. For a scalar function f, ∇f is the (column) gradient vector, $df = [\nabla f]^{\mathsf{T}}$ is the differential, and $d^2 f$ is the Hessian matrix. For a multivariate function $g : \mathbb{R}^p \mapsto \mathbb{R}^q$, $Dg \in \mathbb{R}^{p \times q}$ denotes the Jacobian matrix holding partial derivatives $\frac{\partial g_j}{\partial x_i}$. We start from the Jacobian and Hessian of the systematic part $\eta \equiv g(\mu)$ in (2.4).

Lemma 3. *1. The gradient* $\nabla\eta(\boldsymbol{B}_1,\ldots,\boldsymbol{B}_D) \in \mathbb{R}^{\prod_d R_d + \sum_{d=1}^D p_d R_d}$ *is*

$$\nabla\eta(\boldsymbol{G},\boldsymbol{B}_1,\ldots,\boldsymbol{B}_D) = [\boldsymbol{B}_D \otimes \cdots \otimes \boldsymbol{B}_1 \ \boldsymbol{J}_1 \ \boldsymbol{J}_2 \ \cdots \ \boldsymbol{J}_D]^\intercal vec(\boldsymbol{X}),$$

where $\boldsymbol{J}_d \in \mathbb{R}^{\prod_{d=1}^D p_d \times p_d R_d}$ *is the Jacobian*

$$\boldsymbol{J}_d = D\boldsymbol{B}(\boldsymbol{B}_d) = \boldsymbol{\Pi}_d\{[(\boldsymbol{B}_D \otimes \cdots \otimes \boldsymbol{B}_{d+1} \otimes \boldsymbol{B}_{d-1} \otimes \cdots \otimes \boldsymbol{B}_1)\boldsymbol{G}_{(d)}^\intercal] \otimes \boldsymbol{I}_{p_d}\} \quad (2.5)$$

and $\boldsymbol{\Pi}_d$ *is the* $(\prod_{d=1}^D p_d)$*-by-*$(\prod_{d=1}^D p_d)$ *permutation matrix that reorders* $vec(\boldsymbol{B}_{(d)})$ *to obtain* $vec(\boldsymbol{B})$*, i.e.,* $vec(\boldsymbol{B}) = \boldsymbol{\Pi}_d\, vec(\boldsymbol{B}_{(d)})$.

2. Let the Hessian $d^2\eta(\boldsymbol{G},\boldsymbol{B}_1,\ldots,\boldsymbol{B}_D) \in \mathbb{R}^{(\prod_d R_d + \sum_d p_d R_d)\times(\prod_d R_d + \sum_d p_d R_d)}$ *be partitioned into four blocks* $\boldsymbol{H}_{\boldsymbol{G},\boldsymbol{G}} \in \mathbb{R}^{\prod_d R_d \times \prod_d R_d}$, $\boldsymbol{H}_{\boldsymbol{G},\boldsymbol{B}} = \boldsymbol{H}_{\boldsymbol{B},\boldsymbol{G}}^\intercal \in \mathbb{R}^{\prod_d R_d \times \sum_d p_d R_d}$ *and* $\boldsymbol{H}_{\boldsymbol{B},\boldsymbol{B}} \in \mathbb{R}^{\sum_d p_d R_d \times \sum_d p_d R_d}$*. Then* $\boldsymbol{H}_{\boldsymbol{G},\boldsymbol{G}} = \boldsymbol{0}$*,* $\boldsymbol{H}_{\boldsymbol{G},\boldsymbol{B}}$ *has entries*

$$h_{(r_1,\ldots,r_D),(i_d,s_d)} \ = \ 1_{\{r_d = s_d\}} \sum_{j_d = i_d} x_{j_1,\ldots,j_D} \prod_{d' \neq d} \beta_{j_{d'}}^{(r_{d'})},$$

and $\boldsymbol{H}_{\boldsymbol{B},\boldsymbol{B}}$ *has entries*

$$h_{(i_d,r_d),(i_{d'},r_{d'})} = 1_{\{d \neq d'\}} \sum_{j_d = i_d, j_{d'} = i_{d'}} x_{j_1,\ldots,j_D} \sum_{s_d = r_d, s_{d'} = r_{d'}} g_{s_1,\ldots,s_D} \prod_{d'' \neq d,d'} \beta_{j_{d''}}^{(s_{d''})}.$$

Furthermore, $\boldsymbol{H}_{\boldsymbol{B},\boldsymbol{B}}$ *can be partitioned in* D^2 *sub-blocks as*

$$\begin{pmatrix} \boldsymbol{0} & * & * & * \\ \boldsymbol{H}_{21} & \boldsymbol{0} & * & * \\ \vdots & \vdots & \ddots & * \\ \boldsymbol{H}_{D1} & \boldsymbol{H}_{D2} & \cdots & \boldsymbol{0} \end{pmatrix}.$$

The elements of sub-block $\boldsymbol{H}_{dd'} \in \mathbb{R}^{p_d R_d \times p_{d'} R_{d'}}$ *can be retrieved from the matrix*

$$\boldsymbol{X}_{(dd')}(\boldsymbol{B}_D \otimes \cdots \otimes \boldsymbol{B}_{d+1} \otimes \boldsymbol{B}_{d-1} \otimes \cdots \otimes \boldsymbol{B}_{d'+1} \otimes \boldsymbol{B}_{d'-1} \otimes \cdots \otimes \boldsymbol{B}_1)\boldsymbol{G}_{(dd')}^{\mathrm{T}}.$$

$\boldsymbol{H}_{\boldsymbol{G},\boldsymbol{B}}$ *can be partitioned into* D *sub-blocks as* $(\boldsymbol{H}_1, \ldots, \boldsymbol{H}_D)$. *The sub-block* $\boldsymbol{H}_d \in \mathbb{R}^{\prod_d R_d \times p_d R_d}$ *has at most* $p_d \prod_d R_d$ *nonzero entries which can be retrieved from the matrix*

$$\boldsymbol{X}_{(d)}(\boldsymbol{B}_D \otimes \cdots \otimes \boldsymbol{B}_{d+1} \otimes \boldsymbol{B}_{d-1} \otimes \cdots \otimes \boldsymbol{B}_1).$$

Let $\ell(\boldsymbol{B}_1, \ldots, \boldsymbol{B}_D | y, \boldsymbol{x}) = \ln p(y|\boldsymbol{x}, \boldsymbol{B}_1, \ldots, \boldsymbol{B}_D)$ be the log-density of GLM. Next result derives the score function, Hessian, and Fisher information of the Tucker tensor regression model.

Proposition 2. *Consider the tensor regression model defined by (2.3) and (2.4).*

1. *The score function (or score vector) is*

$$\nabla \ell(\boldsymbol{G}, \boldsymbol{B}_1, \ldots, \boldsymbol{B}_D) = \frac{(y-\mu)\mu'(\eta)}{\sigma^2} \nabla \eta(\boldsymbol{G}, \boldsymbol{B}_1, \ldots, \boldsymbol{B}_D) \tag{2.6}$$

with $\nabla \eta(\boldsymbol{G}, \boldsymbol{B}_1, \ldots, \boldsymbol{B}_D)$ *given in Lemma 3.*

2. *The Hessian of the log-density* ℓ *is*

$$\begin{aligned} & H(\boldsymbol{G}, \boldsymbol{B}_1, \ldots, \boldsymbol{B}_D) \\ = & -\left[\frac{[\mu'(\eta)]^2}{\sigma^2} - \frac{(y-\mu)\theta''(\eta)}{\sigma^2}\right] \nabla \eta(\boldsymbol{G}, \boldsymbol{B}_1, \ldots, \boldsymbol{B}_D) d\eta(\boldsymbol{G}, \boldsymbol{B}_1, \ldots, \boldsymbol{B}_D) \\ & + \frac{(y-\mu)\theta'(\eta)}{\sigma^2} d^2\eta(\boldsymbol{G}, \boldsymbol{B}_1, \ldots, \boldsymbol{B}_D), \end{aligned} \tag{2.7}$$

with $d^2\eta$ defined in Lemma 3.

3. *The Fisher information matrix is*

$$\begin{aligned} & \boldsymbol{I}(\boldsymbol{G}, \boldsymbol{B}_1, \ldots, \boldsymbol{B}_D) \\ = \; & E[-H(\boldsymbol{G}, \boldsymbol{B}_1, \ldots, \boldsymbol{B}_D)] \\ = \; & \mathrm{Var}[\nabla \ell(\boldsymbol{G}, \boldsymbol{B}_1, \ldots, \boldsymbol{B}_D) d\ell(\boldsymbol{G}, \boldsymbol{B}_1, \ldots, \boldsymbol{B}_D)] \\ = \; & \frac{[\mu'(\eta)]^2}{\sigma^2} [\boldsymbol{B}_D \otimes \cdots \otimes \boldsymbol{B}_1 \; \boldsymbol{J}_1 \ldots \boldsymbol{J}_D]^{\mathsf{T}} vec(\boldsymbol{X}) vec(\boldsymbol{X})^{\mathsf{T}} [\boldsymbol{B}_D \otimes \cdots \otimes \boldsymbol{B}_1 \; \boldsymbol{J}_1 \ldots \boldsymbol{J}_D]. \end{aligned} \tag{2.8}$$

Remark: For canonical link, $\theta = \eta$, $\theta'(\eta) = 1$, $\theta''(\eta) = 0$, and the second term of Hessian vanishes. For the classical GLM with linear systematic part ($D = 1$), $d^2\eta(\boldsymbol{G}, \boldsymbol{B}_1, \ldots, \boldsymbol{B}_D)$ is zero and thus the third term of Hessian vanishes. For the classical GLM ($D = 1$) with canonical link, both second and third terms of the Hessian vanish and thus the Hessian is non-stochastic, coinciding with the information matrix.

2.6.2 Identifiability

The Tucker decomposition (2.2) is unidentifiable due to the nonsingular transformation indeterminancy. That is

$$[\![\boldsymbol{G}; \boldsymbol{B}_1, \ldots, \boldsymbol{B}_D]\!] = [\![\boldsymbol{G} \times_1 \boldsymbol{O}_1^{-1} \times \cdots \times_D \boldsymbol{O}_D^{-1}; \boldsymbol{B}_1 \boldsymbol{O}_1, \ldots, \boldsymbol{B}_D \boldsymbol{O}_D]\!]$$

for any nonsingular matrices $\boldsymbol{O}_d \in \mathbb{R}^{R_d \times R_d}$. This implies that the number of free parameters for a Tucker model is $\sum_d p_d R_d + \prod_d R_d - \sum_d R_d^2$, with the last term adjusting for nonsingular indeterminancy. Therefore the Tucker model is identifiable only in terms of

the equivalency classes.

For asymptotic consistency and normality, it is necessary to adopt a specific constrained parameterization. It is common to impose the orthonormality constraint on the factor matrices $\boldsymbol{B}_d^\intercal \boldsymbol{B}_d = \boldsymbol{I}_{R_d}$, $d = 1, \ldots, D$. However the resulting parameter space is a manifold and much harder to deal with. We adopt an alternative parameterization that fixes the entries of the first R_d rows of $\boldsymbol{B}_d$ to be ones

$$\mathcal{B} = \{[\![\boldsymbol{G}; \boldsymbol{B}_1, \ldots, \boldsymbol{B}_D]\!] : \beta_{i_d}^{(r)} = 1, i_d = 1, \ldots, R_d, d = 1, \ldots, D\}.$$

The formulae for score, Hessian and information in Proposition 2 require changes accordingly. The entries in the first R_d rows of $\boldsymbol{B}_d$ are fixed at ones and their corresponding entries, rows and columns in score, Hessian and information need to be deleted. Choice of the restricted space $\mathcal{B}$ is obviously arbitrary, and excludes arrays with any entries in the first rows of $\boldsymbol{B}_d$ equal to zeros. However the set of such exceptional arrays has Lebesgue measure zero. In specific applications, subject knowledge may suggest alternative restrictions on the parameters.

Given a finite sample size, conditions for global identifiability of parameters are in general hard to obtain except in the linear case ($D = 1$). Local identifiability essentially requires linear independence between the "collapsed" vectors: $[\boldsymbol{B}_D \otimes \cdots \otimes \boldsymbol{B}_1 \ \boldsymbol{J}_1 \ldots \boldsymbol{J}_D]^\intercal \text{vec}(\boldsymbol{x}_i) \in \mathbb{R}^{\sum_d p_d R_d + \prod_d R_d - \sum_d R_d^2}$.

Proposition 3 (Identifiability)**.** *Given iid data points* $\{(y_i, \boldsymbol{x}_i), i = 1, \ldots, n\}$ *from the Tucker tensor regression model. Let* $\boldsymbol{B}_0 \in \mathcal{B}$ *be a parameter point and assume there exists an open neighborhood of* $\boldsymbol{B}_0$ *in which the information matrix has a constant rank. Then*

$\boldsymbol{B}_0$ is locally identifiable if and only if

$$I(\boldsymbol{B}_0) = [\boldsymbol{B}_D \otimes \cdots \otimes \boldsymbol{B}_1 \ \boldsymbol{J}_1 \dots \boldsymbol{J}_D]^\intercal \left[\sum_{i=1}^{n} \frac{\mu'(\eta_i)^2}{\sigma_i^2} vec(\boldsymbol{x}_i) vec(\boldsymbol{x}_i)^\intercal \right] \cdot [\boldsymbol{B}_D \otimes \cdots \otimes \boldsymbol{B}_1 \ \boldsymbol{J}_1 \dots \boldsymbol{J}_D]$$

is nonsingular.

2.6.3 Asymptotics

The asymptotics for tensor regression follow from those for MLE or M-estimation. The key observation is that the nonlinear part of tensor model (2.4) is a degree-D polynomial of parameters and the collection of polynomials $\{\langle \boldsymbol{B}, \boldsymbol{X} \rangle, \boldsymbol{B} \in \mathcal{B}\}$ form a Vapnik-Červonenkis (VC) class. Then the classical uniform convergence theory applies (van der Vaart, 1998). For asymptotic normality, we need to establish that the log-likelihood function of tensor regression model is quadratic mean differentiable (Lehmann and Romano, 2005). A sketch of the proof is given in the Appendix.

Theorem 1. *Assume $\boldsymbol{B}_0 \in \mathcal{B}$ is (globally) identifiable up to permutation and the array covariates $\boldsymbol{X}_i$ are iid from a bounded underlying distribution.*

1. *(Consistency) The MLE is consistent, i.e., $\hat{\boldsymbol{B}}_n$ converges to $\boldsymbol{B}_0$ in probability, in following models. (1) Normal tensor regression with a compact parameter space $\mathcal{B}_0 \subset \mathcal{B}$. (2) Binary tensor regression. (3) Poisson tensor regression with a compact parameter space $\mathcal{B}_0 \subset \mathcal{B}$.*

2. *(Asymptotic Normality) For an interior point $\boldsymbol{B}_0 \in \mathcal{B}$ with nonsingular information matrix $\boldsymbol{I}(\boldsymbol{B}_0)$ (2.8) and $\hat{\boldsymbol{B}}_n$ is consistent, $\sqrt{n}[vec(\hat{\boldsymbol{B}}_n) - vec(\boldsymbol{B}_0)]$ converges in distribution to a normal with mean zero and covariance matrix $\boldsymbol{I}^{-1}(\boldsymbol{B}_0)$.*

In practice it is rare that the true regression coefficient $\boldsymbol{B}_{\text{true}} \in \mathbb{R}^{p_1 \times \cdots \times p_D}$ is exactly a low rank tensor. However the MLE of the rank-R tensor model converges to the maximizer of function $M(\boldsymbol{B}) = \mathbb{P}_{\boldsymbol{B}_{\text{true}}} \ln p_{\boldsymbol{B}}$ or equivalently $\mathbb{P}_{\boldsymbol{B}_{\text{true}}} \ln(p_{\boldsymbol{B}}/p_{\boldsymbol{B}_{\text{true}}})$. In other words, the MLE consistently estimates the best approximation (among models in $\mathcal{B}$) of $\boldsymbol{B}_{\text{true}}$ in the sense of Kullback-Leibler distance.

2.7 Regularized Estimation

Regularization plays a crucial role in neuroimaging analysis for several reasons. First, even after substantial dimension reduction by imposing a Tucker structure, the number of parameters p_{T} can still exceed the number of observations n. Second, even when $n > p_{\text{T}}$, regularization could potentially be useful for stabilizing the estimates and improving the risk property. Finally, regularization is an effective way to incorporate prior scientific knowledge about brain structures. For instance, it may sometimes be reasonable to impose symmetry on the parameters along the coronal plane for MRI images.

In our context of Tucker regularized regression, there are two possible types of regularizations, one on the core tensor $\boldsymbol{G}$ *only*, and the other on both $\boldsymbol{G}$ and $\boldsymbol{B}_d$ *simultaneously*. Which regularization to use depends on the practical purpose of a scientific study. In this section, we illustrate the regularization on the core tensor, which simultaneously achieves sparsity in the number of outer products in Tucker decomposition (2.2) and shrinkage. Toward that purpose, we propose to maximize the regularized log-likelihood

$$\ell(\boldsymbol{\gamma}, \boldsymbol{G}, \boldsymbol{B}_1, \ldots, \boldsymbol{B}_D) - \sum_{r_1,\ldots,r_D} P_\eta(|g_{r_1,\ldots,r_D}|, \lambda),$$

where $P_\eta(|x|, \lambda)$ is a scalar penalty function, λ is the penalty tuning parameter, and

η is an index for the penalty family. Note that the penalty term above only involves elements of the core tensor, and thus regularization on $\boldsymbol{G}$ only. This formulation includes a large class of penalty functions, including power family (Frank and Friedman, 1993), where $P_\eta(|x|, \lambda) = \lambda|x|^\eta$, $\eta \in (0, 2]$, and in particular lasso (Tibshirani, 1996) ($\eta = 1$) and ridge ($\eta = 2$); elastic net (Zou and Hastie, 2005), where $P_\eta(|x|, \lambda) = \lambda[(\eta-1)x^2/2+(2-\eta)|x|]$, $\eta \in [1, 2]$; SCAD (Fan and Li, 2001), where $\partial/\partial|x|P_\eta(|x|, \lambda) = \lambda\left\{1_{\{|x|\leq\lambda\}} + (\eta\lambda - |x|)_+/(\eta-1)\lambda 1_{\{|x|>\lambda\}}\right\}$, $\eta > 2$; and MC+ penalty (Zhang, 2010), where $P_\eta(|x|, \lambda) = \{\lambda|x| - x^2/(2\eta)\}\, 1_{\{|x|<\eta\lambda\}} + 0.5\lambda^2\eta 1_{\{|x|\geq\eta\lambda\}}$, among many others.

Two aspects of the proposed regularized Tucker regression, parameter estimation and tuning, deserve some discussion. For regularized estimation, it incurs only slight changes in Algorithm 1. That is, when updating $\boldsymbol{G}$, we simply fit a penalized GLM regression problem,

$$\boldsymbol{G}^{(t+1)} = \text{argmax}_{\boldsymbol{G}}\, \ell(\boldsymbol{\gamma}^{(t)}, \boldsymbol{B}_1^{(t+1)}, \ldots, \boldsymbol{B}_D^{(t+1)}, \boldsymbol{G}) - \sum_{r_1,\ldots,r_D} P_\eta(|g_{r_1,\ldots,r_D}|, \lambda),$$

for which many software packages exist. Our implementation utilizes an efficient MATLAB toolbox for sparse regression (Zhou et al., 2011). Other steps of Algorithm 1 remain unchanged. For the regularization to remain legitimate, we constrain the column norms of $\boldsymbol{B}_d$ to be one when updating factor matrices $\boldsymbol{B}_d$. For parameter tuning, one can either use the general cross validation approach, or employ Bayesian information criterion to tune the penalty parameter λ.

2.8 Numerical Study

We have carried out intensive numerical experiments to study the finite sample performance of the Tucker regression. Our simulations focus on four aspects: first, we demonstrate the capacity of the Tucker regression in identifying various shapes of signals; second, we compare the performance of the regularized Tucker regression with other standard statistical learning techniques on vectorized images; third, we study the consistency property of the method by gradually increasing the sample size; fourth, we compare the performance of the Tucker regression with the CP regression of Zhou et al. (2013). We also examine a real MRI imaging data to illustrate the Tucker downsizing and to further compare the two tensor models.

2.8.1 Identification of Various Shapes of Signals

In our first example, we demonstrate that the proposed Tucker regression model, though with substantial reduction in dimension, can manage to identify a range of two dimensional signal shapes with varying ranks. In Figure 2.2, we list the 2D signals $\boldsymbol{B} \in \mathbb{R}^{64\times 64}$ in the first row, along with the estimates by Tucker tensor models in the second to fourth rows with orders $(1,1),(2,2)$ and $(3,3)$, respectively. Note that, since the orders along both dimensions are made equal, the Tucker model is to perform essentially the same as a CP model in this example, and the results are presented here for completeness. We will examine differences of the two models in later examples. The regular covariate vector $\boldsymbol{Z} \in \mathbb{R}^5$ and image covariate $\boldsymbol{X} \in \mathbb{R}^{64\times 64}$ are randomly generated with all elements being independent standard normals. The response Y is generated from a normal model with mean $\mu = \boldsymbol{\gamma}^\intercal \boldsymbol{Z} + \langle \boldsymbol{B}, \boldsymbol{X} \rangle$ and variance $\mathrm{var}(\mu)/10$. The vector coefficient $\boldsymbol{\gamma} = \mathbf{1}_5$, and the coefficient array $\boldsymbol{B}$ is binary, with the signal region equal to one and the rest

zero. Note that this problem differs from the usual edge detection or object recognition in imaging processing (Qiu, 2005, 2007). In our setup, all elements of the image $\boldsymbol{X}$ follow the same distribution. The signal region is defined through the coefficient matrix $\boldsymbol{B}$ and needs to be inferred from the relation between Y and $\boldsymbol{X}$ after adjusting for $\boldsymbol{Z}$. It is clear to see in Figure 2.2 that, the Tucker model yields a sound recovery of the true signals, even for those of high rank or natural shape, e.g., "disk" and "butterfly". We also illustrate in the plot the BIC criterion in Section 2.4.4.

2.8.2 Comparison of the Regularized Tucker and Standard Statistical Models

In our second example, we demonstrate the comparison of regularized Tucker model with LASSO penalty (R-Tucker), vectorized ν-SVM regression model (V-ν-SVM) and vectorized regularized linear regression model with LASSO penalty (V-GLM). The regular covariate vector $\boldsymbol{Z} \in \mathbb{R}^5$ and image covariate $\boldsymbol{X} \in \mathbb{R}^{64\times 64}$ are randomly generated with all elements being independent standard normals. The response Y is generated from a normal model with mean $\mu = \boldsymbol{\gamma}^\intercal \boldsymbol{Z} + \langle \boldsymbol{B}, \boldsymbol{X} \rangle$ and variance $\mathrm{var}(\mu)/10$. The vector coefficient $\boldsymbol{\gamma} = \mathbf{1}_5$, and the coefficient array $\boldsymbol{B}$ is binary, with the signal region equal to one and the rest zero. Regularization parameter λ is tuned for the three models above. For V-ν-SVM, parameter ν is tuned in $\{0.2, 0.4, 0.6, 0.8\}$. For Tucker model, ranks are also tuned in $\{1, 1, 1\}, \{2, 2, 2\}, \{3, 3, 3\}$. We fix the sample size for both training set and tuning set to be 1000. Parameters are tuned by minimizing the mean squared error of prediction in the tuning set. The results shown in Table 2.2 are based on 100 data replicates. In addition, we also give the recovered shapes of $\boldsymbol{B}$ in Figure 2.3. The results in both number and figure show that the proposed R-Tucker yields a more sound recovery

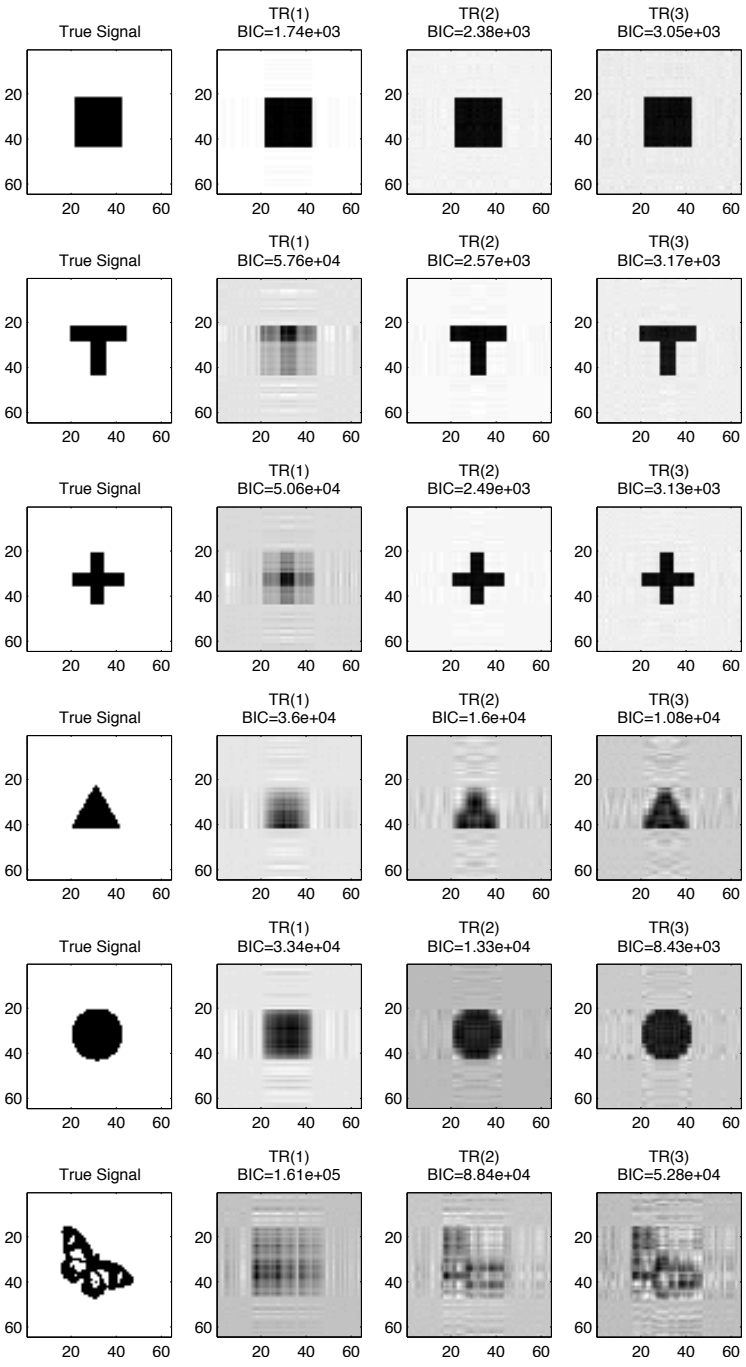

Figure 2.2: True and recovered image signals by Tucker regression. The regression coefficient for each entry is either 0 (white) or 1 (black). TR(r) means estimate from the Tucker regression with an r-by-r core tensor.

of the true signals.

