LUISS Guido Carli
LIBERA UNIVERSITÀ INTERNAZIONALE DEGLI STUDI SOCIALI

PhD THESIS

PhD Candidate: *Giuseppe Brandi*

Thesis Title:

Decompose et Impera: Tensor methods in high-dimensional data

Keywords:

Tensor methods, Factor analysis, Multilinear regression

PhD in Economics

XXX Cycle

LUISS Guido Carli
Supervisor: Prof. Giuseppe Ragusa

Thesis Defense:

May, 2018

Thesis Committee:

Prof. *Giuseppe Ragusa, LUISS*

Prof. *Mathew Harding, University of California at Irvine*

Prof. *Anima Anandkumar, California Institute of Technology*

DISCLAIMER - LIBERATORIA

This PhD thesis by *Name Surname*, defended at LUISS Guido Carli University on *Month Day Year* is submitted in partial fulfillment of the requirements for the degree of Doctor of Philosophy in Economics. May be freely reproduced fully or partially, with citation of the source. This is without prejudice to the rights of LUISS Guido Carli University to reproduction for research and teaching purposes, with citation of the source.

Questa tesi di Dottorato di *Nome Cognome*, discussa presso l'Università LUISS Guido Carli in data *Giorno Mese Anno*, viene consegnata come parziale adempimento per l'ottenimento del titolo di Dottore di Ricerca in Economia. Liberamente riproducibile in tutto o in parte, con citazione della fonte. Sono comunque fatti salvi i diritti dell'Università LUISS Guido Carli di riproduzione per scopi di ricerca e didattica, con citazione della fonte.

"As humans, we are all explorers. Some explore new places, others explore old places. I explore ***data****."*

To Ilaria.

"Look at the stars, look how they shine for you"

Acknowledgements

Many people contributed in a way or another to this thesis, and I am grateful to all of them. First of all, I have to thank my supervisor, Giuseppe Ragusa who gave my the freedom to choose the topics of my interest. He is one of the smartest person I had the pleasure to meet. I was, and I am, extremely proud to be recognized in this faculty as one of the "Ragusa's boys".

I have to thank Anima Anandkumar and Matthew Harding, which are the ones who introduced me to the world of tensor methods. I am grateful to them for the passion they instilled in me about tensors.

I would like to thank participants to my PhD seminars and talks at international conferences. All their suggestions had a great impact on the drawing up of this work.

I want to thank Siria, the one I consider my academic (and not only) sister. She is of crucial inspiration to me both as a researcher and teacher. The passion she puts in her work is pervasive.

I have to thank Chiara. I wouldn't be what I am today if I didn't meet her. She infused in me her passion for knowledge and hard work. I will always remember her contribution on my personal and academic path.

I am in debt with Daniela. We started this research path together with all the fears and enthusiasm it brought with it. Her support during periods of frustration were a relief. Her faith in my capabilities made me think I was a good PhD student. I am, and I will always be, grateful for what she gave me, both as a researcher and as a person.

I feel I have to thank my students. They pushed me to go always forward, to the next step. In particular, I want to thank Diletta and Pierluigi, which became good friends of mine.

I want to thank my family. My mother is always with me. She is the most inspirational

person I have ever encounter in my life. She grew up four children alone and I cannot be more grateful to her for what she did for me and my brothers. She is my superhero. My brother were always supportive and they always made me proud of what I am and what I do.
I want to thank my friends which tolerate me and love me even if I denied too often their requests for Saturday's night outs because of the PhD. I want to specially thank Chiara, she is a very good friend. She is smart and deep. I learned a lot from her.

A special mention must be given to my better half, Ilaria. She is the colour in my life of greys. I love her and I am grateful for every single moment we pass together. She is my present and my future. She is the most beautiful thing about me. She embodies everything I search for in a partner and most of all, she tolerates my temper. She helped me a lot in the drafting process of this thesis and she never felt the weight in doing this. Nevertheless, she is always supportive and as a matter of fact, she loves me back. Thank you babe.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

"Begin at the beginning," the King said gravely, "and go on till you come to the end: then stop."

— Lewis Carroll, *Alice in Wonderland*

Figure 1.1 represents Escher's *Man with the cuboid*, a graphical metaphor of a tensor methods researcher.

Figure 1.1: Man with the cuboid

This thesis is written with the scope of exploring multiway data. Multiway data, also referred to as tensor data, is a collection of datapoints in multidimensional matrices. At a first glance one may think that these objects are only a convenient representation of a datasets. They are not *just* a collection of data, they have their own structure. For

this reason, multiway data need specific models to be correctly analysed. In this spirit, I developed my personal idea on data analysis which can be represented by following statement:

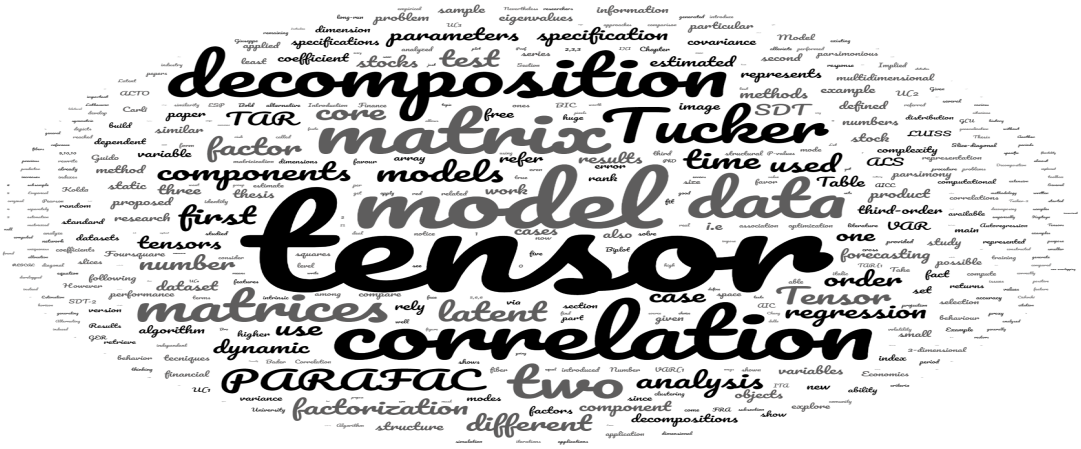
"It is not the data that should fit models, but models that should fit the data"

However, this should not be taken literary I do think that models are important: giving a structure to our techniques is necessary. Nevertheless, I do think that data should be the main driver.This means that instead of trimming data at our necessity to fit existing models, researchers should develop new models to reflect the complexity of the data.
The purpose of this work is to provide an overview of tensor methods applied to Economics and Finance. Yet, the most important aspect of this thesis are ideas and applications rather than the mathematical content. New models are proposed and fitted to data in order to test their performance and get insights from the datasets analysed.
The description of the tensor methods provided in this thesis is not intended to be complete but rather restricted to the model applicable to the analysed data.

1.1 Content of the Thesis

As already stated, the main objective of this thesis is to overview existing tensor methods and develop new ones. In particular, the aim is to contribute in two ways. The first, is to link the world of tensor and related models to Economics and Finance, topic not yet well established in the two research communities. The second, more related to the development of new techniques, is to introduce two new tensor methods, namely Tensor Autoregression and the Slice-diagonal tensor Decomposition (SDT). Even if these methods are applied to economic data, their applicability remains general.
Figure 1.2 depicts the words cloud of this thesis. It is not surprising to find as main words tensor, model, regression, data, matrices Tucker or PARAFAC since are the main topics of the whole work. In particular, Tucker and PARAFAC, are the most used tensor factorization methods and have proved to be powerful techniques in analysing multiway data. I will heavily rely on Tucker factorization in the development of the Tensor Autoregression model. Furthermore, I will use both the Tucker and PARAFAC decompositions to test the performance of the proposed Slice-diagonal Tensor decomposition technique

Figure 1.2: Cloud of words of the thesis



1.2 How to read the Thesis

This thesis is intended to be a starting point for researchers who want to have an overview of some Tensor methods applied to Economics and Finance. In the remaining part of this introduction, I will explain the history and future directions of the tensor data analysis. Chapter 2 is devoted to the Tensor Autoregressive model while Chapter 3 to the analysis of correlation and covariance matrices over time with the help of the Slice-Diagonal Tensor decomposition. A brief summary of the two chapter is provided below:

Chapter 2: Tensor Autoregression in Economics and Finance

Multidimensional data (tensor data) is a relevant topic in statistical and machine learning research. Given their complexity, such data objects are usually reshaped into matrices and then analysed. However, this methodology presents other drawdowns. First of all, it destroys the intrinsic interconnections among datapoints in the multidimensional space. Secondly, the number of parameters to be estimated in a model increases exponentially. To alleviate these issues, we treat the data as it is and we develop a model able to deal with the multidimensionality of the dataset. In particular, a parsimonious tensor regression (in which both the regressor and the response are tensors) is build such that it retains the intrinsic multidimensional structure of the dataset. Tucker decomposition is employed to achieve parsimony and an ALS algorithm is developed to estimate the model parameters. A simulation exercise is produced to validate the model. In order to

compare it with the existing model in a forecasting setting, an empirical application to Foursquares spatio-temporal dataset and macroeconomic time series is performed.

Chapter 3: Latent stock correlation projection via tensor decomposition

Correlation matrices are ubiquitous in financial statistics both in research papers and in the industry. The correlation is the most used method of association between stock returns. Portfolio allocation, risk management and network analysis are based on correlation matrices. Applied researchers debate about the correct correlation matrix to use. In this regard, one of the issues is related to the choice of the sample period used in the reference models. This is because the true correlation is not available. Several approaches exist in order to handle this problem; yet, all of them are heavily dependent on the time horizon used. In this paper, we introduce a method which aims to alleviate the problem of true correlation availability by decomposing the time series of correlations into a static and a dynamic component. The static component encapsulates the latent, long-run behaviour of the correlations while the dynamic part represents the time dependent structure in the latent space. Hence, the latent correlation matrix projected by the tensor factorization can be plugged in the models as an alternative to the standard correlation matrix. We will show that the hidden correlation is empirically almost time invariant.

1.3 Where it started, where we are and where we are going

Multiway analysis is a relatively old concept. However, the difficulties in understanding models and datasets put aside this research field in favour of vector and matrix analysis. The common practice was (is) to reshape multidimensional data in matrices and then the analysis is performed on these objects.

1.3.1 Where it started

Tucker (1964, 1966) was one of the first who developed models to deal with the inherent structure of the data. He theorized one of the decomposition techniques nowadays still at the forefront of the field. Since his pioneering work in psychometrics, several papers appeared in the chemometrics and psychometric community. Carrol and Chang (1970) and Hashman (1970) developed the first PARAFAC model in order to study patients for which the variables of interest were observed at different occasions. Since then, books and papers came out. Among all Kroonenberg (2008), Smilde et al. (2004), Cichocki et al. (2007), as well as the review paper of Kolda and Bader (2009).

1.3.2 Where we are

The increase of datasets' size and dimensionality asks for a new computational paradigm. Tensor-level thinking is now at the forefront of the computational thinking. Several efforts from the tensor research community are leading to the transition from matrix-based to tensor-based computations. Papers on dimensionality reduction are countless. The decomposition can be used to extract structural information from huge noisy data as in the matrix factor analysis case. For an excellent review of the most used decompositions techniques, refer to Kolda and Bader (2009) and references therein.

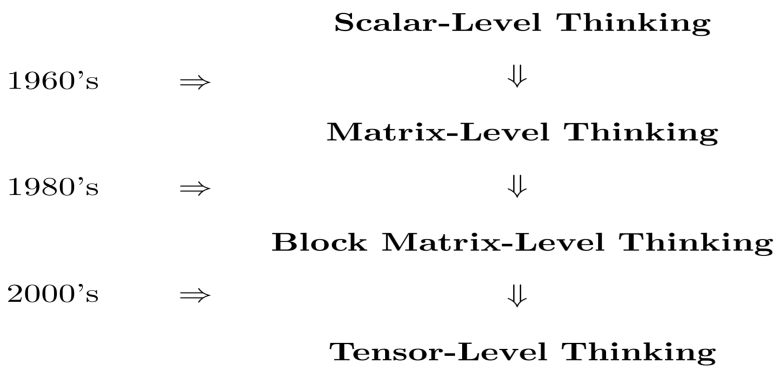
Large dynamic network can be analysed via tensor factorization to retrieve the latent components of the network, studying hyper clustering, communities and anomalies. See Bader et al. (2007), Kolda et al. (2005), Nikel et al. (2011), Kossaifi et al. (2017) and Sun et al. (2006) to have more details about this topic.
Nonnegative factorization has its own strand of literature. It is applied when the analysed data is expressed in units which cannot be negative, such as count data, monetary exchanges or pixels of an image. In this regards refer to Paetero and Tapper (1994), Lee and Seung (1999), Hu et al. (2015) and Welling and Weber (2001). For the use of tensor calculus in latent variable models, refer to Anandkumar et al. (2014) and reference therein. However, the analysis of tensor data needs a computational infrastructure. The community has found in Matlab the software more akin to the scope. Andersson and Bro

(2000) developed the N-way toolbox which collect a huge amount of algorithms on tensor methods. Another well known toolbox is the Tensor toolbox by Bader and Kolda (2007). These, among other toolboxes, are the most complete and used computational tools by the tensor community nowadays.

1.3.3 Where we are going

The future of data analysis is multilinear. Datasets intrinsically multidimensional are already here and will increase in number, dimension and complexity asking for methods always more computationally efficient. Tensors are generalization of matrices, so they encompass all the matrix features. Therefore, in the near future, matrix methods will be incorporated in the more general tensor methods. Figure 1.3[1] shows the history of the computational level thinking, from scalar to tensor. The research community should take the challenge of multidimensional data analysis to foster a new, pervasive, computational paradigm.

Figure 1.3: Computational-level thinking history



The title of the thesis is quite clear. *Decompose et Impera* means the ability to use tensor factorization techniques to solve high dimensional data analysis problems for which no close (and easy) solutions are available.

[1] NSF (2009) on "Future directions in Tensor-based computation and modeling".

Chapter 2

Tensor Autoregression in Economics and Finance

2.1 Introduction

Big Data and multidimensional data are becoming a relevant topic in statistical and machine learning research. In fact, working with such huge and complex objects is difficult for a plethora of reasons. Among all the difficulties we can encounter working with such objects, there are the dimension of the dataset and the parsimony of the model used to analyse them. What is generally done when we encounter a huge dataset is to try to shrink it to a smaller dimension or to use models which reduce the computational burden of such datasets via approximate analysis. The issue is even more pronounced when we deal with multidimensional data. Multidimensional data consists of datasets which are characterized by more than two dimensions. Examples are 3D astronomical images (cube of pixels), panel data (individuals $\times$ variables $\times$ time $\times$ location) or higher order (> 2) multivariate portfolio moments (the Kurtosis tensor is defined as $M_4 = r \otimes r^{'} \otimes r \otimes r^{'}$). Given their intrinsic complexity, such datasets are usually reshaped into matrices or vectors and then the analysis is done on these simpler objects. However, this manipulation of multidimensional datasets creates other issues. First of all, it destroys the intrinsic interconnections between the data points in the multidimensional space. Secondly, the number of parameters to be estimated increases exponentially. To alleviate this problem we propose to work with the data as it is and to build a model able to

deal with the multidimensionality of the dataset. In particular, we build a parsimonious tensor regression which retains the intrinsic multidimensional structure of the dataset. We contribute to the literature of tensor regression in two ways: we develop a tensor regression in which the tensor structure is on both the response and the predictor variables and we build the regression via contracted product proposing an ALS algorithm to estimate the parameters.

The paper is structured as follows. Section 2 and 3 are related respectively to notation and preliminaries on tensors and operation on tensors. Section 4 introduces the tensor regression in the general case in which both the dependent and the independent variables are tensors. Section 5 proposes an ALS algorithm for the coefficients estimation. Section 6 is devoted to the simulation study and Section 7 to the empirical application with the results. Section 8 concludes.

