CLARKSON UNIVERSITY

Variants of ALS on Tensor Decompositions and Applications

A dissertation
by

Na Li

Department of Mathematics

Submitted in partial fulfillment of the requirements
for the degree of
Doctor of Philosophy

(Mathematics)

2013

Accepted by the Graduate School

Date

Dean

The undersigned have examined the dissertation entitled

Dissertation's title

presented by

Author

a candidate for the degree of

Doctor of Philosophy (Mathematics),

and here by certify that it is worthy of acceptance.

Examing Committee:

Date	Professor 1 (Advisor)
Date	Professor 2
Date	Professor 3
Date	Professor 4
Date	Professor 5

Abstract

This thesis studies the numerical methods of different types of tensor decompositions and their applications.

Numerically, both the tensor CP decomposition and the Block Term Decomposition (BTD) problems can be reformed into several least-squares subproblems so that they can be solved by an algorithm called the Alternating Least-Squares (ALS) method (for BTD, we call it the BTD-ALS). However, the ALS/BTD-ALS method has a swamp phenomena which makes the convergence of the algorithm extremely slow. Moreover, this algorithm cannot guarantee to converge to a stationary point. In an attempt to overcome the issue of a swamp, this thesis studies the regularized alternating least-squares method (RALS) for solving the CP and the BTD (for BTD, we call it the BTD-RALS). An important aspect of this thesis is a convergence result of RALS showing that given the existence of stationary points of the ALS/BTD-ALS method, the limit points of the converging subsequences of the RALS/BTD-RALS are the stationary points of the original cost functional. Some numerical examples indicate a faster convergence rate for the RALS/BTD-RALS in comparison to the usual alternating least-squares method.

In addition, this thesis studies the decompositions for a partially symmetric tensor, called Symmetric Outer Product Decomposition (SOPD) and proposed a Partial Column-Wise Alternating Least-Squares (PCW-ALS) method for solving such decomposition. It has been shown that PCW-ALS is better than the usual ALS method in

terms of the number of iterations and CPU times.

A specific type of BTD, three-way receptor model, is studied in this thesis. The non-uniqueness of the receptor model is shown based on a new formulation. In the application of the receptor model, an variant of ALS method called Weighted Alternating Least-Squares (WALS) is introduced.

Two applications of tensor decompositions are provided. One is the application of the receptor model on an air sampled dataset to identify the different factor sources at the Washington-Dulles international airport area. The other application is CP decomposition on the signal processing. It studies a two-channel mixture waveforms of several sound sources and identify these different sound sources.

Acknowledgments

This dissertation is made possible through the help and support from everyone, including: parents, teachers, family, friends, and in essence, all sentient beings. Especially, please allow me to dedicate my acknowledgment of gratitude toward the following significant advisors and contributors:

I would like to express the deepest appreciation to my advisor, Professor Carmeliza Navasca, who guided my every step in my PhD study for these four years. She has a spirit of adventure in regard to research and an excitement in regard to teaching. She is also very patient and conscientious. Without her guidance and persistent help this dissertation would not have been possible.

I would like to thank my committee member, Professor Philip K. Hopke, for his understanding and assistance. He taught me a lot and provided me many help in the studying of our project and paper publication.

I wish to thank my committee member, Professor Stefan Kindermann, who provided me a lot of very useful ideas in our Skype meetings. We also had very fruitful discussions on one of our project.

I would like to thank my committee member, Professor Scott R. Fulton for his most support and encouragement. His words really helped me and made me more confident.

I also wish to thank my committee member, Professor Sumona Mondal, who provides me a lot of help and valuable advice during my PhD study.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Matrix decompositions have always had an extremely important role in the signal, circuit and system theory. In particular, the Singular Value Decomposition (SVD) has been applied in image compression, signal processing, modal analysis and many other scientific fields [79] . For example, in the study of data from food industry, the standard multivariate data analysis arranges those data in a two-way structure [16], where each row of a data matrix denotes a sample and each column expresses the absorbance at a particular wavelength. Also in studying of air sample in Chemometrics, the data is arranged in a matrix structure [102], where each row is a sample time and each column denotes one chemical element. However, data can be much more complex. For instance, sometimes in chemometics, each sample of the fluorescence emission is determined at several wavelengths for several different excitation wavelengths. In this case, the data need one more index. Therefore, it is natural to think about multi-way arrays which we call tensors. Tensors can be considered as a generalization of matrices. Similar with the popularity of matrix decomposition, the study of tensor decomposition becomes an important topic since it can be applied to data arrays for extracting and explaining their properties [70].

Hitchcock in 1927 first proposed tensor decomposition [48] and [54], which is also the origin of CANDECOMP/PARAFAC (CP) decomposition. Tensor decomposi-

tion was a popular method until Tucker proposed Tucker decomposition in the 1960s [123, 124]. In 1970, due to the work of Harshman[49], Carroll and Chang [17], Hitchcock's idea was named CANDECOMP/PARAFAC (CP). Recently, in 2008, De Lathauwer [28, 29, 36] proposed a more general tensor decomposition model which can be considered a framework that unifies the CP and Tucker decomposition. The author calls it Block Term Decomposition (BTD). The first two tensor decompositions (CP and Tucker decompositions) have become the most basic and popular factorizations and have been applied in chemometrics [4, 102, 57, 43, 52, 128] and signal processing [36, 31, 32, 26, 82]. In the last decade, the tensor decomposition has attracted more scientific field including numerical linear algebra [71, 34, 33], data mining[117, 85] and more. Moreover, several books about multiway data have appeared [73, 72]. The latter one [72] mainly deals with the third-order tensors (three-way data) while [73] focuses on multiway data applications. There are also several software packages available for working with tensors and their decompositions [2, 7, 6].

CP decomposition is a natural generalization of matrix singular value decomposition (SVD). However, the CP does not have similar properties to matrix SVD. One issue with CP is that the determination of the rank of a tensor is NP-hard [51]. Several papers discuss tensor rank and propose new definitions like maximum rank and typical rank to describe the rank properties of tensor [77, 62, 118, 63]. Also the uniqueness of CP decomposition is more complicated, and has become a popular research area. Kruskal [75, 77] provides a sufficient condition for third-order tensor uniqueness, and his results were followed by many general extended results. See papers [27, 61, 116, 114, 115]. The extension of the Eckart-Young theorem is also an interesting topic [38] since the best rank-r approximation problem for higher-order tensors is a problem of central importance in the statistical analysis of multiway data [16, 33, 34].

Another research area is studying the numerical methods for the CP decomposi-

tion. Currently, most numerical methods for CP begin with a guess for the rank R since the determination of tensor rank is NP-hard [51]. Basically, starting from a small R, we keep adding one until we find a perfect fit. This method does cause problems in practice relating to the degeneracy problem [95]. Among those CP decomposition algorithms, the alternating least-squares (ALS) [49, 17] method is a popular and convenient one. But it can take many iterations to converge, such a phenomena is called the swamp. Moreover, the ALS method is not guaranteed to converge to a global minimum or even a stationary point. Several papers discuss this problem and propose some methods to improve the efficiency of ALS [106, 92, 91], where [91] deals with non-degeneracy problem and is an regularized method which can remove some swamps.

The fully and partially symmetric tensors have a wide application to the field of signal processing [19] and the problem of decomposing a fully symmetric tensor into a sum of a number of rank-one fully symmetric tensors is related to the independent component analysis (ICA) [20], [60]. Therefore, such symmetric tensor decomposition has been studied in [21, 14, 31, 69]. However, these methods are not robust, and they cannot apply to all kinds of symmetric tensors. The method proposed in [31] is only for fourth-order fully symmetric tensor called Fourth-Order-Only Blind Identification (FOOBI) and it is based on Joint Diagonalization algorithm. The Higher-Order Power Method (HOPM) proposed in [69] is only for rank-one approximation of a fully symmetric tensor and it cannot generalize to rank-R approximation.

Tucker decomposition factors a tensor into a core tensor of the same order with a matrix multiplied along each mode. In 2000, De Lathauwer, De Moor and Vandewalle [33] constrained that all the factor matrices must be orthogonal, and they called it the Higher-Order Singular Value Decomposition (HOSVD). They also show that HOSVD exists for any higher-order tensor and they provided a convenient method to solve it. In some applications, a reduced core tensor (the size is much smaller than the

original tensor) is always expected, so some methods appear for solving a truncated HOSVD [74, 34]. The latter paper discusses a method called Higher-Order Orthogonal Iteration (HOOI) which is faster and simpler.

Several researchers study the BTD model such as De Almeida, Favier and Mota [25], who give an overview of BTD models and its applications to problems in blind beamforming and multiantenna coding. De Lathauwer explores the BTD, concludes that BTD is a general framework that unifies the CP and Tucker decompositions. In addition, he proposes a numerical method to compute the tensor block term decomposition [28, 29, 36]. This method is also based on the alternating least-squares algorithm. He further studies the uniqueness properties of BTD which can be considered a generalization of Krusdal's theorem. Some applications in wireless communications are in [36]. Separately, in the study of source of air pollutants in chemometrics, a three-way receptor model [102] is proposed. The block term decomposition in rank-$(L, L, 1)$ is actually a special case of the receptor model. A different type of method called Multilinear Engine (ME) is used to solve the receptor model in [102], which treats the original problem as a nonlinear optimization.

Chapter 2

Introduction to tensors

1 Tensor definition and basics

We follow the survey paper [70] and De Lathauwer's papers [26, 27, 33] to provide the following definitions. The definition of tensor refers to [38].

Definition 1.1 (Nth-order tensor) *Let $\mathbb{R}^{I_1}, \mathbb{R}^{I_2}, \ldots, \mathbb{R}^{I_N}$ be N vector spaces, each of which has its own coordinate system, and let the vector $\mathbf{x}_i \in \mathbb{R}^{I_i}$, $i = 1, 2, \ldots, N$. There is a multilinear map ϕ on the space $\mathbb{R}^{I_1} \times \mathbb{R}^{I_2} \times \cdots \times \mathbb{R}^{I_N}$ defined by*

$$\phi(\mathbf{x}_1, \ldots, \mathbf{x}_N) \mapsto [x_{i_1}^{(1)} \cdots x_{i_N}^{(N)}]_{i_1,\cdots,i_N=1}^{I_1,\ldots,I_N},$$

where $x_{i_r}^{(r)}$ means the i_rth element of the rth vector $\mathbf{x}_r$.

We use $\mathcal{X}$ to denote the right hand of the above map definition and the element of $\mathcal{X}$ is $x_{i_1 i_2 \ldots i_N} = x_{i_1}^{(1)} \cdots x_{i_N}^{(N)}$. So $\mathcal{X}$ is in $\mathbb{R}^{I_1 \times I_2 \times \cdots \times I_N}$ and is called a Nth-order tensor.

This is a general definition in multilinear algebra. Here, we can consider a Nth-order tensor as a N dimensional array or N-way array [27, 70]. So, if all the above vector spaces are $\mathbb{R}^{I_n}$, then a Nth-order tensor $\mathcal{X}$ is a N-way real-valued array, i.e., $\mathcal{X} \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_N}$. If the above vector spaces are $\mathbb{C}^{I_n}$, $\mathcal{X} \in \mathbb{C}^{I_1 \times I_2 \times \cdots \times I_N}$ is a complex-

valued tensor. Through this thesis, a Nth-order tensor $\mathcal{X}$ is always real-valued except when we specify it as a complex tensor.

Example 1.2 A zero-order tensor is a scalar, a first-order tensor is a vector and a second-order tensor is a matrix. A third-order tensor $\mathcal{T} \in \mathbb{R}^{I \times J \times K}$ has three indices, as shown in Figure 2.1. The element of $\mathcal{T}$ is $t_{ijk} \in \mathcal{T}$, $i = 1, 2, \ldots, I$; $j = 1, 2, \ldots, J$ and $k = 1, 2, \ldots, K$.

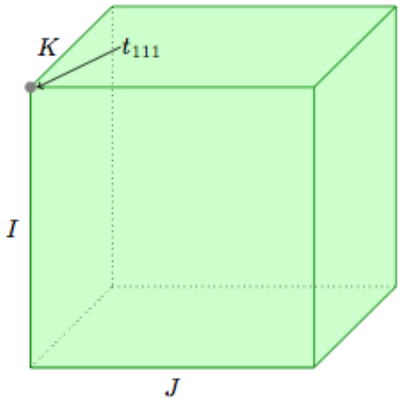


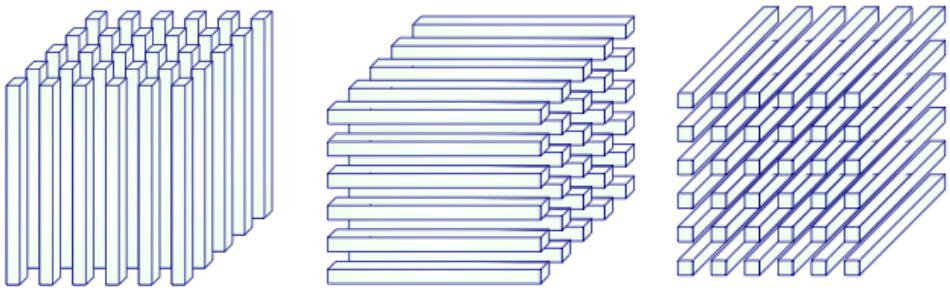
Figure 2.1: **A third-order tensor:** $\mathcal{T} \in \mathbb{R}^{I \times J \times K}$

Definition 1.3 (Mode-n fibers) *A mode-n fiber of a Nth-order tensor is a vector defined by fixing every index but the nth one. So, for a Nth-order tensor* $\mathcal{T} \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_N}$*, the mode-n fiber can be denoted as* $\mathbf{t}_{\mathbf{i_1} \cdots \mathbf{i_{n-1}} : \mathbf{i_{n+1}} \cdots \mathbf{i_N}}$.

The following Figure 2.2 is the different mode fibers for the third-order tensor.

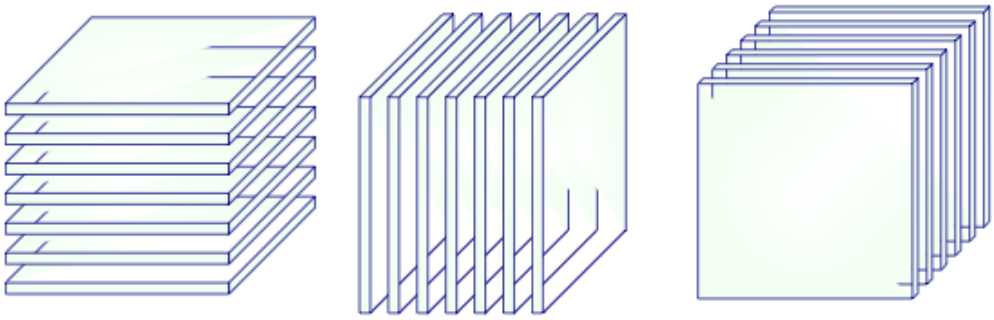
Example 1.4 For a matrix, the columns are the mode-1 fibers and the rows are the mode-2 fibers.

Definition 1.5 (Slices) *A slice of a Nth-order tensor is a two-dimensional section defined by fixing all but two indices. If the unfixed indices are* i_t *and* i_s*, then the corresponding slice can be expressed as* $\mathbf{T}_{\mathbf{i_1} \cdots \mathbf{i_{t-1}} : \mathbf{i_{t+1}} \cdots \mathbf{i_{s-1}} : \mathbf{i_{s+1}} \cdots \mathbf{i_N}}$.



(a) The mode-1 fibers: $\mathbf{t}_{:\mathbf{jk}}$ (b) The mode-2 fibers: $\mathbf{t}_{\mathbf{i}:\mathbf{k}}$ (c) The mode-3 fibers: $\mathbf{t}_{\mathbf{ij}:}$

Figure 2.2: **The three mode-n fibers of a third-order tensor $\mathcal{T} \in \mathbb{R}^{I \times J \times K}$**



(a) The horizontal slices: $\mathbf{T}_{\mathbf{i}::}$ (b) The lateral slices: $\mathbf{T}_{:\mathbf{j}:}$ (c) The frontal slices: $\mathbf{T}_{::\mathbf{k}}$

Figure 2.3: **The three different slices of a third-order tensor $\mathcal{T} \in \mathbb{R}^{I \times J \times K}$**

The Figure 2.3 shows the three mode slices ($\mathbf{T}_{\mathbf{i}::}$, $\mathbf{T}_{:\mathbf{j}:}$, $\mathbf{T}_{::\mathbf{k}}$) for the third-order tensor, we also call them horizontal, lateral and frontal slices respectively.

Definition 1.6 (Mode-n matricization: transforming a tensor into a matrix) *[70] Matricization (or Unfolding) is the process of reordering the elements of an Nth-order tensor into a matrix. The mode-n matricization of a tensor $\mathcal{T} \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_N}$ is denoted by $\mathbf{T}_{(\mathbf{n})}$ and arranges the mode-n fibers to be the columns of the resulting matrix. The dimension of $\mathbf{T}_{(\mathbf{n})}$ is $\mathbb{R}^{I_n \times J}$, where $J = I_1 \cdots I_{n-1} I_{n+1} \cdots I_N$.*

If we use a map to express such matricization process for any Nth-order tensor $\mathcal{T} \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_N}$, that is, the tensor element $(i_1, i_2, \ldots, i_N)$ maps to matrix element

(i_n, j), then there is a formula [70] to calculate j:

$$j = 1 + \sum_{\substack{k=1 \\ k \neq n}}^{N} (i_k - 1) J_k \quad \text{with} \quad J_k = \prod_{\substack{m=1 \\ m \neq n}}^{k-1} I_m.$$

So, given a third-order tensor $\mathcal{T} \in \mathbb{R}^{I \times J \times K}$, the mode-1, mode-2 and mode-3 matricizations of $\mathcal{T}$, respectively, are:

$$\begin{aligned}
\mathbf{T}_{(\mathbf{1})} &= [\mathbf{t}_{:\mathbf{11}}, \ldots, \mathbf{t}_{:\mathbf{J1}}, \mathbf{t}_{:\mathbf{12}} \ldots, \mathbf{t}_{:\mathbf{J2}}, \ldots, \mathbf{t}_{:\mathbf{1K}}, \ldots, \mathbf{t}_{:\mathbf{JK}}], \\
\mathbf{T}_{(\mathbf{2})} &= [\mathbf{t}_{\mathbf{1}:\mathbf{1}}, \ldots, \mathbf{t}_{\mathbf{I}:\mathbf{1}}, \mathbf{t}_{\mathbf{1}:\mathbf{2}} \ldots, \mathbf{t}_{\mathbf{I}:\mathbf{2}}, \ldots, \mathbf{t}_{\mathbf{1}:\mathbf{K}}, \ldots, \mathbf{t}_{\mathbf{I}:\mathbf{K}}], \\
\mathbf{T}_{(\mathbf{3})} &= [\mathbf{t}_{\mathbf{11}:}, \ldots, \mathbf{t}_{\mathbf{I1}:}, \mathbf{t}_{\mathbf{12}:} \ldots, \mathbf{t}_{\mathbf{I2}:}, \ldots, \mathbf{t}_{\mathbf{1J}:}, \ldots, \mathbf{t}_{\mathbf{IJ}:}].
\end{aligned}$$

Example 1.7 Given a third-order tensor $\mathcal{X} \in \mathbb{R}^{3 \times 3 \times 2}$. The frontal slices of $\mathcal{X}$ are

$$\mathbf{X}_{::\mathbf{1}} = \begin{bmatrix} 1 & 3 & 5 \\ 2 & 4 & 6 \\ 7 & 9 & 11 \end{bmatrix}, \qquad \mathbf{X}_{::\mathbf{2}} = \begin{bmatrix} 13 & 15 & 17 \\ 14 & 16 & 18 \\ 19 & 21 & 23 \end{bmatrix}.$$

So the three mode-n unfoldings are

$$\begin{aligned}
\mathbf{X}_{(\mathbf{1})} &= \begin{bmatrix} 1 & 3 & 5 & 13 & 15 & 17 \\ 2 & 4 & 6 & 14 & 16 & 18 \\ 7 & 9 & 11 & 19 & 21 & 23 \end{bmatrix}, \\
\mathbf{X}_{(\mathbf{2})} &= \begin{bmatrix} 1 & 2 & 7 & 13 & 14 & 19 \\ 3 & 4 & 9 & 15 & 16 & 21 \\ 5 & 6 & 11 & 17 & 18 & 23 \end{bmatrix}, \\
\mathbf{X}_{(\mathbf{3})} &= \begin{bmatrix} 1 & 2 & 7 & 3 & 4 & 9 & 5 & 6 & 11 \\ 13 & 14 & 19 & 15 & 16 & 21 & 17 & 18 & 23 \end{bmatrix}.
\end{aligned}$$

Other ordering exists based on permutation of mode-n unfolding, see [33] and [66]. An alternative way to define the mode-n unfolding is provided in [33], which is

Definition 1.8 (Mode-n matricization) *[33] Assume an Nth-order tensor $\mathcal{T} \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_N}$. The mode-n matricization (unfolding) $\mathbf{T}_{(\mathbf{n})} \in \mathbb{R}^{I_n \times (I_1 I_2 \ldots I_{n-1} I_{n+1} \ldots I_N)}$ contains the element $t_{i_1 i_2 \ldots i_N}$ at the position with row number i_n and column number j, where*

$$j = 1 + \sum_{k=1}^{n-1} (i_k - 1) J_k + \sum_{l=n+1}^{N} (i_l - 1) \widetilde{J}_l, \quad \text{where } J_k = \prod_{m=k+1}^{n-1} I_m, \widetilde{J}_l = \prod_{\substack{m=1 \\ m \neq n}}^{N} I_m.$$

For the same example 1.7, the three mode-n unfolding should be

$$\mathbf{X}_{(\mathbf{1})} = \begin{bmatrix} 1 & 13 & 3 & 15 & 5 & 17 \\ 2 & 14 & 4 & 16 & 6 & 18 \\ 7 & 19 & 9 & 21 & 11 & 23 \end{bmatrix},$$

$$\mathbf{X}_{(\mathbf{2})} = \begin{bmatrix} 1 & 2 & 7 & 13 & 14 & 19 \\ 3 & 4 & 9 & 15 & 16 & 21 \\ 5 & 6 & 11 & 17 & 18 & 23 \end{bmatrix},$$

$$\mathbf{X}_{(\mathbf{3})} = \begin{bmatrix} 1 & 3 & 5 & 2 & 4 & 6 & 7 & 9 & 11 \\ 13 & 15 & 17 & 14 & 16 & 18 & 19 & 21 & 23 \end{bmatrix}.$$

From the above example, we can see that the difference between these two is just the order of the fibers of the tensor. In general, this difference in definitions 1.8 and 1.6 does not affect the calculation as long as the same one is used during the process of calculation. In the following chapters, if we do not mention, the definition1.6 is used. We will switch to the definition 1.8 in the study of third-order receptor model and we will mention it in that section. Moreover, sometimes we may need to vectorize a tensor. Once again, the order of the arrays is not important as long as it is consistent with its calculation.

Definition 1.9 (Vectorization) *The vectorization of a matrix*

$$\mathbf{M} = [\mathbf{m_1}, \mathbf{m_2}, \cdots, \mathbf{m_n}] \in \mathbb{R}^{m \times n},$$

where $\mathbf{m_i}$ *is the ith column of* $\mathbf{M}$*, is denoted by vec*(M) *which is a vector of size mn defined by*

$$vec(\mathbf{M}) = \begin{bmatrix} \mathbf{m_1} \\ \mathbf{m_2} \\ \vdots \\ \mathbf{m_n} \end{bmatrix}.$$

So, the vectorization of a tensor $\mathcal{T} \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_N}$ *is denoted by vec*$(\mathcal{T})$*. We define it by vectorizing its mode-1 unfolding,*

$$vec(\mathcal{T}) = vec(\mathbf{T_{(1)}}).$$

Definition 1.10 (unvec) *Given a vector* $\mathbf{v} \in \mathbb{R}^{I^2}$*, then unvec*$(\mathbf{v})$ *is a matrix of size* $I \times I$*. It is defined by dividing the vector* $\mathbf{v}$ *into* I *smaller vectors and each of them is length* I*, and the ith pieces is the ith column of resulting matrix.*

Definition 1.11 (ten) *Given a vector* $\mathbf{v} \in \mathbb{R}^{I^3}$*, then ten*$(\mathbf{v})$ *is a tensor of size* $I \times I \times I$*. It is defined as follows: by dividing the vector* $\mathbf{v}$ *into* I *vectors* $\mathbf{v}_i$ *(each one is a smaller vector of length* I^2*),* $i = 1, 2, \ldots, I$*, the ith frontal slide of resulting tensor is*

$$ten(\mathbf{v})(:, :, i) = unvec(\mathbf{v}_i).$$

Definition 1.12 (Frobenius-norm) *The Frobenius norm of a tensor* $\mathcal{T} \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_N}$ *is the square root of the sum of the squares of all its elements. The formula is*

$$\|\mathcal{T}\|_F = \sqrt{\sum_{i_1=1}^{I_1} \sum_{i_2=1}^{I_2} \cdots \sum_{i_N=1}^{I_N} t_{i_1 i_2 \cdots i_N}^2}.$$

This is analogous to the matrix Frobenius norm. So far, it is the only norm we can very easily to generalize from the matrix case. There are no uniform definitions of tensor's eigenvalues, eigenvectors or tensor's 1-norm, 2-norm, infinite-norm. Lim [83] (also see Qi [105]) discusses the theory of eigenvalues, eigenvectors for higher-order tensors. Brazell, Li, Navasca and Tamon in [15] also provide the eigenvalue decomposition (EVD) for a forth-order symmetric tensor.

2 Basic tensor operations

The tensor space $\mathbb{R}^{I_1 \times I_2 \times \cdots \times I_N}$ is a vector space. So for any two tensors $\mathcal{A}, \mathcal{B} \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_N}$ and $\lambda \in \mathbb{R}$, we have

$$\begin{aligned} \mathcal{A} + \mathcal{B} = \mathcal{C} \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_N} \quad \text{where} \;\; c_{i_1 i_2 \ldots i_N} &:= a_{i_1 i_2 \ldots i_N} + b_{i_1 i_2 \ldots i_N} \\ \lambda \cdot \mathcal{A} = \mathcal{D} \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_N} \quad \text{where} \;\; d_{i_1 i_2 \ldots i_N} &:= \lambda \cdot a_{i_1 i_2 \ldots i_N} + b_{i_1 i_2 \ldots i_N}. \end{aligned}$$

Different from matrix, tensor does not have inversion in the common sense. However, under some specific product on some tensors, we may define the tensor inversion and the identity tensor so that we can obtain a tensor group. For example, Brazell et al. [15] defines a tensor group on all the fourth-order tensor. But in general, the tensor inversion is an open question.

Definition 2.1 (Inner product) *The inner product of two same-sized tensors* $\mathcal{T}, \mathcal{W} \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_N}$ *is the sum of the products of their elements, i.e.,*

$$\langle \mathcal{T}, \mathcal{W} \rangle = \sum_{i_1=1}^{I_1} \sum_{i_2=1}^{I_2} \cdots \sum_{i_N=1}^{I_N} t_{i_1 i_2 \ldots i_N} w_{i_1 i_2 \ldots i_N}.$$

Example 2.2 According to the definition of Frobenius norm of a tensor, we have

$$\langle \mathcal{T}, \mathcal{T} \rangle = \|\mathcal{T}\|_F^2.$$

Definition 2.3 (Outer product) *The outer product of two tensors* $\mathcal{T} \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_N}$ *and* $\mathcal{W} \in \mathbb{R}^{J_1 \times J_2 \times \cdots \times J_M}$ *is a* $(N+M)$*th-tensor denoted by* $\mathcal{T} \circ \mathcal{W}$*, and the element of* $\mathcal{T} \circ \mathcal{W}$ *is defined by*

$$(\mathcal{T} \circ \mathcal{W})_{i_1 i_2 \ldots i_N j_1 j_2 \ldots j_M} = t_{i_1 i_2 \ldots i_N} w_{j_1 j_2 \ldots j_M},$$

for $i_k = 1, 2, \ldots, I_k$*,* $k = 1, 2, \ldots, N$ *and* $j_s = 1, 2, \ldots, J_s$*,* $s = 1, 2, \ldots, M$*.*

Notice that if we take $\mathcal{T}$ *and* $\mathcal{W}$ *as two vectors (first-order tensors), the outer product of* $\mathcal{T}$ *and* $\mathcal{W}$ *(as two tensors) is consistent with the outer product of* $\mathcal{T}$ *and* $\mathcal{W}$ *as two vectors.*

Example 2.4 Given N vectors: $\mathbf{x}^{(1)} \in \mathbb{R}^{I_1}$, $\mathbf{x}^{(2)} \in \mathbb{R}^{I_2}$, $\ldots$, $\mathbf{x}^{(N)} \in \mathbb{R}^{I_N}$, we can generate an Nth-order tensor $\mathcal{X}$ by using outer product of these vectors, so the element $x_{i_1 i_2 \ldots i_N}$ of $\mathcal{X}$ is

$$x_{i_1 i_2 \ldots i_N} = \mathbf{x}^{(1)}_{i_1} \mathbf{x}^{(2)}_{i_2} \cdots \mathbf{x}^{(N)}_{i_N}.$$

And the tensor $\mathcal{X}$ is

$$\mathcal{X} = \mathbf{x}^{(1)} \circ \mathbf{x}^{(2)} \circ \cdots \circ \mathbf{x}^{(N)}.$$

Definition 2.5 (Contraction) *[26] The contraction* $\langle \mathcal{T} \rangle_{p,q}$ *of a tensor* $\mathcal{T} \in R^{I_1 \times I_2 \times \cdots \times I_N}$*, over the indices* i_p *and* i_q *(*$I_p = I_q$*), is a* $(N-2)$*th-order tensor and is defined by*

$$(\langle \mathcal{T} \rangle)_{i_1 i_2 \ldots i_{p-1} i_{p+1} \ldots i_{q-1} i_{q+1} \ldots i_N} = \sum_{r=1}^{I_p = I_q} t_{i_1 i_2 \ldots i_{p-1} r i_{p+1} \ldots i_{q-1} r i_{q+1} \ldots, i_N}.$$

More generally, it is possible to define contractions over several indices.

3 Tensor multiplication

Tensor multiplication is much more complex than the matrix multiplication since each tensor has many indices. Bader and Kolda [7] discuss the different situations of tensor multiplication. Here, we can consider only the Tucker-n product (n-mode product) which defines the multiplication between a tensor and a matrix (or a vector).

Definition 3.1 (Multiplication of a higher-order tensor by a matrix) *The n-mode product of a tensor $\mathcal{T} \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_N}$ and a matrix $\mathbf{M} \in \mathbb{R}^{J \times I_n}$, denoted by $\mathcal{T} \times_n \mathbf{M}$, is an Nth-order tensor $\mathcal{Y}$ of size $(I_1 \times I_2 \times \cdots \times I_{n-1} \times J \times I_{n+1} \cdots \times I_N)$. The element of the new tensor is*

$$(\mathcal{T} \times_n \mathbf{M})_{i_1 i_2 \ldots j \ldots i_N} = \sum_{i_n=1}^{I_n} t_{i_1 i_2 \ldots i_n \ldots i_N} m_{j i_n}.$$

By using mode-n matricization, we can get

$$\mathcal{Y} = \mathcal{T} \times_n \mathbf{M} \Leftrightarrow \mathbf{Y}_{(\mathbf{n})} = \mathbf{M}\mathbf{T}_{(\mathbf{n})}.$$

Example 3.2 Let $\mathcal{X} \in \mathbb{R}^{3 \times 3 \times 2}$ be the tensor defined in example 1.7 and let $\mathbf{M} = \begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \end{bmatrix}$, then the 2-mode product $\mathcal{Y} = \mathcal{X} \times_2 \mathbf{M}$ can be calculated, which is

$$\mathbf{Y}_{::\mathbf{1}} = \begin{bmatrix} 22 & 49 \\ 28 & 64 \\ 58 & 139 \end{bmatrix}, \qquad \mathbf{Y}_{::\mathbf{2}} = \begin{bmatrix} 94 & 229 \\ 100 & 244 \\ 130 & 319 \end{bmatrix}.$$

There are two properties about the n-mode product, which can be expressed as following:

Proposition 3.3 *[26] Given the tensor $\mathcal{X} \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_N}$ and the matrices $\mathbf{M} \in$*

$\mathbb{R}^{J_m \times I_m}$, $\mathbf{N} \in \mathbb{R}^{J_n \times I_n}$, *so we have:*

$$\mathcal{X} \times_m \mathbf{M} \times_n \mathbf{N} = \mathcal{X} \times_n \mathbf{N} \times_m \mathbf{M} \quad \textit{when } m \neq n.$$

Proposition 3.4 *[26] Given a Nth-order tensor* $\mathcal{X} \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_N}$ *and the matrices* $\mathbf{M} \in \mathbb{R}^{J_n \times I_n}$, $\mathbf{N} \in \mathbb{R}^{K_n \times J_n}$, *then we have:*

$$\mathcal{X} \times_n \mathbf{M} \times_n \mathbf{N} = \mathcal{X} \times_n (\mathbf{NM}).$$

Actually, we can use the n-mode product to describe the matrix multiplication. For the matrices $\mathbf{A} \in \mathbb{R}^{I \times J}$, $\mathbf{B} \in \mathbb{R}^{K \times J}$ and $\mathbf{C} \in \mathbb{R}^{L \times I}$, then we have

$$\begin{aligned} \mathbf{AB}^{\mathrm{T}} &= \mathbf{A} \times_2 \mathbf{B} \\ \mathbf{CA} &= \mathbf{A} \times_1 \mathbf{C}. \end{aligned}$$

Therefore, we have

$$\mathbf{CAB}^{\mathrm{T}} = \mathbf{A} \times_1 \mathbf{C} \times_2 \mathbf{B}. \tag{3.1}$$

De Lathauwer [26] explains the above equation (3.1) as the columns of $\mathbf{C}$ are associated to the "1-mode space" of $\mathbf{A}$; in exactly the same way the columns of $\mathbf{B}$ are associated to the "2-mode space".

Definition 3.5 (Multiplication of a higher-order tensor by a vector) *[70] The n-mode vector product of a tensor* $\mathcal{T} \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_N}$ *with a vector* $\mathbf{v} \in \mathbb{R}^{I_n}$ *is denoted by* $\mathcal{T} \bar{\times}_n \mathbf{v}$. *It generates an* $(N-1)$*th-order tensor and the size is* $I_1 \times \cdots \times I_{n-1} \times I_{n+1} \cdots I_N$. *The element of the product can be calculated by*

$$(\mathcal{T} \bar{\times}_n \mathbf{v})_{i_1 \ldots i_{n-1} i_{n+1} \ldots i_N} = \sum_{i_n=1}^{I_n} t_{i_1 i_2 \ldots i_N} v_{i_n}.$$

Example 3.6 For the tensor in example 1.7, we give a vector $\mathbf{v} = [1 \quad 2 \quad 3]^{\mathrm{T}}$, so we have

$$\mathcal{X} \bar{\times}_1 \mathbf{v} = \begin{bmatrix} 26 & 98 \\ 38 & 110 \\ 50 & 122 \end{bmatrix}, \qquad \mathcal{X} \bar{\times}_2 \mathbf{v} = \begin{bmatrix} 22 & 94 \\ 28 & 100 \\ 58 & 130 \end{bmatrix}.$$

Since the n-mode vector product changes the order of the tensor, the following property is different from the matrix case.

Proposition 3.7 *Given a Nth-order tensor $\mathcal{X} \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_N}$ and two vectors $\mathbf{v} \in \mathbb{R}^{I_n}$, $\mathbf{w} \in \mathbb{R}^{I_m}$ and $n < m$, so we have*

$$\mathcal{X} \bar{\times}_n \mathbf{v} \bar{\times}_m \mathbf{w} = (\mathcal{X} \bar{\times}_n \mathbf{v}) \bar{\times}_{m-1} \mathbf{w} = (\mathcal{X} \bar{\times}_m \mathbf{w}) \bar{\times}_n \mathbf{v}.$$

Definition 3.8 (Matrix Kronecker Product) *The Kronecker product of two matrices $\mathbf{A}$ and $\mathbf{B}$ is defined as*

$$\mathbf{A} \otimes \mathbf{B} = \begin{bmatrix} a_{11}\mathbf{B} & a_{12}\mathbf{B} & \dots \\ a_{21}\mathbf{B} & a_{22}\mathbf{B} & \dots \\ \vdots & \vdots & \end{bmatrix}.$$

Definition 3.9 (Matrix Khatri-Rao Product) *The Khatri-Rao product [107] is the "matching columnwise" Kronecker product. Given matrices $\mathbf{A} \in \mathbb{R}^{I \times K}$ and $\mathbf{B} \in \mathbb{R}^{J \times K}$, their Khatri-Rao product is denoted by $\mathbf{A} \odot \mathbf{B}$. The result is a matrix of size $(IJ \times K)$ defined by*

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

If $\mathbf{a}$ and $\mathbf{b}$ are vectors, then the Khatri-Rao and Kronecker products are identical, i.e., $\mathbf{a} \otimes \mathbf{b} = \mathbf{a} \odot \mathbf{b}$.

These matrix products have some properties (see [86, 107]). We provide some in the following,

Proposition 3.10 (The mixed-product property) *If* $\mathbf{A}, \mathbf{B}, \mathbf{C}$ *and* $\mathbf{D}$ *are matrices of such size that one can form the matrix products* $\mathbf{AC}$ *and* $\mathbf{BD}$*, then*

$$(\mathbf{A} \otimes \mathbf{B})(\mathbf{C} \otimes \mathbf{D}) = \mathbf{AC} \otimes \mathbf{BD}.$$

It follows that $\mathbf{A} \otimes \mathbf{B}$ *is invertible if and only is* $\mathbf{A}$ *and* $\mathbf{B}$ *are invertible. So the inversion of* $\mathbf{A} \otimes \mathbf{B}$ *is*

$$(\mathbf{A} \otimes \mathbf{B})^{-1} = \mathbf{A}^{-1} \otimes \mathbf{B}^{-1}.$$

Proposition 3.11 (Associative) *Given three matrices* $\mathbf{A} \in \mathbb{R}^{I_1 \times J_1}$, $\mathbf{B} \in \mathbb{R}^{I_2 \times J_2}$ *and* $\mathbf{C} \in \mathbb{R}^{I_3 \times J_3}$*, we have*

$$(\mathbf{A} \otimes \mathbf{B}) \otimes \mathbf{C} = \mathbf{A} \otimes (\mathbf{B} \otimes \mathbf{C}).$$

If $J_1 = J_2 = J_3$*, then*

$$(\mathbf{A} \odot \mathbf{B}) \odot \mathbf{C} = \mathbf{A} \odot (\mathbf{B} \odot \mathbf{C}).$$

Example 3.12 Let $\mathcal{X} \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_N}$ and $\mathbf{A}^{(n)} \in \mathbb{R}^{J_n \times I_n}$ for all $n = 1, 2, \ldots, N$. Then we have

$$\begin{aligned} \mathcal{Y} &= \mathcal{X} \times_1 \mathbf{A}^{(1)} \times_2 \mathbf{A}^{(2)} \times_3 \cdots \times_N \mathbf{A}^{(N)} \\ \Leftrightarrow \mathbf{Y}_{(\mathbf{n})} &= \mathbf{A}^{(n)} \mathbf{X}_{(\mathbf{n})} (\mathbf{A}^{(N)} \otimes \cdots \otimes \mathbf{A}^{(n+1)} \otimes \mathbf{A}^{(n-1)} \otimes \cdots \otimes \mathbf{A}^{(1)})^{\mathrm{T}}. \end{aligned}$$

The proof of this equation is in [71].