Table 2.2: Comparison of regularized Tucker model with LASSO penalty (R-Tucker), vectorized ν-SVM regression model (V-ν-SVM) and vectorized regularized linear regression model with LASSO penalty (V-GLM) in terms of root mean squared error (RMSE) of $\boldsymbol{B}$ Estimate. Results are averaged over 100 replications. Numbers in parentheses are standard errors

	Rectangular	'T' Shape	Cross	Triangle	Disk	Butterfly
R-Tucker	0.041(0.003)	0.049(0.003)	0.046(0.003)	0.108(0.005)	0.121(0.008)	0.223(0.009)
V-ν-SVM	0.299(0.002)	0.233(0.001)	0.215(0.001)	0.207(0.001)	0.280(0.002)	0.284(0.002)
V-GLM	0.313(0.005)	0.210(0.006)	0.179(0.006)	0.162(0.008)	0.285(0.006)	0.290(0.005)

2.8.3 Performance with Increasing Sample Size

In our third example, we continue to employ a similar model as in Figure 2.2 but with a three dimensional image covariate. The dimension of $\boldsymbol{X}$ is set as $p_1 \times p_2 \times p_3$, with $p_1 = p_2 = p_3 = 16$ and 32, respectively. The signal array $\boldsymbol{B}$ is generated from a Tucker structure, with the elements of core tensor $\boldsymbol{G}$ and the factor matrices $\boldsymbol{B}$'s all coming from independent standard normals. The dimension of the core tensor $\boldsymbol{G}$ is set as $R_1 \times R_2 \times R_3$, with $R_1 = R_2 = R_3 = 2, 5$ and 8, respectively. We gradually increase the sample size, starting with an n that is in hundreds and no smaller than the degrees of freedom of the generating model. We aim to achieve two purposes with this example: first, we verify the consistency property of the proposed estimator, and second, we gain some practical knowledge about the estimation accuracy with different values of the sample size. Figure 2.4 summarizes the results. It is clearly seen that the estimation improves with the increasing sample size. Meanwhile, we observe that, unless the core tensor dimension is small, one would require a relatively large sample size to achieve a good estimation

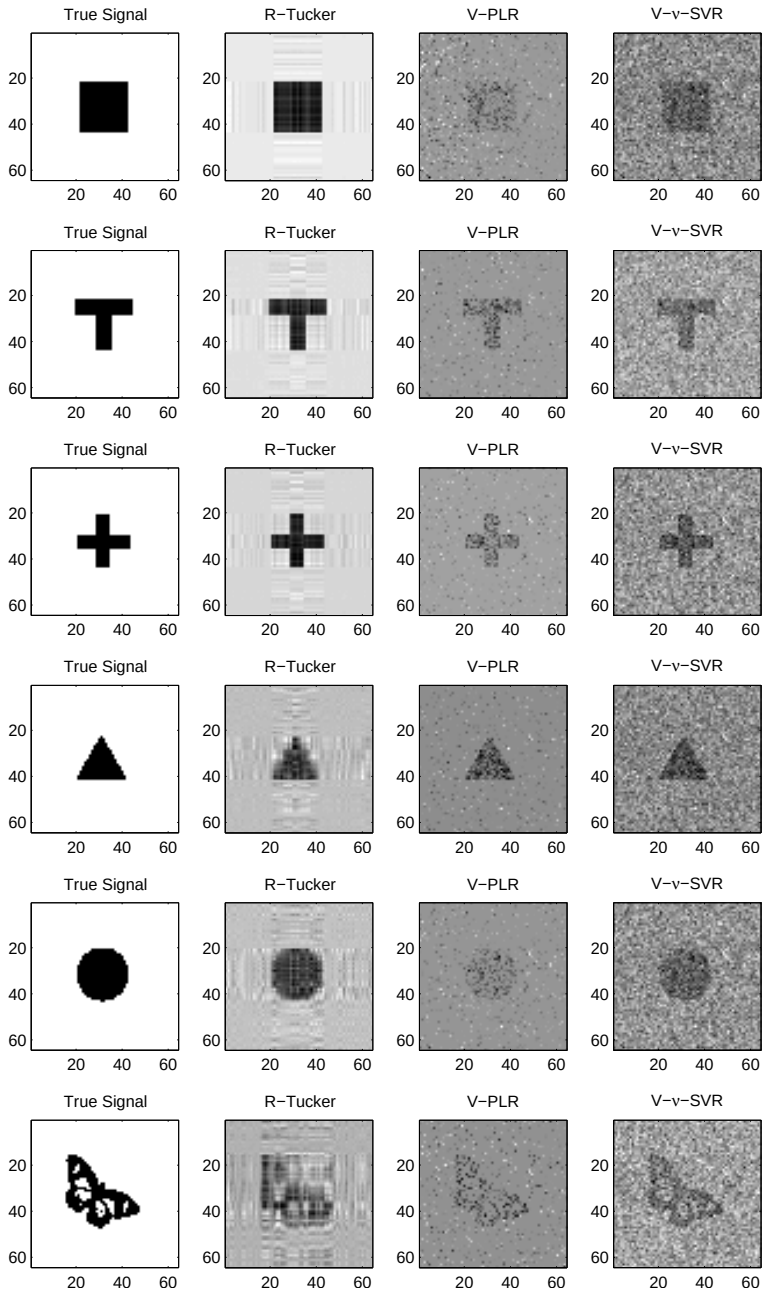

Figure 2.3: True and recovered image signals by different models. The matrix variate has size 64 by 64 with entries generated as independent standard normals. The regression coefficient for each entry is either 0 (white) or 1 (black). The sample size is 1000.

accuracy. This is not surprising though, considering the number of parameters of the model and that regularization is not employed here. The proposed tensor regression approach has been primarily designed for imaging studies with a reasonably large number of subjects. Recently, a number of such large-scale brain imaging studies are emerging. For instance, the Attention Deficit Hyperactivity Disorder Sample Initiative (ADHD, 2013) consists of over 900 participants and the Alzheimer's Disease Neuroimaging Initiative (ADNI, 2013) database accumulates over 3,000 participants. In addition, regularization discussed in Section 2.7 and the Tucker downsizing in Section 2.4.3 can both help improve estimation given a limited sample size.

2.8.4 Comparison of the Tucker and CP Models

In our fourth example, we focus on comparison between the Tucker tensor model with the CP tensor model of Zhou et al. (2013). We generate a normal response, and the 3D signal array $\boldsymbol{B}$ with dimensions p_1, p_2, p_3 and the d-ranks r_1, r_2, r_3. Here, the d-rank is defined as the column rank of the mode-d matricization $\boldsymbol{B}_{(d)}$ of $\boldsymbol{B}$. We set $p_1 = p_2 = p_3 = 16$ and 32, and $(r_1, r_2, r_3) = (5, 3, 3), (8, 4, 4)$ and $(10, 5, 5)$, respectively. The sample size is 2000. We fit a Tucker model with $R_d = r_d$, and a CP model with $R = \max r_d$, $d = 1, 2, 3$. We report in Table 2.3 the degrees of freedom of the two models under different setup, as well as the root mean squared error (RMSE) out of 100 data replications. It is seen that the Tucker model requires a smaller number of free parameters, while it achieves a more accurate estimation compared to the CP model. Such advantages come from the flexibility of the Tucker decomposition that permits different orders R_d along directions.

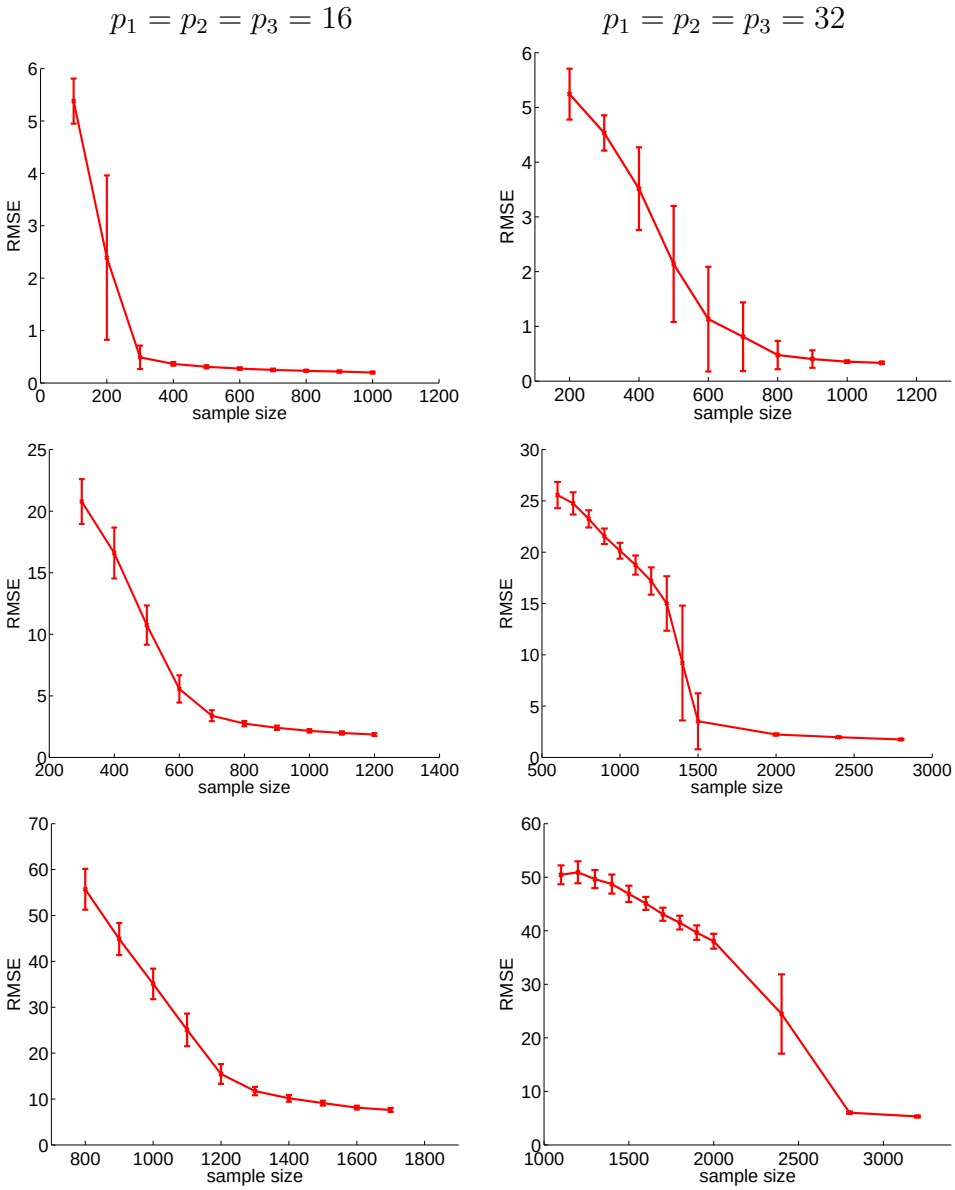

Figure 2.4: Root mean squared error (RMSE) of the tensor parameter estimate versus the sample size. Reported are the average and standard deviation of RMSE based on 100 data replications. Top: $R_1 = R_2 = R_3 = 2$; Middle: $R_1 = R_2 = R_3 = 5$; Bottome: $R_1 = R_2 = R_3 = 8$.

Table 2.3: Comparison of the Tucker and CP models. Reported are the average and standard deviation (in the parenthesis) of the root mean squared error, all based on 100 data replications.

Dimension	Criterion	Model	$(5, 3, 3)$	$(8, 4, 4)$	$(10, 5, 5)$
$16 \times 16 \times 16$	Df	Tucker	178	288	420
		CP	230	368	460
	RMSE	Tucker	0.202 (0.013)	0.379 (0.017)	0.728 (0.030)
		CP	0.287 (0.033)	1.030 (0.081)	2.858 (0.133)
$32 \times 32 \times 32$	Df	Tucker	354	544	740
		CP	470	752	940
	RMSE	Tucker	0.288 (0.013)	0.570 (0.023)	1.236 (0.045)
		CP	0.392 (0.046)	1.927 (0.172)	16.238 (3.867)

2.8.5 Attention Deficit Hyperactivity Disorder Data Analysis

We analyze the attention deficit hyperactivity disorder (ADHD) data from the ADHD-200 Sample Initiative (ADHD, 2013) to illustrate our proposed method as well as the Tucker downsizing. ADHD is a common childhood disorder and can continue through adolescence and adulthood. Symptoms include difficulty in staying focused and paying attention, difficulty in controlling behavior, and over-activity. The data set that we analyzed is part of the ADHD-200 Global Competition data sets. It was pre-partitioned into a training data of 770 subjects and a testing data of 197 subjects. We removed those subjects with missing observations or poor image quality, resulting in 762 training subjects and 169 testing subjects. In the training set, there were 280 combined ADHD subjects, 482 normal controls, and the case-control ratio is about 3:5. In the testing set, there were 76 combined ADHD subjects, 93 normal controls, and the case-control ratio is about 4:5. T1-weighted images were acquired for each subject, and were preprocessed by standard steps. The data we used is obtained from the Neuro Bureau after preprocessing (the

Burner data, http://neurobureau.projects.nitrc.org/ADHD200/Data.html). In addition to the MRI image predictor, we also include the subjects' age, gender and handiness as regular covariates. The response is the binary diagnosis status.

The original image size was $p_1 \times p_2 \times p_3 = 121 \times 145 \times 121$. We employ the Tucker downsizing in Section 2.4.3. More specifically, we first choose a wavelet basis for $\boldsymbol{B}_d \in \mathbb{R}^{p_d \times \tilde{p}_d}$, then transform the image predictor from $\boldsymbol{X}$ to $\tilde{\boldsymbol{X}} = [\![\boldsymbol{X}; \boldsymbol{B}_1^\mathsf{T}, \ldots, \boldsymbol{B}_D^\mathsf{T}]\!]$. We pre-specify the values of $\tilde{p}_d$'s that are about tenth of the original dimensions p_d, and equivalently, we fit a Tucker tensor regression with the image predictor dimension downsized to $\tilde{p}_1 \times \tilde{p}_2 \times \tilde{p}_3$. In our example, we have experimented with a set of values of $\tilde{p}_d$'s, and the results are qualitatively similar. We report two sets, $\tilde{p}_1 = 12$, $\tilde{p}_2 = 14$, $\tilde{p}_3 = 12$, and $\tilde{p}_1 = 10$, $\tilde{p}_2 = 12$, $\tilde{p}_3 = 10$. We have also experimented with the Haar wavelet basis (Daubechies D2) and the Daubechies D4 wavelet basis, which again show similar qualitative patterns.

For $\tilde{p}_1 = 12, \tilde{p}_2 = 14, \tilde{p}_3 = 12$, we fit a Tucker tensor model with $R_1 = R_2 = R_3 = 3$, resulting in 114 free parameters, and fit a CP tensor model with $R = 4$, resulting in 144 free parameters. For $\tilde{p}_1 = 10, \tilde{p}_2 = 12, \tilde{p}_3 = 10$, we fit a Tucker tensor model with $R_1 = R_2 = 2$ and $R_3 = 3$, resulting in 71 free parameters, and fit a CP tensor model with $R = 4$, resulting in 120 free parameters. We have chosen those orders based on the following considerations. First, the number of free parameters of the Tucker and CP models are comparable. Second, at each step of GLM model fit, we ensure that the ratio between the sample size n and the number of parameters under estimation in that step $\tilde{p}_d \times R_d$ satisfies a heuristic rule of greater than two in normal models and greater than five in logistic models. In the Tucker model, we also ensure the ratio between n and the number of parameters in the core tensor estimation $\prod_d R_d$ satisfies this rule. We note that this selection of Tucker orders is heuristic; however, it seems to be a useful guideline especially when the data is noisy. We also fit a regularized Tucker model and a regularized

Table 2.4: ADHD testing data misclassification error.

Basis	Reduced dimension	Reg-Tucker	Reg-CP	Tucker	CP
Haar (D2)	$12 \times 14 \times 12$	0.361	0.367	0.379	0.438
	$10 \times 12 \times 10$	0.343	0.390	0.379	0.408
Daubechies (D4)	$12 \times 14 \times 12$	0.337	0.385	0.385	0.414
	$10 \times 12 \times 10$	0.320	0.396	0.367	0.373

CP model with the same orders, while the penalty parameter is tuned based on 5-fold cross validation of the training data. Figure 2.5 shows that we select λ with the smallest number of misclassifications in regularized Tucker model.

We evaluate each model by comparing the misclassification error rate on the independent testing set. The results are shown in Table 2.4. We see from the table that, the regularized Tucker model performs the best, which echoes the findings in our simulations above. We also remark that, considering the fact that the ratio of case-control is about 4:5 in the testing data, the misclassification rate from 0.32 to 0.36 achieved by the regularized Tucker model indicates a fairly sound classification accuracy. On the other hand, we note that, a key advantage of our proposed approach is its capability of suggesting a useful model rather than the classification accuracy per se. This is different from black-box type machine learning based imaging classifiers.

It is also of interest to compare the run times of the two tensor model fittings. We record the run times of fitting the Tucker and CP models with the ADHD training data in Table 2.5. They are comparable.

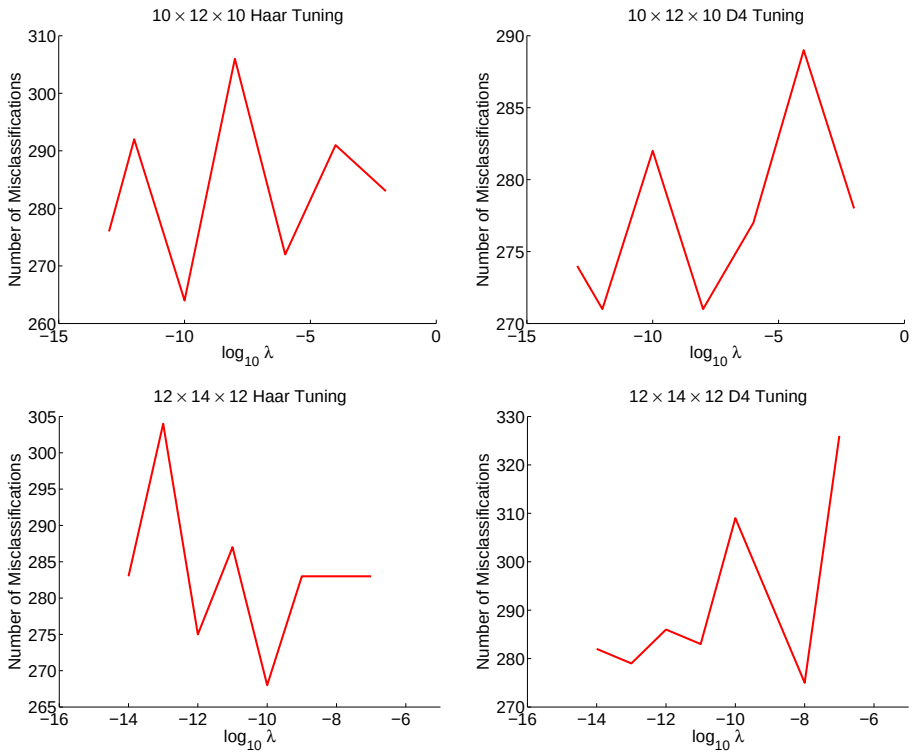

Figure 2.5: Grid search for regularization parameter, λ, in regularized Tucker model. λ with smallest number of misclassifications in the cross validation is selected. Different settings for resizing are reported.

Table 2.5: ADHD model fitting run time (in seconds).

Basis	Reduced dimension	Reg-Tucker	Reg-CP	Tucker	CP
Haar (D2)	$12 \times 14 \times 12$	3.68	4.39	31.25	22.43
	$10 \times 12 \times 10$	1.36	2.79	9.08	25.10
Daubechies (D4)	$12 \times 14 \times 12$	3.30	2.18	16.87	26.34
	$10 \times 12 \times 10$	1.92	1.90	9.96	17.10

2.9 Discussion

2.9.1 Summary

We have proposed a tensor regression model based on the Tucker decomposition. Including the CP tensor regression (Zhou et al., 2013) as a special case, Tucker model provides a more flexible framework for regression with imaging covariates. We develop a fast estimation algorithm, a general regularization procedure, and the associated asymptotic properties. In addition, we provide a detailed comparison, both analytically and numerically, of the Tucker and CP tensor models.

In real imaging analysis, the signal hardly has an exact low rank. On the other hand, given the limited sample size, a low rank estimate often provides a reasonable approximation to the true signal. This is why the low rank models such as the Tucker and CP could offer a sound recovery of even a complex signal.

The tensor regression framework established in this chapter is general enough to encompass a large number of potential extensions, including but not limited to imaging multi-modality analysis, imaging classification, and longitudinal imaging analysis. These extensions consist of our future research.

Table 2.6: Fitting run time for recovering 'triangle' shape (in seconds) at rank 1, 2, 3 and 4. Reported are the median time of 100 runs.

	$r = 1$	$r = 2$	$r = 3$	$r = 4$
Triangle	0.30	1.21	4.22	9.62

Table 2.7: Computing time comparison of the Tucker and CP models in Section 2.8.4. Reported are median time (in seconds) over 100 runs.

Dimension	Model	$(5, 3, 3)$	$(8, 4, 4)$	$(10, 5, 5)$
$16 \times 16 \times 16$	Tucker	1.03	1.71	3.03
	CP	7.47	11.51	15.64
$32 \times 32 \times 32$	Tucker	9.93	21.51	50.52
	CP	28.36	59.82	89.81

2.9.2 Computation Time

We conduct a numerical experiment to study the computing time of proposed Tucker model. We adopt the setting in Section 2.8.1 and use the 'triangle' shape. We test the computing time by running the algorithm on one data set from 100 random starting points. Table 2.6 displays the median wall clock run times.

In addition, we compare the computing time for Tucker and CP under the simulation setup in Section 2.8.4. We test the computing time by running the algorithms on one data set from 100 random starting points. Table 2.7 displays the results.

Our proposed Tucker tensor regression model aims to address the computational and modeling challenges of large-scale data set, which has been considered in Section 2.8.5. Table 2.5 has shown that the run times of Tucker and CP model are comparable for the ADHD data, while Tucker model leads to higher classification accuracy. Meanwhile, our approach is equally applicable to smaller scale imaging dataset under the regularization

strategy given in Section 2.7.

All run times were recorded on a standard laptop computer with a 3.4 GHz Intel i7 CPU.

2.9.3 Limitation

In addition, there are also limitations with the Tucker tensor regression framework in this chapter. The first one is the computability. Although the presence of the core tensor offers Tucker model the flexibility over CP model, it adds more parameters. Especially when all tensor modes have comparable sizes, Tucker model tends to assign identical order for each dimension and results in substantially more parameters compared to CP model. It can be seen in Table 2.1 that when the Tucker ranks go up, the number of parameters for Tucker far exceeds that for CP model. For example a rank 5 model for a 3D signal, Tucker has $5 \times 4 \times 3 = 60$ more parameters than CP; a rank 5 model for a 4D signal such as fMRI, Tucker has $5 \times (5^3 - 4 \times 5 + 4 - 1) = 540$ more parameters than CP. Another difficulty with Tucker model is that we need to specify the model rank along each tensor mode. While it offers the model more flexibility, it causes a combinatorial problem especially for three or higher order tensors. In practice, we only provided some heuristic guidelines of selecting ranks.

Chapter 3

Tensor Regression with Measurement Error

3.1 Introduction

In many areas of statistical analysis, covariate is not directly observable for some reason. Instead, it can only be measured indirectly and imprecisely, which results in data with measurement error. Suppose we have covariate X that for some reason is not directly observable and a substitute measurement W that can be observed instead. Ignoring measurement error and treating W as X in estimation may result in substantial bias (Stefanski and Carroll, 1985). During the past few decades, many studies have been focused on obtaining strategies to correct bias of estimate due to the measurement error. Gleser (1981) proposed a functional model which treats X as nonrandom parameter that needs to be estimated. Based on functional model method, establishing unbiased estimation equation is the strategy in many studies (Stefanski and Carroll, 1985, 1987; Nakamura, 1990). Buzas and Stefanski (1996) proposed estimation equation approach as

an extension of Stefanski and Carroll (1987) at the presence of instrumental variables. In addition, other typical methods include using surrogate variables for X and then fitting models (Carroll and Stefanski, 1990; Gleser et al., 1990), and simulation based method for reducing bias (Cook and Stefanski, 1994; Stefanski and Cook, 1995).

Measurement error problem is prevalent in neuroimaging data as well. In neuroimaging data analysis, a primary goal is to better understand associations between brains and clinical outcomes. Since directly obtaining the brain image of a subject is impossible, neuroimages such as EEG, MRI, PET and fMRI are obtained as measurements for the brain. In many studies multiple brain measurements are available for analysis (Wang et al., 2003; Mitchell et al., 2004; Kamitani and Tong, 2005; Hung and Wang, 2013).

For measurement error problem, most studies so far have investigated the setting that covariates with measurement error are often scalars or very low dimensional vectors. In this chapter we propose a method to deal with multidimensional array covariate with measurement error. The proposed method is an extension of conditional score method (Stefanski and Carroll, 1987) based on CP tensor decomposition. In Section 3.4 we review several typical techniques to handle measurement error model and discuss the possibility of extending these methods to tensor setup. We list advantages and disadvantages for a variety of methods and find it more natural to extend conditional score method.

3.2 Overview

In this chapter, we still focus on generalized linear model (GLM) with tensor covariate. However, we assume the tensor covariate is not directly observable and a substitute measurement can be observed. We adopt a classical measurement error model (ME) where the error of the observable measurement is normally distributed with mean zero. Under

the assumptions of GLM and ME, conditional score method derives unbiased estimation equation of which the solution corrects the bias due to measurement error. However, in the tensor setup, it is formidable to find the root for the estimation equation. Motivated by the conditional score method, we create a loss function and turn the problem to an optimization problem. We impose a CP structure on the array coefficient and develop an efficient algorithm for minimizing the loss. Simulation study demonstrates that estimate obtained by the proposed method successfully corrects the bias.

Section 3.5 reviews the case that X is a vector and extends conditional score equation when covariates with no measurement error are present. We exploit the conditional score method in tensor setup and develop algorithms under linear and logistic regression framework in Section 3.6. In Section 3.7 we show the proposed method outperforms naive estimate obtained by tensor regression (Zhou et al., 2013) in simulation studies.

3.3 Notations

In this section we list the notations throughout this chapter. We introduce models in both tensor setup and classical setup, where covariates are vectors.

3.3.1 Classical Setup

Let Y be the response random variable, Z be the conventional $p_0 \times 1$ vector-valued covariate with no measurement error and X be a $p \times 1$ covariate with measurement error. We denote the observed values of Y, Z and X by y_i, z_i and x_i, $i = 1, \ldots, n$ where n is the sample size. The classical measurement error model states that

$$W = X + U \tag{3.1}$$

where W is an unbiased measurement and U is the error term. As is assumed in most literature, U is distributed as Normal$(\mathbf{0}_p, \mathbf{\Omega}_0)$, where $\mathbf{\Omega}_0$ is called error variance matrix. Assume there are m observed measurements for each $x_i, i = 1, \ldots, n$ and we let $w_{ij}, i = 1, \ldots, n, j = 1, \ldots, m$ be the observed values of these measurements. Also, we denote the sample mean of the m measurements by $\bar{W}$, and the observed values by $\bar{w}_i, i = 1, \ldots, n$.

We adopt the generalized linear model (GLM) on Y given Z and X, which has the form

$$f_{Y|(Z,X)}(y|z,x) = \exp\left\{\frac{y\eta - b(\eta)}{a(\phi)} + c(y,\phi)\right\}$$
$$\eta = \alpha + \gamma^{\mathsf{T}} z + \beta^{\mathsf{T}} x, \tag{3.2}$$

where $a(\cdot)$, $b(\cdot)$ and $c(\cdot)$ are some known functions and α, γ, β and ϕ are parameters.

3.3.2 Tensor Setup

In our setup, the covariate X with measurement error becomes a general multidimensional array (tensor). Assume $X \in \mathbb{R}^{p_1 \times \ldots \times p_D}$, where D is the dimension. Accordingly, we can extend the distribution of U in model (3.1) to an array normal distribution: MN$(\mathbf{0}_{p_1 \times \ldots \times p_D}$, $\mathbf{\Sigma}_1, \ldots, \mathbf{\Sigma}_D)$; $\mathbf{\Sigma}_d$ is a positive definite $p_d \times p_d$ matrix, $d = 1, \ldots, D$ (Hoff, 2011). Also we would use the equivalent vectorized distribution of W, vec$(W) \sim$Normal $(\text{vec}(X), \mathbf{\Omega}_0)$, where $\mathbf{\Omega}_0 = \mathbf{\Sigma}_D \otimes \cdots \otimes \mathbf{\Sigma}_1$, a $\prod_{d=1}^{D} p_d \times \prod_{d=1}^{D} p_d$ matrix.

In the model of Y given Z and X, we change the notation of the coefficient, β, to B. Note that B has the same size of x, i.e. $B \in \mathbb{R}^{p_1 \times \ldots \times p_D}$. And correspondingly, model

(3.2) turns to be

$$f_{Y|(Z,X)}(y|z,x) = \exp\left\{\frac{y\eta - b(\eta)}{a(\phi)} + c(y,\phi)\right\}$$
$$\eta = \alpha + \gamma^{\intercal} z + \langle B, x\rangle, \tag{3.3}$$

where $\langle B, x\rangle$ means $\mathrm{vec}(B)^{\intercal}\mathrm{vec}(x)$.

3.4 Literature Review

In this section we review some classical methods to deal with parametric measurement error models.

3.4.1 Functional Model

- Classical Setup:

  Let us consider the classical setup first. In functional model, the unknown $x_i, i = 1, \ldots, n$, are modeled as nonrandom constants, or parameters to be estimated. Consider the joint log likelihood below:

$$L = \sum_{i=1}^{n} \log\left\{ f_{Y|ZX}(y_i|z_i,x_i) \cdot \prod_{j=1}^{m} \frac{(2\pi)^{-\frac{p}{2}}}{|\mathbf{\Omega}_0|^{\frac{1}{2}}} \exp\big[-\frac{1}{2}(w_{ij}-x_i)^{\intercal}\mathbf{\Omega}_0^{-1}(w_{ij}-x_i)\big]\right\}. \tag{3.4}$$

  The estimate are obtained by maximizing L. When Y given Z and X is normally distributed, Normal(η, σ^2), η given in (3.2), the first order condition with respect

to $(\alpha, \beta, \gamma, \sigma^2, x_1, \ldots, x_n)$ is

$$\begin{aligned}
\frac{\partial L}{\partial x_i} &= \frac{1}{\sigma^2}\left\{(y-\eta_i)\beta - \mathbf{\Omega}_0^{*-1}(x_i - \bar{w}_i)\right\} = \mathbf{0}, i = 1, \ldots, n \\
\frac{\partial L}{\partial \alpha} &= \frac{1}{\sigma^2}\sum_{i=1}^{n}(y-\eta_i) = 0 \\
\frac{\partial L}{\partial \beta} &= \frac{1}{\sigma^2}\sum_{i=1}^{n}(y-\eta_i)x_i = \mathbf{0} \\
\frac{\partial L}{\partial \gamma} &= \frac{1}{\sigma^2}\sum_{i=1}^{n}(y-\eta_i)z_i = \mathbf{0} \\
\frac{\partial L}{\partial \sigma^2} &= \frac{\sum_{i=1}^{n}(y-\eta_i)^2}{2\sigma^4} - \frac{n}{2\sigma^2} = 0
\end{aligned} \tag{3.5}$$

where $\mathbf{\Omega}_0^* = \frac{\mathbf{\Omega}_0}{m\sigma^2}$ is regarded as known following the assumptions in Stefanski and Carroll (1987). One could find that $x_i = (\boldsymbol{I}_p + \mathbf{\Omega}_0^* \beta\beta^\intercal)^{-1}[\bar{w}_i + \mathbf{\Omega}_0^*(y_i - \alpha - \gamma^\intercal z_i)\beta]$ from $\frac{\partial L}{\partial x_i} = \mathbf{0}$, and further get explicit equations with respect to $(\alpha, \beta, \gamma, \sigma^2)$ by canceling x_i. Also it is known that (3.5) results in a consistent estimator of β (Gleser, 1981).

- Advantages:

  It bundles the measurement error model (3.1) into the regression model (3.2), and results in explicit estimation equations for the parameters of interests.

- Limitations:

  The trouble with functional model is that $x_i, i = 1, \ldots, n$ along with the parameters of interest in the GLM are estimated via the maximum likelihood approach. Then the number of parameters to be estimated would be $1 + p_0 + n \times p$, $n \times p$ out of which are due to $x_i, i = 1, \ldots, n$. When p is large, this method would suffer computational problems. Moreover, for models other than the linear regression, consistency may not generally hold for the maximum

likelihood estimate of β. Stefanski and Carroll (1985) gave the example of logistic regression where the consistency fails. The problem is due to the large number of nuisance parameters, $x_i, i = 1, \ldots, n$ (Stefanski and Carroll, 1987).