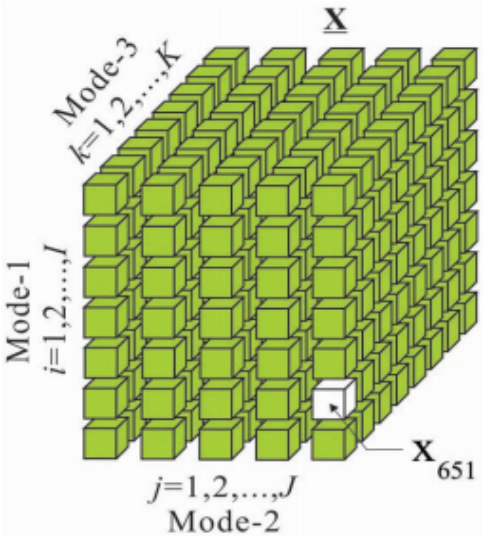
2.2 Notation and preliminaries

Tensors are generalization of scalars, vectors and matrices and their *order* defines them. The order of a tensor, which is the number dimensions that characterises it, is also referred to as ways or modes. Scalars are 0th order tensors, vectors are 1st order tensors and matrices are 2nd order tensors. Whatever has more than two dimensions is referred to as higher order tensor or just as tensor. Throughout this work the notation will follow the standard convention: a is a scalar, $\mathbf{a}$ is a vector, $\mathbf{A}$ is a matrix and $\mathscr{A}$ is a tensor. A tensor can be represented in several way. We will use the standard representation used in Kolda and Bader(2009). Figure 2.1[1] is an illustration of a third order tensor with dimensions 7,5 and 8.

As in matrix algebra, tensor can be evaluated in terms of their subparts, which in the matrix case are column and row vectors, while for tensor are called *subarrays*. Subarrays are defined as the result of fixing a subset of indexes of the tensor. Fixing all the but one index we have *fibers*. These objects are the higher order version of columns and rows of

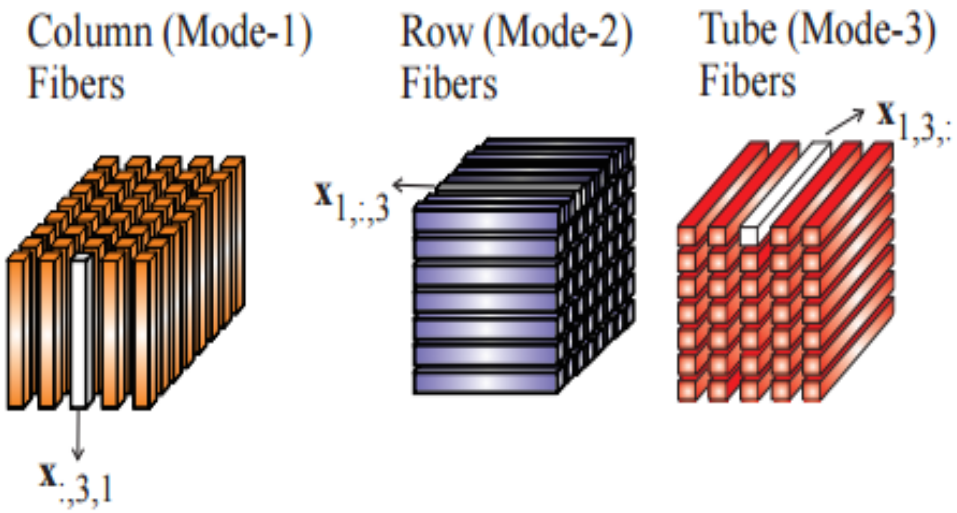
[1] Cichocki et al., 2009

Figure 2.1: Example of a third-order tensor



a matrix. For a third order tensor $\mathscr{Y}$ we have that $y_{:,j,k}$ is a column fiber, $y_{i,:,k}$ is a row fiber while $y_{i,j,:}$ is a tube fiber. Figure 2.2[2] give a graphical explanation of these objects.

Figure 2.2: Example of a third-order tensor slices modes and fibers

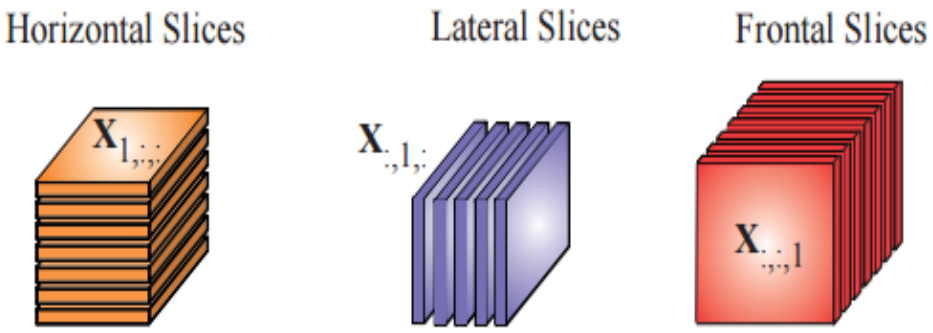


Instead, by fixing all but two indexes, we get what are called *slices*. Slices are two-dimensional objects that are a section of the tensor. For a third order tensor $\mathscr{Y}$ we have that $Y_{i,:,:}$ are $i = 1, 2, ..., I$ horizontal slices,$Y_{:,j,:}$ are $j = 1, 2, ..., J$ horizontal slices while

[2]Cichocki et al., 2009

$Y_{:,:,k}$ are $k = 1, 2, ..., K$ frontal slices. A graphical representation of slices is provided in Figure 2.3 [3]

Figure 2.3: Example of a third-order tensor slices



2.2.1 Examples

Once provided the definition of a tensor and of its components, we will now provide some examples of such components with real numbers. Take as an example a tensor $\mathscr{Y}$ of size 3x4x2. A mode-1 fiber (**column** fiber) can be given by:

$$Y_{(:,1,1)} = \begin{pmatrix} 1 \\ 2 \\ 3 \end{pmatrix} \quad or \quad Y_{(:,3,2)} = \begin{pmatrix} 19 \\ 20 \\ 21 \end{pmatrix}.$$

A mode-2 fiber (**row** fiber) can be represented as:

$$Y_{(1,:,1)} = \begin{pmatrix} 1 & 4 & 7 & 10 \end{pmatrix} \quad or \quad Y_{(4,:,2)} = \begin{pmatrix} 15 & 18 & 21 & 24 \end{pmatrix}$$

The two **frontal** slices (or modes) are given by:

$$Y_{(:,:,1)} = \begin{pmatrix} 1 & 4 & 7 & 10 \\ 2 & 5 & 8 & 11 \\ 3 & 6 & 9 & 12 \end{pmatrix}$$

[3] Cichocki et al., 2009

$$Y_{(:,:,2)} = \begin{pmatrix} 13 & 16 & 19 & 22 \\ 14 & 17 & 20 & 23 \\ 15 & 18 & 21 & 24 \end{pmatrix}$$

We now define some operations defined on tensors.

2.3 Operations on tensors

Kolda(2006), Bader and Kolda (2004, 2009), De Lathauwer et al. (2000) Anandkumar et al. (2014) and Liu and Trenkler (2008) are just few of the papers focused on tensor operations. They explain the various calculations involving tensors and matrices both formally and with examples. There are different operations which can be performed in tensor analysis but here we will present just the ones inherent to this work.

2.3.1 Matricization

The matricization of a tensor (also called unfolding) is the process of reshaping a tensor into a matrix. Take a tensor $\mathscr{X}$ of size $I_1 \times I_2 \cdots \times I_N$. Let the ordered sets $\mathscr{R} = \{r_1, ..., r_L\}$ and $\mathscr{C} = \{c_1, ..., c_M\}$ be a partitioning of the modes of the tensor. The matricized tensor is defined as:

$X_{\{\mathscr{R}\times\mathscr{C},I_N\}} \in R^{J\times K}$ where $J = \prod_{I_N\in\mathscr{R}} I_N$ and $K = \prod_{I_N\in\mathscr{C}} I_N$

Take the previous tensor $\mathscr{Y}$ of size 3x4x2:

General matricization:

$$Y_{(\{3,1\}\times\{2\}:3\times4\times2)} = \begin{pmatrix} 1 & 4 & 7 & 10 \\ 13 & 16 & 19 & 22 \\ 2 & 5 & 8 & 11 \\ 14 & 17 & 20 & 23 \\ 3 & 6 & 9 & 12 \\ 15 & 18 & 21 & 24 \end{pmatrix}$$

Mode-n matricization:

The ***mode-n matricization*** is a special case in which the set $\mathscr{R}$ is a singleton equal to n, $\mathscr{C} = \{1, ..., n-1, n+1, ..., N\}$ and it is defined as $X_{\{\mathscr{R}\times\mathscr{C},I_N\}} \equiv X_{(n)}$ The fibers of mode n are aligned as the columns of the resulting matrix. This is given by:

$$Y_{(\{1\}\times\{2,3\}:3\times4\times2)} = Y_{(1)} = \begin{pmatrix} 1 & 4 & 7 & 10 & 13 & 16 & 19 & 22 \\ 2 & 5 & 8 & 11 & 14 & 17 & 20 & 23 \\ 3 & 6 & 9 & 12 & 15 & 18 & 21 & 24 \end{pmatrix}$$

2.3.2 N-mode product

The *mode-n product* of the tensor $\mathscr{Y}$ of size $I_1 \times I_2 \cdots \times I_N$ with the matrix $\mathbf{V}$ of size $I_n \times J$ is denoted as

$$\mathscr{Y} \times_n \mathbf{V}$$

and is of size $I_1 \times \cdots \times I_{n-1} \times J \times I_{n+1} \times \cdots \times I_N$.
This can be expressed in terms of matricized tensors as:

$$\mathscr{X} = \mathscr{Y} \times_n \mathbf{V} \quad \Leftrightarrow \quad \mathbf{X}_{(n)} = \mathbf{V}\mathbf{Y}_{(n)}.$$

Take $V = \begin{pmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \end{pmatrix}$ .The product $\mathscr{X} = \mathscr{Y} \times_1 \mathbf{V}$ results in:

$$X_{(:,:,1)} = \begin{pmatrix} 14 & 32 & 50 & 68 \\ 32 & 77 & 122 & 167 \end{pmatrix}$$

$$X_{(:,:,2)} = \begin{pmatrix} 86 & 104 & 122 & 140 \\ 212 & 257 & 302 & 347 \end{pmatrix}$$

2.3.3 Contracted product

The contracted product between tensors can be seen as a matrix product. It is written as:

$$\mathscr{Y} = \langle \mathscr{X}, \mathscr{B} \rangle_{(\mathscr{I}_{\mathscr{X}};\mathscr{I}_{\mathscr{B}})}$$

For example, if we want to compute it over two modes we have:

$$\mathscr{Y} = \langle \mathscr{X}, \mathscr{B} \rangle_{(n,m;k,l)}$$

Where the subscripts are the modes over which the product must be computed. Given this formulation, the n mode of $\mathscr{X}$ must be of the same dimension of the k mode of $\mathscr{B}$ and the same for m and l.

This can be computed (via matricization) as:

$$\mathscr{Y} = X_{(\{n,m\}\times\{\mathscr{C}_{\mathscr{X}}\})} B_{(\{k,l\}\times\{\mathscr{C}_{\mathscr{B}}\})}$$

We will heavily rely on constructed product for the tensor regression model proposed in this work.

2.3.4 Tensor decomposition

Tensors, like matrices, can be decomposed in smaller (in terms of rank) objects. One decomposition method is the Tucker decomposition. The Tucker decomposition was theorized by Tucker (1964, 1966). It represents an extension of the bilinear factor analysis to the multi-way case. It is also referred to as N-mode PCA in Kapteyn et al. (1986) and Higher-order SVD by De Lathauwer et al. (2000). Take a third-order tensor $\mathscr{Y} \in R^{I\times J\times R}$. The decomposition takes the form of a *n-mode* product, i.e.:

$$\mathscr{B} \approx \mathscr{G} \times_1 \mathbf{U}^{(1)} \times_2 \mathbf{U}^{(2)} \cdots \times_N \mathbf{U}^{(N)} = \mathscr{G} \times \{\mathbf{U}^{(n)}\}. \tag{2.1}$$

Kolda (2006) showed that the Tucker decomposition can be rewritten in matricized form as:

$$\tilde{B} \approx (\mathbf{U}^{(r_L)} \otimes \mathbf{U}^{(r_{L-1})} \otimes \cdots \otimes \mathbf{U}^{(r_1)}) \tilde{G} (\mathbf{U}^{(c_M)} \otimes \mathbf{U}^{(c_{M-1})} \otimes \cdots \otimes \mathbf{U}^{(c_1)})^T \tag{2.2}$$

Where $\mathscr{R} = \{r_1, ..., r_L\}$ and $\mathscr{C} = \{c_1, ..., c_M\}$ are partitioning sets of the modes of the tensor in rows and columns Figure 2.4 shows the Tucker decomposition for a third-order tensor, which factors the tensor in a small core tensor and three factor matrices.

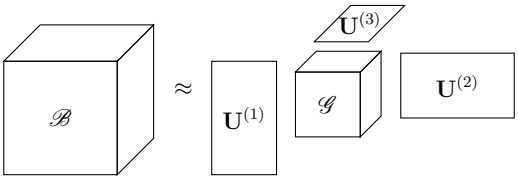


Figure 2.4: Tucker decomposition for $N = 3$.

2.4 Tensor regression

Tensor regression can be formulated in different ways: the tensor structure is on the dependent variable, the independent variable or both. The literature on the first two specifications is vast, including among all Hoff (2015), Zhao et al. (2011) and Yu et al. (2015). In this work we rely on the latter. For the regression model proposed and analysed in this work, we make use of the contracted product operator and the model can be expressed as:

$$\mathscr{Y} = \langle \mathscr{X}, \mathscr{B} \rangle_{(\mathscr{I}_{\mathscr{X}};\mathscr{I}_{\mathscr{B}})} + \mathscr{E} \tag{2.3}$$

More specifically, we will use as predictor tensor a lagged version of the response tensor, where both the tensor are three dimensional.[4] In particular, we rely on the the following specification:

$$\mathscr{Y} = \langle \mathscr{X}, \mathscr{B} \rangle_{(2,3;1,3)} + \mathscr{E}$$

where:

- $\mathscr{Y} \in R^{T \times W \times Q}$ is the response tensor (dependent variable).
- $\mathscr{X} \in R^{T \times V \times P}$ is the predictor tensor (independent variable).
- $\mathscr{B} \in R^{V \times W \times P \times Q}$ is the tensor of coefficients.
- $\mathscr{E} \in R^{T \times W \times Q}$ is the error term.

Matricizing the tensors we can rewrite them as:

- $\tilde{Y} = \mathscr{Y}_{(\{2,3\} \times \{1\}:(T,W,Q))}$

[4] In the following we will rely on the three dimensional case, but the model can be applied to every tensor dimension.

- $\tilde{X} = \mathscr{X}_{(\{2,3\}\times\{1\}:(T,V,P))}$

- $\tilde{B} = \mathscr{B}_{(\{1,3\}\times\{2,4\}:(V,W,P,Q))}$

- $\tilde{E} = \mathscr{E}_{(\{2,3\}\times\{1\}:(T,W,Q))}$

The regression takes the form of a multivariate regression, i.e.:

$$\tilde{Y} = \tilde{X}\tilde{B} + \tilde{E} \tag{2.4}$$

For which the LS solution (given sufficient data) of $\tilde{B}$ is:

$$\tilde{B} = (\tilde{X}^T\tilde{X})^{-1}\tilde{X}^T\tilde{Y} \tag{2.5}$$

However,the $\mathscr{B}$ coefficient is high dimensional so, to improve parsimony, we impose a Tucker structure on $\mathscr{B}$ such that it is possible to recover the original $\mathscr{B}$ with smaller objects, i.e.:

$$\mathscr{B} \approx \mathscr{G} \times_1 \mathbf{U}^{(1)} \times_2 \mathbf{U}^{(2)} \times_3 \mathbf{U}^{(3)} \times_4 \mathbf{U}^{(4)} \tag{2.6}$$

Using matricization of the Tucker decomposition defined in Kolda (2006), it is possible to rewrite the $\mathscr{B}$ tensor as:

$$\tilde{B} \approx (\mathbf{U}^{(3)} \otimes \mathbf{U}^{(1)})\tilde{G}(\mathbf{U}^{(4)} \otimes \mathbf{U}^{(2)})^T \tag{2.7}$$

Where $\tilde{G} = \mathscr{G}_{(\{1,3\}\times\{2,4\})}$.

The regression model can be then rewritten as:

$$\tilde{Y} = \tilde{X}(\mathbf{U}^{(3)} \otimes \mathbf{U}^{(1)})\tilde{G}(\mathbf{U}^{(4)} \otimes \mathbf{U}^{(2)})^T + \tilde{E}$$

This specification can be seen as a reduced rank regression with multilinear interactions. We now introduce the estimation procedure.

2.5 Estimation

In this section, we develop an alternating least squares (ALS) algorithm to estimate the coefficient tensor. The ALS was introduced for the Tucker decomposition by Kroonenberg and De Leeuw (1980) and Kapteyn et al. (1986). This methodology solves the optimization problem by dividing it into small least squares problems. This allows to solve the optimization for each factor separately, keeping constant the other ones at the previous iteration value. Then, it iterates alternatively among all the factors until the algorithm converges (the error ε reach a threshold α) or a maximum number of iterations is reached.

