TENSOR REGRESSION AND TENSOR TIME SERIES ANALYSES FOR HIGH DIMENSIONAL DATA

by

Herath Mudiyanselage Wiranthe Bandara Herath

B.S., University of Peradeniya, 2015

A Thesis
Submitted in Partial Fulfillment of the Requirements for the
Master of Science Degree

Department of Mathematics
in the Graduate School
Southern Illinois University Carbondale
August, 2019

THESIS APPROVAL

TENSOR REGRESSION AND TENSOR TIME SERIES ANALYSES FOR HIGH DIMENSIONAL DATA

By

Herath Mudiyanselage Wiranthe Bandara Herath

A Thesis Submitted in Partial

Fulfillment of the Requirements

for the Degree of

Master of Science

in the field of Mathematics

Approved by:

Dr. S. Yaser Samadi, Chair

Dr. David J. Olive

Dr. Harry R. Hughes

Graduate School
Southern Illinois University Carbondale
June 27, 2019

AN ABSTRACT OF THE THESIS OF

Herath Mudiyanselage Wiranthe Bandara Herath, for the Master of Science degree in MATHEMATICS, presented on June 27, 2019, at Southern Illinois University Carbondale.

TITLE: TENSOR REGRESSION AND TENSOR TIME SERIES ANALYSES FOR HIGH DIMENSIONAL DATA

MAJOR PROFESSOR: Dr. S. Yaser Samadi

Many real data are naturally represented as a multidimensional array called a tensor. In classical regression and time series models, the predictors and covariate variables are considered as a vector. However, due to high dimensionality of predictor variables, these types of models are inefficient for analyzing multidimensional data. In contrast, tensor structured models use predictors and covariate variables in a tensor format. Tensor regression and tensor time series models can reduce high dimensional data to a low dimensional framework and lead to efficient estimation and prediction. In this thesis, we discuss the modeling and estimation procedures for both tensor regression models and tensor time series models. The results of simulation studies and a numerical analysis are provided.

ACKNOWLEDGMENTS

I would like to express my sincere gratitude and special thanks to my advisor, Dr. S. Yaser Samadi for his support, guidance, motivation and encouragement throughout my research and writing of this thesis. Without his persistent help I would not have had the courage to complete my work.

I would also like to thank Dr. David J. Olive and Dr. H. R. Hughes for being the members of my committee and for their comments and suggestions for further improvement of my thesis.

I am very grateful to the faculty members of the Department of Mathematics, SIUC for their instruction and care over the past two years. Also, I am grateful for faculty members of the University of Peradeniya, Sri Lanka and all my teachers in schools who guided me throughout the years.

I would like to thank my fellow graduate students who have been helping and supporting in various ways.

Last but not least, I want to thank my family for their love and support throughout in my all endeavors.

TABLE OF CONTENTS

LIST OF FIGURES

CHAPTER 1

INTRODUCTION

1.1 TENSOR REGRESSION-MOTIVATIONS

Classical regression models use the predictors and covariate variables as a vector to identify the effects of covariates on a given response variable. However, due to the nature of high dimensional data these types of models are inefficient for analyzing array structured data. For example, consider an image array that we use for medical studies. Converting an image array into a vector and learning the associations is unsatisfactory. Because, it would lead to a very high dimensional problem and require large number of regression parameters to be estimated. Also, when converting image array to a vector it would destroy the inherent structure of the data and lose important information that are needed for the study. In contrast, tensor structured models use covariates and predictor variables in a multidimensional array or tensor format. Tensor regression models can reduce high dimensional data to a low dimensional framework and lead to efficient estimation and prediction. These models can be used to recover many imaging signals. For instance, electroencephalography (EEG) image signal can be considered as a second order tensor, anatomical magnetic resonance images (MRI) can be considered as a third order tensor, functional magnetic resonance images (fMRI) can be considered as a fourth order tensor, and so on.

1.2 TENSOR TIME SERIES-MOTIVATIONS

In classical time series models, lag variables and covariates are considered as vectors. But, these types of models are inefficient for analyzing array valued data. In contrast, tensor structured time series models use covariates and lag variables in a tensor format. For example, spatiotemporal atmospheric data in climate modeling can be modeled as a tensor time series with dimensions of $a \times b \times c$, where a, b and c are denoted as the

numbers of latitude, numbers of longitude, and the numbers of elevation grid points. In the analysis of climate data, we can use tensor time series models to predict whether or not the ocean temperature increases. In general, we can say tensor time series models are very useful to reduce high dimensional data to a low dimensional framework and lead to efficient estimations and predictions.

1.3 THESIS OUTLINE

Chapter 2 contains tensor preliminaries. In Chapter 3, we discuss a tensor regression model, parameter estimation of the tensor regression model, and we conduct simulation studies and numerical analysis to illustrate the application of tensor regression model. In Chapter 4, we introduce a tensor time series model by using a multilinear dynamical system. Moreover, we provide a parameter estimation procedure for the tensor time series model and we present the simulation results in section 4.8. Finally, the Appendices contain all proofs.

CHAPTER 2

TENSOR PRELIMINARIES

This chapter contains the formal definition, and several basic properties and operations of tensors that are useful for understanding tensor regression and tensor time series in the remaining chapters of the thesis. It begins with basic definitions and notations of tensors. Then, we explain about the tensor decomposition which is also called tensor factorization. We also use the tensor decomposition that provides multilinear models. Tensor decomposition is a very useful tool to decompose a tensor into a low dimensional framework.

2.1 BASIC PROPERTIES OF TENSOR

2.1.1 Definitions and Notations

Tensors are multidimensional arrays and can be considered as generalizations of vectors and matrices. The *order* of a tensor is the number of dimensions, ways, or modes [10]. As an example, a vector is a first-order tensor, a matrix is a second-order tensor, and tensors of order three or higher are referred as higher-order tensors. Throughout this thesis, higher-order tensors are denoted by calligraphy letters as $\mathcal{X} \in \mathbb{R}^{I_1 \times \cdots \times I_D}$, where D is the order of the tensor $\mathcal{X}$, I_d is the dimension of mode d, d $= 1, \ldots,$ D. Matrices are represented by boldface capital letters like $\mathbf{X} \in \mathbb{R}^{I_1 \times I_2}$, while the vectors are denoted by boldface low-case letters like $\mathbf{x} \in \mathbb{R}^{I_1}$. The ith entry of a vector $\mathbf{x}$ is denoted by x_i and the (i, j)th entry of a matrix $\mathbf{X}$ is denoted by x_{ij} . Also, we denote the entry $(i_1, i_2, \ldots, i_D)$ of D-order tensor $\mathcal{X} \in \mathbb{R}^{I_1 \times \cdots \times I_D}$ as $x_{i_1 i_2 \ldots i_D}$. We use the "$\bullet$" symbol to the coordinates of tensor that are not fixed. For instance, consider the tensor $\mathcal{X} \in \mathbb{R}^{I_1 \times I_2 \times I_3 \times I_4}$. Then, $\mathcal{X}_{i_1 i_2 i_3 i_4}$ is a scalar, where as $\mathcal{X}_{i_1 i_2 \bullet\bullet}$ is a matrix of dimension $I_3 \times I_4$. As an example of a third order tensor $\mathcal{X} \in \mathbb{R}^{I_1 \times I_2 \times I_3}$ is demonstrated in Figure 2.1.

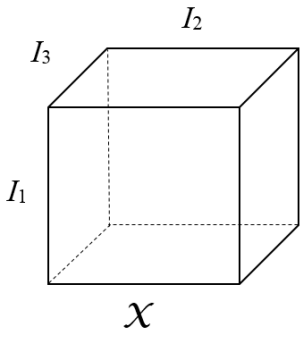

Figure 2.1: Third-order tensor

2.1.2 The Norm of a Tensor

The Frobenius norm of tensor $\mathcal{X} \in \mathbb{R}^{I_1 \times \cdots \times I_D}$ is defined as the square root of the sum of the squares of all of its elements, and denoted by

$$\| \mathcal{X} \| = \sqrt{\sum_{i_1=1}^{I_1} \sum_{i_2=1}^{I_2} \cdots \sum_{i_d=1}^{I_D} x_{i_1 i_2 \ldots i_D}^2}.$$

2.1.3 Tensor Vectorization

Vectorization of a tensor is the process of converting a tensor to a vector. The operator vec($\mathcal{X}$) stacks the entries of a D-dimensional tensor $\mathcal{X} \in \mathbb{R}^{I_1 \times \cdots \times I_D}$ into a column vector with dimension $I_1 \ldots I_D$. More precisely, the tensor element $x_{i_1 \cdots i_D}$ maps to the kth element of vec($\mathcal{X}$), where $k = 1 + \sum_{d=1}^{D} \prod_{d'=1}^{d-1} I_{d'}(i_d - 1)$. For an example, when D = 2, the matrix entry $x_{i_1 i_2}$ maps to the position (index) $k = 1 + i_1 - 1 + I_1(i_2 - 1) = i_1 + I_1(i_2 - 1)$, and when D = 3, the matrix element $x_{i_1 i_2 i_3}$ maps to the position (index) $k = 1 + i_1 - 1 + I_1(i_2 - 1) + I_1 I_2(i_3 - 1) = i_1 + I_1(i_2 - 1) + I_1 I_2(i_3 - 1)$. For instance, consider $\mathcal{X} \in \mathbb{R}^{2 \times 3 \times 2}$ as follows

$$\mathcal{X}_{\bullet\bullet 1} = \begin{pmatrix} a & c & e \\ b & d & f \end{pmatrix}, \mathcal{X}_{\bullet\bullet 2} = \begin{pmatrix} g & i & k \\ h & j & l \end{pmatrix}$$

$$\text{then, } \text{vec}(\mathcal{X}) = (a\,b\,c\,d\,e\,f\,g\,h\,i\,j\,k\,l)^T.$$

2.1.4 Tensor Matricization

Matricization of a tensor is the process of converting a tensor to a matrix, and it is denoted by mat $(\mathcal{X})$. This operation is also called the operation of unfolding or flattening a tensor, see Kolda and Bader (2009)[10]. The mode-d matricization of $\mathcal{X} \in \mathbb{R}^{I_1 \times I_2 \times \ldots I_D}$ is represented as $\mathbf{X}_{(d)}$. The mode-d matricization maps a tensor $\mathcal{X}$ into a $I_d \times \prod_{d' \neq d} I'_d$ matrix such that the tensor element $(i_1, i_2, \ldots, i_D)$ maps to the (i_d, k) entry of $\mathbf{X}_{(d)}$, where $k = 1 + \sum_{d' \neq d}(i_{d'} - 1) \prod_{d'' < d', d'' \neq d} I_{d''}$. For instance, consider a tensor of size $3 \times 4 \times 5$. It can be flatten to 5×12 matrix $(\mathbf{X}_{(3)})$, 3×20 matrix $(\mathbf{X}_{(1)})$, or 4×15 matrix $(\mathbf{X}_{(2)})$. Now, consider a fourth order tensor $\mathcal{X} \in \mathbb{R}^{2\times2\times2\times2}$ as follows

$$\mathcal{X}_{\bullet\bullet 11} = \begin{pmatrix} a & c \\ b & d \end{pmatrix}, \mathcal{X}_{\bullet\bullet 21} = \begin{pmatrix} e & g \\ f & h \end{pmatrix}, \mathcal{X}_{\bullet\bullet 12} = \begin{pmatrix} i & k \\ j & l \end{pmatrix}, \mathcal{X}_{\bullet\bullet 22} = \begin{pmatrix} m & o \\ n & p \end{pmatrix}$$

$$\text{then, } \text{mat}(\mathcal{X}) = \begin{pmatrix} a & e & i & m \\ b & f & j & n \\ c & g & k & o \\ d & h & l & p \end{pmatrix}.$$

2.1.5 Tensor Multiplication

I The outer product of tensors

Consider two tensors $\mathcal{X} \in \mathbb{R}^{I_1 \times \cdots \times I_P}$ and $\mathcal{Y} \in \mathbb{R}^{J_1 \times \cdots \times J_Q}$. Then, the outer product of $\mathcal{X}$ and $\mathcal{Y}$ is the size of $I_1 \times \cdots \times I_P \times J_1 \times \cdots \times J_Q$ and denoted as

$$\mathcal{W} = \mathcal{X} \circ \mathcal{Y}. \tag{2.1}$$

where each element of $\mathcal{W}$ satisfies

$$w_{I_1 \ldots I_P J_1 \ldots J_Q} = x_{I_1 \ldots I_P} y_{J_1 \ldots J_Q}. \tag{2.2}$$

II The inner product of tensors

Consider two tensors $\mathcal{X}, \mathcal{Y} \in \mathbb{R}^{I_1 \times \cdots \times I_D}$ with the same order and the same size. Then, the inner product of $\mathcal{X}$ and $\mathcal{Y}$ is defined as

$$w = \langle \mathcal{X}, \mathcal{Y} \rangle = \langle \text{vec}(\mathcal{X}), \text{vec}(\mathcal{Y}) \rangle, \tag{2.3}$$

where w is a scalar. More precisely, the element-wise form of the inner product is given by

$$w = \langle \mathcal{X}, \mathcal{Y} \rangle = \sum_{i_1=1}^{I_1} \sum_{i_2=1}^{I_2} \cdots \sum_{i_D=1}^{I_D} x_{i_1 \ldots i_D} y_{i_1 \ldots i_D}. \tag{2.4}$$

III The tensor-matrix multiplication

This multiplication is for multiplying a tensor with a matrix. Suppose $\mathcal{X} \in \mathbb{R}^{I_1 \times \cdots \times I_D}$ is a tensor and $\mathbf{A} \in \mathbb{R}^{J_d \times I_d}$ is a matrix. This can be done first by unfolding the tensor $\mathcal{X}$ in dth mode and obtain $\mathbf{X}_{(d)}$ then, take the matrix product with $\mathbf{A}$ as follows

$$\mathbf{W} = \mathbf{A}\mathbf{X}_{(d)}, \tag{2.5}$$

once the matrix $\mathbf{W}$ is obtained, convert it back to a tensor form, i.e., $\mathcal{W} \in \mathbb{R}^{I_1 \times \cdots \times I_{d-1} \times J_d \times I_{d+1} \times \ldots I_D}$ as follows

$$\mathcal{W} = \mathcal{X} \times_d \mathbf{A} \in \mathbb{R}^{I_1 \times \cdots \times I_{d-1} \times J_d \times I_{d+1} \times \ldots I_D}, \tag{2.6}$$

where $\times_d$ represents the d-mode product, and each element can be expressed as

$$w_{i_1 \ldots i_{d-1} j_d i_{d+1} \ldots i_D} = \sum_{i_d=1}^{I_d} w_{i_1 \ldots i_d \ldots i_D} a_{j_d i_d}. \tag{2.7}$$

IV **The product($\circledast$) of two tensors**[14]

Let $\mathcal{C} \in \mathbb{R}^{I_1 \times I_2 \times \ldots I_D \times J_1 \times J_2 \times \cdots \times J_D}$ and $\mathcal{X} \in \mathbb{R}^{I_1 \times I_2 \times \ldots I_D}$, then, the product of $\mathcal{C} \circledast \mathcal{X}$ can be defined as follows

$$(\mathcal{C} \circledast \mathcal{X})_{i_1 \ldots i_D} = \sum_{j_1=1}^{I_1} \sum_{j_2=1}^{I_2} \cdots \sum_{j_D=1}^{I_D} c_{i_1 \ldots i_D j_1 \ldots j_D} x_{j_1 \ldots j_D}. \tag{2.8}$$

Lemma 2.1. *If* $\mathcal{C} \in \mathbb{R}^{I_1 \times I_2 \times \ldots I_D \times J_1 \times J_2 \times \cdots \times J_D}$ *and* $\mathcal{X} \in \mathbb{R}^{I_1 \times I_2 \times \ldots I_D}$

then, $\text{vec}(\mathcal{C} \circledast \mathcal{X}) = \text{mat}(\mathcal{C})\,\text{vec}(\mathcal{X})$.

Moreover, if $\mathcal{C}$ *can be factorized with matrices* $\mathcal{C}^{(d)}$, *then* $\text{vec}(\mathcal{C} \circledast \mathcal{X}) = [\mathbf{C}^{(D)} \otimes \cdots \otimes \mathbf{C}^{(1)}]\,\text{vec}(\mathcal{X})$.