- Tensor Setup:

  Although this method can be extended to our tensor setup naturally, the computational problem would be much more severe as $n \times \prod_{d=1}^{D} p_d \times \prod_{d=1}^{D} p_d$ nuisance parameters due to $x_i, i = 1, \ldots, n$, are counted in now. In addition, the application is still narrow since consistency for the estimate does not generally hold in models other than linear regression.

Structural Model

- Classical Setup:

  In structural measurement error model, $x_i, i = 1, \ldots, n$, are regarded as random, which is different from the functional model where x_i are modeled as fixed. So models, usually parametric, are placed on the distribution of the random x_is. Suppose there are q parameters in x_i's distribution, where q is a moderate number. Then compared with the functional model, structural model substantially reduces the number of parameters to $1 + p_0 + p + q$.

  – Advantages:

    Compared with functional model, it significantly reduces the number of parameters.

  – Limitations:

    One concern is the robustness of inference to assumptions made about x_i's

distribution. Full specification of a structural measurement error model is not trivial, since we have to impose an appropriate distribution on x, and thus suffer a risk of misspecification.

- Tensor Setup:

  Assigning an appropriate distribution on x is even harder under our high dimensional circumstance, as there are much fewer formalized distribution candidates for a tensor variable x. This issue would consequently cause a higher misspecification risk than the classical setup. Moreover, this method would cause computational burden in our setup.

3.4.2 Regression Calibration

- Classical Setup:

  Applying regression calibration to measurement error models was initially developed by Carroll and Stefanski (1990) and Gleser et al. (1990). This method involves two steps: first obtain an approximation of X, denoted by C, and then replace X by C and perform a model as if there is no measurement error. To get C under model (3.1), Carroll et al. (2006) summarized the best linear approximation using multiple measurements. This approximation was used by Liu and Liang (1992) and Wang et al. (1996) for quasilikelihood models, which include GLM as special cases. The resulting estimated calibration function, C, is

$$C_i = \hat{\mu}_w + \begin{pmatrix} \hat{\Sigma}_{xx} & \hat{\Sigma}_{zz} \end{pmatrix} \begin{pmatrix} \hat{\Sigma}_{xx} + \hat{\mathbf{\Omega}}_0^*/m & \hat{\Sigma}_{xz} \\ \hat{\Sigma}_{xz}^{\intercal} & \hat{\Sigma}_{zz} \end{pmatrix}^{-1} \begin{pmatrix} \bar{w}_i - \hat{\mu}_w \\ z_i - \hat{\mu}_z \end{pmatrix}, i = 1, \ldots, n, \tag{3.6}$$

where

$$\hat{\mathbf{\Omega}}_0^* = \frac{\sum_{i=1}^n \sum_{j=1}^m (w_{ij} - \bar{w}_i)(w_{ij} - \bar{w}_i)^\intercal}{n(m-1)};$$
$$\hat{\mu}_w = \sum_{i=1}^n \sum_{j=1}^m w_{ij}/(mn); \hat{\mu}_z = \sum_{i=1}^n z_i/n;$$
$$\hat{\Sigma}_{zz} = \sum_{i=1}^n (z_i - \hat{\mu}_z)(z_i - \hat{\mu}_z)^\intercal/(n-1);$$
$$\hat{\Sigma}_{xz} = \frac{\sum_{i=1}^n m(\bar{w}_i - \hat{\mu}_w)(z_i - \hat{\mu}_z)^\intercal}{(n-1)m};$$
$$\hat{\Sigma}_{xx} = \frac{\left[\{\sum_{i=1}^n m(\bar{w}_i - \hat{\mu}_w)(\bar{w}_i - \hat{\mu}_w)^\intercal\} - (n-1)\hat{\mathbf{\Omega}}_0^*\right]}{(n-1)m}.$$

  - Advantages:

    It is very convenient that (once Cs are made) the regression model (3.2) can be employed as if X were observed. Compared with functional model and structural model, only the parameters of interests are considered and no nuisance parameter due to X is involved.

  - Limitations:

    The approximation C_i given above involves inverting a $(p_0 + p) \times (p_0 + p)$ matrix, then a heavy computational burden will be accompanied with when p is large. In addition, as Carroll et al. (2006, Chapter 4.7, pp 79-80) stated, regression calibration "is often not appropriate for highly nonlinear problems."

- Tensor Setup:

  Computational issue could be more severe in our setup, because the size of the matrix that needs to be inverted becomes $(p_0 + \prod_{d=1}^D p_d) \times (p_0 + \prod_{d=1}^D p_d)$. Note

that $\hat{\Sigma}_{xx} + \hat{\mathbf{\Omega}}_0^*/m$, the upper left $\prod_{d=1}^{D} p_d \times \prod_{d=1}^{D} p_d$ block of the matrix in (3.6), does not have low dimensional structure as $\hat{\mathbf{\Omega}}_0^*$ has, therefore inverting such a huge matrix is a formidable task.

3.4.3 SIMEX

- Classical Setup:

  SIMEX is a simulation-based method of estimating and reducing bias due to measurement error. The technique was proposed by Cook and Stefanski (1994) and further developed by Stefanski and Cook (1995). Let us take simple linear regression ($p_0 = 0, p = 1$) as an example to demonstrate this method.

  Assume there is only one observed measurement ($m = 1$) denoted by $w_{i1}, i = 1, \ldots, n$. We follow the steps below

  1. Fix a series of $K + 1$ numbers: $0 = \lambda_0 < \lambda_1 < \cdots < \lambda_K \leq 2$.
  2. For each λ_k, generate S error inflated data sets $(y_i, w_{i1} + \sqrt{\lambda_k}\mathbf{\Omega}_0^{*\frac{1}{2}} v_{is})_{i=1}^n, s = 1, \ldots, S$, where v_{is} are independently generated from Normal(0,1). Fit linear regression as if there is no measurement error for each data set and obtain $\hat{\beta}_s(\lambda_k)$. Let $\hat{\beta}(\lambda_k) = \frac{1}{S}\sum_{s=1}^{S} \hat{\beta}_s(\lambda_k)$.
  3. Fit a curve for $(\lambda_k, \hat{\beta}(\lambda_k))_{k=1}^K$.
  4. Extrapolate the curve back to $\lambda = -1$. The corresponding $\hat{\beta}(-1)$ is the estimate we need.

  The logic of SIMEX is adding a small quantity to the measurement and thus adjust the error variance. Then an adjusted estimator corresponding to the small quantity is obtained. Repeat this procedure many times for different small quantity. One can

establish a trend of the small quantities versus the corresponding estimators and extrapolate the trend to the estimator with zero error variance.

  - Advantages:

    This method is completely general, which means it has no restrictions on the regression model of Y given X and Z. It is useful in applications since GLM in (3.2) is not required.

  - Limitations:

    First, this method is simulation based and there is no uniform criterion or theory suggesting the choice of "small quantity". Second is about the extrapolant function. It follows from the results in Stefanski and Cook (1995) that, asymptotically there is a function of λ that, when extrapolated to $\lambda = -1$, the true parameter β is obtained. However, this extrapolant function is seldom known. This is what makes SIMEX an approximate method in practice (Carroll et al., 2006). This problem could be more severe when β's dimension p is large. Third, in order to achieve acceptable Monte Carlo estimation precision, the number of remeasured data sets, S, is required to be large compared to β's dimension p. Therefore in the context of high dimensionality of β, SIMEX requires tremendous computational effort for comparable levels of precision.

- Tensor Setup

  Actually limitations discussed above also exist in our tensor setup. In our context of ultra-high dimensionality, identifying the extrapolant function becomes much more difficult. Also, S and K in step 1, 2 are required to be very large due to B's high dimension. This makes the computation extremely expensive.

3.4.4 Conditional Score

- Classical Setup:

  Stefanski and Carroll (1987) proposed the conditional score method, which gives explicit estimation equations. Similar to the functional model, conditional score method models x_i as nonrandom constants. But instead of estimating x_i by maximizing the full log likelihood, it derives 'sufficient' statistics for x_i supposing the other parameters are fixed. Conditioning on the 'sufficient' statistics, Stefanski and Carroll (1987) provided explicit unbiased estimation equations and established a firm theoretical support for GLM models. Conditional score method given by Stefanski and Carroll (1987) is only for one measurement case ($m = 1$). We consider the log likelihood given in (3.4). If $m = 1$, we use $w_{i1}, i = 1, \ldots, n$ as the only observed measurement for x_i to keep the notation consistent. Treating other parameters as fixed, one can derive the 'sufficient' statistics Δ for X, where $\Delta = W + \frac{Y\mathbf{\Omega}_0\beta}{a(\phi)}$. Stefanski and Carroll (1987) assumed the ratio $\mathbf{\Omega}_0/a(\phi)$ to be known. Then the observed value of Δ, denoted by δ_i, has the relation $\delta_i = w_{i1} + \frac{y_i\mathbf{\Omega}_0\beta}{a(\phi)}, i = 1, \ldots, n$. Consequently the distribution of Y given Δ is free of X. From the conditional distribution of Y given Δ, one can derive score equations with respect to β, which leads to the unbiased estimation equations.

  When there are multiple measurements ($m > 1$), Stefanski and Carroll (1987) suggested to use the sample mean, $\bar{W}$, to be the one measurement and turn it to $m = 1$ case. Note that $\bar{W}$ is Normal$(X, \mathbf{\Omega}_0/m)$. Thus L in (3.4) turns to be $\sum_{i=1}^{n} \log \left\{ f_{Y|ZX}(y_i|z_i, x_i) \cdot \frac{(2\pi)^{-\frac{p}{2}}}{|\mathbf{\Omega}_{0/m}|^{\frac{1}{2}}} \exp\left[-\frac{1}{2}(\bar{w}_i - x_i)^{\intercal}(\mathbf{\Omega}_0/m)^{-1}(\bar{w}_i - x_i)\right] \right\}$. Correspondingly, Δ should be modified to $\bar{W} + \frac{Y\mathbf{\Omega}_0\beta}{ma(\phi)}$. Then we use the score equations derived from the conditional distribution Y given Δ.

Conditional score estimator aims to reduce the bias and applies generally to parametric measurement error models. When it comes to the linear model, conditional score estimation equations are the same as the score equations to maximize the full log likelihood given in (3.4).

  - Advantages:

    Compared with other methods, conditional score method has both firm theoretical support and explicit estimation equations. More importantly, only parameters of interest $(\alpha, \beta, \gamma, \phi)$ in regression model (3.2) are involved in the estimation equations and no nuisance parameter due to x_i exists. This feature not only leads to consistent estimators of β but makes computation feasible as well.

- Tensor Setup

  Conditional score method can be extended to our setup easily. Although B is very high dimensional, the computation can be moderate if tensor structure is applied.

3.4.5 Instrumental Variables

- Classical Setup:

  Sometimes data contain an instrumental variable T , in addition to the unbiased measurement, $W = X + U$. Instrumental variables serve as complement variables in the absence of information about the measurement error variance $\mathbf{\Omega}_0$ (Carroll et al., 2006). Carroll et al. (2006) stated an instrument variable T must possesses three key properties: (i) T must not be independent of X; (ii) T must be uncorrelated with the measurement error U; (iii) T must be uncorrelated with Y - $E(Y|X, Z)$. Falsely

assuming a variable is an instrument can lead to erroneous inferences even in the case of large sample size and small measurement error (Carroll et al., 2006). Note that the assumption that a variable is an instrument is weaker than the assumption that it is a replicate measurement.

Instrumental variables are supposed to be taken into account when they are available. Stefanski and Buzas (1995) and Buzas and Stefanski (1996) investigated the instrumental variable estimation method for generalized linear models. For one observed measurement case ($m = 1$), they assumed a $q \times 1$ instrumental variable T, that is related to (Z, X) via the distribution,

$$T|(Z, X) \sim \text{Normal}(\zeta, \Sigma_\zeta),$$

where $\zeta = \gamma_z z + \gamma_x x$. Note that γ_z and γ_x are $q \times p_0$ and $q \times p$ matrices. Then they essentially used similar technique to conditional score method. They obtained the joint distribution of (Y, W, T) given Z and X and constructed 'sufficient' statistics Υ for X supposing all other parameters are fixed and known, where $\Upsilon = W + Y\mathbf{\Omega}_0\beta/a(\phi) + \mathbf{\Omega}_0\gamma_x^\intercal\Sigma_\zeta^{-1}T$. The conditional distribution of Y given Υ does not depend on X. As in work of Stefanski and Carroll (1987), this conditional distribution can be used to establish explicit unbiased estimation equations for $(\alpha, \gamma, \beta, \phi)$ that are free of X.

The context where there are multiple independent measurements is stronger than the assumptions for general instrumental variables or those given in Stefanski and Buzas (1995) and Buzas and Stefanski (1996). Therefore it is safer to use the replicate measurements as the source of instrumental variables and find out the 'sufficient' statistics for x_i. Hence, we can see the only difference from conditional

score method is that instrumental variable method uses all measurements while conditional score method only uses the sample mean of the measurements. After some algebra, the 'sufficient' statistics obtained from instrumental variable method is $\bar{W}+\frac{Y\mathbf{\Omega}_0\beta}{ma(\phi)}$, which is the same as Δ, the 'sufficient' statistics from conditional score method. This result is not surprising because that measurements follow a normal distribution, and thus sample mean is the complete sufficient statistic for x_i in the joint likelihood of $w_{ij}, j = 1, \ldots, m$. Therefore, essentially instrumental variable method that uses all the measurements obtains the same results as the conditional score method where only the sample mean of the measurements is used.

  - Advantages:

    Similar to the conditional score method, instrumental variable method leads to explicit estimation equations and no nuisance parameter due to X is involved.

  - Limitations:

    As we can see, the condition for an eligible instrumental variable is strict. Sometimes instrumental variables may not be available in applications.

- Tensor Setup:

  Since we assume multiple measurements ($m > 1$) throughout the paper, instrumental variable method is equivalent to the conditional score method either in the classical setup or our tensor setup. The analysis in the classical setup also applies in tensor setup.

3.5 Model – Vector X

We first consider the case that $\boldsymbol{X}$ is a vector, which lays a foundation for tensor setup. Section 3.5.1 reviews conditional score method proposed in Stefanski and Carroll (1987), which considered the case that only covariates with measurement error were present. In the situations where covariates with no measurement error such as age and gender are involved in the model, Section 3.5.2 extends the conditional score equation in Stefanski and Carroll (1987).

3.5.1 Review of Conditional Score Method

We first work on the conditional score method given by Stefanski and Carroll (1987) in the classical setup where X is a $p \times 1$ vector. They mainly discussed one measurement case ($m = 1$). If $m > 1$, we use the sample mean of the m measurements, $\bar{W}$ as if it is the only measurement. The distribution of $\bar{W}$ is Normal$(X, \boldsymbol{\Omega}_0/m)$. We follow the assumption in Stefanski and Carroll (1987) that the ratio of $\boldsymbol{\Omega}_0/m$ and $a(\phi)$ is known, denoted by $\boldsymbol{\Omega}_0^*$. This assumption ensures identifiability of the parameters in the model (3.1) and (3.2) (Stefanski and Carroll, 1987).

When $p_0 = 0$, which means there is no Z or γ in the model, the problem is the same as that given in Stefanski and Carroll (1987). If X is regarded as an unknown parameter and all other parameters are supposed to be known, then

$$\Delta = \bar{W} + Y\boldsymbol{\Omega}_0^*\beta \tag{3.7}$$

is a sufficient statistic for X (Stefanski and Carroll, 1987). Furthermore, the conditional distribution of Y given Δ, $f_{Y|\Delta}$, can be found and the score function ψ_s is defined as the

first derivative of $\log f_{Y|\Delta}$ with respect to (α, β, ϕ) evaluated at $\delta = \bar{w} + y\boldsymbol{\Omega}_0^*\beta$. Noticing that δ is treated as a fixed conditioning argument until (3.5.1), wherein δ is set equal to $\delta = \bar{w} + y\boldsymbol{\Omega}_0^*\beta$, we have ψ_s expressed as

$$\psi_s(y, \bar{w}, \alpha, \beta, \phi) = \left(\frac{\partial \log f_{Y|\Delta}}{\partial \alpha}, \frac{\partial \log f_{Y|\Delta}}{\partial \beta}, \frac{\partial \log f_{Y|\Delta}}{\partial \phi} \right)^{\mathsf{T}}_{\delta=\bar{w}+y\boldsymbol{\Omega}_0^*\beta}$$

$$= \begin{pmatrix} \{y - E(Y|\Delta = \delta)\}/a(\phi) \\ \{y - E(Y|\Delta = \delta)\}\delta/a(\phi) - \{y^2 - E(Y^2|\Delta = \delta)\}\boldsymbol{\Omega}_0^*\beta/a(\phi) \\ r(y, \bar{w}, \alpha, \beta, \phi) - E\{r(Y, \bar{W}, \alpha, \beta, \phi)|\Delta = \delta\} \end{pmatrix}_{\delta=\bar{w}+y\boldsymbol{\Omega}_0^*\beta}, \tag{3.8}$$

where

$$r(y, \bar{w}, \alpha, \beta, \phi) = \frac{\partial c(y, \phi)}{\partial \phi} - y\frac{\alpha + \beta^{\mathsf{T}}\delta}{a^2(\phi)}a'(\phi) + y^2\frac{\beta^{\mathsf{T}}\boldsymbol{\Omega}_0^*\beta}{2a^2(\phi)}a'(\phi).$$

We can see X is no longer involved in ψ_s. Also ψ_s is unbiased for (α, β, ϕ) (Stefanski and Carroll, 1987). The estimates of (α, β, ϕ) are obtained by solving

$$\sum_{i=1}^{n} \psi_s(y_i, \bar{w}_i, \alpha, \beta, \phi) = 0. \tag{3.9}$$

Dealing with (3.9) is illustrated by studying it in some particular GLM such as linear, logistic and Poisson regression in Stefanski and Carroll (1987). We only review the cases of linear and logistic regression in this chapter.

Suppose first that Y given X has a normal distribution with mean $\alpha + \beta^{\mathsf{T}}X$ and variance σ^2. Then in model (3.2) we have $\phi = \sigma^2, a(\phi) = \phi$. According to Stefanski and

Carroll (1987), the distribution of Y given $\Delta = \delta$ is also normal with mean $(\alpha+\beta^\intercal\delta)/(1+\beta^\intercal\boldsymbol{\Omega}_0^*\beta)$ and variance $\sigma^2/(1+\beta^\intercal\boldsymbol{\Omega}_0^*\beta)$. Based on the conditional distribution of Y given $\Delta = \delta$, $E(Y|\Delta = \delta)$ and $E(Y^2|\Delta = \delta)$ can be easily found and plugged into (3.5.1), which leads the estimation equations (3.9) to

$$\sum_{i=1}^{n}(y_i - \alpha - \beta^\intercal\delta_i^*)\begin{pmatrix} 1 \\ \delta_i^* \end{pmatrix} = \mathbf{0}$$
$$\sigma^2 = \frac{1+\beta^\intercal\boldsymbol{\Omega}_0^*\beta}{n}\sum_{i=1}^{n}(y_i - \frac{\alpha+\beta^\intercal\delta_i}{1+\beta^\intercal\boldsymbol{\Omega}_0^*\beta})^2,$$

where $\delta_i^* = (\boldsymbol{I}+\boldsymbol{\Omega}_0^*\beta\beta^\intercal)^{-1}\{\delta_i - \alpha\boldsymbol{\Omega}_0^*\beta\}$.

Now consider logistic regression in which $\mathrm{P}(Y=1|X) = F(\alpha+\beta^\intercal X)$, where $F(t) = 1/(1+e^{-t})$. For this model $a(\phi)\equiv 1$, and thus ϕ does not need to be estimated and the last term in (3.5.1) disappears. According to Stefanski and Carroll (1987), the distribution of Y given $\Delta = \delta$ is $\mathrm{P}(Y=1|\Delta=\delta) = F(\alpha+\beta^\intercal(\delta - \frac{1}{2}\boldsymbol{\Omega}_0^*\beta))$. After plugging $E(Y|\Delta=\delta)$ and $E(Y^2|\Delta=\delta)$, one could figure out the equivalent equations corresponding to (3.9)

$$\sum_{i=1}^{n}(y_i - F\{\alpha+\beta^\intercal[\delta_i - \frac{1}{2}\boldsymbol{\Omega}_0^*\beta]\})\begin{pmatrix} 1 \\ \delta_i - \frac{1}{2}\boldsymbol{\Omega}_0^*\beta \end{pmatrix} = \mathbf{0}.$$

3.5.2 Extension of Conditional Score Equation

In this subsection, we are considering a more complicated case where $p_0 > 0$. We still use Δ given in (3.7) as the 'sufficient' statistic for X if $(\alpha, \beta, \gamma, \phi)$ are viewed as known. Then one can find the conditional distribution of Y given Δ and Z

$$f_{Y|(Z,\Delta)}(y|z,\delta) = \exp\left\{y\mu - \log[S(\mu,\beta,\phi)] + c(y,\phi) - \frac{1}{2}y^2\frac{\beta^\intercal\boldsymbol{\Omega}_0^*\beta}{a(\phi)}\right\}, \qquad (3.10)$$

where $\mu = (\alpha + \gamma^\intercal z + \beta^\intercal \delta)/a(\phi)$ and $S(\cdot, \cdot, \cdot)$ is

$$S(\mu, \beta, \phi) = \int \exp\left\{ y\mu + c(y, \phi) - \frac{1}{2} y^2 \frac{\beta^\intercal \mathbf{\Omega}_0^* \beta}{a(\phi)} \right\} dy.$$

The derivation of (3.10) can be find in Appendix B.1. We still use ψ_s to denote the score function, which is the first derivative of $\log f_{Y|(Z,\Delta)}$ with respect to $(\alpha, \beta, \gamma, \phi)$ evaluated at $\delta = \bar{w} + y\mathbf{\Omega}_0^* \beta$. Then we have

$$\psi_s(y, z, \bar{w}, \alpha, \beta, \gamma, \phi) = \left(\frac{\partial \log f_{Y|(Z,\Delta)}}{\partial \alpha}, \frac{\partial \log f_{Y|(Z,\Delta)}}{\partial \gamma}, \frac{\partial \log f_{Y|(Z,\Delta)}}{\partial \beta}, \frac{\partial \log f_{Y|(Z,\Delta)}}{\partial \phi} \right)^\intercal_{\delta = \bar{w} + y\mathbf{\Omega}_0^* \beta}$$

$$= \begin{pmatrix} \{y - E(Y|Z = z, \Delta = \delta)\}/a(\phi) \\ \{y - E(Y|Z = z, \Delta = \delta)\} \cdot z/a(\phi) \\ \{y - E(Y|Z = z, \Delta = \delta)\}\delta/a(\phi) - \{y^2 - E(Y^2|Z = z, \Delta = \delta)\}\mathbf{\Omega}_0^* \beta/a(\phi) \\ r(y, z, \bar{w}, \alpha, \beta, \gamma, \phi) - E\{r(Y, Z, \bar{W}, \alpha, \beta, \gamma, \phi)|Z = z, \Delta = \delta\} \end{pmatrix}, \tag{3.11}$$

where

$$r(y, z, \bar{w}, \alpha, \beta, \gamma, \phi) = \frac{\partial c(y, \phi)}{\partial \phi} - y \frac{\alpha + \gamma^\intercal z + \beta^\intercal \delta}{a^2(\phi)} a'(\phi) + y^2 \frac{\beta^\intercal \mathbf{\Omega}_0^* \beta}{2a^2(\phi)} a'(\phi).$$

Note that (3.10) is an exponential family and thus $E(Y|Z = z, \Delta = \delta)$ and $E(Y^2|Z = z, \Delta = \delta)$ can be obtained based on (3.10). Carroll et al. (2006) gave another but essentially equivalent form of (3.5.2). Furthermore, the resulting estimation equation is

$$\sum_{i=1}^{n} \psi_s(y_i, z_i, \bar{w}_i, \alpha, \beta, \gamma, \phi) = 0. \tag{3.12}$$

We also use linear and logistic regression to illustrate (3.5.2) and (3.12).

First suppose the distribution of Y given X and Z is normal with mean $\alpha+\beta^\intercal X+\gamma^\intercal Z$ and variance σ^2. Then in model (3.2) we have $\phi=\sigma^2, a(\phi)=\phi$. Then from (3.10) one can find that Y given Δ and Z, where Δ is given in (3.7), is normal distributed with mean $\frac{\alpha+\gamma^\intercal z+\beta^\intercal\delta}{1+\beta^\intercal\boldsymbol{\Omega}_0^*\beta}$ and variance $\frac{\sigma^2}{1+\beta^\intercal\boldsymbol{\Omega}_0^*\beta}$. From (3.5.2), we obtain the score functions

$$\psi_s(y,z,\bar{w},\alpha,\beta,\gamma,\sigma^2) \;=\; \left(\begin{array}{c} (y-\xi)/\sigma^2 \\ (y-\xi)\cdot z/\sigma^2 \\ \frac{\boldsymbol{\Omega}_0^*\beta}{s} - \frac{(y-\xi)^2\boldsymbol{\Omega}_0^*\beta-(y-\xi)(\delta-2\xi\boldsymbol{\Omega}_0^*\beta)}{\sigma^2} \\ \frac{-1}{2\sigma^2}+\frac{(y-\xi)^2 s}{2\sigma^4} \end{array}\right)_{\delta=\bar{w}+y\boldsymbol{\Omega}_0^*\beta}, \quad (3.13)$$

where $\xi=\frac{\alpha+\gamma^\intercal z+\beta^\intercal\delta}{1+\beta^\intercal\boldsymbol{\Omega}_0^*\beta}$ and $s=1+\beta^\intercal\boldsymbol{\Omega}_0^*\beta$. Define $\delta^*=(\boldsymbol{I}+\boldsymbol{\Omega}_0^*\beta\beta^\intercal)^{-1}\{\delta-(\alpha+\gamma^\intercal z)\boldsymbol{\Omega}_0^*\beta\}$, where $\delta=\bar{w}+y\boldsymbol{\Omega}_0^*\beta$, and consider the equations

$$\sum_{i=1}^n(y_i-\alpha-\gamma^\intercal z_i-\beta^\intercal\delta_i^*)\left(\begin{array}{c}1\\ z_i\\ \delta_i^*\end{array}\right)=\mathbf{0}$$

$$\sigma^2-\frac{s}{n}\sum_{i=1}^n(y_i-\xi_i)^2=0 \quad (3.14)$$

Every solution to (3.14) is also a solution to $\sum_{i=1}^n\psi_s(y_i,z_i,\bar{w}_i,\alpha,\beta,\gamma,\sigma^2)=0$, where ψ_s is given by (3.13).

We let $\boldsymbol{M}_{\mathbf{1}Z}$ be the $n\times(p_0+1)$ design matrix for the intercept and the covariate with no measurement error, Z, and $\boldsymbol{M}_{\bar{W}}$ be the $n\times p$ design matrix for the $\bar{W}$. Then (3.14)

results in the following equations

$$(\boldsymbol{M}_{\mathbf{1}Z}^{\intercal}\boldsymbol{M}_{\mathbf{1}Z})(\alpha\ \gamma^{\intercal})^{\intercal} = \boldsymbol{M}_{\mathbf{1}Z}^{\intercal}(\boldsymbol{y} - \boldsymbol{M}_{\bar{W}}\beta), \boldsymbol{y} = (y_1, \ldots, y_n)^{\intercal} \tag{3.15}$$

$$-\beta^{\intercal}\sum_{i=1}^{n}\{(\bar{w}_i - \tilde{w}_i)y_i\}\,\boldsymbol{\Omega}_0^*\beta +$$

$$\sum_{i=1}^{n}\{(y_i - \tilde{y}_i)y_i\boldsymbol{\Omega}_0^* - \bar{w}_i(\bar{w}_i^{\intercal} - \tilde{w}_i^{\intercal})\}\,\beta + \sum_{i=1}^{n}\{(y_i - \tilde{y}_i)\bar{w}_i\} = \mathbf{0} \tag{3.16}$$

$$\sigma^2 - \frac{s}{n}\sum_{i=1}^{n}(y_i - \xi_i)^2 = 0, \tag{3.17}$$

where $\tilde{y}_i = (1\ z_i^{\intercal})(\boldsymbol{M}_{\mathbf{1}Z}^{\intercal}\boldsymbol{M}_{\mathbf{1}Z})^{-1}\boldsymbol{M}_{\mathbf{1}Z}^{\intercal}\boldsymbol{y}$; $\tilde{w}_i = ((1\ z_i^{\intercal})(\boldsymbol{M}_{\mathbf{1}Z}^{\intercal}\boldsymbol{M}_{\mathbf{1}Z})^{-1}\boldsymbol{M}_{\mathbf{1}Z}^{\intercal}\boldsymbol{M}_{\bar{W}})^{\intercal}$ is a $p \times 1$ vector. Derivations from (3.13) to (3.14)-(3.17) can be found in Appendix B.2. To solve equations (3.15)-(3.17), one could first solve β in (3.16), which is free of α, γ and σ^2, and then solve α, γ in (3.15) and σ^2 in (3.17).