Chapter 3

Basic tensor decomposition

The idea of tensor decomposition is basically breaking down a tensor into several simple components. Therefore, we start this chapter from talking about the specific simple tensors.

1 Simple structure

1.1 Rank-one tensors

Definition 1.1 (Rank-one tensor) *A Nth-order tensor $\mathcal{T} \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_N}$ is a rank-one if it can be written as the outer product of N vectors (see Example 2.4 in Chap. 2), i.e.,*

$$\mathcal{T} = \mathbf{a}^{(1)} \circ \mathbf{a}^{(2)} \circ \cdots \circ \mathbf{a}^{(N)},$$

where $\mathbf{a}^{(r)} \in \mathbb{R}^{I_r}$, $1 \leq r \leq N$.

This definition is also a direct generalization of rank-one matrix. The Figure 3.1 illustrates a third-order rank-one tensor $\mathcal{X} = \mathbf{a} \circ \mathbf{b} \circ \mathbf{c}$.

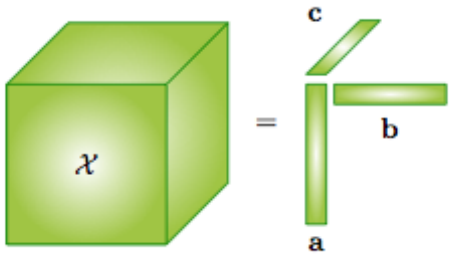


Figure 3.1: **Rank-one third-order tensor $\mathcal{X} = \mathbf{a} \circ \mathbf{b} \circ \mathbf{c}$. The element is $x_{ijk} = a_i b_j c_k$.**

1.2 Symmetric tensors

Definition 1.2 (Fully symmetric tensor) *A Nth-order tensor $\mathcal{T} \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_N}$ is called cubical if every mode is the same size, i.e., $I_1 = I_2 = \cdots = I_N = I$ [21]. A cubical tensor is fully symmetric if it is invariant under any permutation of its indices. Thus for a permutation σ of the symbols $\{i_1, i_2, \ldots, i_N\}$, the elements $t_{i_1 i_2 \ldots i_N}$ of the tensor $\mathcal{T}$ have the following property,*

$$t_{i_1 i_2 \ldots i_N} = t_{\sigma(i_1)\sigma(i_2)\ldots\sigma(i_N)}. \tag{1.1}$$

For example, a third-order tensor $\mathcal{X} \in \mathbb{R}^{I \times I \times I}$ is symmetric if

$$x_{ijk} = x_{ikj} = x_{jik} = x_{kij} = x_{jki} = x_{kji} \quad \text{for all} \quad i, j, k = 1, \ldots, I.$$

For some tensors, the elements cannot satisfy the above equation (1.1) for any arbitrary permutation. For instance, if a third-order tensor $\mathcal{X} \in \mathbb{R}^{I \times I \times K}$ only satisfies $x_{ijk} = x_{jik}$ for all $i, j = 1, \ldots, I$, $k = 1, \ldots, K$, then we call it partially symmetric tensor on mode 1 and mode 2. So if a Nth-order tensor $\mathcal{T} \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_N}$ has the same size on the mode $i_1, i_2, \ldots, i_n$, where $i_1, \ldots, i_n \in \{1, 2, \ldots, N\}$, i.e., $I_{i_1} = I_{i_2} = \cdots = I_{i_n}$, and for any permutation σ on the sets $\{i_1, i_2, \ldots, i_n\}$, the elements satisfy $t_{i_1 i_2 \ldots i_N} = t_{\sigma(i_1)\sigma(i_2)\ldots\sigma(i_N)}$, then we call such tensor $\mathcal{T}$ partially symmetric on modes

$i_1, i_2, \ldots, i_n$.

There is some analysis of symmetric tensors. In [14], Comon et al. discuss the decomposition of a symmetric tensor into a sum of symmetric rank-one tensors under some rank conditions. Such problem is relevant to the field of algebraic statistics.

Definition 1.3 (Rank-one fully symmetric tensor) *A Nth-order tensor $\mathcal{T} \in \mathbb{R}^{I \times I \times \cdots \times I}$ is a rank-one fully symmetric tensor if it is fully symmetric and can be written as the outer product of N identical vectors, i.e.*

$$\mathcal{T} = \underbrace{\mathbf{a} \circ \mathbf{a} \circ \cdots \circ \mathbf{a}}_{N},$$

where $\mathbf{a} \in \mathbb{R}^I$.

Similarly, we can give the definition of rank-one partially symmetric tensor.

Definition 1.4 (Rank-one partially symmetric tensor) *A Nth-order tensor $\mathcal{T} \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_N}$ is a rank-one partially symmetric tensor if it is partially symmetric on modes $i_1, i_2, \ldots, i_n \in \{1, 2, \ldots, N\}$, and can be written as the outer product of N vectors, i.e.*

$$\mathcal{T} = \underbrace{\mathbf{a}^{(1)} \circ \mathbf{a}^{(2)} \circ \cdots \circ \mathbf{a}^{(N)}}_{N},$$

where $\mathbf{a}^{(i_1)} = \mathbf{a}^{(i_2)} = \cdots = \mathbf{a}^{(i_n)}$.

1.3 Diagonal tensors

Definition 1.5 (Diagonal tensor) *A Nth-order tensor $\mathcal{T} \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_N}$ is called a diagonal tensor if $t_{i_1 i_2 \ldots i_N} \neq 0$ only for $i_1 = i_2 = \cdots = i_N$.*

For a third-order cubical tensor $\mathcal{T} \in \mathbb{R}^{I \times I \times I}$, we define its super-diagonal is the set of elements t_{iii}, $i = 1, 2, \ldots, I$. The following Figure 3.2 shows a diagonal tensor with 1s on the super-diagonal of the cubic.

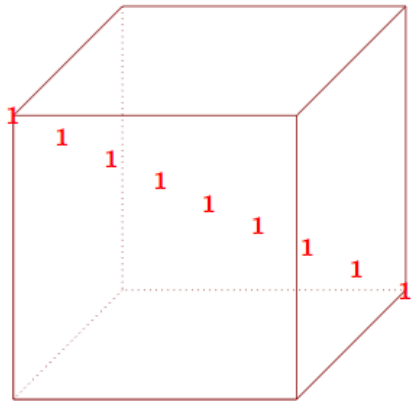


Figure 3.2: **Third-order tensor of size $I \times I \times I$ with ones along the superdiagonal**

So, for a cubical third-order diagonal tensor, its non-zero elements should on the superdiagonal and all the other elements should be zeros.

We will see the major differences between the matrices and tensors in the following discussion of basic tensor decomposition.

2 CANDECOMP/PARAFAC decomposition

In 1927, Hitchcock [55, 54] proposed the idea of the polyadic form of a tensor, i.e., expressing a tensor as the sum of a finite number of rank-one tensors. Currently, this decomposition is called the CANDECOMP/PARAFAC (CP) decomposition. The Parallel Factor Decomposition (PARAFAC) first appeared in [49] in the context of psychometrics. Independently, Carroll and Chang [17] introduced this decomposition as the Canonical Decomposition (CANDECOMP) in phonetics.

Recall that a Nth-order rank-one tensor $\mathcal{T} \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_R}$ is the outer product of N vectors, i.e.,

$$\mathcal{T} = \mathbf{a}_1 \circ \mathbf{a}_2 \circ \cdots \circ \mathbf{a}_N.$$

The CP decomposition factors a tensor as the sum of a finite number of rank-one

tensors:

$$\mathcal{T} = \sum_{r=1}^{R} \mathbf{a}_1^{(r)} \circ \mathbf{a}_2^{(r)} \circ \cdots \circ \mathbf{a}_N^{(r)}, \tag{2.1}$$

where $\mathbf{a}_i^{(r)} \in \mathbb{R}^{I_r}$ denotes the rth vector at the ith position for $r = 1, \ldots, R$, $i = 1, \ldots, N$. The following Figure 3.3 shows the CP decomposition for a third-order tensor.

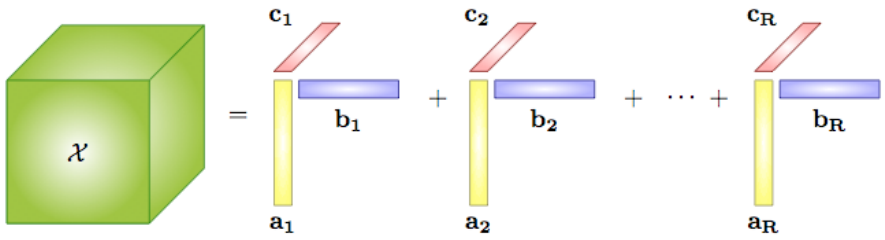


Figure 3.3: **CP decomposition for a third-order tensor**

Elementwise, (2.1) is written as

$$t_{i_1 i_2 \ldots i_N} = \sum_{r=1}^{R} a_{i_1 1}^{(r)} a_{i_2 2}^{(r)} \cdots a_{i_N N}^{(r)} \quad \text{for } i_1 = 1, \ldots, I_1, \ldots, i_N = 1, \ldots, I_N.$$

The ***factor matrices*** are the combination of the vectors from the rank-one components; i.e.,

$$\mathbf{A}_i = [\mathbf{a}_i^{(1)} \ \mathbf{a}_i^{(2)} \ \cdots \ \mathbf{a}_i^{(R)}] \in \mathbb{R}^{I_i \times R} \quad i = 1, 2, \ldots, N.$$

In the following discussion, for third-order tensors, we use $\mathbf{A} = [\mathbf{a}_1 \ \mathbf{a}_2 \ \cdots \ \mathbf{a}_R]$ to denote $\mathbf{A}_1$, $\mathbf{B} = [\mathbf{b}_1 \ \mathbf{b}_2 \ \cdots \ \mathbf{b}_R]$ to denote $\mathbf{A}_2$ and $\mathbf{C} = [\mathbf{c}_1 \ \mathbf{c}_2 \ \cdots \ \mathbf{c}_R]$ to express $\mathbf{A}_3$.

For a fully (partially) symmetric tensor, each component of CP decomposition can be a rank-one fully (partially) symmetric tensor or not. So, we define a decomposition for fully (partially) symmetric tensors which decomposes a fully (partially) symmetric tensor into a summation of rank-one fully (partially) symmetric tensors. We call it

Symmetric Outer Product Decomposition (SOPD) [21]. Similarly, we can take all the vectors in the components from the same modes to get a factor matrix of SOPD. Therefore, it is easy to tell that all factor matrices of fully symmetric tensor are the same and the factor matrices in the according symmetric modes are the same for the partially symmetric tensor. We will talk about the numerical methods of SOPD for partially symmetric tensor in Chap. 5.

Example 2.1 Let $\mathcal{X} \in \mathbb{R}^{3\times3\times3}$ be a real-valued third-order tensor, and its mode-1 matricization is

$$\mathbf{X}_{(1)} = \left[\begin{array}{ccc|ccc|ccc} 1 & 4 & 5 & 2 & 7 & 8 & 1 & 5 & 7 \\ 0 & 4 & 8 & 0 & 6 & 12 & 0 & 6 & 12 \\ 2 & 6 & 6 & 4 & 11 & 10 & 2 & 7 & 8 \end{array}\right],$$

which has the following decomposition:

$$\mathcal{X} = \begin{bmatrix} 1 \\ 0 \\ 2 \end{bmatrix} \circ \begin{bmatrix} 1 \\ 2 \\ 1 \end{bmatrix} \circ \begin{bmatrix} 1 \\ 2 \\ 1 \end{bmatrix} + \begin{bmatrix} 1 \\ 2 \\ 1 \end{bmatrix} \circ \begin{bmatrix} 0 \\ 1 \\ 2 \end{bmatrix} \circ \begin{bmatrix} 2 \\ 3 \\ 3 \end{bmatrix}.$$

So, the CP decomposition factors this tensor into three factor matrices,

$$\mathbf{A} = \begin{bmatrix} 1 & 1 \\ 0 & 2 \\ 2 & 1 \end{bmatrix}, \qquad \mathbf{B} = \begin{bmatrix} 1 & 0 \\ 2 & 1 \\ 1 & 2 \end{bmatrix}, \qquad \mathbf{C} = \begin{bmatrix} 1 & 2 \\ 2 & 3 \\ 1 & 3 \end{bmatrix}.$$

We define the ***rank*** of a tensor $\mathcal{T}$, denoted by rank($\mathcal{T}$), to be the smallest R which satisfies the equation (2.1) [55, 75]. Obviously, for any tensor $\mathcal{T} \in \mathbb{R}^{I_1\times I_2\times\cdots\times I_N}$, there is a trivial CP decomposition with $R = I_1 I_2 \cdots I_N$. However, such R is not the rank of $\mathcal{T}$ since it is the largest R we can find but not the smallest one.

It is known that one way to define the matrix rank is using the singular value

decomposition (SVD). We see that the definition of tensor rank is an analogue to the definition of matrix rank. Actually, the CP decomposition of a given tensor can be considered as a tensor generalization of the matrix SVD. If we let the tensor in the equation (2.1) be second-order, which means that the tensor is a matrix, then CP decomposition is the SVD for the matrix. However, there are several differences between these two ranks. One is that the rank of a real-valued tensor may be different over $\mathbb{R}$ and $\mathbb{C}$. Kruskal provided an example in [76].

Example 2.2 [76] Let $\mathcal{X} \in \mathbb{R}^{2\times 2\times 2}$ be a real-valued third-order tensor with mode-1 matricization

$$\mathbf{X}_{(\mathbf{1})} = \left[\begin{array}{cc|cc} 1 & 0 & 0 & 1 \\ 0 & 1 & -1 & 0 \end{array}\right].$$

This tensor is rank three over $\mathbb{R}$ and rank two over $\mathbb{C}$. The three real-valued factor matrices are

$$\mathbf{A} = \begin{bmatrix} 1 & 0 & 1 \\ 0 & 1 & -1 \end{bmatrix}, \qquad \mathbf{B} = \begin{bmatrix} 1 & 0 & 1 \\ 0 & 1 & 1 \end{bmatrix}, \qquad \mathbf{C} = \begin{bmatrix} 1 & 1 & 0 \\ -1 & 1 & 1 \end{bmatrix}.$$

whereas the three complex-valued factor matrices are

$$\mathbf{A} = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 & 1 \\ -i & i \end{bmatrix}, \qquad \mathbf{B} = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 & 1 \\ i & -i \end{bmatrix}, \qquad \mathbf{C} = \begin{bmatrix} 1 & 1 \\ i & -i \end{bmatrix}.$$

Another difference is that the problem of determining the rank of a given tensor is NP-hard [51]. Even the determination of the maximal rank value over the Nth-order tensor $\mathcal{X} \in \mathbb{R}^{I_1\times I_2\times\cdots\times I_N}$ is still an open problem (it is not bounded by $\min\{I_1, I_2, \ldots, I_N\}$). Comon and Kruskal separately [24, 77] provide examples to discuss the ranks of given tensors from the view of upper and lower bounds.

For a third-order tensor $\mathcal{X} \in \mathbb{R}^{I \times J \times K}$, Kruskal in paper [77] provides a weak upper bound on its maximum rank:

$$\text{rank}(\mathcal{X}) \leq \min\{IJ, IK, JK\}.$$

Ten Berge and Kiers [62] (also see [118]) discuss the set of all third-order tensors of size $I \times J \times 2$ and further study the difference between the ranks over $\mathbb{R}$ and $\mathbb{C}$. When we restrict the tensor in some way, we can get a different result. Ten Berge [63] considers the case where tensor is partially symmetric in two modes and [21] investigates the special case of a fully tensor over $\mathbb{C}$. For a fully symmetric Nth-order tensor $\mathcal{X} \in \mathbb{C}^{I \times I \times \cdots \times I}$, he defines the *symmetric rank* (over $\mathbb{C}$) [70] of $\mathcal{X}$ to be

$$\text{rank}_s(\mathcal{X}) = \min\{R : \mathcal{X} = \sum_{r=1}^{R} \mathbf{a}_r \circ \mathbf{a}_r \circ \cdots \circ \mathbf{a}_r, \quad \text{where } \mathbf{A} \in \mathbb{C}^{I \times C}\},$$

i.e., the minimum number of fully symmetric rank-one factors.

Additionally, the uniqueness properties of the CP are very different from (and also much more complicated than) their matrix equivalents. For some history of uniqueness results of CP, see [110, 119]. Harshman in 1970 [49] presented the earliest uniqueness result and further provided some results in 1972 [50].

We know that the matrix decomposition cannot be unique. Consider a matrix $\mathbf{M} \in \mathbb{R}^{I \times J}$ with rank R, then it can be writen as

$$\mathbf{M} = \mathbf{A}\mathbf{B}^{\mathrm{T}} = \sum_{r=1}^{R} \mathbf{a}_r \circ \mathbf{b}_r,$$

where $\mathbf{A} \in \mathbb{R}^{I \times R}$ and $\mathbf{B}^{J \times R}$. If we introduce an orthogonal matrix $\mathbf{W} \in \mathbb{R}^{R \times R}$ (not a identity matrix), then the $\mathbf{M}$ has another decomposition,

$$\mathbf{M} = (\mathbf{A}\mathbf{W})(\mathbf{B}\mathbf{W})^{\mathrm{T}}.$$

Notice that the SVD of a matrix is unique (assuming all the singular values are distinct) only because it has the orthogonality constraints and the ordered singular values.

On the other hand, for the CP decomposition of a third-order tensor,

$$\mathcal{X} = \sum_{r=1}^{R} \mathbf{a}_r \circ \mathbf{b}_r \circ \mathbf{c}_r,$$

the unique conditions are much weaker. Uniqueness means that this is the only possible combination of rank-one tensors that sums to $\mathcal{X}$, with the exception of the elementary indeterminacies of scaling and permutation [70]. The permutation refers to the fact that for any permutation σ on the indices set $\{1, 2, \ldots, R\}$,

$$\mathcal{X} = \sum_{r=1}^{R} \mathbf{a}_{\sigma(r)} \circ \mathbf{b}_{\sigma(r)} \circ \mathbf{c}_{\sigma(r)}.$$

The scaling indeterminacy refers to the fact that for $\alpha_r, \beta_r, \gamma_r \in \mathbb{R}$, $\alpha_r \beta_r \gamma_r = 1$ for $r = 1, \ldots, R$,

$$\mathcal{X} = \sum_{r=1}^{R} (\alpha_r \mathbf{a}_r) \circ (\beta_r \mathbf{b}_r) \circ (\gamma_r \mathbf{c}_r).$$

The most well-known result on uniqueness is due to Kruskal in 1977 [75, 77] and depends on the concept of k-rank. The ***k-rank*** of a matrix $\mathbf{A}$, denoted by $k_{\mathbf{A}}$, is defined as the maximum value k such that any k columns are linearly independent in $\mathbf{A}$. For a third-order tensor $\mathcal{T} \in \mathbb{R}^{I \times J \times K}$, he provides a sufficient condition for uniqueness up to permutation and scalings for the CP decomposition, which is,

$$k_{\mathbf{A}} + k_{\mathbf{B}} + k_{\mathbf{C}} \geq 2R + 2.$$

Later on, De Lathauwer [27] and Jiang and Sidiropoulous [61] gave new sufficient

conditions for uniqueness by assuming only one full-rank factor with the new bound,

$$\frac{R(R-1)}{2} \leq \frac{I(I-1)(J(J-1))}{4}.$$

Stegeman discussed the CP decomposition uniqueness in several papers [116, 114, 115], including some results on partially symmetric tensors. The following Table 3.1 shows the uniqueness conditions of CP decomposition and Table 3.2 presents the uniqueness conditions for the partially symmetric CP decomposition by Stegeman. In the table 3.1, he also discusses generic uniqueness conditions. We call a property ***generic*** when it holds with probability one when the parameters of the problem are drawn from continuous probability density functions. For instance, let $\mathbf{A} \in \mathbb{R}^{I \times R}$, generically, $k_{\mathbf{A}} = \min\{I, R\}$ where $k_{\mathbf{A}}$ is the k-rank of $\mathbf{A}$. In terms of the generic uniqueness, Stegeman considers the situation that the first $N-1$ factor matrices $\mathbf{A}^{(1)}$, $\ldots$, $\mathbf{A}^{(N)}$ are generic.

Uniqueness conditions I			
n	size tensor	bound on R	bound on R (generic case)
$n=3$	$4 \times 4 \times I_3$, $I_3 \geq R$	$R \leq 6$	$R \leq 9$ [114]
$n=4$	$2 \times 3 \times 4 \times I_4$, $I_4 \geq R$	$R \leq 6$	$R \leq 6$ [114]
	$4 \times 4 \times 4 \times I_4$, $I_4 \geq R$	$R \leq 9$	$R \leq 46$ [115]
$n=5$	$2 \times 3 \times 4 \times 2 \times I_5$, $I_5 \geq R$	$R \leq 7$	$R \leq 36$ [114]
	$4 \times 4 \times 4 \times 4 \times I_5$, $I_5 \geq R$	$R \leq 12$	$R \leq 214$ [115]
$n=6$	$2 \times 3 \times 2 \times 2 \times 2 \times I_6$, $I_6 \geq R$	$R \leq 7$	$R \leq 58$ [114]
	$4 \times 4 \times 4 \times 4 \times 4 \times I_6$, $I_6 \geq R$	$R \leq 12$	$R \leq 214$ [115]

Table 3.1: **Comparison of tensor rank bounds for the uniqueness of CP decomposition**

In the above two tables, we let the last factor matrix $\mathbf{A}^{(N)}$ be full-column rank and a tall matrix (the number of rows I_N is greater than the number of columns R), so $\text{rank}(\mathbf{A}^{(N)}) = R$. The generic case means that the factor matrices $\mathbf{A}^{(1)}, \ldots, \mathbf{A}^{(N)}$ are generic with $\text{rank}(\mathbf{A}^{(N)}) = R$. In the last column of Table 3.2, $\mathbf{A}^{(1)} = \mathbf{A}^{(2)}$ contains the tensor to be a partially symmetric on mode-1 and mode-2, i.e., the elements of the

Uniqueness conditions II		
n	size tensor	bound on R with symmetry
$n = 3$	$4 \times 4 \times I_3$, $I_3 \geq R$	$R \leq 6 \quad (\mathbf{A}^{(1)} = \mathbf{A}^{(2)})$ [115]
$n = 4$	$4 \times 4 \times 4 \times I_4$, $I_4 \geq R$	$R \leq 31 \quad (\mathbf{A}^{(1)} = \mathbf{A}^{(2)})$ [115]
$n = 5$	$4 \times 4 \times 4 \times 4 \times I_5$, $I_5 \geq R$	$R \leq 137 \quad (\mathbf{A}^{(1)} = \mathbf{A}^{(2)})$ [115]
$n = 6$	$4 \times 4 \times 4 \times 4 \times 4 \times I_6$, $I_6 \geq R$	$R \leq 87 \quad (\mathbf{A}^{(1)} = \mathbf{A}^{(2)}$ and $\mathbf{A}^{(3)} = \mathbf{A}^{(4)})$ [115]

Table 3.2: **Comparison of rank bounds for the uniqueness of partially tensor CP decomposition**

tensor have the property of $a_{i_1 i_2 \ldots i_N} = a_{i_2 i_1 \ldots i_N}$, for $i_n = 1, 2 \ldots, I_n$ and $n = 1, \ldots, N$. The last tensor needs be symmetric on mode-1 and mode-2 as well as mode-3 and mode-4.

3 HOSVD/Tucker decomposition

We know that the CANDECOMP/PARAFAC (CP) decomposition can be considered as a tensor generalization of the matrix singular value decomposition. There is another generalization of the matrix SVD which is called *higher-order singular value decomposition* (HOSVD) [33] or *Tucker decomposition*[123].

In 1963, Tucker first introduced the Tucker [123] decomposition of a higher-order tensor and proposed a method to compute such a decomposition for a third-order tensor. The most comprehensive of the early literature is Tucker's 1966 article [124], which is generally the one most cited. Later in 2000, the work of De Lathauwer, De Moor, and Vandewalle [33] showed Tucker's method is valid for tensors of arbitrary order and hold for the complex-valued case too. They call it the ***Higher-Order Singular Value Decomposition*** (**HOSVD**).

Since 1966, Tucker decomposition has been applied in a lot of scientific areas. Henrion [53] provided several examples of applying the Tucker decomposition in chemical analysis. De Lathauwer and Vandewalle [37] use Tucker in the applications of signal processing, also see [90]. Moreover, it has also been applied in the field of data mining

and computer vision [108, 125]; see the survey paper [70] and the references therein.

3.1 HOSVD

Tucker introduces the Tucker decomposition for a third-order tensor $\mathcal{T} \in \mathbb{R}^{I \times J \times K}$, which decomposes $\mathcal{T}$ into a core tensor multiplied by a matrix along each mode. Thus, we have

$$\mathcal{T} = \mathcal{G} \times_1 \mathbf{A} \times_2 \mathbf{B} \times_3 \mathbf{C} = \sum_{p=1}^{P} \sum_{q=1}^{Q} \sum_{r=1}^{R} g_{pqr} \mathbf{a}_p \circ \mathbf{b}_q \circ \mathbf{c}_r, \tag{3.1}$$

where $\mathbf{A} \in \mathbb{R}^{I \times P}$, $\mathbf{B} \in \mathbb{R}^{J \times Q}$ and $\mathbf{C} \in \mathbb{R}^{K \times R}$ are the factor matrices. The tensor $\mathcal{G} \in \mathbb{R}^{P \times Q \times R}$ is called the *core tensor* and g_{pqr} is the component of $\mathcal{G}$.

Elementwise, the Tucker decomposition in (3.1) is

$$t_{ijk} = \sum_{p=1}^{P} \sum_{q=1}^{Q} \sum_{r=1}^{R} g_{pqr} a_{ip} b_{jq} c_{kr}, \tag{3.2}$$

in which a_{ip}, b_{jq}, c_{kr} are the entries of the factor matrices, and P, Q, R are the number of columns in the factor matrices. The Figure 3.4 illustrates the Tucker decomposition.

De Lathauwer, De Moor, and Vandewalle generalized Tucker's method to any higher-order tensor in [33] and provided the following theorem.

Theorem 3.1 (HOSVD) *[33] Every complex-valued tensor $\mathcal{T} \in \mathbb{C}^{I_1 \times I_2 \times \cdots \times I_N}$ can be written as the product*

$$\mathcal{T} = \mathcal{G} \times_1 \mathbf{U}^{(1)} \times_2 \mathbf{U}^{(2)} \times_3 \cdots \times_N \mathbf{U}^{(N)}, \tag{3.3}$$

in which

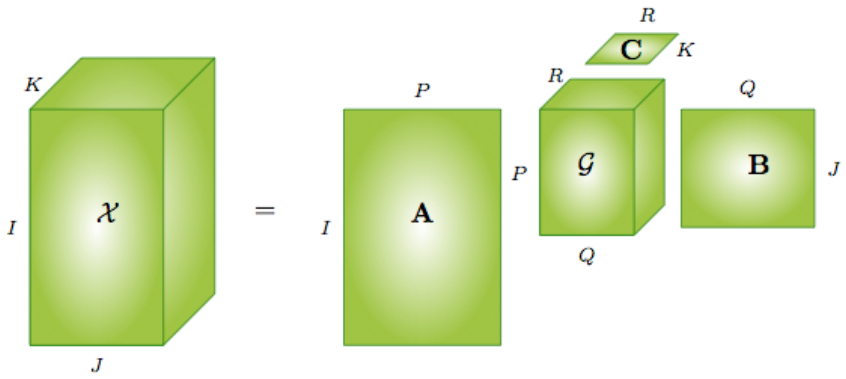


Figure 3.4: **Tucker decomposition of a third-order tensor**

1. $\mathbf{U}^{(n)} = [\mathbf{u}_1^{(n)}, \mathbf{u}_2^{(n)}, \cdots, \mathbf{u}_{I_n}^{(n)}] \in \mathbb{C}^{I_n \times I_n}$ *is a unitary matrix, where* $\mathbf{u}_i^{(n)}$ *is the* i*th column vector of the matrix* $\mathbf{U}^{(n)}$.

2. $\mathcal{G} \in \mathbb{C}^{I_1 \times I_2 \times \cdots \times I_N}$ *is a complex-valued tensor of which the* ***subtensors*** $\mathcal{G}_{i_n=\alpha}$*, obtained by fixing the* n*th index to* α*, have the properties of*

   (a) *all-orthogonality: two subtensors* $\mathcal{G}_{i_n=\alpha}$ *and* $\mathcal{G}_{i_n=\beta}$ *are orthogonal for all possible values of* n*,* α *and* β *subject to* $\alpha \neq \beta$*:*

$$\langle \mathcal{G}_{i_n=\alpha}, \mathcal{G}_{i_n=\beta} \rangle = 0 \qquad \text{when} \quad \alpha \neq \beta, \tag{3.4}$$

   where $\langle \mathcal{G}_{i_n=\alpha}, \mathcal{G}_{i_n=\beta} \rangle$ *means the inner product of the two tensors.*

   (b) *ordering:*

$$\|\mathcal{G}_{i_n=1}\| \geq \|\mathcal{G}_{i_n=2}\| \geq \cdots \geq \|\mathcal{G}_{i_n=I_n}\| \geq 0 \tag{3.5}$$

   for all possible values of n.

The Frobenius-norms $\|\mathcal{G}_{i_n=i}\|$*, symbolized by* $\sigma_i^{(n)}$*, are* n*-mode singular values of* $\mathcal{T}$ *and the vector* $\mathbf{u}_i^{(n)}$ *is an* i*th* n*-mode singular vector.*

Example 3.2 Let $\mathcal{X} \in \mathbb{R}^{2\times 2\times 2}$ is a third-order tensor whose mode-1 matricization is

$$\mathbf{X}_{(1)} = \left[\begin{array}{cc|cc} 1 & 2 & 5 & 6 \\ 3 & 4 & 7 & 8 \end{array}\right].$$

Then we have the factor matrices

$$\mathbf{A} = \begin{bmatrix} -0.5667 & -0.8239 \\ -0.8239 & 0.5667 \end{bmatrix}, \quad \mathbf{B} = \begin{bmatrix} -0.6414 & -0.7672 \\ -0.7672 & -0.6414 \end{bmatrix}, \quad \mathbf{C} = \begin{bmatrix} -0.3762 & -0.9266 \\ -0.9266 & -0.3762 \end{bmatrix},$$

and mode-1 matricization of the core tensor $\mathcal{G} \in \mathbb{R}^{2\times 2\times 2}$ is

$$\mathbf{G}_{(1)} = \left[\begin{array}{cc|cc} -14.2254 & 0.0160 & 0.0046 & 0.5438 \\ 0.0083 & 0.2386 & 1.1159 & 0.2001 \end{array}\right].$$

So the exact Tucker decomposition is $\mathcal{X} = \mathcal{G} \times_1 \mathbf{A} \times_2 \mathbf{B} \times_3 \mathbf{C}$. We can check this by using the equation (3.2). Furthermore, it is a HOSVD decomposition since all the three matrices are orthogonal, i.e., $\mathbf{A}^{\mathrm{T}}\mathbf{A} = \mathbf{I}^{2\times 2}$, $\mathbf{B}^{\mathrm{T}}\mathbf{B} = \mathbf{I}^{2\times 2}$ and $\mathbf{C}^{\mathrm{T}}\mathbf{C} = \mathbf{I}^{2\times 2}$, where $\mathbf{I}^{2\times 2}$ is a 2×2 identity matrix.

We define the ***n*-rank** (also called n-mode rank) of a tensor $\mathcal{T} \in \mathbb{R}^{I_1\times I_2\times\cdots\times I_N}$ to be the column rank of $\mathbf{T}_{(n)}$, denoted by $\mathrm{rank}_n(\mathcal{T})$. If we let $R_n = \mathrm{rank}_n(\mathcal{T})$ for $n = 1, \ldots, N$, then we say that $\mathcal{T}$ is a rank-$(R_1, R_2, \ldots, R_N)$ tensor. This definition is introduced by Kruskal [77] and it is further studied by De Lathauwer, De Moor, and Vandewalle [33].

Tucker/HOSVD decomposition is not unique since the Tucker model has rotational freedom. For the decomposition (3.1), let $\mathbf{U} \in R^{P\times P}$, $\mathbf{V} \in \mathbb{R}^{Q\times Q}$ and $\mathbf{W} \in \mathbb{R}^{R\times R}$ be nonsingular matrices. Then we have

$$\mathcal{T} = \mathcal{G} \times_1 \mathbf{A} \times_2 \mathbf{B} \times_3 \mathbf{C} = (\mathcal{G} \times_1 \mathbf{U} \times_2 \mathbf{V} \times_3 \mathbf{W}) \times_1 \mathbf{A}\mathbf{U}^{-1} \times_2 \mathbf{B}\mathbf{V}^{-1} \times_3 \mathbf{C}\mathbf{W}^{-1}.$$

So we have a different decomposition with the new core tensor $\mathcal{G} \times_1 \mathbf{U} \times_2 \mathbf{V} \times_3 \mathbf{W}$ and the new three factor matrices $\mathbf{A}\mathbf{U}^{-1}$, $\mathbf{B}\mathbf{V}^{-1}$ and $\mathbf{C}\mathbf{W}^{-1}$.

Therefore, when we use Tucker decomposition for a given tensor, we can simplify the core tensor so that most of the elements of $\mathcal{G}$ are zero. This is a way to improve the uniqueness. It has been shown that making the core tensor $\mathcal{G}$ be a super-diagonal is impossible (even in the symmetric case) see [70, 21, 23]. However, it is possible to try to make as many elements very small as possible. This was first observed by Tucker in [123]. Later, there are some researchers to study this problem; see, e.g., [3, 52, 89].

Chapter 4

Alternating least-squares and regularized alternating least-squares

1 Introduction

In this chapter we study the numerical methods for computing a CP decomposition for a given tensor and introduce a regularized method for PARAFAC. Also, we propose a convergence result of the regularized method and provide the proof in Section 5.

The tensor decomposition analysis techniques rely on numerical methods in optimization and numerical linear algebra. Recall the definition of CP decomposition for a given tensor $\mathcal{T}$, from the perspective of optimization, the problem can be written as

$$\underset{\widehat{\mathcal{T}}}{\text{minimize}} \quad \|\mathcal{T} - \widehat{\mathcal{T}}\|_F^2, \quad \text{where} \quad \widehat{\mathcal{T}} = \sum_{r=1}^{R} \mathbf{a}_r \circ \mathbf{b}_r \circ \mathbf{c}_r. \tag{1.1}$$

Alternating least-squares (ALS) method is a popular algorithm for solving the problem (1.1). Independently, the ALS was introduced by Carol and Chang [17] and

Harshman [49] in 1970. It has been extensively applied to many problems across various engineering [1, 39, 112, 111] and science [73, 113] fields; see the survey papers [70, 30] and the references therein. For example, Beylkin and Mohlenkamp [11, 12] utilize ALS to compute optimal separation rank for certain operators like inverse Laplacian and the multiparticle Schrödinger equation to reduce computational complexity.

The widespread success of ALS can be attributed to its simplicity. Moreover, Bro et al. [43, 122] found that the ALS gives superior quality solutions with fewer memory and time requirements than the other CP methods. Despite its success, ALS has some drawbacks. For example, initialization of the factor matrices, collinearity in the factor matrices or degeneracy problems may require a high number of iterations for the ALS method to converge. This slowed convergence characterized by a flat curve in a log error plot is referred to as the *swamp*. The following Figure 4.1 which is also in Section 3 shows a swamp plot of ALS numerically.

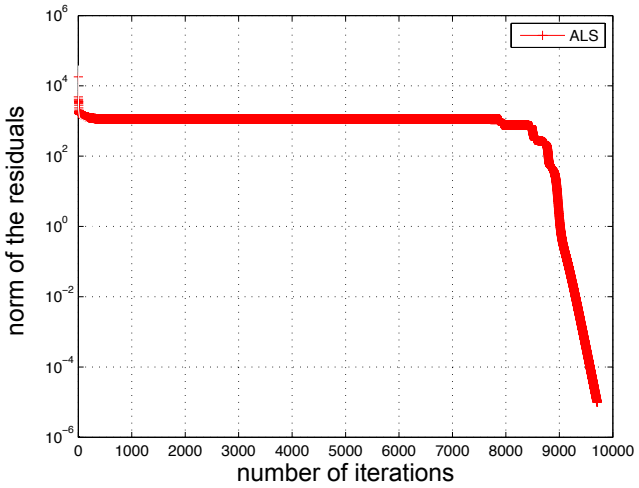


Figure 4.1: **A swamp phenomenon. x-axis: the number of iterations; y-axis: Frobeinus norm of the residual of the original tensor and the tensor obtained at each iteration.**

Swamps can be present in the non-degenerate and degenerate cases. The degener-

ate case is a more challenging problem; see [96, 84] for some regularization techniques for the degenerate swamps. In terms of the non-degenerate case, several numerical techniques have been used to address it. Line search schemes [106, 92] have been used to accelerate the ALS algorithm. An entirely different approach by De Lathauwer, Demoor and Vandewalle obviates the swamp issues by considering a simultaneous matrix diagonalization for CP decomposition [35, 27]. Paatero [93] has applied an initial regularization along with a gradient descent based method for CP.

A regularization method (RALS) is introduced by Navasca, Kindermann and De Lathawer [91] for addressing the issues of the swamp occurrences in the non-degenerate case. It is no more complicated than the ALS algorithm. Unlike the ALS regularization method found in [96, 84], RALS is an unconstrained optimization problem since there is no uniform constraint in the penalty terms that are sequentially changing at each iteration, and hence, the limit points of the sequences of RALS can be unbounded. Thus, RALS does not address the degeneracy problem. Here we will show that if a limit point of RALS exists, then it is a critical point of the original functional. The statement of the theorem (see Theorem 5.1) is following:

Suppose that the sequence $\{\mathcal{T}^k\}$ *obtained from RALS has limit points, then every limit point* $\overline{\mathcal{T}}$ *is a critical point of the above problem (1.1).* The proof of the theorem 5.1 is also included in Section 5.

The organization of this chapter is following, in Section 2 we introduce the ALS method for solving CP decomposition. In Section 3 we analyze the ALS method in nonlinear programming and introduce the Block Nonlinear Gauss-Seidel (GS) method. Further in Section 3, we explain the link between ALS and GS and properties of both are analyzed. At the end of this section, a "swamp" phenomena in the ALS method is pointed out. In Section 4 we examine the method proposed in [91], a regularization of the ALS (RALS). Several numerical examples show that RALS converges faster than ALS and can remove some swamps. Later in Section 4, we make

a link between RALS and Proximal Point Modification of the Gauss-Seidel (PGS) method. Finally, Section 5 proves that given the existence of some critical points of the RALS method, the limit points of the converging subsequences of the RALS are the critical points of the least-squares cost functional.