2.1.6 Matrix Products

I **The Kronecker product**

The *Kronecker product* [10] of two matrices $\mathbf{X} \in \mathbb{R}^{K \times L}$ and $\mathbf{Y} \in \mathbb{R}^{M \times N}$ is denoted by $\mathbf{X} \otimes \mathbf{Y}$, and it is a KL-by-MN matrix given as

$$\mathbf{X} \otimes \mathbf{Y} = \begin{pmatrix} x_{11}\mathbf{Y} & x_{12}\mathbf{Y} & \ldots & x_{1l}\mathbf{Y} \\ x_{21}\mathbf{Y} & x_{22}\mathbf{Y} & \ldots & x_{2l}\mathbf{Y} \\ \ldots & \ldots & \ldots & \ldots \\ x_{k1}\mathbf{Y} & x_{k2}\mathbf{Y} & \ldots & x_{kl}\mathbf{Y} \end{pmatrix}. \tag{2.9}$$

II **The Khatri-rao product**

The *Khatri-rao product* [10] is the column wise Kronecker product of two matrices $\mathbf{X} = [\mathbf{x}_1, \ldots, \mathbf{x}_L] \in \mathbb{R}^{K \times L}$ and $\mathbf{Y} = [\mathbf{y}_1, \ldots, \mathbf{y}_L] \in \mathbb{R}^{M \times L}$ is denoted by $\mathbf{X} \odot \mathbf{Y}$ and gives a KM-by-L matrix as follows

$$\mathbf{X} \odot \mathbf{Y} = [\mathbf{x}_1 \otimes \mathbf{y}_1, \mathbf{x}_1 \otimes \mathbf{y}_1, \ldots, \mathbf{x}_L \otimes \mathbf{y}_L] \in \mathbb{R}^{KM \times L} \tag{2.10}$$

where $\mathbf{x}_i$ and $\mathbf{y}_i$, $i = 1, \ldots, L$ are the columns of $\mathbf{X}$ and $\mathbf{Y}$, respectively.

III **The Hadamard product**

The *Hadamard product* [10] is also called the elementwise matrix product. Suppose, $\mathbf{X} \in \mathbb{R}^{K \times L}$ and $\mathbf{Y} \in \mathbb{R}^{K \times L}$ are two matrices with same dimensions, then the Hadamard product of them is a matrix of size $K \times L$ which is denoted by $\mathbf{X} * \mathbf{Y}$ and defined as

$$\mathbf{X} * \mathbf{Y} = \begin{pmatrix} x_{11}y_{11} & x_{12}y_{12} & \dots & x_{1l}y_{1l} \\ x_{21}y_{21} & x_{22}y_{22} & \dots & x_{2l}y_{2l} \\ \dots & \dots & \dots & \dots \\ x_{k1}y_{k1} & x_{k2}y_{k2} & \dots & x_{kl}y_{kl} \end{pmatrix}. \tag{2.11}$$

2.1.7 Important Properties

I $(\mathbf{W} \otimes \mathbf{X})(\mathbf{Y} \otimes \mathbf{Z}) = \mathbf{WY} \otimes \mathbf{XZ}$

II $(\mathbf{X} \odot \mathbf{Y})^T(\mathbf{X} \odot \mathbf{Y}) = \mathbf{X}^T\mathbf{X} * \mathbf{Y}^T\mathbf{Y}$

III $\operatorname{tr}(\mathbf{XYZ}) = \operatorname{tr}(\mathbf{ZXY}) = \operatorname{tr}(\mathbf{YZX})$

IV $\operatorname{vec}(\mathbf{X})^T \operatorname{vec}(\mathbf{Y}) = \operatorname{tr}(\mathbf{X}^T\mathbf{Y})$

V $\operatorname{vec}(\mathbf{XYZ}) = (\mathbf{Z}^T \otimes \mathbf{X}) \operatorname{vec}(\mathbf{Y})$,

where the operators "$\otimes$", "$\odot$", and "$*$" Kronecker product, Khatri-rao product, and Hadamrd product respectively.

2.1.8 Tensor Rank

The D-order tensor $\mathcal{X}$ is called *rank-one* tensor if it can be written as the outer product of D vectors as follows

$$\mathcal{X} = \mathbf{a}^{(1)} \circ \cdots \circ \mathbf{a}^{(D)}, \tag{2.12}$$

where $\mathbf{a}^{(d)}$, $d = 1, \dots, D$, are vectors from D modes.

The *rank* of tensor $\mathcal{X}$ is denoted as $rank(\mathcal{X})$ [10], and is defined to be the minimum number

of rank-one tensors that exactly sum up to tensor $\mathcal{X}$. In other words, we can write $rank(\mathcal{X})$ as follows

$$rank(\mathcal{X}) := \arg\min\{R \in \mathbb{N} : \mathcal{X} = \sum_{r=1}^{R} \mathbf{a}_r^{(1)} \circ \cdots \circ \mathbf{a}_r^{(D)}\}. \tag{2.13}$$

2.2 TENSOR DECOMPOSITIONS

In this section, we are going to discuss tensor decompositions which is also called tensor factorizations. The concept of decomposition was introduced by Hitchcock (1927, 1928). These techniques are very popular in many fields such as neuroscience, image and video analysis, numerical anaysis, chemometrics, data mining and signal processing, etc. Here, we focus on two decomposition techniques, i.e., Canonical Polyadic decomposition, and Tucker decomposition.

2.2.1 Canonical Polyadic(CP) Decomposition

The *CP decomposition* was first introduced in 1927 by Hitchcock [5] as the polyadic form of a tensor. That means, the CP procedure decomposes a tensor into a sum of rank-one tensors. For example, consider a third-order tensor $\mathcal{X} \in \mathbb{R}^{I_1 \times I_2 \times I_3}$, then we can write the CP decomposition of tensor $\mathcal{X}$ as a sum of R components of rank-one tensors as

$$\mathcal{X} \approx \sum_{r=1}^{R} \lambda_r \mathbf{a}_r^{(1)} \circ \mathbf{a}_r^{(2)} \circ \mathbf{a}_r^{(3)}, \tag{2.14}$$

where $\lambda_r, r = 1, \ldots R$, is the scalar weighting of the rank-one tensors, R is the rank of the tensor, the operator "$\circ$" is the outer product, and $\mathbf{a}_r^{(1)} \in \mathbb{R}^{I_1}$, $\mathbf{a}_r^{(2)} \in \mathbb{R}^{I_2}$, and $\mathbf{a}_r^{(3)} \in \mathbb{R}^{I_3}$ are normalized unit vectors.

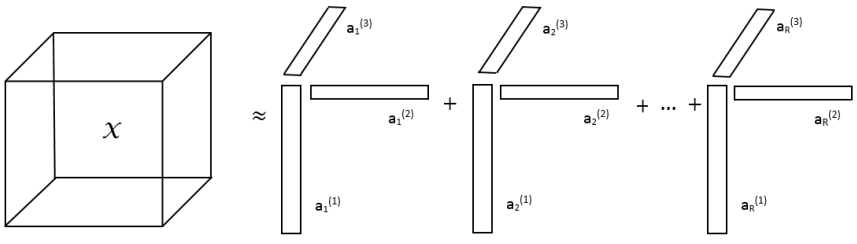

Figure 2.2: CP decomposition of a third-order tensor

Now, CP decomposition for D-order tensor $\mathcal{X} \in \mathbb{R}^{I_1 \times \cdots \times I_D}$ is given as follows

$$\mathcal{X} \approx \sum_{r=1}^{R} \lambda_r \mathbf{a}_r^{(1)} \circ \mathbf{a}_r^{(2)} \circ \cdots \circ \mathbf{a}_r^{(D)}, \tag{2.15}$$

where $\lambda_r, r = 1, \ldots R$, is the scalar weighting of the rank-one tensors.

2.2.2 Tucker Decomposition

The *Tucker Decomposition* is a technique that can be used to decompose a tensor $\mathcal{X}$ into a core tensor $\mathcal{G}$ multiplied by a factor matrix $\mathbf{A}^{(d)}$ along each mode. The factor matrices $\mathbf{A}^{(d)}$ are usually orthogonal. This decomposition was introduced by Tucker[16] in 1963. The Tucker decomposition of a third-order tensor $\mathcal{X} \in \mathbb{R}^{I_1 \times I_2 \times I_3}$ can be written as

$$\mathcal{X} \approx \mathcal{G} \times_1 \mathbf{A}^{(1)} \times_2 \mathbf{A}^{(2)} \times_3 \mathbf{A}^{(3)}, \tag{2.16}$$

where $\mathbf{A}^{(d)} \in \mathbb{R}^{I_d \times R_d}$ for $d = 1, 2, 3$ are (orthogonal) factor matrices and $\mathcal{G} \in \mathbb{R}^{R_1 \times R_2 \times R_3}$ is the core tensor. Here, R_d is the d-rank of tensor $\mathcal{X}$ in mode d.

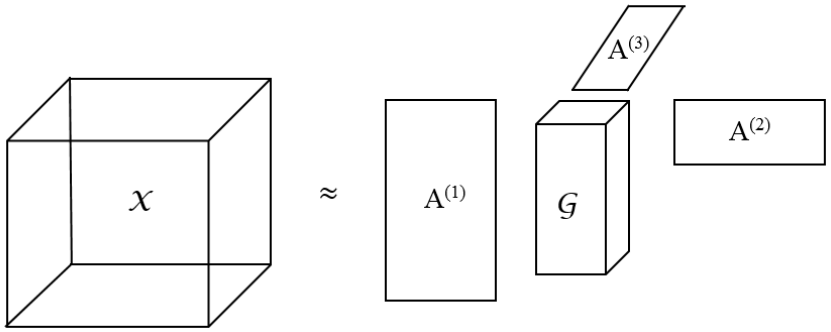

Figure 2.3: Tucker decomposition of a third-order Tensor

Tucker decomposition of the given third-order tensor can be represented in the following elementwise form

$$x_{i_1 i_2 i_3} = \sum_{j_1=1}^{R_1} \sum_{j_2=1}^{R_2} \sum_{j_3=1}^{R_3} g_{i_1 i_2 i_3} \mathbf{a}_{j_1} \circ \mathbf{a}_{j_2} \circ \mathbf{a}_{j_3}, \tag{2.17}$$

Now, consider a D-order array $\mathcal{X} \in \mathbb{R}^{I_1 \times \cdots \times I_D}$. Then, the Tucker decomposition can be written as follows

$$\mathcal{X} = \mathcal{G} \times_1 \mathbf{A}^{(1)} \times_2 \mathbf{A}^{(2)} \times_3 \cdots \times_D \mathbf{A}^{(D)}, \tag{2.18}$$

with the equivalent elementwise form as

$$x_{i_1 i_2 \ldots i_D} = \sum_{j_1=1}^{R_1} \sum_{j_2=1}^{R_2} \cdots \sum_{j_d=1}^{R_D} g_{i_1 i_2 \ldots i_D} a^{(1)}_{i_1 j_1} a^{(2)}_{i_2 j_2} \ldots a^{(D)}_{i_D j_D}, \tag{2.19}$$

where $i_d = 1, \ldots, I_d, d = 1, \ldots, D$.

The following lemma is very useful for the tensor decomposition, and will be used in the next chapter.

Lemma 2.2. *Suppose a tensor $\mathcal{B} \in \mathbb{R}^{I_1 \times \cdots \times I_D}$ can be decomposed into a sum of R rank-one tensors. Then,*

$$\begin{aligned} \mathcal{B}_{(d)} &= \mathcal{B}_d (\mathcal{B}_D \odot \cdots \odot \mathcal{B}_{d+1} \odot \mathcal{B}_{d-1} \odot \cdots \odot \mathcal{B}_1)^T \\ \textit{and} \quad \text{vec}(\mathcal{B}) &= (\mathcal{B}_D \odot \cdots \odot \mathcal{B}_1) \mathbf{1}_R, \end{aligned}$$

where $\mathcal{B}_d = [\boldsymbol{\beta}_d^{(1)}, \ldots, \boldsymbol{\beta}_d^{(R)}]$ for $d = 1, \ldots, D$ and $\mathbf{1}_R$ is the vector of ones of size R and "$\odot$" is the Khatri-Rao product defined in Section 2.1.

CHAPTER 3

TENSOR REGRESSION

In this chapter of the thesis, we mainly discuss a tensor regression model that has a univariate response and array structured predictors. We start with Generalized linear model (GLM) where the covariates are collected into a vector, then a GLM model with matrix covariates and we extend the regression model to a rank-R generalized linear tensor regression model proposed by Zhou et al. (2013)[17]. The tensor regression model is useful because it can be applied when we have imaging data and it helps to reduce the dimension of the regression parameters. For example, we can use an image array as a tensor predictor and observe the image recovery through the tensor regression model. We conduct simulation studies and a numerical analysis to investigate the recovery of a low rank image signal through the tensor regression model. Throughout this chapter, we use a higher order array $\mathcal{B}$ to represent the regression coefficient parameters in tensor-based regression model, $\mathbf{B}$ to represent the regression coeffcients in matrix-based regression model, and vector $\boldsymbol{\beta}$ to represent the regression coefficient parameters in the vector-based regression model. Moreover, we consider Y as the univariate response, $\mathbf{z} \in \mathbb{R}^{I_0}$ as the vector covariates, and $\mathcal{X} \in \mathbb{R}^{I_1 \times \cdots \times I_D}$ is the tensor predictor which represents a binary signal.

3.1 BASIC GENERALIZED REGRESSION MODEL

First, we begin with predictor vectors $\mathbf{x}$ and $\mathbf{z} \in \mathbb{R}^{I_0}$ that represent vectors of covariates, and Y denotes a univariate response variable. Assume that, $\mathbf{z} \in \mathbb{R}^{I_0}$ absorbed into $\mathbf{x}$. In the classical Generalized Linear Model(GLM) proposed by McCullagh and Nelder (1983)[12], Y belongs to an exponential family with probability function in the following form

$$p(y \mid \theta, \phi) = exp\{\frac{y\theta - b(\theta)}{a(\phi)} + c(y, \phi)\}, \tag{3.1}$$

where $\phi > 0$ and θ are dispersion parameter and natural parameter respectively. Moreover, y represents the sample instance of Y. The classical GLM associates a vector $\mathbf{x} \in \mathbb{R}^I$ to the mean function $\mu = E(y \mid \mathbf{x})$ through the link function $g(\mu)$ as follows.

$$g(\mu) = \eta = \alpha + \boldsymbol{\beta}^T \mathbf{x}, \tag{3.2}$$

where $g(\cdot)$ is a strictly increasing function, η represents the linear systematic component of the GLM, α is the intercept, and $\boldsymbol{\beta} \in \mathbb{R}^I$ is the vector of coefficient parameters.

Now, assume the covariates are collected into a matrix $\mathbf{X} \in \mathbb{R}^{I_1 \times I_2}$. Then, the linear term $\boldsymbol{\beta}^T \mathbf{X}$ in model (3.2) can be extended as $\boldsymbol{\beta}_1^T \mathbf{X} \boldsymbol{\beta}_2$ and systematic part of the GLM model can be written as

$$g(\mu) = \eta = \alpha + \boldsymbol{\beta}_1^T \mathbf{X} \boldsymbol{\beta}_2, \tag{3.3}$$

where $\boldsymbol{\beta}_1 \in \mathbb{R}^{I_1}$ and $\boldsymbol{\beta}_2 \in \mathbb{R}^{I_2}$. Furthermore, the bilinear form $\boldsymbol{\beta}_1^T \mathbf{X} \boldsymbol{\beta}_2$ can be represented as $\boldsymbol{\beta}_1^T \mathbf{X} \boldsymbol{\beta}_2 = (\boldsymbol{\beta}_2 \otimes \boldsymbol{\beta}_1)^T vec(\mathbf{X})$. Suppose, $\mathbf{z}$ is vector-valued covariate and the main covariates are contained in tensor $\mathcal{X} \in \mathbb{R}^{I_1 \times \cdots \times I_D}$. Then, the systematic part of the GLM model defined in (3.2) and (3.3) can be written as

$$g(\mu) = \eta = \alpha + \boldsymbol{\gamma}^T \mathbf{z} + (\boldsymbol{\beta}_D \otimes \cdots \otimes \boldsymbol{\beta}_1)^T vec(\mathcal{X}), \tag{3.4}$$

where $\boldsymbol{\gamma} \in \mathbb{R}^{I_0}$, $\boldsymbol{\beta}_d \in \mathbb{R}^{I_d}$ for $d = 1, 2, \ldots D$, and the operation "$\otimes$" denotes the Kronecker product defined in Section 2.1. Model (3.4) is the basic model for regression with tensor covariates. The main advantage of the tensor-based regression model (3.4) is that the dimensionality of the features is reduced from the order $\prod_d I_d$ in the vector-based regression to $\sum_d I_d$ in the tensor-based regression model (3.4). For example, consider MRI image of size $128 \times 128 \times 128 = 2,097,152$. If we use classical methods that use predictors in vector format then, we would have a model with more than 2 million parameters to be estimated which would make it difficult to handle and computationally would be very expensive. However, if we use the tensor-based model then we have a $128+128+128 = 384$-dimensional problem. Therefore, this model reduce high dimensional data problem to a

low dimensional framework. Figure 3.1 in Section 3.5 is provided an illustrative example for the tensor-based regression model that can capture regions of interest of image shapes, such as "Square", "T-shape", "Cross", "Disk", "Triangle" and "Star".