Expressing the regression error as:

$$E = \tilde{Y} - \tilde{X}(\mathbf{U}^{(3)} \otimes \mathbf{U}^{(1)})\tilde{G}(\mathbf{U}^{(4)} \otimes \mathbf{U}^{(2)})^T$$

The objective is to find the $\widehat{\mathbf{U}}^{(n)}$ such that

$$\frac{\partial \|E\|^2}{\partial \mathbf{U}^{(n)}} = \mathbf{0}$$

and than the core tensor is the computed as

$$\widehat{\mathscr{G}} \approx \widehat{\mathscr{B}} \times_1 \widehat{\mathbf{U}}^{(1)} \times_2 \widehat{\mathbf{U}}^{(2)} \times_3 \widehat{\mathbf{U}}^{(3)} \times_4 \widehat{\mathbf{U}}^{(4)}$$

with the founded $\widehat{\mathbf{U}}^{(n)}$.

This is a nonlinear optimization problem which can be solved by iterative algorithms such as ALS.

The partial least square solutions over each component $\mathbf{U}^{(n)}$ and G for the general case of $N - dimensional$ coefficient tensor are retrieved by Algorithm 1.[5]

[5]In the tensor regression model proposed $N = 4$.

Algorithm 1: Alternating least squares

1: **Initialize the algorithm to some $\mathbf{U}_0^{(n)}$ and G_0.**
2: **repeat**
3: $\quad \mathbf{U}_i^{(1)} = f(\mathbf{U}_{i-1}^{(2)}, \mathbf{U}_{i-1}^{(3)}, ..., \mathbf{U}_{i-1}^{(N)}, G_{i-1}, X, Y)$
4: $\quad \mathbf{U}_i^{(2)} = f(\mathbf{U}_{i-1}^{(1)}, \mathbf{U}_{i-1}^{(3)}, ..., \mathbf{U}_{i}^{(N)}, G_{i-1}, X, Y)$
5: $\quad \vdots$
6: $\quad \mathbf{U}_i^{(N)} = f(\mathbf{U}_{i-1}^{(1)}, \mathbf{U}_{i}^{(2)}, ..., \mathbf{U}_{i}^{(N-1)}, G_{i-1}, X, Y)$
7: $\quad G_i = f(\mathbf{U}_i^{(1)}, \mathbf{U}_i^{(2)}, \mathbf{U}_i^{(3)}, ..., \mathbf{U}_i^{(N)}, X, Y)$
8: **until** Convergence or Maximum iterations reached.
9: **Return** $\widehat{\mathscr{B}}$

Iterating over each component, taking the others fixed, permits to solve the *"big"* problem as *"small"* least squares problems. Let us consider the following identities:

a) $A \otimes B = (A \otimes I)(I \otimes B)$,

b) $(AB)' = B'A'$,

c) $(A \otimes B)' = A' \otimes B'$,

d) $A^\dagger = (A'A)^{-1}A'$,

e) $A^\dagger B^\dagger = (AB)^\dagger$.

Combining equations 2.5 and 2.7 and using identity (d) we get:

$$(\mathbf{U}^{(3)} \otimes \mathbf{U}^{(1)})\tilde{G}(\mathbf{U}^{(4)} \otimes \mathbf{U}^{(2)})' = \tilde{X}^\dagger \tilde{Y} \tag{2.8}$$

Using identity (a) we can rewrite it as:

$$(\mathbf{U}^{(3)} \otimes I)(I \otimes \mathbf{U}^{(1)})\tilde{G}[(\mathbf{U}^{(4)} \otimes I)(I \otimes \mathbf{U}^{(2)})]' = \tilde{X}^\dagger \tilde{Y} \tag{2.9}$$

Applying identity (b)[6] to equation 2.9, we get:

$$(\mathbf{U}^{(3)} \otimes I)(I \otimes \mathbf{U}^{(1)})\tilde{G}(I \otimes \mathbf{U}^{(2)})'(\mathbf{U}^{(4)} \otimes I)' = \tilde{X}^{\dagger}\tilde{Y} \tag{2.10}$$

For the sake of readability, let's write $(\mathbf{U}^{(n)} \otimes I) = \tilde{\mathbf{U}}^{(n)}$ so that we can reformulate equation 2.10 as:

$$\tilde{\mathbf{U}}^{(3)}\tilde{\mathbf{U}}^{(1)}\tilde{G}\tilde{\mathbf{U}}^{(2)\prime}\tilde{\mathbf{U}}^{(4)\prime} = \tilde{X}^{\dagger}\tilde{Y} \tag{2.11}$$

Now, we have to solve the equation for every $\tilde{\mathbf{U}}^{(n)}$ and $\tilde{G}$. Using identity (d) and the rules of matrix algebra (Magnus & Neudecker (1988)) we can retrieve the following solutions (from the most left to the most right factor):

$$\begin{aligned}
\tilde{\mathbf{U}}^{(3)} &= (\tilde{X}\tilde{\mathbf{U}}^{(4)}\tilde{\mathbf{U}}^{(2)}\tilde{G}\tilde{\mathbf{U}}^{(1)\prime})^{\dagger}\tilde{Y} \\
\tilde{\mathbf{U}}^{(1)\prime} &= (\tilde{X}\tilde{\mathbf{U}}^{(4)}\tilde{\mathbf{U}}^{(2)}\tilde{G})^{\dagger}\tilde{Y}(\tilde{\mathbf{U}}^{(3)\prime})^{\dagger} \\
\tilde{G} &= (\tilde{X}\tilde{\mathbf{U}}^{(4)}\tilde{\mathbf{U}}^{(2)})^{\dagger}\tilde{Y}(\tilde{\mathbf{U}}^{(1)\prime}\tilde{\mathbf{U}}^{(3)\prime})^{\dagger} \\
\tilde{\mathbf{U}}^{(2)\prime} &= (\tilde{X}\tilde{\mathbf{U}}^{(4)})^{\dagger}\tilde{Y}(\tilde{G}\tilde{\mathbf{U}}^{(1)\prime}\tilde{\mathbf{U}}^{(3)\prime})^{\dagger} \\
\tilde{\mathbf{U}}^{(4)} &= \tilde{X}^{\dagger}\tilde{Y}(\tilde{\mathbf{U}}^{(2)}\tilde{G}\tilde{\mathbf{U}}^{(1)\prime}\tilde{\mathbf{U}}^{(3)\prime})^{\dagger}
\end{aligned} \tag{2.12}$$

Algorithm 2 shows he ALS algorithm used to solve equation 2.12 for the tensor regression

Algorithm 2: Alternating least squares for the Tensor regression

1: **Initialize the algorithm to some** $\tilde{\mathbf{U}}_0^{(n)}$ **and** $\tilde{G}_0$**.**
2: **repeat**
3: $\quad \tilde{\mathbf{U}}_i^{(4)} = \tilde{X}^{\dagger}\tilde{Y}(\tilde{\mathbf{U}}_{i-1}^{(2)}\tilde{G_{i-1}}\tilde{\mathbf{U}}_{i-1}^{(1)\prime}\tilde{\mathbf{U}}_{i-1}^{(3)\prime})^{\dagger}$
4: $\quad \tilde{\mathbf{U}}_i^{(3)} = (\tilde{X}\tilde{\mathbf{U}}_i^{(4)}\tilde{\mathbf{U}}_{i-1}^{(2)}\tilde{G}_{i-1}\tilde{\mathbf{U}}_{i-1}^{(1)\prime})^{\dagger}\tilde{Y}$
5: $\quad \tilde{\mathbf{U}}_i^{(2)\prime} = (\tilde{X}\tilde{\mathbf{U}}_i^{(4)})^{\dagger}\tilde{Y}(\tilde{G}_{i-1}\tilde{\mathbf{U}}_{i-1}^{(1)\prime}\tilde{\mathbf{U}}_i^{(3)\prime})^{\dagger}$
6: $\quad \tilde{\mathbf{U}}_i^{(1)\prime} = (\tilde{X}(\tilde{\mathbf{U}}_i^{(4)}\tilde{\mathbf{U}}_i^{(2)})\tilde{G}_{i-1})^{\dagger}\tilde{Y}(\tilde{\mathbf{U}}_i^{(3)\prime})^{\dagger}$
7: $\quad \tilde{G}_i = (\tilde{X}\tilde{\mathbf{U}}_i^{(4)}\tilde{\mathbf{U}}_i^{(2)})^{\dagger}\tilde{Y}(\tilde{\mathbf{U}}_i^{(1)\prime}\tilde{\mathbf{U}}_i^{(3)\prime})^{\dagger}$
8: **until** Convergence or Maximum iterations reached.
9: **Return** $\widehat{\mathscr{B}}$

The ALS has several applications and it is workhorse algorithm for Tensor factoriza-

[6] Alternatively, we can use identity (c) on equation 2.8 and then identity (a) leads to the same numerical result.

tion. In the following section, we will test the performances of this procedure in tensor regression with simulated data.

2.6 Simulation study

In this section, we test the reliability of the model and its estimation procedure via simulation study. For this purpose, we implement two small experiments. The first one is the capability of the model to correctly predict the tensor $\mathscr{Y}$. The second experiment relies on the ability of the model to retrieve the real, structural $\mathscr{B}$ coefficient. For the first experiment the data are generated as follows:

- $\mathscr{X} \in R^{600\times30\times30}$ (540000 data points in 3 dimensions) is generated to be $N(0,1)$.
- $\mathscr{B} \in R^{30\times30\times30\times30}$ (810000 parameters) is generated to be a sparse tensor having zeros and ones.
- $\mathscr{E} \in R^{600\times30\times30}$ is generated to be $N(0,1)$.

$\mathscr{Y}$ is generate according to

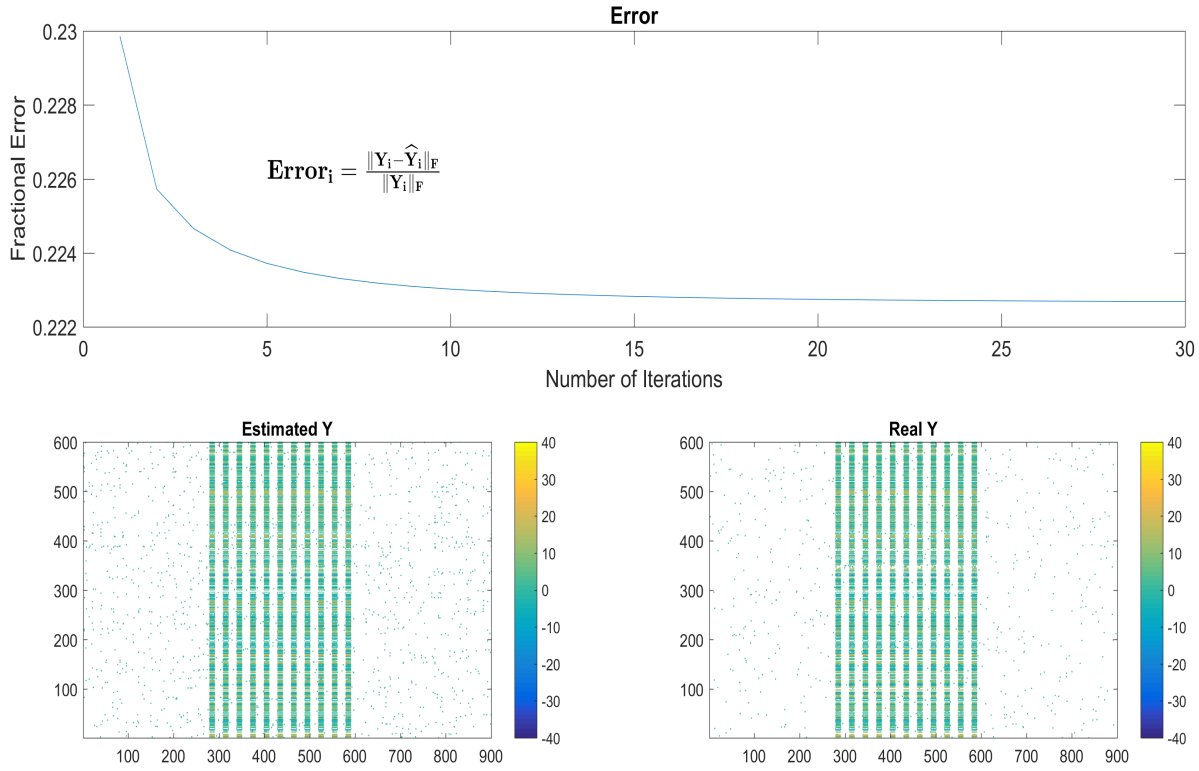
$$\mathscr{Y} = \langle \mathscr{X}, \mathscr{B} \rangle_{(2,3;1,3)} + \mathscr{E} \tag{2.13}$$

For this experiment, the Tucker rank (dimension of the core tensor) of $\mathscr{B}$ is [5 5 5 5] giving rise to 1225 free parameters to be estimated (each sub regression matrix coefficient is 30×5).

As shown by Figure 2.5, the model converges quite fast and the predicted $\hat{\mathscr{Y}}$, here represented as the heatmap of the matricized version, is close to the one of $\mathscr{Y}$, suggesting the good performance of the model.
In the second exercise, we will use a similar example, but in this case we impose the $\mathscr{B}$ to be an image. More specifically, we import the image and tensorize the matrix of pixel, generating a fourth-order tensor $\mathscr{B}$ and we generate $\mathscr{Y}$ as in equation 2.13.[7]

[7]The image is reshaped to a 900×900 pixels matrix and then tensorized as a $30 \times 30 \times 30 \times 30$ array.

Figure 2.5: ALS convergence and prediction accuracy of the Tensor regression



We will use three different images, with increasing level of complexity. The images are shown in Figure 2.6.

The first image represents a cross. It is the easiest image to retrieve since it is symmetric and has straight edges. Furthermore, all the data is clustered in one segment of the figure. The second image represents a face. This picture has an higher level of complexity compared to the cross. Yet, it is not symmetric and has smooth edges. The main difficulty in this case is to retrieve the shapes of the eyes and of the mouth since they are thinner lines. The last figure portrays a body. Indeed, the edges are tiny and the data is not symmetrically distributed. The non-zero coefficients (black pixels) are not accumulated in a portion of the figure, as in the cross or in some parts of the face image (beard, eyes and hair). Yet, they are evenly distributed throughout the image.

In Figure 2.7, we show the results of the regression coefficient for the cross image with five different specifications of the core array. As the figure shows, a very small rank specification imposed on the coefficient tensor $\mathscr{B}$ is enough to retrieve the real image.

Figure 2.6: Images used to generate $\mathscr{Y}$

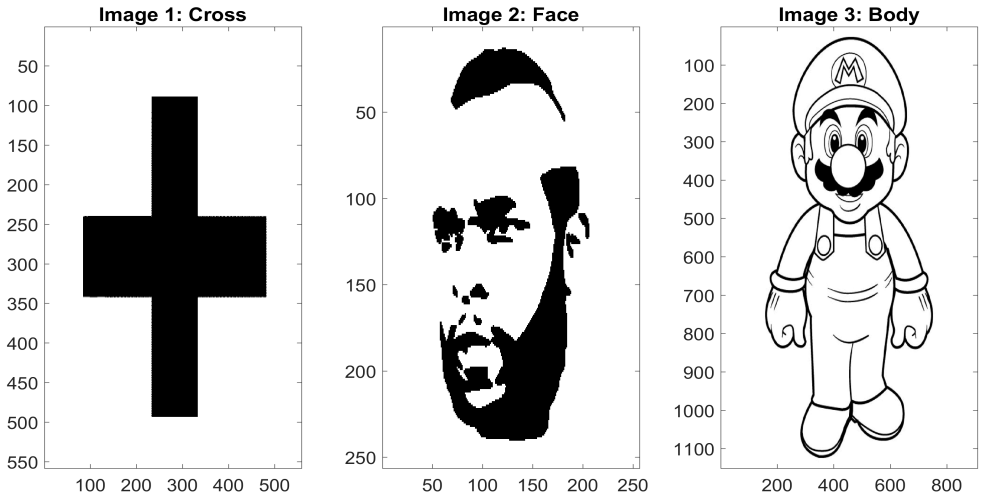


Figure 2.7: Results for the cross with five specifications of the core array.