Now consider logistic regression in which

$$P(Y = 1|X, Z) = \frac{\exp(\alpha + \gamma^{\intercal}z + \beta^{\intercal}x)}{1 + \exp(\alpha + \gamma^{\intercal}z + \beta^{\intercal}x)}.$$

Then in model (3.2) we have $a(\phi) \equiv 1$. From (3.10) one can find out the probability mass of Y given $\Delta = \delta$, where Δ is given in (3.5.1), is

$$P(Y = 1|\Delta = \delta) = \frac{\exp[\alpha + \gamma^{\intercal}z + \beta^{\intercal}(\delta - \boldsymbol{\Omega}_0^*\beta/2)]}{1 + \exp[\alpha + \gamma^{\intercal}z + \beta^{\intercal}(\delta - \boldsymbol{\Omega}_0^*\beta/2)]}.$$

Therefore the conditional score can be obtained from (3.5.2)

$$\psi_s(y, z, \bar{w}, \alpha, \beta, \gamma) = \{y - F([\alpha + \gamma^\intercal z + \beta^\intercal(\delta - \mathbf{\Omega}_0^*\beta/2)])\} \begin{pmatrix} 1 \\ z \\ \delta - \mathbf{\Omega}_0^*\beta \end{pmatrix}_{\delta=\bar{w}+y\beta\mathbf{\Omega}_0^*} \tag{3.18}$$

where $F(t) = 1/(1 + e^{-t})$. Therefore the estimation equations corresponding to (3.12) is

$$\sum_{i=1}^{n} \{y_i - F([\alpha + \gamma^\intercal z_i + \beta^\intercal(\delta_i - \mathbf{\Omega}_0^*\beta/2)])\} \begin{pmatrix} 1 \\ z_i \\ \delta_i - \mathbf{\Omega}_0^*\beta \end{pmatrix} = \mathbf{0} \tag{3.19}$$

3.5.3 Conversion to Optimization Problem

Conditional score method typically estimates $(\alpha, \beta, \gamma, \phi)$ by solving (3.12). Note that the left hand side of (3.12) is a $(2 + p_0 + p) \times 1$ vector. Thus when p is large, finding roots for (3.12) could be difficult. Instead, we turn to consider a one step generalized method of moments estimate (GMM) (Hansen, 1982), that is, we estimate $(\alpha, \beta, \gamma, \phi)$ by minimizing

$$h(\alpha, \beta, \gamma, \phi) = \| \sum_{i=1}^{n} \psi_s(y_i, z_i, \bar{w}_i, \alpha, \beta, \gamma, \phi) \|_2. \tag{3.20}$$

Problem (3.20) could still be formidable under the general GLM setup (3.2). Following the context above, we illustrate this optimization strategy in linear regression and logistic regression.

For linear regression, the estimate of $\alpha, \beta, \gamma, \sigma^2$ can be obtained by solving (3.15) - (3.17). Note that β can be solved in (3.16), which is free of α, γ and σ^2. Hence, we can

just consider the following optimization problem corresponding to (3.16)

$$h_1(\beta) = \| - \beta^{\mathsf{T}} \sum_{i=1}^{n} \{(\bar{w}_i - \tilde{w}_i) y_i\} \, \mathbf{\Omega}_0^* \beta + \sum_{i=1}^{n} \{(y_i - \tilde{y}_i) y_i \mathbf{\Omega}_0^* - \bar{w}_i (\bar{w}_i^{\mathsf{T}} - \tilde{w}_i^{\mathsf{T}})\} \, \beta + \sum_{i=1}^{n} \{(y_i - \tilde{y}_i) \bar{w}_i\} \, \|_2. \quad (3.21)$$

After β is solved, α, γ and ϕ can be solved in (3.15) and (3.17).

For logistic regression, α, β, γ have to be solved together. The corresponding problem to (3.19) is minimizing

$$h_2(\alpha, \beta, \gamma) = \| \sum_{i=1}^{n} \psi_s(y_i, z_i, \bar{w}_i, \alpha, \beta, \gamma) \|_2,$$

where ψ_s is given in (3.18).

3.6 Model – Tensor X

We only consider the case that $p_0 > 0$. In this Section, X, W and the corresponding coefficient B are $p_1 \times \ldots \times p_D$ tensors. We use the notations and assumptions in Section 3.3.2 and consider GLM (3.3). In tensor setup, we follow the optimization strategy developed in Section 3.5.3. To accommodate tensor setup, one can simply turn W and B into vectors and replace W and β in (3.20) respectively. The resulting problem is minimizing

$$h(\alpha, \mathrm{vec}(B), \gamma, \phi) = \| \sum_{i=1}^{n} \psi_s(y_i, z_i, \mathrm{vec}(\bar{w}_i), \alpha, \mathrm{vec}(B), \gamma, \phi) \|_2 \quad (3.22)$$

The biggest issue of optimizing (3.22) is the ultra high dimensionality of B, which has $\prod_{d=1}^{D} p_d$ parameters. This number usually far exceeds the usual sample size and makes

Algorithm 2 Block relaxation algorithm for minimizing $h(\alpha, \text{vec}(B), \gamma, \phi)$.

Initialize: $B_d^{(0)} \in \mathbb{R}^{p_d \times R}$ a random matrix for $d = 1, \ldots, D$; ($\alpha^{(0)}$, $\gamma^{(0)}$) is the linear estimate given generated $B^{(0)}$.
repeat
  for $d = 1, \ldots, D$ **do**
    $B_d^{(t+1)} = \text{argmin}_{B_d} h(\alpha^{(t)}, \text{vec}(B_d), \gamma^{(t)}, \phi^{(t)} | B_1^{(t+1)}, \ldots, B_{d-1}^{(t+1)}, B_{d+1}^{(t)}, \ldots, B_D^{(t)})$
  end for
  $(\alpha^{(t+1)}, \gamma^{(t+1)}) = \text{argmin}_{\alpha,\gamma} h(\alpha, \text{vec}(\llbracket B_1^{(t+1)}, \ldots, B_D^{(t+1)} \rrbracket), \gamma, \phi^{(t)})$
  $\phi^{(t+1)} = \text{argmin}_{\phi} h(\alpha^{(t+1)}, \text{vec}(\llbracket B_1^{(t+1)}, \ldots, B_D^{(t+1)} \rrbracket), \gamma^{(t+1)}, \phi)$
until
$h(\alpha^{(t)}, \text{vec}(\llbracket B_1^{(t)}, \ldots, B_D^{(t)} \rrbracket), \gamma^{(t)}, \phi^{(t)}) -$
$h(\alpha^{(t+1)}, \text{vec}(\llbracket B_1^{(t+1)}, \ldots, B_D^{(t+1)} \rrbracket), \gamma^{(t+1)}, \phi^{(t+1)}) < \epsilon$.

the estimation inefficient.

To address this issue, we impose a rank-R CP decomposition on $B \in \mathbb{R}^{p_1 \times \cdots \times p_D}$:

$$B = \sum_{r=1}^{R} \beta_1^{(r)} \circ \cdots \circ \beta_D^{(r)},$$

where $\beta_d^{(r)} \in \mathbb{R}^{p_d}, d = 1, \ldots, D, r = 1, \ldots, R$, and the shorthand $B = \llbracket B_1, \ldots, B_D \rrbracket$ where $B_d = [\beta_d^{(1)}, \ldots, \beta_d^{(R)}] \in \mathbb{R}^{p_d \times R}$, $d = 1, \ldots, D$. We adopt a block relaxation algorithm for minimizing $h(\alpha, \text{vec}(B), \gamma, \phi)$ in (3.22). Algorithm 2 gives the general framework. Again, we provide estimation procedures for linear regression and logistic regression.

3.6.1 Optimization – Linear Regression

When X, W and the corresponding coefficient B are $p_1 \times \cdots \times p_D$ tensors, we convert them into $(\prod_{d=1}^{D} p_d) \times 1$ vectors. The size of $\boldsymbol{M}_{\bar{W}}$ becomes $n \times (\prod_{d=1}^{D} p_d)$ and the length of $\tilde{w}_i$ becomes $\prod_{d=1}^{D} p_d$. Then (3.15) becomes

$$(\boldsymbol{M}_{\mathbf{1}Z}^{\mathsf{T}} \boldsymbol{M}_{\mathbf{1}Z})(\alpha\ \gamma^{\mathsf{T}})^{\mathsf{T}} = \boldsymbol{M}_{\mathbf{1}Z}^{\mathsf{T}} (\boldsymbol{y} - \boldsymbol{M}_{\bar{W}} \text{vec}(B)), \tag{3.23}$$

and $h_1(\cdot)$ in (3.21) becomes

$$\begin{aligned} h_1(B) &= \| - \mathrm{vec}(B)^\intercal \sum_{i=1}^{n} \{(\mathrm{vec}(\bar{w}_i) - \mathrm{vec}(\tilde{w}_i))y_i\} \boldsymbol{\Omega}_0^* \mathrm{vec}(B) + \\ &\quad \sum_{i=1}^{n} \{(y_i - \tilde{y}_i)y_i \boldsymbol{\Omega}_0^* - \mathrm{vec}(\bar{w}_i)(\mathrm{vec}(\bar{w}_i)^\intercal - \mathrm{vec}(\tilde{w}_i)^\intercal)\} \mathrm{vec}(B) + \\ &\quad \sum_{i=1}^{n} \{(y_i - \tilde{y}_i)\mathrm{vec}(\bar{w}_i)\} \|_2. \end{aligned} \tag{3.24}$$

Define

$$\begin{aligned} a_1 &= \sum_{i=1}^{n} \{(\mathrm{vec}(\bar{w}_i) - \mathrm{vec}(\tilde{w}_i))y_i\} \\ a_2 &= \boldsymbol{\Omega}_0^* \\ a_3 &= \sum_{i=1}^{n} \{(y_i - \tilde{y}_i)y_i \boldsymbol{\Omega}_0^* - \mathrm{vec}(\bar{w}_i)(\mathrm{vec}(\bar{w}_i)^\intercal - \mathrm{vec}(\tilde{w}_i)^\intercal)\} \\ a_4 &= \sum_{i=1}^{n} \{(y_i - \tilde{y}_i)\mathrm{vec}(\bar{w}_i)\}. \end{aligned}$$

Note that a_1, a_4 are known $\prod_{d=1}^{D} p_d \times 1$ vectors and a_2, a_3 are known $\prod_{d=1}^{D} p_d \times \prod_{d=1}^{D} p_d$ matrices. Minimizing (3.24) is simplified to minimizing

$$h_1(B|a_1, a_2, a_3, a_4) = \| - \mathrm{vec}(B)^\intercal a_1 a_2 \mathrm{vec}(B) + a_3 \mathrm{vec}(B) + a_4 \|_2. \tag{3.25}$$

Recall that a CP decomposition has been applied to B. We utilize Algorithm 3 to minimize (3.25) and update $B_d, d = 1, \ldots, D$ cyclically.

Define $B_{(d)}, d = 1, \ldots D$ as the mode-d matricization of tensor B (Kolda and Bader, 2009). Let $\boldsymbol{\Pi}_d$ be the $\Pi_{d=1}^{D} p_d \times \Pi_{d=1}^{D} p_d$ permutation matrix that reorders $\mathrm{vec}(B_{(d)})$ to

Algorithm 3 Block relaxation algorithm for minimizing $h_1(B)$ in (3.24)

Initialize: $B_d^{(0)} \in \mathbb{R}^{p_d \times R}$ a random matrix for $d = 1, \ldots, D$;
repeat
  for $d = 1, \ldots, D$ **do**
    $B_d^{(t+1)} = \text{argmin}_{B_d} h_1(B_d | B_1^{(t+1)}, \ldots, B_{d-1}^{(t+1)}, B_{d+1}^{(t)}, \ldots, B_D^{(t)})$
  end for
until $h_1(B^{(t)}) - h_1(B^{(t+1)}) < \epsilon$.

obtain $\text{vec}(B)$, i.e., $\text{vec}(B) = \mathbf{\Pi}_d \text{vec}(B_{(d)})$. Then

$$\begin{aligned}
\text{vec}(B) &= \mathbf{\Pi}_d \text{vec}(B_{(d)}) \\
&= \mathbf{\Pi}_d \text{vec}\left[B_d (B_D \odot \cdots \odot B_{d+1} \odot B_{d-1} \odot \cdots \odot B_1)^\intercal\right] \\
&= \mathbf{\Pi}_d \left[(B_D \odot \cdots \odot B_{d+1} \odot B_{d-1} \odot \cdots \odot B_1) \otimes \boldsymbol{I}_{p_d}\right] \text{vec}(B_d),
\end{aligned}$$

and thus B_d can be updated by $B_D, \ldots, B_{d+1}, B_{d-1}, \ldots, B_1$. In addition, for any $\Pi_{d=1}^D p_d \times \Pi_{d=1}^D p_d$ matrix $\boldsymbol{H}$,

$$\begin{aligned}
\boldsymbol{H} \cdot \text{vec}(B) &= \boldsymbol{H} \mathbf{\Pi}_d \left[(B_D \odot \cdots \odot B_{d+1} \odot B_{d-1} \odot \cdots \odot B_1) \otimes \boldsymbol{I}_{p_d}\right] \text{vec}(B_d) \\
&= \boldsymbol{H}_{\pi_d} \left[(B_D \odot \cdots \odot B_{d+1} \odot B_{d-1} \odot \cdots \odot B_1) \otimes \boldsymbol{I}_{p_d}\right] \text{vec}(B_d),
\end{aligned}$$

where $\boldsymbol{H}_{\pi_d} = \boldsymbol{H} \mathbf{\Pi}_d$ should be pre-computed and stored for each d before block relaxation loop if computer memory capacity permits. Note $\boldsymbol{H}_{\pi_d}$ is just a column-permuted version of $\boldsymbol{H}$.

Therefore in Algorithm 3, for updating B_d, the optimization problem given B_d^t solves $B_d^{(t+1)}$ by minimizing

$$h_1(B_d | a_1^*, a_2^*, a_3^*, a_4^*) \;=\; \| - \text{vec}(B_d)^\intercal a_1^* a_2^* \text{vec}(B_d) + a_3^* \text{vec}(B_d) + a_4^* \|_2.$$

where a_1^*, a_2^*, a_3^* and a_4^* are updated at each B_d step in each iteration

$$\begin{aligned}
a_1^* &= [(B_D^{(t)} \odot \cdots \odot B_{d+1}^{(t)} \odot B_{d-1}^{(t+1)} \odot \cdots \odot B_1^{(t+1)}) \otimes \boldsymbol{I}_{p_d}]^\intercal \boldsymbol{\Pi}_d^\intercal \cdot a_1, \\
a_2^* &= a_2 \cdot \boldsymbol{\Pi}_d[(B_D^{(t)} \odot \cdots \odot B_{d+1}^{(t)} \odot B_{d-1}^{(t+1)} \odot \cdots \odot B_1^{(t+1)}) \otimes \boldsymbol{I}_{p_d}], \\
a_3^* &= a_3 \cdot \boldsymbol{\Pi}_d[(B_D^{(t)} \odot \cdots \odot B_{d+1}^{(t)} \odot B_{d-1}^{(t+1)} \odot \cdots \odot B_1^{(t+1)}) \otimes \boldsymbol{I}_{p_d}], \\
a_4^* &= a_4.
\end{aligned}$$

After solving B iteratively, one can find α and γ from (3.23).

3.6.2 Optimization – Logistic Regression

When X, W and B are tensors, we convert them into vectors and follow (3.22). Then the resulting problem is solving (α, B, γ) by minimizing

$$h_2(\alpha, \text{vec}(B), \gamma) = \| \sum_{i=1}^{n} \psi_s(y_i, z_i, \text{vec}(\bar{w}_i), \alpha, \text{vec}(B), \gamma) \|_2 \tag{3.26}$$

We directly apply Algorithm 2 to minimize (3.26). Also note that $\phi = 1$ and need not to be estimated under a logistic regression model.

3.7 Numerical Study

The conditional score is regarded as a method of estimating equations. Then the common issue of multiple roots come up with estimating equation method. Stefanski and Carroll (1987) stated the conditional score method yields multiple roots and not all of them are consistent. Also Heyde and Morton (1998) proved that there exists inconsistent roots for linear measurement error model. In the presence of multiple roots, Stefanski and Carroll

(1987) suggested using the one closest to the naive estimate, which is obtained after ignoring the measurement error. To implement this idea, they used the naive estimate as the starting point to solve the score equations iteratively. Although this is a heuristic idea, it performs well when the dimension of parameter space is very low. However, in our case of ultra-high dimensionality, it is difficult or usually impossible to find the naive estimate under a limited sample size. Zhou et al. (2013) developed the tensor regression to solve generalized linear model under a low-dimensional structure in high-dimensional systems. In this simulation study we use the estimate obtained by tensor regression as the naive estimate and apply Algorithm 2. We set the naive estimate as the starting point in Algorithm 2.

3.7.1 Simulation Setup

Linear Regression

Simulation is designed for linear models with measurement errors. We use 4 dimensional regular covariate Z ($p_0 = 4$). For array covariate X, we consider 2D image signal, or matrix signal with dimension 32×32. For each unobserved x_i, we assign m measurements $w_{ij}, j = 1, \ldots, m$, and we set m at three levels: 2, 5 and 20. Considering that the number of parameters is $1 + 4 + 32 \times 32 = 1029$, we let the sample size n in this study have three levels: 800, 1000 and 1500. We set the true intercept α be 1 and the true regular coefficient γ be $(1\ 1\ 1\ 1)^\intercal$. For the 2D tensor coefficient B, we consider four shapes: '**T**', 'cross', 'disk', and 'pentagon', which are all 32×32 square matrices and consist of zeros and ones. Each element of the observed regular covariate z_i is independently drawn from standard normal distribution, and so is the element of unobserved tensor covariate x_i. The measurement error U is independently generated from 2D tensor normal distribution,

or matrix normal distribution, with mean $\mathbf{0}_{p_1 \times p_2}$, and variance $\mathbf{\Omega}_0 = \Sigma_2 \otimes \Sigma_1$, where $\Sigma_1 = 2\boldsymbol{I}_{p_1}$ and $\Sigma_2 = \boldsymbol{I}_{p_2}$, $p_1 = 32, p_2 = 32$. In addition, we set the model error to be normal with mean zero and variance $\sigma^2 = 1$.

In the simulation study, we first estimate the error covariance matrix $\mathbf{\Omega}_0$. Note that $\mathbf{\Omega}_0$ has a separable structure that $\mathbf{\Omega}_0 = \mathbf{\Sigma}_2 \otimes \mathbf{\Sigma}_1$. Hoff (2011) gave a maximum likelihood estimate for $\mathbf{\Omega}_0$ under this separable structure. We follow this method and obtain the estimate $\hat{\mathbf{\Omega}}_0$.

Since the estimations of tensor coefficient B and the regular coefficient α, γ are separated in the case of linear regression, we obtain estimate of α and γ in (3.15) after solving B by minimizing (3.24). The simulation results are presented in tables (3.1)-(3.4), where the tensor regression estimates (Naive) and the conditional score estimates (CS) are compared in terms of mean absolute error (MAE). The results are based on 100 data replicates.

Logistic Regression

The setup of α, B, γ as well as Z, X, W is the same as the linear regression above. Y follows a Bernulli distribution with probability $\frac{\exp(\alpha+\gamma^\intercal z+\langle B,x\rangle)}{1+\exp(\alpha+\gamma^\intercal z+\langle B,x\rangle)}$. $\hat{\mathbf{\Omega}}_0$ will also be obtained by the method given in Hoff (2011). Noting $a(\phi) \equiv 1$ in logistic regression, we assume the ratio $\hat{\mathbf{\Omega}}_0/a(\phi) = \hat{\mathbf{\Omega}}_0$ is known when minimizing (3.26). The simulation results are also presented in tables (3.1)-(3.4). The results are based on 100 data replicates.

3.7.2 Simulation Results

According to Stefanski and Carroll (1987); Stefanski (2000); Carroll et al. (2006), conditional score method for measurement error model aims to reduce estimate bias. In this

simulation study, we evaluate our method's performance by looking at the MAE of the estimate. Our proposed method (CS) is compared with tensor regression method in Zhou et al. (2013) (Naive) in Table 3.1-3.4. It is shown that CS consistently outperforms Naive along all the settings in the context of linear regression. While in the context of logistic regression, CS sometimes performs equally to Naive, especially when the number of replicate measurements m is large. This is not surprising since we are using the mean of all measurements in the model as if there is only one measurement. So as the number of measurements goes up, the mean of measurements, $\bar{W}$, approximates to the true value X and the Naive gives a good estimate. Also the performances of both CS and Naive are improved as m goes up, which agrees with the fact that $\bar{W}$ approximates to the true value X as m increases.

3.8 Discussion

3.8.1 Summary

In this chapter we have proposed a tensor measurement error model based on the CP decomposition. We extend the classical conditional score method to tensor setup. Especially for linear measurement error model, we develop a simpler estimation equation than that in Carroll et al. (2006). We transform the original roots finding problem to a optimization problem and thus successfully integrate the CP decomposition to achieve a stable solution. We also integrate the method in Hoff (2011) to estimate the error covariance matrix, which is treated as known in classical conditional score method.

Table 3.1: Mean Absolute Error (MAE) of B estimate for '**T**' shape in linear and logistic regression. The true regression coefficient for each entry is either 0 or 1. The matrix variate has size 32 by 32 with entries generated as standard normals. The estimate is averaged over 100 data replicates.

	Linear regression				Logistic regression			
$(m\ n)$		$r=2$	$r=3$	$r=4$		$r=2$	$r=3$	$r=4$
(2 800)	Naive	0.145	0.149	0.152	Naive	0.252	0.238	0.108
	CS	**0.101**	**0.054**	**0.082**	CS	**0.214**	**0.221**	**0.108**
(2 1000)	Naive	0.143	0.146	0.148	Naive	0.253	0.247	0.226
	CS	**0.044**	**0.068**	**0.086**	CS	**0.204**	**0.215**	**0.217**
(2 1500)	Naive	0.142	0.145	0.146	Naive	0.253	0.251	0.248
	CS	**0.026**	**0.045**	**0.059**	CS	**0.191**	**0.203**	**0.210**
(5 800)	Naive	0.084	0.088	0.091	Naive	0.222	0.074	0.078
	CS	**0.021**	**0.037**	**0.050**	CS	**0.153**	**0.074**	**0.078**
(5 1000)	Naive	0.084	0.086	0.088	Naive	0.227	0.198	0.047
	CS	**0.018**	**0.029**	**0.039**	CS	**0.141**	**0.156**	**0.047**
(5 1500)	Naive	0.082	0.085	0.086	Naive	0.231	0.224	0.209
	CS	**0.012**	**0.019**	**0.026**	CS	**0.126**	**0.144**	**0.154**
(20 800)	Naive	0.050	0.053	0.056	Naive	0.039	0.027	0.088
	CS	**0.012**	**0.022**	**0.029**	CS	**0.033**	**0.027**	**0.083**
(20 1000)	Naive	0.048	0.051	0.053	Naive	0.123	0.078	0.039
	CS	**0.009**	**0.016**	**0.021**	CS	**0.032**	**0.078**	**0.038**
(20 1500)	Naive	0.048	0.050	0.051	Naive	0.163	0.085	0.079
	CS	**0.007**	**0.011**	**0.014**	CS	**0.048**	**0.035**	**0.079**

Table 3.2: Mean Absolute Error (MAE) of B estimate for 'cross' shape in linear and logistic regression. The true regression coefficient for each entry is either 0 or 1. The matrix variate has size 32 by 32 with entries generated as independent standard normals. The estimate is averaged over 100 data replicates.

	Linear regression				Logistic regression			
$(m\ n)$		$r=2$	$r=3$	$r=4$		$r=2$	$r=3$	$r=4$
(2 800)	Naive	0.126	0.130	0.133	Naive	0.215	0.203	0.092
	CS	**0.049**	**0.074**	**0.091**	CS	**0.180**	**0.189**	**0.092**
(2 1000)	Naive	0.123	0.126	0.128	Naive	0.215	0.210	0.193
	CS	**0.037**	**0.059**	**0.073**	CS	**0.169**	**0.181**	**0.184**
(2 1500)	Naive	0.122	0.124	0.125	Naive	0.216	0.213	0.211
	CS	**0.025**	**0.041**	**0.052**	CS	**0.156**	**0.168**	**0.175**
(5 800)	Naive	0.072	0.075	0.078	Naive	0.188	0.065	0.055
	CS	**0.018**	**0.033**	**0.043**	CS	**0.121**	**0.065**	**0.055**
(5 1000)	Naive	0.072	0.075	0.077	Naive	0.193	0.167	0.061
	CS	**0.015**	**0.026**	**0.035**	CS	**0.113**	**0.128**	**0.061**
(5 1500)	Naive	0.071	0.073	0.075	Naive	0.196	0.189	0.175
	CS	**0.011**	**0.018**	**0.023**	CS	**0.098**	**0.115**	**0.125**
(20 800)	Naive	0.042	0.045	0.048	Naive	0.030	0.041	0.066
	CS	**0.010**	**0.018**	**0.025**	CS	0.050	**0.041**	**0.061**
(20 1000)	Naive	0.042	0.045	0.046	Naive	0.099	0.097	0.026
	CS	**0.008**	**0.014**	**0.019**	CS	**0.048**	**0.097**	**0.026**
(20 1500)	Naive	0.042	0.044	0.045	Naive	0.136	0.066	0.093
	CS	**0.006**	**0.010**	**0.013**	CS	**0.065**	**0.052**	**0.093**

Table 3.3: Mean Absolute Error (MAE) of B estimate for 'disk' shape in linear and logistic regression. The true regression coefficient for each entry is either 0 or 1.The matrix variate has size 32 by 32 with entries generated as independent standard normals. The estimate is averaged over 100 data replicates.

		Linear regression				Logistic regression		
$(m\ n)$		$r=2$	$r=3$	$r=4$		$r=2$	$r=3$	$r=4$
(2 800)	Naive	0.234	0.232	0.231	Naive	0.371	0.349	0.189
	CS	**0.115**	**0.139**	**0.158**	CS	**0.329**	**0.334**	**0.189**
(2 1000)	Naive	0.230	0.228	0.226	Naive	0.374	0.366	0.337
	CS	**0.099**	**0.117**	**0.135**	CS	**0.323**	**0.331**	**0.329**
(2 1500)	Naive	0.226	0.224	0.222	Naive	0.376	0.372	0.368
	CS	**0.088**	**0.088**	**0.101**	CS	**0.310**	**0.318**	**0.325**
(5 800)	Naive	0.150	0.147	0.144	Naive	0.339	0.155	0.167
	CS	**0.071**	**0.076**	**0.086**	CS	**0.270**	**0.155**	**0.167**
(5 1000)	Naive	0.150	0.144	0.142	Naive	0.347	0.309	0.099
	CS	**0.067**	**0.064**	**0.075**	CS	**0.259**	**0.267**	**0.099**
(5 1500)	Naive	0.148	0.138	0.136	Naive	0.353	0.344	0.326
	CS	**0.067**	**0.052**	**0.052**	CS	**0.236**	**0.252**	**0.261**
(20 800)	Naive	0.104	0.095	0.094	Naive	0.161	0.103	0.187
	CS	**0.064**	**0.052**	**0.055**	CS	**0.101**	**0.103**	**0.184**
(20 1000)	Naive	0.103	0.092	0.090	Naive	0.259	0.076	0.119
	CS	**0.067**	**0.046**	**0.045**	CS	**0.091**	**0.076**	**0.119**
(20 1500)	Naive	0.103	0.089	0.085	Naive	0.297	0.228	0.052
	CS	**0.067**	**0.040**	**0.035**	CS	**0.066**	**0.079**	**0.051**

Table 3.4: Mean Absolute Error (MAE) of B estimate for 'pentagon' shape in linear and logistic regression. The true regression coefficient for each entry is either 0 or 1. The matrix variate has size 32 by 32 with entries generated as independent standard normals. The estimate is averaged over 100 data replicates.

		Linear regression				Logistic regression		
$(m\ n)$		$r=2$	$r=3$	$r=4$		$r=2$	$r=3$	$r=4$
(2 800)	Naive	0.199	0.198	0.197	Naive	0.307	0.289	0.150
	CS	**0.106**	**0.126**	**0.140**	CS	**0.270**	**0.274**	**0.150**
(2 1000)	Naive	0.195	0.194	0.193	Naive	0.309	0.302	0.280
	CS	**0.091**	**0.108**	**0.121**	CS	**0.262**	**0.272**	**0.270**
(2 1500)	Naive	0.194	0.190	0.188	Naive	0.310	0.307	0.303
	CS	**0.101**	**0.080**	**0.089**	CS	**0.255**	**0.260**	**0.265**
(5 800)	Naive	0.134	0.130	0.128	Naive	0.281	0.132	0.130
	CS	**0.070**	**0.073**	**0.081**	CS	**0.219**	**0.132**	**0.130**
(5 1000)	Naive	0.133	0.128	0.125	Naive	0.287	0.255	0.082
	CS	**0.068**	**0.064**	**0.069**	CS	**0.208**	**0.215**	**0.082**
(5 1500)	Naive	0.130	0.123	0.118	Naive	0.292	0.284	0.269
	CS	**0.072**	**0.054**	**0.052**	CS	**0.190**	**0.203**	**0.211**
(20 800)	Naive	0.099	0.091	0.087	Naive	0.150	0.079	0.142
	CS	**0.070**	**0.056**	**0.056**	CS	**0.080**	**0.079**	**0.139**
(20 1000)	Naive	0.096	0.087	0.082	Naive	0.222	0.099	0.076
	CS	**0.069**	**0.051**	**0.047**	CS	**0.072**	**0.099**	**0.076**
(20 1500)	Naive	0.096	0.085	0.080	Naive	0.248	0.197	0.052
	CS	**0.070**	**0.050**	**0.041**	CS	**0.091**	**0.056**	**0.052**

Table 3.5: Fitting run time for 'cross' shape (in seconds) at rank 2, 3 and 4 in linear regression setup (Section 3.7).

	Linear regression			
$(m\ n)$		$r = 2$	$r = 3$	$r = 4$
(2 800)	CS	14.36	23.98	54.48
(2 1000)	CS	15.81	28.91	57.90
(2 1500)	CS	15.45	21.13	59.32
(5 800)	CS	21.18	36.81	77.92
(5 1000)	CS	16.36	26.61	86.05
(5 1500)	CS	15.51	23.99	66.39
(20 800)	CS	20.77	32.53	71.60
(20 1000)	CS	19.45	25.82	80.15
(20 1500)	CS	15.92	22.72	66.39

3.8.2 Computation Time

We conduct a numerical experiment to study the computing time of the proposed method. We adopt the linear regression setting in Section 3.7 and use the 'cross' shape. Note that we use the tensor regression estimate as the starting point. Table 3.5 displays the run times for single data replicate under different setups.