2 ALS algorithm

2.1 Computing the CP: ALS method

Recall that the CP decomposition of a tensor $\mathcal{T}$ decomposes the tensor as a sum of a finite number of rank-one tensors:

$$\mathcal{T} = \sum_{r=1}^{R} \mathbf{a}_1^{(r)} \circ \mathbf{a}_2^{(r)} \circ \cdots \circ \mathbf{a}_N^{(r)}. \tag{2.1}$$

In this section, we study the numerical method to solve the above equation, i.e., finding the vectors $\mathbf{a}_1^{(r)}$, $\mathbf{a}_2^{(r)}$, $\ldots$, $\mathbf{a}_N^{(r)}$, $r = 1, 2, \ldots, R$ to satisfy the equation.

The first issue of computing the CP is to determine the rank of a tensor. However, as mentioned in previous chapter, this problem is NP-hard [51]. Therefore, one common method is to do CP computation for $R = 1, 2, 3, \ldots$ until the first value of R that can fit the equation (2.1) perfectly. However this method has some issues, for example, Paatero [95] shows that a third-order tensor of rank-three can be approximated arbitrarily closely by a rank-two tensor.

Thus, we assume the number of components R is fixed in the following algorithm discussion.

A very popular numerical method to compute the CP decomposition for a given tensor is the ***Alternating Least-Squares*** (ALS) method. For the simplicity of the exposition, we just look at third-order tensors, but all the analysis holds for higher-order tensors.

The problem we want to solve is the following: given a third-order tensor $\mathcal{T} \in \mathbb{R}^{I\times J\times K}$, compute its CP decomposition with R components of rank-one tensors that best approximates $\mathcal{T}$. So, it can be considered as an optimization problem:

$$\underset{\widehat{\mathcal{T}}}{\text{minimize}} \quad \|\mathcal{T}-\widehat{\mathcal{T}}\|_F^2, \quad \text{where} \quad \widehat{\mathcal{T}}=\sum_{r=1}^{R}\mathbf{a}_r\circ\mathbf{b}_r\circ\mathbf{c}_r. \tag{2.2}$$

This problem is equivalent to

$$\min_{\mathbf{A},\mathbf{B},\mathbf{C}} \quad \|\mathcal{T}-\sum_{r=1}^{R}\mathbf{a}_r\circ\mathbf{b}_r\circ\mathbf{c}_r\|_F^2, \tag{2.3}$$

with respect to factor matrices $\mathbf{A}$, $\mathbf{B}$ and $\mathbf{C}$.

$\widehat{\mathcal{T}}$ satisfies the CP decomposition equation (2.1), so by using the Khatri-Rao product and tensor matricization, it can be written in three matricized forms,

$$\begin{aligned}
\widehat{\mathcal{T}} &= \sum_{r=1}^{R}\mathbf{a}_r\circ\mathbf{b}_r\circ\mathbf{c}_r \\
&\Downarrow \\
\widehat{\mathbf{T}}_{(\mathbf{1})} &= \mathbf{A}(\mathbf{C}\odot\mathbf{B})^{\mathrm{T}}, \\
\widehat{\mathbf{T}}_{(\mathbf{2})} &= \mathbf{B}(\mathbf{C}\odot\mathbf{A})^{\mathrm{T}}, \\
\widehat{\mathbf{T}}_{(\mathbf{3})} &= \mathbf{C}(\mathbf{B}\odot\mathbf{A})^{\mathrm{T}}.
\end{aligned}$$

where $\mathbf{A}\in\mathbb{R}^{I\times R}$, $\mathbf{B}\in\mathbb{R}^{J\times R}$ and $\mathbf{C}\in\mathbb{R}^{K\times R}$.

Then for the original problem (2.2), by the three types of matricizations for both $\mathcal{T}$ and $\widehat{\mathcal{T}}$, it has the following three expressions,

$$\begin{aligned}
&\min_{\mathbf{A},\mathbf{B},\mathbf{C}} \quad \|\mathbf{T}_{(\mathbf{1})}-\mathbf{A}(\mathbf{C}\odot\mathbf{B})^{\mathrm{T}}\|_F^2, \\
&\min_{\mathbf{A},\mathbf{B},\mathbf{C}} \quad \|\mathbf{T}_{(\mathbf{2})}-\mathbf{B}(\mathbf{C}\odot\mathbf{A})^{\mathrm{T}}\|_F^2, \\
&\min_{\mathbf{A},\mathbf{B},\mathbf{C}} \quad \|\mathbf{T}_{(\mathbf{3})}-\mathbf{C}(\mathbf{B}\odot\mathbf{A})^{\mathrm{T}}\|_F^2.
\end{aligned}$$

These three are equivalent. Instead of solving (2.2) for the three variables one time, we can use these three equations by fixing all factor matrices but one each time, then the problem reduces to three coupled linear least-squares subproblems.

We have

$$\begin{aligned}
\mathbf{A}^{k+1} &= \underset{\widehat{\mathbf{A}}\in\mathbb{R}^{I\times R}}{\operatorname{argmin}} \|\mathbf{T}_{(\mathbf{1})} - \widehat{\mathbf{A}}(\mathbf{C}^k \odot \mathbf{B}^k)^{\mathrm{T}}\|_F^2, \\
\mathbf{B}^{k+1} &= \underset{\widehat{\mathbf{B}}\in\mathbb{R}^{J\times R}}{\operatorname{argmin}} \|\mathbf{T}_{(\mathbf{2})} - \widehat{\mathbf{B}}(\mathbf{C}^k \odot \mathbf{A}^{k+1})^{\mathrm{T}}\|_F^2, \\
\mathbf{C}^{k+1} &= \underset{\widehat{\mathbf{C}}\in\mathbb{R}^{K\times R}}{\operatorname{argmin}} \|\mathbf{T}_{(\mathbf{3})} - \widehat{\mathbf{C}}(\mathbf{B}^{k+1} \odot \mathbf{A}^{k+1})^{\mathrm{T}}\|_F^2,
\end{aligned} \tag{2.4}$$

where $\mathbf{T}_{(\mathbf{1})} \in \mathbb{R}^{I\times JK}$, $\mathbf{T}_{(\mathbf{2})} \in \mathbb{R}^{J\times IK}$ and $\mathbf{T}_{(\mathbf{3})} \in \mathbb{R}^{K\times IJ}$ are the mode-1, mode-2 and mode-3 matricizations of tensor $\mathcal{T}$.

Thus, given three initial factor matrices $\mathbf{A}^0$, $\mathbf{B}^0$ and $\mathbf{C}^0$, the ALS method solves the three least-squares subproblems in (2.4) to obtain the factor matrices $\mathbf{A}$, $\mathbf{B}$ and $\mathbf{C}$. Starting from the initial guesses $\mathbf{A}^0$, $\mathbf{B}^0$, $\mathbf{C}^0$, the ALS approach fixes $\mathbf{B}$ and $\mathbf{C}$ to solve for $\mathbf{A}$, then fixes $\mathbf{A}$ and $\mathbf{C}$ to solve for $\mathbf{B}$, and then fixes $\mathbf{A}$ and $\mathbf{B}$ to solve for $\mathbf{C}$. This process continues iteratively until some convergence criterion is satisfied. Therefore, this method translates the original nonlinear minimization problem to three subproblems where each one is just a least-squares problem.

The full ALS procedure for a third-order tensor is shown in the following Table 4.1. The initial guesses can be random or set by using some theoretic analysis. See [113] for more analysis on initializing the ALS method.

The number of iterations M is set to a large number; otherwise a convergence stopping criterion can be used. The notation "/" in the algorithm is MATLAB notation to calculate the least square solution $\mathbf{X} = \mathbf{B}/\mathbf{A}$ of equation $\mathbf{XA} = \mathbf{B}$.

For the general case, a Nth-order tensor $\mathcal{T} \in \mathbb{R}^{I_1\times I_2\times\cdots\times I_N}$, by fixing the rank R,

ALS-Algorithm

procedure CP-ALS($\mathcal{T}, R, M$)

give initial guess $\mathbf{A}^0 \in \mathbb{R}^{I\times R}$, $\mathbf{B}^0 \in \mathbb{R}^{J\times R}$, $\mathbf{C}^0 \in \mathbb{R}^{K\times R}$

for $n = 1, \ldots, M$ **do**

$\mathbf{A}^{n+1} \leftarrow \mathbf{T}_{(\mathbf{1})}/(\mathbf{C}^n \odot \mathbf{B}^n)^{\mathrm{T}}$ —— % solving least squares to update $\mathbf{A}$

$\mathbf{B}^{n+1} \leftarrow \mathbf{T}_{(\mathbf{2})}/(\mathbf{C}^n \odot \mathbf{A}^{n+1})^{\mathrm{T}}$ —% solving least squares to update $\mathbf{B}$

$\mathbf{C}^{n+1} \leftarrow \mathbf{T}_{(\mathbf{3})}/(\mathbf{B}^{n+1} \odot \mathbf{A}^{n+1})^{\mathrm{T}}$ —— % solving least squares to update $\mathbf{C}$

end for

return $\mathbf{A}^M$, $\mathbf{B}^M$, $\mathbf{C}^M$

end procedure

Table 4.1: **ALS algorithm of CP decomposition with rank R for a third-order tensor $\mathcal{X} \in \mathbb{R}^{I\times J\times K}$**

the CP problem is

$$\underset{\widehat{\mathcal{T}}}{\text{minimize}} \quad \|\mathcal{T} - \widehat{\mathcal{T}}\|_F^2, \quad \text{where} \quad \widehat{\mathcal{T}} = \sum_{r=1}^{R} \mathbf{a}_1^{(r)} \circ \mathbf{a}_2^{(r)} \circ \cdots \circ \mathbf{a}_N^{(r)}, \tag{2.5}$$

then by matricizing the tensor $\widehat{\mathcal{T}}$, we have

$$\widehat{\mathbf{T}}_{(\mathbf{n})} = \mathbf{A}_n(\mathbf{A}_N \odot \mathbf{A}_{N-1} \odot \cdots \odot \mathbf{A}_{n+1} \odot \mathbf{A}_{n-1} \odot \cdots \odot \mathbf{A}_1)^{\mathrm{T}}.$$

Therefore, in the algorithm for the third-order tensor, we can use

$$\mathbf{A}_n^{k+1} = \mathbf{T}_{(\mathbf{n})}/(\mathbf{A}_N^k \odot \mathbf{A}_{N-1}^k \odot \cdots \odot \mathbf{A}_{n+1}^k \odot \mathbf{A}_{n-1}^{k+1} \odot \cdots \odot \mathbf{A}_1^{k+1})^{\mathrm{T}}$$

to update the nth factor matrix at the $(k+1)$th iteration.

3 CP as a nonlinear optimization

In this section, we discuss the CP decomposition problem (2.5) and the ALS method (2.4) from the view of nonlinear optimization.

3.1 Block nonlinear Gauss-Seidel method

Consider the problem:

$$\begin{aligned} &\text{minimize} \quad f(\mathbf{x}) \\ &\text{subject to} \quad \mathbf{x} \in X = X_1 \times X_2 \times \cdots X_m \subseteq \mathbb{R}^n, \end{aligned} \tag{3.1}$$

where f is a continuously differentiable function from $\mathbb{R}^n$ to $\mathbb{R}$ and X is the cartesian product of closed, nonempty and convex subsets $X_i \subseteq \mathbb{R}^{n_i}$, for $i = 1, \ldots, m$, with $\sum_{i=1}^{m} n_i = n$. If the vector $\mathbf{x} \in \mathbb{R}^n$ is partitioned into m component vectors $\mathbf{x}_i \in \mathbb{R}^{n_i}$, then we can consider f as a function from $\mathbb{R}^{n_1} \times \mathbb{R}^{n_2} \times \cdots \mathbb{R}^{n_m}$ to $\mathbb{R}$ with

$$f(\mathbf{x}) = f(\mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_m).$$

The **Nonlinear Block Gauss-Seidel** (GS) [5, 9, 10, 46, 47] method is used to find a minimizer of such a nonlinear functional. The solution of (3.1) can be found using the iterative technique,

$$\mathbf{x}_i^{k+1} = \underset{\mathbf{y}_i \in X_i}{\operatorname{argmin}} f(\mathbf{x}_1^{k+1}, \ldots, \mathbf{x}_{i-1}^{k+1}, \mathbf{y}_i, \mathbf{x}_{i+1}^{k}, \ldots, \mathbf{x}_m^{k}), \tag{3.2}$$

which updates the components of $\mathbf{x}$, starting from a given initial guess $\mathbf{x}^0 = (\mathbf{x}_1^0, \mathbf{x}_2^0, \ldots, \mathbf{x}_m^0) \in X$ and generating a sequence $\{\mathbf{x}^k\} = \{(\mathbf{x}_1^k, \mathbf{x}_2^k, \ldots, \mathbf{x}_m^k)\}$.

Naturally, this method makes sense if the minimization in (3.2) is easily solved. For example, when each $\mathbf{x}_i$ is a scalar, we can implement GS conveniently. However,

it can be complicated for the case that $\mathbf{x}_i$ is a multidimensional vector.

Let us discuss some basic convergence properties of the GS method.

Definition 3.1 (Critical Point) *Let $f : X \to \mathbb{R}$, $X \subset \mathbb{R}^n$ be a continuously differentiable function, a **critical point** of f is a point $\bar{\mathbf{x}} \in X$ such that*

$$\nabla f(\overline{\mathbf{x}})^T(\mathbf{y} - \overline{\mathbf{x}}) \geq 0, \quad \forall\, \mathbf{y} \in X, \tag{3.3}$$

where $\nabla f(\mathbf{x}) \in \mathbb{R}^n$ denotes the gradient of f at $\mathbf{x}$. If $X = \mathbb{R}^n$ or if $\overline{\mathbf{x}}$ is an iterior point of X, then the condition (3.3) reduces to the stationarity condition $\nabla f(\overline{\mathbf{x}}) = \mathbf{0}$ of unconstrained optimization.

Definition 3.2 (Limit Point) *We say that a vector $\mathbf{x} \in \mathbb{R}^n$ is a **limit point** of a sequence $\{\mathbf{x}^k\}_{k=1}^{\infty}$ in $\mathbb{R}^n$ if there exists a subsequence of $\{\mathbf{x}^k\}_{k=1}^{\infty}$ that converges to $\mathbf{x}$.*

Definition 3.3 (Convex Function) *A real-valued function $f(x)$ defined on a convex set is called **convex** if for any two points x_1 and x_2, in its domain and any $t \in [0, 1]$,*

$$f(tx_1 + (1-t)x_2) \leq tf(x_1) + (1-t)f(x_2).$$

If furthermore,

$$f(tx_1 + (1-t)x_2) < tf(x_1) + (1-t)f(x_2),$$

*with $x_1 \neq x_2$, then f is **strictly convex**.*

Definition 3.4 (Quasiconvex Function) *A function $f : S \to \mathbb{R}$ defined on a convex subset S of a real vector space is **quasiconvex** if whenever $x, y \in S$ and $\lambda \in [0, 1]$, then*

$$f(\lambda x + (1-\lambda)y) \leq \max(f(x), f(y)).$$

If furthermore,

$$f(\lambda x + (1-\lambda)y) < \max(f(x), f(y)),$$

$x \neq y$, then f is ***strictly quasiconvex****.*

Consider the function f in (3.1), which is defined on a subset $X = X_1 \times X_2 \times \cdots \times X_m$, we say that f is quasiconvex with respect to $x_i \in X_i$ on X if for every $x \in X$ and $y_i \in X_i$, we have

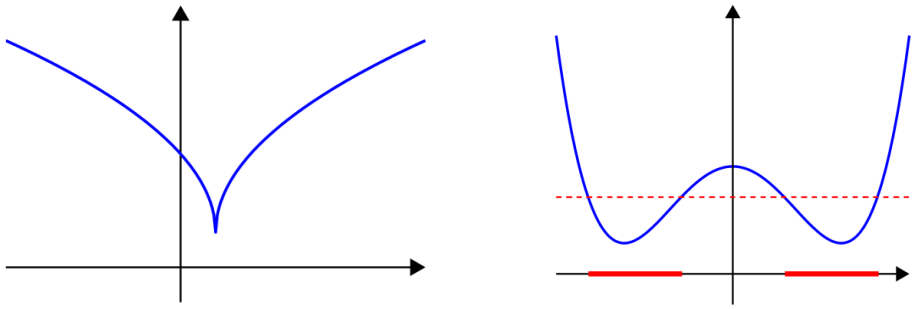
$$f(x_1, \ldots, tx_i + (1-t)y_i, \ldots, x_m) \leq \max\{f(x), f(x_1, \ldots, y_i, \ldots, x_m)\},$$

for all $t \in (0, 1)$. If furthermore,

$$f(x_1, \ldots, tx_i + (1-t)y_i, \ldots, x_m) < \max\{f(x), f(x_1, \ldots, y_i, \ldots, x_m)\},$$

with $y_i \neq x_i$, then f is strictly quasiconvex.

Clearly, a convex function is a quasiconvex function. However, a quasiconvex function may not be a convex function (see the following Figure 4.2a).



(a) Quasiconvex function, but not convex

(b) Not a quasiconvex function

Figure 4.2: **Examples of quasiconvex and non-quasiconvex functions**

Theorem 3.5 (Optimality Condition) *(a) If $\overline{\mathbf{x}}$ is a local minimum of f over X, then it satisfies the optimality condition (3.3), i.e.,*

$$\nabla f(\overline{\mathbf{x}})^T(\mathbf{x} - \overline{\mathbf{x}}) \geq 0, \quad \forall\, \mathbf{x} \in X.$$

(b) If f is convex over X, then the condition of part (a) is also sufficient for $\overline{\mathbf{x}}$ to

minimize f over X.

If $X = \mathbb{R}^n$ or if $\overline{\mathbf{x}}$ is an interior point of X, then the condition is $\nabla f(\overline{\mathbf{x}}) = \mathbf{0}$.

The convergence of the GS method is studied under different assumptions (see e.g. [5, 9, 10, 46, 87, 100]).

Theorem 3.6 (see [9]) *Suppose that f is continuously differentiable over the set X of (3.1). Furthermore, suppose that for each i and $\mathbf{x} \in X$, the minimum of*

$$\min_{\xi \in X_i} f(\mathbf{x}_1, \ldots, \mathbf{x}_{i-1}, \xi, \mathbf{x}_{i+1}, \ldots, \mathbf{x}_m)$$

is uniquely attained. If $\mathbf{x}^k$ is the sequence generated by GS, then every limit point of $\mathbf{x}^k$ is a critical point.

Theorem 3.7 (see [46]) *Suppose that the function f in (3.1) is strictly quasiconvex with respect to $\mathbf{x_i}$ on X, for each $i = 1, \ldots, m-2$ in the sense of definition 3.4 and that the sequence $\{\mathbf{x}^k\}$ generated by the GS method has limit points. Then, every limit point $\overline{x}$ of $\{\mathbf{x}^k\}$ is a critical point of (3.1).*

These theorems show that the GS method can produce a converging sequence with limit points that are critical points of the problem. But, in general, the GS method may not converge, in the sense that it may generate a sequence with limit points that are not critical points of the original problem. A counterexample of Powell [104] (also see [46]) shows that for a non-convex function, which is component-wise convex but not strictly quasiconvex with respect to its each component, the limit points obtained by GS method need not be critical points.

3.2 Some analysis about ALS

From the CP decomposition (2.5) and the definition of rank-one tensor, we have

$$\|\mathcal{T} - \widehat{\mathcal{T}}\|_F^2 = \sum_{i_1}^{I_1}\sum_{i_2}^{I_2}\cdots\sum_{i_N}^{I_N}(t_{i_1 i_2 \ldots i_N} - \sum_{r=1}^{R} a_{i_1 1}^{(r)} a_{i_2 2}^{(r)} \cdot a_{i_N N}^{(r)})^2.$$

Then, $\|\mathcal{T} - \widehat{\mathcal{T}}\|_F^2$ is a function on $\mathbf{x}$ to $\mathbb{R}$, and

$$\mathbf{x} = vec([vec(\mathbf{A}_1), vec(\mathbf{A}_2), \ldots, vec(\mathbf{A}_N)]) \in \mathbb{R}^n,$$

where $\mathbf{A}_i$ is the factor matrix of $\widehat{\mathcal{T}}$ and $n = (I_1 + I_2 + \cdots + I_N)R$. The $vec(\cdot)$ is vectorization defined in Chap. 1.

Let $vec(\mathbf{A}_i) = \mathbf{x}_i \in \mathbb{R}^{I_i R}$, $i = 1, 2, \ldots, N$, so that we partition the vector $\mathbf{x} \in \mathbb{R}^n$ into N component vectors $\mathbf{x}_i \in \mathbb{R}^{n_i}$, $i = 1, 2, \ldots, N$, where $n_i = I_i R$. It follows that $\mathbf{x} = \mathbf{x}_1 \times \mathbf{x}_2 \times \cdots \times \mathbf{x}_N \in \mathbb{R}^{n_1} \times \mathbb{R}^{n_2} \times \cdots \times \mathbb{R}^{n_N} = \mathbb{R}^n$. Thus, the CP decomposition can be reformulated to the following problem,

$$\begin{aligned} &\text{minimize} \quad f(\mathbf{x}) = \sum_{i_1}^{I_1}\sum_{i_2}^{I_2}\cdots\sum_{i_N}^{I_N}(t_{i_1 i_2 \ldots i_N} - \sum_{r=1}^{R} a_{i_1 1}^{(r)} a_{i_2 2}^{(r)} \cdot a_{i_N N}^{(r)})^2 \\ &\text{subject to} \quad \mathbf{x} \in \mathbb{R}^{n_1} \times \mathbb{R}^{n_2} \times \cdots \times \mathbb{R}^{n_N} = \mathbb{R}^n. \end{aligned} \tag{3.4}$$

According to the ALS algorithm, the updates are in terms of the components of $\mathbf{x}$, starting from a given initial point $\mathbf{x}^0 = vec([vec(\mathbf{A}_1^0), vec(\mathbf{A}_2^0), \ldots, vec(\mathbf{A}_N^0)]) \in \mathbb{R}^n$, and a sequence $\{(\mathbf{x}_1^k, \mathbf{x}_2^k, \ldots, \mathbf{x}_N^k)\}$ is generated by the following equation

$$\mathbf{x}_i^{k+1} = \underset{\mathbf{y}_i \in \mathbb{R}^{n_i}}{\operatorname{argmin}} f(\mathbf{x}_1^{k+1}, \ldots, \mathbf{x}_{i-1}^{k+1}, \mathbf{y}_i, \mathbf{x}_{i+1}^k, \ldots, \mathbf{x}_N^k). \tag{3.5}$$

Notice that this is the exact GS method. Therefore, the ALS algorithm is the block nonlinear Gauss-Seidel method for solving the CP decomposition of a given tensor. Since we have studied some convergence properties of the GS method, then

we can use these GS results to analyze the ALS algorithm.

Recall that the nth subproblem of ALS method at kth iteration is

$$\min_{\widehat{\mathbf{A}}_n \in \mathbb{R}^{I_n \times R}} \|\mathbf{T}_{(n)} - \widehat{\mathbf{A}}_n (\mathbf{A}_N^k \odot \mathbf{A}_{N-1}^k \odot \cdots \odot \mathbf{A}_{n+1}^k \odot \mathbf{A}_{n-1}^{k+1} \odot \cdots \odot \mathbf{A}_1^{k+1})^{\mathrm{T}}\|^2.$$

It is a least-squares problem. Comparing the convergence results of the GS method with the least-squares cost functionals, we observe that neither one of the hypotheses in theorem 3.6 or the theorem 3.7 is satisfied. Indeed, the least-squares cost functional is convex (even quadratic) in each component and therefore, quasiconvex. However, in the case that the Kathri-Rao product of two factor matrices involved is rank deficient, then the least-squares function will not be strictly quasiconvex (see the following proposition).

Proposition 3.8 *Let* $f(x) = \|\mathbf{A}\mathbf{x} - \mathbf{b}\|^2$ *where* $\mathbf{A} \in \mathbb{R}^{m \times n}$, $m > n$, $\mathbf{x} \in \mathbb{R}^n$ *and* $\mathbf{b} \in \mathbb{R}^m$. *If* $\mathbf{A}$ *is rank deficient, then* $f(\mathbf{x})$ *is not strictly convex.*

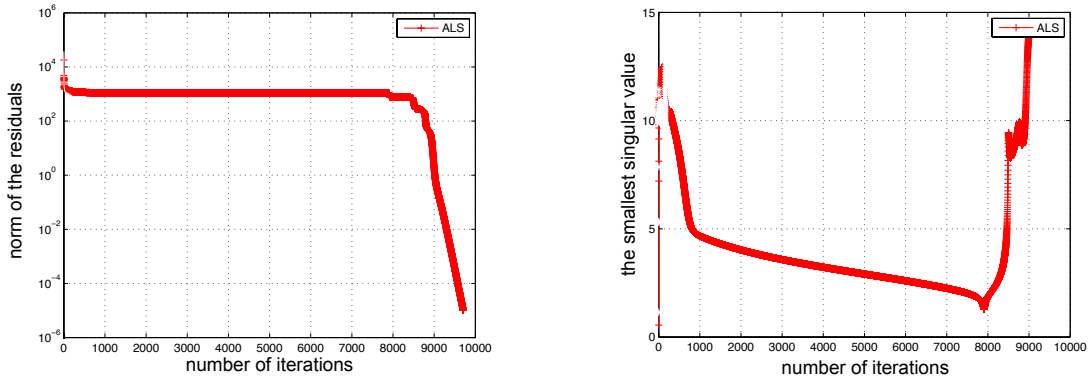
Proof. Since $\mathbf{A}$ is rank deficient, then assume $\text{rank}(\mathbf{A}) = r$ which implies that $\dim(\text{Nul}(\mathbf{A})) = n - r$. Take $\mathbf{x}, \tilde{\mathbf{x}} \in \text{Nul}(\mathbf{A})$ where $\mathbf{x} \neq \tilde{\mathbf{x}}$. Then, according to the definition of a strictly convex function, for any $t \in [0, 1]$, $f(t\mathbf{x} + (1-t)\widetilde{\mathbf{x}}) = \|\mathbf{A}[t\mathbf{x} + (1-t)\widetilde{\mathbf{x}}] - \mathbf{b}\|^2 = \|\mathbf{b}\|^2$ and $tf(\mathbf{x}) + (1-t)f(\widetilde{\mathbf{x}}) = \|\mathbf{b}\|^2$. Thus, $f(t\mathbf{x} + (1-t)\widetilde{\mathbf{x}}) = tf(\mathbf{x}) + (1-t)f(\widetilde{\mathbf{x}})$. □

It follows from the proposition above that f is not a strictly quasiconvex function since $f(t\mathbf{x} + (1-t)\widetilde{\mathbf{x}}) = f(\mathbf{x}) = f(\widetilde{\mathbf{x}})$. Thus from theorem 3.7, a limit point of the ALS sequence is not guaranteed to be a critical point.

3.3 ALS swamp

We have seen in the previous section that there are drawbacks of the ALS method: it is not guaranteed to converge to a global minimum or even a critical point of (2.5),

only to a solution where the objective function ceases to decrease. Also we have seen the numerical results heavily depend on the initial guesses. In this section, we discuss another artifact of the ALS algorithm which is the so-called ***swamp***. Swamp behavior occurs when there are exceedingly high number of iterations causing the convergence rate to slow down dramatically (see [91]). Figure 4.3a shows a swamp behavior occures in the ALS method for a given third-order tensor of size $5 \times 5 \times 5$.



(a) It takes almost 8000 iterations to make the error converge.

(b) Smallest singular values of $\mathbf{C}^k \odot \mathbf{B}^k$

Figure 4.3: **Numerical example for swamp in ALS**

Theoretically, we have shown that the least-squares functional is not strictly quasi-convex when the coefficient matrix is rank deficient, and thus the sequence generated by ALS method cannot be guaranteed to converge to a critical point of the original cost function. This indicates one reason for the occurrence of swamps, namely if the Khatri-Rao products (coefficient matrices) are almost singular, the associated least-squares functional will be flat. Such analysis is verified by the numerical experiment where swamps have been observed when the component matrices are ill-conditioned, or when collinearity occurs in the columns of these factor matrices (the coefficient matrix is rank deficient). For the example in 4.3a, the Figure 4.3b shows the singularity of the corresponding coefficient matrices ($\mathbf{C}^k \odot \mathbf{B}^k$).

4 Regularized ALS

Some disadvantages of the ALS method are shown and discussed in the previous section. So, several techniques have been proposed for improving the efficiency of ALS (see [121]). For example, several researchers have proposed improving ALS with line searches, including the ELS approach of Rajih and Comon [106], which adds a line search after each major iteration that updates all component matrices simultaneously based on the standard ALS search directions (see also [92]). Navasca, De Lathauwer, and Kinderman [91] proposed adding a regularized term to each subproblem for increasing the convergence rate. In this section, we examine the method proposed in [91], a regularization of the ALS, which we denote by RALS. This technique is also called the proximal point modification of the Gauss-Seidel method (PGS) (see [9], [46]). In practice, RALS converges faster than ALS and decreases the high number of ALS iterations, thereby removing the swamp.

4.1 Tikhonov regularization

Some drawbacks of ALS comes from the least-square problem, as we have seen that the rank deficiency of coefficient matrix $\mathbf{C}^k \odot \mathbf{B}^k$ causes the swamp in Figure 4.3. Therefore, before we talk about the regularized technique in [91], it is necessary to introduce a very important and popular regularization method for solving an ill-posed problem. When we want to solve such a problem,

$$\mathbf{A}\mathbf{x} = \mathbf{b}, \tag{4.1}$$

where $\mathbf{A}$ is an $m \times n$ matrix with a large condition number, $m \geq n$, and $\mathbf{b} = \mathbf{b}_t + \epsilon$ consists of true data plus a noise. A most common and well-known regularization

method is called Tikhonov (see [120]) which can be presented as

$$\min_{\mathbf{x}\in\mathbb{R}^n} J_\alpha(\mathbf{x}) = \min_{\mathbf{x}\in\mathbb{R}^n}\{\|\mathbf{A}\mathbf{x}-\mathbf{b}\|^2 + \alpha\|\mathbf{x}\|^2\}, \tag{4.2}$$

where $\alpha \in \mathbb{R}$ is called the regularization parameter. So the solution $\mathbf{x}_\alpha$ to (4.2) solves the problem

$$(\mathbf{A}^{\mathrm{T}}\mathbf{A} + \alpha\mathbf{I})\mathbf{x} = \mathbf{A}^{\mathrm{T}}\mathbf{b}, \tag{4.3}$$

where $\mathbf{I}$ is the identity matrix of size $n \times n$. It can be easily shown that for any positive parameter α there exists a unique $\mathbf{x}_\alpha \in \mathbb{R}^n$ for which the function J_α (4.2) attains its minimum [88].

We see that the regularized solution is sought as a minimizer of a weighted combination of the residual norm and a constraint. The regularization parameter gives the control on the minimization of the constraint. Therefore, the quality of the regularized solution is controlled by the regularization parameter. So the central question in Tikhonov regularization is how to choose the parameter α in order to produce a solution $\mathbf{x}_\alpha$ close to the true noise-free solution $\mathbf{x}_t$ which solves the equation $\mathbf{A}\mathbf{x} = \mathbf{b}_t$.

There are several possible strategies that determine the parameter for the Tikhonov regularization (see [42, 88]). The discrepancy principle is an *a-posteriori* strategy for choosing α as a function of an error level [88]. It has been shown that the regularization method together with this parameter choice rule results in a convergent method; i.e., as the noise level $\epsilon \to 0$, the regularized solution will tend to the true one (see [91, 42]).

Another practical method for choosing α is the L-curve criterion (see [48]). The method is based on the plot of the norm of the regularized solution versus the norm of the corresponding residual. The L-curve criterion is used to choose a regularization parameter related to the L-shaped "corner" of the graph.

Therefore, the Tikhonov regularization has already been used in the ALS algorithm to improve its efficiency [84, 96]. The Tikhonov functional of the original problem (2.3) to be minimized is

$$\|\mathcal{T} - \sum_{r=1}^{R} \mathbf{a}_r \circ \mathbf{b}_r \circ \mathbf{c}_r\|_F^2 + \lambda(\|\mathbf{A}\|_F^2 + \|\mathbf{B}\|_F^2 + \|\mathbf{C}\|_F^2). \tag{4.4}$$

If ALS is applied to this regularized functional, then the corresponding subproblem are

$$\begin{aligned}
\mathbf{A}^{k+1} &= \underset{\widehat{\mathbf{A}} \in \mathbb{R}^{I \times R}}{\operatorname{argmin}} \|\mathbf{T}_{(\mathbf{1})}{}^{I \times JK} - \widehat{\mathbf{A}}(\mathbf{C}^k \odot \mathbf{B}^k)^{\mathrm{T}}\|_F^2 + \lambda\|\widehat{\mathbf{A}}\|_F^2, \\
\mathbf{B}^{k+1} &= \underset{\widehat{\mathbf{B}} \in \mathbb{R}^{J \times R}}{\operatorname{argmin}} \|\mathbf{T}_{(\mathbf{2})}{}^{J \times IK} - \widehat{\mathbf{B}}(\mathbf{C}^k \odot \mathbf{A}^{k+1})^{\mathrm{T}}\|_F^2 + \lambda\|\widehat{\mathbf{B}}\|_F^2, \\
\mathbf{C}^{k+1} &= \underset{\widehat{\mathbf{C}} \in \mathbb{R}^{K \times R}}{\operatorname{argmin}} \|\mathbf{T}_{(\mathbf{3})}{}^{K \times IJ} - \widehat{\mathbf{C}}(\mathbf{B}^{k+1} \odot \mathbf{A}^{k+1})^{\mathrm{T}}\|_F^2 + \lambda\|\widehat{\mathbf{C}}\|_F^2.
\end{aligned} \tag{4.5}$$

Observe that each subproblem is exactly the Tikhonov regularization for the original subproblem. The penalization terms, $\|\widehat{\mathbf{A}}\|_F^2$, $\|\widehat{\mathbf{B}}\|_F^2$ and $\|\widehat{\mathbf{C}}\|_F^2$ are independent of k, which are viewed as uniform constraints on the norm of the matrices. From [84], this constrained optimization problem (4.4) always has globally optimal solution. However, the price to pay here is that the optimal solution is not a critical point of the problem (2.3), but it is a critical point of the regularized functional [81].

4.2 Regularized ALS

Now, let us introduce another type of regularized ALS method called RALS. Similar with the discussion of ALS method, we also use third-order tensors to explain how the RALS method works. Recall that the CP decomposition problem for a given third-order tensor $\mathcal{T}$ in (2.3),

$$\min_{\mathbf{A},\mathbf{B},\mathbf{C}} \|\mathcal{T} - \sum_{r=1}^{R} \mathbf{a}_r \circ \mathbf{b}_r \circ \mathbf{c}_r\|_F^2$$

with respect to the three factor matrices $\mathbf{A}$, $\mathbf{B}$ and $\mathbf{C}$. We have recast it to 3 least-squares subproblems (2.4). The RALS method adds the corresponding regularization term to each subproblems,

$$\begin{aligned}
\mathbf{A}^{k+1} &= \underset{\widehat{\mathbf{A}}\in\mathbb{R}^{I\times R}}{\operatorname{argmin}} \|\mathbf{T}_{(\mathbf{1})} - \widehat{\mathbf{A}}(\mathbf{C}^k \odot \mathbf{B}^k)^{\mathrm{T}}\|_F^2 + \lambda_k\|\mathbf{A}^k - \widehat{\mathbf{A}}\|_F^2, \\
\mathbf{B}^{k+1} &= \underset{\widehat{\mathbf{B}}\in\mathbb{R}^{J\times R}}{\operatorname{argmin}} \|\mathbf{T}_{(\mathbf{2})} - \widehat{\mathbf{B}}(\mathbf{C}^k \odot \mathbf{A}^{k+1})^{\mathrm{T}}\|_F^2 + \lambda_k\|\mathbf{B}^k - \widehat{\mathbf{B}}\|_F^2, \\
\mathbf{C}^{k+1} &= \underset{\widehat{\mathbf{C}}\in\mathbb{R}^{K\times R}}{\operatorname{argmin}} \|\mathbf{T}_{(\mathbf{3})} - \widehat{\mathbf{C}}(\mathbf{B}^{k+1} \odot \mathbf{A}^{k+1})^{\mathrm{T}}\|_F^2 + \lambda_k\|\mathbf{C}^k - \widehat{\mathbf{C}}\|_F^2,
\end{aligned} \tag{4.6}$$

where $\lambda_k > 0$ is the regularization parameter. The regularization terms $\lambda_k\|\mathbf{A}^k - \widehat{\mathbf{A}}\|_F^2$, $\lambda_k\|\mathbf{B}^k - \widehat{\mathbf{B}}\|_F^2$ and $\lambda_k\|\mathbf{C}^k - \widehat{\mathbf{C}}\|_F^2$ are the fitting terms for the factors $\mathbf{A}$, $\mathbf{B}$ and $\mathbf{C}$. So, we can see the difference between (4.5) and (4.6), where the fitting terms of the later one dependent on the previous results.

In fact, RALS also gives us three least-squares subproblems. For example, the first subproblem in (4.6) actually is equivalent to solving a least-squares problem:

$$\begin{bmatrix} (\widetilde{\mathbf{C}}^k \odot \widetilde{\mathbf{B}}^k) \\ \lambda_k \cdot \mathbf{I}^{R\times R} \end{bmatrix} \mathbf{X} = \begin{bmatrix} \mathbf{T}_{(\mathbf{1})}{}^{\mathrm{T}} \\ \lambda_k \cdot (\widetilde{\mathbf{A}}^k)^{\mathrm{T}} \end{bmatrix}, \tag{4.7}$$

which is different from the least-squares obtained from ALS, that is,

$$(\mathbf{C}^k \odot \mathbf{B}^k)\mathbf{X} = \mathbf{T}_{(\mathbf{1})}{}^{\mathrm{T}}. \tag{4.8}$$

Here, we use $\widetilde{\mathbf{A}}$, $\widetilde{\mathbf{B}}$ and $\widetilde{\mathbf{C}}$ to denote the factor matrices generated by RALS while $\mathbf{A}$, $\mathbf{B}$ and $\mathbf{C}$ are generated by ALS.

The following Table 4.2 shows the algorithm of RALS. The number of iterations N is set to a large number; and a stopping criterion can be used.