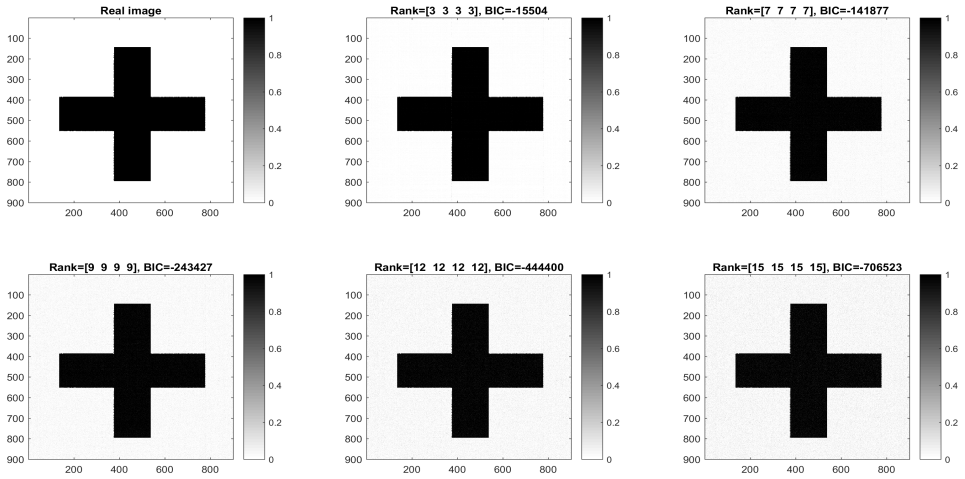


Figure 2.8 shows the results of the tensor regression for the face image. Also in this case, the model reproduces the structural coefficient with fairly high accuracy even with a small rank specification. It is worth mentioning that in this case the specification that is able to retrieve the full image without distortions is not the one with the smallest rank specification, as in the first example. This is what we would expect, given the intrinsic higher difficulty. In fact, the [3 3 3 3] specification is not enough and a slightly higher rank size is required.

Figure 2.8: Results for the face with five specifications of the core array.

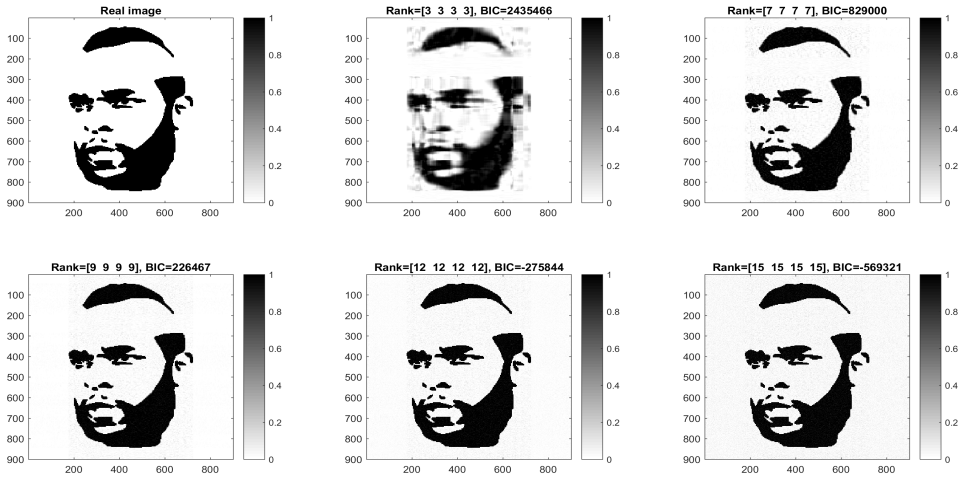
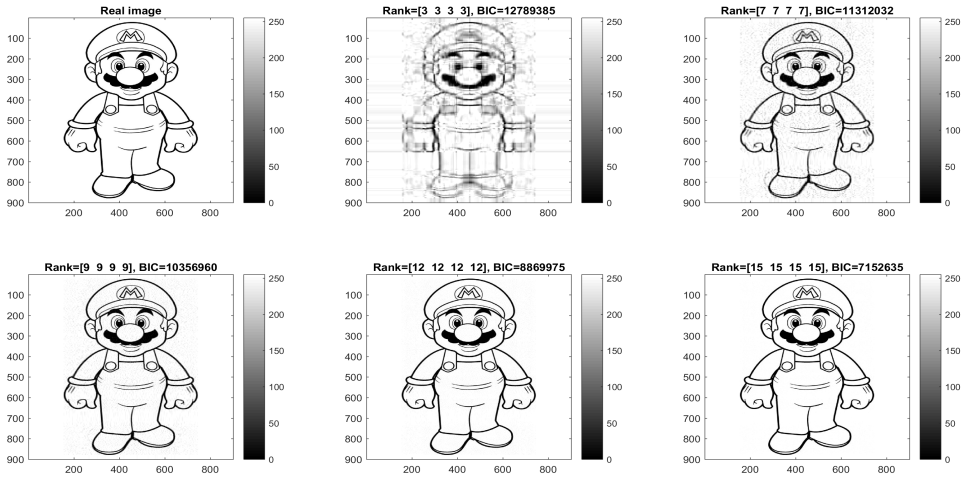


Figure 2.9 shows the results of the last example. Again, the low rank specification provides impressive accuracy, with small noise error in the [7 7 7 7] specification. The error is almost all eliminated in the higher rank specifications.

Figure 2.9: Results for the body with five specifications of the core array.



We showed with two simulation examples that the model is able to correctly estimate the true structural coefficient and to predict the response tensor variable. In the next section, we will test the Tensor regression model on real data.

2.7 Empirical application

In this section, we will apply our model to forecasting. The aim is to test its prediction performance against the existing models. In particular, we will apply the model to two different datasets. The first one is the Foursquare dataset, which is already used in the literature (Yu et al., 2015) to test the forecasting ability of multitasking learning models. The second one consists in a panel of macroeconomic time series for various European countries.

2.7.1 Datasets

The Foursquare dataset contains the users' check-in records in the Pittsburgh area between 24/02/2012 and 24/05/2012. On Foursquare, a person can check-in at his current location, leave tips about the venue, explore discounts around his current location, add other people as his friends and so on. The dataset at hand records the number of check-ins by 121 users belonging to the 15 different types of venues included, such as Art & Entertainment, College & University, and Food over 1198 time periods. Yu et. al (2015) compare their RMSE with the one provided by other models. We perform the same exercise and compare the RMSE of the TAR model with the one of their model.
The second dataset used is a subsample of the data presented in Pesaran et al. (2009) on Global VAR forecasting ability. The subsample contains quarterly data of GDP growth, inflation, short rate and stock index returns for France, Germany, Italy, Spain and UK from 1980Q1 to 2011Q4. The data is made stationary and normalized. The dataset is a tensor of $132 \times 4 \times 5$ variables. The $\mathscr{B}$ to be estimated is a $4 \times 4 \times 5 \times 5 = 400$ coefficients tensor.

2.7.2 Forecasting

We now consider the Tensor regression in which the predictor ($\mathscr{X}$) is a lagged version of the response variable ($\mathscr{Y}$). It is the case of autoregression in which a set of variables is regressed on their past values. In the case of an Autoregressive model, the coefficient tensor can be seen as a sort of Multilayer Granger causality network in which each coefficient determines the effect in time and space from one variable to itself and to the

other variables. We will test the forecasting ability of the proposed model comparing the Root mean squared error of the two models to the Vector Autoregression (VAR) specification. In particular, we will compare the TAR with the Accelerated Low-Rank Tensor Learning (ALTO) model proposed by Yu et al. (2015) for the Foursquare dataset and with a standard VAR model to the Macroeconomic example.
To test the TAR forecasting ability and compare it with alternative models, we compute the mean root square forecasted error, i.e.:

$$RMSFE_t = \sqrt{\frac{\sum_{i=1}^{t}(\widehat{Y}_{t+i} - Y_{t+i})^2}{t}}$$

where t is the forecasting horizon.

2.7.3 Empirical results

In the following subsection, we show the empirical results of the Tensor Autoregression for the two datasets.

Foursquare

For the Foursquare dataset we compare the forecasting error of the Accelerated Low-Rank Tensor Learning (ALTO) model proposed by Yu et al. (2015)[8] with the TAR model. We use the authors' code for the ALTO model and the TAR model (written in Matlab) and compare the RMSFE of the two models with up to 3 lags. In Table 2.1 are summarized the results obtained using two different settings: 80% of training set and 90% of training set.[9]

Table 2.1: RMSFE of the two models for the Foursquare data

Lag	**ALTO**	**TAR**	**ALTO**	**TAR**
	80% training set		**90% training set**	
1	0,13487	0,12520	0.13120	0.12499
2	0,12986	0,12517	0.12649	0.12494
3	0,14116	0,12517	0.12611	0.12494

[8]The data and the Matlab code are available at the webpage of Rose Yu, http://roseyu.com/code.html
[9]The training set is the subsample corresponding to the first portion of the data that is used to estimate the model coefficients. The remaining part, the test set, is used to compute the forecasting error.

We can infer from the results that the TAR model overperformes the ALTO at any lag specification an in both the test settings.

Macro data

Now, the main results of the paper will be provided. The comparison between the TAR and the standard VAR specification is made over four forecasting steps (four quarter ahead) for which the Diebold-Mariano (DM) test (Diebold and Mariano (1995))is performed between the forecast errors of the TAR and VAR.[10]

For this empirical study, we use as baseline VAR model a VAR(1) which is estimated for each country separately. For the TAR, we used a 1 lag specification with three different level of parsimony: high parsimony (HP), which is retrieved by imposing a low rank core tensor, a medium parsimony (MP) and a low parsimony (LP) level. For this purpose, we use a core tensor with ranks {1,1,2,2} for HP, {2,2,3,3} for MP and for {3,3,4,4} LP. Table 2.2 summarizes the number of parameters for each level of parsimony.

Table 2.2: Complexity of the TAR with different rank specifications

	Size of $\mathscr{B}$	HP	MP	LP	VAR model
Number of free parameters	400	32	82	208	80

We present the results of the Diebold-Mariano test in Tables 2.3, 2.4, 2.5. The tables show the p-values associated to the modified Diebold-Mariano test theorized by Harvey et al.(1997). Their test corrects the standard DM test in order to be robustly used in the cases in which the number of forecasting errors is small. The Null hypothesis states that the two competing models have the same predictive power against the alternative for which they have different forecasting accuracy. Values lower than $\alpha = 0.05$ represents a rejection of the Null. We can notice that the two models have similar predictive ability in the majority of the cases. Exceptions to this behaviour are in favour of the TAR even if the victory is not overwhelming. It is also possible to notice that the HP specification of the TAR already overperformes the VAR model. By increasing the free parameters, the model performance is significantly enhanced both in sample fit and in forecasting accuracy.

[10]The data is split in two samples. The training set corresponds to 90% of the entire sample. The 10% test sample is used to compute the forecasting errors.

Table 2.3: P-values of the DM test between TAR{1; [1, 1, 2, 2]} and the VAR(1). Bold numbers refer to cases in favour of the TAR while red italic numbers refer to cases in favor of the VAR.

	FRA	GER	ITA	ESP	UK	FRA	GER	ITA	ESP	UK
	t+1					**t+3**				
y	*0.0328*	0.3416	0.5319	0.1454	0.1899	0.9143	0.9530	0.6203	0.0846	0.9327
π	0.5148	0.8375	0.0515	0.1749	0.2883	0.5517	0.6157	0.1244	**0.0016**	0.4796
r	0.3915	**0.0104**	0.0811	0.1228	0.1516	0.5478	**0.0208**	0.4142	0.2196	0.2116
R	0.7755	0.8827	0.3111	0.3104	0.8337	0.7212	0.4710	0.2646	0.2745	0.6209
	t+2					**t+4**				
y	0.5679	0.6875	0.7230	0.0940	0.1748	0.3752	0.6469	0.5355	0.3811	0.1890
π	0.8333	0.2530	0.1035	**0.0259**	0.4078	0.3600	0.3580	0.1351	**0.0095**	0.7168
r	0.4631	**0.0187**	0.2233	0.1958	0.2075	0.5793	**0.0388**	0.4765	0.2755	0.2207
R	0.8119	0.6743	0.1789	0.2765	0.9731	0.7835	0.9702	0.3804	0.2628	0.8220

Table 2.4: P-values of the DM test between TAR{1; [2, 2, 3, 3]} and the VAR(1). Bold numbers refer to cases in favour of the TAR while red italic numbers refer to cases in favor of the VAR.

	FRA	GER	ITA	ESP	UK	FRA	GER	ITA	ESP	UK
	t+1					**t+3**				
y	0.0667	0.3721	0.9268	0.1849	*0.0292*	0.5936	0.9977	0.1910	0.2011	0.8389
π	0.8636	0.6040	0.5859	0.6655	0.3411	0.2692	0.2889	0.6126	**0.0033**	0.3015
r	0.2192	**0.0098**	0.6562	0.6830	0.2645	0.4774	**0.0207**	0.5564	0.7060	0.2956
R	0.8232	0.6306	0.4557	0.2567	0.4859	0.5288	0.2129	0.2759	0.2870	0.3313
	t+2					**t+4**				
y	0.8116	0.4721	0.7240	0.1448	0.4403	0.6416	0.3847	0.1511	0.5620	0.3710
π	0.2795	0.1663	0.9002	0.1674	0.3768	0.1585	0.3796	0.2947	**0.0084**	0.2865
r	0.3629	**0.0204**	0.9204	0.7021	0.3187	0.6692	**0.0396**	0.9195	0.6206	0.2791
R	0.5406	0.3990	0.2314	0.2654	0.4178	0.7086	0.8757	0.3191	0.3038	0.4896

Table 2.5: P-values of the DM test between TAR{1; [3, 3, 4, 4]} and the VAR(1). Bold numbers refer to cases in favour of the TAR while red italic numbers refer to cases in favor of the VAR.

	FRA	GER	ITA	ESP	UK	FRA	GER	ITA	ESP	UK
	t+1					**t+3**				
y	0.0921	0.9756	0.4058	0.2767	0.1813	0.6739	0.4082	0.5050	0.3772	0.7507
π	0.4579	0.2684	0.5981	0.3470	0.1560	*0.0264*	0.1157	0.5772	**0.0106**	0.1095
r	0.3480	**0.0129**	0.8484	0.4005	0.8496	0.6822	**0.0251**	0.2685	0.4758	0.8838
R	0.5862	0.5815	0.3918	0.2798	0.9028	0.6027	0.2280	0.2423	0.4487	0.1744
	t+2					**t+4**				
y	0.9636	0.5665	0.3252	0.2200	0.3247	0.4597	0.5911	0.5187	0.9650	0.8390
π	0.0550	0.0583	0.8420	**0.0068**	0.1860	*0.0200*	0.1442	0.3054	**0.0313**	0.0632
r	0.4299	**0.0243**	0.5997	0.3969	0.8627	0.9264	**0.0397**	0.7069	0.3275	0.9297
R	0.5278	0.4140	0.1937	0.4857	0.4093	0.4279	0.8722	0.2571	0.5784	0.1852

2.8 Conclusions

In this paper, we have introduced a tensor (Auto)regression (TAR) in which both the dependent and independent variables are tensors. To estimate the huge dimensional coefficient tensor, a parsimonious extension to handle big data is developed. An ALS algorithm is built and tested on a simulation exercise The simulations results are good and the TAR is applied to the Foursquare spatio-temporal dataset and to a panel of macroeconomic time series. The forecasting ability of this model is than tested against two different models: the ALTO model developed by Yu et al. (2015) and the Vector autoregression. Results show an overperformance of the TAR model over the ALTO model applied to the Foursquare dataset. The application to the Macroeconomic data confirms that, even if in the majority of the cases the two model have the same statistical accuracy in forecasting, the proposed model is better than the VAR specification when analysed over the time horizon and the countries evaluated. Future works would impose a structure on the coefficient tensor, leading to the structural tensor autoregression (STAR).

Chapter 3

Latent stock correlation projection via tensor decomposition

Correlation matrices are ubiquitous in financial statistics both in research papers and in the industry. The correlation is the most used method of association between stock returns. Portfolio allocation, risk management and network analysis are based on correlation matrices. Applied researchers debate about the correct correlation matrix to use. In this regard, one of the issues is related to the choice of the sample period used in the reference models. This is because the true correlation is not available. Several approaches exist in order to handle this problem; yet, all of them are heavily dependent on the time horizon used. In this paper, we introduce a method which aims to alleviate the problem of true correlation availability by decomposing the time series of correlations into a static and a dynamic component. The static component encapsulates the latent, long-run behaviour of the correlations while the dynamic part represents the time dependent structure in the latent space. Hence, the latent correlation matrix projected by the tensor factorization can be plugged in the models as an alternative to the standard correlation matrix. We will show that the hidden correlation is empirically almost time invariant.