3.2 TENSOR REGRESSION MODEL

Now, we can generalize the basic model introduces in Section 3.1. To this end, suppose, both predictor array $\mathcal{X} \in \mathbb{R}^{I_1 \times I_2 \times \ldots \times I_D}$ and regression coefficient array $\mathcal{B} \in \mathbb{R}^{I_1 \times I_2 \times \ldots \times I_D}$ have the same size where the regression coefficient tensor $\mathcal{B}$ captures the effects of all array elements of the covariate array. Then, the linear systematic part of the basic GLM model (3.4) can be written as

$$g(\mu) = \alpha + \boldsymbol{\gamma}^T \mathbf{z} + \langle \mathcal{B}, \mathcal{X} \rangle,$$

where $\langle \cdot \rangle$ is the inner product of $\mathcal{B}$ and $\mathcal{X}$. The main drawback of this model is that the regression coefficient tensor $\mathcal{B}$ has the same number of parameters(p) as the covariate array $\mathcal{X}$, which is high dimensional and exceeds the usual sample size(n). i.e., we have the problem of $p > n$. Therefore, it would be very useful, and ideal to approximate $\mathcal{B}$ with smaller number of parameters. To this end, assume that $\mathcal{B}$ can be decomposed into a rank-one tensor. Consider the rank-1 decomposition of $\mathcal{B}$. Then, from (2.12) $\mathcal{B}$ can be written as

$$\mathcal{B} = \boldsymbol{\beta}_1 \circ \cdots \circ \boldsymbol{\beta}_D,$$

where $\boldsymbol{\beta}_d \in \mathbb{R}^{I_d}$. Then, by lemma 2.2, we have the following result

$$\begin{aligned} \text{vec}(\mathcal{B}) &= \text{vec}(\boldsymbol{\beta}_1 \circ \cdots \circ \boldsymbol{\beta}_D) \\ &= \boldsymbol{\beta}_D \odot \cdots \odot \boldsymbol{\beta}_1 \\ &= \boldsymbol{\beta}_D \otimes \cdots \otimes \boldsymbol{\beta}_1. \end{aligned}$$

Therefore, model (3.4) is a rank-one approximation model, the approximation to the regression coefficient tensor $\mathcal{B}$. Now, we can extend this idea and propose a more flexible model.

To this end, suppose tensor $\mathcal{B}$ can be decomposed into rank - R as

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

Then, the systematic part of the family of rank-R generalized tensor regression model can be expressed as

$$\begin{aligned} \mathbf{g}(\mu) &= \alpha + \boldsymbol{\gamma}^T \mathbf{z} + \langle \sum_{r=1}^{R} \boldsymbol{\beta}_1^{(r)} \circ \cdots \circ \boldsymbol{\beta}_D^{(r)} \ , \ \mathcal{X} \rangle \\ &= \alpha + \boldsymbol{\gamma}^T \mathbf{z} + \langle (\mathcal{B}_D \odot \cdots \odot \mathcal{B}_1) \mathbf{1}_R \ , \ \mathcal{X} \rangle \end{aligned} \tag{3.5}$$

where $\mathcal{B}_d = [\boldsymbol{\beta}_d^{(1)}, \ldots, \boldsymbol{\beta}_d^{(R)}]$ for $d = 1, \ldots, D$ and $\mathbf{1}_R$ is the vector of ones of size R and $(\mathcal{B}_D \odot \cdots \odot \mathcal{B}_1) \in \mathbb{R}^{\prod_d I_d \times R}$ is the Khatri-Rao product in section 2.1. (See Zhou et al. (2013)[17])

The number of parameters in model (3.5) is $I_0 + R \sum_d I_d$ which is way smaller than $I_0 + R \prod_d I_d$. However, it provides a good recovery to many low rank image signals. We use model (3.5) to recover some images, like "Square", "T-shape", "Cross", "Disk", "Triangle" and "Star". Some results are shown in Figure 3.1 in Section 3.5.

3.3 ESTIMATION

This section of the thesis describes an estimation procedure to estimate the parameters of the tensor regression model introduced in Section 3.2, by using the maximum likelihood method. Suppose we have n independent and identitically distributed (i.i.d) data i.e., n data$\{(y_i, \mathbf{x}_i, \mathbf{z}_i), i = 1, \ldots, n\}$. Then, the log-likelihood function of (3.1) can be written as,

$$L(\alpha, \boldsymbol{\gamma}, \mathcal{B}_1, \ldots, \mathcal{B}_D) = \sum_{i=1}^{n} \left[\frac{y_i \theta_i - b(\theta_i)}{a(\phi)} + c(y_i, \phi) \right], \tag{3.6}$$

where θ_i is function of related regression parameters $(\alpha, \gamma, \mathcal{B}_1, \ldots, \mathcal{B}_D)$ given in (3.5). Zhou et.al (2013)[17] proposed the following block relaxation algorithm to obtain maximum likelihood estimators of the parameters of the model (3.5).

Algorithm 1 Algorithm Block relaxation algorithm for maximizing (3.6)

Initialize: $(\alpha^{(0)}, \boldsymbol{\gamma}^{(0)}) = \arg\max_{\alpha,\boldsymbol{\gamma}} L(\alpha, \boldsymbol{\gamma}, \mathbf{0}, \ldots, \mathbf{0})$, $\mathcal{B}_d^{(0)} \in I_d \times R$ a random matrix for $d = 1, \ldots, D$

repeat

for $d = 1, \ldots, D$ **do**

$\mathcal{B}_d^{(t+1)} = \arg\max_{\mathcal{B}_d} L(\alpha^{(t)}, \boldsymbol{\gamma}^{(t)}, \mathcal{B}_1^{(t+1)}, \ldots, \mathcal{B}_{d-1}^{(t+1)}, \mathcal{B}_d, \mathcal{B}_{d+1}^{(t)}, \ldots, \mathcal{B}_D^{(t)})$

end for

$(\alpha^{(t+1)}, \boldsymbol{\gamma}^{(t+1)}) = \arg\max_{\alpha,\boldsymbol{\gamma}} L(\alpha, \boldsymbol{\gamma}, \mathcal{B}_1^{(t+1)}, \ldots, \mathcal{B}_D^{(t+1)})$

until $L(\theta^{(t+1)}) - L(\theta^{(t)}) < \epsilon$

This algorithm starts with initial values of intercept α and vector $\boldsymbol{\gamma}$ and each iteration updating a block $\mathcal{B}_d$ while other parameters fixed. When updating $\mathcal{B}_d \in \mathbb{R}^{I_d \times R}$ this algorithm works as a classical GLM problem as follows

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

where R is the rank and the operator "$\odot$" is the Khatri-rao product. As a result, the problem will convert in to a GLM regression with R_{I_d} regression parameters and the estimation process will be similar to a sequence of low dimensional optimizations. When estimating regression parameters, this algorithm takes a known rank. However, if we want to find a specific rank for a tensor regression model, we can consider this problem as a model selection criteria such as Akaike information criterion (AIC) or Bayesian information criterion (BIC). AIC is $-2ln\ L(\theta)+2p_k$, and BIC criterion is $-2ln\ L(\theta)+p_k$, where p_k is the effective number of parameters for the tensor regression model.

3.4 REGULARIZED ESTIMATION

The regularized estimation procedure is useful in a situation where the sample size is less than the number of parameters of the model. This technique is essential even for a low rank tensor model. The regularization method is important not only to handle "large p, small n" problem, but it is also a great way to stabilize the estimates and to improve the risk properties of the estimators when $p < n$. Zhou et al. (2013)[17] used the Sparsity regularization technique to overcome "large p, small n" problem. The regularization technique can be used to identify sub-regions that are associated with the response variable. Now, the goal is to maximize the likelihood function by using regularization technique as follows

$$L(\alpha, \gamma, \mathcal{B}_1, \ldots, \mathcal{B}_D) - \sum_{d=1}^{D} \sum_{r=1}^{R} \sum_{i=1}^{I_d} P_\lambda(|\boldsymbol{\beta}_{di}^{(r)}|, \rho), \tag{3.7}$$

where $P_\lambda(|\boldsymbol{\beta}|, \rho)$ denotes a scalar penalty function, λ is an index for the penalty family, and ρ is the penalty tuning parameter. Also, there are many penalties including lasso by Tibshirani (1996)[15] where the index is $\lambda = 1$ to achieve sparsity solution, ridge regression with $\lambda = 2$, the power family by Frank and Friedman (1993)[4], in which scalar penalty function is $P_\lambda(|\beta|, \rho) = \rho|\beta|^\lambda$, where $\lambda \in (0, 2]$, elastic net by Zou and Hastie (2005)[20] for correlated variables, in which scalar penalty function is $P_\lambda(|\beta|, \rho) = \rho[(\lambda - 1)\beta^2/2 + (2 - \lambda)|\beta|]$ and index $\lambda \in [1, 2]$, and so on. The regularized estimation for tensor models incurs small changes in algorithm 1 we discussed in section 3.3. That is, when updating $\mathcal{B}_d$, now simply fit a penalized GLM regression problem in the following way,

$$\mathcal{B}_d^{(t+1)} = \arg\max_{\mathcal{B}_d} L(\alpha^{(t)}, \boldsymbol{\gamma}^{(t)}, \mathcal{B}_1^{(t+1)}, \ldots, \mathcal{B}_{d-1}^{(t+1)}, \mathcal{B}_d, \mathcal{B}_{d+1}^{(t)}, \ldots, \mathcal{B}_D^{(t)}) - \sum_{r=1}^{R} \sum_{i=1}^{I_d} P_\lambda(|\beta_{di}^{(r)}|, \rho). \tag{3.8}$$

The regularization techniques that is discussed above can be applied to estimate the regression coefficient array in tensor regression model. We can compare the results obtained by regularization technique with results obtained when there is no regularization. We used some regularization techniques in the numerical analysis section to compare the recovery

of some given image signals.

3.5 SIMULATION STUDY

In the simulation study, we consider several two-dimensional images where $\mathbf{B} \in \mathbb{R}^{64\times 64}$ and recover images by estimating the parameters of tensor regression model described in Section 3.2. Here, we use the standard normal distribution to generate responses y_i. We simulated 1000 univariate responses according to a normal distribution model with mean $\mu = \gamma \mathbf{z}_i + \langle \mathbf{B}, \mathbf{x_i} \rangle$, where covariate vector $\boldsymbol{\gamma} = 1_5$, the inner product between two arrays is $\langle \mathbf{B}, \mathbf{x}_i \rangle = \langle \text{vec}(\mathbf{B}), \text{vec}(\mathbf{x}_i) \rangle = \sum_{i_1 \ldots i_D} \beta_{i_1 \ldots i_D} x_{i_1 \ldots i_D}$ defined in section 2.1. We consider the regression coefficient array $\mathbf{B}$ as a binary array, that means the true image signal region of the array equal to 1 for black and 0 for white. We generate a regular covariate $\mathbf{z}_i$ and image covariate $\mathbf{x}_i$ from standard normal distribution. The purpose of this simulation study is to check whether the tensor regression model (3.5) can identify the true image with the regression coefficient array $\mathbf{B}$ using the simulated data.

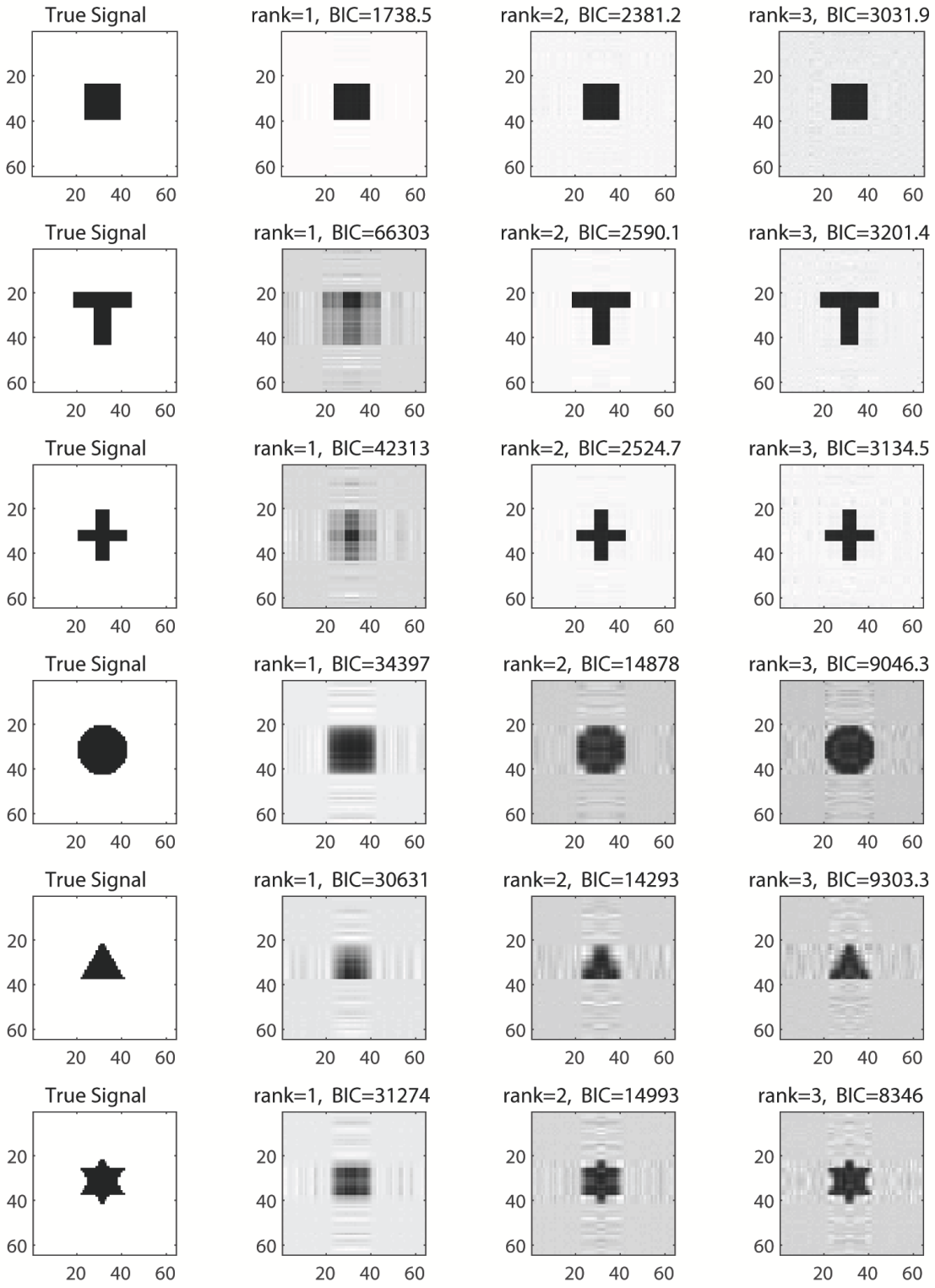

Figure 3.1: True and recovered signals by tensor regression model

Each image signal in Figure 3.1 represents a matrix variate of a size 64×64 with randomly generated standard normal values. The regression coefficient for each entry in the matrix is 1 for black and 0 for white. The simulated sample size n = 1000. We consider several two dimensional images where the regression coefficient array $\mathbf{B} \in \mathbb{R}^{64\times64}$ which is in the first column of Figure 3.1, images estimated by model (3.4) in the second column of Figure 3.1 and recovered images by model (3.5) are in the third and fourth column in Figure 3.1. By observing the image recovery, it is clear that the first image signal "Square" can be perfectly recovered by a rank-1 tensor regression model, while other rank models shows a fair recovery and some signs of overfitting. The image signals"T-shape" and "Cross" is perfectly recovered by rank-2 tensor regression model. The rank-3 tensor regression model gives a good recovery for "Disk", "Triangle" and "Star" image signals. According to the results of Figure 3.1, we can compare basic model, with the rank-R tensor regression model (3.5) and conclude is able to identify more image signals than the basic model (3.4).

3.6 NUMERICAL ANALYSIS

This numerical study is performed to investigate the results of proposed method by Zhou et al. (2013)[17]. In this section we consider a few two-dimensional image signals "T-shape", "Triangle" and "Disk". Our goal is to investigate the performance of the tensor regression model by changing the sample size and signal strengths. First, we change the sample size to five different values and observe the recovery of image signals. The sample sizes we considered in this analysis are n = 200, 300, 400, 500 and 750. Second, we vary the signal strength to five different values, $\sigma = 50\%, 20\%, 10\%, 5\%$ and 1% and investigate the recovery of images by using tensor regression model. For both cases we compare the estimates without regularization and with lasso regularization. In this analysis, our response variable is normally distributed with mean, $\eta = \boldsymbol{\gamma}^T\mathbf{z} + \langle \mathbf{B}, \mathcal{X} \rangle$, and standard deviation σ. The corresponding matrix of the image signal is $\mathbf{X}$ is a 64×64 with entries generated from standard normal distribution, $\mathbf{z}$ is a covariate vector with five dimensional

standard normal entries, $\boldsymbol{\gamma}$ is the vector of $(1,1,1,1,1)^T$, and regression coefficient array $\mathbf{B}$ is binary with true signal region equal to 1 for black and 0 for white. This study was performed using rank-3 tensor regression model. The number of parameters in the rank-3 model for this example is 380 can be calculated as $5+3\times(64+64)-3^2$. The regularization technique is specially useful when we have a small samples like 200 or 300 which is less than the number of parameters. We can pick an image signal arbitrarily if there multiple solutions available.