3.8.3 Limitation

Multiple roots is a common issue in solving the estimation equation derived from the classical conditional score method, and when the dimension is large, the problem becomes more severe. We have done an experiment in classical setup to show conditional score method may fail when p is large. In Table 3.6, we investigate the performance of conditional score method in classical setup. We set five replicate measurements and 500 sample size. Elements of the $p \times 1$ covariate with measurement error, X, are independently simulated from Uniform(-1,1). We let the error covariance matrix $\mathbf{\Omega}_0$ be an identity matrix. Elements of the true coefficient for X are independently simulated from

Table 3.6: MAE of β estimate in classical setup (vector case). Sample size is 500. The estimate is averaged over 100 data replicates.

		$p=1$	$p=5$	$p=10$	$p=20$	$p=50$
Linear Regression	Naive	0.312	0.219	0.178	0.169	0.197
	CS	0.003	0.011	0.007	0.012	0.020
Logistic Regression	Naive	0.016	0.169	0.207	0.208	0.228
	CS	0.007	0.019	0.054	0.479	0.457

Uniform(-1,1). We also set the model error of linear regression to be $\sigma^2 = 1$. Simulation is based 100 data replicates. The results show conditional score method fails when p is relatively large in logistic regression, although it performs well in linear regression.

We follow the suggestion in Stefanski and Carroll (1987) to use the naive estimate as the starting point to iteratively solve the optimization problem. The results we obtained seem optimistic. In fact, the problem of local minimum may also be present in our high dimensional array setup and it should be addressed in future research.

Another drawback of our proposed estimate is its large variance. This is also noticed in Stefanski and Carroll (1987) and Carroll et al. (2006). Although the proposed estimate corrects bias, it can result in larger variance and mean squared error than the naive estimate.

For application, our method can give an unbiased estimate of the image coefficient. However it suffers the limitation in prediction which is common for all measurement error models. Since the estimate obtained through measurement error model is a correction based on naive estimate obtained from $(y_i, W_i), i = 1, \ldots, n$, directly, it may not be suitable for prediction based on new observed W.

Chapter 4

Tensor Linear Support Vector Machines

4.1 Introduction

With advances in the medical imaging technologies, a growing number of neuroimage data have become accessible for statistical analysis. In the last few years there has been increasing interest in applying statistical learning classifiers to neuroimage data analysis. Some problems of interest can be identifying voxels that are more active in the task condition than in the control condition, or classifying disease group and control group by examining subjects' neuroimages.

To be consistent with literature in image analysis, we use features as independent variables or predictors in this chapter. In a neuroimaging setting, the features could be voxels and the response could be the clinical outcome of patients. In some papers identifying regions of active voxels, the response could be the type of visual stimulus the subject received when the voxel values were recorded. Based on the summary of

Pereira et al. (2009), classical studies for neuroimage data always turn an image into a long vector at first. In image data analysis, there are generally many more features than subjects. Thus reducing the number of features that a classifier has to consider could be very crucial. In neuroimaging setting, studies solving this problem can be divided in two groups. The first group uses feature selection methods based on prior hypotheses, including regions of interest (ROIs) or key biological concerns. The other group adopts dimension reduction path to reduce the number of features fed into a classifier. Typical methods include singular value decomposition/principal component analysis (SVD/PCA) or independent component analysis (ICA). Other methods such as wavelet basis transform or autoregressive model can also be applied to transform the original feature space into a new, low-dimensional feature space.

After reducing the number of features, one trains classifier on the selected features. Commonly used methods in neuroimaging data include Gaussian Naive Bayes (GNB), linear discriminant analysis (LDA), logistic regression, and support vector machine (SVM). In this chapter we mainly focus on SVM and propose a new tensor linear SVM method, which utilizes information of the whole neuroimaging data without reducing the number of features. Unlike most studies in the literature that extract features from the original image data, we keep the array structure of the original images and use information from all the voxels in the proposed method. The new proposed classifier is more general since it can be trained without knowing much *a priori* information that varies case by case.

4.2 Overview

In this chapter we propose an SVM classifier with features in the form of tensor. SVM can be formulated as an optimization problem minimizing the 'hinge loss + penalty' cri-

terion. Motivated by this formulation, we impose a CP structure on the array coefficient of the linear term in the hinge loss and L_2 penalty on the factor matrices of the CP decomposition. We develop a highly efficient algorithm for the optimization. The proposed method performs well and stably with the original array covariate being upsized to ultra-high dimensionality by different basis functions. This demonstrates flexibility for ultra-high dimensional data. Both simulation study and real data analysis show that the proposed method outperforms other competitors such as tensor regression model (Zhou et al., 2013) and standard SVM.

The rest of this chapter is organized as follows. Section 4.3 reviews classification problems in neuroimage data based on SVM. We first review works in the two groups for reducing the number of features in Section 4.3.1 and 4.3.2. All of these works are based on SVM classifier. In Section 4.3.3, we simply review some classification methods. In Section 4.4.1 we introduce the basic principle of classical linear SVM. In Section 4.4.2 we propose the new tensor linear SVM method. We demonstrate its usefulness based on the numerical examples in Section 4.5.

4.3 Literature Review

4.3.1 Feature Selection

In neuroimaging setting, because of the high dimensionality of the feature (voxel) vectors, many studies focused on finding a subset of voxels, not only for reducing the number of features considered, but also for biological interest. We review three categories of feature selection methods here: voxel-based method, ROI-based method, and key biological feature selection. All these works embrace SVM for classification.

- Voxel-based selection: Cox and Savoy (2003); Mitchell et al. (2004); Hanson and Halchenko (2008); Costafreda et al. (2009)

- ROI-based selection: Cox and Savoy (2003); Mitchell et al. (2004); Davatzikos et al. (2005); Fan et al. (2005); Kamitani and Tong (2005); Dukart et al. (2013); Zhang et al. (2011); Klöppel et al. (2008); Magnin et al. (2009); Shen et al. (2010); Ecker et al. (2010)

- Key biological feature selection: Kamitani and Tong (2005); Dukart et al. (2013); Klöppel et al. (2008); Gerardin et al. (2009); Oliveira Jr et al. (2010); Ecker et al. (2010)

Cox and Savoy (2003) classified patterns of fMRI activation evoked by the visual presentation of various categories of objects. In the study, four subjects participated in 21 sessions of experiments. In each session, they were shown 10 categories of gray scale images and MRI activations evoked by the 10 stimuli were recorded. The size of the fMRI was $64 \times 64 \times 21$ (21 slices). The study used a univariate one-way ANOVA to do feature selection. Voxels that varied significantly across at least one of the categories of stimuli (p-value < 0.05, Bonferroni corrected for multiple comparisons) were selected. Various number of voxels from 50 to 200 were selected as input for classifiers. Linear SVM, cubic polynomial SVM and LDA were compared. SVM performed much better than LDA along different numbers of voxels used. However, the nonlinear (cubic polynomial) SVM did not significantly outperform linear SVM.

In the work of Mitchell et al. (2004), three experiments were conducted on fMRI image data. Classifiers were trained to distinguish cognitive states such as (1) whether the human subject was looking at a picture or a sentence, (2) whether the subject was reading an ambiguous or non-ambiguous sentence, and (3) whether the word the subject

was viewing is a word describing food, people, buildings, etc. They considered voxels within anatomically defined brain regions (e.g., dorsolateral prefrontal cortex), or ROI. Numbers of ROIs considered in the three experiments above were 7, 4 and 30. The numbers of voxels for the three experiments above were approximately 160000, 20000 and 10000. Numbers of subjects in the three experiments were 13, 10 and 10. For each subject, the numbers of examples were 80, 20 and 384 in the three experiments. A separate classifier for each subject was trained. Since in the experiments there was rest time between two presenting stimuli, the study considered subjects' condition in this time as the third class. Voxels were selected based on their ability to distinguish either target class from the fixation condition. The paper experimented with three classification learning methods: Gaussian Naive Bayes (GNB) classifier, k-nearest neighbor (kNN) and linear SVM. The GNB and SVM classifiers outperformed kNN. Their work focused mainly on training a different classifier for each human subject. Based on this work, Wang et al. (2003) trained classifiers that can be used across multiple human subjects in the first two case studies in Mitchell et al. (2004). They used the mean fMRI activity in each of several predefined ROIs as input to classifiers.

Davatzikos et al. (2005) performed lie detection on fMRI image data using high-dimensional nonlinear SVM. The fMRI images were $64 \times 64 \times 33$ and they were obtained from 22 right handed male students. The original data were preprocessed using statistical parametric mapping (SPM2, Wellcome Department of Cognitive Neurology, London, UK)(Friston et al., 1994). Then they calculated the hemodynamic response function (HRF) regressor absorbing the work in Friston et al. (1998). Afterwards, they divided regression coefficients of the HRF regressor into 560 cubes and used average for each cube as feature input in the SVM, so that the number of features was reduced to 560.

Costafreda et al. (2009) applied linear SVM to discriminate between patients with depressive disorder and healthy individuals based on their MRI data. They studied 37 patients with major depressive disorder (MD) and 37 healthy individuals. The size of the MRI was $256 \times 256 \times 124$. For feature selection, voxel-based morphometry (VBM) was first applied to the structural MRI images using SPM5 (Wellcome Trust Centre for Neuroimaging, UCL, London, UK). Then whole-brain ANOVA filtering was used to select the areas that differed significantly between patients and controls. Voxels in those areas were used as feature inputs in linear SVM.

Kamitani and Tong (2005) investigated classifying the perception of edge orientation using fMRI data. The size of the image was $256 \times 256 \times 25$. The responses were eight stimulus orientations. Four subjects participated in the experiment and they viewed a total of 20-24 trials for each orientation, so the sample size was 160-192 for each subject. Voxels used as feature inputs were selected on the cortical surface of 5 particular key biological areas, each of which had 200. They found voxels that might not be significant under univariate contrast test may still play an role in improving classification performance. They demonstrated that the line orientation of a grating was classifiable with a linear SVM using only voxels from one area. This was surprising since the microscopic organization of orientation-selective neurons in the cortical columns of that area was not significant with a mass univariate GLM. They argued that this might be because tiny orientation-selective biases in individual voxel can become informative when many such voxels get aggregated. Also, they found orientation decoding performance progressively improved with increasing number of voxels, as long as voxels were selected from the retinotopic region corresponding to the stimulated visual field.

Dukart et al. (2013) applied SVM to classify Alzheimers disease (AD) patients and healthy control subjects using MRI and PET data. SVM was trained on 28 AD patients

and 28 healthy control subjects and tested on 21 patients with AD and 13 control subjects. In the study, the sizes of MRI and PET were $128 \times 128 \times 128$ and $128 \times 128 \times 63$. Voxels of interest (VOIs) were selected to represent the maxima of atrophy or reductions in glucose utilization in AD. Six VOIs were extracted from each of the modalities by drawing a sphere with a radius of 5 mm around the reported coordinate and restricting the VOI to non-zero intensities within the sphere. The mean value of each VOI was used for classification.

Zhang et al. (2011) applied linear SVM to discriminate between Alzheimer's disease (or mild cognitive impairment) and healthy controls combining three modalities of biomarkers: MRI, FDG-PET, and CSF. The experiment of classifying AD and healthy controls was conducted in the data consisting of 51 AD patients and 52 healthy controls. The experiment of classifying MCI and healthy controls was conducted in the data consisting of 99 MCI patients and the same control group. For each MR or FDG-PET image, 93 ROIs were labeled and the volume of GM tissue was computed as feature input in each ROI. For CSF biomarkers, their original values are directly used as features. Then, a linear support vector machine (SVM) was adopted for classification.

Klöppel et al. (2008) applied linear SVM to classify the grey matter segment of MR scans from AD patients and normal elderly individuals obtained from two centers with different scanning equipment. Size of the scan was 256×192. Four groups of human subjects participated in the study. Three were used to discriminate between AD patients and normal individuals and the other was used to differentiate AD patients and frontotemporal lobar degeneration (FTLD) patients. Group I consisted of 20 AD patients and 20 normal controls. Group II consisted of 14 AD patients and 14 cognitively normal controls. Group III consisted of 33 patients with probable mild AD and 57 normal controls. Group IV consisted of 18 AD patients and 19 FTLD patients. The study used two kinds

of feature inputs for SVM classifier. The first kind was grey matter from the whole brain, and the second was only grey matter of antero-medial lobe volume of interest. Linear SVM was applied to data from different centers that used different scanners. The results showed that linear SVM generalized well.

Magnin et al. (2009) applied SVM classification of whole-brain MRI to discriminate between patients with Alzheimers disease (AD) and elderly control subjects. The database consisted of 38 samples: 22 elderly control and 16 AD patients. ROIs based feature selection method was used in this study. Specifically, the brain image was split into 90 ROIs. For each ROI, a histogram analysis of the distribution of the intensity in the voxels was performed, which identifies the respective contributions of gray matter (GM), white matter (WM), and cerebrospinal fluid (CSF) in the region. Afterwards, radial kernel SVM was performed based on the 90 dimensional feature vector.

Shen et al. (2010) applied RBF kernel SVM to classify patients with schizophrenia (SCH) and health controls based on fMRI data. Participants in the experiment included 32 schizophrenic patients and 20 healthy controls. The fMRI data size was $64 \times 64 \times 20$ with 180 volumes. They first obtained 6670 dimensional feature vectors based on their region partition of the fMRI data. They ranked the features according to their predefined discriminative power and the top 150 features were selected. Then features' dimension was further reduced by projecting feature space onto low-dimensional manifold. Afterwards, RBF kernel SVM was applied to the dimensionality-reduced feature inputs.

Gerardin et al. (2009) discriminated Alzheimer's disease (AD) patients and mild cognitive impairment (MCI) patients from normal aging subjects. They studied 23 patients with AD, 23 patients with MCI and compared them to 25 elderly healthy controls. MRI for each subjects had the size of $256 \times 192 \times 128$. Spherical harmonics (SPHARM) coefficients were used to model the shape of the hippocampi, which are segmented from MRI

(Chupin et al., 2007, 2009). These coefficients were used as feature inputs in the SVM classifier. RBF kernel SVM was used in this study.

Oliveira Jr et al. (2010) applied RBF kernel SVM to classify Alzheimers disease (AD) patients and healthy control subjects using MRI data. The database consisted of 14 AD patients and 20 healthy controls. Size of the MR scan was 256×256. Representations of cortical thickness were measured using MRI cortical surface-based analysis. Volumetric measurements of subcortical and cortical structures were used as feature inputs for the RBF kernel SVM classifier.

Ecker et al. (2010) applied linear SVM to discriminate between people with Autism spectrum disorder (ASD) and healthy controls based on their MRI data. Participants in the experiment included 20 individuals with ASD and 20 healthy controls. The MRI data size was $256 \times 192 \times 124$. In the study, cortical reconstruction and volumetric segmentation were performed on the MRI, then the cerebral cortex was parcellated into units based on gyral and sulcal structure (Fischl et al., 2004). For each parcellation unit, a set of five morphometric measurements that accounted for geometric characteristics of each parcellation were obtained. All of the morphometric measurements were taken as feature inputs and linear SVM classifier was built on them.

First, we can see that feature selection algorithms do not use the whole fMRI spatial information but only selected information. Second, the prior information for feature selection depends on the user, and is usually based on some *a prior* hypotheses, e.g. wishing to test whether voxels in a particular region of interest have information. However, in practice, *a priori* knowledge is not always available. Even if the regions of interest are able to be figured out, it is not guaranteed that classifier based on the regions of interest can lead to a good prediction. Third, these feature selection methods are accompanied with different and complex data preprocessing due to different experiment scenarios (Pereira

et al., 2009).

Recursive feature elimination (RFE) is a different way to do feature selection. This method begins with all features and removes features while performance increases. The essential idea of RFE is recursively training the classifier, computing the ranking criterion for input features, and removing the feature with the smallest ranking criterion. Guyon et al. (2002) used this method to select a small subset of genes from broad patterns of gene expression data and used the selected genes to build an SVM classifier. The study used the L_2 norm of the linear coefficient of the hyperplane as the feature ranking criterion, that is, in each recursion, the feature whose corresponding linear coefficient has smallest L_2 norm was eliminated. Through this procedure, the top ranked features were selected for the final SVM model. Different sizes of the selected features were considered in this study.

RFE can be combined with other feature selection techniques. Fan et al. (2005) applied SVM-RFE to a group of previously rank-ordered features that are computed from the extracted regions. The problem was classifying 61 MR T1 brain images of normal controls and schizophrenia patients. They actually performed 4 steps for feature selection. First, they segmented images into three tissues: GM, WM and CSF using the approach in Davatzikos et al. (2001). Then a high dimensional image warping method given in Shen and Davatzikos (2002) was used to transform the segmented images into a template space. This resulted in three tissue density maps, $f_0(u), f_1(u), f_2(u)$, in the template space, with 0, 1, 2 corresponding to GM, WM and CSF respectively at location u. Second, for each feature $f_i(u), i = 0, 1, 2$, a single score $s_i(u)$ was generated to reflect the classification power of this feature, which included discriminative power and reliability. Then a watershed segmentation method proposed in Vincent and Soille (1991) was employed to partition a brain into different regions according to the score $s_i(u)$. Third, For

each region generated above, the corresponding volumetric measure was computed by summing up all tissue density values ($f_i(u)$, $i = 0, 1, 2$) in this region. Then the volumetric measures from all regions constituted an attribute vector to represent morphological information of the brain. Last, SVM-RFE given in Guyon et al. (2002) was employed to do further feature selection. After the 4 steps above, a final feature vector was obtained for each observation, then Gaussian kernel SVM was applied to do classification.

However, there are limitations using RFE in neuroimaging setting. First, given the often prohibitive computational expense, this is in general infeasible (Pereira et al., 2009), although there were heuristic techniques that have been successfully used with fMRI data with only approximately 40000 voxels (Hanson and Halchenko, 2008). One should also notice that the work in Fan et al. (2005) applied SVM-RFE only after a few steps of preprocessing and feature extraction. However in many other neuroimaging settings, the length of the vectorized images can reach up to millions. Then searching exhaustively the whole feature set would require a large number of iteration. Second, deciding how many of the top-ranked voxels to use becomes much harder due to the ultra-high dimensionality. Third, RFE aims to finding or ranking all potentially relevant features, which contrasts with the problem of selecting subsets of features that are useful to build a good classifier (Guyon and Elisseeff, 2003). Selecting the most relevant features is usually suboptimal for building a predictor (Guyon and Elisseeff, 2003). And this drawback also exists in other feature selection methods relative to regions of interest.

4.3.2 Dimension Reduction Methods

Dimension reduction methods build new features based on the original features and transform the original feature space into a new, low-dimensional feature space. Typical

algorithms like singular value decomposition/principal component analysis, or independent component analysis generate basis vectors in the directions of maximum variance (SVD/PCA) or which are close to independence (ICA). These methods generate linear uncorrelated combinations of the original features. Other than that, wavelet decomposition has also been used in applications for dimension reduction. In addition for EEG data, sometimes autoregressive (AR) models are performed for each channel and the AR coefficients are used as feature inputs to classifiers. We review several dimension reduction methods here

- PCA/SVD: LaConte et al. (2003); Mouro-Miranda et al. (2005); Álvarez et al. (2009)
- ICA: Martino et al. (2007)
- Wavelet decomposition: Chaplot et al. (2006); Lao et al. (2004)
- Autoregressive model based method: Garrett et al. (2003); Schlögl et al. (2005)
- Signal cross-correlation based method: Chandaka et al. (2009)

PCA or SVD is widely used for dimension reduction in fMRI data analysis. Many studies applied the classifier on PCA/SVD basis without prior selection of spatial features. LaConte et al. (2003) examined performance of SVM in terms of misclassification error by using PCA inputs. Mouro-Miranda et al. (2005) demonstrated SVM outperforms LDA in classification performance as well as in the robustness of the spatial maps obtained.

LaConte et al. (2003) compared SVM and canonical variates analysis (CVA) in terms of misclassification error based on fMRI data. In the study, eight right-handed volunteers performed finger task using right or left hand. Linear SVM was applied to classify left-

hand vs. right-hand tasks. The size of the fMRI was $64 \times 64 \times 30$, with 135 volumes. PCA was used to reduce the number of feature inputs to 364 (the number of PCs used).

Mouro-Miranda et al. (2005) studied classification of brain states from whole fMRI volumes. In their work, SVM and LDA methods were applied to distinguish two brain states by looking at the fMRI. Two multisubject attention experiments were considered: a face matching and a location matching task. The fMRI data were from 16 human subjects. Sliced images of 64×64 and 69 functional volumes were acquired. Each fMRI volume was treated as a feature vector in a high-dimensional space. PCA was applied for dimension reduction. For each subject, the corresponding fMRI data matrix $\boldsymbol{D}_{M\times T}$, where M was the number of voxels and T was the number of time points ($M \gg T$), was projected to a much smaller matrix $\boldsymbol{D}^{p}_{T\times T}$ using principal components of SVD. The superscript p indicated matrix after projection. Using the principal components as input features, the study compared SVM and LDA in terms of prediction accuracy and SVM outperformed LDA in the experiment.

Álvarez et al. (2009) applied linear SVM to classify Alzheimers disease (AD) patients and healthy control subjects using Single Photon Emission Computed Tomography (SPECT) images. The database consists of 79 samples: 41 healthy control subjects and 38 AD patients. The image size was $79 \times 95 \times 69$. Images of the brain were reconstructed from the projection data using the filtered back-projection (FBP) algorithm in combination with a Butter-worth noise removal filter. The SPECT images were spatially normalized using the SPM software. Afterwards, PCA was used for dimension reduction in this study.

Martino et al. (2007) performed SVM classification of independent components (ICs) extracted from fMRI data sets. In the experiment, seven human subjects participated in the study and each subject conducted two fMRI runs (14 obervations in total). Original

image's size was $256 \times 256 \times 192$. Then 60 ICs were extracted from fMRI data using cortex based ICA (Formisano et al., 2004), so dimension was reduced from 256×256 to 60. Afterwards, for each IC, values of eleven descriptive measures (IC-fingerprints) were derived from the IC's voxel values' distribution (kurtosis, skewness, entropy), spatial layout (degree of clustering in the anatomical space) as well as their temporal (one-lag autocorrelation, entropy) and spectral (power contribution in five different frequency bands) properties. Dimension was further reduced to 11 from 60. They then proceeded to classify the eleven-length IC-fingerprints with SVM classifier.

However, PCA decomposition is nontrivial in the high dimensionality framework, since one still has to convert images into vectors at first. For instance, 1024×256 images would be turned into vectors of length $1024 \times 256 = 262144$. Performing PCA based on such feature vectors could still be quite computationally intensive. Second, linear transformation such as PCA is not effective in capturing complex relationships in high-dimensional spaces (Fan et al., 2005).

Other than PCA or ICA based methods, there were methods using wavelet transform to extract features from images. In the work of classifying 52 MR brain images of normal controls and Alzheimer patients (Chaplot et al., 2006), wavelet transform (Mallat, 1989) was used to extract features from the original 256×256 images. Daubechies-4 (DB4) wavelet (Daubechies et al., 1992) was chosen to extract wavelet approximation coefficients of MR brain images. These coefficients were used as feature vector for classification. The original 256×256 images were downsized to 69×69, which further resulted in a $69 \times 69 = 4761$ dimensional feature vector. SVM and neural network methods were compared and SVM performed better than neural network in the experiment.

Lao et al. (2004) applied SVM to differentiate images from different groups (e.g. older/younger, male/female). Total number of subjects was 153. Regional analysis of

volumes examined in normalized space (RAVENS) maps were obtained from the original MRIs. The original size of the RAVENS maps was $256 \times 256 \times 124$. Daubechies wavelet decomposition was applied to the RAVENS maps and then the wavelet features were ranked according to their discrimination measures. The top 2000 features were selected as feature inputs for the SVM classifier. Classification of younger and older, or male and female was considered in the study.

Garrett et al. (2003) examined the application of SVM to the problem of EEG classification and compared the results to those obtained using neural networks and LDA. They performed the analysis on two data sets. In the first study, subjects were asked to perform four different activities including mentally composing a letter, doing math, counting numbers and rotating a solid. EEG data were recorded with size 125 samples $\times$ 6 channels. Sixth-order autoregressive (AR) models were performed for each channel independently and the data were reduced to 36 dimensions (6 AR coefficients $\times$ 6 channels). Twenty five responses were obtained with five for each activity and five for the fixation time. SVM were tested with polynomial kernels and RBF kernels. The results showed SVM outperformed neural networks and LDA. In the second study, a subject performed a typewriting task. A total of 516 keystrokes were done. EEG data were recorded to classify the laterality of the left/right hand signal. Six channels were considered and, for each channel, 11 features were selected using genetic algorithms (Goldberg et al., 1989; Mitchell, 1998; Whitley et al., 1997; Yang and Honavar, 1998); so there were 66 feature inputs. Afterwards, SVM was used and it gave a good accuracy.

Schlögl et al. (2005) compared the performance of different classifiers applied to four-class EEG data. In the study, EEG data were recorded with 60 electrodes from five subjects performing four different motor-imagery tasks. Number of observations for each subject was 240-360. Classifiers were applied individually for each subject and the per-

formance of classifiers was compared using t-test along five subjects. Three adaptive autoregressive (AAR) parameters were calculated in this study for each of the 60 channels, resulting in a feature input of 180 dimensions. Linear SVM, LDA and k-NN were compared and SVM performed the best.

Chandaka et al. (2009) used SVM to classify healthy volunteers and epileptic patients based on their EEG data. Cross-correlation based method was applied to reduce the number of feature inputs. In correlation, a cross-correlation sequence between two energy signals measures the extent of similarity between these two signals (Proakis and Manolakis, 2007). In the case study, the original data set consisted of 100 healthy volunteers and 100 epileptic patients. The dimension of the EEG data was 4096. They first selected one healthy volunteers as reference. The reference data were cross-correlated with the data of the rest of healthy volunteers and all the epilepsy patients. So sample sizes for healthy volunteers and epilepsy patients were 99 and 100. Based on the cross-correlation sequence, five features were selected: peak value, instant at which peak occurs, equivalent width, centroid and mean square abscissa. Linear SVM was performed based on the five dimensional feature inputs.

One common drawback of these dimension reduction methods is that they ignore class labels and do not consider classification performance but rather how well the original data can be reconstructed from the basis and the low-dimensional representation (Müller et al., 2004); there is no guarantee that a classifier based on reduced dimensionality data produced by these algorithms will work well (Pereira, 2007; Pereira et al., 2009).

4.3.3 Classification Models

In general setting, images are first turned into feature vectors. Let the feature vector be $\boldsymbol{X} = (X_1, \ldots, X_p)^\intercal$, where p is the number of voxels, and the response variable Y be binary, i.e. $Y \in \{-1, 1\}$. Let the observed value be $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n$ and $y_1, \ldots, y_n$, where n is the number of observations.

Gaussian Naive Bayes

Bayes rule tells how to predict Y from an example $\boldsymbol{X}$

$$P(Y = k|\boldsymbol{X}) = \frac{P(\boldsymbol{X}|Y = k)P(Y = k)}{P(\boldsymbol{X})}$$

in terms of the distribution $P(\boldsymbol{X}|Y = k)$, $k = \pm 1$. The predicted Y is labeled by the class with the highest posterior distribution $P(Y = k|\boldsymbol{X})$.

If conditional distribution of each feature is normal and independent of the others, i.e. $P(\boldsymbol{X}|Y = k) = \prod_{j=1}^{p} P(X_j|Y = k), k = \pm 1$, then we have a Gaussian Naive Bayes (GNB) classifier (Mitchell, 1997).

Linear Discriminant Analysis

Using the Bayes rule above, LDA considers a ratio

$$R = \frac{P(Y = 1|\boldsymbol{X})}{P(Y = -1|\boldsymbol{X})} = \frac{P(\boldsymbol{X}|Y = 1)P(Y = 1)}{P(\boldsymbol{X}|Y = -1)P(Y = -1)},$$

where $P(\boldsymbol{X}|Y = k), k = \pm 1$, are normal distributed with equal variance. Then the decision rule becomes "$Y = 1$ if $R > 1$ or $Y = -1$ if $R < 1$". We can see that GNB is equivalent to LDA if the covariance matrix of distribution $\boldsymbol{X}$ given $Y = k$ is diagonal.

Logistic Regression

Logistic regression models $P(Y = k|\boldsymbol{X})$ as

$$\begin{aligned} P(Y = 1|\boldsymbol{X}) &= \frac{\exp(\alpha + \boldsymbol{X}^\intercal\boldsymbol{\beta})}{1 + \exp(\alpha + \boldsymbol{X}^\intercal\boldsymbol{\beta})}; \\ P(Y = -1|\boldsymbol{X}) &= \frac{1}{1 + \exp(\alpha + \boldsymbol{X}^\intercal\boldsymbol{\beta})}. \end{aligned}$$

The parameters α and $\boldsymbol{\beta}$ are estimated by maximizing the log likelihood $\sum_{i=1}^{n} \log(P(Y = y_i|\boldsymbol{x}_i))$.

Linear Support Vector Machine

A linear Support Vector Machine (SVM) estimate parameters α and $\boldsymbol{\beta}$ by minimizing

$$\sum_{i=1}^{n} [1 - y_i(\alpha + \boldsymbol{x}_i^\intercal\boldsymbol{\beta})]_+ + \frac{\lambda}{2}\left(\|\alpha\|^2 + \|\boldsymbol{\beta}\|^2\right),$$

where λ is the penalization parameter. The decision rule is "$Y = 1$ if $\alpha + \boldsymbol{X}^\intercal\boldsymbol{\beta} > 0$ or $Y = -1$ if $\alpha + \boldsymbol{X}^\intercal\boldsymbol{\beta} < 0$".