3.1 Introduction

Since the pioneering work by Markowitz (1952, 1959), correlation matrices are ubiquitous in financial statistics both in re-search papers and the industry. The Pearson correlation is the most used measure of association between stock returns because of its simple application and interpretation. Portfolio allocation and risk management heavily rely on correlation matrices. Applied researchers struggle on the choice of the representative correlation matrix to use. One of the issues which arises in this context is the selection of the appropriate sample period to use in order to implement models. This is because the true correlation is not available. Several approaches exist to tackle the problem but all of them strongly rely on the time period selected. In this paper we introduce a method to alleviate this issue by decomposing the time series of correlations into a static and a dynamic component. More precisely, Tensors decomposition is introduced to break down the time series of correlation matrices into the two separated parts. We will treat the time series of matrices as a tensor of dimension $N \times N \times T$, where N is the number of stocks and T is the number of time stamps. The static component induced by the factorization contains the latent, long-run, behaviour of the correlations while the dynamic part represents the time dependent structure in the latent space. The hidden correlation matrix projected by the tensor factorization can be plugged in the models as an alternative to the standard correlation matrix. We then show that the latent correlation matrix is (empirically) almost time invariant, while the standard correlation matrix is not. The former matrix is a good proxy for future correlations since it does not rely on the selected sample period.

3.2 Pearson correlation matrix

Co-movements between stocks is a fundamental concept in Finance. It helps understanding the behaviour of stock indexes and building portfolios. Furthermore, in specific financial conditions, like distress periods, association between stocks reveals more information about risk variations than the variance of the market indexes. Various form of association between variables are available from the statistical toolbox. In financial

economics, the reference measure for association is the Pearson correlation, which is a linear form of association between random variables. Such measure is generally applied to stock returns. There are two di different categories of stock returns: simple returns and log-returns. Simple returns for stock i at time t are defined as follows:

$$s_{i,t} = \frac{P_{i,t} - P_{i,t-1}}{P_{i,t-1}}$$

While, log returns are defined as:

$$r_{i,t} = ln(P_{i,t}) - ln(P_{i,t-1})$$

In this work we will employ synchronous correlation between log-returns. To define the correlation coefficient let $\mathbb{E}[a]$ denote the expected value of a random variable a. The variance of a random variable X is defined as:

$$Var(X) = \mathbb{E}[(X - \mathbb{E}[X])^2],$$

while the covariance between a pair of random variables X and Y is defined as:

$$Cov(X, Y) = [(X - \mathbb{E}[X])(Y - \mathbb{E}[Y])].$$

Given this formulation we can write the Pearson correlation coefficient between two random variable, say X and Y, as:

$$corr(X, Y) = \rho_{X,Y} = \frac{Cov(X, Y)}{\sqrt{Var(X)Var(Y)}}.$$

If X is a random vector, we can write the following matrix representation of the previous equations:

$$Var(X) = \Sigma = \mathbb{E}[(X - \mathbb{E}[X])(X - \mathbb{E}[X])^T].$$

Let C be the correlation matrix associated to $corr(X)$ and D a diagonal matrix such that

$D_{ii} = \sqrt{Var(X_i)}$. We can than define:

$$\Sigma = DCD$$

and

$$C = D^{-1}\Sigma D^{-1}.$$

Every entry of the C matrix is associated with the correlation between two random variables, i.e. C_{ij} is the correlation between X_i and X_j.

Despite its easiness of computation and interpretation, not all sample correlation matrices constructed via pairwise correlation are correlation matrices. In fact, to define a matrix as âcorrelation matrixâ it is not enough to have a symmetric matrix with bounded values between -1 and +1 and with all ones on the main diagonal. A correlation matrix must fulfil an additional constraint. The matrix is in fact required to be positive semidefinite. All correlation matrices are positive semidefinite but not all sample correlation matrices are guaranteed to have this property. Take as an example the following matrix A

$$A = \begin{bmatrix} 1 & -0,6 & 0,8 \\ -0,6 & 1 & 0,8 \\ 0,8 & 0,8 & 1 \end{bmatrix}$$

A positive semidefinite matrix has non-negative eigenvalues. Nevertheless, if we compute the eigenvalues of the matrix A, $eig(A)$ we find that the three eigenvalues (in increasing order) are -0.47, 1.60 and 1.87.

This example provides evidence of a symmetric matrix, with values in the interval [-1,+1] and values equal to 1 on the diagonal. Despite the matrix is symmetric, yet it is not a correlation matrix. In this regard, Higham (2002) introduced a way to compute the closest correlation matrix given a symmetric matrix A using a weighted Frobenius norm loss function and an alternating projection method, i.e:

$$min\{\|A - X\|_F \text{: } X \text{ is a correlation matrix}\}$$

We refer to the original paper by Higham (2002) for technical details on the numerical

method implemented. We will rely on this feature to the construction of the latent correlation matrix.

3.3 Tensor decomposition

A tensor, as in the case of SVD for matrices, can be factorized in smaller objects. These objects encapsulate information on the latent space of a specific dimension of the tensor. In the case of a time series of matrices, representing correlations, covariances or networks, the factorization will be achieved on the time dimension and on the *in-out* and *out-in* dimensions.

The first factor matrix represents the dynamic part of the tensor while the other two factor matrices represent the static (long-run) behaviour of the interconnections between stocks. The latter, when combined, will resemble the latent correlation matrix. There are different decompositions we can use to âbreak downâ a tensor. We will rely on the two most important factorization methods, i.e. PARAFAC (PARallele FACtor) and Tucker decomposition. Further, we will proceed with their comparison. Finally, we will also propose and explore a new factorization method which we will name the Slice-Diagonal Tensor (SDT) decomposition. This new factorization strategy is a trade-off between parsimony and feasibility. Hence, it represents a generalization of the PARAFAC model as well as a parsimonious version of the Tucker factorization.

3.3.1 PARAFAC decomposition

The PARAFAC decomposition introduced in psychometrics by Harshman in 1970 (Carrol and Chang (1970) independently proposed the same model naming it Canonical Decomposition (CANDECOMP)) is an extension of the bilinear factor model to the multilinear case. Mathematically, it is based on the idea of rewriting a tensor as the sum of rank-one tensors. For example, take a 3^{rd} order tensor $\mathscr{Y} \in \mathbb{R}^{I \times J \times R}$. We want to rewrite this tensor as:

$$\mathscr{Y} \approx \sum_{k=1}^{K} \mathbf{a}_k \circ \mathbf{b}_k \circ \mathbf{c}_k, \tag{3.1}$$

where K is a positive integer and represents the number of componets used in the decomposition, $\mathbf{a}_k \in \mathbb{R}^I$, $\mathbf{b}_k \in \mathbb{R}^J$ abd $\mathbf{c}_k \in \mathbb{R}^R$ for $k = 1, ..., K$. A grahical representation of the PARAFAC decomposition of a 3^{rd} order tensor is illustrated in Figure 3.1

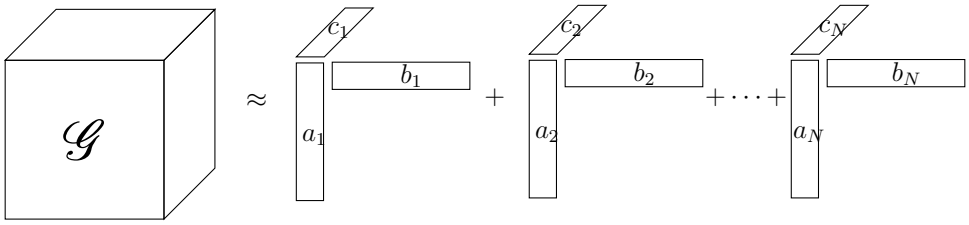


Figure 3.1: PARAFAC decomposition of a 3-dimensional tensor.

PARAFAC decomposition is currently applied to different fields of research. For example, it is often applied to psychometry. More precisely, this technique was firstly introduced in the sector by Carrol and Chang (1970) and Hashman (1970) in order to study patients for which the variables of interest were observed at different occasions, generating an generating an individuals $\times$ variables $\times$ occasions third-order tensor. A very successful field of research in which the PARAFAC decomposition was used is chemometrics, where it is used to model fluorescence excitation-emission data. In this sector, the model was introduced by Appellog and Davidson (1981) and then further studied by Anderson and Bro (2003), Anderson and Henrion (1999), Bro (1997, 1998,2006) and Bro and Anderson (1998). Other applications include data mining (Acar et. al (2005, 2006)) and text analysis and semantic networks (Bader et al. (2007)).

3.3.2 Tucker decomposition

The Tucker decomposition was theorized by Tucker (1964, 1966) .As the PARAFAC decomposition, it represents an extension of the biliniear factor analysis to the multi-way case. It is also referred to as N-mode PCA in Kapteyn et al. (1986) and Higher-order SVD by De Lathauwer et al. (2000). Take a third-order tensor $\mathscr{Y} \in \mathbb{R}^{I \times J \times R}$. The decomposition take the form of a *n-mode* product, i.e.:

$$\mathscr{G} \approx \mathscr{C} \times_1 L \times_2 R \times_3 T, \tag{3.2}$$

where $\mathscr{C} \in \mathbb{R}^{P \times Q \times K}$ is a core tensor with weights of the relationship between the modes of the tensor, $L \in \mathbb{R}^{I \times P}$, $R \in \mathbb{R}^{J \times Q}$ and $T \in \mathbb{R}^{R \times K}$. The Tucker model for a third-order tensor, which factors the tensor in a small core tensor and three factor matrices. This is shown in Figure 3.2 below.

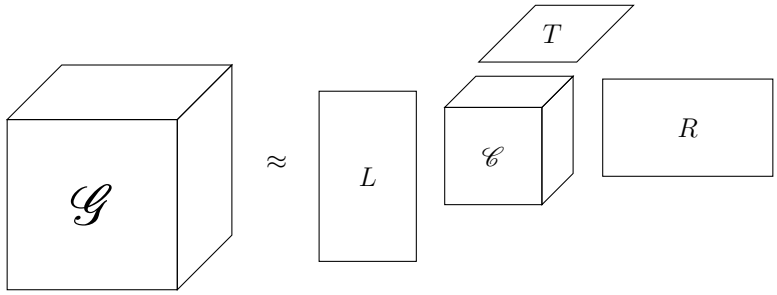


Figure 3.2: Tucker decomposition of a 3-dimensional tensor.

Equation 3.2 can be rewritten in terms of the outer product as:

$$\mathscr{Y} \approx \sum_{p=1}^{P} \sum_{q=1}^{Q} \sum_{k=1}^{K} \mathbf{c}_{pqk} \mathbf{a}_p \circ \mathbf{b}_q \circ \mathbf{c}_k. \tag{3.3}$$

Equation 3.3 makes the comparison between the PARAFAC and Tucker models easier. It is possible to see that if $P = Q = R$ and the core tensor $\mathscr{C}$ is superdiagonal, i.e. $c_{i_1 i_2 \ldots i_N} \neq 0$ iff $i_1 = i_2 \ldots = i_N$, the two models are equivalent. This means that the PARAFAC can be seen as a special (constrained) case of the Tucker model. The higher flexibility embedded in the Tucker model come to a cost. The Tucker model is generally less parsimonious than the PARAFAC model, unless a strongly unbalanced tensor is to evaluate.

Take as example a third-order array $\mathscr{X} \in \mathbb{R}^{1000 \times 100 \times 100}$. Assuming that the first mode can be explained by 100 factors while the second and third modes can be represented by 10 components each, a PARAFAC model will use 100 components to rewrite the tensor. Therefore, this will cost $1000 \times 100 + 100 \times 100 + 100 \times 100 = 120000$ parameters to be estimated. A Tucker model can be set to have 100 components on the first mode and 10 in the remaining two. This will lead to the estimation of $1000 \times 100 + 100 \times 10 + 100 \times 10 + 100 \times 10 \times 10 = 112000$ parameters. In this case the Tucker model is more convenient since it is more flexible in the dimension of each factor matrix and since it provides a more parsimonious representation. However, it is not generally the case since the Tucker

model has the core tensor to be estimated in addition to the PARAFAC model, which can be high-dimensional itself. Another issue concerning the Tucker model is the lack of uniqueness. It is in fact possible to modify the core tensor by some nonsingular matrices and apply the inverse of those matrices to the factor matrices without alterate the fit of the model. The Tucker model is then unique up to rotation of the components. The Tucker model is then unique up to rotation of the components. Applications of the Tucker model are various. De Lathauwer and Vandewalle (2004) apply it to signal processing, Kiers and Van Mechelen (2001) to problems in psychometrics Henrion (1994) instead uses the Tucker factorization in chemical analysis. Other applications in computer vision, image recognition, text analysis and optimization are also well documented in the literature.

3.3.3 Slice-diagonal tensor decomposition

After introducing the two most used tensor decomposition, we propose a new tensor decomposition method which represents a more flexible version of the PARAFAC model. Similarly to the Tucker decomposition this methodology allows to impose different dimensions over each mode factor. Nevertheless, it differs from the Tucker version since it is much more parsimonious. In fact, the Slice-diagonal tensor decomposition is characterized by a core tensor which is slice-diagonal, embedding stronger uniqueness properties. The mathematical representation is equivalent to the Tucker decomposition, i.e. :

$$\mathscr{G} \approx \boldsymbol{\Lambda} \times_1 L \times_2 R \times_3 T, \tag{3.4}$$

where $\boldsymbol{\Lambda} \in \mathbb{R}^{P \times Q \times K}$ is a slice-diagonl core tensor, $L \in \mathbb{R}^{I \times P}$, $R \in \mathbb{R}^{I \times Q}$ and $T \in \mathbb{R}^{I \times K}$. The Slice diagondal tensor decomposition is depicted in Figure 3.3

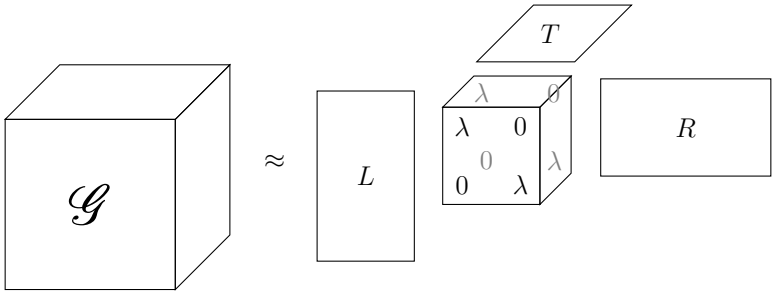


Figure 3.3: SD-Tucker decomposition of a 3-dimensional tensor.

This new decomposition technique has several advantages. The first one is surely the parsimony of the model, which does not eliminate the flexibility of the Tucker decomposition. A second advantage is the restrictions on the core tensor act reducing the possible rotations of the components, entailing improved uniqueness properties. A third advantage of this representation is achieved when we deal with skewed, asymmetric tensors as for example the bilateral trade matrix. In this context, the Tucker decomposition will have difficulties to retrieve the full asymmetry in the bilateral exposures. This is because part of this asymmetry is incorporated on the core tensor, while the remaining part is incorporated in the factor matrices. RESCAL decomposition (Nickel et al. (2011)) tackles this drawback by restricting the Tucker-2 model (A Tucker factorization with only two factor matrices estimated and the remaining constrained to be the Identity matrix) such that the factor matrices over the bilateral exposure are constrained to be equal so that the asymmetry is only reflected in the core tensor. However, the RESCAL consider the decomposition in a core tensor and the two factor matrices, while in our context we want to have the third, dynamical, factor matrix. The proposed method, having a core tensor which is slice-diagonal, has only the variance weights on the core and the possible asymmetries are encoded in the factor matrices. This decomposition has also a version in which only two factor matrices are estimated as for Tucker-2, i.e.:

$$\mathscr{G} \approx \boldsymbol{\Lambda} \times_1 L \times_2 R, \tag{3.5}$$

where $\boldsymbol{\Lambda} \in \mathbb{R}^{P \times Q \times T}$ is a slice-diagonal core tensor which incorporates also the dynamics of the variance structure of the tensor, $L \in \mathbb{R}^{I \times P}$ and $R \in \mathbb{R}^{I \times Q}$ are the factor matrices containing information on the long-run asymmetric relationships between the *in* and *out* entries of the original matrices. The SDT-2 decomposition is depicted in Figure 3.4

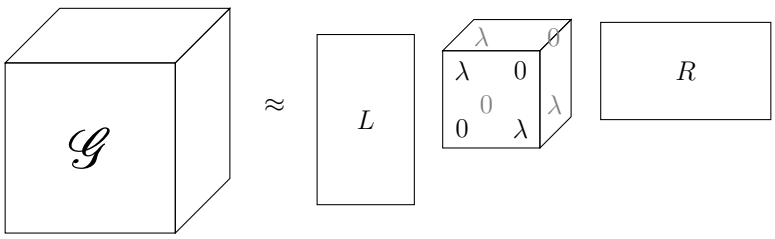


Figure 3.4: SDT-2 decomposition of a 3-dimensional tensor.