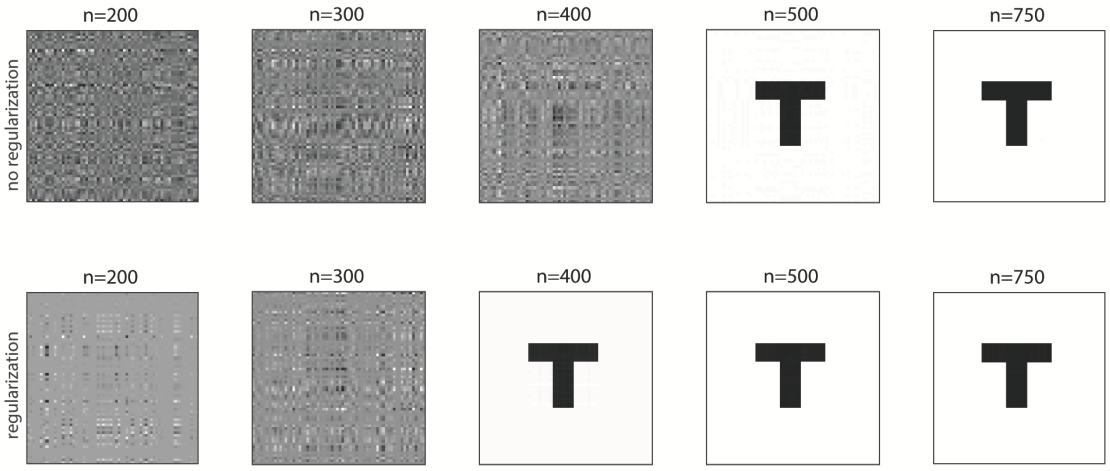

Figure 3.2: Image recovery of "T-shape" signal by varying the sample size

The matrix variate in Figure 3.2 has dimension 64×64 with simulated independent standard normal entries. The coefficient for each entry is 0 for white and 1 for black. The sample size n is varying and observe the recovery of "T-shape" signal with 10% noise level.

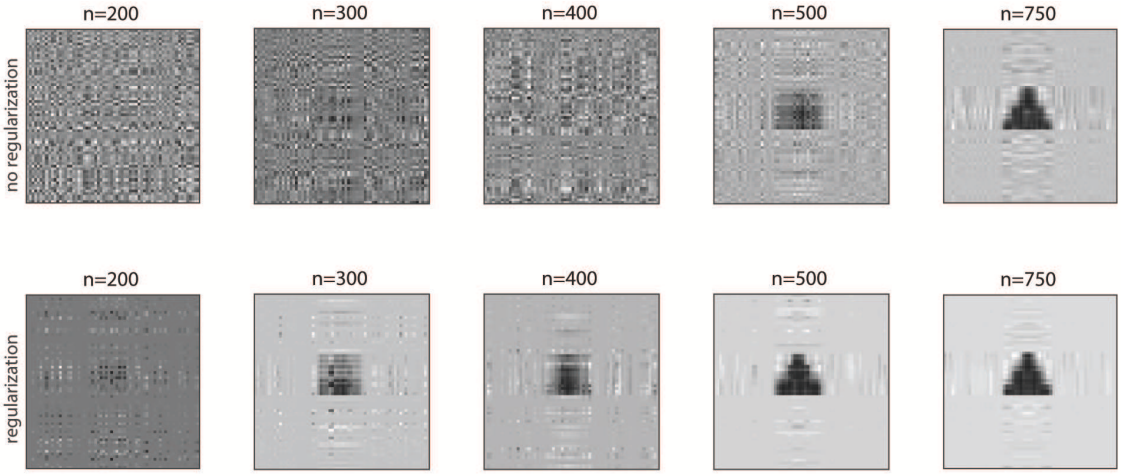

Figure 3.3: Image recovery of "Triangle" signal by varying the sample size

The matrix variate in Figure 3.3 has dimension 64×64 with simulated independent standard normal entries. The coefficient for each entry is 0 for white and 1 for black. The sample size n is varying and observe the recovery of "Triangle" signal with 10% noise level.

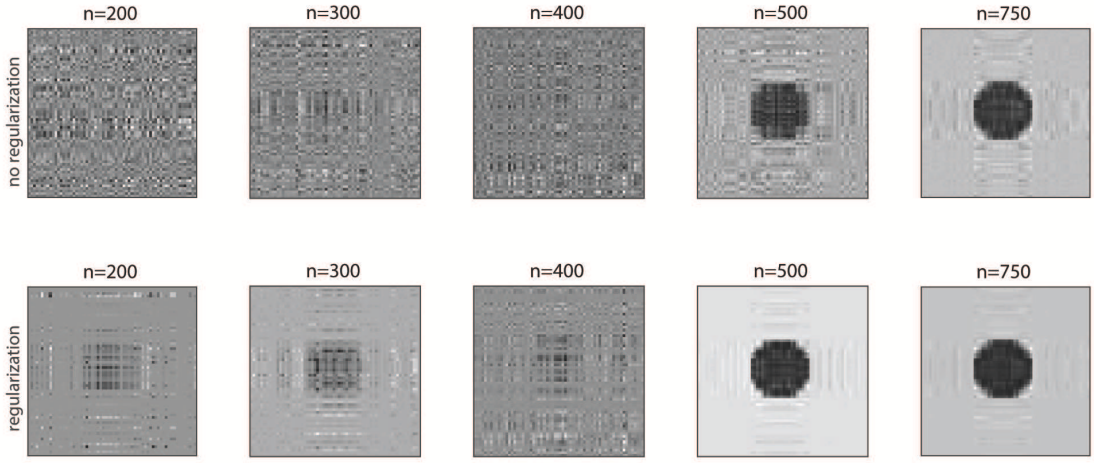

Figure 3.4: Image recovery of "Disk" signal by varying the sample size

The matrix variate in Figure 3.4 has dimension 64×64 with simulated independent standard normal entries. The coefficient for each entry is 0 for white and 1 for black. The sample size n is varying and observe the recovery of "Disk" signal with 10% noise level.

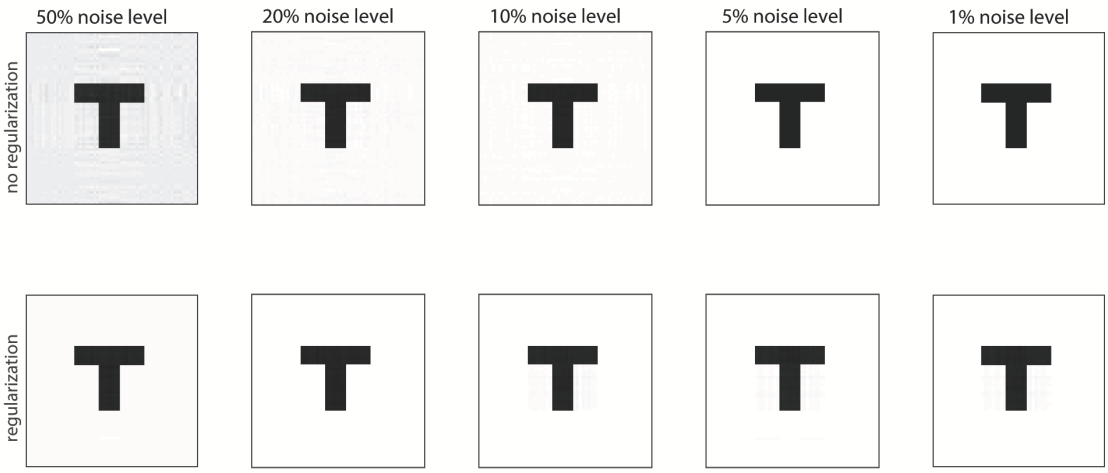

Figure 3.5: Tensor estimation of "T-shape" signal by varying the noise level

The matrix variate in Figure 3.5 has dimension 64×64 with simulated independent standard normal entries and the sample size n=500. The coefficient for each entry is 0 for white and 1 for black. The noise level σ is varying and observe the recovery of "T-shape" signal.

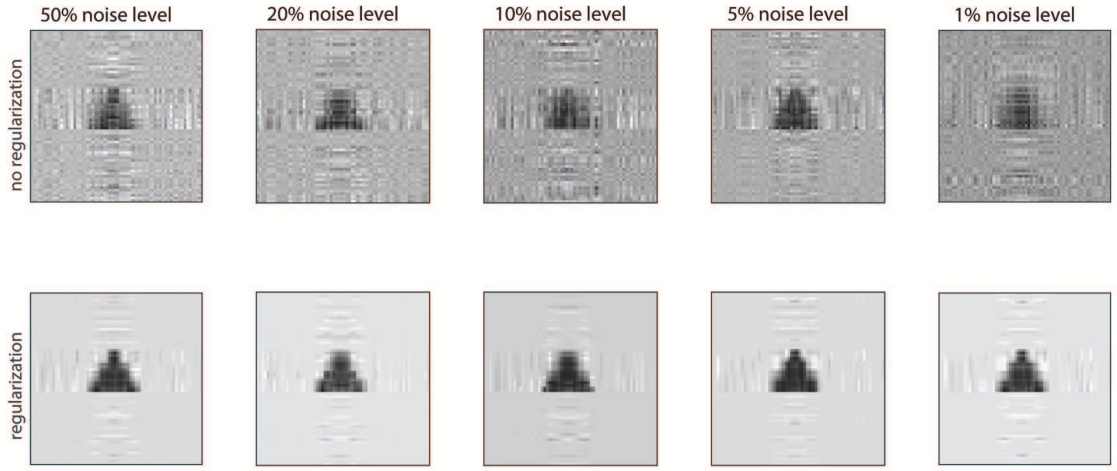

Figure 3.6: Tensor estimation of "Triangle" signal by varying the noise level

The matrix variate in Figure 3.6 has dimension 64×64 with simulated independent standard normal entries and the sample size n=500. The coefficient for each entry is 0 for white and 1 for black. The noise level σ is varying and observe the recovery of "Triangle" signal.

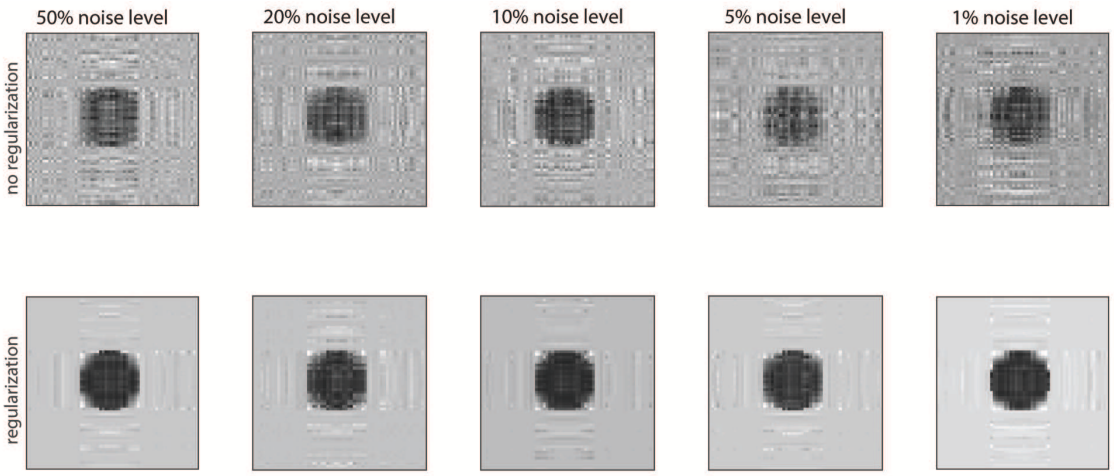

Figure 3.7: Tensor estimation of "Disk" signal by varying the noise level

The matrix variate in Figure 3.7 has dimension 64×64 with simulated independent standard normal entries and the sample size n=500. The coefficient for each entry is 0 for white and 1 for black. The noise level σ is varying and observe the recovery of "Disk" signal.

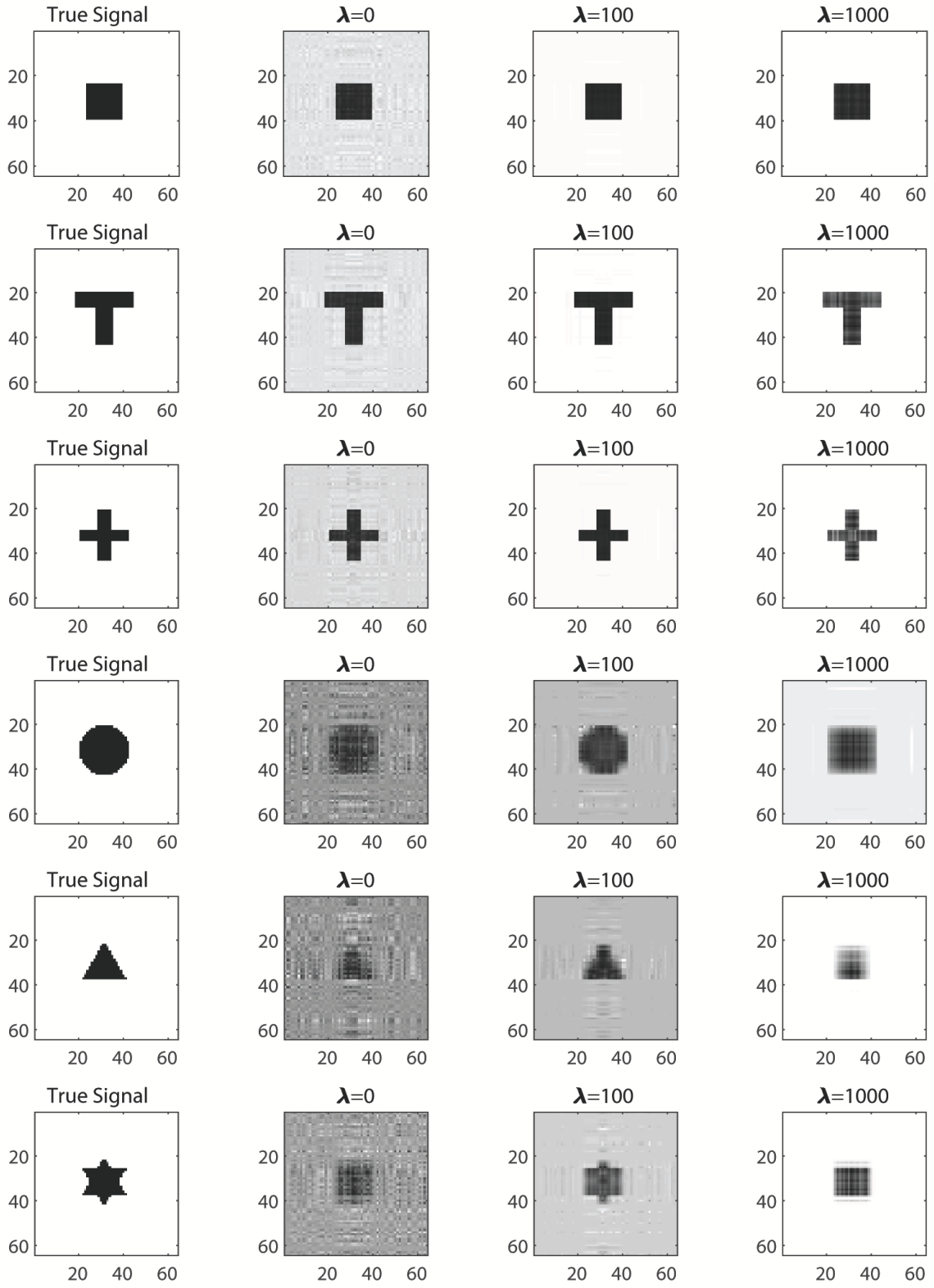

Figure 3.8: Demonstration of lasso regularization

Figure 3.8 shows the results of the outcome when applying the lasso penalty to $\mathbf{B}_d$ in the rank-3 tensor regression model when the sample size $n = 500$. Here, we displayed recovered signals at three different values of $\lambda = 0, 100, 1000$. Without the regularization technique, i.e., $\lambda = 0$, the rank-3 tensor regression was difficulty recovering some image signals such as "Triangle" "Disk", and "Star", mainly due to a very small sample size. However, all most all images recover well at $\lambda = 100$, where as $\lambda = 1000$ some times the quality of the recovered image signal decreases. Therefore, in practical scenarios we need to choose the tuning parameter by using a model selection criterion such as AIC, BIC or cross validation.