Classifier Choice

According to Pereira (2007); Pereira et al. (2009), logistic regression and linear SVM tend to be roughly equivalent in terms of performance. In cases where there are a large number of features, some other classifiers such as Gaussian Naive Bayes (GNB) and Fisher's Linear Discriminant Analysis (LDA) are inferior to logistic regression and linear SVM. For some other more complex classifiers such as nonlinear SVM or neural networks, it is not clear that they always provide a significant advantage in practical performance

relative to linear classifiers (in existing studies or the multi-study comparison in Pereira (2007)). In addition, the relationship between features and the prediction can become harder to interpret when these classifiers are used (Pereira et al., 2009). In this chapter we mainly discuss linear SVM and logistic regression.

4.4 Model and Optimization

4.4.1 Introduction to Support Vector Machine (SVM)

We describe the principle of linear SVM here. The SVM methodology uses separating hyperplanes for binary classification problem. It adjusts a discriminating function to establish the optimal separating hyperplane, which separates the two classes and maximizes the distance to the closest point from either class (Vapnik, 1996; Cortes and Vapnik, 1995; Hastie et al., 2009). Suppose we have a set of n training samples $(\boldsymbol{x}_1, y_1), \ldots, (\boldsymbol{x}_n, y_n)$, where $\boldsymbol{x}_i \in \mathbb{R}^p, i = 1, \ldots, n$ are feature vectors or inputs and $y_i \in \{-1, 1\}, i = 1, \ldots, n$, are the class label that $\boldsymbol{x}_i$s belong to. The linear discriminating function is $f(\boldsymbol{x}) = \boldsymbol{x}^\intercal \boldsymbol{\beta} + \alpha$ and the classification rule is

$$G(\boldsymbol{x}) = \text{sign}(\boldsymbol{x}^\intercal \boldsymbol{\beta} + \alpha).$$

One can normalize $f(\boldsymbol{x})$ such that $|f(\boldsymbol{x}_i)| \geq 1, i = 1, \ldots, n$, and the distance from the hyperplane $f(\boldsymbol{x}) = 0$ to the closest training point is $1/\|\boldsymbol{\beta}\|_2$. The optimal separating hyperplane algorithm selects the hyperplane that maximize the distance, which leads to

the following optimization problem

$$\min_{\alpha,\boldsymbol{\beta}} \|\boldsymbol{\beta}\|_2$$
$$\text{subject to } y_i f(\boldsymbol{x}_i) \geq 1, i = 1, \ldots, n.$$

When the training data are not linearly separable, a set of slack variables $\xi_i, i = 1, \ldots, n$ are introduced. The corresponding optimization problem is

$$\min_{\alpha,\boldsymbol{\beta}} \|\boldsymbol{\beta}\|_2$$
$$\text{subject to } y_i f(\boldsymbol{x}_i) \geq 1 - \xi_i, i = 1, \ldots, n;$$
$$\xi_i \geq 0, \sum_{i=1}^{n} \xi_i \leq \text{constant}.$$

The problem above can be expressed in the equivalent form below, which is also the usual form for support vector classifier in non-separable case

$$\min_{\alpha,\boldsymbol{\beta}} \frac{1}{2}\|\boldsymbol{\beta}\|_2^2 + C\sum_{i=1}^{n} \xi_i$$
$$\text{subject to } \xi_i \geq 0, y_i f(\boldsymbol{x}_i) \geq 1 - \xi_i, i = 1, \ldots, n, \tag{4.1}$$

where the parameter C represents the trade-off between the misclassification rate and the separation of two classes. Furthermore, one could use Karush-Kuhn-Tucker (KKT) conditions (Kuhn and Tucker, 1951) to formulate another equivalent problem of (4.1) based on hinge loss

$$\min_{\alpha,\boldsymbol{\beta}} \sum_{i=1}^{n} [1 - y_i(\alpha + \boldsymbol{\beta}^{\mathsf{T}}\boldsymbol{x}_i)]_+ + \frac{\lambda}{2}\|\boldsymbol{\beta}\|_2^2, \tag{4.2}$$

with $\lambda = 1/C$. The subscript "+" means the positive part. The solution of $\boldsymbol{\beta}$ has a

representation

$$\boldsymbol{\beta} = \sum_{i=1}^{n} u_i y_i \boldsymbol{x}_i,$$

where $u_i, i = 1, \ldots, n$ are non-negative Lagrange multipliers associated with the constraints in (4.1). They can be obtained by solving a constrained quadratic programming problem

$$\max_{u_i} \sum_{i=1}^{n} u_i - \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} u_i u_j y_i y_j \boldsymbol{x}_i^{\intercal} \boldsymbol{x}_j$$
$$\text{subject to } 0 \leq u_i \leq C \text{ and } \sum_{i=1}^{n} u_i y_i = 0.$$

4.4.2 Tensor Linear SVM

The optimization problem (4.2) has the representation of "loss + penalty". The first piece is in the form of "hinge" loss, which is reasonable for two-class classification, when compared to other more traditional loss function (Hastie et al., 2009). In multi-dimensional array setting, we follow this idea of using hinge loss function for the classification problem.

Let $\boldsymbol{X}$ be the tensor covariate, $\boldsymbol{X} \in \mathbb{R}^{p_1 \times \ldots \times p_D}$, where D is the dimension. Let Y be the class label, $Y \in \{-1, 1\}$, and $\boldsymbol{Z}$ be the conventional $p_0 \times 1$ vector-valued covariate. We denote the observed values of Y, $\boldsymbol{Z}$ and $\boldsymbol{X}$ by y_i, $\boldsymbol{Z}_i$ and $\boldsymbol{X}_i$, $i = 1, \ldots, n$, where n is the sample size. To handle this problem, we consider the following hinge loss function

$$\ell_0 = \sum_{i=1}^{n} \left[1 - y_i \left\{\alpha + \boldsymbol{\gamma}^{\intercal} \boldsymbol{Z}_i + \langle \boldsymbol{B}, \boldsymbol{X} \rangle\right\}\right]_+, \tag{4.3}$$

where $\langle \boldsymbol{B}, \boldsymbol{X} \rangle = \text{vec}(\boldsymbol{X})^{\intercal} \text{vec}(\boldsymbol{B})$.

We impose a CP decomposition introduced in (1.1) on $\boldsymbol{B}$:

$$\boldsymbol{B} = \sum_{r=1}^{R} \boldsymbol{\beta}_1^{(r)} \circ \cdots \circ \boldsymbol{\beta}_D^{(r)}$$

where $\boldsymbol{\beta}_d^{(r)} \in \mathbb{R}^{p_d}, d = 1, \ldots, D, r = 1, \ldots, R$, and the shorthand $\boldsymbol{B} = [\![\boldsymbol{B}_1, \ldots, \boldsymbol{B}_D]\!]$, where $\boldsymbol{B}_d = [\boldsymbol{\beta}_d^{(1)}, \ldots, \boldsymbol{\beta}_d^{(R)}] \in \mathbb{R}^{p_d \times R}$, $d = 1, \ldots, D$.

Under the CP decomposition, ℓ_0 given in (4.3) can be expressed as

$$\ell_0(\alpha, \boldsymbol{\gamma}, \boldsymbol{B}_1, \ldots, \boldsymbol{B}_D) = \sum_{i=1}^{n} \left[1 - y_i \left\{\alpha + \boldsymbol{\gamma}^\intercal \boldsymbol{Z}_i + \langle [\![\boldsymbol{B}_1, \ldots, \boldsymbol{B}_D]\!], \boldsymbol{X}_i \rangle\right\}\right]_+ .$$

The standard SVM uses L_2 penalty as shown in (4.2). Although there are many other widely used penalties with extension to SVM application such as L_1 penalty (Bradley and Mangasarian, 1998; Zhu et al., 2003), L_0 penalty (Weston et al., 2003), F_∞ penalty (Zou and Yuan, 2008), and SCAD penalty (Zhang et al., 2006), they mainly focused on variable selection. In our context, the application of tensor decomposition facilitates to reduce number of features and thus we focus on applying L_2 penalty on the coefficients. Specifically we solve the problem by minimizing the following hinge loss based objective function

$$\begin{aligned} \ell(\alpha, \boldsymbol{\gamma}, \boldsymbol{B}_1, \ldots, \boldsymbol{B}_D) = & \sum_{i=1}^{n} \left[1 - y_i \left\{\alpha + \boldsymbol{\gamma}^\intercal \boldsymbol{Z}_i + \langle [\![\boldsymbol{B}_1, \ldots, \boldsymbol{B}_D]\!], \boldsymbol{X}_i \rangle\right\}\right]_+ + \\ & \frac{\lambda}{2} \left(\|\boldsymbol{\gamma}\|^2 + \sum_{r=1}^{R} \sum_{d=1}^{D} \|\boldsymbol{\beta}_d^{(r)}\|^2 \right). \end{aligned} \tag{4.4}$$

We impose penalization only on parameters of the CP decomposition, i.e. $\boldsymbol{B}_1, \ldots, \boldsymbol{B}_D$, although it is conceptually possible to apply penalization to the full coefficient array $\boldsymbol{B} \in \mathbb{R}^{\prod_{d=1}^{D} p_d}$ without considering the CP decomposition structure. However for neu-

Algorithm 4 Block relaxation algorithm for minimizing (4.4).

Initialize: $\boldsymbol{B}_d^{(0)} \in \mathbb{R}^{p_d \times R}$ a random matrix for $d = 1, \ldots, D$, $(\alpha^{(0)}, \boldsymbol{\gamma}^{(0)}) = \text{argmin}_{\alpha,\boldsymbol{\gamma}}\, \ell(\alpha, \boldsymbol{\gamma}, \boldsymbol{B}_1^{(0)}, \ldots, \boldsymbol{B}_D^{(0)})$.
repeat
  for $d = 1, \ldots, D$ **do**

$$\boldsymbol{B}_d^{(t+1)} = \text{argmin}_{\boldsymbol{B}_d} \Big[\ell_0(\alpha^{(t)}, \boldsymbol{\gamma}^{(t)}, \boldsymbol{B}_1^{(t+1)}, \ldots, \boldsymbol{B}_{d-1}^{(t+1)}, \boldsymbol{B}_d, \boldsymbol{B}_{d+1}^{(t)}, \ldots, \boldsymbol{B}_D^{(t)}) + \frac{\lambda}{2} \sum_{r=1}^{R} \|\boldsymbol{\beta}_d^{(r)}\|^2 \Big]$$

  end for
  $(\alpha^{(t+1)}, \boldsymbol{\gamma}^{(t+1)}) = \text{argmin}_{\alpha,\boldsymbol{\gamma}} \left[\ell_0(\alpha, \boldsymbol{\gamma}, \boldsymbol{B}_1^{(t+1)}, \ldots, \boldsymbol{B}_D^{(t+1)}) + \frac{\lambda}{2}\|\gamma\|^2 \right]$
until $\ell(\boldsymbol{\theta}^{(t+1)}) - \ell(\boldsymbol{\theta}^{(t)}) < \epsilon$

roimaging data, we should bear in mind the dimensionality of the imaging arrays and ultrahigh dimensionality in penalization corrupts the statistical properties of the penalized estimates (Zhou et al., 2013).

We adopt an efficient algorithm for minimizing $\ell(\alpha, \boldsymbol{\gamma}, \boldsymbol{B}_1, \ldots, \boldsymbol{B}_D)$. Notice that although the inner product in (4.4) is not linear in $(\boldsymbol{B}_1, \ldots, \boldsymbol{B}_D)$ jointly, it is linear in $\boldsymbol{B}_d$ individually. We then alternately update $(\alpha, \boldsymbol{\gamma})$ and $\boldsymbol{B}_d, d = 1, \ldots, D$, while keeping other components fixed. This was so-called block relaxation algorithm (de Leeuw, 1994) and recently applied to maximum likelihood estimation of the tensor regression model (Zhou et al., 2013).

Algorithm 4 outlines the optimization procedure. At initialization, we first generate random $\boldsymbol{B}_d^{(0)}, d = 1, \ldots, D$. Specifically, each element of $\boldsymbol{B}_d^{(0)}$ is independently generated from uniform(-1,1); $(\alpha^{(0)}, \boldsymbol{\gamma}^{(0)})$ can be obtained by fitting a low dimension SVM. Then the loss function (4.4) is iteratively minimized. In each iteration, we alternately update $\boldsymbol{B}_d \in \mathbb{R}^{p_d \times R}, d = 1, \ldots, D$, and $(\alpha, \boldsymbol{\gamma})$. When updating $\boldsymbol{B}_d \in \mathbb{R}^{p_d \times R}$, we transform the

inner product in (4.4) to

$$\langle \sum_{r=1}^{R} \boldsymbol{\beta}_1^{(r)} \circ \boldsymbol{\beta}_2^{(r)} \circ \cdots \circ \boldsymbol{\beta}_D^{(r)}, \boldsymbol{X} \rangle = \langle \boldsymbol{B}_d, \boldsymbol{X}_{(d)} (\boldsymbol{B}_D \odot \cdots \odot \boldsymbol{B}_{d+1} \odot \boldsymbol{B}_{d-1} \odot \cdots \odot \boldsymbol{B}_1) \rangle.$$

Then updating each $\boldsymbol{B}_d$ turns into an SVM model in a manageable dimension of Rp_d. The whole estimation procedure breaks into a sequence of optimizations that can be easily implemented using ready SVM package LIBSVM (Chang and Lin, 2011) in MATLAB.

4.5 Numerical Study

4.5.1 Example 1

We generate the data in the linear discriminant analysis (LDA) setup. LDA is often used in simulation studies on various types of SVM (Koo et al., 2008; Wang et al., 2008). Throughout this example, the sample sizes for training, tuning and testing data sets are 200, 100 and 100. Response $Y \in \{1, -1\}$ is generated with equal probability 0.5. We let X have size 16×16 or 32×32. If $Y = 1$, $\text{vec}(X)$ is generated from $\text{Normal}(c\mathbf{1}, \boldsymbol{\Sigma}_2 \otimes \boldsymbol{\Sigma}_1)$, and if $Y = -1$, $\text{vec}(X)$ is generated from $\text{Normal}(-c\mathbf{1}, \boldsymbol{\Sigma}_2 \otimes \boldsymbol{\Sigma}_1)$. Let $c = 0.16, 0.17, 0.18$ if X is 16×16, $c = 0.05, 0.06, 0.07$ if X is 32×32. Also, we set $\boldsymbol{\Sigma}_1 = \boldsymbol{\Sigma}_2$, and both have first-order autoregressive structure with homogenous variances. We denote the correlation between two adjacent elements by ρ, and set $\rho = 0.6$ if X is 16×16, $\rho = 0.4$ if X is 32×32. Bayes rule is used to control the optimal misclassification rate. In this study, we compare classical linear and radial kernel SVM after vectorizing $\boldsymbol{X}$ and our proposed tensor SVM. We also include regularized tensor logistic regression (Zhou et al., 2013) as a reference in this study. We use grid-search method for tuning the regularization parameter λ, the kernel width for radial kernel SVM and the fitted tensor rank R for

Table 4.1: Comparison of tensor SVM (tsvm), linear SVM (lsvm), radial kernel SVM (rsvm), and tensor logistic regression with L_1 and L_2 regularization (glm(L_1),glm(L_1)) in terms of misclassification rate in linear learning setup. Results are averaged over 100 replications. Numbers in parentheses are standard errors.

		$c = 0.16$	$c = 0.17$	$c = 0.18$
	Bayes rule	0.224	0.210	0.196
16×16	lsvm	0.258(0.005)	0.243(0.005)	0.228(0.005)
	rsvm	0.253(0.005)	0.238(0.004)	0.221(0.004)
	glm(L_1)	0.292(0.005)	0.273(0.006)	0.257(0.005)
	glm(L_2)	0.263(0.005)	0.244(0.005)	0.233(0.004)
	tsvm	0.246(0.005)	0.233(0.005)	0.217(0.005)
		$c = 0.05$	$c = 0.06$	$c = 0.07$
	Bayes rule	0.238	0.196	0.159
32×32	lsvm	0.327(0.005)	0.281(0.005)	0.228(0.004)
	rsvm	0.320(0.005)	0.268(0.005)	0.220(0.005)
	glm(L_1)	0.383(0.005)	0.321(0.005)	0.270(0.005)
	glm(L_2)	0.331(0.005)	0.281(0.005)	0.226(0.005)
	tsvm	0.299(0.005)	0.245(0.005)	0.200(0.005)

tensor SVM. Specifically for tensor SVM, various pairs of (λ, R) values are tried and the one with the best validation accuracy is picked. For vectorized linear SVM, only λ is tuned. We search λ from values suggested by Chang and Lin (2011). The fitted tensor rank R is searched from 2, 3 and 4. Table 4.1 shows the results. Averaged misclassification error over 100 data replicates is reported, with standard error in the parentheses. 'lsvm' and 'rsvm' refer to linear and radial kernel SVM after vectorizing $\boldsymbol{X}$. 'tsvm' denotes our proposed tensor SVM method. 'glm(L_1)' and 'glm(L_2)' denote tensor logistic regression method with L_1 and L_2 regularization. It is clear to see that tensor SVM significantly outperforms other competitors.

4.5.2 Example 2

We set up another simulation study for the classification problem here. We generate Y from a Bernoulli distribution with probability $p(\boldsymbol{B}, \boldsymbol{X})$. We consider logit and probit link for p. Specifically, $p = \exp\{\langle \boldsymbol{B}, \boldsymbol{X} \rangle\}/(1 + \exp\{\langle \boldsymbol{B}, \boldsymbol{X} \rangle\})$ under logit link, and $p = \Phi(\langle \boldsymbol{B}, \boldsymbol{X} \rangle)$, where Φ is the cumulative distribution function of standard normal, under probit link. We let the true signal of $\boldsymbol{B}$ be rectangle, 'T' shape, cross, triangle, pentagon and disk. Each element of $\boldsymbol{X}$ is independently generated from standard normal. Three types of dimension on $\boldsymbol{X}$ are considered: 16×16, 32×32 and 64×64. Sample sizes for training, tuning and testing data sets are (100, 100, 100) for 16×16 $\boldsymbol{X}$, (150, 150, 150) for 32×32 $\boldsymbol{X}$ and (300, 300, 300) for 64×64 $\boldsymbol{X}$. The ratio of 0/1 response is approximately 1:1 in all three data sets. The grid-search is used for tuning parameters.

Table 4.2: Comparison of tensor SVM (tsvm), linear SVM (lsvm), radial kernel SVM (rsvm), and tensor logistic regression with L_1 and L_2 regularization (glm(L_1),glm(L_2)) in terms of misclassification rate in logit link setup. Results are averaged over 100 replications. Numbers in parentheses are standard errors.

		Rectangle	T	Cross	Triangle	Pentagon	Disk
16×16	lsvm	0.354(0.005)	0.366(0.005)	0.374(0.006)	0.366(0.006)	0.368(0.005)	0.363(0.005)
	rsvm	0.369(0.005)	0.369(0.005)	0.375(0.005)	0.369(0.005)	0.380(0.006)	0.377(0.005)
	glm(L_1)	0.239(0.006)	0.318(0.005)	0.319(0.006)	0.322(0.007)	0.294(0.005)	0.296(0.006)
	glm(L_2)	0.207(0.005)	0.285(0.005)	0.292(0.006)	0.302(0.006)	0.266(0.005)	0.259(0.005)
	tsvm	0.217(0.005)	0.287(0.005)	0.302(0.006)	0.304(0.006)	0.268(0.005)	0.262(0.005)
32×32	lsvm	0.410(0.004)	0.411(0.004)	0.410(0.004)	0.414(0.004)	0.405(0.004)	0.412(0.004)
	rsvm	0.426(0.004)	0.426(0.004)	0.421(0.005)	0.426(0.004)	0.424(0.004)	0.428(0.004)
	glm(L_1)	0.333(0.006)	0.378(0.006)	0.393(0.005)	0.376(0.005)	0.368(0.006)	0.363(0.005)
	glm(L_2)	0.265(0.005)	0.334(0.005)	0.340(0.005)	0.329(0.005)	0.301(0.005)	0.297(0.005)
	tsvm	0.273(0.004)	0.343(0.005)	0.344(0.005)	0.339(0.005)	0.313(0.005)	0.308(0.005)
64×64	lsvm	0.439(0.003)	0.438(0.003)	0.437(0.003)	0.435(0.003)	0.436(0.003)	0.439(0.003)
	rsvm	0.446(0.003)	0.440(0.003)	0.445(0.003)	0.444(0.003)	0.446(0.003)	0.447(0.003)
	glm(L_1)	0.370(0.005)	0.417(0.004)	0.431(0.004)	0.420(0.004)	0.393(0.004)	0.395(0.003)
	glm(L_2)	0.262(0.004)	0.347(0.004)	0.355(0.004)	0.347(0.004)	0.311(0.004)	0.310(0.004)
	tsvm	0.276(0.003)	0.351(0.004)	0.360(0.004)	0.355(0.004)	0.316(0.004)	0.319(0.004)

Table 4.3: Comparison of tensor SVM (tsvm), linear SVM (lsvm), radial kernel SVM (rsvm), and tensor logistic regression with L_1 and L_2 regularization (glm(L_1),glm(L_2)) in terms of misclassification rate in probit link setup. Results are averaged over 100 replications. Numbers in parentheses are standard errors.

		Rectangle	T	Cross	Triangle	Pentagon	Disk
16×16	lsvm	0.351(0.005)	0.366(0.005)	0.362(0.006)	0.354(0.005)	0.362(0.005)	0.361(0.005)
	rsvm	0.365(0.005)	0.366(0.006)	0.370(0.005)	0.364(0.005)	0.367(0.006)	0.375(0.005)
	glm(L_1)	0.225(0.006)	0.298(0.006)	0.293(0.006)	0.294(0.005)	0.292(0.006)	0.290(0.006)
	glm(L_2)	0.213(0.005)	0.294(0.006)	0.287(0.005)	0.296(0.005)	0.272(0.006)	0.270(0.006)
	tsvm	0.213(0.005)	0.277(0.005)	0.287(0.005)	0.282(0.005)	0.261(0.005)	0.254(0.005)
32×32	lsvm	0.405(0.005)	0.410(0.004)	0.410(0.004)	0.408(0.004)	0.406(0.004)	0.411(0.004)
	rsvm	0.421(0.005)	0.426(0.004)	0.420(0.005)	0.428(0.004)	0.425(0.005)	0.425(0.004)
	glm(L_1)	0.334(0.006)	0.379(0.005)	0.390(0.006)	0.373(0.005)	0.357(0.006)	0.370(0.006)
	glm(L_2)	0.284(0.005)	0.347(0.005)	0.360(0.005)	0.347(0.005)	0.321(0.006)	0.318(0.005)
	tsvm	0.270(0.004)	0.338(0.005)	0.338(0.005)	0.338(0.005)	0.310(0.005)	0.308(0.004)
64×64	lsvm	0.440(0.003)	0.435(0.003)	0.435(0.004)	0.432(0.003)	0.437(0.003)	0.438(0.003)
	rsvm	0.444(0.003)	0.441(0.003)	0.444(0.004)	0.438(0.003)	0.445(0.003)	0.444(0.003)
	glm(L_1)	0.372(0.005)	0.417(0.004)	0.424(0.004)	0.420(0.005)	0.401(0.004)	0.401(0.005)
	glm(L_2)	0.272(0.004)	0.353(0.004)	0.359(0.004)	0.353(0.004)	0.324(0.004)	0.319(0.004)
	tsvm	0.272(0.003)	0.351(0.004)	0.357(0.003)	0.351(0.004)	0.323(0.005)	0.319(0.004)

Table 4.2 and 4.3 show results under logit and probit models. Mean misclassification error over 100 data replicates is reported, with standard error in the parentheses. We can see that tensor SVM and tensor logistic regression with L_2 regularization perform the best. But these two methods do not have significantly different performance. The fact that tensor logistic regression performs well is not surprising since the true signal is generated in tensor regression framework.

4.5.3 EEG Data Analysis

We analyze EEG data and compare a number of SVM methods. The data set consists of 77 individuals with alcoholism and 45 healthy controls. For each subject, 64 channels of electrodes were placed at different locations of scalp and the voltage values are recorded at 256 time points. So the image covariates $\boldsymbol{X}_i$ are 256×64 matrices, and the responses y_i are binary whether the ith subject is alcoholic ($y_i = 1$) or not ($y_i = 0), i = 1, \ldots 122$. We applied classical SVM with linear and radial kernels, and our proposed tensor SVM to the data. We evaluate each solution via cross-validation based misclassification rate. More specifically, we divided the full data into a training and a testing sample using 3-fold cross-validation. Then for the training data, we further employ a 5-fold cross-validation to tune the shrinkage parameter and tensor rank. We then apply the tuned model that is fully based on the training data now to the testing data and evaluate the misclassification rate for the testing data.

We also employ the Tucker tensor decomposition to upsize the image predictor. More specifically, we transform $\boldsymbol{X}$ to $\tilde{\boldsymbol{X}} = [\![\boldsymbol{X}; \boldsymbol{B}_1^{\mathrm{T}}, \ldots, \boldsymbol{B}_D^{\mathrm{T}}]\!]$, where $\boldsymbol{B}_d \in \mathbb{R}^{p_d \times \tilde{p}_d}$ are functional basis. One could refer to the duality in Section 2.4.3 for more details. We experiment with the Daubechies' extremal phase wavelets, and discrete cosine transform (DCT). For

Daubechies wavelets, we apply Haar wavelet basis (Daubechies D2) and the Daubechies D4 wavelet basis. We upsize $\boldsymbol{X}$ to several settings and fit SVM with image predictor $\tilde{\boldsymbol{X}}$.

The computation scheme for $\boldsymbol{B}_d \in \mathbb{R}^{p_d \times \tilde{p}_d}$ is not difficult when using an orthogonal wavelet, for example Daubechies wavelet and symlet wavelet. We start with the two filters of length $2L$, denoted $h(l)$ and $g(l)$, $l = 1, \ldots, 2L$. For example, if we have Haar wavelet then $L = 1$, $h(0) = h(1) = 1/\sqrt{2}$ and $g(0) = -g(1) = 1/\sqrt{2}$. The following sequence of functions $f_m(x), m = 0, 1, \ldots$ are computed recursively:

$$f_{2m}(x) = \sqrt{2} \sum_{k=0,\ldots,2L-1} h(k) f_m(2x - k) \tag{4.5}$$

$$f_{2m+1}(x) = \sqrt{2} \sum_{k=0,\ldots,2L-1} g(k) f_m(2x - k), \tag{4.6}$$

where $f_0(x) = \phi(x)$ is the scaling function and $f_1(x) = \psi(x)$ is the wavelet function. For example for Haar wavelet:

$$f_{2m}(x) = f_m(2x) + f_m(2x - 1)$$
$$f_{2m+1}(x) = f_m(2x) - f_m(2x - 1),$$

and $f_0(x)$ is the haar scaling function

$$f_0(x) = \begin{cases} 1, & \text{if } 0 \le x < 1 \\ 0, & \text{otherwise} \end{cases};$$

$f_1(x)$ is the haar wavelet function

$$f_1(x) = \begin{cases} 1, & \text{if } 0 \leq x < 1/2 \\ -1, & \text{if } 1/2 \leq x < 1 \cdot \\ 0, & \text{otherwise} \end{cases}$$

Then for $\boldsymbol{B}_d^{\mathsf{T}} \in \mathbb{R}^{\tilde{p}_d \times p_d}$, we take a row vector $\boldsymbol{v}$ containing p_d points linearly spaced from 0 to $2L-1$, rows of $\boldsymbol{B}_d^{\mathsf{T}}$ are evaluated by $f_k(\boldsymbol{v}), k = 0, \ldots, \tilde{p}_d - 1$.

In addition, the DCT is also often used in signal and image processing, especially for image data resizing (Ahmed et al., 1974; Rao et al., 1990). For $\boldsymbol{B}_d \in \mathbb{R}^{p_d \times \tilde{p}_d}$, we specify $\boldsymbol{B}_d$ to be DCT basis functions. We apply multi-dimensional DCT (Lee et al., 1999) in our setup.

$$\boldsymbol{B}_d(t,s) = \cos \frac{\pi(2t+1)s}{2p_d}, t = 1, \ldots, p_d; s = 1, \ldots, \tilde{p}_d, \tag{4.7}$$

where $\boldsymbol{B}_d(t,s)$ denotes the element at tth row and sth column.

Table 4.4 shows the results for different sizes of image. It is clear that the proposed tensor SVM performs the best on the original image. Also as the image size goes up, tensor SVM shows very stable performance. This feature is present for a variety methods of resizing image, which indicates the flexibility of proposed method.

Table 4.4: Misclassification rate for the electroencephalography data in different resizing setup.

		lsvm	rsvm	tsvm	glm(L_1)	glm(L_2)
256×64	Original	0.304	0.368	0.279	0.418	0.418
		lsvm	rsvm	tsvm	glm(L_1)	glm(L_2)
	Haar (D2)	0.246	0.368	0.229	0.351	0.343
256×256	Daubechies (D4)	0.246	0.279	0.221	0.385	0.401
	DCT	0.304	0.368	0.287	0.368	0.351
		lsvm	rsvm	tsvm	glm(L_1)	glm(L_2)
	Haar (D2)	0.229	0.279	0.172	0.392	0.376
512×512	Daubechies (D4)	0.295	0.303	0.254	0.319	0.418
	DCT	0.246	0.368	0.238	0.368	0.368
		lsvm	rsvm	tsvm	glm(L_1)	glm(L_2)
	Haar (D2)	0.188	0.368	0.181	0.327	0.327
512×128	Daubechies (D4)	0.279	0.287	0.271	0.352	0.377
	DCT	0.197	0.369	0.189	0.352	0.327
		lsvm	rsvm	tsvm	glm(L_1)	glm(L_2)
	Haar (D2)	0.222	0.339	0.221	0.319	0.327
1024×256	Daubechies (D4)	0.295	0.311	0.286	0.377	0.360
	DCT	0.254	0.368	0.237	0.319	0.310

4.6 Discussion

4.6.1 Summary

We have proposed a tensor linear SVM model based on CP decomposition. We develop a fast algorithm to optimize the hinge loss function in tensor setup. We provide a comprehensive comparison in terms of prediction among tensor regression model in Zhou et al. (2013), classical SVM, and the proposed tensor linear SVM under a variety of environments. Both simulations and real data analysis show the usefulness and flexibility of the proposed model. In addition, the proposed model enjoys an advantage over other classifiers for neuroimaging data in the literature, which is the generality. In our new method, we keep the array structure of the original images and use information from all the voxels. The new method does not need prior information or feature selection, which is complicated and varies case by case in many studies.