Before discussing the application of the three different factorizations, it is necessary to

analyse the number of components to be estimated for each model in the case of a third-order tensor. We divide the analysis of the model size in two separate subsets, one related to the decompositions concerning three factor matrices (3FM), i.e. PARAFAC, Tucker and SDT, and one concerning two factor matrices (2FM), named Tucker-2, RESCAL and SDT-2. We rely on the notation used in equations 3.1, 3.2, 3.4 for the dimensionality of the components. Table 3.1 represents the analysis of the 3FM case:

Table 3.1: Number of free parameters to estimated in each model

Decomposition model	**Number of free parameters**
PARAFAC	$K \times (I + J + R)$
Tucker	$P \times I + Q \times J + K \times R + P \times Q \times K$
SDT	$P \times I + Q \times J + K \times R + P \times K$

It is clear that the Tucker model, except for some rare circumstances, is the model with the highest number of free parameters to estimate. Nevertheless, the flexibility given by the core tensor in this model increases the possibility to have decompositions. The number of parameters estimated in the decompositions is similar to the PARAFAC model, which in fact is forced to have the same number of components in each mode of the factorization. Now, let us consider again the third-order tensor of section 3.2, $\mathscr{X} \in \mathbb{R}^{1000\times100\times100}$. The first mode can be explained by 100 factors while the second and third modes can be represented by 10 components each. The PARAFAC(10) and Tucker(100,10,10) then have respectively 120000 and 112000 free parameters to be estimated. On the other hand, the SDT(100,10,10) has a model complexity equal to $1000\times100+100\times10+100\times10+100\times10 = 103000$. There are examples in which the PARAFAC model is more parsimonious than the SDT factorization, while for the Tucker it is never the case. Table 3.2 refers to the 2FM models:

Table 3.2: Number of free parameters to estimated in each model

Decomposition model	**Number of free parameters**
Tucker-2	$P \times I + Q \times T + P \times Q \times R$
RESCAL	$P \times I + P \times Q \times R$
SDT-2	$P \times I + Q \times J + P \times R$

In this case the parsimony of the SDT-2 decomposition is even more pronounced since the core tensor has a huge impact on the total computational burden. Despite the comparison

of the performance of the SDT-2 compared to the other 2FM models is of huge scientific interest, it is not the focus of this work.

3.4 Application

In this section we will analyse the different factorization techniques and compare their results. We use financial data to build a tensor and we will study it in details

3.4.1 Data

For the purpose of this research, we use a subsample of intraday data of the S&P100 from January 2005 up to June 2017. In particular, we use 65 out of the 100 available stocks since these stocks are observable throughout the entire sample period. With this data we are able to compute three different tensors, one for the covariance ($\mathscr{S}$) and one for the correlation($\mathscr{C}$). Each tensor is constructed concatenating the respective matrices over time. We build every matrix using one month of data so that we have twelve matrices per year. After the concatenation of the set of matrices we have third-order tensor of dimension $65 \times 65 \times 150$. The first tensor contains the stocks covariances over time, so for example $\mathscr{S}_{1,2,10}$ represents the covariance between stock 1 and 2 in the 10th month, while $\mathscr{S}_{10,10,100}$ is the variance of stock number 10 in the 100th month. The same argument applies for the other two tensors. What is worth mentioning is that in the case of the Asymmetric correlation tensor $\mathscr{S}_{i,j,t} \neq \mathscr{S}_{j,i,t}$ so that the correlation between stock i and j at any point in time is not symmetric.

3.4.2 Estimation

The decomposition is performed by an optimization problem which tries to minimize the Frobenious norm of the error. Let's write the decomposition as $F(\mathscr{X}) = \widehat{\mathscr{X}}$, where $F(\cdot)$ is one of the three decomposition under examination. The minimization problem can be written as:

$$\min_{f_1,\dots,f_n} \| \mathscr{X} - F(\mathscr{X}) \|_F \tag{3.6}$$

Where $f_1, ..., f_n$ are factors of the decomposition at hand. This op-timizaion is a non-linear least square problem that cannot be with a simple operation. To solve this we can rely on an iterative algorithm such as the Alternating least squares (ALS). ALS was introduced for PARAFAC by Carrol and Chang (1970) and Harshman (1970) and for the Tucker decomposition by Kroonenberg and Dw Leeuw (1980) and Kapteyn et al. (1986). This methodology solves the optimization problem by dividing it into small least squares problems. This allows to solve the optimization for each factor separately, keeping constant the other ones at the previous iteration value. Then, it iterates alternatively among all the factors until the algorithm converges (the error ε reach a threshold α) or a maximum number of iterations is reached. The ALS has several applications and it is workhorse alghorithm for Tensor factorization. A sketch of the algorithm can be found in 3 below:

Algorithm 3: Alternating least squares

1: Initialize by random generated factors $f_1, f_2, ..., f_N$.
2: **repeat**
3: **for** *i=1 to N* **do**
4: $\hat{f}_i = \text{argmin}_{f_i} \parallel \mathscr{X} - F(\mathscr{X})_{\{f_1,...,f_{i-1},f_{i+1},...,f_N\}} \parallel_F$
5: **end for**
6: $\varepsilon = \frac{\|\mathscr{X} - F(\mathscr{X})_{\{\hat{f}_1,\hat{f}_2,...,\hat{f}_N\}}\|_F}{\|\mathscr{X}\|_F}$
7: **until** $\varepsilon < \alpha$ or Maximum iterations reached.
8: Return $\hat{f}_1, \hat{f}_2, ..., \hat{f}_N$

3.4.3 Model selection

Another key topic to discuss is the choice of the number of factors to be used in the decompositions. In some circumstances, the number of components comes from a theoretical analysis, e.g. in Chemometrics. In absence of theoretical guidelines we rely on data - driven approaches for model selection. In this sense, we will use the Information criterion to deduce the best model specification to use for each factorization technique In particular we will use as Information criteria the Bayesian information criteria (BIC), the Akaike information criterion (AIC) and the corrected (for finite samples) Akaike

information criterion (AIC_C). The BIC, AIC and AIC_C are computed as:

$$BIC = n \ln\left(\frac{RSS}{n}\right) + \ln(np) \tag{3.7}$$

$$AIC = n \ln\left(\frac{RSS}{n-p}\right) + 2p \tag{3.8}$$

$$AIC_C = AIC + \frac{2p(p+1)}{n-k-1} \tag{3.9}$$

where n is the number of datapoints, p is the number of estimated coefficients and RSS is the residual sum of squares of the model. The AIC is the one, among the information criteria discussed in this work, which penalize less non-parimonious models. Its correction, the AIC_C instead penalize much more over-parametrized models in case of small sample data. Finally, the BIC penalizes more for the number of free parameters compared to the Akaike information criteria. For this reason we will rely mostly on the AIC_C and BIC. The results for the model selection for the covariance and correlation tensors over the three different factorization approaches are reported in Tables 3.3 and 3.4.[1]

Table 3.3: **Model selection for the covariance tensor**

	Model specification	**BIC**	**AIC**	**AIC$_C$**
PARAFAC	*3*	-7.72e+06	-7.73e+06	-7.73e+06
	5	-7.85e+06	-7.87e+06	-7.87e+06
	10	**-8.02e+06**	**-8.05e+06**	**-8.05e+06**
Tucker	*(2,3,3)*	-7.44e+06	-7.45e+06	-7.45e+06
	(5,6,6)	-7.89e+06	-7.91e+06	-7.91e+06
	(5,10,10)	**-7.94e+06**	**-7.97e+06**	**-7.97e+06**
SDT	*(2,3,3)*	-7.21e+06	-7.22e+06	-7.22e+06
	(5,6,6)	**-7.27e+06**	**-7.29e+06**	**-7.29e+06**
	(5,10,10)	-7.25e+06	-7.27e+06	-7.27e+06

The results shown in the tables are in line over the different information criteria used. We can see in fact that all the three IC choose the same model specialisation between different model specifications. In particular, for the covariance tensor, we have that the best (penalized) fit is achieved by a PARAFAC(10), Tucker(5,10,10) and SDT(5,6,6). This converts to a model complexity of 2800 free parameters for the PARAFAC, 2550 for

[1]We display only few specifications among all the ones tested. We show the specification with the best performances.

Table 3.4: **Model selection for the correlation tensor**

	Model specification	BIC	AIC	AIC_c
PARAFAC	*3*	**-1.27e+06**	**-1.28e+06**	**-1.28e+06**
	5	-1.25e+06	-1.27e+06	-1.27e+06
	10	-1.22e+06	-1.25e+06	-1.25e+06
Tucker	*(2,3,3)*	**-1.28e+06**	**-1.29e+06**	**-1.29e+06**
	(5,6,6)	-1.25e+06	-1.26e+06	-1.26e+06
	(5,10,10)	-1.23e+06	-1.25e+06	1.25e+06
SDT	*(2,3,3)*	**-1.28e+06**	**-1.29e+06**	**-1.29e+06**
	(5,6,6)	-1.26e+06	-1.28e+06	-1.28e+06
	(5,10,10)	-1.25e+06	-1.28e+06	-1.28e+06

the Tucker decomposition and only 1560 for the SDT factorization. This huge reduction of free parameters given by SDT increases the uniqueness properties of the decomposition. The results for the covariance and correlation tensor are shown in Table 3.5.

Table 3.5: Best model specification and complexity

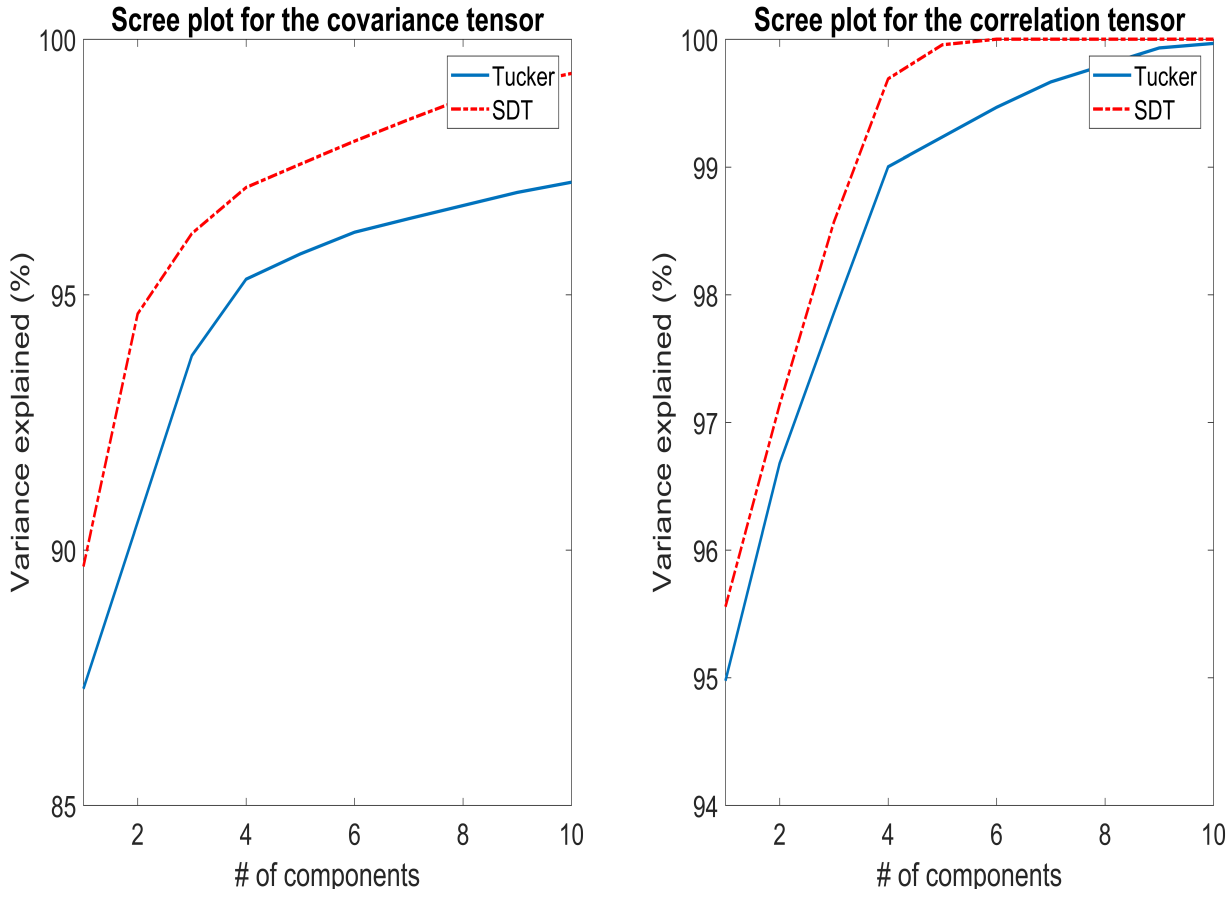
	Model	Specification	Model complexity
Covariance	PARAFAC	10	2800
	Tucker	(5,10,10)	2550
	SDT	(5,6,6)	1560
Correlation	PARAFAC	3	840
	Tucker	(2,3,3)	708
	SDT	(2,3,3)	696

3.4.4 Component analysis

We can now explore the components found by the Tensor factorization. We will explore the covariance and correlation tensor. In the Tucker and the SDT decomposition, the core array gives insight on which are the components that contribute the most in the reconstruction of the tensor. Given that those coefficients in the core array embed the variance of the tensor, we can use a scree plot to see the amount of variance explained as function of the number of variance factors. Figure 3.5 represents the results for the Tucker and SDT decompositions.[2] From Figure 3.5 it is possible to notice that already with few components a huge fraction of the core variance is retrieved. It is also important to notice that given the lower number of free component parameters in the SDT decomposition, it reflects a higher amount of variance with fewer components.

[2]The number of components is set at their optimal value given by the BIC.

Figure 3.5: Scree plot for the first 10 core tensor components



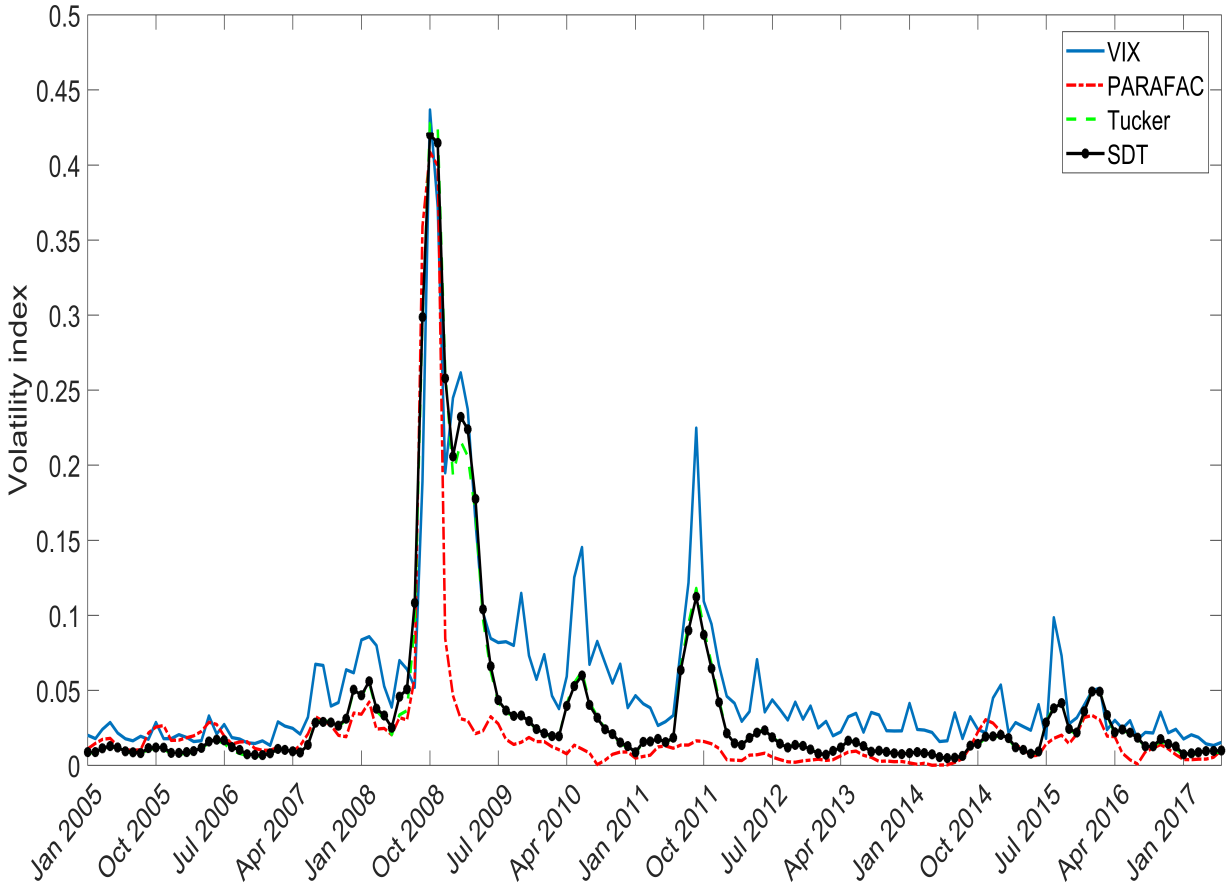
We now explore the dynamic and static componets of the decomposition

Dynamic component

In this subsection we analyse the Dynamic factor loading retrieved by the 3 different decompositions. To better understand the behaviour the dynamic components we will compare them with some financial indicator. In particular we will compare the first dynamic component of the covariance tensor (which accounts for some 90% of the variance) with the Volatility index (VIX) of the S&P and the first dynamic component of the covariance tensor with the Implied correlation index (ICI). Both indexes are provided by the Chicago Board Options Exchange (CBOE). Figure 3.6 depicts the first dynamic component of the three different decomposition techniques We can easily notice that all the decompositions are capable of reproducing the volatility behaviour over time. It is worth to notice that PARAFAC is the worst performer over the factorization techniques anal-

ysed. It correctly identify the October 2008 volatility surge but it doesn't show any effect on the subsequent three events on beginning of 2009, mid 2010 and September 2011. In contrast, Tucker and SDT decompositions are able to spot all the main volatility increases and its dynamic.