CHAPTER 4

TENSOR TIME SERIES

4.1 MOTIVATION

In many situations, tensors or multidimensional arrays data of interest behave as a sequence of time ordered observations. These kinds of data sequences are called tensor time series. Latent or hidden variables are very important in many studies to capture unobserved and hidden dependence structure and when it comes to tensor time series analysis, the role of latent variables are more important due to the complex independence structure in the array data. Therefore, in this study we consider both latent and observed tensor time series variables. Usually, observed tensors in a tensor time series model shows the state of a dynamical system which is measured at a discrete period of time. As like any time series model, the main objective of a tensor time series analysis is to forecast future data. The traditional approach to model tensor data uses linear dynamical system (LDS) with Gaussian noise terms. In LDS approach, the latent (hidden) state and observation arrays at each time frame need to be converted to vector. Converting the tensor variables to vectors not only lose the inherent structure in the array data, but also dramatically increase the number of the parameters of the model. However, multilinear dynamical system (MLDS) proposed by Rogers et al. (2013)[14] to model tensor time series data which use an expectation-maximization (EM) algorithm to estimate the parameters of the model is more useful and efficient than LDS method. The MLDS models each tensor time series observation as the multilinear function of the latent tensor time series. Then, the latent tensor time series are again derived and updated with respect to a multilinear projection. We discuss LDS approach, EM algorithm, MLDS approach, the parameter estimation procedures, and a simulation study in this chapter.

4.2 LINEAR DYNAMICAL SYSTEM (LDS)

Linear Dynamical System (LDS)[19] can be defined as a multivariate time series model that represents the observation states through the hidden states. The LDS models the dynamics of the observation states and the hidden states according to the state transition probability $p(\mathbf{z}_{t+1}|\mathbf{z}_t)$, and the state observation probability $p(\mathbf{x}_t|\mathbf{z}_t)$, respectively. The following equations can be used to model these state probabilities.

$$\begin{cases} \mathbf{x}_t = \mathbf{C}\mathbf{z}_t + \mathbf{w}_t \qquad \mathbf{w}_t \sim N(0, \mathbf{R}) \\ \\ \mathbf{z}_{t+1} = \mathbf{A}\mathbf{z}_t + \mathbf{v}_t \qquad \mathbf{v}_t \sim N(0, \mathbf{Q}), \end{cases} \tag{4.1}$$

where $\mathbf{x}_t$ is the observed state at each time frame t, $\mathbf{z}_t$ is the hidden state at each time frame t, $\mathbf{A}$ is the *transition matrix* that linearly transforms $\mathbf{z}_{t+1}$ to $\mathbf{z}_t$, $\mathbf{C}$ is the *observation matrix* that linearly transforms $\mathbf{z}_t$ to $\mathbf{x}_t$, noise terms $\mathbf{w}_t$ and $\mathbf{v}_t$ are modeled as normal distributions with zero mean and variance matrices $\mathbf{R}$ and $\mathbf{Q}$ respectively. The initial latent state distribution $\mathbf{z}_1$ with mean $\mathbf{u}_0$ and covariance $\mathbf{Q}_0$, i.e., $\mathbf{z}_1 \sim \mathbf{N}(\mathbf{u}_0, \mathbf{Q}_0)$. Then, the parameter set in LDS can be written as $\boldsymbol{\theta} = (\mathbf{A}, \mathbf{C}, \mathbf{R}, \mathbf{Q}, \mathbf{u}_0, \mathbf{Q}_0)$. Next, the expectation-maximization algorithm is useful for estimate the set of parameters $\boldsymbol{\theta}$.

4.3 EXPECTATION MAXIMIZATION (EM) ALGORITHM

The EM algorithm is proposed by Dempster et al. (1977)[2] is an iterative procedure to estimate the parameters of the model by maximizing the likelihood function of observations when there are some latent or missing variables. We, first discuss the EM algorithm, and then explain how the EM estimation procedure for LDS works. In the EM algorithm there are two steps, expectation step and a maximization step. Assume that the joint distribution of observation and latent variable $p(\mathbf{Z}, \mathbf{X})$ and $\boldsymbol{\theta}^{old}$ is the current estimate of $\boldsymbol{\theta}$. Then the steps of the EM algorithm is as follows

1. **E-step**: Calculate the expectation of the log-likelihood function

$$F(\boldsymbol{\theta}, \boldsymbol{\theta}^{old}) = E_{\mathbf{z}|\boldsymbol{\theta}^{old}} \left[\ln p(\mathbf{Z}, \mathbf{X}), \boldsymbol{\theta}^{old} \right]$$

2. **M-step**: Maximize $F(\boldsymbol{\theta}, \boldsymbol{\theta}^{old})$ and update $\boldsymbol{\theta}$

$$\boldsymbol{\theta}^{new} = \arg\max_{\boldsymbol{\theta}}[F(\boldsymbol{\theta}, \boldsymbol{\theta}^{old})].$$

Iteratively apply step 1 and step 2 until the likelihood convergence obtained.

4.4 EM ALGORITHM FOR LDS MODEL

In practical situations, instead of maximizing the likelihood function of the data directly, the EM algorithm can be used to maximize the expectation of the joint likelihood function of observed and latent variables with respect to the distribution of the latent variables. The expectation of the joint likelihood function of both observed and latent variables can be derived as follows.

Suppose, we have only observed time series variables $\mathbf{Y} = (\mathbf{y}_1, \ldots, \mathbf{y}_N)$ in the model. We can write the joint probability function of $\mathbf{Y}$ under the condition of $\mathbf{y}_{t-1}$ and $\mathbf{y}_{t+1}$ are independent of $\mathbf{y}_t$ as follows

$$p(\mathbf{Y}) = p(\mathbf{y}_1) \prod_{t=1}^{N-1} p(\mathbf{y}_{t+1}|\mathbf{y}_t). \tag{4.2}$$

This probability function is a direct extention of observed Markov chain model obtained by using the product rule, see Bishop (2006)[1]. Now, consider the LDS model in Section 4.2, and assume it contains the observed and the hidden variables $\mathbf{X}$ and $\mathbf{Z}$ respectively. Then, the joint probability function of observed and hidden variables can be obtained by substituting $\mathbf{Y} = (\mathbf{Z}, \mathbf{X})$ into (4.2) as follows

$$\begin{aligned} p(\mathbf{Z}, \mathbf{X}) &= p(\mathbf{z}_1, \mathbf{x}_1) \prod_{t=1}^{N-1} p\Big((\mathbf{z}_{t+1}, \mathbf{x}_{t+1})|(\mathbf{z}_t, \mathbf{x}_t)\Big) \\ &= p(\mathbf{x}_1|\mathbf{z}_1)p(\mathbf{z}_1) \prod_{t=1}^{N-1} p(\mathbf{x}_{t+1}|\mathbf{z}_{t+1}, \mathbf{x}_t, \mathbf{z}_t) \prod_{t=1}^{N-1} p(\mathbf{z}_{t+1}|\mathbf{x}_t, \mathbf{z}_t) \\ &= p(\mathbf{x}_1|\mathbf{z}_1)p(\mathbf{z}_1) \prod_{t=1}^{N-1} p(\mathbf{x}_{t+1}|\mathbf{z}_{t+1}) \prod_{t=1}^{N-1} p(\mathbf{z}_{t+1}|\mathbf{z}_t) \end{aligned}$$

We can write the joint probability function of observed and hidden variables as

$$p(\mathbf{Z}, \mathbf{X}) = p(\mathbf{z}_1) \prod_{t=1}^{N} p(\mathbf{x}_t|\mathbf{z}_t) \prod_{t=1}^{N-1} p(\mathbf{z}_{t+1}|\mathbf{z}_t). \tag{4.3}$$

By taking the natural logarithm on both sides of (4.3) we can obtain the complete log-likelihood function as

$$\ln p(\mathbf{Z}, \mathbf{X}) = \ln p(\mathbf{z}_1) + \sum_{t=1}^{N} \ln p(\mathbf{x}_t|\mathbf{z}_t) + \sum_{t=1}^{N-1} \ln p(\mathbf{z}_{t+1}|\mathbf{z}_t). \tag{4.4}$$

Now, let us take the expectation with respect to the distribution of latent variables $\mathbf{Z} = (\mathbf{z}_1, \ldots, \mathbf{z}_N)$ from both sides of (4.4), and call it $F(\boldsymbol{\theta}, \boldsymbol{\theta}^{old})$. That is, we have

$$\begin{aligned} F(\boldsymbol{\theta}, \boldsymbol{\theta}^{old}) &= E_{\mathbf{z}|\boldsymbol{\theta}^{old}} \left[\ln p(\mathbf{Z}, \mathbf{X})\right] \\ &= E_{\mathbf{z}|\boldsymbol{\theta}^{old}} \left[\ln p(\mathbf{z}_1)\right] + E_{\mathbf{z}|\boldsymbol{\theta}^{old}} \left[\sum_{t=1}^{N} \ln p(\mathbf{x}_t|\mathbf{z}_t)\right] \\ &\quad + E_{\mathbf{z}|\boldsymbol{\theta}^{old}} \left[\sum_{t=1}^{N-1} \ln p(\mathbf{z}_{t+1}|\mathbf{z}_t)\right]. \end{aligned} \tag{4.5}$$

As it described in Section 4.3 the EM algorithm alternates between maximizing the function $F(\boldsymbol{\theta}, \boldsymbol{\theta}^{\boldsymbol{old}})$ with respect to $\boldsymbol{\theta}^{new} = \arg\max_{\boldsymbol{\theta}} F(\boldsymbol{\theta}, \boldsymbol{\theta}^{\boldsymbol{old}})$ and with respect to the distribution of latent states, keeping others fixed. The E-step in the EM algorithm depends on $E[\mathbf{z}_t|\mathbf{X}]$, $E[\mathbf{z}_t\mathbf{z}_t^T|\mathbf{X}]$ and $E[\mathbf{z}_{t+1}\mathbf{z}_t^T|\mathbf{X}]$, which are sufficient statistics to compute (4.5). We can use Kalman filtering proposed by Kalman (1960)[8] and Kalman smoothing proposed by Rauch (1963)[13] algorithms to calculate these sufficient statistics.

Algorithm 2 Kalman Filter Algorithm for LDS[19]

INPUT: $\boldsymbol{\theta} = (\mathbf{A}, \mathbf{C}, \mathbf{R}, \mathbf{Q}, \mathbf{u}_0, \mathbf{Q}_0)$

Initialize:

$\hat{\mathbf{z}}_{1|1} = \mathbf{u}_0$ and $P_{1|1} = \mathbf{Q}_0$

for $t = 1 \rightarrow N - 1$ **do**

$\hat{\mathbf{z}}_{t+1|t} = \mathbf{A}\hat{\mathbf{z}}_{t|t}$

$\mathbf{P}_{t+1|t} = \mathbf{A}\mathbf{P}_{t|t}\mathbf{A}^T + \mathbf{Q}$

Updating measures:

$K_{t+1} = \mathbf{P}_{t+1|t}\mathbf{C}^T(\mathbf{C}\mathbf{P}_{t+1|t}\mathbf{C}^T + \mathbf{R})^{-1}$

$\hat{\mathbf{z}}_{t+1|t+1} = \hat{\mathbf{z}}_{t+1|t} + K_{t+1}(\mathbf{x}_t - \mathbf{C}\hat{\mathbf{z}}_{t+1|t})$

$\mathbf{P}_{t+1|t+1} = \mathbf{P}_{t+1|t} - K_{t+1}\mathbf{C}\mathbf{P}_{t+1|t}$

end for

The algorithm 2 takes the input of the set of parameters $\boldsymbol{\theta}$ and output $\hat{\mathbf{z}}_{t+1|t}$, $\hat{\mathbf{z}}_{t|t}$, $\mathbf{P}^T_{t+1|t+1}$,$\mathbf{P}_{t+1|t+1}$, and K_{t+1} for $t = 1, \ldots, N-1$. Where,

$$\begin{aligned}
\hat{\mathbf{z}}_{t+1|t} &= E[\mathbf{z}_{t+1}|\{\mathbf{x}\}_1^{t-1}] \\
\hat{\mathbf{z}}_{t|t} &= E[\mathbf{z}_t|\{\mathbf{x}\}_1^{t-1}] \\
\mathbf{P}_{t+1|t} &= E[(\mathbf{z}_{t+1} - \hat{\mathbf{z}}_{t+1|t})(\mathbf{z}_{t+1} - \hat{\mathbf{z}}_{t+1|t})^T] \\
\mathbf{P}_{t+1|t+1} &= E[(\mathbf{z}_t - \hat{\mathbf{z}}_{t|t})(\mathbf{z}_t - \hat{\mathbf{z}}_{t|t})^T].
\end{aligned}$$

Algorithm 3 Kalman Smoothing Algorithm for LDS[19]

INPUT: $\boldsymbol{\theta} = (\mathbf{A}, \mathbf{C}, \mathbf{R}, \mathbf{Q}, \mathbf{u}_0, \mathbf{Q}_0)$ and output from Kalman filter algorithm

Initialize:

$W_{N|N} = P_{N|N} + \hat{\mathbf{z}}_{N|N}\hat{\mathbf{z}}_{N|N}^T$

$H_{N-1} = P_{N-1|N-1}\mathbf{A}^T(P_{N|N-1})^{-1}$

$P_{N-1|N} = P_{N-1|N-1} + H_{N-1}(P_{N|N} - P_{N|N-1})H_{N-1}^T$

$\hat{\mathbf{z}}_{N-1|N} = \hat{\mathbf{z}}_{N-1|N-1} + H_{N-1}(\hat{\mathbf{z}}_{N|N} - \mathbf{A}\hat{\mathbf{z}}_{N-1|N-1})$

$P_{N,N-1|N} = (I - K_N\mathbf{C})\mathbf{A}P_{N-1|N-1}$

$W_{N,N-1|N} = P_{N,N-1|N} + \hat{\mathbf{z}}_{N|N}\hat{\mathbf{z}}_{N-1|N}^T$

for $t = N-2 \to 1$ **do**

    $W_{t+1|N} = P_{t+1|N} + \hat{\mathbf{z}}_{t+1|N}\hat{\mathbf{z}}_{t+1|N}^T$

    $H_t = P_{t|t}\mathbf{A}^T(P_{t+1|t})^{-1}$

    $P_{t+1,t|T} = P_{t+1|t+1}H_t^T + H_{t+1}(P_{t+2,t+1|N} - \mathbf{A}P_{t+1|t+1})H_t^T$

    $W_{t+1,t|T} = P_{t+1,t|N} + \hat{\mathbf{z}}_{t+1|N}\hat{\mathbf{z}}_{t|N}^T$

    $\hat{\mathbf{z}}_{t|N} = \hat{\mathbf{z}}_{t|t} + H_t(\hat{\mathbf{z}}_{t+1|N} - \mathbf{A}\hat{\mathbf{z}}_{t|t})$

    $P_{t|N} = P_{t|t} + H_t(P_{t+1|N} - P_{t+1|t})H_t^T$

end for

The algorithm 3 take input as the set of parameters $\boldsymbol{\theta}$ and the output from the algorithm 2. Then, gives the output $\hat{\mathbf{z}}_{t|N}$, $W_{t+1|N}$, and $W_{t+1,t|N}$ for $t = 1, \ldots, N-1$. Where,

$$\begin{aligned} W_{t|N} &= E[\mathbf{z}_t\mathbf{z}_t^T|\mathbf{x}] \\ W_{t+1,t|N} &= E[\mathbf{z}_{t+1}\mathbf{z}_t^T|\mathbf{x}]. \end{aligned}$$

The M-step in the EM algorithm estimate and update each parameters in $\boldsymbol{\theta}$ by taking partial derivative of the expected log-likelihood with respect to the corresponding parameter and then solve it with respect to the parameter.

First, consider the parameters $\mathbf{u}_0$ and $\mathbf{Q}_0$ defined in Section 4.2. Now, we substitute $p(\mathbf{z}_1|\mathbf{u}_0, \mathbf{Q}_0)$ by using the Gaussian distribution to equation (4.5) and obtain

$$F(\boldsymbol{\theta}, \boldsymbol{\theta}^{old}) = -\frac{1}{2} ln|\mathbf{Q}_0| - E_{\mathbf{z}|\boldsymbol{\theta}^{old}} \left[\frac{1}{2} (\mathbf{z}_1 - \mathbf{u}_0)^T \mathbf{Q}_0^{-1} (\mathbf{z}_1 - \mathbf{u}_0) \right] + \text{constant},$$

where the constant term represents all terms that do not depend on $\mathbf{u}_0$ and $\mathbf{Q}_0$.