4.6.2 Computation Time

The optimization problem in this chapter is solved by a highly scalable algorithm and implemented with an efficient Matlab toolbox, LIBSVM (Chang and Lin, 2011). We conduct a numerical experiment to study the computing time of proposed tensor SVM. We adopt the logit link setting in Section 4.5.2 and use the 'cross' shape. We test tsvm's computing time by running the algorithm on one data set from 100 random starting points. Table 4.5 displays the median wall clock run times for lsvm, rsvm, and tsvm.

We can see that the computing of tsvm is slower than rsvm or lsvm in Table 4.5. However, the run time is remarkably fast for high dimensional data. For instance, fitting a rank-3 tensor SVM to upsized 512×512 EEG data in Section 4.5.3 cost 1 sec, however

Table 4.5: Run time comparison (in seconds) of lsvm, rsvm, and tsvm at rank 2, 3, and 4 for 'cross' signal. Reported are the median time of 100 runs.

Dimension	tsvm			lsvm	rsvm
	Rank 2	Rank 3	Rank 4		
16×16	0.28	0.03	0.12	0.01	0.01
32×32	0.12	0.06	0.09	0.03	0.03
64×64	10.43	2.40	2.59	0.58	0.52

standard linear SVM took 2.5 sec and standard radial kernel SVM took 3 sec.

4.6.3 Limitation

In the real data analysis in Section 4.5.3, EEG data are upsized within a Tucker decomposition framework. But when building tensor SVM classifier, we impose CP decomposition on the array coefficient. Although combining CP decomposition and SVM already gives decent results in prediction accuracy, we can expect a better performance if Tucker decomposition is applied for the optimization of hinge loss.

The optimization problem considered in this chapter is within a 'loss + penalty' framework. Using L_2 penalty, we achieve decent classification accuracy in both simulation studies and real data analysis. However, L_2 SVM generally does not select variables (Bradley and Mangasarian, 1998). Other penalties such as L_1 and SCAD (Bradley and Mangasarian, 1998; Zhu et al., 2003; Zhang et al., 2006) should receive more emphasis in scenarios where the interest is identifying the discriminant regions of the images.

REFERENCES

ADHD (2013). The ADHD-200 sample. [Online; accessed 03-2013].

ADNI (2013). Alzheimer's disease neuroimaging initiative. [Online; accessed 03-2013].

Ahmed, N., Natarajan, T., and Rao, K. (1974). Discrete cosine transform. *Computers, IEEE Transactions on*, C-23(1):90–93.

Allen, G., Grosenick, L., and Taylor, J. (2011). A generalized least squares matrix decomposition. *Rice University Technical Report No. TR2011-03*, arXiv:1102:3074.

Álvarez, I., Górriz, J. M., Ramírez, J., Salas-Gonzalez, D., López, M., Segovia, F., Puntonet, C. G., and Prieto, B. (2009). Alzheimers diagnosis using eigenbrains and support vector machines. In *Bio-Inspired Systems: Computational and Ambient Intelligence*, pages 973–980. Springer.

Aston, J. A. and Kirch, C. (2012). Estimation of the distribution of change-points with application to fmri data. *Annals of Applied Statistics*, 6:1906–1948.

Blankertz, B., Curio, G., and Müller, K.-R. (2001). Classifying single trial EEG: Towards brain computer interfacing. In *NIPS*, pages 157–164.

Bradley, P. S. and Mangasarian, O. L. (1998). Feature selection via concave minimization and support vector machines. In *ICML*, volume 98, pages 82–90.

Buzas, J. S. and Stefanski, L. A. (1996). Instrumental variable estimation in generalized linear measurement error models. *Journal of the American Statistical Association*, 91(435):pp. 999–1006.

Caffo, B., Crainiceanu, C., Verduzco, G., Joel, S., S.H., M., Bassett, S., and Pekar, J. (2010). Two-stage decompositions for the analysis of functional connectivity for fMRI with application to Alzheimer's disease risk. *Neuroimage*, 51(3):1140–1149.

Carroll, R., Ruppert, D., Stefanski, L., and Crainiceanu, C. (2006). *Measurement Error in Nonlinear Models: A Modern Perspective, Second Edition*. Chapman & Hall/CRC Monographs on Statistics & Applied Probability. Taylor & Francis.

Carroll, R. J. and Stefanski, L. A. (1990). Approximate quasi-likelihood estimation in models with surrogate predictors. *Journal of the American Statistical Association*, 85(411):652–663.

Chandaka, S., Chatterjee, A., and Munshi, S. (2009). Cross-correlation aided support vector machine classifier for classification of eeg signals. *Expert Systems with Applications*, 36(2):1329–1336.

Chang, C.-C. and Lin, C.-J. (2011). Libsvm: a library for support vector machines. *ACM Transactions on Intelligent Systems and Technology (TIST)*, 2(3):27.

Chaplot, S., Patnaik, L., and Jagannathan, N. (2006). Classification of magnetic resonance brain images using wavelets as input to support vector machine and neural network. *Biomedical Signal Processing and Control*, 1(1):86–92.

Chen, S. S., Donoho, D. L., and Saunders, M. A. (2001). Atomic decomposition by basis pursuit. *SIAM Rev.*, 43(1):129–159.

Chupin, M., Hammers, A., Liu, R. S., Colliot, O., Burdett, J., Bardinet, E., Duncan, J. S., Garnero, L., and Lemieux, L. (2009). Automatic segmentation of the hippocampus

and the amygdala driven by hybrid constraints: method and validation. *Neuroimage*, 46(3):749–761.

Chupin, M., Mukuna-Bantumbakulu, A. R., Hasboun, D., Bardinet, E., Baillet, S., Kinkingnéhun, S., Lemieux, L., Dubois, B., and Garnero, L. (2007). Anatomically constrained region deformation for the automated segmentation of the hippocampus and the amygdala: Method and validation on controls and patients with alzheimers disease. *Neuroimage*, 34(3):996–1019.

Cook, J. R. and Stefanski, L. A. (1994). Simulation-extrapolation estimation in parametric measurement error models. *Journal of the American Statistical Association*, 89(428):pp. 1314–1328.

Cortes, C. and Vapnik, V. (1995). Support-vector networks. *Machine learning*, 20(3):273–297.

Costafreda, S. G., Chu, C., Ashburner, J., and Fu, C. H. (2009). Prognostic and diagnostic potential of the structural neuroanatomy of depression. *PLoS One*, 4(7):e6353.

Cox, D. D. and Savoy, R. L. (2003). Functional magnetic resonance imaging (fmri) brain reading: detecting and classifying distributed patterns of fmri activity in human visual cortex. *NeuroImage*, 19(2):261 – 270.

Crainiceanu, C. M., Caffo, B. S., Luo, S., Zipunnikov, V. M., and Punjabi, N. M. (2011). Population value decomposition, a framework for the analysis of image populations. *J. Amer. Statist. Assoc.*, 106(495):775–790.

Daubechies, I. et al. (1992). *Ten lectures on wavelets*, volume 61. SIAM.

Davatzikos, C., Genc, A., Xu, D., and Resnick, S. M. (2001). Voxel-based morphometry using the ravens maps: methods and validation using simulated longitudinal atrophy. *NeuroImage*, 14(6):1361–1369.

Davatzikos, C., Ruparel, K., Fan, Y., Shen, D., Acharyya, M., Loughead, J., Gur, R., and Langleben, D. (2005). Classifying spatial patterns of brain activity with machine learning methods: Application to lie detection. *NeuroImage*, 28(3):663 – 668.

De Lathauwer, L. and Vandewalle, J. (2004). Dimensionality reduction in higher-order signal processing and rank-(r1,r2,,rn) reduction in multilinear algebra. *Linear Algebra and its Applications*, 391:31–55.

de Leeuw, J. (1994). Block-relaxation algorithms in statistics. In *Information Systems and Data Analysis*, pages 308–325. Springer, Berlin.

Dukart, J., Mueller, K., Barthel, H., Villringer, A., Sabri, O., and Schroeter, M. L. (2013). Meta-analysis based svm classification enables accurate detection of alzheimer's disease across different clinical centers using fdg-pet and mri. *Psychiatry Research: Neuroimaging*, 212(3):230–236.

Ecker, C., Marquand, A., Mourão-Miranda, J., Johnston, P., Daly, E. M., Brammer, M. J., Maltezos, S., Murphy, C. M., Robertson, D., Williams, S. C., et al. (2010). Describing the brain in autism in five dimensionsmagnetic resonance imaging-assisted diagnosis of autism spectrum disorder using a multiparameter classification approach. *The Journal of Neuroscience*, 30(32):10612–10623.

Fan, J. and Li, R. (2001). Variable selection via nonconcave penalized likelihood and its oracle properties. *J. Amer. Statist. Assoc.*, 96(456):1348–1360.

Fan, Y., Shen, D., and Davatzikos, C. (2005). Classification of structural images via high-dimensional image warping, robust feature extraction, and svm. In *Medical Image Computing and Computer-Assisted Intervention–MICCAI 2005*, pages 1–8. Springer.

Fischl, B., Salat, D. H., van der Kouwe, A. J., Makris, N., Ségonne, F., Quinn, B. T., and Dale, A. M. (2004). Sequence-independent segmentation of magnetic resonance images. *Neuroimage*, 23:S69–S84.

Formisano, E., Esposito, F., Salle, F. D., and Goebel, R. (2004). Cortex-based independent component analysis of fmri time series. *Magnetic Resonance Imaging*, 22(10):1493 – 1504.

Frank, I. E. and Friedman, J. H. (1993). A statistical view of some chemometrics regression tools. *Technometrics*, 35(2):109–135.

Friston, K. J., Holmes, A. P., Worsley, K. J., Poline, J.-P., Frith, C. D., and Frackowiak, R. S. (1994). Statistical parametric maps in functional imaging: a general linear approach. *Human brain mapping*, 2(4):189–210.

Friston, K. J., Josephs, O., Rees, G., and Turner, R. (1998). Nonlinear event-related responses in fmri. *Magnetic resonance in medicine*, 39(1):41–52.

Garrett, D., Peterson, D. A., Anderson, C. W., and Thaut, M. H. (2003). Comparison of linear, nonlinear, and feature selection methods for eeg signal classification. *Neural Systems and Rehabilitation Engineering, IEEE Transactions on*, 11(2):141–144.

Gerardin, E., Chételat, G., Chupin, M., Cuingnet, R., Desgranges, B., Kim, H.-S., Niethammer, M., Dubois, B., Lehéricy, S., Garnero, L., et al. (2009). Multidimensional

classification of hippocampal shape features discriminates alzheimer's disease and mild cognitive impairment from normal aging. *Neuroimage*, 47(4):1476–1486.

Gleser, L. J. (1981). Estimation in a multivariate "errors in variables" regression model: Large sample results. *Ann. Stat.*, 9:24–44.

Gleser, L. J., Brown, P. J., and Fuller, W. A. (1990). Improvements of the naive approach to estimation in non-linear errors-in-variables regression models. In *Statistical Analysis of Measurement Error Models and Application.* American Mathematics Society, Providence, RI.

Goldberg, D. E. et al. (1989). *Genetic algorithms in search, optimization, and machine learning*, volume 412. Addison-wesley Reading Menlo Park.

Guyon, I. and Elisseeff, A. (2003). An introduction to variable and feature selection. *The Journal of Machine Learning Research*, 3:1157–1182.

Guyon, I., Weston, J., Barnhill, S., and Vapnik, V. (2002). Gene selection for cancer classification using support vector machines. *Machine learning*, 46(1-3):389–422.

Hansen, L. P. (1982). Large sample properties of generalized method of moments estimators. *Econometrica: Journal of the Econometric Society*, pages 1029–1054.

Hanson, S. J. and Halchenko, Y. O. (2008). Brain reading using full brain support vector machines for object recognition: there is no face identification area. *Neural Computation*, 20(2):486–503.

Hastie, T., Tibshirani, R., Friedman, J., Hastie, T., Friedman, J., and Tibshirani, R. (2009). *The elements of statistical learning*, volume 2. Springer.

Haxby, J. V., Gobbini, M. I., Furey, M. L., Ishai, A., Schouten, J. L., and Pietrini, P. (2001). Distributed and overlapping representations of faces and objects in ventral temporal cortex. *Science*, 293(5539):2425–2430.

Heyde, C. and Morton, R. (1998). Multiple roots in general estimating equations. *Biometrika*, 85(4):954–959.

Hoff, P. (2011). Hierarchical multilinear models for multiway data. *Computational Statistics and Data Analysis*, 55:530–543.

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

Hung, H. and Wang, C.-C. (2013). Matrix variate logistic regression model with application to eeg data. *Biostatistics*, 14(1):189–202.

Kamitani, Y. and Tong, F. (2005). Decoding the visual and subjective contents of the human brain. *Nature neuroscience*, 8(5):679–685.

Kilmer, M. E. and Martin, C. D. M. (2004). Decomposing a tensor. *SIAM News*, 37(9):19–20.

Klöppel, S., Stonnington, C. M., Chu, C., Draganski, B., Scahill, R. I., Rohrer, J. D., Fox, N. C., Jack, C. R., Ashburner, J., and Frackowiak, R. S. (2008). Automatic classification of mr scans in alzheimer's disease. *Brain*, 131(3):681–689.

Kolda, T. G. (2006). Multilinear operators for higher-order decompositions. Technical report.

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

Kontos, D., Megalooikonomou, V., Kontos, D., Faloutsos, C., Megalooikonomou, V., Ghubade, N., and Faloutsos, C. (2003). Detecting discriminative functional MRI activation patterns using space filling curves. In *in Proc. of the 25th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC*, pages 963–967. Springer-Verlag.

Koo, J.-Y., Lee, Y., Kim, Y., and Park, C. (2008). A bahadur representation of the linear support vector machine. *The Journal of Machine Learning Research*, 9:1343–1368.

Kuhn, H. W. and Tucker, A. W. (1951). Nonlinear programming.

LaConte, S., Strother, S., Cherkassky, V., Anderson, J., and Hu, X. (2005). Support vector machines for temporal classification of block design fMRI data. *Neuroimage*, 26:317–329.

LaConte, S., Strother, S., Cherkassky, V., and Hu, X. (2003). Predicting motor tasks in fmri data with support vector machines. *ISMRM, Toronto. May*.

Lange, K. (2004). *Optimization*. Springer Texts in Statistics. Springer-Verlag, New York.

Lange, K. (2010). *Numerical Analysis for Statisticians*. Statistics and Computing. Springer, New York, second edition.

Lao, Z., Shen, D., Xue, Z., Karacali, B., Resnick, S. M., and Davatzikos, C. (2004). Morphological classification of brains via high-dimensional shape transformations and machine learning methods. *Neuroimage*, 21(1):46–57.

Lee, J.-H., Kim, D.-H., and Chung, C.-W. (1999). Multi-dimensional selectivity estimation using compressed histogram information. In *ACM SIGMOD Record*, volume 28, pages 205–214. ACM.

Lehmann, E. L. and Romano, J. P. (2005). *Testing Statistical Hypotheses*. Springer Texts in Statistics. Springer, New York, third edition.

Liu, N., Zhang, B., Yan, J., Chen, Z., Liu, W., Bai, F., and Chien, L. (2005). Text representation: From vector to tensor. In *Data Mining, Fifth IEEE International Conference on*, pages 4–pp. IEEE.

Liu, X. and Liang, K.-Y. (1992). Efficacy of repeated measures in regression models with measurement error. *Biometrics*, pages 645–654.

Magnin, B., Mesrob, L., Kinkingnéhun, S., Pélégrini-Issac, M., Colliot, O., Sarazin, M., Dubois, B., Lehéricy, S., and Benali, H. (2009). Support vector machine-based classification of alzheimers disease from whole-brain anatomical mri. *Neuroradiology*, 51(2):73–83.

Mallat, S. G. (1989). A theory for multiresolution signal decomposition: the wavelet representation. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 11(7):674–693.

Martino, F. D., Gentile, F., Esposito, F., Balsi, M., Salle, F. D., Goebel, R., and Formisano, E. (2007). Classification of fmri independent components using ic-fingerprints and support vector machine classifiers. *NeuroImage*, 34(1):177 – 194.

McCullagh, P. and Nelder, J. A. (1983). *Generalized Linear Models*. Monographs on Statistics and Applied Probability. Chapman & Hall, London.

Mckeown, M. J., Makeig, S., Brown, G. G., Jung, T.-P., Kindermann, S. S., Kindermann, R. S., Bell, A. J., and Sejnowski, T. J. (1998). Analysis of fMRI data by blind separation into independent spatial components. *Human Brain Mapping*, 6:160–188.

Mitchell, M. (1998). *An introduction to genetic algorithms*. MIT press.

Mitchell, T. M. (1997). Machine learning. 1997. *Burr Ridge, IL: McGraw Hill*, 45.

Mitchell, T. M., Hutchinson, R., Niculescu, R. S., Pereira, F., Wang, X., Just, M., and Newman, S. (2004). Learning to decode cognitive states from brain images. *Machine Learning*, 57(1-2):145–175.

Mouro-Miranda, J., Bokde, A. L., Born, C., Hampel, H., and Stetter, M. (2005). Classifying brain states and determining the discriminating activation patterns: Support vector machine on functional {MRI} data. *NeuroImage*, 28(4):980 – 995.

Müller, K.-R., Krauledat, M., Dornhege, G., Curio, G., and Blankertz, B. (2004). Machine learning techniques for brain-computer interfaces.

Muti, D. and Bourennane, S. (2005). Multidimensional filtering based on a tensor approach. *Signal Processing*, 85(12):2338–2353.

Nakamura, T. (1990). Corrected score function for errors-in-variables models: Methodology and application to generalized linear models. *Biometrika*, 77(1):127–137.

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

Oliveira Jr, P. P. d. M., Nitrini, R., Busatto, G., Buchpiguel, C., Sato, J. R., and Amaro Jr, E. (2010). Use of svm methods with surface-based cortical and volumetric subcortical measurements to detect alzheimer's disease. *Journal of Alzheimer's Disease*, 19(4):1263–1272.

Pereira, F. (2007). *Beyond brain blobs: machine learning classifiers as instruments for analyzing functional magnetic resonance imaging data*. ProQuest.

Pereira, F., Mitchell, T., and Botvinick, M. (2009). Machine learning classifiers and fmri: A tutorial overview. *NeuroImage*, 45(1, Supplement 1):S199 – S209.

Proakis, J. and Manolakis, D. (2007). *Digital Signal Processing*. Pearson Prentice Hall.

Qiu, P. (2005). *Image Processing and Jump Regression Analysis*. Wiley series in probability and statistics. John Wiley.

Qiu, P. (2007). Jump surface estimation, edge detection, and image restoration. *Journal of the American Statistical Association*, 102:745–756.

Rao, C. R. and Mitra, S. K. (1971). *Generalized inverse of matrices and its applications*, volume 7. Wiley New York.

Rao, K. R., Yip, P., and Rao, K. R. (1990). *Discrete cosine transform: algorithms, advantages, applications*, volume 226. Academic press Boston.

Rothenberg, T. J. (1971). Identification in parametric models. *Econometrica*, 39(3):577–91.

Savas, B. and Eldén, L. (2007). Handwritten digit classification using higher order singular value decomposition. *Pattern recognition*, 40(3):993–1003.

Schlögl, A., Lee, F., Bischof, H., and Pfurtscheller, G. (2005). Characterization of four-class motor imagery eeg data for the bci-competition 2005. *Journal of neural engineering*, 2(4):L14.

Shen, D. and Davatzikos, C. (2002). Hammer: hierarchical attribute matching mechanism for elastic registration. *Medical Imaging, IEEE Transactions on*, 21(11):1421–1439.

Shen, H., Wang, L., Liu, Y., and Hu, D. (2010). Discriminative analysis of resting-state functional connectivity patterns of schizophrenia using low dimensional embedding of fmri. *Neuroimage*, 49(4):3110–3121.

Shinkareva, S. V., Ombao, H. C., Sutton, B. P., Mohanty, A., and Miller, G. A. (2006). Classification of functional brain images with a spatio-temporal dissimilarity map. *NeuroImage*, 33(1):63–71.

Stefanski, L. A. (2000). Measurement error models. *Journal of the American Statistical Association*, 95(452):1353–1358.

Stefanski, L. A. and Buzas, J. S. (1995). Instrumental variable estimation in binary regression measurement error models. *Journal of the American Statistical Association*, 90(430):pp. 541–550.

Stefanski, L. A. and Carroll, R. J. (1985). Covariate measurement error in logistic regression. *The Annals of Statistics*, 13(4):pp. 1335–1351.

Stefanski, L. A. and Carroll, R. J. (1987). Conditional scores and optimal scores for generalized linear measurement- error models. *Biometrika*, 74(4):pp. 703–716.

Stefanski, L. A. and Cook, J. R. (1995). Simulation-extrapolation: The measurement error jackknife. *Journal of the American Statistical Association*, 90(432):pp. 1247–1256.

Stonnington, C. M., Chu, C., Klöppel, S., Jack Jr, C. R., Ashburner, J., and Frackowiak, R. S. (2010). Predicting clinical scores from magnetic resonance scans in alzheimer's disease. *Neuroimage*, 51(4):1405–1413.

Sun, J., Tao, D., and Faloutsos, C. (2006). Beyond streams and graphs: dynamic ten-

sor analysis. In *Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 374–383. ACM.

Sun, J.-T., Zeng, H.-J., Liu, H., Lu, Y., and Chen, Z. (2005). Cubesvd: a novel approach to personalized web search. In *Proceedings of the 14th international conference on World Wide Web*, pages 382–390. ACM.

Tibshirani, R. (1996). Regression shrinkage and selection via the lasso. *J. Roy. Statist. Soc. Ser. B*, 58(1):267–288.

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

van der Vaart, A. W. (1998). *Asymptotic Statistics*, volume 3 of *Cambridge Series in Statistical and Probabilistic Mathematics*. Cambridge University Press, Cambridge.

Vapnik, V. (1996). *The nature of statistical learning theory*. springer.

Vasilescu, M. A. O. (2002). Human motion signatures: Analysis, synthesis, recognition. In *Pattern Recognition, 2002. Proceedings. 16th International Conference on*, volume 3, pages 456–460. IEEE.

Vasilescu, M. A. O. and Terzopoulos, D. (2002). Multilinear analysis of image ensembles: Tensorfaces. In *Computer VisionECCV 2002*, pages 447–460. Springer.

Vincent, L. and Soille, P. (1991). Watersheds in digital spaces: an efficient algorithm based on immersion simulations. *IEEE transactions on pattern analysis and machine intelligence*, 13(6):583–598.

Vlasic, D., Brand, M., Pfister, H., and Popović, J. (2005). Face transfer with multilinear models. In *ACM Transactions on Graphics (TOG)*, volume 24, pages 426–433. ACM.

Wang, H. and Ahuja, N. (2003). Facial expression decomposition. In *Computer Vision, 2003. Proceedings. Ninth IEEE International Conference on*, pages 958–965. IEEE.

Wang, L., Zhu, J., and Zou, H. (2008). Hybrid huberized support vector machines for microarray classification and gene selection. *Bioinformatics*, 24(3):412–419.

Wang, N., Carroll, R., and Liang, K.-Y. (1996). Quasilikelihood estimation in measurement error models with correlated replicates. *Biometrics*, pages 401–411.

Wang, X., Hutchinson, R., and Mitchell, T. M. (2003). Training fmri classifiers to detect cognitive states across multiple human subjects. In *In NIPS03*.

Weston, J., Elisseeff, A., Schölkopf, B., and Tipping, M. (2003). Use of the zero norm with linear models and kernel methods. *The Journal of Machine Learning Research*, 3:1439–1461.

Whitley, L. D., Beveridge, J. R., Guerra-Salcedo, C., and Graves, C. R. (1997). Messy genetic algorithms for subset feature selection. In *ICGA*, pages 568–575.

Yang, J. and Honavar, V. (1998). Feature subset selection using a genetic algorithm. In *Feature extraction, construction and selection*, pages 117–136. Springer.

Zhang, C.-H. (2010). Nearly unbiased variable selection under minimax concave penalty. *Ann. Statist.*, 38(2):894–942.

Zhang, D. and Shen, D. (2012). Multi-modal multi-task learning for joint prediction of multiple regression and classification variables in alzheimer's disease. *Neuroimage*, 59(2):895–907.

Zhang, D., Wang, Y., Zhou, L., Yuan, H., and Shen, D. (2011). Multimodal classification of alzheimer's disease and mild cognitive impairment. *Neuroimage*, 55(3):856–867.

Zhang, H. H., Ahn, J., Lin, X., and Park, C. (2006). Gene selection using support vector machines with non-convex penalty. *Bioinformatics*, 22(1):88–95.

Zhou, H., Armagan, A., and Dunson, D. (2011). Path following and empirical Bayes model selection for sparse regressions. *arXiv:1201.3528*.

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.

Zhu, J., Rosset, S., Hastie, T., and Tibshirani, R. (2003). 1-norm support vector machines. In *NIPS*, volume 15, pages 49–56.

Zou, H. and Hastie, T. (2005). Regularization and variable selection via the elastic net. *J. R. Stat. Soc. Ser. B Stat. Methodol.*, 67(2):301–320.

Zou, H. and Yuan, M. (2008). The fo-norm support vector machine. *Statistica Sinica*, 18:379–398.

APPENDICES

Appendix A

Proofs in Chapter 2

A.1 Proof of Lemma 2

We rewrite the array inner product

$$\begin{aligned}
& \langle \boldsymbol{B}, \boldsymbol{X} \rangle \\
= \; & \langle \boldsymbol{B}_{(d)}, \boldsymbol{X}_{(d)} \rangle \\
= \; & \langle \boldsymbol{B}_d \boldsymbol{G}_{(d)} (\boldsymbol{B}_D \otimes \cdots \otimes \boldsymbol{B}_{d+1} \otimes \boldsymbol{B}_{d-1} \otimes \cdots \otimes \boldsymbol{B}_1)^{\mathsf{T}}, \boldsymbol{X}_{(d)} \rangle \\
= \; & \langle \boldsymbol{G}_{(d)}, \boldsymbol{B}_d^{\mathsf{T}} \boldsymbol{X}_{(d)} (\boldsymbol{B}_D \otimes \cdots \otimes \boldsymbol{B}_{d+1} \otimes \boldsymbol{B}_{d-1} \otimes \cdots \otimes \boldsymbol{B}_1) \rangle \\
= \; & \langle \boldsymbol{G}_{(d)}, \tilde{\boldsymbol{X}}_{(d)} \rangle \\
= \; & \langle \boldsymbol{G}, \tilde{\boldsymbol{X}} \rangle,
\end{aligned}$$

where the second and fourth equalities follow from (2.3) and the third follows from the invariance of trace function under cyclic permutation.

A.2 Proof of Proposition 1

It is easy to see that the block relaxation algorithm monotonically increases the objective values, i.e., $\ell(\boldsymbol{\theta}^{(t+1)}) \geq \ell(\boldsymbol{\theta}^{(t)})$ for all $t \geq 0$. Therefore its global convergence property follows from the standard theory for monotone algorithms (de Leeuw, 1994; Lange, 2004, 2010). Specifically global convergence is guaranteed under the following conditions: (i) ℓ is coercive, (ii) the stationary points of ℓ are isolated, (iii) the algorithmic mapping is continuous, (iv) $\boldsymbol{\theta}$ is a fixed point of the algorithm if and only if it is a stationarity point of ℓ, and (v) $\ell(\boldsymbol{\theta}^{(t+1)}) \geq \ell(\boldsymbol{\theta}^{(t)})$ with equality if and only if $\theta^{(t)}$ is a fixed point of the algorithm. Condition (i) is guaranteed by the compactness of the set $\{\boldsymbol{\theta} : \ell(\boldsymbol{\theta}) \geq \ell(\boldsymbol{\theta}^{(0)})$. Condition (ii) is assumed. Condition (iii) follows from the strict concavity assumption and implicit function theorem. By Fermat's principle, $\boldsymbol{\theta} = (\boldsymbol{G}, \boldsymbol{B}_1, \ldots, \boldsymbol{B}_D)$ is a fixed point of the block relaxation algorithm if $D\ell(\boldsymbol{G}) = \mathbf{0}$ and $D\ell(\boldsymbol{B}_d) = \mathbf{0}$ for all d. Thus $\boldsymbol{\theta}$ is a fixed point if and only if it is a stationarity point of ℓ, i.e., condition (iv) is satisfied. Condition (v) follows from the monotonicity of the block relaxation algorithm. See (Zhou et al., 2013) for more details.

Local convergence follows from the classical Ostrowski theorem, which states that the algorithmic sequence $\boldsymbol{\theta}^{(t)}$ is local attracted to strictly local minimum $\boldsymbol{\theta}^{(\infty)}$ if the spectral radius of the differential of the algorithmic map $\rho[dM(\boldsymbol{\theta}^{(\infty)})]$ is strictly less than one. This follows from the strict concavity assumption of the block updates. See (Zhou et al., 2013) for more details.