As we said in the introduction, RALS addresses the non-degenerate swamp issues. The Tikhonov regularization of ALS proposed by Paatero in 2000 [96] can deal with

RALS-Algorithm

procedure CP-RALS($\mathcal{T}, R, N, \lambda$)

give initial guess $\mathbf{A}^0 \in \mathbb{R}^{I\times R}$, $\mathbf{B}^0 \in \mathbb{R}^{J\times R}$, $\mathbf{C}^0 \in \mathbb{R}^{K\times R}$, λ_0

for $n = 1, \ldots, N$ **do**

$\mathbf{W} \leftarrow [(\mathbf{C}^n \odot \mathbf{B}^n); \lambda_n \mathbf{I}^{R\times R}] \in \mathbb{R}^{(JK+R)\times R}$

$\mathbf{S} \leftarrow [{\mathbf{T}_{(\mathbf{1})}}^{\mathrm{T}}; \lambda_n (\mathbf{A}^n)^{\mathrm{T}}] \in \mathbb{R}^{(JK+R)\times I}$

$\mathbf{A}^{n+1} \leftarrow \mathbf{W}/\mathbf{S}$ —— % solving least squares to update $\mathbf{A}$

$\mathbf{W} \leftarrow [(\mathbf{C}^n \odot \mathbf{A}^{n+1}); \lambda_n \mathbf{I}^{R\times R}] \in \mathbb{R}^{(IK+R)\times R}$

$\mathbf{S} \leftarrow [{\mathbf{T}_{(\mathbf{2})}}^{\mathrm{T}}; \lambda_n (\mathbf{B}^n)^{\mathrm{T}}] \in \mathbb{R}^{(IK+R)\times J}$

$\mathbf{B}^{n+1} \leftarrow \mathbf{W}/\mathbf{S}$ —— % solving least squares to update $\mathbf{B}$

$\mathbf{W} \leftarrow [(\mathbf{B}^{n+1} \odot \mathbf{A}^{n+1}); \lambda_n \mathbf{I}^{R\times R}] \in \mathbb{R}^{(IJ+R)\times R}$

$\mathbf{S} \leftarrow [{\mathbf{T}_{(\mathbf{3})}}^{\mathrm{T}}; \lambda_n (\mathbf{C}^n)^{\mathrm{T}}] \in \mathbb{R}^{(IJ+R)\times K}$

$\mathbf{C}^{n+1} \leftarrow \mathbf{W}/\mathbf{S}$ —— % solving least squares to update $\mathbf{C}$

$\lambda_{n+1} \leftarrow c \cdot \lambda_n$ —— % update regularization parameter

end for

return $\mathbf{A}^N$, $\mathbf{B}^N$, $\mathbf{C}^N$

end procedure

Table 4.2: **RALS algorithm of CP decomposition with rank R and parameter sequence $\{\lambda_k\}$ for a third-order tensor $\mathcal{X} \in \mathbb{R}^{I\times J\times K}$**

the degenerate case.

According to [70], a tensor is ***degenerate*** if it may be approximated arbitrarily well by a factorization of lower rank. Both [96] and [38] provide several degenerate models. Here we give one which is presented in [70].

Example 4.1 (Degenerate example) Let $\mathbf{A} = [\mathbf{a}_1\ \mathbf{a}_2] \in \mathbb{R}^{I\times 2}$, $\mathbf{B} = [\mathbf{b}_1\ \mathbf{b}_2] \in \mathbb{R}^{J\times 2}$ and $\mathbf{C} = [\mathbf{c}_1\ \mathbf{c}_2] \in \mathbb{R}^{K\times 2}$, and let $\mathcal{X} \in \mathbb{R}^{I\times J\times K}$ be a third-order tensor defined by

$$\mathcal{X} = \mathbf{a}_1 \circ \mathbf{b}_1 \circ \mathbf{c}_2 + \mathbf{a}_1 \circ \mathbf{b}_2 \circ \mathbf{c}_1 + \mathbf{a}_2 \circ \mathbf{b}_1 \circ \mathbf{c}_1,$$

where the each matrix has linearly independent columns. $\mathcal{X}$ can be approximated arbitrarily closely by a rank-two tensor of the following form:

$$\mathcal{Y} = \epsilon(\mathbf{a}_1 + \frac{1}{\epsilon}\mathbf{a}_2) \circ (\mathbf{b}_1 + \frac{1}{\epsilon}\mathbf{b}_2) \circ (\mathbf{c}_1 + \frac{1}{\epsilon}\mathbf{c}_2) - \epsilon \cdot \mathbf{a}_1 \circ \mathbf{b}_1 \circ \mathbf{c}_1,$$

where ϵ is an arbitrary positive number. Thus,

$$\|\mathcal{X} - \mathcal{Y}\| = \frac{1}{\epsilon}\left\|\mathbf{a}_2 \circ \mathbf{b}_2 \circ \mathbf{c}_1 + \mathbf{a}_2 \circ \mathbf{b}_1 \circ \mathbf{c}_2 + \mathbf{a}_1 \circ \mathbf{b}_2 \circ \mathbf{c}_2 + \frac{1}{\epsilon}\mathbf{a}_2 \circ \mathbf{b}_2 \circ \mathbf{c}_2\right\|,$$

can be made arbitrarily small. Therefore, such a rank-two third-order tensor can be approximated well by a rank-one tensor.

In [96], several numerical examples show that such regularization can help the algorithm keep distance from the degenerate swamps, i.e., the degenerate regions where convergence is slow.

4.3 Regularized parameter choice

Paper [91] discusses the regularized parameter choice for RALS. It basically uses the discrepancy principle to find a geometrically decaying sequence $\lambda_k = q^k$ with $0 < q < 1$ so that the discrepancy principle essentially terminates the procedure when the parameter has the same level with the noise. This method works very well. The parameters are different at different iterations but are the same for the three subproblems at the same iteration. Following this idea, we can make a more general parameter choice which provides three geometrically decaying sequences $\alpha_k = q_1^k$, $\beta_k = q_2^k$ and $\gamma_k = q_3^k$ with $0 < q_1, q_2, q_3 < 1$, where α_k is for $\mathbf{A}^k$, β_k is for $\mathbf{B}^k$ and γ_k is for $\mathbf{C}^k$.

Next, we will use L-curve method to determine q_1, q_2 and q_3. We show that the each subproblem also can be considered as a Tikhonov regularization.

Recall that the original subproblems of CP decomposition by the ALS method are

least-squares in terms of the corresponding factor matrix. The RALS adds an extra term to each subproblem so that we get a regularization method. Let us look at the first equation of (4.6) at the kth iteration,

$$\begin{aligned} g(\widehat{\mathbf{A}}) &= \|\mathbf{T}_{(\mathbf{1})} - \widehat{\mathbf{A}}(\mathbf{C}^k \odot \mathbf{B}^k)^{\mathrm{T}}\|_F^2 + \lambda_k \|\mathbf{A}^k - \widehat{\mathbf{A}}\|_F^2 \qquad (4.9) \\ &= \|\mathbf{T}_{(\mathbf{1})} - \mathbf{A}^k(\mathbf{C}^k \odot \mathbf{B}^k)^{\mathrm{T}} - (\widehat{\mathbf{A}} - \mathbf{A}^k)(\mathbf{C}^k \odot \mathbf{B}^k)^{\mathrm{T}}\|_F^2 + \lambda_k \|\mathbf{A}^k - \widehat{\mathbf{A}}\|_F^2 \\ &= h(\widehat{\mathbf{A}} - \mathbf{A}^k). \end{aligned}$$

So, the function $g(\widehat{\mathbf{A}})$ is also a function of $(\widehat{\mathbf{A}} - \mathbf{A}^k)$. Then the first equation of (4.6) is a Tikhonv regularization with parameter λ_k of the least-squares problem

$$\|[\mathbf{T}_{(\mathbf{1})} - \mathbf{A}^k(\mathbf{C}^k \odot \mathbf{B}^k)^{\mathrm{T}}] - (\widehat{\mathbf{A}} - \mathbf{A}^k)(\mathbf{C}^k \odot \mathbf{B}^k)^{\mathrm{T}}\|_F^2,$$

and the regularization term is just $\lambda_k \|\mathbf{A}^k - \widehat{\mathbf{A}}\|_F^2$. Note, however, that RALS can not be considered a Tikhonov regularization, but each subproblem can be considered as a Tikhonov regularization.

Since each subproblem is a classical Tikhonov regularization, we can use L-curve to find the corresponding regularization parameter for each subproblem at every iteration k. Therefore, we can obtain α_k, β_k and γ_k. By running a large number of simulations, we can find the ratio for each sequence.

4.4 Proximal point modification of the Gauss-Seidel method

In the preceding section, we have shown that the GS method may not converge or it needs some convexity/quasiconvexity assumption to guarantee the convergence results. So, for solving problem (3.1), Grippo and Sciandrone consider a modification

of GS by adding some extra item in each iteration,

$$\mathbf{x}_i^{k+1} = \underset{\mathbf{y}_i \in X_i}{\operatorname{argmin}} f(\mathbf{x}_1^{k+1}, \ldots, \mathbf{y}_i, \ldots, \mathbf{x}_m^k) + \frac{1}{2}\tau_i \|\mathbf{y}_i - \mathbf{x}_i^k\|^2. \tag{4.10}$$

This method is called a proximal point modification of the GS (PGS) method (see [9, 46]). It is also referred as partial proximal minimization [10]. The advantage of PGS is that it does not require the convexity assumption for convergence to critical points.

Definition 4.2 *The GS and PGS methods are well-defined if every subproblem has solutions.*

Proposition 4.3 (Convergence proposition of PGS (see [46])) *Suppose that the PGS method is well defined and that the sequence* $\{\mathbf{x}^k\}$ *has limit points. Then every limit point* $\overline{\mathbf{x}}$ *of* $\{\mathbf{x}^k\}$ *is a critical point of problem (3.1).*

In last section, we showed that the ALS method is the GS method for solving CP decomposition with respect to the factor matrices $\mathbf{A}$, $\mathbf{B}$ and $\mathbf{C}$. By using the same technique for RALS (4.6), through vectorization of the three factor matrices, we have

$$\begin{aligned}
\mathbf{x}_1^{k+1} &= \underset{\mathbf{y}_1 \in \mathbb{R}^{n_1}}{\operatorname{argmin}} \{f(\mathbf{y}_1, \mathbf{x}_2^k, \mathbf{x}_3^k) + \lambda_k \|\mathbf{x}_1^k - \mathbf{y}_1\|_F^2\}, \\
\mathbf{x}_2^{k+1} &= \underset{\mathbf{y}_2 \in \mathbb{R}^{n_2}}{\operatorname{argmin}} \{f(\mathbf{x}_1^{k+1}, \mathbf{y}_2, \mathbf{x}_3^k) + \lambda_k \|\mathbf{x}_2^k - \mathbf{y}_2\|_F^2\}, \\
\mathbf{x}_3^{k+1} &= \underset{\mathbf{y}_3 \in \mathbb{R}^{n_3}}{\operatorname{argmin}} \{f(\mathbf{x}_1^{k+1}, \mathbf{x}_2^{k+1}, \mathbf{y}_3) + \lambda_k \|\mathbf{x}_3^k - \mathbf{y}_3\|_F^2\}.
\end{aligned}$$

Observe that the regularized ALS is the same as PGS method. Now, we can also analyze the convergence property of RALS through the properties of the PGS method.

5 Convergence results of RALS

We have already mentioned that even though the ALS algorithm can generate a converging sequence of factor matrices $\{(\mathbf{A}^k, \mathbf{B}^k, \mathbf{C}^k)\}$, it cannot guarantee that the limit point is a critical point (local minimum). In this section, we will show that the converging sequence obtained from the RALS method does give us a critical point.

We adapt the proposition in Section 7 of the paper [46] to our problem to give a convergence result for the RALS method.

Theorem 5.1 *Suppose that the sequence* $\{(\mathbf{A}^k, \mathbf{B}^k, \mathbf{C}^k)\}$ *obtained from RALS has limit points. Then every limit point* $(\overline{\mathbf{A}}, \overline{\mathbf{B}}, \overline{\mathbf{C}})$ *is a critical point of the Problem 2.3.*

Proof. Recall the vectorization of a matrix which allows us to re-express $\{(\mathbf{A}^k, \mathbf{B}^k, \mathbf{C}^k)\}$ as $(\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_3)$ and the cost function as

$$f(\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_3) = \sum_{k=1}^{K} \sum_{j=1}^{J} \sum_{i=1}^{I} (t_{ijk} - \sum_{r=1}^{R} a_{ir} b_{jr} c_{kr})^2,$$

where $\mathbf{x}_1 = vec(\mathbf{A}) \in \mathbb{R}^{IR}$, $\mathbf{x}_2 = vec(\mathbf{B}) \in \mathbb{R}^{JR}$ and $\mathbf{x}_3 = vec(\mathbf{C}) \in \mathbb{R}^{KR}$. Let $\{\mathbf{x}^{n_k}\}_{k=1}^{\infty} = \{(\mathbf{x}_1^{n_k}, \mathbf{x}_2^{n_k}, \mathbf{x}_3^{n_k})\}_{k=1}^{\infty}$ be the converging subsequence of $\{(\mathbf{x}_1^k, \mathbf{x}_2^k, \mathbf{x}_3^k)\}$ which has the limit point $(\overline{\mathbf{x}}_1, \overline{\mathbf{x}}_2, \overline{\mathbf{x}}_3)$.

The subproblem in the RALS method provides the following inequality:

$$f(\mathbf{x}_1^{n_k+1}, \mathbf{x}_2^{n_k}, \mathbf{x}_3^{n_k}) \leq f(\mathbf{x}_1^{n_k}, \mathbf{x}_2^{n_k}, \mathbf{x}_3^{n_k}) + \lambda_{n_k} \|\mathbf{x}_1^{n_k+1} - \mathbf{x}_1^{n_k}\|^2. \tag{5.1}$$

Applying the inequality above repeatedly, we have

$$\begin{aligned} f(\mathbf{x}_1^{n_k+1}, \mathbf{x}_2^{n_k+1}, \mathbf{x}_3^{n_k+1}) &\leq f(\mathbf{x}_1^{n_k+1}, \mathbf{x}_2^{n_k+1}, \mathbf{x}_3^{n_k}) \\ &\leq f(\mathbf{x}_1^{n_k+1}, \mathbf{x}_2^{n_k}, \mathbf{x}_3^{n_k}) \\ &\leq f(\mathbf{x}_1^{n_k}, \mathbf{x}_2^{n_k}, \mathbf{x}_3^{n_k}). \end{aligned} \tag{5.2}$$

By the Squeeze Theorem, the continuity of f, and the condition that $(\mathbf{x}_1^{n_k}, \mathbf{x}_2^{n_k}, \mathbf{x}_3^{n_k}) \longrightarrow (\overline{\mathbf{x}}_1, \overline{\mathbf{x}}_2, \overline{\mathbf{x}}_3)$ as $k \to \infty$, then we have the following

$$\lim_{k\to\infty} f(\mathbf{x}_1^{n_k+1}, \mathbf{x}_2^{n_k}, \mathbf{x}_3^{n_k}) = \lim_{k\to\infty} f(\mathbf{x}_1^{n_k}, \mathbf{x}_2^{n_k}, \mathbf{x}_3^{n_k}) = f(\overline{\mathbf{x}}_1, \overline{\mathbf{x}}_2, \overline{\mathbf{x}}_3).$$

Now taking the limits in (5.1) for $k \to \infty$ on both sides, we have

$$\lim_{k\to\infty} \|\mathbf{x}_1^{n_k+1} - \mathbf{x}_1^{n_k}\|^2 = 0 \tag{5.3}$$

which implies

$$\lim_{k\to\infty} (\mathbf{x}_1^{n_k+1}, \mathbf{x}_2^{n_k}, \mathbf{x}_3^{n_k}) = (\overline{\mathbf{x}}_1, \overline{\mathbf{x}}_2, \overline{\mathbf{x}}_3). \tag{5.4}$$

Similarly, we obtain

$$\lim_{k\to\infty} (\mathbf{x}_1^{n_k+1}, \mathbf{x}_2^{n_k+1}, \mathbf{x}_3^{n_k}) = (\overline{\mathbf{x}}_1, \overline{\mathbf{x}}_2, \overline{\mathbf{x}}_3). \tag{5.5}$$

Since every RALS subproblem is well defined, then each point in the subsequence satisfies the corresponding optimality condition (Theorem 3.5); i.e.

$$\nabla_1 f(\mathbf{x}_1^{n_k+1}, \mathbf{x}_2^{n_k}, \mathbf{x}_3^{n_k}) + 2\lambda_{n_k}(\mathbf{x}_1^{n_k+1} - \mathbf{x}_1^{n_k}) = 0, \tag{5.6}$$

$$\nabla_2 f(\mathbf{x}_1^{n_k+1}, \mathbf{x}_2^{n_k+1}, \mathbf{x}_3^{n_k}) + 2\lambda_{n_k}(\mathbf{x}_2^{n_k+1} - \mathbf{x}_2^{n_k}) = 0, \tag{5.7}$$

$$\nabla_3 f(\mathbf{x}_1^{n_k+1}, \mathbf{x}_2^{n_k+1}, \mathbf{x}_3^{n_k+1}) + 2\lambda_{n_k}(\mathbf{x}_3^{n_k+1} - \mathbf{x}_3^{n_k}) = 0. \tag{5.8}$$

Then, taking $k \to \infty$ in (5.6–5.8), using the arguments in (5.3), (5.4), (5.5) and the continuity of ∇f, we obtain

$$\nabla_i f(\overline{\mathbf{x}}_1, \overline{\mathbf{x}}_2, \overline{\mathbf{x}}_3) = 0, \quad i = 1, 2, 3.$$

Thus, this proves that the limit point $(\overline{\mathbf{x}}_1, \overline{\mathbf{x}}_2, \overline{\mathbf{x}}_3)$ is a critical point of the cost function $f(\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_3)$. Furthermore, we obtain

$$\begin{aligned} \nabla_{\mathbf{A}} f(\overline{\mathbf{A}}, \overline{\mathbf{B}}, \overline{\mathbf{C}}) &= 0, \\ \nabla_{\mathbf{B}} f(\overline{\mathbf{A}}, \overline{\mathbf{B}}, \overline{\mathbf{C}}) &= 0, \\ \nabla_{\mathbf{C}} f(\overline{\mathbf{A}}, \overline{\mathbf{B}}, \overline{\mathbf{C}}) &= 0. \end{aligned} \tag{5.9}$$

through the inverse mapping of the vectorization. Therefore, $(\overline{\mathbf{A}}, \overline{\mathbf{B}}, \overline{\mathbf{C}})$ is a critical point. □

Here are some remarks:

1. Following from the discussion and the theorem above, we showed that the RALS method solves the same cost function, i.e., minimizing the distance between the given tensor and the approximation tensor. Moreorver, we proved that the limit point obtained from RALS is a critical point of the original minimization problem of $\|\mathcal{T} - \widehat{\mathcal{T}}\|_F^2$.

2. The main theorem above solves the CP decomposition on the whole space, so we use the optimality condition, $\nabla f(\overline{\mathbf{x}}_1, \overline{\mathbf{x}}_2, \overline{\mathbf{x}}_3) = 0$. If the solution is not in the whole space, namely, in the problem of non-negative tensor decomposition, then the optimality condition,$\nabla f(\overline{\mathbf{x}}_1, \overline{\mathbf{x}}_2, \overline{\mathbf{x}}_3)^{\mathrm{T}}(\mathbf{y} - \overline{\mathbf{x}}_i) \geq 0$, must be used.

3. For the ALS method, under the same assumption in Theorem 5.1, the theorem may not be true. From the assumption, we know that the sequence $\{(\mathbf{A}^{n_k}, \mathbf{B}^{n_k}, \mathbf{C}^{n_k})\}$ converges to a limit point $(\overline{\mathbf{A}}, \overline{\mathbf{B}}, \overline{\mathbf{C}})$, but we cannot obtain the sequences $\{(\mathbf{A}^{n_k+1}, \mathbf{B}^{n_k}, \mathbf{C}^{n_k})\}$ and $\{(\mathbf{A}^{n_k+1}, \mathbf{B}^{n_k+1}, \mathbf{C}^{n_k})\}$ to converge. Furthermore, we also cannot prove that these two sequences converge to the same limit point $(\overline{\mathbf{A}}, \overline{\mathbf{B}}, \overline{\mathbf{C}})$.

4. The optimality conditions in (5.9) are equivalent to the normal equations of the subproblems:

$$\mathbf{T}_{(\mathbf{1})}{}^{I\times JK}(\overline{\mathbf{C}}\odot\overline{\mathbf{B}}) = \overline{\mathbf{A}}(\overline{\mathbf{C}}\odot\overline{\mathbf{B}})^{\mathrm{T}}(\overline{\mathbf{C}}\odot\overline{\mathbf{B}}),$$
$$\mathbf{T}_{(\mathbf{2})}{}^{J\times IK}(\overline{\mathbf{C}}\odot\overline{\mathbf{A}}) = \overline{\mathbf{B}}(\overline{\mathbf{C}}\odot\overline{\mathbf{A}})^{\mathrm{T}}(\overline{\mathbf{C}}\odot\overline{\mathbf{A}}),$$
$$\mathbf{T}_{(\mathbf{3})}{}^{K\times IJ}(\overline{\mathbf{B}}\odot\overline{\mathbf{A}}) = \overline{\mathbf{C}}(\overline{\mathbf{B}}\odot\overline{\mathbf{A}})^{\mathrm{T}}(\overline{\mathbf{B}}\odot\overline{\mathbf{A}}).$$

5. Theorem 5.1 is a *conditional* convergence proof, depending upon the existence of the ALS limit points. Thus this result does not address the degeneracy problems. Analysis of the existence of the limits of the (R)ALS is a challenging problem that would require a careful study of the degenerate cases of the CP decomposition. The regularization (4.4) considered by Paatero [96] is a good approach in finding approximation to the degenerate case, but the solutions satisfy the *regularized* cost functional and not the original least-squares functional. Moreover, a similar conditional convergent analysis [46] can be established for the regularized functional (4.4). In fact, if $\lambda > 0$, then the cost functional (4.4) will be component-wise strictly quasiconvex. Thus Theorem 3.7 applies and hence, the limit points of (4.5) will be critical points of the regularized functional (4.4).

6 Numerical examples

In this section, we compare the performance of RALS against ALS. We give three examples of third-order tensor CP decomposition to demonstrate the swamp shortening property of the iterated regularization and one example of large real third-order tensor data.

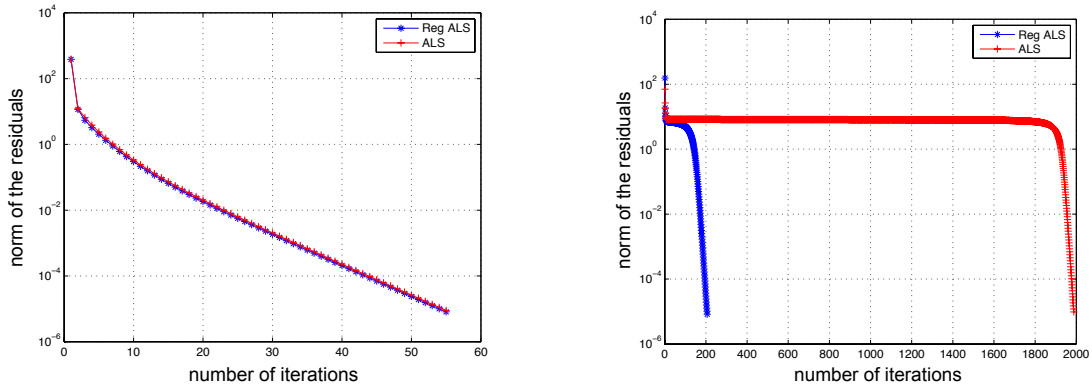
Example 6.1 Initial Factors Dependent Swamp

Let the matrices

$$\mathbf{A} = \begin{bmatrix} 1 & 2 \\ 2 & 1 \\ 3 & 2 \end{bmatrix}, \quad \mathbf{B} = \begin{bmatrix} 2 & 1 \\ -1 & 3 \\ 1 & -1 \end{bmatrix}, \quad \mathbf{C} = \begin{bmatrix} 3 & 1 \\ 1 & 2 \\ 2 & 2 \end{bmatrix},$$

be the three factor matrices of a third-order tensor $\mathcal{T} \in \mathbb{R}^{3\times3\times3}$ of rank-two:

$$\mathcal{T} = \mathbf{a}_1 \circ \mathbf{b}_1 \circ \mathbf{c}_1 + \mathbf{a}_2 \circ \mathbf{b}_2 \circ \mathbf{c}_2.$$



(a) right initial guess (b) random initial guess

Figure 4.4: **Plots for the example 6.1**

In the two figures, the plots show the error $\|\mathcal{T} - \mathcal{T}_{est}\|_F^2$ versus the number of iterations it takes to obtain an error of 10^{-5}, where $\mathcal{T}_{est}$ denotes the obtained tensor after every iteration.

Two initial guesses are compared in Figure 4.4a and Figure 4.4b in terms of ALS. In Figure 4.4a, $\mathbf{A}^0 = \mathbf{A}$, $\mathbf{B}^0 = \mathbf{B}\begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}$ and $\mathbf{C}^0 = \mathbf{C}$. For Figure 4.4b, we randomly generated 3×2 matrices as the initial factors. With $\left\{\mathbf{A}, \mathbf{B}\begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}, \mathbf{C}\right\}$ as the initial

guess, ALS takes 55 iterations to reach an error within 10^{-5} while it takes 1988 iterations by using random initial guess. Observe in Figure 4.4a that ALS and RALS have the same convergence speed and take the same number of iterations to reach an error within 10^{-5}. However, in Figure 4.4b, RALS can reduce the swamp by only taking 206 iterations in comparison to that of 1988 ALS iterations. Moreover, the RALS is faster than ALS since the CPU time of RALS is 0.2982s while the ALS is 2.6547s.

In some cases, randomly generated factors can lead to swamp in the implementation of the ALS. However this swamp phenomena induced by the initial factors is not observed if the RALS method is used.

Example 6.2 Rank Specific Swamp

Let the matrices

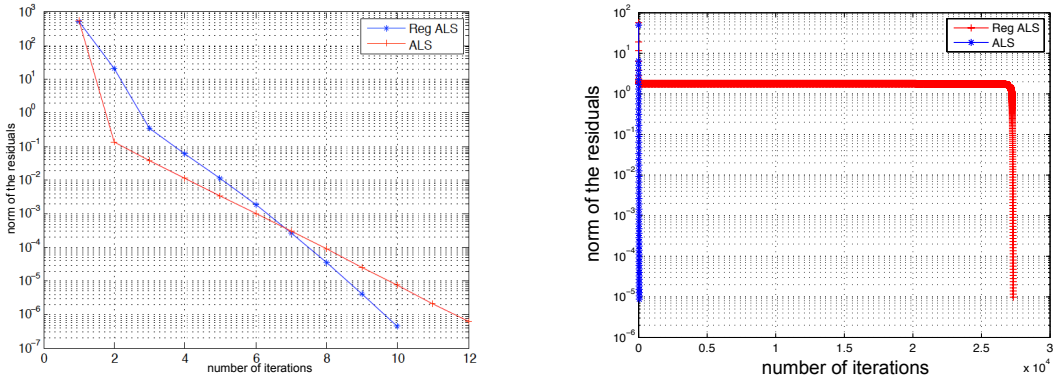
$$\mathbf{A} = \begin{bmatrix} 1 & 2 & 3 \\ 2 & 1 & 2 \end{bmatrix}, \quad \mathbf{B} = \begin{bmatrix} 2 & 1 & 1 \\ -1 & 3 & 1 \end{bmatrix}, \quad \mathbf{C} = \begin{bmatrix} 3 & 1 & 2 \\ 1 & 2 & -1 \end{bmatrix}.$$

be the three factor matrices of a third-order tensor $\mathcal{T} = \sum_{r=1}^{3} \mathbf{a}_r \circ \mathbf{b}_r \circ \mathbf{c}_r \in \mathbb{R}^{2\times 2\times 2}$ that is a rank-three tensor. Rank-two (see Figure 4.5b) and rank-three (see Figure 4.5a) approximations are calculated with the following initial matrices:

$$\mathbf{A}^0 = \begin{bmatrix} 0.1679 & 0.7127 \\ 0.9787 & 0.5005 \end{bmatrix}, \quad \mathbf{B}^0 = \begin{bmatrix} 0.4711 & 0.6820 \\ 0.0596 & 0.0424 \end{bmatrix}, \quad \mathbf{C}^0 = \begin{bmatrix} 0.0714 & 0.0967 \\ 0.5216 & 0.8181 \end{bmatrix}.$$

The following picture shows the error plots using ALS and RALS separately:

Notice from Figure 4.5a that the rank-three tensor approximations present no problem in either ALS and RALS. However, in Figure 4.5b, the rank-two tensor approximation requires only 53 iterations RALS to reach an error within 10^{-5} while ALS needs 27322 iterations, which indicates a swamp. We also calculate the CPU



(a) Rank-three (b) Rank-two

Figure 4.5: **Plots for the example 6.2**

time for each ALS and RALS in the swamp situation, which shows that the CPU time of ALS is 41.1603s while the CPU time of RALS is just 0.0928s.

Further investigation is needed to understand the degeneracy problems with respect to the RALS algorithm.

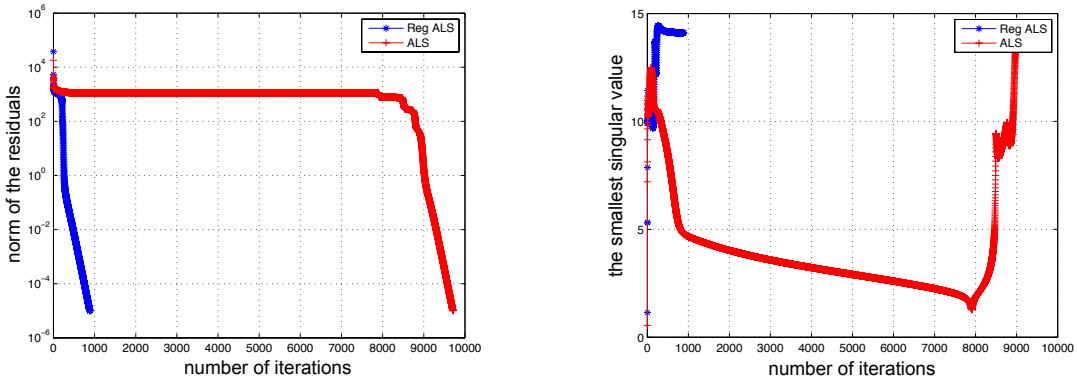
Example 6.3 Induced Rank-Deficiency Swamp

From the example in the Section 3, the RALS and ALS are compared. Recall that the rank deficiency of the Khatri-Rao products induce an ALS swamp. In Figure 4.6a, the error plots show a swamp for ALS with 9707 iterations while RALS exhibits no swamp with only 884 iterations. Here, we also computer the CPU time for both algorithms, the RALS is 2.9720 but the ALS takes 28.5156 to converge.

To understand why RALS is not hampered by a swamp, let us look at the normal equation of the subproblem (we have already mentioned this in the last Section 4.7):

$$\begin{bmatrix} (\widetilde{\mathbf{C}}^k \odot \widetilde{\mathbf{B}}^k) \\ \lambda_k \cdot \mathbf{I}^{R\times R} \end{bmatrix} \mathbf{X} = \begin{bmatrix} {T_{(1)}}^{\mathrm{T}} \\ \lambda_k \cdot (\widetilde{\mathbf{A}}^k)^{\mathrm{T}} \end{bmatrix}$$

where the least squares solution is $\widetilde{\mathbf{A}}^{k+1}$. The submatrix $\lambda_k \cdot \mathbf{I}^{R\times R}$ in (4.7) embeds the range space of $(\widetilde{\mathbf{C}}^k \odot \widetilde{\mathbf{B}}^k)$ in a higher dimensional space while induces a full rank linear



(a) Fifth order tensor (b) Singularity of Khatri-Rao

Figure 4.6: **Plots for the example 6.3**

least-squares subproblem. Thus, the regularization keeps the cost function strictly component-wise quasiconvex.

In Figure 4.6b, we can see that when the swamp happens in the ALS method, the smallest singular value of the coefficient matrix $\mathbf{C}^k \odot \mathbf{B}^k$ is almost zero. Therefore, the coefficient matrix is almost rank deficient and the corresponding least-squares subproblem in the ALS causes the swamp problem.

Example 6.4 Large Real Datasets

Since ALS type algorithms have been particularly useful in real large datasets, a comparison study of the ALS and RALS algorithms was made on a tensor $\mathcal{X} \in \mathbb{R}^{170\times274\times35}$ from the paper of Bro et al. [128] in detecting and characterizing active photosensitizers in butter. The light exposure experimental data is obtained from different colors of light, variation in oxygen availability, and time of exposure while measuring the fluorescence EEMs (excitation emission matrices) and sensory evaluation of the samples. Thus the element x_{ijk} represents the fluorescence intensity for sample i at excitation wavelength j and emission wavelength k. CP algorithms offer decomposition into factors of sample scores, emission loadings and excitation loadings.

The ALS and RALS algorithms were used to analyze the fluorescence landscapes

with rank $R = 7$ and 100 different random initial starters for ALS and RALS were used on tensor $\mathcal{X}$. The error ($\|\mathcal{X}^k - \mathcal{X}^{k-1}\|_F^2$), we call adjacent error, was used as the stopping criterion, but the absolute error ($\|\mathcal{X} - \mathcal{X}_{\text{final}}\|_F^2$) was measured as well. The Table 4.3 shows that RALS performed slightly better than ALS with respect to both relative and absolute errors as well as the number of iterations.

	abs. error	adj. error	iterations
ALS	3850.8	1.2717×10^{-4}	3025
RALS	3832.9	1.3842×10^{-4}	2968

Table 4.3: **The comparison of ALS and RALS for the real data**

7 Conclusion

The RALS method proposed by Navasca, Kindermann and De Lathauwer [91] is a numerical technique for the classical problem of solving the CP decomposition of a given tensor. We examined the RALS method to find some theoretical explanations of what we observed numerically. In many instances, several examples showed that RALS converges faster than ALS. Moreover, RALS decreases the high number of ALS iterations, thereby removing the swamp to some degree. Furthermore, our numerical experiments provide us a numerical justification that ALS swamping is related to the rank deficiency of the Khatri-Rao products. This phenomena is not present when the RALS algorithm is implemented. Based on these observations, it is important to study the theoretical properties of RALS and its differences from ALS. Both the ALS and the RALS are related to the GS and the proximal modification of GS (PGS), respectively, by vectorizing the three factor matrices in the cost functionals. Using the properties of PGS, we have proved that the limit point of a converging sequence obtained from the RALS algorithm is a critical point of the original ALS problem. Some difficulties arise when proving the same convergent results for ALS due to the

lack of strict quasiconvexity. These same difficulties are exhibited numerically as swamps.

Chapter 5

Partial column-wise alternating least-squares

1 Introduction

In this chapter, we study the symmetric outer product decomposition (SOPD) for partially symmetric and fully symmetric tensors, which decomposes a fully (partially) symmetric tensor into a number of rank-one fully (partially) symmetric tensors. Such symmetric tensor decomposition is related to the independent component analysis (ICA) [20, 60], or blind source separation (BSS), which is used to separate a useful signal from noise and interferences in signal processing [31, 22].

The existence of the symmetric tensor decomposition for fully symmetric tensors has been discussed in [21]. It shows that for any order fully symmetric tensor $\mathcal{X}$ over any field, such decomposition always exists. In addition, it shows that such decomposition is different from the CP decomposition for symmetric tensors. There are some strict conditions that can guarantee they are the same.

There are several methods proposed for computing the symmetric tensor decomposition for fully symmetric tensor. Paper [14] introduced a numerical method based

on a multi-variable polynomial set, which was inspired by Sylvester's theorem [78] and reduced the symmetric tensor decomposition to the decomposition of a linear form as a linear combination of evaluations at distinct points. De Lathauwer in [31] proposed the Fourth-Order-Only Blind Identification (FOOBI) algorithm factoring a fourth-order fully symmetric tensor. The algorithm is based on Joint Diagonalization for a set of symmetric matrices. Under a convexity assumption and based on the Higher-Order Power Method (HOPM), Regalia showed a method in [69] to solve rank-one approximation of a higher-order fully symmetric tensor. But this method cannot generalize to rank-R approximation. Another possible way is using the classical ALS method introduced in Chap. 4 to factor the fully symmetric tensors. However, there are several disadvantages. It cannot guarantee that all the vectors in each component are same since it updates every factor matrix separated in each iteration. Additionally, redundant calculation occurs in every iteration, which would cost a lot of unnecessary time.

For the partially symmetric tensor, ALS can also be used to calculate the symmetric tensor decomposition, in which each component is a rank-one partially symmetric tensor. But the same problems happen in the partially symmetric case since a tensor with partially symmetry on some modes has the same factor matrices in the corresponding modes, while the ALS algorithm will calculate each factor in every iteration no matter whether it has been updated or not. Therefore, we reformulate the problem and introduce a new algorithm to decompose the partial symmetric tensors. It is also an iterative method and only calculates the different factor matrices every iteration.

The text is organized as follows. In Section 2, we introduce the partial column-wise alternating least-squares (PCW-ALS) method to compute the symmetric tensor decompositions for third-order partially symmetric tensors and two types of fourth-order partially symmetric tensors. Numerical examples are provided in Section 3 to compare the ALS and PCW-ALS algorithms for both third-order and fourth-order

partially symmetric tensor.

2 PCW-ALS algorithm

2.1 SOPD for third-order partially symmetric tensor

In this section, without loss of generality, we consider the symmetric outer product decomposition of a third-order partially symmetric tensor on mode-1 and mode-2. Let $\mathcal{X} \in \mathbb{R}^{I\times I\times K}$ with $x_{ijk} = x_{jik}$ for all $i, j \in \{1, 2, \ldots, I\}$ and $k \in \{1, 2, \ldots, K\}$, then the SOPD factors such partially symmetric tensor as the sum of a finite number of rank-one partially symmetric tensors:

$$\mathcal{X} = \sum_{r=1}^{R_{ps}} \mathbf{a}_r \circ \mathbf{a}_r \circ \mathbf{c}_r. \tag{2.1}$$

Notice that R_{ps} is not the CP rank of $\mathcal{X}$. For such a partially symmetric tensor $\mathcal{X}$, we can also have a CP decomposition which factors the tensor into R (tensor rank) rank-one tensors:

$$\mathcal{X} = \sum_{r=1}^{R} \bar{\mathbf{a}}_r \circ \bar{\mathbf{b}}_r \circ \bar{\mathbf{c}}_r. \tag{2.2}$$

Since the rank R is defined as the smallest one to make the equation (2.2) holds, then we have $R \leq R_{ps}$. However, for a specific partially symmetric tensor, we do not know Cwhat R and R_{ps} are. Berge et al. [63] studied the partially symmetric tensors of order $2 \times J \times J$ and $I \times 3 \times 3$ and showed that

- Partially symmetric tensors of size $2 \times J \times J$ have typical rank $\{J, J + 1\}$.
- Fully symmetric tensor of size $3 \times 3 \times 3$ has typical rank 4.
- Partially symmetric tensor of size $4 \times 3 \times 3$ has typical rank $\{4, 5\}$.

- Partially symmetric tensor of size $5 \times 3 \times 3$ has typical rank $\{5, 6\}$.

The definition of typical rank is any rank that occurs with probability greater than zero (i.e., on a set with positive Lebesgue measure).

Comon [21] discussed the CP with rank R and symmetric rank R_s for a fully symmetric tensor. It shows that

$$R \leq R_s,$$

for a fully symmetric tensor. In some cases, the equality holds. But it is not known if both ranks are equal on all fully symmetric tensors in general.

Since the difficulty about deciding the number R_{ps}, we can do the SOPD computation for $R_{ps} = 1, 2, 3, \ldots$ until the first value of R_{ps} that can fit the equation (2.1) perfectly. Thus, we assume that the number of components R_{ps} is fixed in the following discussion.