Similarly, to optimize $\mathbf{C}$ and $\mathbf{R}$ we can substitute $p(\mathbf{x}_t|\mathbf{z}_t, \mathbf{C}, \mathbf{R})$ by using the Gaussian distribution to equation (4.5) and obtain

$$F(\boldsymbol{\theta}, \boldsymbol{\theta}^{old}) = -\frac{N}{2} ln|\mathbf{R}| - E_{\mathbf{z}|\boldsymbol{\theta}^{old}} \left[\frac{1}{2} \sum_{t=1}^{N} (\mathbf{x}_t - \mathbf{C}\mathbf{z}_t)^T \mathbf{R}^{-1} (\mathbf{x}_t - \mathbf{C}\mathbf{z}_t) \right] + \text{constant},$$

where the constant term comprises all of the terms that are not related to $\mathbf{C}$ and $\mathbf{R}$. Maximizing above equation with respect to $\mathbf{C}$ and $\mathbf{R}$ then gives

$$\mathbf{C}^{new} = \left(\sum_{t=1}^{N} \mathbf{x}_t E[\mathbf{z}_t^T] \right) \left(\sum_{t=1}^{N} E[\mathbf{z}_t \mathbf{z}_t^T] \right)^{-1} \tag{4.6}$$

$$\mathbf{R}^{new} = \frac{1}{N} \sum_{t=1}^{N} \{ \mathbf{x}_t \mathbf{x}_t^T - \mathbf{C}^{new} E[\mathbf{z}_t] \mathbf{x}_t^T - \mathbf{x}_t E[\mathbf{z}_t^T] \mathbf{C}^{new} + \mathbf{C}^{new} E[\mathbf{z}_t \mathbf{z}_t^T] \mathbf{C}^{new} \}. \tag{4.7}$$

Finally, to find the new values of $\mathbf{A}$ and $\mathbf{Q}$ defined in Section 4.2 we can substitute $p(\mathbf{z}_{t+1}|\mathbf{z}_t, \mathbf{A}, \mathbf{Q})$ by using the Gaussian distribution to equation (4.5) and obtain

$$F(\boldsymbol{\theta}, \boldsymbol{\theta}^{old}) = -\frac{N-1}{2} ln|\mathbf{Q}| - E_{\mathbf{z}|\boldsymbol{\theta}^{old}} \left[\frac{1}{2} \sum_{t=1}^{N-1} (\mathbf{z}_{t+1} - \mathbf{A}\mathbf{z}_t)^T \mathbf{Q}^{-1} (\mathbf{z}_{t+1} - \mathbf{A}\mathbf{z}_t) \right] + \text{constant},$$

where the constant term represents all terms that do not related to $\mathbf{A}$ and $\mathbf{Q}$. Maximizing above equation with respect to these parameters then gives

$$\mathbf{A}^{new} = \left(\sum_{t=1}^{N-1} E[\mathbf{z}_{t+1} \mathbf{z}_t^T] \right) \left(\sum_{t=1}^{N} E[\mathbf{z}_t \mathbf{z}_t^T] \right)^{-1} \tag{4.8}$$

$$\mathbf{Q}^{new} = \frac{1}{N-1} \sum_{t=1}^{N-1} \{ E[\mathbf{z}_{t+1} \mathbf{x}_{t+1}^T] - \mathbf{A}^{new} E[\mathbf{z}_t \mathbf{z}_{t+1}^T] - E[\mathbf{z}_{t+1} \mathbf{z}_t^T] \mathbf{A}^{new} + \mathbf{A}^{new} E[\mathbf{z}_t \mathbf{z}_t^T] (\mathbf{A}^{new})^T \}. \tag{4.9}$$

4.5 RANDOM TENSORS

Let N be the set of all positive integers and let $I = I_1 \times I_2 \times \cdots \times I_M \in N^M$, where $M \in N$. Define $\mathcal{X} \in \mathbb{R}^{\mathbf{I_1} \times \cdots \times \mathbf{I_M}}$ to be a random tensor.

Now, suppose $\mathcal{U} \in \mathbb{R}^I$ and $\mathcal{S} \in \mathbb{R}^{II}$. The multivariate normal distribution for $\text{vec}(\mathcal{X})$ with mean $\text{vec}(\mathcal{U})$ and covariance matrix $\text{mat}(\mathcal{S})$ can be denoted as $\mathcal{N}(\text{vec}(\mathcal{U}), \text{mat}(\mathcal{S}))$. Then, $\mathcal{X}$ has the tensor normal distribution with mean tensor $\mathcal{U}$ and covariance $\mathcal{S}$ and represent as $\mathcal{X} \sim \mathcal{N}(\mathcal{U}, \mathcal{S})$. Therefore, we have

$$\mathcal{X} \sim \mathcal{N}(\mathcal{U}, \mathcal{S}) \Leftrightarrow \text{vec}(\mathcal{X}) \sim \mathcal{N}(\text{vec}(\mathcal{U}), \text{mat}(\mathcal{S})). \tag{4.10}$$

We consider a particular case of tensor normal distribution for tensors $\mathcal{X}$ and $\mathcal{C}$ which is called the multilinear Gaussian distribution. Suppose, $I = I_1 \times I_2 \times \cdots \times I_M, J = J_1 \times J_2 \times \cdots \times J_M, IJ = I_1 \times I_2 \times \cdots \times I_M \times J_1 \times J_2 \times \cdots \times J_M$, and let tensor $\mathcal{X} \in \mathbb{R}^I$ and tensor $\mathcal{Z} \in \mathbb{R}^J$ then the joint distribution of $\mathcal{X}$ and $\mathcal{Z}$ is given as

$$\mathcal{Z} \sim \mathcal{N}(\mathcal{U}, \mathcal{G}) \text{ and } \mathcal{X}|\mathcal{Z} \sim \mathcal{N}(\mathcal{C} \circledast \mathcal{Z}, \mathcal{S}), \tag{4.11}$$

where "$\circledast$" is the product of two tensors $\mathcal{C} \in \mathbb{R}^{IJ}$ and $\mathcal{Z} \in \mathbb{R}^J$ which is defined in section 2.1 where $(\mathcal{C} \circledast \mathcal{Z})_{i_1 \ldots i_M} = \sum_{j_1 \ldots j_M} c_{i_1 \ldots i_M j_1 \ldots j_M} z_{j_1 \ldots j_M}$. Given the joint distribution of tensor $\mathcal{X}$ and tensor $\mathcal{Z}$ in (4.11), the marginal distribution of $\mathcal{X}$ and the conditional distribution of $\mathcal{Z}|\mathcal{X}$ can be obtain by using the following Lemma.

Lemma 4.1. *Suppose both $I, J \in N^M, M \in N$, and the joint distribution of tensor $\mathcal{X} \in \mathbb{R}^I$ and tensor $\mathcal{Z} \in \mathbb{R}^J$ is given by equation* (4.11)*. Then, the marginal distribution of tensor $\mathcal{X}$ is given by*

$$\mathcal{X} \sim \mathcal{N}(\mathcal{C} \circledast \mathcal{U}, \mathcal{C} \circledast \mathcal{G} \circledast \mathcal{C}^T + \mathcal{S}), \tag{4.12}$$

where $\mathcal{C}^T \in \mathbb{R}^{JI}$ is the transpose of tensor $\mathcal{C}$, which means, $c^T_{j_1 \ldots j_M i_1 \ldots i_M} = c_{i_1 \ldots i_M j_1 \ldots j_M}$. Moreover, the conditional distribution of $\mathcal{Z}|\mathcal{X}$ is

$$\mathcal{Z}|\mathcal{X} \sim \mathcal{N}(\hat{\mathcal{U}}, \hat{\mathcal{G}}), \tag{4.13}$$

where $\text{vec}(\hat{\mathcal{U}}) = \big(\mu + W(\text{vec}(\mathcal{X}) - \text{mat}(C)\mu)\big)$, $\text{mat}(\hat{\mathcal{G}}) = \big(\Omega - W\,\text{mat}(C)\Omega\big)$, $\mu = \text{vec}(\mathcal{U})$, $\Omega = \text{mat}(\mathcal{G})$, $\Sigma = \text{mat}(\mathcal{S})$, *and* $W = \Omega\,\text{mat}(\mathcal{C})^T[\text{mat}(\mathcal{C})\Omega\,\text{mat}(\mathcal{C})^T + \Sigma]^{-1}$.

The proof of Lemma 4.1 is provided in the Appendix II.

4.6 MULTILINEAR DYNAMICAL SYSTEM (MLDS)

In this section of the thesis, we explain the MLDS procedure introduced by Rogers et al. (2013)[14] to model tensor time series data. The goal here is to build a tensor time series model based on tensor time series $\mathcal{X}_1, \mathcal{X}_2, \ldots, \mathcal{X}_N$ where $\mathcal{X}_t \in \mathbb{R}^{I_1 \times \cdots \times I_M}$ for $t = 1, \ldots, N$, that preserves the tensor structure. The sequence of latent tensors are $\mathcal{Z}_\mathbf{1}, \mathcal{Z}_\mathbf{2}, \ldots, \mathcal{Z}_\mathbf{N}$ where $\mathcal{Z}_t \in \mathbb{R}^{J_1 \times \cdots \times J_M}$ for $t = 1, \ldots, N$. For the MLDS model, we vectorize each tensor $\mathcal{X}_\mathbf{t}$ and then we estimate the parameters by using a similar EM algorithm given for the LDS models, and finally we convert back the results to then tensor structures. The MLDS model consists of a sequence of latent tensors $\mathcal{Z}_t$ where $\mathcal{Z}_\mathbf{t} \in \mathbb{R}^{J_1 \times \cdots \times J_M}$ for $t = 1, \ldots, N$. Assume, for each observable tensor $\mathcal{X}_t \in \mathbb{R}^{I_1 \times \cdots \times I_M}$ there is an associated latent tensor variable $\mathcal{Z}_t$, where $\mathcal{Z}_t \in \mathbb{R}^{J_1 \times \cdots \times J_M}$. Every latent tensor $\mathcal{Z}_t$ emits a tensor observation $\mathcal{X}_t \in \mathbb{R}^{I_1 \times \cdots \times I_M}$. Moreover, assume the initial latent tensor $\mathcal{Z}_1$ is normally distributed as,

$$\mathcal{Z}_\mathbf{1} \sim \mathcal{N}(\mathcal{U}_\mathbf{0}, \mathcal{Q}_\mathbf{0}). \tag{4.14}$$

Then following the conditional distribution of $\mathcal{Z}_{t+1}$ given $\mathcal{Z}_t$ for $t = 1, \ldots, N-1$ is given as follows

$$\mathcal{Z}_{\mathbf{t+1}} | \mathcal{Z}_\mathbf{t} \sim \mathcal{N}(\mathcal{A} \circledast \mathcal{Z}_\mathbf{t}, \mathcal{Q}), \tag{4.15}$$

where $\mathcal{Q}$ is the conditional covariance tensor with dimension $J_1 \times \cdots \times J_M \times J_1 \times \cdots \times J_M$, and $\mathcal{A}$ is the *transition tensor* with dimension $J_1 \times \cdots \times J_M \times J_1 \times \cdots \times J_M$ which is dynamically related to the latent tensors. In the MLDS model, the transition tensor $\mathcal{A}$ can be factorized as $\text{mat}(\mathcal{A}) = \mathbf{A}^{(M)} \otimes \cdots \otimes \mathbf{A}^{(1)}$ along the modes of $\mathcal{Z}_t$. For each latent tensor $\mathcal{Z}_\mathbf{t}$, the corresponding observable tensor $\mathcal{X}_\mathbf{t}$ is given by the following distribution

$$\mathcal{X}_t | \mathcal{Z}_t \sim \mathcal{N}(\mathcal{C} \circledast \mathcal{Z}_t, \mathcal{R}), \tag{4.16}$$

where $\mathcal{R}$ is the covariance tensor with the dimension $I_1 \times \cdots \times I_M \times I_1 \times \cdots \times I_M$ of $\mathcal{X}_t$, and $\mathcal{C}$ is the *Projection tensor* with the dimension $I_1 \times \cdots \times I_M \times J_1 \times \cdots \times J_M$. Here, the role of projection tensor $\mathcal{C}$ is to transform the latent tensor $\mathcal{Z}_t$ into the observable tensor $\mathcal{X}_t$ multilinearly. We can factorize $\mathcal{C}$ as $\text{mat}(\mathcal{C}) = \mathbf{C}^{(M)} \otimes \cdots \otimes \mathbf{C}^{(1)}$ in a similar way as $\mathcal{A}$. Since, both $\mathcal{A}$ and $\mathcal{C}$ can be factorized in the MLDS model, we need to use a gradient based method to estimate the parameters. Also, note that the MLDS becomes an LDS by vectorizing $\mathcal{Z}_t$ and $\mathcal{X}_t$ with factorized projection matrix $\text{mat}\,(\mathcal{C})$ and transition matrix $\text{mat}\,(\mathcal{A})$ respectively. These factorizations of transition and projection tensors help us to reduce the dimensionality of tensors and reduce the number of parameters from $\prod_{m=1}^{M} J_m^2 + \prod_{m=1}^{M} I_m J_m$ to $\sum_{m=1}^{M} J_m^2 + \sum_{m=1}^{M} I_m J_m$. Figure 4.1 describes a MLDS with three modes.

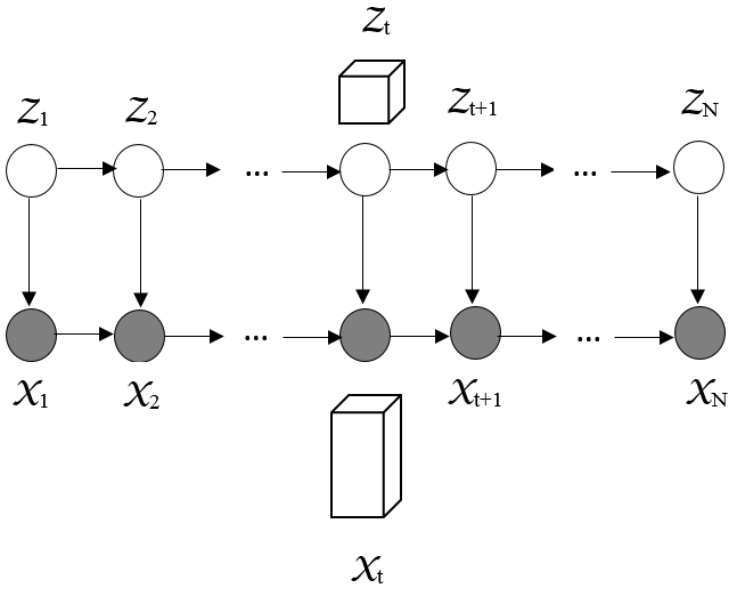

Figure 4.1: Structure of the MLDS with three modes

4.7 ESTIMATION

Suppose, the tensor observations $\mathcal{X}_1, \ldots, \mathcal{X}_N$ are given. Let $\boldsymbol{\theta} = (\mathbf{A}, \mathbf{C}, \mathbf{R}, \mathbf{Q}, \mathcal{U}_0, \mathcal{Q}_0)$ be the set of all parameters in the tensor time series model. This set of parameters $\boldsymbol{\theta}$ can be estimated from the MLDS model. Since the MLDS model contains the hidden or latent

tensor variables $\mathcal{Z}_t$, we cannot directly maximize the likelihood function of the data with respect to the model parameters. However, we can overcome this difficulty by using the EM algorithm. We alternatively update the expectations of latent tensor variables and the parameters of the model until the complete likelihood of the data converges. Once the convergence criterion satisfied for $\boldsymbol{\theta}$, the EM algorithm would return the estimated values of each parameter. The complete likelihood of the MLDS model can be written in the following form

$$L(\boldsymbol{\theta}|\mathcal{Z}_1, \mathcal{X}_1, \ldots, \mathcal{Z}_N, \mathcal{X}_N) = L\big(\operatorname{vec}(\boldsymbol{\theta})|\operatorname{vec}(\mathcal{Z}_1), \operatorname{vec}(\mathcal{X}_1), \ldots, \operatorname{vec}(\mathcal{Z}_N), \operatorname{vec}(\mathcal{X}_N)\big), \quad (4.17)$$

where the model parameters of MLDS approach are vectorized to $\operatorname{vec}(\boldsymbol{\theta}) = \big(\operatorname{vec}(\mathcal{U}_0), \operatorname{vec}\big(\mathcal{Q}_0\big), \operatorname{vec}(\mathcal{Q}\big), \operatorname{mat}(\mathcal{A}), \operatorname{mat}(\mathcal{R}), \operatorname{mat}(\mathcal{C})\big)$. We use tensor normal distribution (4.10) for computations in (4.17). The complete log-likelihood function can be written as

$$p\big(\operatorname{vec}(\mathcal{Z}), \operatorname{vec}(\mathcal{X})\big) = p\big(\operatorname{vec}(\mathcal{Z}_1)\big) \prod_{t=1}^{N} p\big(\operatorname{vec}(\mathcal{X}_t)|\operatorname{vec}(\mathcal{Z}_t)\big) \prod_{t=1}^{N-1} p\big(\operatorname{vec}(\mathcal{Z}_{t+1})|\operatorname{vec}(\mathcal{Z}_t)\big).$$