Figure 3.6: **Implied volatility index vs the first dynamic factor**

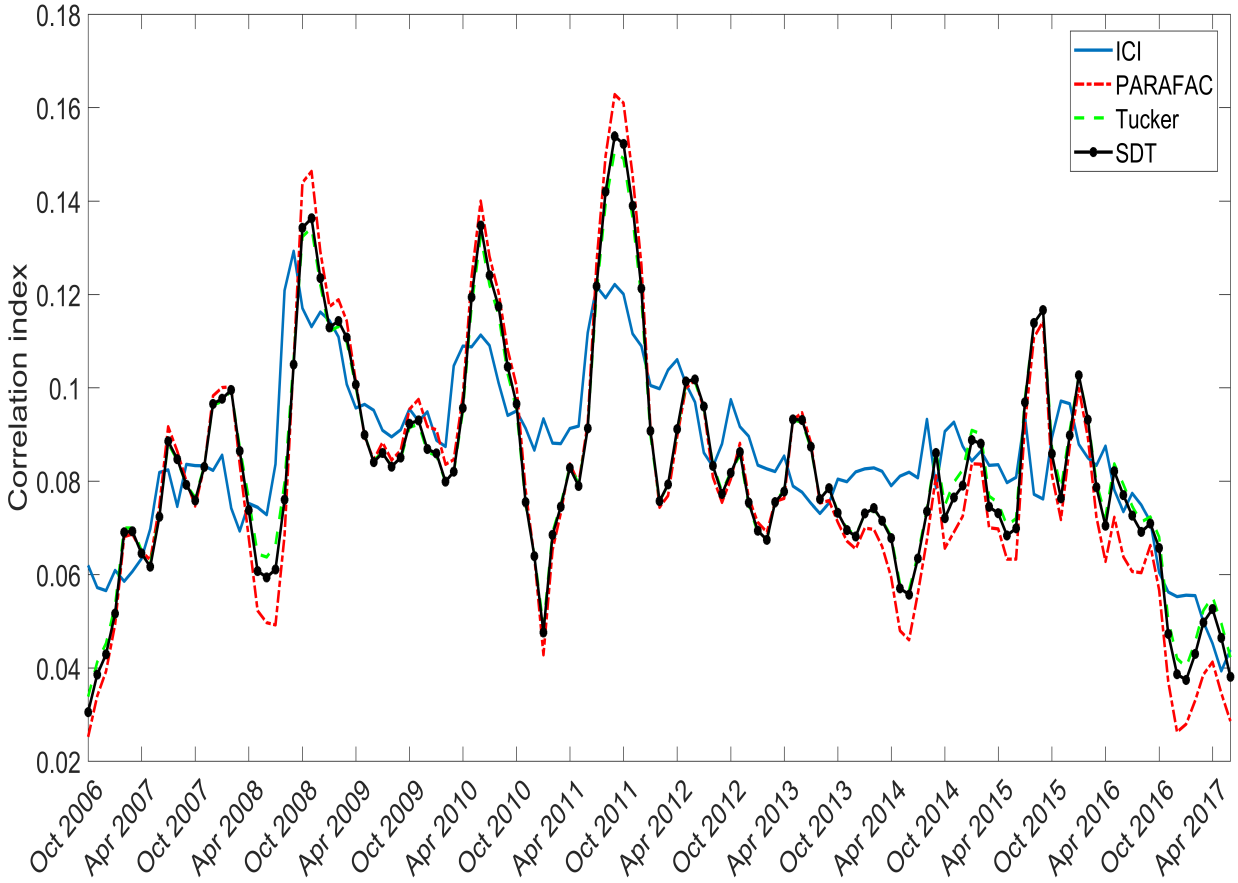


In Figure 3.7 we have the same comparison using the first component of the correlation tensor and its real world counterpart, the Implied correlation index. We can again appreciate that the decompositions capture the main dynamic of the correlation proxy, revealing a good empirical performance of the decomposition at retrieving dynamic information.

Static components

In this subsection, we analyse the static components which are represented by the latent space induced by the stocks. These loadings give information on the linkages between

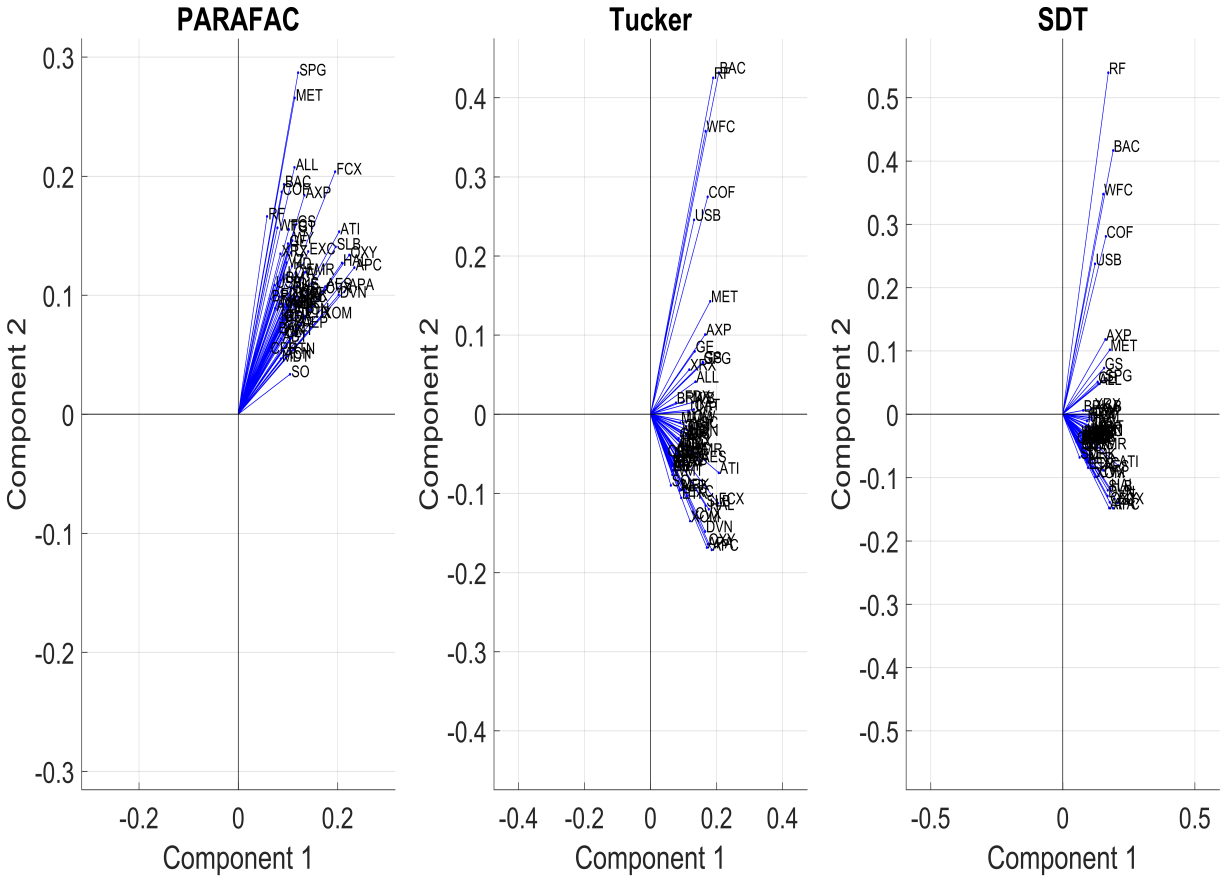
Figure 3.7: **Implied correlation index vs the first dynamic factor**



These results can also be interpreted as a whitening procedure. In fact these dynamics clean the other components from the time dependency.

stocks and its magnitude. An high level of the loading means a great effect on each component while the same direction of the loadings is a proxy of the linkages. Stocks which are matched together (both in magnitude and direction) have similar behaviour and the best way to explore this characteristics is to use a biplot. Figure 3.8 depicts the first and second static components for the PARAFAC, the Tucker and SDT decompositions. PARAFAC put all the stocks together, failing in finding similarities and dissimilarities between stocks. Tucker and SDT decomposition however, find more groups of stocks with similar behaviour. These static factors can be used to build the link matrix which acts as a latent correlation matrix.

Figure 3.8: **Biplot**



3.4.5 Latent Correlation matrix

In this subsection, we analyse the latent correlation matrix built by the tensor factorization. The Multilinear decomposition acts as a dynamic whitening of the correlation. This procedure cleans the factor loading from any time specific behavior, leaving only the long-run mean behavior of the links. Taking the decomposition induced by Equations 3.2 and 3.4, the latent link matrix O is computed as $O = LR'$ while the corresponding latent correlation matrix Ω is defined as the normalized version of O, i.e.:

$$\Omega = \Phi^{-1} O \Phi^{-1} \tag{3.10}$$

where Φ is a diagonal matrix defined as $\Phi_{ii} = \sqrt{O_{ii}}$. After the normalization we have a a matrix with ones on the main diagonal and bounded values on the off-diagonal. However, as we mentioned in section 2, a matrix with those features is not by default a correlation matrix. To prevent the matrix to be ill posed, we build the closest correlation matrix Θ

of Ω, i.e.:

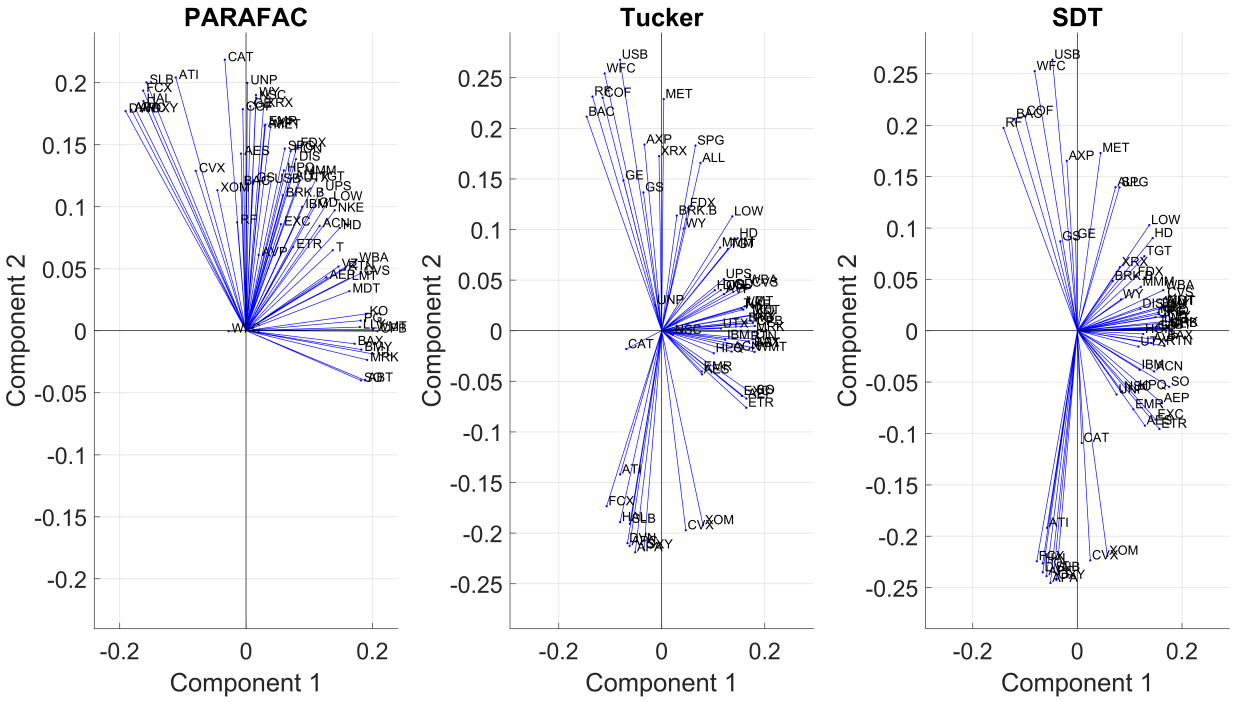
$$min\{\| \Omega - \Theta \|_F : \Theta \text{ s a correlation matrix}\}$$

It is worth to mention that for the correlation matrices studied in this work, Θ turns out to be the same as Ω for all the cases considered. We now study two different features of this correlation matrix. Firstly, we study the capabilities of this matrix to cluster the stocks, in terms of similar sectors in which the companies operate. Secondly, we analyse whether this feature remains similar in two non-overlapping periods in order to test the decompositionâs ability to retrieve structural features from the data. Finally, we explore the distribution of the eigenvalues of the latent correlation matrices of the two subsamples and we perform a Kruskal-Wallis non-parametric test to understand whether the two sets of eigenvalues come from the same distribution.

Clustering analysis

In this subsection we will use a Biplot of the first two principal components of the latent correlation matrix to depicts common features of the stocks. We will employ all the three different factorization techniques studied in this work to see if all the models capable of finding common features. This is depicted in Figure 3.9. How it is shown, the performance of the PARAFAC to cluster stocks is not accurate. In fact, it is difficult to spot clear groups of stocks. Indeed, the clustering performance of the Tucker and SDT decompositions improves considerably with respect to the PARAFAC. More over the behaviour of the two techniques is remarkably similar. It is possible to spot clearly the group of financial firms at the top of the plot and the group of basic materials industry in the bottom part. Conversely, it appears more difficult to group stocks of the remaining cloud of data. Nevertheless, we can see that on the bottom section of the cloud there are stocks related to the utility industry, while on the top there are stocks belonging to the technology & service sectors, such as Xerox, Fedex, Target group, Lowe's company and The home.