The problem we want to solve is the following: given a third-order partial symmetic tensor $\mathcal{X} \in \mathbb{R}^{I \times I \times K}$ with $x_{ijk} = x_{jik}$, compute its SOPD with R_{ps} components of rank-one partial symmetric tensors that best approximates $\mathcal{X}$. So, it can be considered as an optimization problem:

$$\underset{\widehat{\mathcal{X}}}{\text{minimize}} \quad \left\| \mathcal{X} - \widehat{\mathcal{X}} \right\|_F^2, \quad \text{where} \quad \widehat{\mathcal{X}} = \sum_{r=1}^{R_{ps}} \mathbf{a}_r \circ \mathbf{a}_r \circ \mathbf{c}_r. \tag{2.3}$$

This problem is equivalent to

$$\min_{\mathbf{A},\mathbf{C}} \quad \left\| \mathcal{X} - \sum_{r=1}^{R_{ps}} \mathbf{a}_r \circ \mathbf{a}_r \circ \mathbf{c}_r \right\|_F^2, \tag{2.4}$$

with respect to factor matrices $\mathbf{A} = [\mathbf{a}_1 \ \mathbf{a}_2 \ \cdots \ \mathbf{a}_{R_{ps}}]$ and $\mathbf{C} = [\mathbf{c}_1 \ \mathbf{c}_2 \ \cdots \ \mathbf{c}_{R_{ps}}]$.

Since $\widehat{\mathcal{X}}$ satisfies the SOPD equation (2.1), similarly with the ALS method, we can use the Khatri-Rao product and tensor matricization. Therefore, equation (2.1)

can be written in the following matricized form:

$$\begin{aligned} \widehat{\mathbf{X}} &= \sum_{r=1}^{R_{ps}} \mathbf{a}_r \circ \mathbf{a}_r \circ \mathbf{c}_r \\ &\Downarrow \\ \widehat{\mathbf{X}}_{(\mathbf{3})} &= \mathbf{C}(\mathbf{A} \odot \mathbf{A})^{\mathrm{T}}, \end{aligned}$$

where $\widehat{\mathbf{X}}_{(\mathbf{3})} \in \mathbb{R}^{K \times I^2}$ is the mode-3 matricization of tensor $\widehat{\mathcal{X}}$. $\mathbf{A} \in \mathbb{R}^{I \times R_{ps}}$ and $\mathbf{C} \in \mathbb{R}^{K \times R_{ps}}$.

Then for the original problem (2.4), this problem has the following expression,

$$\min_{\mathbf{A},\mathbf{C}} \quad \left\| \mathbf{X}_{(\mathbf{3})} - \mathbf{C}(\mathbf{A} \odot \mathbf{A})^{\mathrm{T}} \right\|_F^2 . \tag{2.5}$$

The idea is also using alternative method to update the factor matrices $\mathbf{A}$ and $\mathbf{C}$ until some convergence criterion is satisfied. From the above equation, we can fix one variable to solve for the other one each time, then the problem reduces to the following subproblems,

$$\mathbf{A}^{k+1} = \underset{\widehat{\mathbf{A}} \in \mathbb{R}^{I \times R_{ps}}}{\operatorname{argmin}} \left\| \mathbf{X}_{(\mathbf{3})} - \mathbf{C}^k (\widehat{\mathbf{A}} \odot \widehat{\mathbf{A}})^{\mathrm{T}} \right\|_F^2 , \tag{2.6}$$

$$\mathbf{C}^{k+1} = \underset{\widehat{\mathbf{C}} \in \mathbb{R}^{K \times R_{ps}}}{\operatorname{argmin}} \left\| \mathbf{X}_{(\mathbf{3})} - \widehat{\mathbf{C}} (\mathbf{A}^{k+1} \odot \mathbf{A}^{k+1})^{\mathrm{T}} \right\|_F^2 . \tag{2.7}$$

The problem (2.7) is the standard linear least-squares problem. So we need to focus on (2.6) and find a method to solve for the factor matrix $\mathbf{A}$. According to the

definition of Khatri-Rao product, we can have

$$\begin{aligned}
& \mathbf{X}_{(\mathbf{3})} = \mathbf{C}^k(\widehat{\mathbf{A}} \odot \widehat{\mathbf{A}})^{\mathrm{T}} \\
\Leftrightarrow \quad & \widehat{\mathbf{A}} \odot \widehat{\mathbf{A}} = ((\mathbf{C}^k)^{\dagger}\mathbf{X}_{(\mathbf{3})})^{\mathrm{T}} \\
\Leftrightarrow \quad & \widehat{\mathbf{a}}_r \otimes \widehat{\mathbf{a}}_r = ((\mathbf{C}^k)^{\dagger}\mathbf{X}_{(\mathbf{3})})^{\mathrm{T}}(:, r), \\
\Leftrightarrow \quad & \widehat{\mathbf{a}}_r \cdot \widehat{\mathbf{a}}_r^{\mathrm{T}} = unvec\left(((\mathbf{C}^k)^{\dagger}\mathbf{X}_{(\mathbf{3})})^{\mathrm{T}}(:, r)\right), \quad r = 1, 2, \ldots, R_{ps}.
\end{aligned} \tag{2.8}$$

where $(\cdot)^{\dagger}$ denotes the Moore-Penrose pseudoinverse; $\widehat{\mathbf{a}}_r$ is the rth column of matrix $\widehat{\mathbf{A}}$; $unvec\left(((\mathbf{C}^k)^{\dagger}\mathbf{X}_{(\mathbf{3})})^{\mathrm{T}}(:, r)\right)$ is a matrix which divides the vector to I parts and put each smaller vector into a matrix of size $I \times I$ (see Definition 1.10).

Therefore, instead of solving for the whole matrix $\widehat{\mathbf{A}}$ in (2.6), we can use (2.8) to solve for it column by column. Then the problem reduces to solving for the column $\widehat{\mathbf{a}}_r$ in (2.8), which can be solved alternatively through the same idea that fixing other variables but one in the objective vector.

For the sake of convenience, we let $\mathbf{x} \in \mathbb{R}^I = [x_1 \ x_2 \ \cdots \ x_I]^{\mathrm{T}}$ denote the unknown vector $\widehat{\mathbf{a}}_r$ and $\mathbf{Y} = unvec\left(((\mathbf{C}^k)^{\dagger}\mathbf{X}_{(\mathbf{3})})^{\mathrm{T}}(:, r)\right) \in \mathbb{R}^{I \times I}$. So we can solve for one component of the vector $\mathbf{x}$ each time by fixing other elements. For example, the first element x_1 can be solved for by rewriting the problem (2.8) to the following

$$\begin{bmatrix} x_1^2 & x_1x_2 & \cdots & x_1x_I \\ x_1x_2 & x_2^2 & & \\ \vdots & & \ddots & \\ x_1x_I & & & x_I^2 \end{bmatrix} = \mathbf{Y}.$$

The unknown x_1 is involved in the first column and first row, so we can take the first column and first row of $\mathbf{Y}$ to write out the problem, which is

$$\min_{x_1} \quad (y_{11} - x_1^2)^2 + \sum_{i=2}^{I} \left[(y_{i1} - x_i x_1)^2 + (y_{1i} - x_i x_1)^2\right]. \tag{2.9}$$

This cost function is a fourth-order polynomial with one variable x_1, so it is not hard to find its minima. Therefore, once we obtain the value of x_1, we can move on to the next element of $\mathbf{x}$ so that the column of $\widehat{\mathbf{a}}_r$ can be obtained enventially.

We call this method ***Partial Column-Wise ALS*** (PCW-ALS). Given three initial factor matrices $\mathbf{A}^0$ and $\mathbf{C}^0$, it solves the following subproblems

$$\begin{aligned} \mathbf{a}_r^{k+1} &= \underset{\widehat{a}_r \in \mathbb{R}^I}{\operatorname{argmin}} \left\| unvec\left(((\mathbf{C}^k)^\dagger \mathbf{X}_{(\mathbf{3})})^{\mathrm{T}}(:,r)\right) - \widehat{\mathbf{a}}_r \cdot \widehat{\mathbf{a}}_r^{\mathrm{T}} \right\|_F^2, \\ & \quad r = 1, \ldots, R_{ps}, \end{aligned} \tag{2.10}$$

$$\mathbf{C}^{k+1} = \underset{\widehat{\mathbf{C}} \in \mathbb{R}^{K \times R_{ps}}}{\operatorname{argmin}} \left\| \mathbf{X}_{(\mathbf{3})} - \widehat{\mathbf{C}}(\mathbf{A}^{k+1} \odot \mathbf{A}^{k+1})^{\mathrm{T}} \right\|_F^2 \tag{2.11}$$

to obtain the factor matrices $\mathbf{A}$ and $\mathbf{C}$. Starting from the initial guesses, the PCW-ALS approach fixes $\mathbf{C}$ to solve for each column $\mathbf{a}_r$ of $\mathbf{A}$, then fixes $\mathbf{A}$ to solve for $\mathbf{C}$. This process continues iteratively until some convergence criterion is satisfied.

The PCW-ALS algorithm essentially is also a Guass-Seidel method. Therefore, similar with the ALS method, the PCW-ALS algorithm can only guarantee that the cost function decreases as the number of iterations increases and can find a solution where the cost function ceases to decrease.

2.2 SOPD for fourth-order partially symmetric tensors

In this section, we consider the SOPD for two types of fourth-order partial symmetric tensors. The PCW-ALS method can also be applied to solve the fourth-order cases.

Let us consider the following partially symmetric fourth-order tensor $\mathcal{X} \in \mathbb{R}^{I \times I \times I \times L}$ with $x_{ijkl} = x_{\sigma(ijk)l}$, where σ is a permutation on (ijk), so it is symmetric on mode-1, mode-2 and mode-3. We call it Type I fourth-order partially symmetric tensor. So the problem we want to solve is as follows,

$$\underset{\widehat{\mathcal{X}}}{\text{minimize}} \quad \left\| \mathcal{X} - \widehat{\mathcal{X}} \right\|_F^2, \quad \text{where} \quad \widehat{\mathcal{X}} = \sum_{r=1}^{R_{ps}} \mathbf{a}_r \circ \mathbf{a}_r \circ \mathbf{a}_r \circ \mathbf{c}_r. \tag{2.12}$$

Notice that we also use R_{ps} since it is not the CP rank. This problem is equivalent to

$$\min_{\mathbf{A},\mathbf{C}} \left\| \mathcal{X} - \sum_{r=1}^{R_{ps}} \mathbf{a}_r \circ \mathbf{a}_r \circ \mathbf{a}_r \circ \mathbf{c}_r \right\|_F^2, \tag{2.13}$$

with respect to factor matrices $\mathbf{A}$ and $\mathbf{C}$.

Since $\widehat{\mathcal{X}}$ satisfies the SOPD equation (2.12), by using the tensor matricization, we have

$$\begin{aligned} \widehat{\mathcal{X}} &= \sum_{r=1}^{R_{ps}} \mathbf{a}_r \circ \mathbf{a}_r \circ \mathbf{a}_r \circ \mathbf{c}_r \\ &\Downarrow \\ \widehat{\mathbf{X}}_{(3)} &= \mathbf{C}(\mathbf{A} \odot \mathbf{A} \odot \mathbf{A})^{\mathrm{T}} \end{aligned}$$

where $\widehat{\mathbf{X}}_{(3)} \in \mathbb{R}^{L \times I^3}$ is the mode-4 matricization of tensor $\widehat{\mathcal{X}}$; $\mathbf{A} \in \mathbb{R}^{I \times R_{ps}}$ and $\mathbf{C} \in \mathbb{R}^{L \times R_{ps}}$.

Then the original SOPD problem (2.13) has the following expression,

$$\min_{\mathbf{A},\mathbf{C}} \left\| \mathbf{X}_{(4)} - \mathbf{C}(\mathbf{A} \odot \mathbf{A} \odot \mathbf{A})^{\mathrm{T}} \right\|_F^2. \tag{2.14}$$

The idea that solving the two factor matrices alternatively can also be used on the problem (2.14). So assume that $\mathbf{C}$ is fixed in (2.14), then we can have

$$\begin{aligned} & \mathbf{X}_{(4)} = \mathbf{C}(\mathbf{A} \odot \mathbf{A} \odot \mathbf{A})^{\mathrm{T}} \\ \Leftrightarrow \quad & \mathbf{a}_r \otimes \mathbf{a}_r \otimes \mathbf{a}_r = ((\mathbf{C})^{\dagger}\mathbf{X}_{(4)})^{\mathrm{T}}(:, r) \\ \Leftrightarrow \quad & \mathbf{a}_r \circ \mathbf{a}_r \circ \mathbf{a}_r = ten(((\mathbf{C})^{\dagger}\mathbf{X}_{(4)})^{\mathrm{T}}(:, r)),\ r = 1, \ldots, R_{ps}, \end{aligned} \tag{2.15}$$

where $\mathbf{a}_r$ is the rth column of matrix $\mathbf{A}$, $ten(((\mathbf{C})^{\dagger}\mathbf{X}_{(4)})^{\mathrm{T}}(:, r))$ is a tensor of size $I \times I \times I$ which divides the vector to I^2 parts and put each piece into a tensor (see the Definition 1.11 in Chap. 2).

So, the PCW-ALS method can be used to solve the problem (2.14). Given the initial guesses $\mathbf{A}^0$ and $\mathbf{C}^0$, it solves the following subproblems

$$\begin{aligned}\mathbf{a}_r^{k+1} &= \underset{\widehat{a}_r \in \mathbb{R}^I}{\operatorname{argmin}} \left\| ten(((\mathbf{C})^{\dagger}\mathbf{X}_{(4)})^{\mathrm{T}}(:,r)) - \widehat{\mathbf{a}}_r \circ \widehat{\mathbf{a}}_r \circ \widehat{\mathbf{a}}_r \right\|_F^2, \\ & \quad r = 1, \ldots, R_{ps}, \qquad (2.16)\\ \mathbf{C}^{k+1} &= \underset{\widehat{\mathbf{C}} \in \mathbb{R}^{K \times R_{ps}}}{\operatorname{argmin}} \left\| \mathbf{X}_{(4)} - \widehat{\mathbf{C}}(\mathbf{A}^{k+1} \odot \mathbf{A}^{k+1} \odot \mathbf{A}^{k+1})^{\mathrm{T}} \right\|_F^2 \qquad (2.17)\end{aligned}$$

to obtain the factor matrices $\mathbf{A}$ and $\mathbf{C}$. The updating process continues iteratively until some convergence criterion is satisfied. And the problem (2.16) can be solved the same way as the third-order version. We solve one element in $\widehat{\mathbf{a}}_r$ each time by fixing other elements, so the problem becomes to find the minima of a six-order polynomial in terms of the unknown element. We repeat it for each element so that the vector $\widehat{\mathbf{a}}_r$ can be computed in the least-squares sense.

Now, let us consider a different type of fourth-order partially symmetric tensor, $\mathcal{X} \in \mathbb{R}^{I \times J \times I \times J}$ with $x_{ijkl} = x_{kjil}$ and $x_{ijkl} = x_{ilkj}$. This means the tensor is symmetric in mode-1 and mode-3 and also symmetric in mode-2 and mode-4, we call it Type II partially symmetric tensor. So the CP problem is as follows,

$$\underset{\widehat{\mathcal{X}}}{\text{minimize}} \quad \left\| \mathcal{X} - \widehat{\mathcal{X}} \right\|_F^2, \quad \text{where} \quad \widehat{\mathcal{X}} = \sum_{r=1}^{R_{ps}} \mathbf{a}_r \circ \mathbf{b}_r \circ \mathbf{a}_r \circ \mathbf{b}_r. \qquad (2.18)$$

Again here the R_{ps} is different from the CP rank for the tensor $\mathcal{T}$. So this problem is equivalent to

$$\min_{\mathbf{A},\mathbf{B}} \left\| \mathcal{X} - \sum_{r=1}^{R_{ps}} \mathbf{a}_r \circ \mathbf{b}_r \circ \mathbf{a}_r \circ \mathbf{b}_r \right\|_F^2, \qquad (2.19)$$

with respect to factor matrices $\mathbf{A} = [\mathbf{a}_1\ \mathbf{a}_2\ \cdots\ \mathbf{a}_{R_{ps}}] \in \mathbb{R}^{I \times R_{ps}}$ and $\mathbf{B} = [\mathbf{b}_1\ \mathbf{b}_2\ \cdots\ \mathbf{b}_{R_{ps}}] \in \mathbb{R}^{J \times R_{ps}}$.

In order to solve this problem by PCW-ALS, we need to introduce a different

type of matricization for forth-order tensor, which is inspired from the matricization definition in [31].

Definition 2.1 (square matricization) *For a fourth-order tensor $\mathcal{X} \in \mathbb{R}^{I\times J\times K\times L}$, the square matricization is denoted by $mat(\mathcal{X}) \in \mathbb{R}^{IK\times JL}$ and is defined as*

$$\mathbf{X} = mat(\mathcal{X}) \Leftrightarrow (\mathbf{X})_{(i-1)K+k,(j-1)L+l} = \mathcal{X}_{ijkl}. \tag{2.20}$$

So for the partial symmetric tensor $\mathcal{X} \in \mathbb{R}^{I\times J\times I\times J}$ we study here, we use the square matricization on $\widehat{\mathcal{X}}$, then we have

$$\begin{aligned} \widehat{\mathcal{X}} &= \sum_{r=1}^{R_{ps}} \mathbf{a}_r \circ \mathbf{b}_r \circ \mathbf{a}_r \circ \mathbf{b}_r \\ &\Downarrow \\ mat(\widehat{\mathcal{X}}) &= (\mathbf{A}\odot\mathbf{A})(\mathbf{B}\odot\mathbf{B})^{\mathrm{T}}, \end{aligned} \tag{2.21}$$

where $mat(\widehat{\mathcal{X}}) \in \mathbb{R}^{I^2\times J^2}$ is the square matricization. So our original SOPD problem (2.19) can be written as the following expression,

$$\min_{\mathbf{A},\mathbf{B}} \left\| mat(\mathcal{X}) - (\mathbf{A}\odot\mathbf{A})(\mathbf{B}\odot\mathbf{B})^{\mathrm{T}} \right\|_F^2. \tag{2.22}$$

Therefore instead of solving two matrices in (2.22), the PCW-ALS approach solves the following subproblems by given initial guesses $\mathbf{A}^0$ and $\mathbf{B}^0$,

$$\begin{aligned} \mathbf{A}^{k+1} &= \operatorname*{argmin}_{\widehat{\mathbf{A}}\in\mathbb{R}^{I\times R_{ps}}} \left\| mat(\mathcal{X}) - (\widehat{\mathbf{A}}\odot\widehat{\mathbf{A}})(\mathbf{B}^{k+1}\odot\mathbf{B}^{k+1})^{\mathrm{T}} \right\|_F^2, \\ \mathbf{B}^{k+1} &= \operatorname*{argmin}_{\widehat{\mathbf{B}}\in\mathbb{R}^{J\times R_{ps}}} \left\| mat(\mathcal{X}) - (\mathbf{A}^{k+1}\odot\mathbf{A}^{k+1})(\widehat{\mathbf{B}}\odot\widehat{\mathbf{B}})^{\mathrm{T}} \right\|_F^2 \end{aligned} \tag{2.23}$$

to update the two factor matrices until some convergence criterion is satisfied. And both the problems in (2.23) can be solved by using the exact technique as solving

problem (2.6). Therefore, the updating rule is as follows,

$$\begin{aligned}
\mathbf{a}_r^{k+1} &= \underset{\widehat{\mathbf{a}}_r \in \mathbb{R}^I}{\operatorname{argmin}} \left\| unvec((mat(\mathcal{X})((\widehat{\mathbf{B}} \odot \widehat{\mathbf{B}})^{\mathrm{T}})^{\dagger})(:,r)) - \widehat{\mathbf{a}}_r \cdot \widehat{\mathbf{a}}_r^T \right\|_F^2, \\
&\quad r = 1, 2, \ldots, R, \\
\mathbf{b}_r^{k+1} &= \underset{\widehat{\mathbf{b}}_r \in \mathbb{R}^I}{\operatorname{argmin}} \left\| unvec(((\mathbf{A}^{k+1} \odot \mathbf{A}^{k+1})^{\dagger} mat(\mathcal{X}))^{\mathrm{T}}(:,r)) - \widehat{\mathbf{b}}_r \cdot \widehat{\mathbf{b}}_r^T \right\|_F^2. \\
&\quad r = 1, 2, \ldots, R.
\end{aligned}$$

3 Numerical examples

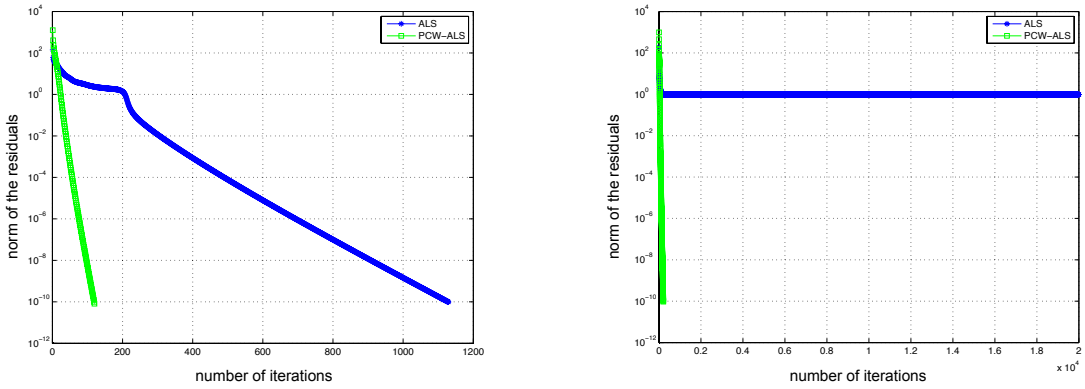
In this section, we compare the performance of ALS against PCW-ALS for the third-order partial symmetric tensors and two types of fourth-order partial symmetric tensors. The numerical examples show that the PCW-ALS is better than ALS method in terms of the number of iterations and the CPU time. Additionally, it can also reduce some swamps in the implementation of the ALS.

Example 3.1 (**third-order partially symmetric tensor**) We generate a partial symmetric tensor $\mathcal{X} \in \mathbb{R}^{17\times17\times18}$ by random data, in which $x_{ijk} = x_{jik}$. Consider a CP decomposition of $\mathcal{X}$ with $R_{ps} = 17$. So it has two different factor matrices $\mathbf{A} \in \mathbb{R}^{17\times17}$ and $\mathbf{C} \in \mathbb{R}^{18\times17}$, and the decomposition is

$$\mathcal{X} = \sum_{r=1}^{R_{ps}} \mathbf{a}_r \circ \mathbf{a}_r \circ \mathbf{c}_r.$$

In the two figures, the plots show the error $\|\mathcal{X} - \mathcal{X}_{est}\|_F^2$ versus the number of iterations it takes to obtain an error of 10^{-10}, where $\mathcal{X}_{est}$ denotes the obtained tensor after every iteration. Since the ALS method needs three initial guesses, here we let $\mathbf{B}^0 = \mathbf{A}^0$ for it.

In Figure 5.1a, the initial guesses are good. Both algorithms work well, but the PCW-ALS method is better than the ALS algorithm. The PCW-ALS only takes 120



(a) good initial guess (b) random initial guess

Figure 5.1: **Plots for the Example 3.1**

iterations in comparison to that of 1129 ALS iterations. Moreover, the PCW-ALS is faster than ALS since the CPU time of PCW-ALS is 3.9919s while the ALS is 6.4126s. Figure 5.1b shows that PCW-ALS can reduce the swamp by only taking 205 iterations to reach an error within 10^{-10}. While the ALS has a swamp and the error stays in 10^0 after 20000 iterations.

Example 3.2 (**Simulation**) For the tensor $\mathcal{X}$ given in the Example 3.1, the ALS and PCW-ALS algorithms are used to decompose it with rank $R_{ps} = 17$. Both of ALS and PCW-ALS are used on tensor $\mathcal{X}$ with 50 different random initial starters and and the average results in terms of number of iterations and CPU time are shown in the Table 5.1. Furthermore, we calculate the standard deviation for those data series and the results are shown in the Table 5.2.

	ALS	PCW-ALS
average CPU time	17.1546s	6.1413s
average number of iterations	3445.0	258.7

Table 5.1: **The comparison of ALS and PCW-ALS (Mean).**

Example 3.3 (**CPU time comparison in terms of tensor size**) We apply the ALS method and PCW-ALS method on the third-order partially symmetric tensors

	ALS	PCW-ALS
standard deviation of CPU time	25.3297s	8.0604s
standard deviation of number of iterations	3166.4	209.0581

Table 5.2: **The comparison of ALS and PCW-ALS (Standard Deviation).**

$\mathcal{X}_1 \in \mathbb{R}^{10\times10\times10}$ with $R_{ps} = 10$, $\mathcal{X}_2 \in \mathbb{R}^{20\times20\times20}$ with $R_{ps} = 20$, ..., $\mathcal{X}_9 \in \mathbb{R}^{90\times90\times90}$ with $R_{ps} = 90$ and compare the CPU times of both methods for the same tensor size. In order to have a fair comparison, for each tensor $\mathcal{X}_i$, we use the technique in Example 3.2 to get the average CPU times of both methods. The following Figure 5.2 shows that as the tensor size increases, the CPU time of ALS increases much faster than the PCW-ALS time.

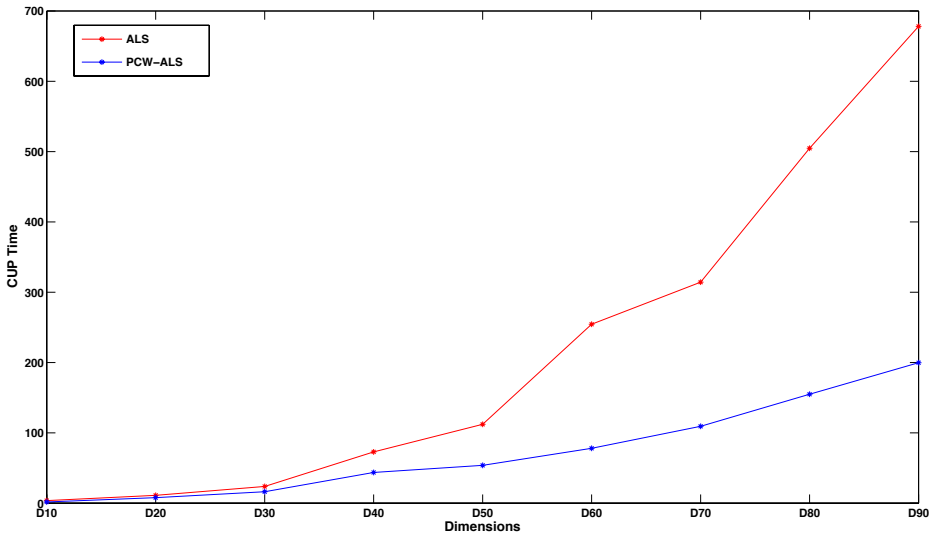


Figure 5.2: **Plots for the Example 3.3**

Example 3.4 (**Type I fourth-order partial symmetric tensor**) The objective in this example is a fourth-order partial symmetric tensor $\mathcal{X} \in \mathbb{R}^{8\times8\times8\times5}$ with $R_{ps} = 5$, in which $x_{ijkl} = x_{\sigma(ijk)l}$, where σ is an any permutation on (ijk). So the ALS and PCW-ALS algorithms are used to solve for the factor matrices $\mathbf{A}$ and $\mathbf{C}$ so that it

has the following decomposition,

$$\mathcal{X} = \sum_{r=1}^{R_{ps}} \mathbf{a}_r \circ \mathbf{a}_r \circ \mathbf{a}_r \circ \mathbf{c}_r.$$

Figure 5.3 shows that the PCW-ALS algorithm just takes less than 100 iterations to reach an error within 10^{-10}, however, the ALS algorithm takes 647 iterations. Additionally, the CPU time of PCW-ALS is 1.5131s while the ALS method is 2.9323s.

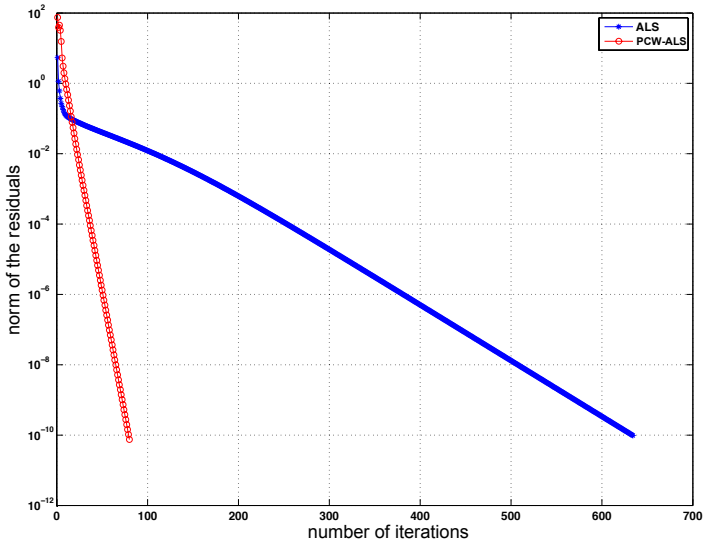


Figure 5.3: **Plots for the Example 3.4**

Example 3.5 (**Type II fourth-order partial symmetric tensor**)

Another type of fourth-order partial symmetric tensor $\mathcal{X} \in \mathbb{R}^{12\times13\times12\times13}$ with $R_{ps} = 12$, in which $x_{ijkl} = x_{kjil}$ and $x_{ijkl} = x_{ilkj}$ is factored in this example by using the PCW-ALS and ALS algorithms. So the decomposition model is

$$\mathcal{X} = \sum_{r=1}^{R_{ps}} \mathbf{a}_r \circ \mathbf{b}_r \circ \mathbf{a}_r \circ \mathbf{b}_r.$$

where $\mathbf{a}_r$ and $\mathbf{b}_r$ are the columns of factor matrices $\mathbf{A}$ and $\mathbf{B}$.

Figure 5.4 shows that the PCW-ALS algorithm takes 3.5113s and 70 iterations to reach an error within 10^{-10} while the ALS algorithm takes 9.4803s and 378 iterations.

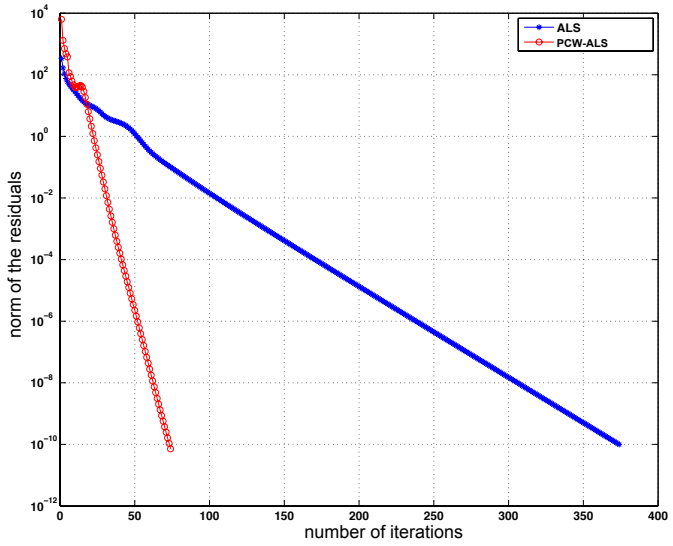


Figure 5.4: **Plots for the Example 3.5**

Chapter 6

Block term decompositions

1 Introduction

We introduced the Higher-Order Singular Value Decomposition (HOSVD) in Chap. 2. In this chapter, we study the numerical methods for solving a HOSVD for a given tensor and introduce a new type of tensor decomposition which leads to a framework that unifies the HOSVD and the CANDECOMP/PARAFAC (CP) decomposition. Such decomposition is called *Block Term Decomposition* (BTD)[28, 29, 36]. The BTD has been used in the area of signal processing [36]. Separately, Hopke et al [102] propose a receptor model for third-order tensor to study the source of air pollutants. Essentially, the receptor model is one type of BTD. In addition, the block term decomposition in rank-$(L, L, 1)$ can be considered as a special case of the receptor model. In [102], the three-way receptor model with non-negative constraints is solved by Multilinear Engine (ME) [94].

De Lathauwer [29] introduces an alternating least-squares method to solve for the BTD of a given tensor which we call BTD-ALS. The algorithm basically breaks the original problem into several least-squares subproblems. Based on this method, we introduce a regularized method which adds a regularized item to each subproblem since the least-squares method has some issues like non-uniqueness. This regularized

method was introduced by Navasca, Kindermann and De Lathawer [91] for an ALS algorithm of CP decomposition. Here we call it BTD-RALS. In terms of improving the efficiency of BTD-ALS, line search schemes [106] [92] can also been used. However, the line search method is much more complicated than the regularized method.

Additionally, the BTD-ALS method may require a high number of iterations to converge which may be due to the poor initialization of the factor matrices or the collinearity in the factor matrices. This slowed convergence characterized by a flat curve in a log error plot is referred to as the *swamp* (see Chap. 3). Here, several numerical examples are provided to show that the regularized method converges faster than the BTD-ALS which, to some degree, can remove swamps. In addition, we study the BTD problem and the BTD-ALS method from the view of optimization to show that the method is in the same framework as the ALS method. Therefore, we can conclude the same limit point statement (see Theorem 5.1 and Theorem 5.2) for the BTD-RALS method with the one in Chap. 4.

Suppose that the sequence $\{\mathcal{T}^k\}$ *obtained from BTD-RALS has limit points, then every limit point* $\bar{\mathcal{T}}$ *is a critical point of the following problem:*

$$\underset{\widehat{\mathcal{T}}}{\text{minimize}} \quad \|\mathcal{T} - \widehat{\mathcal{T}}\|_F^2, \quad \textit{where} \quad \widehat{\mathcal{T}} \ \textit{is a block term decomposition.}$$

In terms of the uniqueness of Block Term Decomposition for the third-order tensor, De Lathauwer [29] provided several sufficient conditions so that the BTD can be essentially unique. We study the relationship of BTD-$(L, L, 1)$ and three-way receptor model and mathematically prove that the three-way receptor model does not have unique solution.

The text is organized as follows. In Section 2 we introduce the algorithm for solving the HOSVD/Tucker decomposition. Different types of BTD are presented in Section 3. Section 4 provides the corresponding algorithms (BTD-ALS) [29] for solving the

block term decompositions. In Section 5, we connect the BTD-ALS method with the block nonlinear Gauss-Seidel (GS) method, introduce the regularized alternating least-squares (BTD-RALS) to solve the BTD problem, and study the properties of the regularization parameter. The convergence conclusion that given the existence of some critical points of the BTD-RALS method, the limit points of the converging subsequences of the BTD-RALS are the critical points of the original problem, is proven in this section. Several numerical examples are also provided in Section 6 to show that the BTD-RALS converges faster than BTD-ALS and can remove some swamps. In Section 7, we focus on the receptor model and prove that it cannot have unique solution mathematically.

2 HOSVD algorithm

2.1 Algorithm to compute HOSVD

Recall that the HOSVD factors an Nth-order tensor $\mathcal{T} \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_N}$ into N orthogonal factor matrices $\mathbf{A}^{(n)}$, $n = 1, 2, \ldots, N$ and a core tensor $\mathcal{G}$ in the form

$$\mathcal{T} = \mathcal{G} \times_1 \mathbf{A}^{(1)} \times_2 \mathbf{A}^{(2)} \times_3 \cdots \times_N \mathbf{A}^{(N)}. \tag{2.1}$$

So, given a tensor $\mathcal{T}$, we want to find the orthogonal factor matrices and core tensor to satisfy the equation (2.1). In [33], Lieven provided a very convenient and efficient method to compute the HOSVD. In this section, we use a real-valued third-order tensor to explain their method, which can be generalized to any complex-valued Nth-order tensor.

By using the matricization on tensors in the equation (3.1), then we can rewrite

it as the following three equations,

$$\begin{aligned}
\mathbf{T}_{(\mathbf{1})} &= \mathbf{A}\mathbf{G}_{(\mathbf{1})}(\mathbf{C} \otimes \mathbf{B})^{\mathrm{T}}, \\
\mathbf{T}_{(\mathbf{2})} &= \mathbf{B}\mathbf{G}_{(\mathbf{2})}(\mathbf{C} \otimes \mathbf{A})^{\mathrm{T}}, \\
\mathbf{T}_{(\mathbf{3})} &= \mathbf{C}\mathbf{G}_{(\mathbf{3})}(\mathbf{B} \otimes \mathbf{A})^{\mathrm{T}}.
\end{aligned}$$

Then, the method in [33] for finding the factor matrices of HOSVD is to take the left singular vectors of $\mathbf{T}_{(1)}$ to be the factor matrix $\mathbf{A}$, the left singular vectors of $\mathbf{T}_{(2)}$ to be the factor matrix $\mathbf{B}$ and the left singular vectors of $\mathbf{T}_{(3)}$ to be the factor matrix $\mathbf{C}$. Since all the factor matrices are orthogonal, we can use the equations above to compute the core tensor $\mathcal{G}$, i.e.,

$$\begin{aligned}
\mathbf{G}_{(\mathbf{1})} &= \mathbf{A}^{\mathrm{T}}\mathbf{T}_{(\mathbf{1})}(\mathbf{C} \otimes \mathbf{B}), \\
\mathbf{G}_{(\mathbf{2})} &= \mathbf{B}^{\mathrm{T}}\mathbf{T}_{(\mathbf{2})}(\mathbf{C} \otimes \mathbf{A}), \\
\mathbf{G}_{(\mathbf{3})} &= \mathbf{C}^{\mathrm{T}}\mathbf{T}_{(\mathbf{3})}(\mathbf{B} \otimes \mathbf{A}).
\end{aligned}$$

So from any equation above we have the following formula to calculate the core tensor,

$$\mathcal{G} = \mathcal{T} \times_1 \mathbf{A}^{\mathrm{T}} \times_2 \mathbf{B}^{\mathrm{T}} \times_3 \mathbf{C}^{\mathrm{T}}. \tag{2.2}$$

The following table shows the outline of the HOSVD algorithm for a third-order tensor.

For an Nth-order tensor, the mode-n matricization on the tensors in the equation (2.1) gives us

$$\mathbf{T}_{(\mathbf{n})} = \mathbf{U}^{(n)}\mathbf{G}_{(\mathbf{n})}(\mathbf{U}^{(N)} \otimes \cdots \otimes \mathbf{U}^{(n+1)} \otimes \mathbf{U}^{(n-1)} \otimes \cdots \otimes \mathbf{U}^{(1)})^{\mathrm{T}}. \tag{2.3}$$

Therefore, we can follow the same steps in the above algorithm by using the equation

HOSVD-Algorithm

procedure HOSVD($\mathcal{T}$)

$\mathbf{A} \leftarrow$ left singular vectors of $\mathbf{T}_{(\mathbf{1})}$—— % computing the factor matrix $\mathbf{A}$

$\mathbf{B} \leftarrow$ left singular vectors of $\mathbf{T}_{(\mathbf{2})}$—— % computing the factor matrix $\mathbf{B}$

$\mathbf{C} \leftarrow$ left singular vectors of $\mathbf{T}_{(\mathbf{3})}$—— % computing the factor matrix $\mathbf{C}$

$\mathcal{G} = \mathcal{T} \times_1 \mathbf{A}^{\mathrm{T}} \times_2 \mathbf{B}^{\mathrm{T}} \times_3 \mathbf{C}^{\mathrm{T}}$—— % computing the core tensor $\mathcal{G}$

return $\mathbf{A}$, $\mathbf{B}$, $\mathbf{C}$, $\mathcal{G}$

end procedure

Table 6.1: **HOSVD algorithm for a third-order tensor $\mathcal{T} \in \mathbb{R}^{I \times J \times K}$**

(2.3).