By taking the natural logarithm from both sides of the above equation, we can obtain the complete log-likelihood function as

$$\ln p\big(\operatorname{vec}(\mathcal{Z}\big), \operatorname{vec}(\mathcal{X})) = \ln p\big(\operatorname{vec}(\mathcal{Z}_1)\big) + \sum_{t=1}^{N} \ln p\big(\operatorname{vec}(\mathcal{X}_t)|\operatorname{vec}(\mathcal{Z}_t)\big) + \sum_{t=1}^{N-1} \ln p\big(\operatorname{vec}(\mathcal{Z}_{t+1})|\operatorname{vec}(\mathcal{Z}_t)\big).$$

Now, taking the expectation on both sides of the log-likelihood function with respect to the distribution of the latent variables, we have

$$\begin{aligned} F(\boldsymbol{\theta}, \boldsymbol{\theta}^{old}) &= E_{\mathcal{Z}|\boldsymbol{\theta}^{old}}\left[\ln p\big(\operatorname{vec}(\mathcal{Z}), \operatorname{vec}(\mathcal{X})\big)\right] \\ &= E_{\mathcal{Z}|\boldsymbol{\theta}^{old}}\left[\ln p\big(\operatorname{vec}(\mathcal{Z}_1)\big)\right] + E_{\mathcal{Z}|\boldsymbol{\theta}^{old}}\left[\sum_{t=1}^{N} \ln p\big(\operatorname{vec}(\mathcal{X}_t)|\operatorname{vec}(\mathcal{Z}_t)\big)\right] \\ &+ E_{\mathcal{Z}|\boldsymbol{\theta}^{old}}\left[\sum_{t=1}^{N-1} \ln p\big(\operatorname{vec}(\mathcal{Z}_{t+1})|\operatorname{vec}(\mathcal{Z}_t)\big)\right]. \end{aligned} \quad (4.18)$$

Since, the vectorized MLDS model indeed is an LDS model, therefore we can use the rules of the EM algorithm introduced for the LDS case in section 4.4 for the E-step

and M-step to estimate all parameters except $\text{mat}(\mathcal{A})$ and $\text{mat}(\mathcal{C})$ since the transition and projection tensors $\mathcal{A}$ and $\mathcal{C}$ are factorizable tensors. Therefore, we need to update $\mathcal{A}$ and $\mathcal{C}$ by using another method than the standard LDS. We maximize the expected complete log-likelihood locally by computing the gradient with respect to the vector $\mathbf{e} = [\text{vec}\, C^{(1)T}, \ldots, \text{vec}\, C^{(M)T}]^T \in \mathbb{R}^{\sum_m I_m J_m}$, which is obtained by putting together all of the vectorized projection matrices in the M-step of $\mathcal{C}$. Now, we can derive, complete log-likelihood as follows. First, notice that the projection tensor $\mathcal{C}$ is only involved in the likelihood function of $\mathcal{X}_t|\mathcal{Z}_t$, which is

$$L = -\frac{N}{2} \ln |\,\text{mat}(\mathcal{R})| - \frac{1}{2} \sum_{t=1}^{N} \big(\,\text{vec}(\mathcal{X}_t) - \text{vec}(\mathcal{C} \circledast \mathcal{Z}_t)\big)^T \big(\,\text{mat}(\mathcal{R})\big)^{-1} \big(\,\text{vec}(\mathcal{X}_t) - \text{vec}(\mathcal{C} \circledast \mathcal{Z}_t)\big)$$

$$+ \text{ constant},$$

where constant term represents all terms that do not depend on $\mathcal{C}$ and $\mathcal{R}$. Then, the complete log-likelihood function associated with $\mathcal{C}$ as follows

$$\ln L(\mathbf{e}) = -\frac{1}{2} \sum_{t=1}^{N} \Big(\big(\,\text{vec}(\mathcal{X}_t) - \text{vec}(\mathcal{C} \circledast \mathcal{Z}_t)\big)^T \big(\,\text{mat}(\mathcal{R})\big)^{-1} \big(\,\text{vec}(\mathcal{X}_t) - \text{vec}(\mathcal{C} \circledast \mathcal{Z}_t)\big) \Big)$$

$$+ \text{ constant},$$

where the constant term represents all terms of the log-likelihood function that are not related to the projection tensor $\mathcal{C}$. By applying lemma 2.1, we have

$$\ln L(\mathbf{e}) - \frac{1}{2} \sum_{t=1}^{N} \Big(\big(\,\text{vec}(\mathcal{X}_t) - \text{mat}(\mathcal{C})\,\text{vec}(\mathcal{Z}_t)\big)^T \big(\,\text{mat}(\mathcal{R})\big)^{-1} \big(\,\text{vec}(\mathcal{X}_t) - \text{mat}(\mathcal{C})\,\text{vec}(\mathcal{Z}_t)\big) \Big)$$

$$+ \text{constant}.$$

After simplifying we obtain,

$$\begin{aligned} \ln L(\mathbf{e}) = {} & \frac{1}{2} \sum_{t=1}^{N} \big(\,\text{vec}(\mathcal{X}_t)^T \,\text{mat}(\mathcal{R})^{-1} \,\text{mat}(\mathcal{C})\,\text{vec}(\mathcal{Z}_t)\big) \\ & + \frac{1}{2} \sum_{t=1}^{N} \big(\,\text{vec}(\mathcal{Z}_t)^T \,\text{mat}(\mathcal{C})^T \,\text{mat}(\mathcal{R})^{-1}\,\text{vec}(\mathcal{X}_t)\big) \\ & - \frac{1}{2} \sum_{t=1}^{N} \big(\,\text{vec}(\mathcal{Z}_t)^T \,\text{mat}(\mathcal{C})^T \,\text{mat}(\mathcal{R})^{-1} \,\text{mat}(\mathcal{C})\,\text{vec}(\mathcal{Z}_t)\big) + \text{constant}. \end{aligned}$$

By using a property of the trace operation introduced in Section 2.1.7, we have

$$\ln L(\mathbf{e}) = -\frac{1}{2}\operatorname{tr}\Big[\sum_{t=1}^{N}\big(\operatorname{mat}(\mathcal{R})^{-1}\operatorname{mat}(\mathcal{C})\operatorname{vec}(\mathcal{Z}_t)\operatorname{vec}(\mathcal{Z}_t)^T\operatorname{mat}(\mathcal{C})^T - 2\operatorname{mat}(\mathcal{R})^{-1}\operatorname{mat}(\mathcal{C})\operatorname{vec}(\mathcal{Z}_t)\operatorname{vec}(\mathcal{X}_t)^T\big)\Big] + \text{constant.}$$

or

$$\ln L(\mathbf{e}) = -\frac{1}{2}tr\Big[\operatorname{mat}(\mathcal{R})^{-1}\operatorname{mat}(\mathcal{C})\big(\sum_{t=1}^{N}\operatorname{vec}(\mathcal{Z}_t)\operatorname{vec}(\mathcal{Z}_t)^T\operatorname{mat}(\mathcal{C})^T - 2\sum_{t=1}^{N}\operatorname{vec}(\mathcal{Z}_t)\operatorname{vec}(\mathcal{X}_t)^T\big)\Big] + \text{constant.}$$

By taking the expectation on both sides of the conditional log-likelihood function $\mathcal{Z}_t|\mathcal{X}_t$, we obtain

$$l(\mathbf{e}) = -\frac{1}{2}\operatorname{tr}\Big[\operatorname{mat}(\mathcal{R})^{-1}\operatorname{mat}(\mathcal{C})\big(\sum_{t=1}^{N}E(\operatorname{vec}(\mathcal{Z}_t)\operatorname{vec}(\mathcal{Z}_t)^T\operatorname{mat}(\mathcal{C})^T) - 2\sum_{t=1}^{N}E(\operatorname{vec}(\mathcal{Z}_t))\operatorname{vec}(\mathcal{X}_t)^T\big)\Big] + \text{constant.}$$

Now, we can write the expected, complete log-likelihood function as follows

$$l(\mathbf{e}) = \operatorname{tr}\{\Gamma\operatorname{mat}(\mathcal{C})[\Phi\operatorname{mat}(\mathcal{C})^T - 2\Psi^T]\} + \text{constant}, \tag{4.19}$$

where constant term represents all terms that are not related to $\mathcal{C}$, $\Gamma = \operatorname{mat}(\mathcal{R})^{-1}$, $\Phi = \sum_{t=1}^{N}E(\operatorname{vec}\mathcal{Z}_t\operatorname{vec}\mathcal{Z}_t^T)$, and $\Psi = \sum_{t=1}^{N}\operatorname{vec}(\mathcal{X}_t)E(\operatorname{vec}\mathcal{Z}_t)^T$. Now, suppose k correspond to some projection matrix $C_{ij}^{(m)}$, and define indicator matrix $\Delta_{ij} \in \mathbf{R}^{I_m \times J_m}$ such that

$$\Delta_{ij} = \begin{cases} 1 & ;(i,j)^{th}entry \\ 0 & ;elsewhere. \end{cases}$$

The gradient $\nabla l(\mathbf{e}) \in \mathbb{R}^{\sum_m I_m J_m}$ can be defined in the elementwise form as

$$\nabla l(\mathbf{e})_k = 2\operatorname{tr}\{\Gamma\partial_{\mathbf{e}_k}\operatorname{mat}(\mathcal{C})[\Phi\operatorname{mat}(\mathcal{C})^T - \Psi^T]\}, \tag{4.20}$$

where $\partial_{\mathbf{e}_k} \operatorname{mat}(\mathcal{C})$ can be written as $\partial_{\mathbf{e}_k} \operatorname{mat}(\mathcal{C}) = C^{(M)} \otimes \cdots \otimes \Delta_{ij} \otimes \cdots \otimes C^{(1)}$. We can use of $\partial_{e_k} \operatorname{mat}(\mathcal{C})$ when $m = M$, by computing the trace of the product of two submatrices each with $\prod_{t \neq M} I_t$ rows and $\prod_{t \neq M} J_t$ columns:

$$\nabla l(\mathbf{e})_k = 2 \operatorname{tr}\{[C^{M-1} \otimes \cdots \otimes C^{(1)}]^T \Omega_{ij}\}, \tag{4.21}$$

where Ω_{ij} is the submatrix of $[\Gamma \operatorname{mat}(\mathcal{C})\Phi - \Psi]$.

Now, we can consider complete log-likelihood function associated with $\mathcal{A}$. The transition tensor is only involved in the log-likelihood function of $\mathcal{Z}_{t+1}|\mathcal{Z}_t$, given by

$$L(\mathbf{e}) = -\frac{1}{2} \sum_{t=1}^{N-1} \big(\operatorname{vec}(\mathcal{Z}_{t+1}) - \operatorname{vec}(\mathcal{A} \circledast \mathcal{Z}_t) \big)^T \big(\operatorname{mat}(\mathcal{Q}) \big)^{-1} \big(\operatorname{vec}(\mathcal{Z}_{t+1}) - \operatorname{vec}(\mathcal{C} \circledast \mathcal{Z}_t) \big) + \text{constant},$$

where constant represents all terms that do not depend on $\mathcal{A}$. By applying lemma 2.1, we have

$$L(\mathbf{e}) = -\frac{1}{2} \sum_{t=1}^{N-1} \big(\operatorname{vec}(\mathcal{Z}_{t+1}) - \operatorname{mat}(\mathcal{A}) \operatorname{vec}(\mathcal{Z}_t)^T \big) \big(\operatorname{mat}(\mathcal{Q}) \big)^{-1} \big(\operatorname{vec}(\mathcal{Z}_{t+1}) - \operatorname{mat}(\mathcal{A}) \operatorname{vec}(\mathcal{Z}_t) \big)$$

$$+ \text{constant}.$$

Then, we simplify the given log-likelihood function and obtain

$$\begin{aligned} \ln L(\mathbf{e}) = & \frac{1}{2} \sum_{t=1}^{N-1} \big(\operatorname{vec}(\mathcal{Z}_{t+1})^T \operatorname{mat}(\mathcal{Q})^{-1} \operatorname{mat}(\mathcal{A}) \operatorname{vec}(\mathcal{Z}_t) \big) \\ & + \frac{1}{2} \sum_{t=1}^{N-1} \big(\operatorname{vec}(\mathcal{Z}_t)^T \operatorname{mat}(\mathcal{A})^T \operatorname{mat}(\mathcal{Q})^{-1} \operatorname{vec}(\mathcal{Z}_{t+1}) \big) \\ & - \frac{1}{2} \sum_{t=1}^{N-1} \big(\operatorname{vec}(\mathcal{Z}_t)^T \operatorname{mat}(\mathcal{A})^T \operatorname{mat}(\mathcal{Q})^{-1} \operatorname{mat}(\mathcal{A}) \operatorname{vec}(\mathcal{Z}_t) \big) + \text{constant}. \end{aligned}$$

We use a property of trace operator, and simplify it as

$$\begin{aligned} \ln L(\mathbf{e}) = -\frac{1}{2} \operatorname{tr} \Big[\sum_{t=1}^{N-1} \Big(& \operatorname{mat}(\mathcal{Q})^{-1} \operatorname{mat}(\mathcal{A}) \operatorname{vec}(\mathcal{Z}_t) \operatorname{vec}(\mathcal{Z}_t)^T \operatorname{mat}(\mathcal{A})^T \\ & - 2 \operatorname{mat}(\mathcal{Q})^{-1} \operatorname{mat}(\mathcal{A}) \operatorname{vec}(\mathcal{Z}_t) \operatorname{vec}(\mathcal{Z}_{t+1})^T \Big) \Big] + \text{constant}. \end{aligned}$$

By simplifying we obtain

$$\ln L(\mathbf{e}) = -\frac{1}{2}\operatorname{tr}\Big[\operatorname{mat}(\mathcal{Q})^{-1}\operatorname{mat}(\mathcal{A})\Big(\sum_{t=1}^{N-1}\operatorname{vec}(\mathcal{Z}_t)\operatorname{vec}(\mathcal{Z}_t)^T\operatorname{mat}(\mathcal{A})^T - 2\sum_{t=1}^{N-1}\operatorname{vec}(\mathcal{Z}_t)\operatorname{vec}(\mathcal{Z}_{t+1})^T\Big)\Big] + \text{constant}.$$

By taking the expectation on both sides,

$$l(\mathbf{e}) = -\frac{1}{2}\operatorname{tr}\Big[\operatorname{mat}(\mathcal{Q})^{-1}\operatorname{mat}(\mathcal{A})\Big(\sum_{t=1}^{N-1}E\big(\operatorname{vec}(\mathcal{Z}_t)\operatorname{vec}(\mathcal{Z}_t)^T\big)\operatorname{mat}(\mathcal{A})^T - 2\sum_{t=1}^{N-1}E\big(\operatorname{vec}(\mathcal{Z}_t)\operatorname{vec}(\mathcal{Z}_{t+1})^T\big)\Big)\Big] + \text{constant}.$$

Now, we can write the expected, complete log-likelihood function as follows

$$l(\mathbf{e}) = \operatorname{tr}\{\Gamma\operatorname{mat}(\mathcal{A})[\Phi\operatorname{mat}(\mathcal{A})^T - 2\Psi^T]\} + \text{constant}, \tag{4.22}$$

where constant term includes all terms that do not depend on $\mathcal{A}$, $\Gamma = \operatorname{mat}(\mathcal{Q})^{-1}$, $\Phi = \sum_{t=1}^{N-1}E(\operatorname{vec}\mathcal{Z}_t\operatorname{vec}\mathcal{Z}_t^T)$, and $\Psi = \sum_{t=1}^{N}E(\operatorname{vec}\mathcal{Z}_{t+1}\operatorname{vec}\mathcal{Z}_t^T)$.

4.8 SIMULATION RESULTS

In this simulation study, we compare the multilinear dynamical system (MLDS) approach with the linear dynamical system (LDS) approach for modeling a tensor time series data. The main difference between these two models is that in MLDS approach the transition and projection tensors are factorizable and the noises are isotropic, i.e., the noises are proportionate to the corresponding identity matrix compared to LDS approach. The mean of the initial latent tensor, i.e., $\mathcal{U}_0$ in the MLDS model is taken from the standard normal distribution. Identity matrices are considered for the unfoldings of the covariance tensors and the columns of each transition and projection matrices are considered to be the first J_m eigenvectors of singular-value-decomposed matrices that their entries taken from the standard normal distribution. We initialize the parameters of the LDS model by setting M = 1 and following the same procedure.