A.3 Proof of Lemma 3

Assume $\boldsymbol{B}$ admits the Tucker decomposition (2.2). By (2.3),

$$\boldsymbol{B}_{(d)} = \boldsymbol{B}_d \boldsymbol{G}_{(d)} (\boldsymbol{B}_D \otimes \cdots \otimes \boldsymbol{B}_{d+1} \otimes \boldsymbol{B}_{d-1} \otimes \cdots \otimes \boldsymbol{B}_1)^\intercal.$$

Using the well-known fact that $\mathrm{vec}(\boldsymbol{XYZ}) = (\boldsymbol{Z}^\intercal \otimes \boldsymbol{X})\mathrm{vec}(\boldsymbol{Y})$,

$$\mathrm{vec}(\boldsymbol{B}_{(d)}) = [(\boldsymbol{B}_D \otimes \cdots \otimes \boldsymbol{B}_{d+1} \otimes \boldsymbol{B}_{d-1} \otimes \cdots \otimes \boldsymbol{B}_1)\boldsymbol{G}_{(d)}^\intercal \otimes \boldsymbol{I}_{p_d}]\mathrm{vec}(\boldsymbol{B}_d).$$

Thus by the chain rule we have

$$\begin{aligned}
\boldsymbol{J}_d &= D\boldsymbol{B}(\boldsymbol{B}_d) \\
&= D\boldsymbol{B}(\boldsymbol{B}_{(d)}) \cdot D\boldsymbol{B}_{(d)}(\boldsymbol{B}_d) \\
&= \boldsymbol{\Pi}_d \frac{\partial \mathrm{vec}(\boldsymbol{B}_{(d)})}{\partial \mathrm{vec}(\boldsymbol{B}_d)^\intercal} \\
&= \boldsymbol{\Pi}_d \{[\boldsymbol{B}_D \otimes \cdots \otimes \boldsymbol{B}_{d+1} \otimes \boldsymbol{B}_{d-1} \otimes \cdots \otimes \boldsymbol{B}_1)\boldsymbol{G}_{(d)}^\intercal] \otimes \boldsymbol{I}_{p_d}\}.
\end{aligned}$$

Again by chain rule, $D\eta(\boldsymbol{B}_d) = D\eta(\boldsymbol{B}) \cdot D\boldsymbol{B}(\boldsymbol{B}_d) = \mathrm{vec}(\boldsymbol{X})^\intercal \boldsymbol{J}_d$. For the derivative in $\boldsymbol{G}$, the duality lemma 2 implies $\langle \boldsymbol{B}, \boldsymbol{X} \rangle = \langle \boldsymbol{G}, \tilde{\boldsymbol{X}} \rangle$ for $\tilde{\boldsymbol{X}} = [\![\boldsymbol{X}; \boldsymbol{B}_1^\intercal, \ldots, \boldsymbol{B}_D^\intercal]\!]$. Then, by (2.3),

$$D\eta(\boldsymbol{G}) = \mathrm{vec}(\tilde{\boldsymbol{X}})^\intercal = \mathrm{vec}(\boldsymbol{X})^\intercal (\boldsymbol{B}_D \otimes \cdots \otimes \boldsymbol{B}_1).$$

Combining gives the gradient displayed in Lemma 3.

Next we consider the Hessian $d^2\eta$. Because $\boldsymbol{B}$ is linear in $\boldsymbol{G}$, the block $\boldsymbol{H}_{\boldsymbol{G},\boldsymbol{G}}$ vanishes.

For the block $\boldsymbol{H}_{\boldsymbol{B},\boldsymbol{B}}$, the $(i_d, r_d, i_{d'}, r_{d'})$-entry is

$$h_{(i_d,r_d),(i_{d'},r_{d'})} = \sum_{j_1,\ldots,j_D} x_{j_1,\ldots,j_D} \frac{\partial^2 b_{j_1,\ldots,j_D}}{\partial \beta_{i_d}^{(r)} \partial \beta_{i_{d'}}^{(r')}} = \sum_{j_1,\ldots,j_D} x_{j_1,\ldots,j_D} \sum_{s_1,\ldots,s_D} g_{s_1,\ldots,s_D} \frac{\partial^2 \beta_{j_1}^{(s_1)} \cdots \beta_{j_D}^{(s_D)}}{\partial \beta_{i_d}^{(r)} \partial \beta_{i_{d'}}^{(r')}}.$$

The second derivative in the summand is nonzero only if $j_d = i_d$, $j_{d'} = i_{d'}$, $s_d = r_d$, $s_{d'} = r_{d'}$, and $d \neq d'$. Therefore

$$h_{(i_d,r_d),(i_{d'},r_{d'})} = 1_{\{d \neq d'\}} \sum_{j_d = i_d, j_{d'} = i_{d'}} x_{j_1,\ldots,j_D} \sum_{s_d = r_d, s_{d'} = r_{d'}} g_{s_1,\ldots,s_D} \prod_{d'' \neq d,d'} \beta_{j_{d''}}^{(s_{d''})}.$$

The first sum is over $\prod_{d'' \neq d,d'} p_{d''}$ terms and the second term is over $\prod_{d'' \neq d,d'} R_{d''}$ terms. A careful inspection reveals that the sub-block $\boldsymbol{H}_{dd'}$ shares the same entries as the matrix

$$\boldsymbol{X}_{(dd')} (\boldsymbol{B}_D \otimes \cdots \otimes \boldsymbol{B}_{d+1} \otimes \boldsymbol{B}_{d-1} \otimes \cdots \otimes \boldsymbol{B}_{d'+1} \otimes \boldsymbol{B}_{d'-1} \otimes \cdots \otimes \boldsymbol{B}_1) \boldsymbol{G}_{(dd')}^{\mathsf{T}}.$$

Finally, for the $\boldsymbol{H}_{\boldsymbol{G},\boldsymbol{B}}$ block, the $\{(r_1, \ldots, r_D), (i_d, r_d)\}$-entry is

$$\begin{aligned}
h_{(r_1,\ldots,r_D),(i_d,s_d)} &= \sum_{j_1,\ldots,j_D} x_{j_1,\ldots,j_D} \frac{\partial^2 b_{j_1,\ldots,j_D}}{\partial g_{r_1,\ldots,r_D} \partial \beta_{i_d}^{(s_d)}} \\
&= \sum_{j_1,\ldots,j_D} x_{j_1,\ldots,j_D} \sum_{t_1,\ldots,t_D} \frac{\partial^2 g_{t_1,\ldots,t_D} \beta_{j_1}^{(t_1)} \cdots \beta_{j_D}^{(t_D)}}{\partial g_{r_1,\ldots,r_D} \partial \beta_{i_d}^{(s_d)}} \\
&= \sum_{j_1,\ldots,j_D} x_{j_1,\ldots,j_D} \frac{\partial \beta_{j_1}^{(r_1)} \cdots \beta_{j_D}^{(r_D)}}{\partial \beta_{i_d}^{(s_d)}} \\
&= 1_{\{r_d = s_d\}} \sum_{j_d = i_d} x_{j_1,\ldots,j_D} \prod_{d' \neq d} \beta_{j_{d'}}^{(r_{d'})},
\end{aligned}$$

where the sum is over $\prod_{d' \neq d} p_{d'}$ terms. The sub-block $\boldsymbol{H}_d \in \mathbb{R}^{\prod_d R_d \times p_d R_d}$ has at most $p_d \prod_d R_d$ nonzero entries. A close inspection suggests that the nonzero entries coincide

with those in the matrix

$$\boldsymbol{X}_{(d)}(\boldsymbol{B}_D \otimes \cdots \otimes \boldsymbol{B}_{d+1} \otimes \boldsymbol{B}_{d-1} \otimes \cdots \otimes \boldsymbol{B}_1).$$

A.4 Proof of Proposition 2

Since $\mu = b'(\theta)$, $d\mu/d\theta = b''(\theta) = \sigma^2/a(\phi)$ and

$$\begin{aligned}\nabla \ell(\boldsymbol{G}, \boldsymbol{B}_1, \ldots, \boldsymbol{B}_D) &= \frac{y - b'(\theta)}{a(\phi)} \frac{d\theta}{d\mu} \frac{d\mu}{d\eta} \nabla \eta(\boldsymbol{G}, \boldsymbol{B}_1, \ldots, \boldsymbol{B}_D) \\ &= \frac{(y-\mu)\mu'(\eta)}{\sigma^2} [\boldsymbol{B}_D \otimes \cdots \otimes \boldsymbol{B}_1 \boldsymbol{J}_1 \ldots \boldsymbol{J}_D]^{\mathsf{T}} \mathrm{vec}(\boldsymbol{X})\end{aligned}$$

by Lemma 3. Further differentiating shows

$$\begin{aligned}& d^2 \ell(\boldsymbol{G}, \boldsymbol{B}_1, \ldots, \boldsymbol{B}_D) \\ =& -\frac{1}{\sigma^2} \nabla \mu(\boldsymbol{G}, \boldsymbol{B}_1, \ldots, \boldsymbol{B}_D) d\mu(\boldsymbol{G}, \boldsymbol{B}_1, \ldots, \boldsymbol{B}_D) + \frac{y-\mu}{\sigma^2} d^2 \mu(\boldsymbol{G}, \boldsymbol{B}_1, \ldots, \boldsymbol{B}_D) \\ =& -\frac{[\mu'(\eta)]^2}{\sigma^2} ([\boldsymbol{B}_D \otimes \cdots \otimes \boldsymbol{B}_1 \boldsymbol{J}_1 \ldots \boldsymbol{J}_D]^{\mathsf{T}} \mathrm{vec}(\boldsymbol{X}))([\boldsymbol{B}_D \otimes \cdots \otimes \boldsymbol{B}_1 \boldsymbol{J}_1 \ldots \boldsymbol{J}_D]^{\mathsf{T}} \mathrm{vec}(\boldsymbol{X}))^{\mathsf{T}} \\ & + \frac{(y-\mu)\theta''(\eta)}{\sigma^2} ([\boldsymbol{B}_D \otimes \cdots \otimes \boldsymbol{B}_1 \boldsymbol{J}_1 \ldots \boldsymbol{J}_D]^{\mathsf{T}} \mathrm{vec}(\boldsymbol{X})) \cdot \\ & ([\boldsymbol{B}_D \otimes \cdots \otimes \boldsymbol{B}_1 \boldsymbol{J}_1 \ldots \boldsymbol{J}_D]^{\mathsf{T}} \mathrm{vec}(\boldsymbol{X}))^{\mathsf{T}} + \frac{(y-\mu)\theta'(\eta)}{\sigma^2} d^2 \eta(\boldsymbol{B}).\end{aligned}$$

It is easy to see that

$$\begin{aligned}\mathbf{E}[\nabla \ell(\boldsymbol{G}, \boldsymbol{B}_1, \ldots, \boldsymbol{B}_D)] &= \mathbf{0} \\ \mathbf{E}[-d^2 \ell(\boldsymbol{G}, \boldsymbol{B}_1, \ldots, \boldsymbol{B}_D)] &= \boldsymbol{I}(\boldsymbol{G}, \boldsymbol{B}_1, \ldots, \boldsymbol{B}_D),\end{aligned}$$

thus (2.8) follows.

A.5 Proof of Proposition 3

The proof follows from a classical result (Rothenberg, 1971) that states that, if θ_0 be a regular point of the information matrix $I(\theta)$, then θ_0 is locally identifiable if and only if $I(\theta_0)$ is nonsingular. The regularity assumptions are satisfied by Tucker regression model: (1) the parameter space $\mathcal{B}$ is open, (2) the density $p(y, \boldsymbol{x}|\boldsymbol{B})$ is proper for all $\boldsymbol{B} \in \mathcal{B}$, (3) the support of the density $p(y, \boldsymbol{x}|\boldsymbol{B})$ is same for all $\boldsymbol{B} \in \mathcal{B}$, (4) the log density $\ell(\boldsymbol{B}|y, \boldsymbol{x}) = \ln p(y, \boldsymbol{x}|\boldsymbol{B})$ is continuously differentiable, and (5) the information matrix

$$\begin{aligned}\boldsymbol{I}(\boldsymbol{B}) =&[\boldsymbol{B}_D \otimes \cdots \otimes \boldsymbol{B}_1 \, \boldsymbol{J}_1 \ldots \boldsymbol{J}_D]^\intercal \left[\sum_{i=1}^{n} \frac{\mu'(\eta_i)^2}{\sigma_i^2} \mathrm{vec}(\boldsymbol{x}_i)\mathrm{vec}(\boldsymbol{x}_i)^\intercal\right] \cdot \\ &[\boldsymbol{B}_D \otimes \cdots \otimes \boldsymbol{B}_1 \, \boldsymbol{J}_1 \ldots \boldsymbol{J}_D]\end{aligned}$$

is continuous in $\boldsymbol{B}$ by Proposition 2. Therefore $\boldsymbol{B} \in \mathcal{B}$ is locally identifiable if and only if $\boldsymbol{I}(\boldsymbol{B})$ is nonsingular.

A.6 Proof of Theorem 1

The asymptotics for tensor regression follow from the standard theory of M-estimation. The key observation is that the nonlinear part of tensor model (2.4) is a degree-$(D+1)$ polynomial of parameters $\boldsymbol{G}$ and $\boldsymbol{B}_d$ and the collection of polynomials $\{\langle \boldsymbol{B}, \boldsymbol{X} \rangle, \boldsymbol{B} \in \mathcal{B}\}$ form a Vapnik-Červonenkis (VC) class. Then the classical uniform convergence theory applies (van der Vaart, 1998). The arguments in (Zhou et al., 2013) extends the classical argument for GLM (van der Vaart, 1998, Example 5.40) to the CP tensor regression model. The same proof also applies to Tucker model with little changes and omitted here.

For asymptotic normality, we need to establish that the log-likelihood function of Tucker regression model is quadratic mean differentiable (q.m.d.) (Lehmann and Romano, 2005). By a well-known result (Lehmann and Romano, 2005, Theorem 12.2.2) or (van der Vaart, 1998, Lemma 7.6), it suffices to verify that the density is continuously differentiable in parameter for μ-almost all x and that the Fisher information matrix exists and is continuous. The derivative of density is

$$\nabla p(\boldsymbol{B}_1, \ldots, \boldsymbol{B}_D) = \nabla e^{\ell(\boldsymbol{B}_1, \ldots, \boldsymbol{B}_D)} = p(\boldsymbol{B}_1, \ldots, \boldsymbol{B}_D) \nabla \ell(\boldsymbol{B}_1, \ldots, \boldsymbol{B}_D),$$

which is well-defined and continuous by Proposition 2. The same proposition shows that the information matrix exists and is continuous. Therefore the Tucker regression model is q.m.d. and the asymptotic normality follows from the classical result for q.m.d. families (van der Vaart, 1998, Theorem 5.39).

Appendix B

Proofs in Chapter 3

B.1 Derivation of (3.10)

We start by looking at the distribution of $Y, \bar{W}$ given X and Z

$$f_{(Y,\bar{W})|(X,Z)}(y, \bar{w}|x, z) \propto \exp\left\{\frac{y\eta - b(\eta)}{a(\phi)} + c(y, \phi) - \frac{1}{2}(\bar{w} - x)^{\tau}(\mathbf{\Omega}_0/m)^{-1}(\bar{w} - x)\right\},$$

where $a(\cdot)$, $b(\cdot)$, $c(\cdot)$ and η can be found in (3.2). Plugging $\eta = \alpha + \gamma^{\tau} z + \beta^{\tau} x$, one can find

$$\begin{aligned} f_{(Y,\bar{W})|(X,Z)}(y, \bar{w}|x, z) \quad \propto \quad & \exp\left\{c(y, \phi) - \frac{1}{2}\bar{w}^{\tau}(\mathbf{\Omega}_0/m)^{-1}\bar{w} + \frac{y(\alpha + \gamma^{\tau} z)}{a(\phi)} - \frac{b(\eta)}{a(\phi)}\right\} \\ & \cdot \exp\left\{-\frac{1}{2}x^{\tau}(\mathbf{\Omega}_0/m)^{-1}x + x^{\tau}(\mathbf{\Omega}_0/m)^{-1}(\bar{w} + \frac{y(\mathbf{\Omega}_0/m)\beta}{a(\phi)})\right\}. \end{aligned}$$

As is defined, we let $\mathbf{\Omega}_0^* = \frac{\mathbf{\Omega}_0}{ma(\phi)}$. Then let $\delta = \bar{w} + y\mathbf{\Omega}_0^*\beta$ and $\mu = \frac{\alpha + \gamma^{\tau} z + \delta^{\tau}\beta}{a(\phi)}$. According to the argument in Stefanski and Carroll (1987), we treat Δ as a sufficient statistic for

X assuming other model parameters are known. Therefore we have

$$\begin{aligned} f_{(Y,\Delta)|Z}(y,\delta|z) &\propto \exp\left\{c(y,\phi) - \frac{1}{2}(\delta - y\mathbf{\Omega}_0^*\beta)^\intercal(\mathbf{\Omega}_0/m)^{-1}(\delta - y\mathbf{\Omega}_0^*\beta) + \right. \\ &\qquad \left. \frac{y(\alpha+\gamma^\intercal z)}{a(\phi)} - \frac{b(\eta)}{a(\phi)}\right\} \\ &\propto \exp\left\{c(y,\phi) - \frac{1}{2}\delta^\intercal\frac{(\mathbf{\Omega}_0^*)^{-1}}{a(\phi)}\delta - \frac{1}{2}y^2\beta^\intercal\mathbf{\Omega}_0^*\beta/a(\phi) + y\mu\right\}. \end{aligned}$$

Therefore the conditional distribution of Y given (Δ, Z) is given

$$\begin{aligned} f_{Y|(\Delta,Z)}(y|(\delta,z)) &= \frac{f_{(Y,\Delta)|Z}(y,\delta|z)}{\int f_{(Y,\Delta)|Z}(y,\delta|z)dy} \\ &= \exp\left\{y\mu - \log[S(\mu,\beta,\phi)] + c(y,\phi) - \frac{1}{2}y^2\frac{\beta^\intercal\mathbf{\Omega}_0^*\beta}{a(\phi)}\right\}, \end{aligned}$$

where $S(\mu,\beta,\phi) = \int \exp\left\{y\mu + c(y,\phi) - \frac{1}{2}y^2\frac{\beta^\intercal\mathbf{\Omega}_0^*\beta}{a(\phi)}\right\}dy$. Then (3.10) is obtained.

B.2 Derivations from (3.13) to (3.14)-(3.17)

Here we derive that every solution to (3.14) is also a solution to

$$\sum_{i=1}^{n}\psi_s(y_i, z_i, \bar{w}_i, \alpha, \beta, \gamma, \sigma^2) = 0,$$

where ψ_s is given by (3.13). First we use a substitution that $\nu = \alpha + \gamma^\intercal z$. Notice the fact that $(\boldsymbol{I} + \boldsymbol{\Omega}_0^* \beta\beta^\intercal)^{-1} = \boldsymbol{I} - \frac{\boldsymbol{\Omega}_0^* \beta\beta^\intercal}{1+\beta^\intercal \boldsymbol{\Omega}_0^* \beta}$. Then

$$
\begin{aligned}
& y - \alpha - \gamma^\intercal z - \beta^\intercal \delta^* = y - \nu - \beta^\intercal \delta^* \\
=& y - \nu - \beta^\intercal (\boldsymbol{I} + \boldsymbol{\Omega}_0^* \beta\beta^\intercal)^{-1} \{\delta - (\alpha + \gamma^\intercal z)\boldsymbol{\Omega}_0^* \beta\} \\
=& y - \nu - \beta^\intercal \left(\boldsymbol{I} - \frac{\boldsymbol{\Omega}_0^* \beta\beta^\intercal}{1 + \beta^\intercal \boldsymbol{\Omega}_0^* \beta} \right) (\delta - \nu \boldsymbol{\Omega}_0^* \beta) \\
=& y - \left(\nu + \frac{\beta^\intercal}{1 + \beta^\intercal \boldsymbol{\Omega}_0^* \beta} (\delta - \nu \boldsymbol{\Omega}_0^* \beta) \right) \\
=& y - \frac{\nu + \beta^\intercal \delta}{1 + \beta^\intercal \boldsymbol{\Omega}_0^* \beta}.
\end{aligned}
$$

Therefore the upper two equations in $\sum_{i=1}^n \psi_s(y_i, z_i, \bar{w}_i, \alpha, \beta, \gamma, \sigma^2) = 0$ just corresponds to the first two equations of (3.14). Also in (3.13), the third term can be simplified to $\frac{(y-\xi)(\delta - 2\xi \boldsymbol{\Omega}_0^* \beta)}{\sigma^2}$ if we plug the fourth term in. Then we would like to show that the solution to (3.14) is also a solution to $\sum_{i=1}^n \frac{(y_i - \xi_i)(\delta_i - 2\xi_i \boldsymbol{\Omega}_0^* \beta)}{\sigma^2} = \boldsymbol{0}$. First we simplify δ^* by

$$
\begin{aligned}
\delta^* &= (\boldsymbol{I} + \boldsymbol{\Omega}_0^* \beta\beta^\intercal)^{-1} (\delta - \nu \boldsymbol{\Omega}_0^* \beta) \\
&= \left(\boldsymbol{I} - \frac{\boldsymbol{\Omega}_0^* \beta\beta^\intercal}{1 + \beta^\intercal \boldsymbol{\Omega}_0^* \beta} \right) (\delta - \nu \boldsymbol{\Omega}_0^* \beta) \\
&= \delta - \nu \boldsymbol{\Omega}_0^* \beta - \frac{\boldsymbol{\Omega}_0^* \beta\beta^\intercal \delta}{1 + \beta^\intercal \boldsymbol{\Omega}_0^* \beta} + \frac{\nu \boldsymbol{\Omega}_0^* \beta\beta^\intercal \boldsymbol{\Omega}_0^* \beta}{1 + \beta^\intercal \boldsymbol{\Omega}_0^* \beta} \\
&= \delta - \left(\nu \boldsymbol{\Omega}_0^* \beta + \frac{\boldsymbol{\Omega}_0^* \beta\beta^\intercal \delta}{1 + \beta^\intercal \boldsymbol{\Omega}_0^* \beta} - \frac{\nu \boldsymbol{\Omega}_0^* \beta\beta^\intercal \boldsymbol{\Omega}_0^* \beta}{1 + \beta^\intercal \boldsymbol{\Omega}_0^* \beta} \right) \\
&= \delta - \frac{\nu \boldsymbol{\Omega}_0^* \beta + \boldsymbol{\Omega}_0^* \beta\beta^\intercal \delta}{1 + \beta^\intercal \boldsymbol{\Omega}_0^* \beta} \\
&= \delta - \frac{(\nu + \beta^\intercal \delta) \boldsymbol{\Omega}_0^* \beta}{1 + \beta^\intercal \boldsymbol{\Omega}_0^* \beta} \\
&= \delta - \xi \boldsymbol{\Omega}_0^* \beta.
\end{aligned}
$$

Also $\delta^* = (\boldsymbol{I} + \boldsymbol{\Omega}_0^*\beta\beta^\intercal)^{-1}(\delta - \nu\boldsymbol{\Omega}_0^*\beta)$, then $\delta = (\boldsymbol{I} + \boldsymbol{\Omega}_0^*\beta\beta^\intercal)\delta^* + \nu\boldsymbol{\Omega}_0^*\beta$. Hence

$$\begin{aligned}
\sum_{i=1}^{n}(y_i - \xi_i)(\delta_i - 2\xi_i\boldsymbol{\Omega}_0^*\beta) &= \sum_{i=1}^{n}(y_i - \xi_i)(\delta_i - 2(\delta_i - \delta_i^*)) \\
&= \sum_{i=1}^{n}(y_i - \xi_i)(2\delta_i^* - \delta_i) \\
&= \sum_{i=1}^{n}(y_i - \xi_i)\left[2\delta_i^* - (\boldsymbol{I} + \boldsymbol{\Omega}_0^*\beta\beta^\intercal)\,\delta_i^* - \nu_i\boldsymbol{\Omega}_0^*\beta\right] \\
&= \sum_{i=1}^{n}(y_i - \xi_i)\left(\boldsymbol{I} - \boldsymbol{\Omega}_0^*\beta\beta^\intercal\right)\delta_i^* - \sum_{i=1}^{n}(y_i - \xi_i)\nu_i\boldsymbol{\Omega}_0^*\beta.
\end{aligned}$$

One should notice $y - \xi = y - \alpha - \gamma^\intercal z - \beta^\intercal\delta^*$ and $\nu = \alpha + \gamma^\intercal z$, therefore if (3.14) holds, $\sum_{i=1}^{n}(y_i - \xi_i)(\delta_i - 2\xi_i\boldsymbol{\Omega}_0^*\beta) = \mathbf{0}$ holds.

We next would like to show that (3.14) is equivalent to (3.15)-(3.17). First follow what is shown above

$$\begin{aligned}
y - \alpha - \gamma^\intercal z - \beta^\intercal\delta^* &= y - \frac{\nu + \beta^\intercal\delta}{1 + \beta^\intercal\boldsymbol{\Omega}_0^*\beta} \\
&= y - \frac{\nu + \beta^\intercal(\bar{w} + y\boldsymbol{\Omega}_0^*\beta)}{1 + \beta^\intercal\boldsymbol{\Omega}_0^*\beta} \\
&= \frac{y - \nu - \beta^\intercal\bar{w}}{1 + \beta^\intercal\boldsymbol{\Omega}_0^*\beta} \\
&= \frac{y - \alpha - \gamma^\intercal z - \beta^\intercal\bar{w}}{1 + \beta^\intercal\boldsymbol{\Omega}_0^*\beta}.
\end{aligned}$$

Then the first two equations of (3.14) can be simplified to

$$\frac{1}{1 + \beta^\intercal\boldsymbol{\Omega}_0^*\beta}\sum_{i=1}^{n}(y_i - \alpha - \gamma^\intercal z_i - \beta^\intercal\bar{w}_i)\begin{pmatrix} 1 \\ \\ z_i \end{pmatrix} = \mathbf{0},$$

which is equivalent to (3.15). We turn to the third messy equation:

$$\begin{aligned}
&\sum_{i=1}^{n}(y_i-\alpha-\gamma^{\intercal}z_i-\beta^{\intercal}\delta_i^*)\delta_i^*\\
=&\sum_{i=1}^{n}(y_i-\nu_i-\beta^{\intercal}\delta_i^*)\delta_i^*\\
=&(\boldsymbol{I}+\boldsymbol{\Omega}_0^*\beta\beta^{\intercal})^{-1}\sum_{i=1}^{n}(y_i-\nu_i-\beta^{\intercal}\delta_i^*)[\bar{w}_i+(y_i-\nu_i)\boldsymbol{\Omega}_0^*\beta].
\end{aligned}$$

Noticing $\sum_{i=1}^{n}(y_i-\nu_i-\beta^{\intercal}\delta_i^*)\nu_i=0$, we have

$$\begin{aligned}
&\sum_{i=1}^{n}(y_i-\alpha-\gamma^{\intercal}z_i-\beta^{\intercal}\delta_i^*)\delta_i^*\\
=&(\boldsymbol{I}+\boldsymbol{\Omega}_0^*\beta\beta^{\intercal})^{-1}\sum_{i=1}^{n}(y_i-\nu_i-\beta^{\intercal}\delta_i^*)[\bar{w}_i+y_i\boldsymbol{\Omega}_0^*\beta]\\
=&\frac{(\boldsymbol{I}+\boldsymbol{\Omega}_0^*\beta\beta^{\intercal})^{-1}}{1+\beta^{\intercal}\boldsymbol{\Omega}_0^*\beta}\sum_{i=1}^{n}(y_i-\nu_i-\beta^{\intercal}\bar{w}_i)(\bar{w}_i+y_i\boldsymbol{\Omega}_0^*\beta).
\end{aligned}$$

From (3.15), we can express ν_i by $\nu_i=(1\ z_i^{\intercal})(\boldsymbol{M}_{\mathbf{1}Z}^{\intercal}\boldsymbol{M}_{\mathbf{1}Z})^{-1}\boldsymbol{M}_{\mathbf{1}Z}^{\intercal}(\boldsymbol{y}-\boldsymbol{M}_{\bar{W}}\beta), i=1,\ldots,n$. Therefore

$$\begin{aligned}
&\sum_{i=1}^{n}(y_i-\alpha-\gamma^{\intercal}z_i-\beta^{\intercal}\delta_i^*)\delta_i^*\\
=&\frac{(\boldsymbol{I}+\boldsymbol{\Omega}_0^*\beta\beta^{\intercal})^{-1}}{1+\beta^{\intercal}\boldsymbol{\Omega}_0^*\beta}\sum_{i=1}^{n}[y_i-(1\ z_i^{\intercal})(\boldsymbol{M}_{\mathbf{1}Z}^{\intercal}\boldsymbol{M}_{\mathbf{1}Z})^{-1}\boldsymbol{M}_{\mathbf{1}Z}^{\intercal}(\boldsymbol{y}-\boldsymbol{M}_{\bar{W}}\beta)-\beta^{\intercal}\bar{w}_i](\bar{w}_i+y_i\boldsymbol{\Omega}_0^*\beta)\\
=&\frac{(\boldsymbol{I}+\boldsymbol{\Omega}_0^*\beta\beta^{\intercal})^{-1}}{1+\beta^{\intercal}\boldsymbol{\Omega}_0^*\beta}\sum_{i=1}^{n}[y_i-\tilde{y}_i-(\bar{w}_i^{\intercal}-\tilde{w}_i^{\intercal})\beta](\bar{w}_i+y_i\boldsymbol{\Omega}_0^*\beta),
\end{aligned}$$

where $\tilde{y}_i=(1\ z_i^{\intercal})(\boldsymbol{M}_{\mathbf{1}Z}^{\intercal}\boldsymbol{M}_{\mathbf{1}Z})^{-1}\boldsymbol{M}_{\mathbf{1}Z}^{\intercal}\boldsymbol{y}$; $\tilde{w}_i=((1\ z_i^{\intercal})(\boldsymbol{M}_{\mathbf{1}Z}^{\intercal}\boldsymbol{M}_{\mathbf{1}Z})^{-1}\boldsymbol{M}_{\mathbf{1}Z}^{\intercal}\boldsymbol{M}_{\bar{W}})^{\intercal}$ is a $p\times 1$ vector. Then (3.16) can be obtained by expanding the algebraic expression above.