2.2 Higher-order orthogonal iteration (HOOI)

Recall that the n-rank of a Nth-order tensor $\mathcal{T} \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_N}$ is the rank of its mode-n matricization. It is clear that $R_n = \mathrm{rank}_n(\mathcal{T}) \leq I_n$. And if $R_n = \mathrm{rank}_n(\mathcal{T})$ for $n = 1, 2, \ldots, N$, we say that $\mathcal{T}$ is a full rank-$(I_1, I_2, \ldots, I_N)$ tensor.

By the HOSVD Theorem 3.1 in Chap. 3, every Nth-order tensor $\mathcal{T}$ can be written as a product

$$\mathcal{T} = \mathcal{G} \times_1 \mathbf{U}^{(1)} \times_2 \mathbf{U}^{(2)} \times_3 \cdots \times_N \mathbf{U}^{(N)}.$$

So for a given tensor $\mathcal{T} \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_N}$, we can find an exact Tucker decomposition with the core tensor $\mathcal{G}$ of size $I_1 \times I_2 \times \cdots \times I_N$. When $\mathrm{rank}_n(\mathcal{T}) = R_n < I_n$ for one or more n, then according the work of De Lathauwer, De Moor and Vandewalle [33], we take the first R_n leading left singular vectors of $\mathbf{T}_{(\mathbf{n})}$ as the factor matrix $\mathbf{U}^{(n)} \in \mathbb{C}^{I_n \times R_n}$ ($\mathbf{U}^{(n)^{\mathrm{T}}}\mathbf{U}^{(n)} = \mathbf{I} \in \mathbb{C}^{R_n \times R_n}$, where $\mathbf{I}$ is the identity matrix). This decomposition is called the ***truncated HOSVD***. Actually, the core tensor of the truncated HOSVD is also all-orthogonal [33].

Therefore, for a given tensor $\mathcal{T} \in \mathbb{R}^{I_1 \times I_2 \times I_3}$ and given numbers R_1, R_2, R_3 with

$R_n \leq \text{rank}_n(\mathcal{T})$, the algorithm of the truncated HOSVD is shown in the Table 6.2. This also can be generalized to any complex-valued Nth-order tensor by using the equation (2.3).

Truncated HOSVD-Algorithm

procedure Truncated HOSVD($\mathcal{T}$, R_1, R_2, R_3)

$\mathbf{A} \leftarrow R_1$ left singular vectors of $\mathbf{T}_{(\mathbf{1})}$—— % computing the factor matrix $\mathbf{A}$

$\mathbf{B} \leftarrow R_2$ left singular vectors of $\mathbf{T}_{(\mathbf{2})}$—— % computing the factor matrix $\mathbf{B}$

$\mathbf{C} \leftarrow R_3$ left singular vectors of $\mathbf{T}_{(\mathbf{3})}$—— % computing the factor matrix $\mathbf{C}$

$\mathcal{G} = \mathcal{T} \times_1 \mathbf{A}^{\mathrm{T}} \times_2 \mathbf{B}^{\mathrm{T}} \times_3 \mathbf{C}^{\mathrm{T}}$—— % computing the core tensor $\mathcal{G}$

return $\mathbf{A}$, $\mathbf{B}$, $\mathbf{C}$, $\mathcal{G}$

end procedure

Table 6.2: **Truncated HOSVD algorithm for a third-order tensor $\mathcal{T} \in \mathbb{R}^{I \times J \times K}$**

So, we can get a smaller core tensor of size $R_1 \times R_2 \times \cdots \times R_N$. However, this is not like the decomposition in HOSVD theorem 3.1. The truncated HOSVD is not optimal in terms of giving the best fit as measured by the norm of the difference [70]. So, for the tensor that $\text{rank}_n(\mathcal{T}) = R_n < I_n$, there are several methods to compute this decomposition to find a best fit, [64, 74, 34]. In these methods, [34] provides a more efficient iterative technique for calculating the factor matrices with the corresponding core tensor and calls it the ***Higher-Order Orthogonal Iteration*** (HOOI). This algorithm is an iterative method and the result of truncated HOSVD can be a good initial starting set for HOOI.

We present some details of the HOOI method [70] here, for more information, please refer to [70, 34]. Given a Nth-order tensor $\mathcal{T} \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_N}$, the Tucker

decomposition with rank constraints problem is

$$\begin{aligned} \min_{\mathcal{G},\mathbf{A}^{(1)},\ldots,\mathbf{A}^{(N)}} \quad & \left\| \mathcal{T} - \mathcal{G} \times_1 \mathbf{A}^{(1)} \times_2 \mathbf{A}^{(2)} \times_3 \cdots \times_N \mathbf{A}^{(N)} \right\|_F^2 \\ \text{subject to} \quad & \mathcal{G} \in \mathbb{R}^{R_1 \times R_2 \times \cdots \times R_N}, \\ & \mathbf{A}^{(n)} \in \mathbb{R}^{I_n \times R_n} \end{aligned} \tag{2.4}$$

By rewriting the above objective function in vectorized form as

$$\left\| vec(\mathcal{T}) - (\mathbf{A}^{(N)} \otimes \mathbf{A}^{(N-1)} \otimes \cdots \otimes \mathbf{A}^{(1)}) vec(\mathcal{G}) \right\|_F^2,$$

it is straightforward to show that the core tensor $\mathcal{G}$ must satisfy

$$\mathcal{G} = \mathcal{T} \times_1 (\mathbf{A}^{(1)})^{\mathrm{T}} \times_2 (\mathbf{A}^{(2)})^{\mathrm{T}} \times_3 \cdots \times_N (\mathbf{A}^{(N)})^{\mathrm{T}}.$$

We can then rewrite the (square of the) objective function as

$$\begin{aligned} & \left\| \mathcal{T} - \mathcal{G} \times_1 \mathbf{A}^{(1)} \times_2 \mathbf{A}^{(2)} \times_3 \cdots \times_N \mathbf{A}^{(N)} \right\|_F^2 \\ = \; & \|\mathcal{T}\|_F^2 - 2\langle \mathcal{T}, \mathcal{G} \times_1 \mathbf{A}^{(1)} \times_2 \mathbf{A}^{(2)} \times_3 \cdots \times_N \mathbf{A}^{(N)} \rangle + \|\mathcal{G} \times_1 \mathbf{A}^{(1)} \times_2 \mathbf{A}^{(2)} \times_3 \cdots \times_N \mathbf{A}^{(N)}\|_F^2 \\ = \; & \|\mathcal{T}\|_F^2 - 2\langle \mathcal{T} \times_1 (\mathbf{A}^{(1)})^{\mathrm{T}} \times_2 (\mathbf{A}^{(2)})^{\mathrm{T}} \times_3 \cdots \times_N (\mathbf{A}^{(N)})^{\mathrm{T}}, \mathcal{G} \rangle + \|\mathcal{G}\|_F^2 \\ = \; & \|\mathcal{T}\|_F^2 - 2\langle \mathcal{G}, \mathcal{G} \rangle + \|\mathcal{G}\|_F^2 \\ = \; & \|\mathcal{T}\|_F^2 - \|\mathcal{G}\|_F^2 \\ = \; & \|\mathcal{T}\|_F^2 - \left\| \mathcal{T} \times_1 (\mathbf{A}^{(1)})^{\mathrm{T}} \times_2 (\mathbf{A}^{(2)})^{\mathrm{T}} \times_3 \cdots \times_N (\mathbf{A}^{(N)})^{\mathrm{T}} \right\|_F^2. \end{aligned}$$

Now, we can use an ALS approach to solve the problem (2.4). Because $\|\mathcal{T}\|_F^2$ is constant, (2.4) can be recast as a series of subproblems involving the following

maximization problem, which solves for the nth component matrix:

$$\begin{aligned} \min_{\mathbf{A}^{(n)}} \quad & \left\| \mathcal{T} \times_1 (\mathbf{A}^{(1)})^{\mathrm{T}} \times_2 (\mathbf{A}^{(2)})^{\mathrm{T}} \times_3 \cdots \times_N (\mathbf{A}^{(N)})^{\mathrm{T}} \right\|_F^2, \\ \text{subject to} \quad & \mathbf{A}^{(n)} \in \mathbb{R}^{I_n \times R_n}. \end{aligned} \tag{2.5}$$

The objective function in (2.5) can be written in matrix form as

$$\|(\mathbf{A}^{(n)})^{\mathrm{T}}\mathbf{W}\|_F^2 \text{ with } \mathbf{W} = \mathbf{T}_{(\mathbf{n})}(\mathbf{A}^{(N)} \otimes \cdots \otimes \mathbf{A}^{(n+1)} \otimes \mathbf{A}^{(n-1)} \otimes \cdots \otimes \mathbf{A}^{(1)}).$$

The solution can be determined using the SVD by simply setting $\mathbf{A}^{(n)}$ to be the R_n leading left singular vectors of $\mathbf{W}$.

Therefore, the HOOI algorithm for third-order tensor can be shown here. For a given tensor $\mathcal{X} \in \mathbb{R}^{I \times J \times K}$, our goal is to decompose it to be a Tucker model with the core tensor is $\mathcal{G} \in \mathbb{R}^{P \times Q \times R}$. We first use truncated HOSVD to get the initial factors $\mathbf{A}^0 \in \mathbb{R}^{I \times P}$, $\mathbf{B}^0 \in \mathbb{R}^{J \times Q}$ and $\mathbf{C}^0 \in \mathbb{R}^{K \times R}$. Then the update $\mathbf{A}^{k+1}$ is the left P singular vectors of $\mathbf{X}_{(\mathbf{1})}(\mathbf{C}^k \otimes \mathbf{B}^k)$, $\mathbf{B}^{k+1}$ is the left Q singular vectors of $\mathbf{X}_{(\mathbf{2})}(\mathbf{C}^k \otimes \mathbf{A}^{k+1})$ and $\mathbf{C}^{k+1}$ is the left R singular vectors of $\mathbf{X}_{(\mathbf{3})}(\mathbf{B}^{k+1} \otimes \mathbf{A}^{k+1})$. Same with the HOSVD, once we get the three factors, the core tensor can be calculated by (2.2). The following Table 6.3 shows the algorithm for a third-order tensor, where N is the number of iterations; a stopping criterion can also be used. We can measure the value of $e_k = \|\mathbf{A}^k - \mathbf{A}^{k-1}\|_2$ for each iteration. So, one possible criterion is when e_k is less some very small number ϵ.

3 Block term decompositions

In this section, we will introduce a more general tensor decomposition which is called ***Block Term Decomposition(BTD)*** [28, 29, 36] and we can see that both the CP decomposition and Tucker decomposition are the special cases of BTD. Here we also

HOOI-Algorithm

procedure HOOI($\mathcal{X}$, P, Q, R)

give initial guess $\mathbf{A}^0 \in \mathbb{R}^{I\times P}$, $\mathbf{B}^0 \in \mathbb{R}^{J\times Q}$, $\mathbf{C}^0 \in \mathbb{R}^{K\times R}$

for $n = 1, \ldots, N$ do

$\mathbf{W} \leftarrow \mathbf{X}_{(\mathbf{1})}(\mathbf{C}^n \otimes \mathbf{B}^n)$

$\mathbf{A}^{n+1} \leftarrow P$ left singular vectors of $\mathbf{W}$—— % computing the factor matrix $\mathbf{A}$

$\mathbf{W} \leftarrow \mathbf{X}_{(\mathbf{2})}(\mathbf{C}^n \otimes \mathbf{A}^{n+1})$

$\mathbf{B}^{n+1} \leftarrow Q$ left singular vectors of $\mathbf{W}$—— % computing the factor matrix $\mathbf{B}$

$\mathbf{W} \leftarrow \mathbf{X}_{(\mathbf{3})}(\mathbf{B}^{n+1} \otimes \mathbf{A}^{n+1})$

$\mathbf{C}^{n+1} \leftarrow R$ left singular vectors of $\mathbf{W}$—— % computing the factor matrix $\mathbf{C}$

end for

$\mathcal{G} \leftarrow \mathcal{X} \times_1 \mathbf{A}^{\mathrm{T}} \times_2 \mathbf{B}^{\mathrm{T}} \times_3 \mathbf{C}^{\mathrm{T}}$

return $\mathbf{A}$, $\mathbf{B}$, $\mathbf{C}$, $\mathcal{G}$

end procedure

Table 6.3: **HOOI algorithm for a third-order tensor $\mathcal{T} \in \mathbb{R}^{I\times J\times K}$**

use real-valued third-order tensors to explain the block term decomposition. Notice that all the analysis holds for any complex-valued Nth-order tensor.

3.1 Decomposition in rank-$(L, L, 1)$ terms

Definition 3.1 *[29] Let* $\mathbf{A} = [\mathbf{A}_1 \ \cdots \ \mathbf{A}_R]$ *and* $\mathbf{B} = [\mathbf{B}_1 \ \cdots \ \mathbf{B}_R]$ *be two partitioned matrices. Then we define a product of* $\mathbf{A}$ *and* $\mathbf{B}$*, denoted* $\odot_p$*, which is*

$$\mathbf{A} \odot_p \mathbf{B} = [\mathbf{A}_1 \otimes \mathbf{B}_1 \ \cdots \ \mathbf{A}_R \otimes \mathbf{B}_R]. \tag{3.1}$$

Notice that this definition can be considered as a generalization of matrix Khatri-

Rao product, which is defined on two matrices $\mathbf{A} \in \mathbb{R}^{I \times K}$ and $\mathbf{B} \in \mathbb{R}^{J \times K}$ as

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

Definition 3.2 (BTD-$(L, L, 1)$) *[29] A decomposition of a tensor $\mathcal{X} \in \mathbb{R}^{I \times J \times K}$ in a sum of rank-$(L, L, 1)$ terms is a decomposition of $\mathcal{X}$ of the form*

$$\mathcal{X} = \sum_{r=1}^{R} \mathbf{E}_r \circ \mathbf{c}_r, \tag{3.2}$$

in which the rank of the matrices $\mathbf{E}_r \in \mathbb{R}^{I \times J}$ are L. $\mathbf{c}_r$ are vectors of length K.

So $\mathcal{X}$ is decomposed into a sum of matrix-vector outer products. If we decompose $\mathbf{E}_r$ into two matrices, i.e., $\mathbf{E}_r = \mathbf{A}_r \mathbf{B}_r^{\mathrm{T}}$, where the matrix $\mathbf{A}_r \in \mathbb{R}^{I \times L}$ and the matrix $\mathbf{B}_r \in \mathbb{R}^{J \times L}$ are rank L, then the equation (3.2) can be written as

$$\mathcal{X} = \sum_{r=1}^{R} (\mathbf{A}_r \mathbf{B}_r^{\mathrm{T}}) \circ \mathbf{c}_r. \tag{3.3}$$

The following Figure 6.1 shows the BTD-$(L, L, 1)$ for a third-order tensor.

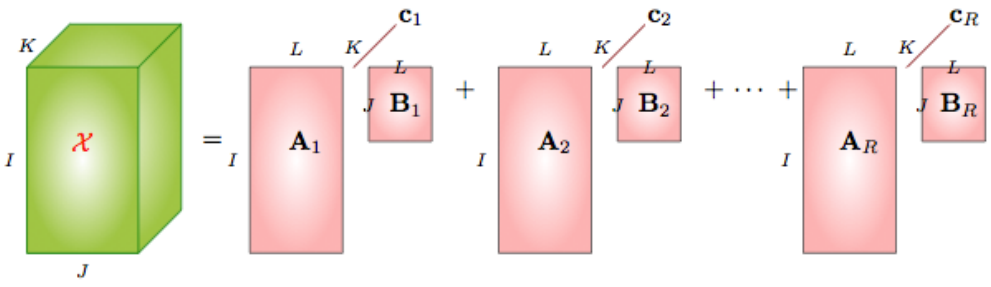


Figure 6.1: **BTD-$(L, L, 1)$ for $\mathcal{X} \in \mathbb{R}^{I \times J \times K}$**

Recall the CP decomposition (equation (2.1) in Chapter 3), for a third-order

tensor, $\mathcal{X}$ can be written as a sum of outer products of three vectors,

$$\mathcal{X} = \sum_{r=1}^{R} \mathbf{a}_r \circ \mathbf{b}_r \circ \mathbf{c}_r,$$

where $\mathbf{a}_r,\ \mathbf{b}_r,\ \mathbf{c}_r,\ r = 1, \ldots, R$ are vectors. When the matrices $\mathbf{A}_r$ and $\mathbf{B}_r$ in the equation (3.3) reduce to vectors, it is exactly same with the CP formula. Therefore, we say that CP decomposition is a special case of the BTD-$(L, L, 1)$.

Similar with the analysis of CP, we define $\mathbf{A} = [\mathbf{A}_1\ \cdots\ \mathbf{A}_R]$, $\mathbf{B} = [\mathbf{B}_1\ \cdots\ \mathbf{B}_R]$, $\mathbf{C} = [\mathbf{c}_1\ \cdots\ \mathbf{c}_R]$ and call them the factor matrices of the BTD. Take the three different modes matricization on the equation (3.3), we have (see [29])

$$\begin{aligned} \mathbf{X}_{(\mathbf{1})} &= \mathbf{A}(\mathbf{C} \odot_p \mathbf{B})^{\mathrm{T}}, \\ \mathbf{X}_{(\mathbf{2})} &= \mathbf{B}(\mathbf{C} \odot_p \mathbf{A})^{\mathrm{T}}, \\ \mathbf{X}_{(\mathbf{3})} &= \mathbf{C}[(\mathbf{B}_1 \odot \mathbf{A}_1)\mathbf{1}_L, \ldots, (\mathbf{B}_R \odot \mathbf{A}_R)\mathbf{1}_L]^{\mathrm{T}}, \end{aligned} \tag{3.4}$$

where $\mathbf{1}_L$ is a column vector of all ones of length L.

3.2 Decomposition in rank-(L, M, N) terms

Definition 3.3 (BTD-(L, M, N)) *[29] A decomposition of a tensor $\mathcal{X} \in \mathbb{R}^{I \times J \times K}$ into a sum of rank-(L, M, N) terms is a decomposition of $\mathcal{T}$ of the form*

$$\mathcal{X} = \sum_{r=1}^{R} \mathcal{G}_r \times_1 \mathbf{A}_r \times_2 \mathbf{B}_r \times_3 \mathbf{C}_r, \tag{3.5}$$

in which $\mathcal{G}_r \in \mathbb{R}^{L \times M \times N}$ are full rank-(L, M, N) and $\mathbf{A}_r \in \mathbb{R}^{I \times L}$ (with $I \geq L$), $\mathbf{B}_r \in \mathbb{R}^{J \times M}$ (with $J \geq M$), and $\mathbf{C}_r \in \mathbb{R}^{K \times N}$ (with $K \geq N$) are full column rank, $1 \leq r \leq R$.

The decomposition is visualized in the following Figure 6.2.

Recall the Tucker decomposition for a third-order tensor (equation (3.3)), so if

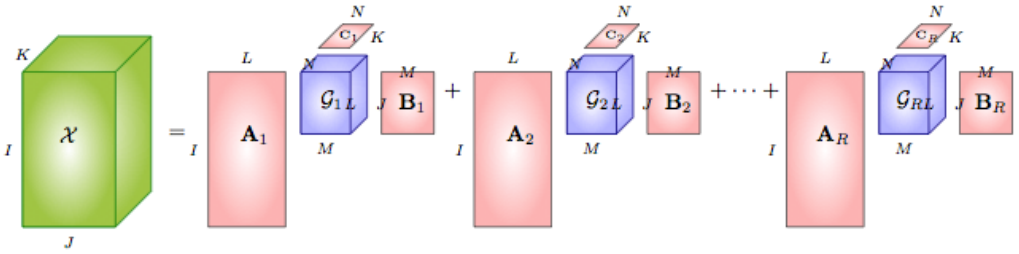


Figure 6.2: **BTD-(L, M, N) for a third-order tensor $\mathcal{X}$**

$R = 1$ in (3.5), the BTD-(L, M, N) is just a Tucker decomposition.

The factor matrices can be defined as $\mathbf{A} = [\mathbf{A}_1 \ \cdots \ \mathbf{A}_R] \in \mathbb{R}^{I \times LR}$, $\mathbf{B} = [\mathbf{B}_1 \ \cdots \ \mathbf{B}_R] \in \mathbb{R}^{J \times MR}$ and $\mathbf{C} = [\mathbf{C}_1 \ \cdots \ \mathbf{C}_R] \in \mathbb{R}^{K \times NR}$. We also define a new core tensor $\mathcal{G} \in \mathbb{R}^{LR \times MR \times NR}$, which is a block-diagonal tensor and its diagonal consists of all the small core tensors $\mathcal{G}_r$, $r = 1, 2, \ldots, R$. Then, the BTD-(L, M, N) can be viewed as one Tucker model,

$$\mathcal{X} = \sum_{r=1}^{R} \mathcal{G}_1 \times_1 \mathbf{A}_r \times_2 \mathbf{B}_r \times_3 \mathbf{C}_r = \mathcal{G} \times_1 \mathbf{A} \times_2 \mathbf{B} \times_3 \mathbf{C}.$$

The following figure 6.3 shows the decomposition,

Actually, BTD-$(L, L, 1)$ is also a special case of BTD-(L, M, N) [29]. The SVD of $\mathbf{E}_r$ in equation (3.2) provides us with a new formula of BTD-$(L, L, 1)$. Since

$$\mathbf{E}_r = \mathbf{A}_r \mathbf{\Sigma}_r \mathbf{B}_r^{\mathrm{T}},$$

then the equation (3.2) is equivalent to

$$\begin{aligned} \mathcal{X} &= \sum_{r=1}^{R} (\mathbf{A}_r \mathbf{\Sigma}_r \mathbf{B}_r^{\mathrm{T}}) \circ \mathbf{c}_r \\ &= \sum_{r=1}^{R} \mathbf{\Sigma}_r \times_1 \mathbf{A}_r \times_2 \mathbf{B}_r \times_3 \mathbf{c}_r. \end{aligned} \tag{3.6}$$

We consider the diagonal matrix $\mathbf{\Sigma}_r$ as a tensor of size $L \times L \times 1$. Therefore, by

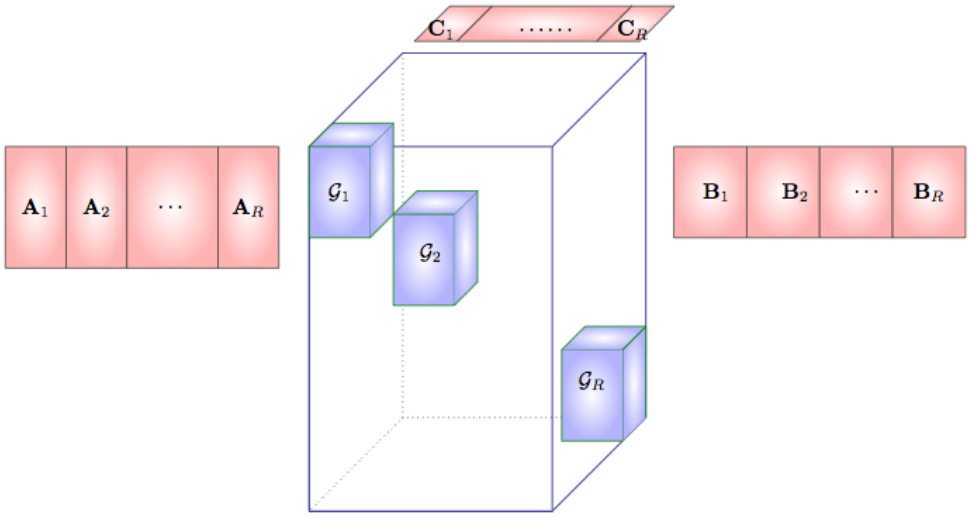


Figure 6.3: **the equivalent decomposition with BTD-(L, M, N) for a third-order tensor $\mathcal{X}$**

comparing the equations (3.6) and (3.5), we can see that BTD-(L, M, N) is a general decomposition which includes the BTD-$(L, L, 1)$.

Now, we also use the matricization technique to analyze the BTD-(L, M, N) equation (3.5):

$$\begin{aligned}
\mathbf{X}_{(1)} &= \mathbf{A}[\mathbf{G}_{\mathbf{1}(1)} \otimes \mathbf{e}_1 \quad \dots \quad \mathbf{G}_{\mathbf{R}(1)} \otimes \mathbf{e}_R](\mathbf{C} \odot_p \mathbf{B})^{\mathrm{T}}, \\
\mathbf{X}_{(2)} &= \mathbf{B}[\mathbf{G}_{\mathbf{1}(2)} \otimes \mathbf{e}_1 \quad \dots \quad \mathbf{G}_{\mathbf{R}(2)} \otimes \mathbf{e}_R](\mathbf{C} \odot_p \mathbf{A})^{\mathrm{T}}, \\
\mathbf{X}_{(3)} &= \mathbf{C}[\mathbf{G}_{\mathbf{1}(3)} \otimes \mathbf{e}_1 \quad \dots \quad \mathbf{G}_{\mathbf{R}(3)} \otimes \mathbf{e}_R](\mathbf{B} \odot_p \mathbf{A})^{\mathrm{T}}
\end{aligned} \tag{3.7}$$

where $\mathbf{e}_i = [0, \dots, 0, 1, 0, \dots, 0]^{\mathrm{T}}$ is a vector of length R and 1 is at the ith position, $1 \leq i \leq R$. So $[\mathbf{G}_{\mathbf{1}(1)} \otimes \mathbf{e}_1 \quad \dots \quad \mathbf{G}_{\mathbf{R}(1)} \otimes \mathbf{e}_R] \in \mathbb{R}^{LR \times MNR}$ is actually a block-diagonal matrix and the matrices $\mathbf{G}_{\mathbf{i}(1)}$, $1 \leq i \leq R$ are on the diagonal position.

Additionally, by using vectorization of $\mathcal{X}$, the equation (3.5) can be written as

$$vec(\mathcal{T}) = (\mathbf{A} \odot_p \mathbf{B} \odot_p \mathbf{C}) \begin{bmatrix} vec(\mathcal{G}_r) \\ \vdots \\ vec(\mathcal{G}_R) \end{bmatrix}, \tag{3.8}$$

where $vec(\mathcal{T})$ and $vec(\mathcal{G}_r)$, $r = 1, 2, \ldots, R$ mean that the vectorization of the tensors $\mathcal{T}$ and $\mathcal{G}_r$, respectively.

De Lathauwer also talks about the BTD uniqueness conditions up to permutation and scalings in [28, 29, 36]. Recall that in the third-order tensor CP decomposition, the earliest uniqueness condition is from Kruskal in 1977 [75, 77] which depends on the concept of k-rank. De Lathauwer introduces an analogous definition and generalizes Kruskal's condition to the BTD cases.

4 Numerical computation of BTD

4.1 Algorithm for solving BTD-$(L, L, 1)$

Given a third-order tensor $\mathcal{X} \in \mathbb{R}^{I \times J \times K}$, our problem is

$$\min_{\widehat{\mathcal{X}}} \|\mathcal{X} - \widehat{\mathcal{X}}\|_F^2, \quad \text{where } \widehat{\mathcal{X}} = \sum_{r=1}^{R} (\mathbf{A}_r \mathbf{B}_r^{\mathrm{T}}) \circ \mathbf{c}_r. \tag{4.1}$$

This problem is equivalent to

$$\min_{\mathbf{A},\mathbf{B},\mathbf{C}} \|\mathcal{X} - \sum_{r=1}^{R} (\mathbf{A}_r \mathbf{B}_r^{\mathrm{T}}) \circ \mathbf{c}_r\|_F^2, \tag{4.2}$$

with respect to the factor matrices $\mathbf{A}$, $\mathbf{B}$ and $\mathbf{C}$.

Then we can use the three-modes matricization (3.4), the problem has the follow-

ing three expressions,

$$\begin{aligned}
&\min_{\mathbf{A},\mathbf{B},\mathbf{C}} \quad \|\mathbf{X}_{(\mathbf{1})} - \mathbf{A}(\mathbf{C} \odot_p \mathbf{B})^{\mathrm{T}}\|_F^2, \\
&\min_{\mathbf{A},\mathbf{B},\mathbf{C}} \quad \|\mathbf{X}_{(\mathbf{2})} - \mathbf{B}(\mathbf{C} \odot_p \mathbf{A})^{\mathrm{T}}\|_F^2, \\
&\min_{\mathbf{A},\mathbf{B},\mathbf{C}} \quad \|\mathbf{X}_{(\mathbf{3})} - \mathbf{C}[(\mathbf{B}_1 \odot \mathbf{A}_1)\mathbf{1}_L, \ldots, (\mathbf{B}_R \odot \mathbf{A}_R)\mathbf{1}_L]^{\mathrm{T}}\|_F^2.
\end{aligned}$$

These three are equivalent. Instead of solving (4.1) for the three variables one time, we can use these three equations by fixing all factor matrices but one each time. Then the problem reduces to three coupled linear least-squares subproblems.

We have

$$\begin{aligned}
\mathbf{A}^{k+1} &= \underset{\widehat{\mathbf{A}} \in \mathbb{R}^{I \times LR}}{\operatorname{argmin}} \|\mathbf{X}_{(\mathbf{1})} - \widehat{\mathbf{A}}(\mathbf{C}^k \odot_p \mathbf{B}^k)^{\mathrm{T}}\|_F^2, \\
\mathbf{B}^{k+1} &= \underset{\widehat{\mathbf{B}} \in \mathbb{R}^{J \times LR}}{\operatorname{argmin}} \|\mathbf{X}_{(\mathbf{2})} - \widehat{\mathbf{B}}(\mathbf{C}^k \odot_p \mathbf{A}^{k+1})^{\mathrm{T}}\|_F^2, \\
\mathbf{C}^{k+1} &= \underset{\widehat{\mathbf{C}} \in \mathbb{R}^{K \times R}}{\operatorname{argmin}} \|\mathbf{X}_{(\mathbf{3})} - \widehat{\mathbf{C}}[(\mathbf{B}_1^{k+1} \odot \mathbf{A}_1^{k+1})\mathbf{1}_L, \ldots, (\mathbf{B}_R^{k+1} \odot \mathbf{A}_R^{k+1})\mathbf{1}_L]^{\mathrm{T}}\|_F^2,
\end{aligned} \tag{4.3}$$

where $\mathbf{X}_{(\mathbf{1})} \in \mathbb{R}^{I \times JK}$, $\mathbf{X}_{(\mathbf{2})} \in \mathbb{R}^{J \times IK}$ and $\mathbf{X}_{(\mathbf{3})} \in \mathbb{R}^{K \times IJ}$ are the mode-1, mode-2 and mode-3 matricizations of tensor $\mathcal{X}$.

Thus, like the ALS method for the CP decomposition, the method of computing the block term decomposition in rank-$(L, L, 1)$ is also based on the three equations in (4.3) to update the three factor matrices $\mathbf{A}$, $\mathbf{B}$ and $\mathbf{C}$ alternatively. So, given the initial factor matrices $\mathbf{A}^0$, $\mathbf{B}^0$ and $\mathbf{C}^0$, then at the $(k+1)$th iteration, we hold $\mathbf{B}^k$ and $\mathbf{C}^k$ to update the factor $\mathbf{A}$ to get $\mathbf{A}^{k+1}$, then $\mathbf{A}^{k+1}$ and $\mathbf{C}^k$ are held to update $\mathbf{B}$ and obtain $\mathbf{B}^{k+1}$. Similarly, we hold $\mathbf{A}^{k+1}$ and $\mathbf{B}^{k+1}$ to obtain $\mathbf{C}^{k+1}$. Usually, the Frobenius norm of the error between the given tensor and the updated tensor is measured at each iteration to provide a convergence stopping criterion. Analogous to the situation of CP algorithm, the each subproblem is a linear least-squares problem. The outline of the algorithm of BTD-$(L, L, 1)$ is given in the following Table 6.4 [36].

BTD-$(L, L, 1)$-Algorithm

procedure BTD-$(L, L, 1)(\mathcal{X}, R, N)$

give initial guess $\mathbf{A}^0 \in \mathbb{R}^{I\times R}$, $\mathbf{B}^0 \in \mathbb{R}^{J\times R}$, $\mathbf{C}^0 \in \mathbb{R}^{K\times R}$

for $n = 1, \ldots, N$ **do**

$\mathbf{A}^{n+1} \leftarrow \mathbf{X}_{(\mathbf{1})}/(\mathbf{C}^n \odot_p \mathbf{B}^n)^{\mathrm{T}}$ —— % solving least squares to update $\mathbf{A}$

$\mathbf{B}^{n+1} \leftarrow \mathbf{X}_{(\mathbf{2})}/(\mathbf{C}^n \odot_p \mathbf{A}^{n+1})^{\mathrm{T}}$ —% solving least squares to update $\mathbf{B}$

$\mathbf{C}^{n+1} \leftarrow \mathbf{X}_{(\mathbf{3})}/[(\mathbf{B}_1^{n+1} \odot \mathbf{A}_1^{n+1})\mathbf{1}_L, \ldots, (\mathbf{B}_R^{n+1} \odot \mathbf{A}_R^{n+1})\mathbf{1}_L]^{\mathrm{T}}$
—— % solving least squares to update $\mathbf{C}$

end for

return $\mathbf{A}^N$, $\mathbf{B}^N$, $\mathbf{C}^N$

end procedure

Table 6.4: **Algorithm of BTD-$(L, L, 1)$ with rank R for a third-order tensor** $\mathcal{X} \in \mathbb{R}^{I\times J\times K}$

The number of iterations N is set to a large number; and a convergence stopping criterion can be used. Notice that in the algorithm, we made the number R (the number of components in the BTD-$(L, L, 1)$ formula) is known. We know that the problem of determining the rank of a tensor is NP-hard. Similarly, it is NP-hard to determine the number of components in BTD-$(L, L, 1)$ and BTD-(L, M, N).

4.2 Algorithm for solving BTD-(L, M, N)

The problem we want to solve is the following: given a third-order tensor $\mathcal{X} \in \mathbb{R}^{I\times J\times K}$, compute the BTD-$(L, M, N)$ with R components that best approximates $\mathcal{X}$. So, it can be considered as the following problem:

$$\min_{\widehat{\mathcal{X}}} \|\mathcal{X} - \widehat{\mathcal{X}}\|_F^2, \quad \text{where } \widehat{\mathcal{X}} = \sum_{r=1}^{R} \mathcal{G}_r \times_1 \mathbf{A}_r \times_2 \mathbf{B}_r \times_3 \mathbf{C}_r. \tag{4.4}$$

This problem is equivalent to

$$\min_{\mathbf{A},\mathbf{B},\mathbf{C},\mathcal{G}_r} \|\mathcal{X} - \sum_{r=1}^{R} \mathcal{G}_r \times_1 \mathbf{A}_r \times_2 \mathbf{B}_r \times_3 \mathbf{C}_r\|_F^2, \tag{4.5}$$

with respect to the factor matrices $\mathbf{A}$, $\mathbf{B}$, $\mathbf{C}$ and the core tensors $\mathcal{G}_r$, $r = 1, 2, \ldots, R$.

By using the three-modes matricization and vectorization (3.7), (3.8) on the above equation (4.5), the problem has the expressions

$$\begin{aligned}
&\min_{\mathbf{A},\mathbf{B},\mathbf{C},\mathcal{G}_r} \quad \|\mathbf{X}_{(\mathbf{1})} - \mathbf{A} \cdot [\mathbf{G}_{\mathbf{1}(\mathbf{1})} \otimes \mathbf{e}_1 \quad \ldots \quad \mathbf{G}_{\mathbf{R}(\mathbf{1})} \otimes \mathbf{e}_R] \cdot (\mathbf{C} \odot_p \mathbf{B})^{\mathrm{T}}\|_F^2, \\
&\min_{\mathbf{A},\mathbf{B},\mathbf{C},\mathcal{G}_r} \quad \|\mathbf{X}_{(\mathbf{2})} - \mathbf{B} \cdot [\mathbf{G}_{\mathbf{1}(\mathbf{2})} \otimes \mathbf{e}_1 \quad \ldots \quad \mathbf{G}_{\mathbf{R}(\mathbf{2})} \otimes \mathbf{e}_R] \cdot (\mathbf{C} \odot_p \mathbf{A})^{\mathrm{T}}\|_F^2, \\
&\min_{\mathbf{A},\mathbf{B},\mathbf{C},\mathcal{G}_r} \quad \|\mathbf{X}_{(\mathbf{3})} - \mathbf{C} \cdot [\mathbf{G}_{\mathbf{1}(\mathbf{3})} \otimes \mathbf{e}_1 \quad \ldots \quad \mathbf{G}_{\mathbf{R}(\mathbf{3})} \otimes \mathbf{e}_R] \cdot (\mathbf{B} \odot_p \mathbf{A})^{\mathrm{T}}\|_F^2, \\
&\min_{\mathbf{A},\mathbf{B},\mathbf{C},\mathcal{G}_r} \quad \left\| vec(\mathcal{X}) - (\mathbf{A} \odot_p \mathbf{B} \odot_p \mathbf{C}) \cdot \begin{bmatrix} vec(\mathcal{G}_1) \\ \vdots \\ vec(\mathcal{G}_R) \end{bmatrix} \right\|_F^2.
\end{aligned} \tag{4.6}$$

These four are equivalent. So, instead of solving (4.4) for the four variables one time, we can use these four equations by fixing all factor matrices except one each time so that the problem reduces to four least-squares subproblems. Then we have

$$\begin{aligned}
\mathbf{A}^{k+1} &= \operatorname*{argmin}_{\widehat{\mathbf{A}} \in \mathbb{R}^{I \times LR}} \|\mathbf{X}_{(1)} - \widehat{\mathbf{A}} \cdot [\mathbf{G}_{\mathbf{1}(1)}^{k} \otimes \mathbf{e}_1 \quad \ldots \quad \mathbf{G}_{\mathbf{R}(1)}^{k} \otimes \mathbf{e}_R] \cdot (\mathbf{C}^k \odot_p \mathbf{B}^k)^{\mathrm{T}}\|_F^2, \\
\mathbf{B}^{k+1} &= \operatorname*{argmin}_{\widehat{\mathbf{B}} \in \mathbb{R}^{J \times MR}} \|\mathbf{X}_{(2)} - \widehat{\mathbf{B}} \cdot [\mathbf{G}_{\mathbf{1}(2)}^{k} \otimes \mathbf{e}_1 \quad \ldots \quad \mathbf{G}_{\mathbf{R}(2)}^{k} \otimes \mathbf{e}_R] \cdot (\mathbf{C}^k \odot_p \mathbf{A}^{k+1})^{\mathrm{T}}\|_F^2, \\
\mathbf{C}^{k+1} &= \operatorname*{argmin}_{\widehat{\mathbf{C}} \in \mathbb{R}^{K \times NR}} \|\mathbf{X}_{(3)} - \widehat{\mathbf{C}} \cdot [\mathbf{G}_{\mathbf{1}(3)}^{k} \otimes \mathbf{e}_1 \quad \ldots \quad \mathbf{G}_{\mathbf{R}(3)}^{k} \otimes \mathbf{e}_R] \cdot (\mathbf{B}^{k+1} \odot_p \mathbf{A}^{k+1})^{\mathrm{T}}\|_F^2, \\
\mathcal{G}_r^{k+1} &= \operatorname*{argmin}_{\widehat{\mathcal{G}}_r \in \mathbb{R}^{L \times M \times N}} \left\| vec(\mathcal{X}) - (\mathbf{A}^{k+1} \odot_p \mathbf{B}^{k+1} \odot_p \mathbf{C}^{k+1}) \cdot \begin{bmatrix} vec(\widehat{\mathcal{G}}_1) \\ \vdots \\ vec(\widehat{\mathcal{G}}_R) \end{bmatrix} \right\|_F^2,
\end{aligned} \tag{4.7}$$

where $\mathbf{X}_{(1)} \in \mathbb{R}^{I \times JK}$, $\mathbf{X}_{(2)} \in \mathbb{R}^{J \times IK}$ and $\mathbf{X}_{(3)} \in \mathbb{R}^{K \times IJ}$ are the mode-1, mode-2 and mode-3 matricizations of tensor $\mathcal{X}$. And $vec(\mathcal{X})$ is the vectorization of $\mathcal{X}$.