For comparison, we measure the prediction error and convergence in likelihood for a synthetic data set obtained with both MLDS and LDS approches. The prediction error is defined $\epsilon_t^M = \|\mathcal{X}_t - \mathcal{X}_t^M\|/\|\mathcal{X}_t\|$ which is the corresponding Euclidean distance of a given model M for the tensor element $\mathcal{X}_t$ of the tensor time series. Note $||\cdot||$ defined in the Section 2.1.2. The estimated tensor $\mathcal{X}_t^M$ is given by $\mathcal{X}_t^M = \text{vec}_I^{-1}\left(\text{mat}(\mathcal{C}^M)\,\text{mat}(\mathcal{A}^M)^t\,\text{vec}(E(\mathcal{Z}_{N_{train}}^M))\right)$ for $t = 1, \ldots, N$, where $E(\mathcal{Z}_{N_{train}}^M)$ is the estimate of the latent tensor in the last element of the training series. The convergence criterion of the likelihood function is specified by observing the marginal likelihood when the number of EM iterations increases. The process is terminated when the difference between two consecutive log-likelihoods of model is less than $\epsilon = 0.001$. We generate the synthetic dataset for the MLDS case with observed and latent variables with dimensions I = (6, 10), J = (2, 4), respectively. A sequence of N = 1100 tensors of the dimensions are generated. Also, we consider the latent dimensionality of the MLDS as J = (2, 4) to obtain the prediction error and convergence results. For each model, we have a training set and a testing set. In this study, the training set contains 1000 tensor observations and the testing set contains the last 100 tensor observations of the tensor time series. Here, the true model represents the exact model that we generated the synthetic data. The results of estimations are represented in Figure 4.2, and Figure 4.3.

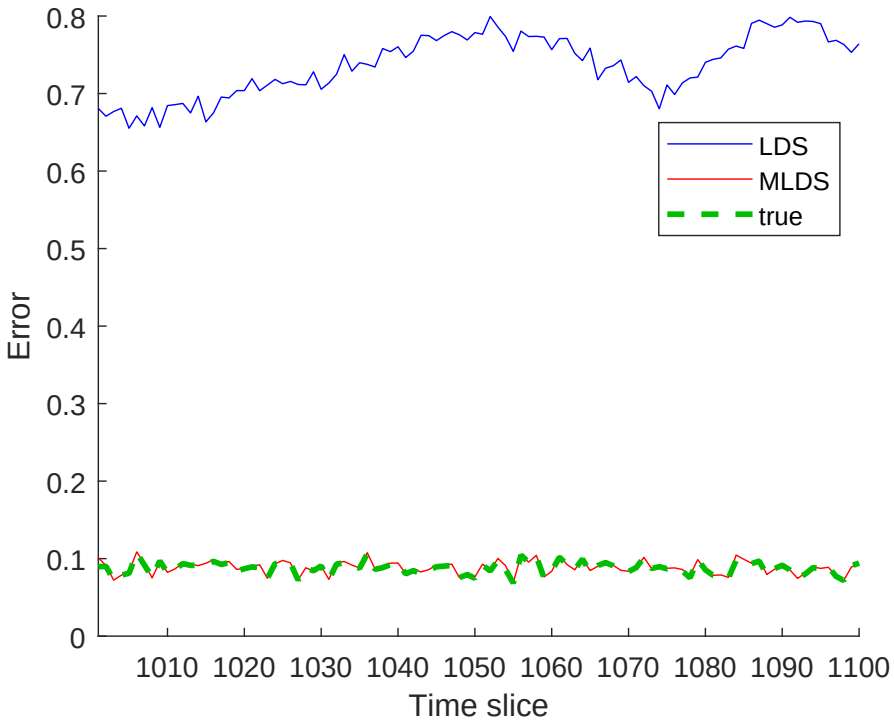

Figure 4.2: Prediction error

According to Figure 4.2, we can see that the prediction error of the MLDS model matches with the prediction error of the true model where as the prediction error of the LDS model is very high and far away from that of the true model. Obviously, since the errors are small in the MLDS model, it achieves a better fit for tensor data than the LDS model.

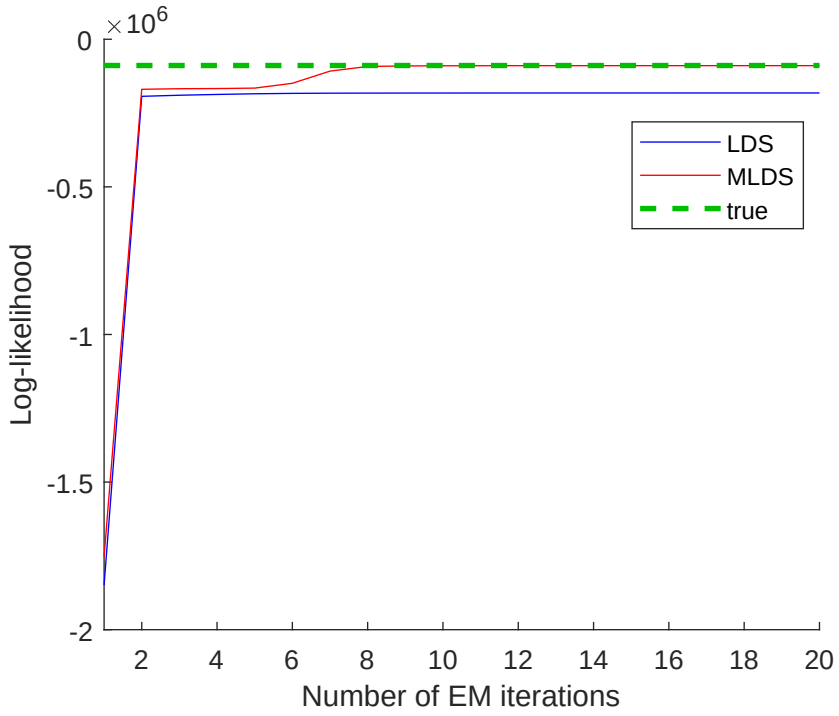

Figure 4.3: Convergence in likelihood

According to Figure 4.3, we can see that MLDS model converges to the likelihood of the true model. Also, it is greater than the convergency of LDS model. Here, the number of EM iterations are approximately 8 when the MLDS converges to the true model.

REFERENCES

[1] Bishop, C.M. (2006), Pattern Recognition and Machine Learning, Springer.

[2] Dempster, A.P., Laird, N.M., Rubin, D.B. (1977), Maximum Likelihood from Incomplete Data via the EM Algorithm. *Journal of the Royal Statistical Society*, Series B. 39 (1): 1-38.

[3] Ding, W., Liu, K.,Belyaev, E., and Cheng, F. (2017), Tensor-based linear dynamical systems for action recognition from 3D skeletons, *Pattern Recognition*, pp. 75-86.

[4] Frank, I. E., and Friedman, J. H. (1993), A Statistical View of Some Chemometrics Regression Tools,*Technometrics*, 35, 109-135.

[5] Hitchcock, F. L. (1927), The expression of a tensor or a polyadic as a sum of products, *Journal of Mathematics and Physics*, 6(1) :164-189.

[6] Hitchcock, F. L. (1928), Multiple invariants and generalized rank of a P-way matrix or tensor, *Journal of Mathematics and Physics*, 7(1) :39-79.

[7] Hou, M., Tensor-based regression models and applications, PhD thesis.

[8] Kalman, R.E. (1960), A new approach to linear filtering and prediction problems, *Journal of Basic Engineering*, 82:35-45.

[9] Kolda, T. G. (2006), Multilinear Operators for Higher-Order Decompositions, *Sandia National Laboratories*, Technical Report SAND2006-2081.

[10] Kolda, T. G., and Bader, B. W. (2009), Tensor Decompositions and Applications, *SIAM Review*, 51, 455-500.

[11] Kolda, T. G., and Bader, B. W., MATLAB Tensor Toolbox Version 2.6, Available online, 2015.

[12] McCullagh, P., and Nelder, J. A. (1983), Generalized Linear Models, *London: Chapman and Hall*.

[13] Rauch, H. (1963), Solutions to the linear smoothing problem, *IEEE Transactions on Automatic Control*, 8:371-372.

[14] Rogers, M., Li, L., Russell, S.J. (2013), Multilinear dynamical systems for tensor time series, *Advances in Neural Information Processing Systems (NIPS)*, pp. 2634-2642.

[15] Tibshirani, R. (1996), Regression Shrinkage and Selection Via the Lasso, *Journal of the Royal Statistical Society*, series B, 58, 267-288.

[16] Tucker, L. R. (1963), Implications of factor analysis of three-way matrices for measurement of change, in Problems in Measuring Change, C. W. Harris, ed., *University of Wisconsin Press*, pp. 122-137.

[17] 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.

[18] Zhou, H., Matlab TensorReg Toolbox Version 1.0, Available online, 2017.

[19] Zitao, L., and Hauskrecht, M. (2016), Learning Linear Dynamical Systems from Multivariate Time Series: A Matrix Factorization Based Framework, *SIAM International Conference on Data Mining*, pp. 810-818.

[20] Zou, H., and Hastie, T. (2005), Regularization and Variable Selection via the Elastic Net, *Journal of the Royal Statistical Society*, Series B, 67, 301-320.

APPENDICES

APPENDIX I

Lemma 2.1: Let $\mathcal{C} \in \mathbb{R}^{I_1 \times I_2 \times \ldots I_D \times J_1 \times J_2 \times \cdots \times J_D}$ and $\mathcal{X} \in \mathbb{R}^{I_1 \times I_2 \times \ldots I_D}$ then,

$$\text{vec}(\mathcal{C} \circledast \mathcal{X}) = \text{mat}(\mathcal{C}) \, \text{vec}(\mathcal{X}).$$

Moreover, if $\mathcal{C}$ can be factorized with matrices $\mathcal{C}^{(d)}$, then $\text{vec}(\mathcal{C} \circledast \mathcal{X}) = [\mathbf{C}^{(D)} \otimes \cdots \otimes \mathbf{C}^{(1)}] \, \text{vec}(\mathcal{X})$.

Proof: Suppose $k = 1 + \sum_{d=1}^{D} \prod_{d'=1}^{d-1} I_{d'}(i_d - 1)$ and $l = 1 + \sum_{d=1}^{D} \prod_{d'=1}^{d-1} J_{d'}(j_d - 1)$ for some $(j_1, \ldots, j_D)$, then we have

$$\text{vec}(\mathcal{C} \circledast \mathcal{X})_k = \sum_{j_1 \ldots j_D} c_{i_1 \ldots i_D j_1 \ldots j_D} x_{j_1 \ldots j_D} = \sum_{l} \text{mat}\,(\mathcal{C})_{kl} \, \text{vec}(\mathcal{X})_l = (\text{mat}(\mathcal{C}) \, \text{vec}\,(\mathcal{X}))_k.$$

Lemma 2.2: Suppose a tensor $\mathcal{B} \in \mathbb{R}^{I_1 \times \cdots \times I_D}$ can be decomposed in to rank R, then

$$\begin{aligned} \mathcal{B}_{(d)} &= \mathcal{B}_d(\mathcal{B}_D \odot \cdots \odot \mathcal{B}_{d+1} \odot \mathcal{B}_{d-1} \odot \cdots \odot \mathcal{B}_1)^T \text{and} \\ \text{vec}(\mathcal{B}) &= (\mathcal{B}_D \odot \cdots \odot \mathcal{B}_1)\mathbf{1}_R, \end{aligned}$$

where $\mathcal{B}_d = [\boldsymbol{\beta}_d^{(1)}, \ldots, \boldsymbol{\beta}_d^{(R)}]$ for $d = 1, \ldots, D$ and $\mathbf{1}_R$ is the vector of ones of size R and $\odot$ is the Khatri-Rao product.

Proof: For the first part of the lemma we can check that the mode-d matricization of $\boldsymbol{\beta}_1 \circ \cdots \circ \boldsymbol{\beta}_D$ is in the elementwise form $\boldsymbol{\beta}_d(\boldsymbol{\beta}_D \otimes \cdots \otimes \boldsymbol{\beta}_{d+1} \otimes \boldsymbol{\beta}_{d-1} \otimes \cdots \otimes \boldsymbol{\beta}_1)^T$. The scalar product $\prod_{d' \neq d} \boldsymbol{\beta}_{d' i_{d'}}$ appears as the k-th element of the row vector $(\boldsymbol{\beta}_D \otimes \cdots \otimes \boldsymbol{\beta}_{d+1} \otimes \boldsymbol{\beta}_{d-1} \otimes \cdots \otimes \boldsymbol{\beta}_1)^T$ where $k = 1 + \sum_{d' \neq d}(i_{d'} - 1) \prod_{d'' < d', d'' \neq d} I_{d''}$. Also, note that the matricization of a sum of arrays equals sum of their matricizations. Thus, the first part of the lemma holds.

For the second part of the lemma,

$$\begin{aligned}
\text{vec}(\mathcal{B}) &= \text{vec}(\sum_{r=1}^{R} \boldsymbol{\beta}_1^{(r)} \circ \cdots \circ \boldsymbol{\beta}_D^{(r)}) \\
&= \sum_{r=1}^{R} \text{vec}(\boldsymbol{\beta}_1^{(r)} \circ \cdots \circ \boldsymbol{\beta}_D^{(r)}) \\
&= \sum_{r=1}^{R} \boldsymbol{\beta}_D^{(r)} \otimes \cdots \otimes \boldsymbol{\beta}_1^{(r)} \\
&= (\mathcal{B}_D \odot \cdots \odot \mathcal{B}_1)\mathbf{1}_R.
\end{aligned}$$

Thus, the second part of the lemma holds.

APPENDIX II

Lemma 4.1: Suppose $I, J \in N^M, M \in N$, and the joint distribution of $\mathcal{X} \in \mathbb{R}^I$ and $\mathcal{Z} \in \mathbb{R}^J$ is given by equation (4.11). Then, the marginal distribution of $\mathcal{X}$ is given by

$$\mathcal{X} \sim \mathcal{N}(\mathcal{C} \circledast \mathcal{U}, \mathcal{C} \circledast \mathcal{G} \circledast \mathcal{C}^T + \mathcal{S}),$$

where $\mathcal{C}^T \in \mathbb{R}^{JI}$ is the transpose of $\mathcal{C}$. There means, $c^T_{j_1 \ldots j_M i_1 \ldots i_M} = c_{i_1 \ldots i_M j_1 \ldots j_M}$. Moreover, the conditional distribution of $\mathcal{Z}|\mathcal{X}$ is

$$\mathcal{Z}|\mathcal{X} \sim \mathcal{N}(\hat{\mathcal{U}}, \hat{\mathcal{G}}),$$

where $\operatorname{vec}(\hat{\mathcal{U}}) = (\mu + W(\operatorname{vec}(\mathcal{X}) - \operatorname{mat}(C)\mu))$, $\operatorname{mat}\left(\hat{\mathcal{G}}\right) = (\Omega - W \operatorname{mat}(C)\Omega)$, $\mu = \operatorname{vec}(\mathcal{U})$, $\Omega = \operatorname{mat}(\mathcal{G})$, $\Sigma = \operatorname{mat}(\mathcal{S})$, and $W = \Omega \operatorname{mat}(\mathcal{C})^T [\operatorname{mat}(\mathcal{C}) \Omega \operatorname{mat}(\mathcal{C})^T + \Sigma]^{-1}$.

Proof: We have $\operatorname{vec}(\mathcal{Z}) \sim \mathcal{N}(\mu, \Gamma)$ and $\operatorname{vec}(\mathcal{X})|\operatorname{vec}(\mathcal{Z}) \sim \mathcal{N}(\operatorname{mat}(\mathcal{C}) \operatorname{vec}(\mathcal{Z}), \Sigma)$ by equations (4.6) and (4.7) and lemma 2.1. We can say $\operatorname{vec}(\mathcal{X}) \sim \mathcal{N}(\operatorname{mat}(\mathcal{C}) \operatorname{vec}(\mathcal{U}), \operatorname{mat}(\mathcal{C}) \Gamma \operatorname{mat}(\mathcal{C}^T) + \Sigma)$ and $\operatorname{vec}(\mathcal{Z})|\operatorname{vec}(\mathcal{X}) \sim \mathcal{N} \operatorname{vec}(\hat{\mathcal{U}}), \operatorname{mat}(\hat{\mathcal{G}}))$ by multivariate normal distribution. Also, $\operatorname{mat}(\mathcal{C} \circledast \mathcal{G} \circledast \mathcal{C}^T) = \operatorname{mat}(\mathcal{C}) \Gamma \operatorname{mat}(\mathcal{C})^T$ by the associative property of the operator $\circledast$. Then, applying lemma 2.1 again and we have, $\mathcal{N}(\mathcal{C} \circledast \mathcal{U}, \mathcal{C} \circledast \mathcal{G} \circledast \mathcal{C}^T + \mathcal{S})$ and $\mathcal{Z}|\mathcal{X} \sim \mathcal{N}(\hat{\mathcal{U}}, \hat{\mathcal{G}})$. Thus, the lemma holds.

VITA

Graduate School
Southern Illinois University

Herath Mudiyanselage Wiranthe Bandara Herath

h.herath@siu.edu

University of Peradeniya
Bachelor of Science, Statistics and Operations Research, December 2015

Thesis Title:
Tensor Regression and Tensor Time Series Analyses for High Dimensional Data

Major Professor: Dr. S. Yaser Samadi