Figure 3.9: **Biplot of the latent correlation matrix components**



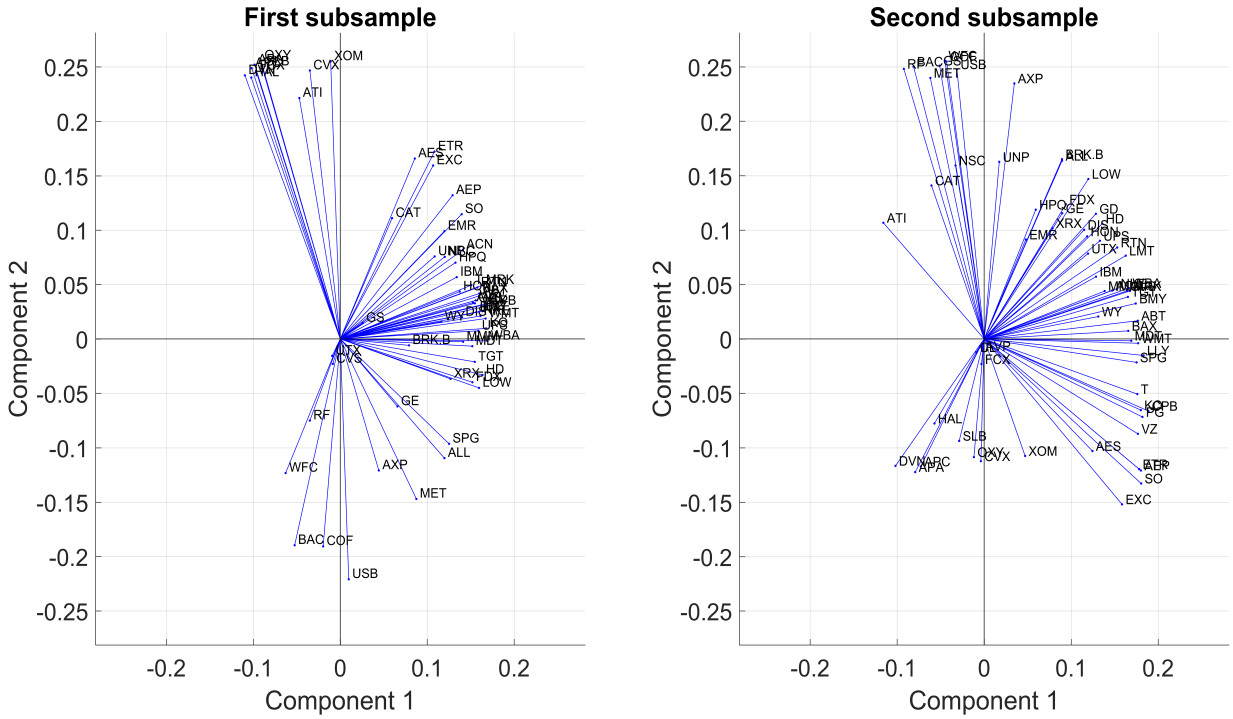
Time dependency analysis

After the analysis of the clustering capabilities of the latent correlation matrix, we study the behaviour over two non-overlapping sample periods of the latent correlation matrix. To be more price, we split the tensor in two on the time dimension, generating two correlation tensor of order $75 \times 65 \times 65$. We then fit the proposed SDT decomposition method to each tensor separately and build the two latent correlation matrices and apply the closest correlation algorithm to each matrix.

Figure 3.10 shows the biplot of the two samples. As it is possible to notice, the two graphs are similar conditional on the rotation of the components. In fact, the clusters of the basic materials and financial industries are clearly visible in both the plots, while for the other components a closer inspection is needed.

The final test to perform consists of checking whether these two correlation matrices are statistically equivalent in the distribution of the eigenvalues. To achieve this we implement Spectral similarity is a concept for which, Let us consider two matrices A and B. If the distributions of the eigenvalues of the two matrices are not statistically different, i.e. a test is not able to reject the null of same distribution, the spectral similarity is

Figure 3.10: **Biplot of the latent correlation matrices**



the concept for which A and be are defined to be similar. In particular, we perform a Kruskal-Wallis test and a two-samples Kolmogorov-Smirnoff non-parametric tests. The failure in rejecting the null hypothesis is in favour of the similar structure of the matrices analysed. The tests conceptually specify the following null and alternative hypothesis:

$$H_0 : F(\lambda_{1,i}) - F(\lambda_{2,i}) = 0$$

$$H_1 : F(\lambda_{1,i}) - F(\lambda_{2,i}) \neq 0$$

where $F(\cdot)$ is the distribution of the data and $\lambda_{j,i}$ is the set of eigenvalues of the $j-$th latent matrix. Specification of the two tests can be found in Wayne(1990). Results are depcted in Table 3.6. We can see that the null cannot be rejected by both tests, suggesting that the two sets of eigenvalues do come from the same distribution.

Table 3.6: Test of similarity the latent correlation matrices eigenvalues

Test	**p-value**
Kruskal-Wallis	0.7498
Kolmogorov-Smirnoff	0.9995

We showed that the correlation matrices constructed via the latent space whitened by dynamic components are empirically similar both in the clustering performance and in terms of eigenvalues distribution. This result allows us to consider the latent correlation matrix as a good substitute of the correlation matrix with enhanced clustering and long-term behaviour.

3.5 Conclusion

In this work, we analysed several aspects of Tensor analysis with financial data. In particular, we explored the decomposition components. We showed that the first component of Covariance tensor is a proxy of the VIX, while the first component of the Correlation tensor has similar behaviour as the Implied correlation index, confirming the reliability of the decompositions. We than moved to the projection into the latent correlation matrix composed by the static components. We appreciated the fact that the correlation matrices constructed with two non-overlapping samples show similarity in both clustering and the distribution of the eigenvalues. This makes this matrix a reliable proxy for the structural linkages.

General conclusions

In this thesis, I explored some of the tensor methods available in the literature. I studied their performance and analysed the insights on the data they provide. I also proposed two novel tensor methods, namely the Tensor (Auto)regression, in which both the dependent variable and the independent variable are tensors. When compared to other models, this new methodology has proven to have satisfactory performance both in sample and in forecasting. In the second paper, I studied the time series of covariance and correlation matrices. In performing this analysis, I also introduce a decomposition approach I named Slice-diagonal tensor decomposition. This approach is at the crossroad between Tucker and PARAFAC decompositions, being a special case of the first and a generalization of the second. It is more parsimonious than the Tucker model but more flexible than the PARAFAC. In the paper, it proved to perform well. In the same paper, I studied the dynamic and static factors induced by the tensor factorization. Future works will be devoted to streams of networks to get information on their dynamics and to find possible anomalies decomposing the adjacency tensor. A further investigation of the topic is realized in the paper "High order portfolio theory" and it will be dedicated to the study of portfolio allocation when higher order multivariate moments are taken into account, namely coskewness and cokurtosis, which are, not surprisingly, tensors.

Bibliography

[1] E. Acar, S. A. Camtepe, M. S. Krishnamoorthy, and B. Yener. Modeling and multiway analysis of chatroom tensors. *ISI*, 2005:256–268, 2005.

[2] A. Anandkumar, R. Ge, D. Hsu, S. M. Kakade, and M. Telgarsky. Tensor decompositions for learning latent variable models. *Journal of Machine Learning Research*, 15:2773–2832, 2014.

[3] A. Anandkumar, P. Jain, Y. Shi, and U. N. Niranjan. Tensor vs. matrix methods: Robust tensor decomposition under block sparse perturbations. In *Artificial Intelligence and Statistics*, pages 268–276, 2016.

[4] C. M. Andersen and R. Bro. Practical aspects of parafac modeling of fluorescence excitation-emission data. *Journal of Chemometrics*, 17(4):200–215, 2003.

[5] C. ANDERSON and R. HENRION. A general algorithm for obtaining simple structure of core arrays in n-way pca with application to fluorometric data. *Computational Statistics and Data Analysis*, 31:255–278, 1999.

[6] C. A. Andersson and R. Bro. The n-way toolbox for matlab. *Chemometrics and intelligent laboratory systems*, 52(1):1–4, 2000.

[7] B. W. Bader and T. G. Kolda. Algorithm 862: Matlab tensor classes for fast algorithm prototyping. *ACM Transactions on Mathematical Software (TOMS)*, 32(4):635–653, 2006.

[8] B. W. Bader and T. G. Kolda. Efficient matlab computations with sparse and factored tensors. *SIAM Journal on Scientific Computing*, 30(1):205–231, 2007.

[9] B. W. Bader, T. G. Kolda, et al. Matlab tensor toolbox version 2.5. *Available online, January*, 7, 2012.

[10] M. T. Bahadori, Q. R. Yu, and Y. Liu. Fast multivariate spatio-temporal analysis via low rank tensor learning. In *Advances in neural information processing systems*, pages 3491–3499, 2014.

[11] L. Barnett and A. K. Seth. The mvgc multivariate granger causality toolbox: a new approach to granger-causal inference. *Journal of neuroscience methods*, 223:50–68, 2014.

[12] R. Bro. Parafac. tutorial and applications. *Chemometrics and intelligent laboratory systems*, 38(2):149–171, 1997.

[13] R. Bro. *Multi-way analysis in the food industry: models, algorithms, and applications*. PhD thesis, Københavns UniversitetKøbenhavns Universitet, LUKKET: 2012 Det Biovidenskabelige Fakultet for Fødevarer, Veterinærmedicin og Naturressourcer-Faculty of Life Sciences, LUKKET: 2012 Institut for FødevarevidenskabDepartment of Food Science, LUKKET: 2012 Kvalitet og TeknologiQuality & Technology, 1998.

[14] R. Bro, R. A. Harshman, N. D. Sidiropoulos, and M. E. Lundy. Modeling multi-way data with linearly dependent loadings. *Journal of Chemometrics*, 23(7-8):324–340, 2009.

[15] R. Bro and H. A. Kiers. A new efficient method for determining the number of components in parafac models. *Journal of chemometrics*, 17(5):274–286, 2003.

[16] R. Bro and A. K. Smilde. Centering and scaling in component analysis. *Journal of Chemometrics*, 17(1):16–33, 2003.

[17] J. D. Carroll and J.-J. Chang. Analysis of individual differences in multidimensional scaling via an n-way generalization of âeckart-youngâ decomposition. *Psychometrika*, 35(3):283–319, 1970.

[18] E. C. Chi and T. G. Kolda. On tensors, sparsity, and nonnegative factorizations. *SIAM Journal on Matrix Analysis and Applications*, 33(4):1272–1299, 2012.

[19] A. Cichocki and A.-H. Phan. Fast local algorithms for large scale nonnegative matrix and tensor factorizations. *IEICE transactions on fundamentals of electronics, communications and computer sciences*, 92(3):708–721, 2009.

[20] A. Cichocki, R. Zdunek, A. H. Phan, and S.-i. Amari. *Nonnegative matrix and tensor factorizations: applications to exploratory multi-way data analysis and blind source separation*. John Wiley & Sons, 2009.

[21] R. Coppi. An introduction to multiway data and their analysis. *Computational statistics & data analysis*, 18(1):3–13, 1994.

[22] R. Coppi and S. Bolasco. Rank, decomposition, and uniqueness for 3-way and iv-way arrays. 1989.

[23] J. E. Davidson. Problems with the estimation of moving average processes. *Journal of Econometrics*, 16(3):295–310, 1981.

[24] L. De Lathauwer, B. De Moor, and J. Vandewalle. A multilinear singular value decomposition. *SIAM journal on Matrix Analysis and Applications*, 21(4):1253–1278, 2000.

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

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

[27] F. X. Diebold. Comparing predictive accuracy, twenty years later: A personal perspective on the use and abuse of diebold–mariano tests. *Journal of Business & Economic Statistics*, 33(1):1–1, 2015.

[28] F. X. Diebold and R. S. Mariano. Comparing predictive accuracy. *Journal of Business & economic statistics*, 20(1):134–144, 2002.

[29] D. M. Dunlavy, T. G. Kolda, and E. Acar. Temporal link prediction using matrix and tensor factorizations. *ACM Transactions on Knowledge Discovery from Data (TKDD)*, 5(2):10, 2011.

[30] N. K. M. Faber, R. Bro, and P. K. Hopke. Recent developments in cande-comp/parafac algorithms: a critical review. *Chemometrics and Intelligent Laboratory Systems*, 65(1):119–137, 2003.

[31] L. Grasedyck, D. Kressner, and C. Tobler. A literature survey of low-rank tensor approximation techniques. *GAMM-Mitteilungen*, 36(1):53–78, 2013.

[32] R. A. Harshman. Foundations of the parafac procedure: models and conditions for an" explanatory" multimodal factor analysis. 1970.

[33] D. Harvey, S. Leybourne, and P. Newbold. Testing the equality of prediction mean squared errors. *International Journal of forecasting*, 13(2):281–291, 1997.

[34] R. Henrion. N-way principal component analysis theory, algorithms and applications. *Chemometrics and intelligent laboratory systems*, 25(1):1–23, 1994.

[35] N. J. Higham. Computing the nearest correlation matrixâa problem from finance. *IMA journal of Numerical Analysis*, 22(3):329–343, 2002.

[36] P. D. Hoff. Multilinear tensor regression for longitudinal relational data. *The annals of applied statistics*, 9(3):1169, 2015.

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

[38] A. J. Izenman. Reduced-rank regression for the multivariate linear model. *Journal of multivariate analysis*, 5(2):248–264, 1975.

[39] A. Kapteyn, H. Neudecker, and T. Wansbeek. An approach ton-mode components analysis. *Psychometrika*, 51(2):269–275, 1986.

[40] H. A. Kiers and I. V. Mechelen. Three-way component analysis: Principles and illustrative application. *Psychological methods*, 6(1):84, 2001.

[41] T. G. Kolda. Multilinear operators for higher-order decompositions. Technical report, Sandia National Laboratories, 2006.

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

[43] T. G. Kolda, B. W. Bader, and J. P. Kenny. Higher-order web link analysis using multilinear algebra. In *Data Mining, Fifth IEEE International Conference on*, pages 8–pp. IEEE, 2005.

[44] J. Kossaifi, Z. C. Lipton, A. Khanna, T. Furlanello, and A. Anandkumar. Tensor regression networks. *arXiv preprint arXiv:1707.08308*, 2017.

[45] P. M. Kroonenberg. *Three-mode principal component analysis: Theory and applications*, volume 2. DSWO press, 1983.

[46] P. M. Kroonenberg. *Applied multiway data analysis*, volume 702. John Wiley & Sons, 2008.

[47] P. M. Kroonenberg and J. De Leeuw. Principal component analysis of three-mode data by means of alternating least squares algorithms. *Psychometrika*, 45(1):69–97, 1980.

[48] P. M. Kroonenberg and F. J. Oort. Three-mode analysis of multimode covariance matrices. *British Journal of Mathematical and Statistical Psychology*, 56(2):305–335, 2003.

[49] M. Lassen, L. Borris, B. Anderson, H. Jensen, H. S. Bro, G. Andersen, A. Petersen, P. Siem, E. Hørlyck, B. Jensen, et al. Efficacy and safety of prolonged thromboprophylaxis with a low molecular weight heparin (dalteparin) after total hip arthroplastyâthe danish prolonged prophylaxis (dapp) study. *Thrombosis research*, 89(6):281–287, 1998.

[50] L. Li and X. Zhang. Parsimonious tensor response regression. *Journal of the American Statistical Association*, pages 1–16, 2017.

[51] X. Li, H. Zhou, and L. Li. Tucker tensor regression and neuroimaging analysis. *arXiv preprint arXiv:1304.5637*, 2013.

[52] S. Liu and G. Trenkler. Hadamard, khatri-rao, kronecker and other matrix products. *Int. J. Inf. Syst. Sci*, 4(1):160–177, 2008.

[53] H. Markowitz. Portfolio selection. *The journal of finance*, 7(1):77–91, 1952.

[54] P. McCullagh. *Tensor methods in statistics*, volume 161. Chapman and Hall London, 1987.

[55] M. Nickel. *Tensor factorization for relational learning*. PhD thesis, lmu, 2013.

[56] M. Nickel, V. Tresp, and H.-P. Kriegel. A three-way model for collective learning on multi-relational data. In *Proceedings of the 28th international conference on machine learning (ICML-11)*, pages 809–816, 2011.

[57] A. Pegg, D. H. Lockwood, and H. Williams-Ashman. Concentrations of putrescine and polyamines and their enzymic synthesis during androgen-induced prostatic growth. *Biochemical journal*, 117(1):17–31, 1970.

[58] B. Romera-Paredes, H. Aung, N. Bianchi-Berthouze, and M. Pontil. Multilinear multitask learning. In *International Conference on Machine Learning*, pages 1444–1452, 2013.

[59] M. Signoretto, R. Langone, M. Pontil, and J. Suykens. Graph based regularization for multilinear multitask learning. 2014.

[60] A. Smilde, R. Bro, and P. Geladi. *Multi-way analysis: applications in the chemical sciences*. John Wiley & Sons, 2005.

[61] L. R. Tucker. Implications of factor analysis of three-way matrices for measurement of change. *Problems in measuring change*, 122137, 1963.

[62] L. R. Tucker. The extension of factor analysis to three-dimensional matrices. *Contributions to mathematical psychology*, 110119, 1964.

[63] L. R. Tucker. Some mathematical notes on three-mode factor analysis. *Psychometrika*, 31(3):279–311, 1966.

[64] L. R. Tucker. Relations between multidimensional scaling and three-mode factor analysis. *Psychometrika*, 37(1):3–27, 1972.

[65] C. F. Van Loan. The ubiquitous kronecker product. *Journal of computational and applied mathematics*, 123(1):85–100, 2000.

[66] R. Yu, D. Cheng, and Y. Liu. Accelerated online low rank tensor learning for multivariate spatiotemporal streams. In *International Conference on Machine Learning*, pages 238–247, 2015.

DISCLAIMER - LIBERATORIA