Therefore, we can use an alternating least-squares method to solve for the BTD-(L, M, N). Given three initial factor matrices $\mathbf{A}^0$, $\mathbf{B}^0$, $\mathbf{C}^0$, and the initial core tensors $\mathcal{G}_r^0$, $r = 1, 2, \ldots, R$, the algorithm solves the four least-squares subproblems in (4.7) to obtain the factor matrices $\mathbf{A}$, $\mathbf{B}$, $\mathbf{C}$ and the corresponding core tensors $\mathcal{G}_r$. Starting from the initials, the approach fixes $\mathbf{B}$, $\mathbf{C}$ and $\mathcal{G}_r$ to solve for $\mathbf{A}$, then fixes $\mathbf{A}$, $\mathbf{C}$ and $\mathcal{G}_r$ to solve for $\mathbf{B}$, and then fixes $\mathbf{A}$, $\mathbf{B}$ and $\mathcal{G}_r$ to obtain the factor $\mathbf{C}$. Finally, by fixing $\mathbf{A}$, $\mathbf{B}$ and $\mathbf{C}$, we can calculate $\mathcal{G}_r$ by using the fourth equation in (4.7). This process continues iteratively until some convergence criterion is satisfied. Usually, two types of stopping criterions are used. One is when the Frobenius norm of the residual between the given tensor and the updated tensor is small enough, and the other one is when the difference between the current update of one factor matrix and the corresponding matrix in the previous iteration is small enough. The outline of the algorithm BTD-(L, M, N) is given in the table 6.5 [36]. The number of iterations N is set to a large number; and a convergence stopping criterion can be used.

The two algorithms in [36] require the coefficient matrix in each subproblem to have at least as many rows as columns, and to have full column rank. Also, in order to prevent the submatrices of the factor matrix from becoming ill-conditioned, [36] QR factorization is used to normalize the each submatrix. But such normalization cannot guarantee the Frobenius norm of the error between the given tensor and the updated tensor will always decrease. However, if the initial guesses are good enough, such problem only happens at the beginning iterations and the error will go down eventually.

BTD-(L, M, N)-Algorithm

procedure BTD-$(L, M, N)(\mathcal{X}, R, N)$

give initial guess $\mathbf{A}^0 \in \mathbb{R}^{I\times R}$, $\mathbf{B}^0 \in \mathbb{R}^{J\times R}$, $\mathbf{C}^0 \in \mathbb{R}^{K\times R}$, $\mathcal{D}_r \in \mathbb{R}^{L\times M\times N}$ for $r = 1, 2, \ldots, R$

for $n = 1, \ldots, N$ **do**

$$\mathbf{A}^{n+1} \leftarrow \mathbf{X}_{(\mathbf{1})}/[(\mathbf{D}_{\mathbf{1}(1)}^n \otimes \mathbf{e}_1, \ldots, \mathbf{D}_{\mathbf{R}(1)}^n \otimes \mathbf{e}_R) \cdot (\mathbf{C}^n \odot_p \mathbf{B}^n)^{\mathrm{T}}]$$

—— % solving least squares to update $\mathbf{A}$

$$\mathbf{B}^{n+1} \leftarrow \mathbf{X}_{(\mathbf{2})}/[(\mathbf{D}_{\mathbf{1}(2)}^n \otimes \mathbf{e}_1, \ldots, \mathbf{D}_{\mathbf{R}(2)}^n \otimes \mathbf{e}_R) \cdot (\mathbf{C}^n \odot_p \mathbf{A}^{n+1})^{\mathrm{T}}]$$

——% solving least squares to update $\mathbf{B}$

$$\mathbf{C}^{n+1} \leftarrow \mathbf{X}_{(\mathbf{3})}/[(\mathbf{D}_{\mathbf{1}(3)}^n \otimes \mathbf{e}_1, \ldots, \mathbf{D}_{\mathbf{R}(3)}^n \otimes \mathbf{e}_R) \cdot (\mathbf{B}^{n+1} \odot_p \mathbf{A}^{n+1})^{\mathrm{T}}]$$

—— % solving least squares to update $\mathbf{C}$

$$\begin{bmatrix} vec(\mathcal{D}_1) \\ \vdots \\ vec(\mathcal{D}_R) \end{bmatrix} \leftarrow (\mathbf{A}^{n+1} \odot_p \mathbf{B}^{n+1} \odot_p \mathbf{C}^{n+1}) \backslash vec(\mathcal{T})$$

—— % solving least squares to update $\mathcal{D}_r$, $r = 1, \ldots, R$

end for

return $\mathbf{A}^N$, $\mathbf{B}^N$, $\mathbf{C}^N$, $\mathcal{D}_r^N$, $r = 1, \ldots, R$

end procedure

Table 6.5: **Algorithm of BTD-(L, M, N) with rank R for a third-order tensor** $\mathcal{X} \in \mathbb{R}^{I\times J\times K}$

5 Regularization method for solving BTD

In this section, we will view the BTD-$(L, L, 1)$ and BTD-(L, M, N) from the perspective of optimization. We will also make a connection between the BTD-ALS for both BTD-$(L, L, 1)$ and BTD-(L, M, N) and the Gauss-Seidel (GS) method.

5.1 BTD-$(L, L, 1)$ as a nonlinear optimization

Recall the BTD-$(L, L, 1)$ problem from Section 4.1. For a given third-order tensor $\mathcal{X} \in \mathbb{R}^{I\times J\times K}$ the problem becomes

$$\min_{\widehat{\mathcal{X}}} \|\mathcal{X} - \widehat{\mathcal{X}}\|_F^2, \quad \text{where } \widehat{\mathcal{X}} = \sum_{r=1}^{R} (\mathbf{A}_r \mathbf{B}_r^{\mathrm{T}}) \circ \mathbf{c}_r.$$

So, we have the following expression about the cost functional,

$$\|\mathcal{X} - \widehat{\mathcal{X}}\|_F^2 = \sum_{i=1}^{I}\sum_{j=1}^{J}\sum_{k=1}^{K}\left(x_{ijk} - \sum_{r=1}^{R}(\sum_{l=1}^{L} a_{il}^{(r)} b_{jl}^{(r)}) c_k^{(r)}\right)^2, \tag{5.1}$$

where $a_{il}^{(r)}$ denotes the il element (ith row and lth column) of the matrix $\mathbf{A}_r$, $b_{jl}^{(r)}$ expresses the jl element of the matrix $\mathbf{B}_r$, and $c_k^{(r)}$ means the kth element in the vector $\mathbf{c}_r$.

Then, $\|\mathcal{X} - \widehat{\mathcal{X}}\|_F^2$ is a function from $\mathbf{x}$ to $\mathbb{R}$, and

$$\mathbf{x} = vec([vec(\mathbf{A}), vec(\mathbf{B}), vec(\mathbf{C})]) \in \mathbb{R}^n,$$

where $\mathbf{A} = [\mathbf{A}_1 \ \ \ldots \ \ \mathbf{A}_R]$, $\mathbf{B} = [\mathbf{B}_1 \ \ \ldots \ \ \mathbf{B}_R]$ and $\mathbf{C} = [\mathbf{c}_1 \ \ \ldots \ \ \mathbf{c}_R]$ are the factor matrices of $\widehat{\mathcal{X}}$ and $n = (IL + JL + K)R$.

Let $vec(\mathbf{A}) = \mathbf{x}_1 \in \mathbb{R}^{IRL}$, $vec(\mathbf{B}) = \mathbf{x}_2 \in \mathbb{R}^{JRL}$ and $vec(\mathbf{C}) = \mathbf{x}_3 \in \mathbb{R}^{KR}$. We partition the vector $\mathbf{x} \in \mathbb{R}^n$ into 3 component vectors $\mathbf{x}_i \in \mathbb{R}^{n_i}$, $i = 1, 2, 3$, where $n_1 = IRL$, $n_2 = JRL$ and $n_3 = KR$. It follows that $\mathbf{x} = \mathbf{x}_1 \times \mathbf{x}_2 \times \mathbf{x}_3 \in \mathbb{R}^{n_1} \times \mathbb{R}^{n_2} \times \mathbb{R}^{n_3} = \mathbb{R}^n$. Thus, the BTD-$(L, L, 1)$ can be reformulated to the following problem,

$$\begin{aligned} \text{minimize} \quad & f(\mathbf{x}) = \sum_{i=1}^{I}\sum_{j=1}^{J}\sum_{k=1}^{K}\left(x_{ijk} - \sum_{r=1}^{R}(\sum_{l=1}^{L} a_{il}^{(r)} b_{jl}^{(r)}) c_k^{(r)}\right)^2 \\ \text{subject to} \quad & \mathbf{x} \in \mathbb{R}^{n_1} \times \mathbb{R}^{n_2} \times \mathbb{R}^{n_3} = \mathbb{R}^n. \end{aligned} \tag{5.2}$$

According to the BTD-ALS algorithm, the updates are in terms of the components of $\mathbf{x}$. Starting from a given initial point $\mathbf{x}^0 = vec([vec(\mathbf{A}^0), vec(\mathbf{B}^0), vec(\mathbf{C}^0)]) \in \mathbb{R}^n$, a sequence $\{(\mathbf{x}_1^k, \mathbf{x}_2^k, \mathbf{x}_3^k)\}$ is generated by the following equations

$$\begin{aligned} \mathbf{x}_1^{k+1} &= \underset{\mathbf{y}\in\mathbb{R}^{n_1}}{\operatorname{argmin}} f(\mathbf{y}, \mathbf{x}_2^k, \mathbf{x}_3^k), \\ \mathbf{x}_2^{k+1} &= \underset{\mathbf{y}\in\mathbb{R}^{n_2}}{\operatorname{argmin}} f(\mathbf{x}_1^{k+1}, \mathbf{y}, \mathbf{x}_3^k), \\ \mathbf{x}_3^{k+1} &= \underset{\mathbf{y}\in\mathbb{R}^{n_3}}{\operatorname{argmin}} f(\mathbf{x}_1^{k+1}, \mathbf{x}_2^{k+1}, \mathbf{y}). \end{aligned} \tag{5.3}$$

Recall the block nonlinear Gauss-Seidel method (GS) for solving the nonlinear minimization problem in Chapter 3. We can see that the BTD-ALS is exactly the GS method for solving the nonlinear problem (5.2) which is equivalent to the original BTD-$(L, L, 1)$ problem.

5.2 BTD-(L, M, N) as a nonlinear optimization

For a given third-order tensor, the BTD-(L, M, N) problem (4.4) is

$$\min_{\widehat{\mathcal{X}}} \|\mathcal{X} - \widehat{\mathcal{X}}\|_F^2, \quad \text{where } \widehat{\mathcal{X}} = \sum_{r=1}^{R} \mathcal{G}_r \times_1 \mathbf{A}_r \times_2 \mathbf{B}_r \times_3 \mathbf{C}_r.$$

So, by expressing the above cost functional in an element-wise way, we have the following equation,

$$\|\mathcal{X} - \widehat{\mathcal{X}}\|_F^2 = \sum_{i=1}^{I}\sum_{j=1}^{J}\sum_{k=1}^{K}\left(x_{ijk} - \sum_{r=1}^{R}\sum_{l=1}^{L}\sum_{m=1}^{M}\sum_{n=1}^{N} g_{lmn}^{(r)} a_{il}^{(r)} b_{jm}^{(r)} c_{kn}^{(r)}\right)^2, \tag{5.4}$$

where $g_{lmn}^{(r)}$ is the element of the rth core tensor $\mathcal{G}_r$ at the lmn position, $a_{il}^{(r)}$ denotes the element of the matrix $\mathbf{A}_r$ at the il position, $b_{jm}^{(r)}$ means the element of the matrix $\mathbf{B}_r$ at the jm position and $c_{kn}^{(r)}$ is the element of the matrix $\mathbf{C}_r$ at the kn position.

Then, $\|\mathcal{X} - \widehat{\mathcal{X}}\|_F^2$ is a function from $\mathbf{x}$ to $\mathbb{R}$, and

$$\mathbf{x} = vec([vec(\mathbf{A}), vec(\mathbf{B}), vec(\mathbf{C}), vec(\mathcal{G}_1), \ldots, vec(\mathcal{G}_R)]) \in \mathbb{R}^n,$$

where $\mathbf{A} = [\mathbf{A}_1 \ \ldots \ \mathbf{A}_R]$, $\mathbf{B} = [\mathbf{B}_1 \ \ldots \ \mathbf{B}_R]$ and $\mathbf{C} = [\mathbf{C}_1 \ \ldots \ \mathbf{C}_R]$ are the factor matrices of $\widehat{\mathcal{X}}$. And $n = (IL + JM + KN + LMN)R$.

Let $vec(\mathbf{A}) = \mathbf{x}_1 \in \mathbb{R}^{ILR}$, $vec(\mathbf{B}) = \mathbf{x}_2 \in \mathbb{R}^{JMR}$, $vec(\mathbf{C}) = \mathbf{x}_3 \in \mathbb{R}^{KNR}$ and $vec([vec(\mathcal{G}_1), \ldots, vec(\mathcal{G}_R)]) = \mathbf{x}_4 \in \mathbb{R}^{LMNR}$, so that we partition the vector $\mathbf{x} \in \mathbb{R}^n$ into 4 component vectors $\mathbf{x}_i \in R^{n_i}$, $i = 1, 2, 3, 4$, where $n_1 = ILR$, $n_2 = JMR$, $n_3 = KNR$ and $n_4 = LMNR$. It follows that $\mathbf{x} = \mathbf{x}_1 \times \mathbf{x}_2 \times \mathbf{x}_3 \times \mathbf{x}_4 \in \mathbb{R}^{n_1} \times \mathbb{R}^{n_2} \times \mathbb{R}^{n_3} \times \mathbb{R}^{n_4} = \mathbb{R}^n$. Thus, the BTD-$(L, M, N)$ can be reformulated to the following problem,

$$\begin{array}{ll} \text{minimize} & f(\mathbf{x}) = \displaystyle\sum_{i=1}^{I}\sum_{j=1}^{J}\sum_{k=1}^{K}\left(x_{ijk} - \sum_{r=1}^{R}\sum_{l=1}^{L}\sum_{m=1}^{M}\sum_{n=1}^{N} g_{lmn}^{(r)} a_{il}^{(r)} b_{jm}^{(r)} c_{kn}^{(r)}\right)^2 \\ \text{subject to} & \mathbf{x} \in \mathbb{R}^{n_1} \times \mathbb{R}^{n_2} \times \mathbb{R}^{n_3} \times \mathbb{R}^{n_4} = \mathbb{R}^n. \end{array} \tag{5.5}$$

Then, we can see that the BTD-ALS algorithm for solving BTD-(L, M, N) is in terms of the components of $\mathbf{x}$, starting from a given initial point

$$\mathbf{x}^0 = vec([vec(\mathbf{A}^0), vec(\mathbf{B}^0), vec(\mathbf{C}^0), vec(\mathcal{G}_1^0), \ldots, vec(\mathcal{G}_R^0)]) \in \mathbb{R}^n,$$

and a sequence $\{(\mathbf{x}_1^k, \mathbf{x}_2^k, \mathbf{x}_3^k, \mathbf{x}_4^k)\}$ is generated by the following equations

$$\begin{array}{rcl} \mathbf{x}_1^{k+1} & = & \underset{\mathbf{y}\in\mathbb{R}^{n_1}}{\operatorname{argmin}} f(\mathbf{y}, \mathbf{x}_2^k, \mathbf{x}_3^k, \mathbf{x}_4^k), \\ \mathbf{x}_2^{k+1} & = & \underset{\mathbf{y}\in\mathbb{R}^{n_2}}{\operatorname{argmin}} f(\mathbf{x}_1^{k+1}, \mathbf{y}, \mathbf{x}_3^k, \mathbf{x}_4^k), \\ \mathbf{x}_3^{k+1} & = & \underset{\mathbf{y}\in\mathbb{R}^{n_3}}{\operatorname{argmin}} f(\mathbf{x}_1^{k+1}, \mathbf{x}_2^{k+1}, \mathbf{y}, \mathbf{x}_4^k), \\ \mathbf{x}_4^{k+1} & = & \underset{\mathbf{y}\in\mathbb{R}^{n_4}}{\operatorname{argmin}} f(\mathbf{x}_1^{k+1}, \mathbf{x}_2^{k+1}, \mathbf{x}_3^{k+1}, \mathbf{y}). \end{array} \tag{5.6}$$

So, we can conclude that the BTD-ALS for BTD-(L, M, N) is exactly the GS method for solving the nonlinear problem (4.5) which is equivalent to the original BTD-(L, M, N) problem.

We can use some of the convergence properties of the GS method discussed in Chap. 3 to discuss the BTD-ALS method. In general, the GS method may not converge in the sense that it may generate a sequence with limit points that are not critical points of the original problem. Theorem 3.6 and Theorem 3.7 in Chap. 3 provide two different conditions for the convergence of the GS method. One condition requires the each step to have a unique solution, and the other one requires that the function f is strictly quasiconvex with respect to all the $\mathbf{x}_i$ on X except two. We see that each subproblem in BTD-$(L, L, 1)$ and BTD-(L, M, N) is a least-squares problem which does not has the unique solution. Also, in Chap. 3, we have showed that the f is not a strictly quasiconvex function which cannot satisfy the condition of Theorem 3.7. Therefore, in general, we cannot guarantee that the BTD-ALS methods converge to a global minimum or even a critical point of the original cost functional.

5.3 BTD-ALS swamp

Now, let us discuss another effect of the BTD-ALS algorithm which is called swamp. Like the ALS method in CP decomposition, we found that the BTD-ALS sometimes illustrates swamp behavior that needs an exceedingly high number of iterations to converge. The following Figure 6.4 shows that this swamp behavior occurs in the BTD-ALS method for BTD-$(2, 2, 2)$ with rank $R = 2$. The given tensor is a third-order tensor of size $(15 \times 16 \times 30)$.

5.4 Regularized method: BTD-RALS

Some disadvantages of the BTD-ALS method are shown and discussed in the previous section. Since both the BTD-$(L, L, 1)$ and BTD-(L, M, N) can be transformed to

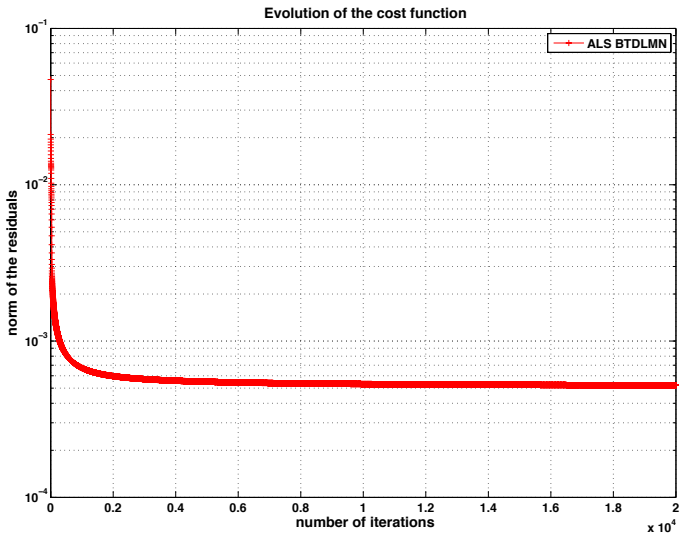


Figure 6.4: **Numerical example for swamp in BTD-ALS, it takes over 20000 iterations**

several least-squared subproblems, they can be solved using an alternating least-squares method. We can introduce a regularized algorithm proposed by Navasca, De Lathauwer and Kindermann [91] for improving the ALS method in CP decomposition. In this section, we examine the regularized method, which we denoted by BTD-RALS and will show that it is also the proximal point modification of the Gauss-Seidel method (PGS) ([9, 46]). In addition, several numerical examples are presented later to show that the BTD-RALS converges faster than ALS and decreases the high number of BTD-ALS iterations, thereby removing the swamp.

According to the discussion above, we have made an equivalent connection between the BTD-ALS method and the GS method for solving the BTD nonlinear optimization problem. Therefore, since the RALS method is also known as Proximal Point modification of the Gauss-Seidel method (PGS) (Section 4.4 in Chap. 3) which is a regularized method for the GS, it is natural to bring the regularized ALS method [91] to regularize the BTD-ALS method.

So, by introducing the regularization parameter λ_n, we can obtain the correspond-

ing new algorithms for BTD-$(L, L, 1)$ and BTD-(L, M, N). We call them RBTD-$(L, L, 1)$ and RBTD-(L, M, N).

We add the regularization item for each subproblem in the equations (3.4),

$$\begin{aligned}
\mathbf{A}^{n+1} &= \underset{\widehat{\mathbf{A}}\in\mathbb{R}^{I\times LR}}{\operatorname{argmin}} \|\mathbf{X}_{(\mathbf{1})} - \widehat{\mathbf{A}}(\mathbf{C}^n \odot_p \mathbf{B}^n)^{\mathrm{T}}\|_F^2 + \lambda_n\|\widehat{\mathbf{A}} - \mathbf{A}^n\|_F^2, \\
\mathbf{B}^{k+1} &= \underset{\widehat{\mathbf{B}}\in\mathbb{R}^{J\times LR}}{\operatorname{argmin}} \|\mathbf{X}_{(\mathbf{2})} - \widehat{\mathbf{B}}(\mathbf{C}^n \odot_p \mathbf{A}^{n+1})^{\mathrm{T}}\|_F^2 + \lambda_n\|\widehat{\mathbf{B}} - \mathbf{B}^n\|_F^2, \\
\mathbf{C}^{k+1} &= \underset{\widehat{\mathbf{C}}\in\mathbb{R}^{K\times R}}{\operatorname{argmin}} \|\mathbf{X}_{(\mathbf{3})} - \widehat{\mathbf{C}}[(\mathbf{B_1}^{n+1} \odot \mathbf{A_1}^{n+1})\mathbf{1}_L, \ldots, (\mathbf{B_R}^{n+1} \odot \mathbf{A_R}^{n+1})\mathbf{1}_L]^{\mathrm{T}}\|_F^2 + \lambda_n\|\widehat{\mathbf{C}} - \mathbf{C}^n\|_F^2.
\end{aligned} \tag{5.7}$$

So the RBTD-$(L, L, 1)$ algorithm is summarized in the Table 6.6. In the algorithm, the regularization parameters λ_n, $n = 0, 1, \ldots$ are given by a nonnegative decreasing sequence. Notice that at each iteration, the parameters are the same for the three updated factor matrices. Such a parameter choice also follows the RALS algorithm (see Chap. 3 and [91]).

In terms of RBTD-(L, M, N) algorithm, we regularize the first three equations for updating the factor matrices $\mathbf{A}$, $\mathbf{B}$ and $\mathbf{C}$. The rule of the parameter choice is also same with the previous regularization. So, the three subproblems in (3.7) become

$$\begin{aligned}
\mathbf{A}^{n+1} &= \underset{\widehat{\mathbf{A}}\in\mathbb{R}^{I\times LR}}{\operatorname{argmin}} \|\mathbf{X}_{(\mathbf{1})} - \widehat{\mathbf{A}}[\mathbf{D}_{\mathbf{1}_{(1)}}^n \otimes \mathbf{e}_1, \ldots, \mathbf{D}_{\mathbf{R}_{(1)}}^n \otimes \mathbf{e}_R](\mathbf{C}^n \odot_p \mathbf{B}^n)^{\mathrm{T}}\|_F^2 \\
&\quad +\lambda_n\|\widehat{\mathbf{A}} - \mathbf{A}^n\|_F^2, \\
\mathbf{B}^{n+1} &= \underset{\widehat{\mathbf{B}}\in\mathbb{R}^{J\times MR}}{\operatorname{argmin}} \|\mathbf{X}_{(\mathbf{2})} - \widehat{\mathbf{B}}[\mathbf{D}_{\mathbf{1}_{(2)}}^n \otimes \mathbf{e}_1, \ldots, \mathbf{D}_{\mathbf{R}_{(2)}}^n \otimes \mathbf{e}_R](\mathbf{C}^n \odot_p \mathbf{A}^{n+1})^{\mathrm{T}}\|_F^2 \\
&\quad +\lambda_n\|\widehat{\mathbf{B}} - \mathbf{B}_n\|_F^2, \\
\mathbf{C}^{n+1} &= \underset{\widehat{\mathbf{C}}\in\mathbb{R}^{K\times NR}}{\operatorname{argmin}} \|\mathbf{X}_{(\mathbf{3})} - \widehat{\mathbf{C}}[\mathbf{D}_{\mathbf{1}_{(3)}}^n \otimes \mathbf{e}_1, \ldots, \mathbf{D}_{\mathbf{R}_{(3)}}^n \otimes \mathbf{e}_R](\mathbf{B}^{n+1} \odot_p \mathbf{A}^{n+1})^{\mathrm{T}}\|_F^2 \\
&\quad +\lambda_n\|\widehat{\mathbf{C}} - \mathbf{C}_n\|_F^2.
\end{aligned} \tag{5.8}$$

The details about the algorithm is shown in the Table 6.7.

In the above two algorithms, the regularized parameter choice follows from the regularized parameter choice rule [88], and also [91] uses it into the RALS algorithm

RBTD-$(L, L, 1)$-Algorithm [91]

procedure RBTD-$(L, L, 1)(\mathcal{X}, R, N, \lambda_n)$

give initial guess $\mathbf{A}^0 \in \mathbb{R}^{I\times R}$, $\mathbf{B}^0 \in \mathbb{R}^{J\times R}$, $\mathbf{C}^0 \in \mathbb{R}^{K\times R}$, λ_0

for $n = 1, \ldots, N$ **do**

$\mathbf{W} \leftarrow [(\mathbf{C}^n \odot_p \mathbf{B}^n); \lambda_n \mathbf{I}^{LR\times LR}] \in \mathbb{R}^{(JK+LR)\times LR}$

$\mathbf{S} \leftarrow [\mathbf{X}_{(1)}; \lambda_n (\mathbf{A}^n)^{\mathrm{T}}] \in \mathbb{R}^{(JK+LR)\times I}$

$\mathbf{A}^{n+1} \leftarrow \mathbf{W}/\mathbf{S}$ —— % solving least squares to update $\mathbf{A}$

$\mathbf{W} \leftarrow [(\mathbf{C}^n \odot_p \mathbf{A}^{n+1}); \lambda_n \mathbf{I}^{LR\times LR}] \in \mathbb{R}^{(IK+LR)\times LR}$

$\mathbf{S} \leftarrow [\mathbf{X}_{(2)}; \lambda_n (\mathbf{B}^n)^{\mathrm{T}}] \in \mathbb{R}^{(IK+LR)\times J}$

$\mathbf{A}^{n+1} \leftarrow \mathbf{W}/\mathbf{S}$ —— % solving least squares to update $\mathbf{B}$

$\mathbf{W} \leftarrow [((\mathbf{B}_1^{n+1} \odot \mathbf{A}_1^{n+1})\mathbf{1}_L, \ldots, (\mathbf{B}_R^{n+1} \odot \mathbf{A}_R^{n+1})\mathbf{1}_L); \lambda_n \mathbf{I}^{R\times R}] \in \mathbb{R}^{(IJ+R)\times R}$

$\mathbf{S} \leftarrow [\mathbf{X}_{(3)}; \lambda_n (\mathbf{C}^n)^{\mathrm{T}}] \in \mathbb{R}^{(IJ+R)\times K}$

$\mathbf{C}^{n+1} \leftarrow \mathbf{W}/\mathbf{S}$ —— % solving least squares to update $\mathbf{C}$

end for

return $\mathbf{A}^N$, $\mathbf{B}^N$, $\mathbf{C}^N$

end procedure

Table 6.6: **Regularized algorithm of BTD-$(L, L, 1)$ with rank R for a third-order tensor $\mathcal{X} \in \mathbb{R}^{I\times J\times K}$**

to make a geometrically decreasing λ_n for each iteration n. We generalize this method in [91] to make three different decreasing sequence $\{\alpha_n\}$, $\{\beta_n\}$, $\{\gamma_n\}$ for each factor respectively. The determination of the ratios $p = \alpha_n/\alpha_{n-1}$, $q = \beta_n/\beta_{n-1}$ and $r = \gamma_n/\gamma_{n-1}$ refers to the L-curve criterion [48]. In the next section, we will provide an example to compare our parameter choice with the original one.

5.5 Convergence results of BTD-RALS

We have already shown that the BTD-ALS is similar to the ALS using CP decomposition. Furthermore, the BTD-RALS is also a PGS method. Therefore, the convergence conclusion in Chap. 4 for the CP decomposition still holds for the BTD-$(L, L, 1)$ and

RBTD-(L, M, N)-Algorithm [91],

procedure RBTD-$(L, M, N)(\mathcal{X}, R, N, \lambda_n)$

give initial guess $\mathbf{A}^0 \in \mathbb{R}^{I\times R}$, $\mathbf{B}^0 \in \mathbb{R}^{J\times R}$, $\mathbf{C}^0 \in \mathbb{R}^{K\times R}$, $\mathcal{D}_r \in \mathbb{R}^{L\times M\times N}$ for $r = 1, 2, \ldots, R$

for $n = 1, \ldots, N$ **do**

$$\mathbf{W} \leftarrow [(\mathbf{C}^n \odot_p \mathbf{B}^n) \cdot (\mathbf{D}_{\mathbf{1}(1)}^n \otimes \mathbf{e}_1, \ldots, \mathbf{D}_{\mathbf{R}(1)}^n \otimes \mathbf{e}_R)^{\mathrm{T}}; \lambda_n \mathbf{I}^{LR\times LR}] \in \mathbb{R}^{(JK+LR)\times LR}$$

$$\mathbf{S} \leftarrow [\mathbf{X}_{(\mathbf{1})}; \lambda_n (\mathbf{A}^n)^{\mathrm{T}}] \in \mathbb{R}^{(JK+LR)\times LR}$$

$\mathbf{A}^{n+1} \leftarrow \mathbf{W}/\mathbf{S}$—— % solving least squares to update $\mathbf{A}$

$$\mathbf{W} \leftarrow [(\mathbf{C}^n \odot_p \mathbf{A}^{n+1}) \cdot (\mathbf{D}_{\mathbf{1}(2)}^n \otimes \mathbf{e}_1, \ldots, \mathbf{D}_{\mathbf{R}(2)}^n \otimes \mathbf{e}_R)^{\mathrm{T}}; \lambda_n \mathbf{I}^{JR\times JR}] \in \mathbb{R}^{(IK+JR)\times JR}$$

$$\mathbf{S} \leftarrow [\mathbf{X}_{(\mathbf{2})}; \lambda_n (\mathbf{B}^n)^{\mathrm{T}}] \in \mathbb{R}^{(IK+JR)\times JR}$$

$\mathbf{B}^{n+1} \leftarrow \mathbf{W}/\mathbf{S}$—— % solving least squares to update $\mathbf{B}$

$$\mathbf{W} \leftarrow [(\mathbf{B}^{n+1} \odot_p \mathbf{A}^{n+1}) \cdot (\mathbf{D}_{\mathbf{1}(3)}^n \otimes \mathbf{e}_1, \ldots, \mathbf{D}_{\mathbf{R}(3)}^n \otimes \mathbf{e}_R)^{\mathrm{T}}; \lambda_n \mathbf{I}^{KR\times KR}] \in \mathbb{R}^{(IJ+KR)\times KR}$$

$$\mathbf{S} \leftarrow [\mathbf{X}_{(\mathbf{3})}; \lambda_n (\mathbf{C}^n)^{\mathrm{T}}] \in \mathbb{R}^{(IJ+KR)\times KR}$$

$\mathbf{C}^{n+1} \leftarrow \mathbf{W}/\mathbf{S}$—— % solving least squares to update $\mathbf{C}$

$$\begin{bmatrix} vec(\mathcal{D}_1) \\ \vdots \\ vec(\mathcal{D}_R) \end{bmatrix} \leftarrow (\mathbf{A}^{n+1} \odot_p \mathbf{B}^{n+1} \odot_p \mathbf{C}^{n+1}) \backslash vec(\mathcal{T})$$

—— % solving least squares to update $\mathcal{D}_r$, $r = 1, \ldots, R$

end for

return $\mathbf{A}^N$, $\mathbf{B}^N$, $\mathbf{C}^N$, $\mathcal{D}_r^N$, $r = 1, \ldots, R$

end procedure

Table 6.7: **Regularized algorithm of BTD-(L, M, N) with rank R for a third-order tensor $\mathcal{X} \in \mathbb{R}^{I\times J\times K}$**

BTD-(L, M, N). So, we state the theorem as follows:

Theorem 5.1 (Convergence result for BTD-$(L, L, 1)$) *Suppose that the sequence* $\{(\mathbf{A}^k, \mathbf{B}^k, \mathbf{C}^k)\}$ *obtained from the BTD-RALS has limit points. Then every limit point* $(\overline{\mathbf{A}}, \overline{\mathbf{B}}, \overline{\mathbf{C}})$ *is a critical point of the Problem (4.2).*

Theorem 5.2 (Convergence result for BTD-(L, M, N)) *Suppose that the sequence* $\{(\mathbf{A}^k, \mathbf{B}^k, \mathbf{C}^k, \mathcal{D}_r^k)\}$ *obtained from the BTD-RALS has limit points. Then every limit point* $(\overline{\mathbf{A}}, \overline{\mathbf{B}}, \overline{\mathbf{C}}, \overline{\mathcal{D}}_r)$ *is a critical point of the Problem (4.5), where* $r = 1, 2, \ldots, R$.

6 Numerical Examples

Example 6.1 (Numerical example for BTD-(L, M, N)) We generate a tensor $\mathcal{X} \in \mathbb{R}^{15\times16\times30}$ by random data and consider a block term decomposition of $\mathcal{X}$ in rank-$(2, 2, 2)$ with $R = 2$. The factor matrices are $\mathbf{A} = [\mathbf{A}_1 \in \mathbb{R}^{15\times2}, \mathbf{A}_2 \in \mathbb{R}^{15\times2}] \in \mathbb{R}^{15\times4}$, $\mathbf{B} = [\mathbf{B}_1 \in \mathbb{R}^{16\times2}, \mathbf{B}_2 \in \mathbb{R}^{16\times2}] \in \mathbb{R}^{16\times4}$, $\mathbf{C} = [\mathbf{C}_1 \in \mathbb{R}^{30\times2}, \mathbf{C}_2 \in \mathbb{R}^{30\times2}] \in \mathbb{R}^{30\times4}$, and the two core tensors are $\mathcal{D}_1, \mathcal{D}_2 \in \mathbb{R}^{2\times2\times2}$.

$$\mathcal{X} = \mathcal{D}_1 \times_1 \mathbf{A}_1 \times_2 \mathbf{B}_1 \times_3 \mathbf{C}_1 + \mathcal{D}_2 \times_1 \mathbf{A}_2 \times_2 \mathbf{B}_2 \times_3 \mathbf{C}_2.$$

We use the same random initial guesses for both BTD algorithm and RBTD algorithm and let $\lambda_0 = 1$, $\lambda_n = 0.85 \cdot \lambda_{n-1}$, $n = 1, 2, \ldots$. The Figure 6.5 shows that the BTD-RALS algorithm just takes just less than 300 iterations to reach an error within 10^{-6} while the regular BTD takes 5041 iterations.

Example 6.2 (Numerical example for BTD-$(L, L, 1)$) For tensor $\mathcal{X} \in \mathbb{R}^{10\times15\times28}$, a block term decomposition of $\mathcal{X}$ in rank-$(3, 3, 1)$ with $R = 3$. The factor matrices are $\mathbf{A} \in \mathbb{R}^{10\times9}$, $\mathbf{B} \in \mathbb{R}^{15\times9}$ and $\mathbf{C} \in \mathbb{R}^{28\times3}$. The factorization equation is

$$\mathbf{X} = \sum_{r=1}^{3} (\mathbf{A}_r \mathbf{B}_r^{\mathrm{T}}) \circ \mathbf{c}_r.$$

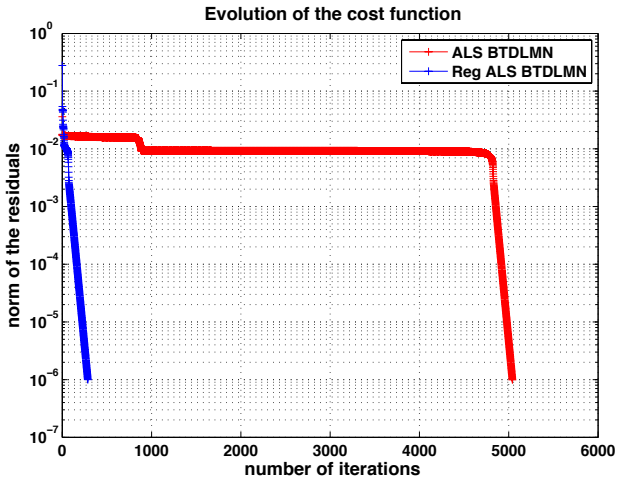


Figure 6.5: **The comparison of the BTD and RBTD with the same initials**

Figure 6.6 shows that the BTD-RALS algorithm just takes 1558 iterations to reach an error within 10^{-4}, however, the BTD-ALS algorithm does not decrease the error within 20,000 iterations.

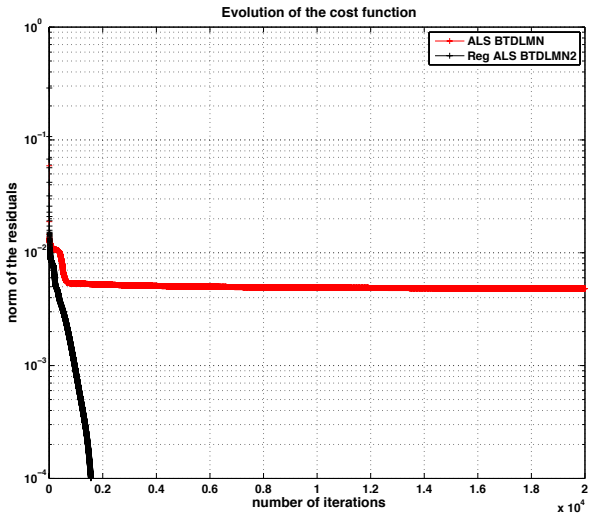


Figure 6.6: **The comparison of the BTD and RBTD with the same initials**

Example 6.3 (Comparison of different parameter choice) For this example, we also use a third-order tensor $\mathcal{X} \in \mathbb{R}^{15\times16\times30}$, and the block term decomposition is BTD-$(2, 2, 1)$ with $R = 2$. The parameter choice in [91] is $\lambda_n = 0.85\lambda_{n-1}$, so it makes

the same λ_n for three factors at the iteration n. Our parameter choices are let $p = \alpha_n/\alpha_{n-1} = 0.65$, $q = \beta_n/\beta_{n-1} = 0.75$ and $r = \gamma_n/\gamma_{n-1} = 0.85$. Observe that we make the changes on the regularization parameters for the first two factor.

Figure 6.7 shows the result. The swamp occurs for the BTD-ALS method (the red line). It shows that our parameter choice (blue line) is slightly better than the original one [91]. The CPU time of our method is 7.81s which is also faster than the original BTD-RALS, 8.64 (with the same parameter for every factors at the same iteration).

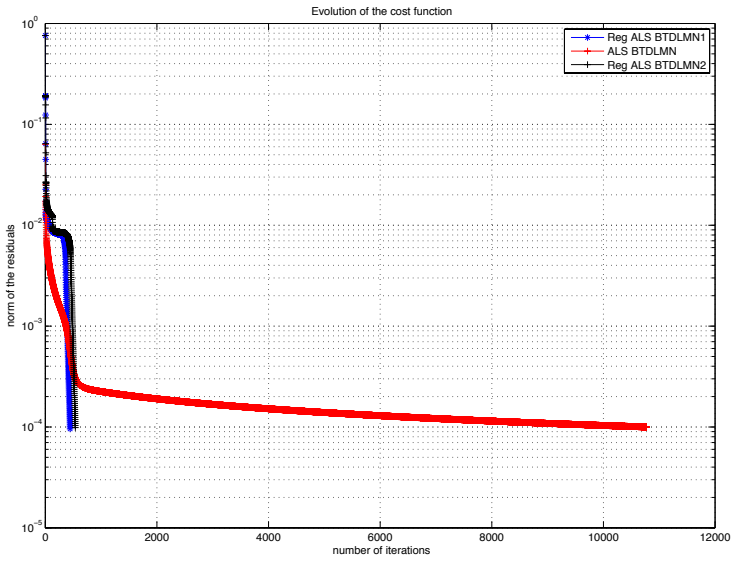


Figure 6.7: **The blue line is our parameter choice and the black line is the parameter choice in [91]**

7 Receptor model

In this section, we introduce the three-way receptor model [102], which is used in the study of source apportionment in chemometrics. We point out that the BTD-$(L, L, 1)$ can be considered as a special case of receptor model with an additional constraint. In addition, we prove that the receptor model does not have unique solution. In the

following computation, we use the matricization definition 1.8.

7.1 Receptor model

Definition 7.1 *[102] Tensor receptor model decomposes a third-order tensor* $\mathcal{X} \in \mathbb{R}^{I\times J\times K}$ *into the form*

$$\mathcal{X} = \sum_{r=1}^{R} \mathbf{A}^{(r)} \circ \mathbf{b}^{(r)}, \tag{7.1}$$

where $\mathbf{A}^{(r)} \in \mathbb{R}^{I\times J}$ *and* $\mathbf{b}^{(r)} \in \mathbb{R}^{K}$.

Figure 6.8 shows the receptor model of a third-order tensor. We let $\mathbf{A} = [\mathbf{A}^{(1)}\ \mathbf{A}^{(2)}\ \cdots\ \mathbf{A}^{(R)}]$

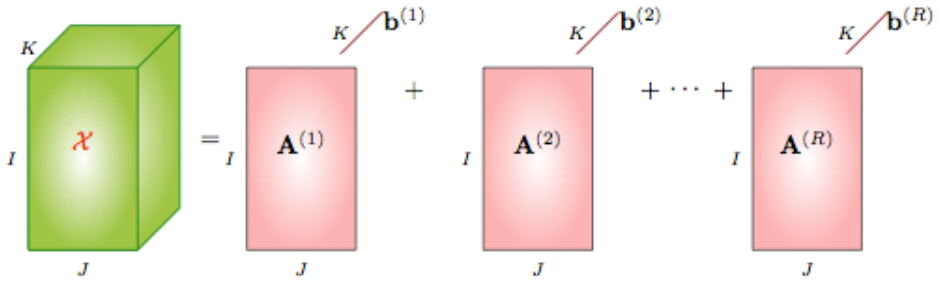


Figure 6.8: **Receptor model for** $\mathcal{X} \in \mathbb{R}^{I\times J\times K}$

and $\mathbf{B} = [\mathbf{b}^{(1)}\ \mathbf{b}^{(2)}\ \cdots\ \mathbf{b}^{(R)}]$ and call them the two factor matrices of the receptor model.

7.2 Receptor model and BTD-$(L, L, 1)$

Let us recall the definition of BTD-$(L, L, 1)$ (Definition 3.2), which decomposes a third-order tensor $\mathcal{X} \in \mathbb{R}^{I\times J\times K}$ into a sum of matrix-vector outer products and the rank of the matrices is L. The formula (see equation (3.2)) of the BTD-$(L, L, 1)$ is

$$\mathcal{X} = \sum_{r=1}^{R} \mathbf{E}_r \circ \mathbf{c}_r.$$

Therefore, comparing the equation (7.1) and (3.2), we can see that both decompose the tensor into a sum of matrix-vector outer products. The only difference is that BTD-$(L, L, 1)$ requires all the matrices $\mathbf{E}_r$ have the same rank L. So, we can see that the BTD-$(L, L, 1)$ is a special case of receptor model, and the solution of equation (3.2) (BTD-$(L, L, 1)$) is a solution of the receptor model (equation (7.1)).

We have shown that the BTD-$(L, L, 1)$ is rewritten as

$$\mathcal{X} = \sum_{r=1}^{R} (\mathbf{A}_r \mathbf{B}_r^{\mathrm{T}}) \circ \mathbf{c}_r$$

and can be solved by the alternative methods BTD-ALS and BTD-RALS. Therefore, one solution of the receptor model (7.1) is

$$\mathbf{A}^{(r)} = \mathbf{A}_r \mathbf{B}_r^{\mathrm{T}}, \quad \mathbf{b}^{(r)} = \mathbf{c}_r. \tag{7.2}$$

Now, we can show a direct way to solve the receptor model (7.1).

Given the third-order tensor $\mathcal{X} \in \mathbb{R}^{I \times J \times K}$, we matricize it to be a matrix $\mathbf{X} \in \mathbb{R}^{IJ \times K}$ as follows

$$\mathbf{X}(:, k) = vec(\mathbf{X}_{:\,:\,\mathbf{k}}), \tag{7.3}$$

which means that each column of the resulting matrix $\mathbf{X}$ is the vectorization of the corresponding slice $\mathbf{X}_{:\,:\,\mathbf{k}}$. Then we calculate the SVD of $\mathbf{X}$

$$\mathbf{X} = \sum_{r=1}^{R} \sigma_r \mathbf{u}_r \circ \mathbf{v}_r = \sum_{r=1}^{R} (\sigma_r \mathbf{u}_r) \circ \mathbf{v}_r = \sum_{r=1}^{R} \mathbf{w}_r \circ \mathbf{v}_r,$$

where $\mathbf{w}_r \in \mathbb{R}^{IJ}$ and $\mathbf{v}_r \in \mathbb{R}^{K}$. For each vector $\mathbf{w}_r$, we can matricize it to be a matrix $\mathbf{W}_r \in \mathbb{R}^{I \times J}$ by defining

$$\mathbf{W}_r(i, j) = \mathbf{w}_r((j-1)I + i).$$

Therefore, the solution of the receptor model (7.1) is

$$\mathbf{A}^{(r)} = \mathbf{W}_r, \quad \mathbf{b}^{(r)} = \mathbf{v}_r. \tag{7.4}$$

The reason why we can use the SVD to obtain the solution (7.4) is that when we using the matricization defined in (7.3), we have

$$\begin{aligned} \mathcal{X} &= \sum_{r=1}^{R} \mathbf{A}^{(r)} \circ \mathbf{b}^{(r)} \\ &\Downarrow \\ \mathbf{X} &= \sum_{r=1}^{R} vec(\mathbf{A}^{(r)}) \circ \mathbf{b}^{(r)}. \end{aligned} \tag{7.5}$$

So we can compute the matrices $\mathbf{A}^{(r)}$ and vectors $\mathbf{b}^{(r)}$ so that they can satisfy the equation (7.5). Notice that they can be obtained by easily computing the SVD on the matrix $\mathbf{X}$.

7.3 Non-uniqueness of receptor model

We have shown that the receptor model at least has two solutions. One is the direct method by computing the SVD of the matricization of $\mathcal{X}$ and is shown in (7.4). And another one is the solution of BTD-$(L, L, 1)$ (7.2). So it seems that the receptor model does not have unique solution. In this section, we will mathematically show that the receptor model equation (7.1) does not have unique solution.

Theorem 7.2 *For a given third-order tensor $\mathcal{X} \in \mathbb{R}^{I \times J \times K}$ and a fixed number R, there are multiple possible solution sets $\{\mathbf{A}^{(r)}, \mathbf{b}^{(r)}\}$ can satisfy the equation (7.1). In addition, there always exists a rotation matrix between two solutions.*

Proof. For the equation (7.1), we have the block matrix $\mathbf{A}$ and factor matrix $\mathbf{B}$ as

follows,

$$\begin{aligned}\mathbf{A} &= [\mathbf{A}^{(1)}\ \mathbf{A}^{(2)}\ \cdots\ \mathbf{A}^{(R)}] \in \mathbb{R}^{I\times JR},\\ \mathbf{B} &= [\mathbf{b}^{(1)}\ \mathbf{b}^{(2)}\ \cdots\ \mathbf{b}^{(R)}] \in \mathbb{R}^{K\times R}.\end{aligned}$$

So for such a block matrix $\mathbf{A}$, we define a matrix $M(\mathbf{A})$ by reordering the elements of $\mathbf{A}$ as following:

$$M(\mathbf{A}) = [vec((\mathbf{A}^{(1)})^{\mathrm{T}})\ vec((\mathbf{A}^{(2)})^{\mathrm{T}})\ \cdots\ vec((\mathbf{A}^{(R)})^{\mathrm{T}})] \in \mathbb{R}^{IJ\times R}. \tag{7.6}$$

The reverse of M is denoted M^{-1}, which implies that $M^{-1}(M(\mathbf{A})) = \mathbf{A}$ and $M(M^{-1}(\mathbf{A})) = \mathbf{A}$.

Therefore, by taking mode-3 matricizaiton on equation (7.1) so that it can be rewritten as follows:

$$\begin{aligned}\mathcal{X} &= \sum_{r=1}^{R} \mathbf{A}^{(r)} \circ \mathbf{b}^{(r)}\\ &\Updownarrow\\ \mathbf{X}_{(3)}^{\mathrm{T}} &= M(\mathbf{A}) \cdot \mathbf{B}^{\mathrm{T}},\end{aligned} \tag{7.7}$$

where $\mathbf{X}_{(3)} \in \mathbb{R}^{K\times IJ}$ is the mode-3 matricization of the given tensor $\mathcal{X}$. $M(\mathbf{A})$ is defined by (7.6).

Hence, given a third-order tensor $\mathcal{X}$ and a fixed number R, the problem that finding matrices $\mathbf{A}^{(r)}$ and vectors $\mathbf{b}^{(r)}$, $r = 1, \ldots, R$ to satisfy the equation (7.1) is equivalent to finding matrices $\mathbf{A}$ and $\mathbf{B}$ to satisfy the equation (7.7).

Let us look at the equation (7.7), if $\mathbf{A}$ and $\mathbf{B}$ are the solutions of equation (7.7), then there exists an invertible matrix $\mathbf{T} \in \mathbb{R}^{R\times R}$ such that the matrices

$$\widetilde{\mathbf{A}} = M^{-1}(M(\mathbf{A}) \cdot \mathbf{T}), \quad \widetilde{\mathbf{B}} = \mathbf{B} \cdot \mathbf{T}^{-\mathrm{T}}$$

are also the solutions of (7.7). That is because

$$\begin{aligned} M(\widetilde{\mathbf{A}}) \cdot \widetilde{\mathbf{B}}^{\mathrm{T}} &= M(M^{-1}(M(\mathbf{A}) \cdot \mathbf{T})) \cdot (\mathbf{B} \cdot \mathbf{T}^{-\mathrm{T}})^{\mathrm{T}} \\ &= M(\mathbf{A}) \cdot \mathbf{T} \cdot \mathbf{T}^{-1} \cdot \mathbf{B}^{\mathrm{T}} \\ &= M(\mathbf{A}) \cdot \mathbf{B}^{\mathrm{T}} \\ &= \mathbf{X}_{(3)}^{\mathrm{T}}. \end{aligned}$$

And we call the $\mathbf{T}$ rotation matrix between the two solution sets $\{\mathbf{A}, \mathbf{B}\}$ and $\{\widetilde{\mathbf{A}}, \widetilde{\mathbf{B}}\}$.

□

Therefore, in the application of receptor model, some constraints are needed to help us find the solutions we expect. We will study this deeply in chapter 7 in the chemometrics application.

Chapter 7

Application I: Source apportionment of time and size resolved ambient particulate matter

In this chapter, we provide an application of the tensor decomposition. We have talked about the CP decomposition, HOSVD decomposition and two types of Block Term Decomposition. Each decomposition was proposed due to the requirement in the fields of application like psychometrics, chemometrics and signal processing. According to Kolda [70], Appellof and Davidson [4] are generally credited as being the first to use tensor decompositions in chemometrics, and tensors have since become extremely popular in that field. So we will study one application in the field of chemometrics, which applies the receptor model (see Chap. 6 [102]) on the environmental data [80].

1 Introduction

Airport emissions are studied with regard to the local air quality in nearby area of an airport for years (see [101, 18, 127]). In order to reduce the exposure of pollutants which emitted from airport operations, it is necessary to quantify the various airport sources (ground vehicles, landings, etc.) in order to develop a reliable emissions inventory.

Receptor modeling is the application of data analysis methods to elicit information on the source of air pollutants. The fundamental principle of receptor modeling is that mass conservation can be assumed and a mass balance analysis can be used to identify and apportion sources of airborne particulate matter (PM) in the atmosphere. Initially, the bilinear multivariate receptor models are used to study a two-dimensional data sample which involves the time-resolved PM and chemical species. Recently, according to a study [40], particle size is also a significant factor in the study of airport emissions. Therefore, the DRUM receptor model [102] is used to study the size- and time-resolved PM samples in this chapter. It can take the size-composition variation into account to properly resolve the ambient data for the apportionment of potential airport emission sources. A weighted alternating least-squares method is introduced to solve this model and five emission sources are identified successfully.

2 Data description

The original size- and time-resolved aerosol samples were collected using eight-stage rotating DRUM impactor samplers at Washington-Dulles International Airport. There measurement campaigns were conducted during 3 different seasons (i) April 17-28, 2009; (ii) January 16-24, 2010; and (iii) July 9-23, 2010. During April, 2009, samples were collected by deploying one Rotating Drum Impactors (RDI) at the Base Station. In the winter and summer seasonal campaigns, two RDIs were deployed; one at the

Fire Station and the other at the Stone House (New Base Station) sites [67].

Particulate matter samples were analyzed by synchrotron X-ray Fluorescence (s-XRF) [68] using a broad-spectrum X-ray beam generated on beamline 10.3.1 at the Advanced Light Source Lawrence Berkeley National Laboratory. The s-XRF analysis provides quantitative elemental data for 27 elements (Mg, Al, Si, P, S, Cl, K, Ca, Ti, V, Cr, Mn, Fe, Co, Ni, Cu, Zn, Ga, As, Se, Br, Rb, Sr, Y, Zr, Mo, and Pb) in 8 size modes (0.1-0.26μm, 0.26-0.34μm, 0.34-0.56μm, 0.56-0.75μm, 0.75-1.15μm, 1.15-2.5μm, 2.5-5μm and 5-10μm) with 3-hour time resolution on the samples collected in this campaign, and the total time samples are 357 during the spring, summer and winter. In addition, mass concentrations were measured using soft beta attenuation. The summary statistics are presented in Table 7.1.

The data were considered as a function of size, time, and chemical composition (i.e., elemental species), which can be organized a third-order tensor $\mathcal{X}_{orig} \in \mathbb{R}^{I \times J \times K}$. If i denotes the chemical species, j particle size, and k the time sample, then a datum point, x_{ijk}, can be expressed as the concentration value of the ith chemical species of the jth particle size of the kth time sample.

There are two problems that need to be addressed before the sample tensor is studied. First, the synchrotron XRF does not provide carbon and nitrate values, so the measured mass minus the reconstructed mass, termed the "unmeasured mass", is introduced in the analysis. Another issue is the influence of high-noise variables (chemical species). For some variables, the data may consist almost entirely of noise which would increase the errors in computed factors. The question of accepting or rejecting individual chemical constituents has been studied by Paatero and Hopke [97]. The signal-to-noise ratio (S/N) and below detection level (BDL) were introduced to determine the noisy variables (containing much more noise than signal). For uncensored data, a variable is defined to be bad if S/N< 0.2. For censored data, a sufficiently large numbers of BDL values ($> 80\%$) may also indicate a noisy variable.

Element	Mean (ng/m^3)	Standard Deviation (ng/m^3)	Median (ng/m^3)	S/N[a]	Number of DBL values[b]
Mg	61.10	270.69	32.14	2.6503	53
Al	37.02	85.52	12.82	1.7189	6
Si	54.44	113.84	10.00	3.2904	5
P	7.65	29.77	4.75	0.7456	6
S	113.26	721.72	31.21	5.0237	2
Cl	5.46	22.14	0	0.9685	30
K	7.47	11.49	3.29	0.9954	9
Ca	30.64	61.97	3.30	1.0085	0
Ti	2.99	5.37	0.90	0.9978	0
V	0.11	0.22	0.05	0.9960	247
Cr	0.05	0.10	0.02	1.0000	402
Mn	0.46	0.87	0.17	0.9998	85
Fe	23.48	41.48	5.84	1.0004	0
Co	0.07	0.10	0.04	0.9981	497
Ni	0.12	0.33	0.06	1.0000	84
Cu	0.99	1.59	0.36	0.9996	0
Zn	1.69	1.80	1.14	0.9979	0
Ga	0.03	0.04	0.02	0.9994	572
As	0.12	0.26	0.03	0.9998	237
Se	0.22	0.35	0.14	0.9995	299
Br	2.00	0.83	1.81	0.9987	0
Rb	0.21	0.22	0.17	0.9999	482
Sr	0.48	0.24	0.43	0.9999	448
Y	0.49	0.43	0.38	0.9999	603
Zr	0.99	0.66	0.78	0.9997	537
Mo	2.32	1.10	2.07	0.9995	489
Pb	1.06	3.10	0.43	0.9995	331

Table 7.1: **The summary statistics of the original data.**
a. Signal to noise ratio. b. Number of values below the method detection limit.

Therefore, four chemical species (P, Ga, Y, Zr) were eliminated on the basis of S/N and large numbers of BDL values, so that the value of I index of the tensor data we are using in the analysis is 24. It includes 23 chemical elements and the unmeasured mass. Consequently, the dimension of the tensor $\mathcal{X}$ is $24 \times 8 \times 357$.

3 Weighted alternating least-squares algorithm

The model we use here to study the tensor sample is the three-way receptor model which decomposes the tensor into R component tensors, each of them is the outer product of a matrix and a vector. So according to [102], each matrix of the component tensor is the profile for a given source. The dimensions are the number of measured variables and the number of measured size fractions. For each source (factor), the vector (corresponding to the matrix) is the mass contributions in terms of time, so the dimension is the number of time samples. And R denotes the number of independent sources (factors).

The receptor model is introduced in Chap. 6, the tensor is factored according to it so we can have

$$\mathcal{X} = \sum_{r=1}^{R} \mathbf{A}^{(r)} \circ \mathbf{b}^{(r)} + \mathcal{E}, \tag{3.1}$$

where $\mathcal{X}$ is the third-order tensor of observed data, $\mathbf{A}^{(r)}$ is the rth source profile array and $\mathbf{b}^{(r)}$ is the corresponding rth contribution vector. The tensor $\mathcal{E}$ having the same size as $\mathcal{X}$ contains the residuals.

In its component form, the model equation becomes:

$$x_{ijk} = \sum_{r=1}^{R} a_{ij}^{(r)} b_{k}^{(r)} + e_{ijk}, \tag{3.2}$$

where $a_{ij}^{(r)}$ is the ith species mass fraction of the jth particle size range from the rth source, $b_{k}^{(r)}$ is the rth source mass contribution during the time units for the kth sample, and e_{ijk} is the residual associated with ith species concentration measured in the kth sample of the jth size range, and R is the total number of independent sources.

As we have already shown that the receptor model does not have unique solu-

tions, which means that if one set of factor matrices and vectors $\{\mathbf{A}^{(r)}, \mathbf{b}^{(r)}\}$ satisfy the model, then we can also have another set of solutions $\{\widetilde{\mathbf{A}}^{(r)}, \widetilde{\mathbf{b}}^{(r)}\}$. The relation between the two sets of solutions is discussed in Chap. 6. Therefore, in order to decrease the rotational freedom in the solution, we introduce the uncertainty estimation which provides a useful tool to decrease the weight of missing and below detection limit data in the solution. The procedures of Polissar et al. [103] are used to assign measured data and the associated uncertainties as the input data. In addition, non-negativity constraints are also applied to the factors.

So from this receptor model and the techniques of controlling rotation, our problem is to find non-negative matrices $\mathbf{A}^{(r)}$ and vectors $\mathbf{b}^{(r)}$, for $r = 1, \ldots, R$, to minimize the following objective function:

$$Q = \sum_{i=1}^{I} \sum_{j=1}^{J} \sum_{k=1}^{K} \frac{\left(x_{ijk} - \sum_{r=1}^{R} a_{ij}^{(r)} b_k^{(r)}\right)^2}{u_{ijk}^2}, \tag{3.3}$$

where u_{ijk} is the uncertainty value associated with data value x_{ijk}.

For convenience, the uncertainties u_{ijk} are organized into a third-order tensor $\mathcal{U}$ and the size is same as the input tensor $\mathcal{X}$. The objective function (3.3) can then be written as:

$$Q = \left\| \frac{\left(\mathcal{X} - \sum_{r=1}^{R} \mathbf{A}^{(r)} \circ \mathbf{b}^{(r)}\right)}{\mathcal{U}} \right\|_F^2, \tag{3.4}$$

where the division between the two tensors is element-wise division. So the optimization problem is

$$\begin{aligned} \underset{\mathbf{A},\mathbf{B}}{\text{minimize}} \quad & \left\| \frac{\left(\mathcal{X} - \sum_{r=1}^{R} \mathbf{A}^{(r)} \circ \mathbf{b}^{(r)}\right)}{\mathcal{U}} \right\|_F^2, \\ \text{subject to} \quad & \mathbf{A}, \mathbf{B} \text{ are non-negative} \end{aligned} \tag{3.5}$$

where $\mathbf{A} = [\mathbf{A}^{(1)}\ \mathbf{A}^{(2)}\ \cdots\ \mathbf{A}^{(R)}]$ and $\mathbf{B} = [\mathbf{b}^{(1)}\ \mathbf{b}^{(2)}\ \cdots\ \mathbf{b}^{(R)}]$.

In order to solve the problem (3.5), let us study the unweighted receptor model first. By the mode-1 and mode-3 matricization (see Definition 1.8 in Chap. 2), we can have the following forms,

$$\begin{aligned} \mathcal{X} &= \sum_{r=1}^{R} \mathbf{A}^{(r)} \circ \mathbf{b}^{(r)} \\ &\Downarrow \\ \mathbf{X}_{(1)}^{\mathrm{T}} &= (\mathbf{D} \odot_l \mathbf{B}) \cdot \mathbf{A}^{\mathrm{T}} \\ \mathbf{X}_{(3)}^{\mathrm{T}} &= M(\mathbf{A}) \cdot \mathbf{B}^{\mathrm{T}}. \end{aligned}$$

Where $\mathbf{D}$ is a $J \times J$ identity matrix, $M(\mathbf{A})$ is defined by the equation (7.6) in Chap. 6, which is

$$M(\mathbf{A}) = [vec((\mathbf{A}^{(1)})^{\mathrm{T}})\ vec((\mathbf{A}^{(2)})^{\mathrm{T}})\ \cdots\ vec((\mathbf{A}^{(R)})^{\mathrm{T}})] \in \mathbb{R}^{IJ \times R}.$$

The product '$\odot_l$' is defined as following

$$\begin{aligned} \mathbf{D} \odot_l \mathbf{B} &= \mathbf{D} \odot_l [\mathbf{b}^{(1)}\ \mathbf{b}^{(2)}\ \cdots\ \mathbf{b}^{(R)}] \\ &= [\mathbf{D} \otimes \mathbf{b}^{(1)}\ \mathbf{D} \otimes \mathbf{b}^{(2)}\ \cdots\ \mathbf{D} \otimes \mathbf{b}^{(R)}] \in \mathbb{R}^{JK \times JR}. \end{aligned} \tag{3.6}$$

These two variations are equivalent, therefore, we can matricize the uncertainty tensor correspondingly so that the objective function (3.4) can have the following two variations:

$$Q1 = \left\| (\mathbf{X}_{(3)}^{\mathrm{T}} - M(\mathbf{A}) \cdot \mathbf{B}^{\mathrm{T}}) / \mathbf{U}_{(3)}^{\mathrm{T}} \right\|_F^2, \tag{3.7}$$

$$Q2 = \left\| (\mathbf{X}_{(1)}^{\mathrm{T}} - (\mathbf{D} \odot_l \mathbf{B}) \cdot \mathbf{A}^{\mathrm{T}}) / \mathbf{U}_{(1)}^{\mathrm{T}} \right\|_F^2, \tag{3.8}$$

where $\mathbf{U}_{(1)}^{T}$ and $\mathbf{U}_{(3)}^{T}$ are the mode-1 and mode-3 matricizations of $\mathcal{U}$, and '/' denotes

element-wise division between two matrices.

We use the same technique as ALS that fixing the factor matrix $\mathbf{B}$ to solve for $\mathbf{A}$ in (3.7), and then fixing the factor matrix $\mathbf{A}$ to solve for $\mathbf{B}$ in (3.8), the problem (3.5) reduces to two coupled weighted least-squares subproblems. Thus, we can solve these two subproblems alternatively until some convergence criterion is satisfied:

$$\mathbf{A}^{k+1} = \underset{\widehat{\mathbf{A}}\in\mathbb{R}^{I\times JR}}{\operatorname{argmin}} \left\|(\mathbf{X}_{(1)}^{\mathrm{T}} - (\mathbf{D}\odot_l \mathbf{B}^k)\cdot\widehat{\mathbf{A}}^{\mathrm{T}})/\mathbf{U}_{(1)}^{\mathrm{T}}\right\|_F^2, \tag{3.9}$$

$$\mathbf{B}^{k+1} = \underset{\widehat{\mathbf{B}}\in\mathbb{R}^{K\times R}}{\operatorname{argmin}} \left\|(\mathbf{X}_{(3)}^{\mathrm{T}} - M(\mathbf{A}^{k+1})\cdot\widehat{\mathbf{B}}^{\mathrm{T}})/\mathbf{U}_{(3)}^{\mathrm{T}}\right\|_F^2, \tag{3.10}$$

where $\mathbf{A}^{k+1}$ and $\mathbf{B}^{k+1}$ are the results obtained at the $(k+1)$th iteration.

This method is called ***Weighted Alternating Least-Squares*** (WALS) and has been used in [126] by Wentzell et al. The problem in [126] is to minimize the following cost function

$$O = \|(\mathbf{X} - \mathbf{CP})/\mathbf{\Sigma}\|_F^2,$$

where $\mathbf{X}, \mathbf{\Sigma} \in \mathbb{R}^{I\times J}$, $\mathbf{C} \in \mathbb{R}^{I\times R}$ and $\mathbf{P} \in \mathbb{R}^{R\times J}$. So the WALS algorithm proposed in [126] is

$$\begin{aligned}\mathbf{C}^{k+1} &= \underset{\widehat{\mathbf{C}}\in\mathbb{R}^{I\times R}}{\operatorname{argmin}} \left\|(\mathbf{X} - \widehat{\mathbf{C}}\mathbf{P}^k)/\mathbf{\Sigma}\right\|_F^2,\\ \mathbf{P}^{k+1} &= \underset{\widehat{\mathbf{P}}\in\mathbb{R}^{R\times J}}{\operatorname{argmin}} \left\|(\mathbf{X} - \mathbf{C}^{k+1}\widehat{\mathbf{P}})/\mathbf{\Sigma}\right\|_F^2.\end{aligned}$$

As we can see that the WALS method in [126] solves for the two factor matrices by using the same objective function. Alternatively we are using two objective functions $Q1$ and $Q2$. The reasoning behind this is that our least squares is more complicated. The factor matrix $\mathbf{A}$ cannot easily be computed by $Q1$, and similarly $\mathbf{B}$ cannot be

computed by $Q2$ directly.

The problem (3.9) and (3.10) with non-negativity constraints can be solved column by column. We take subproblem (3.9) to demonstrate this process. Let $\mathbf{x}_i$ and $\mathbf{u}_i$ denote the ith columns of matrix $\mathbf{X}_{(1)}^{\mathrm{T}}$ and $\mathbf{U}_{(1)}^{\mathrm{T}}$ respectively, $\widehat{\mathbf{a}}_i$ denote the ith column of $\widehat{\mathbf{A}}^{\mathrm{T}}$. So, we have

$$\left\|(\mathbf{X}_{(1)}^{\mathrm{T}} - (\mathbf{D} \odot_l \mathbf{B}^k) \cdot \widehat{\mathbf{A}}^{\mathrm{T}})/\mathbf{U}_{(1)}^{\mathrm{T}}\right\|_F^2 = \sum_{i=1}^{I} \left\|(\mathbf{x}_i - (\mathbf{D} \odot_l \mathbf{B}^k) \cdot \widehat{\mathbf{a}}_i)/\mathbf{u}_i\right\|_2^2. \tag{3.11}$$

For each $\mathbf{u}_i$, $i = 1, 2, \ldots, I$, we define a matrix $\mathbf{Du}_i$ as

$$\mathbf{Du}_i = \begin{bmatrix} 1/(\mathbf{u}_i)_1 & & & \\ & 1/(\mathbf{u}_i)_1 & & \\ & & \ddots & \\ & & & 1/(\mathbf{u}_i)_{JK} \end{bmatrix} \in \mathbb{R}^{JK \times JK}, \tag{3.12}$$

where $(\mathbf{u}_i)_s$ is the sth element of the vector $\mathbf{u}_i$, $s = 1, 2, \ldots, JK$.

Therefore, for the each item in the summation (3.11), we have

$$\begin{aligned} & \left\|(\mathbf{x}_i - (\mathbf{D} \odot_l \mathbf{B}^k) \cdot \widehat{\mathbf{a}}_i)/\mathbf{u}_i\right\|_2^2 \\ = \; & (\mathbf{x}_i - (\mathbf{D} \odot_l \mathbf{B}^k) \cdot \widehat{\mathbf{a}}_i)^{\mathrm{T}} \cdot \mathbf{Du}_i^2 \cdot (\mathbf{x}_i - (\mathbf{D} \odot_l \mathbf{B}^k) \cdot \widehat{\mathbf{a}}_i) \\ = \; & (\mathbf{x}_i - (\mathbf{D} \odot_l \mathbf{B}^k) \cdot \widehat{\mathbf{a}}_i)^{\mathrm{T}} \cdot \mathbf{Du}_i^{\mathrm{T}} \cdot \mathbf{Du}_i \cdot (\mathbf{x}_i - (\mathbf{D} \odot_l \mathbf{B}^k) \cdot \widehat{\mathbf{a}}_i) \\ = \; & (\mathbf{Du}_i(\mathbf{x}_i - (\mathbf{D} \odot_l \mathbf{B}^k) \cdot \widehat{\mathbf{a}}_i))^{\mathrm{T}} \cdot (\mathbf{Du}_i(\mathbf{x}_i - (\mathbf{D} \odot_l \mathbf{B}^k) \cdot \widehat{\mathbf{a}}_i)) \\ = \; & \left\|\mathbf{Du}_i \cdot \mathbf{x}_i - \mathbf{Du}_i(\mathbf{D} \odot_l \mathbf{B}^k) \cdot \widehat{\mathbf{a}}_i\right\|_2^2. \end{aligned} \tag{3.13}$$

Noticed that the equation (3.13) is just a least-squares problem with the coefficient matrix $\mathbf{Du}_i(\mathbf{D} \odot_l \mathbf{B}^k)$, where $\mathbf{Du}_i$ is defined by (3.12). So, instead of computing the whole matrix $\widehat{\mathbf{A}}^{\mathrm{T}}$ once a time, we can calculate each column $\widehat{\mathbf{a}}_i$ by (3.13). In addition, minimizing the objective function (3.13) in terms of $\widehat{\mathbf{a}}_i$ with non-negativity constraint

is easy to solve in MATLAB. We will use standard MATLAB least squares function, 'lsqnonneg', to impose the non-negativity constraints.

To stop the algorithm, we need to provide a convergence criterion. Usually the sum of squared residual (SSR) is used as convergence criterion. The criterion stops the algorithm if the change of objective Q between two iterations is less than some small number called the tolerance. However, this may not be the best convergence criterion to cope with our problem. In [45], a new convergence criterion is introduced. It keeps tracking the change in $\det(\mathbf{B}^{\mathrm{T}}\mathbf{B})$ from one iteration to the next, where 'det' is the determinant of the given matrix and the factor matrix $\mathbf{B}$ is normalized in each iteration. $\det(\mathbf{B}^{\mathrm{T}}\mathbf{B})$ is the squared volume of the space spanned by the column space of $\mathbf{B}$, it has advantages in the resolution process [45].

Therefore, the algorithm will stop if both of the changes (the value of objective function Q and $\det(\mathbf{B}^{\mathrm{T}}\mathbf{B})$) between two iterations are sufficiently small ($\sim 10^{-8}$).

We summarize the algorithm in Table 7.2:

About reducing the rotational freedom, we need to make the following comments.

1. According to Paatero et al. [98], the uncertainty estimates and non-negativity constraints are generally insufficient to wholly eliminate the rotational problem.

2. Several methods are proposed by Paatero et al. [98] to control the rotations. One way is constraining individual factor elements, either scores and/or loading, towards zero values based on some external information about acceptable or desirable shapes of the factors. Therefore, in our analysis and computation, constraints based on the a priori information are imposed. Based on an initial analysis [67], one factor is dominated by large particles with high concentrations of chlorine. This factor should only be contributing during the January sampling campaign and can be associated with the use of salt and sand on snow and ice. Therefore, the mass contribution vector $\mathbf{b}^{(r)}$ for this factor is constrained to zero for the summer and spring samples.

WALS-Algorithm

Input: Tensors $\mathcal{X}$,$\mathcal{U} \in \mathbb{R}^{I\times J\times K}$, factor numbers R.

Output: Nonegative matrices $\mathbf{A} \in \mathbb{R}^{I\times JR}$, $\mathbf{B} \in \mathbb{R}^{K\times R}$ minimize (3.7) and (3.8).

$\mathbf{A}^0 \in \mathbb{R}^{I\times JR}$, $\mathbf{B}^0 \in \mathbb{R}^{K\times R}$, $k = 0$

$\mathbf{U1} = \mathbf{U}_{(1)}^T$, $\mathbf{U3} = \mathbf{U}_{(3)}^T$, $\mathbf{X1} = \mathbf{X}_{(1)}^T$, $\mathbf{X3} = \mathbf{X}_{(3)}^T$, $\mathbf{D} = \text{eye}(J)$

$CC1 = 1$, $CC2 = 1$

while $CC1 > tol$ and $CC2 > tol$ **do**

  for $i = 1, \ldots, I$ **do**

    $\mathbf{Du} = \text{diag}(1./\mathbf{U1}(:, i))$

    $\mathbf{A}^{k+1}(i, :)^T = \text{lsqnonneg}\ (\mathbf{Du} \cdot (\mathbf{D} \odot_l \mathbf{B}^k), \mathbf{Du} \cdot \mathbf{X1}(:, i)))$

  end for

  for $j = 1, \ldots, K$ **do**

    $\mathbf{Du} = \text{diag}(1./\mathbf{U3}(:, i))$

    $\mathbf{B}^{k+1}(j, :)^T = \text{lsqnonneg}\ (\mathbf{Du} \cdot M(\mathbf{A}^{k+1}), \mathbf{Du} \cdot \mathbf{X3}(:, j))$

  end for

  $CC1 = \left\|(\mathbf{X3} - M(\mathbf{A}^{k+1})\mathbf{B}^{k+1})./\mathbf{U3}\right\|_F^2$

  $CC2 = \left|\det((\mathbf{B}^k)^T\mathbf{B}^k) - ((\mathbf{B}^{k+1})^T\mathbf{B}^{k+1})\right|$

  $k = k + 1$

end while

$\mathbf{A} = \mathbf{A}^k, \mathbf{B} = \mathbf{B}^k$

Table 7.2: **WALS algorithm for receptor model in the source apportionment application**

3. For each chemical element, the mass fraction values across the eight particle size ranges should be relatively smooth, which means that there should not be very low or zero concentrations in an intermediate particle size while both its adjacent particle sizes (smaller one and larger one) have high concentrations. Thus, a constraint on the size mode is imposed to make sure the change across the size mode is smooth.

4. The resulting apportionments are only good to a scale constant so the results

are normalized by regressing the apportioned masses for each source for each sampling period to the total measured mass as per Hopke et al. [58].

4 Numerical Results and Interpretation

We apply the WALS to the sample tensor $\mathcal{X} \in \mathbb{R}^{24\times8\times357}$ to calculate the source profile matrices $\mathbf{A}^{(r)}$ and the corresponding contribution vectors $\mathbf{b}^{(r)}$. Two criteria were used to assess the number of factors. The fits to the data are examined by reviewing the distributions of scaled residuals. These distributions should be symmetric and the values should generally range from -3 to $+3$. In addition, the profiles have to be physically realistic. The interpretability of the factors includes appropriate behavior across the particle size dimension since there should be a degree of smoothness in that direction. The pattern of elements and their appearance in physically meaningful size ranges were used to access the appropriateness of the various solutions. In our experiment, five factors are ultimately chosen to adequately reproduce the data and provide interpretable factors.

Figure 7.1 shows the profiles for each factor (source) for all three sampling campaigns. They are shown as grouped bar plots so that the size variation of the chemical species in each source profile can be observed. The time series of the source contributions are shown in Figure 7.2. The average mass contributions of each source for each season to particulate matter less than 10 μm in aerodynamic diameter (PM_{10}) are presented in Table 7.3.

The first factor shows high concentrations of crustal elements (Al, Si, Fe, Ca, Ti) peaking in the two largest size ranges. This factor can therefore be attributed to "soil". There can be some wind-driven aerosolization of surface solis, but more of the soil is probably re-suspended by various forms of traffic including the cars bringing passengers to the airport, ground activities at the airport, along with the taxing, take-off, and landings of the aircraft.

The second factor is dominated by large particles with high concentrations of chlorine, calcium, and magnesium with some iron and other crustal species. This factor only had contributions only during the January sampling campaign and can be associated with the use of salt and sand on snow and ice. This factor was only seen during the January campaign and we constrained the other values to be zero in the final model.

Factor 3 shows a very different pattern with small particle sulfur, zinc, bromine, zirconium and molybdenum. This factor is assigned to particles that are emitted during landings. The sulfur and zinc come from tire wear. These elements are key constituents in tires. Often a visible puff of smoke is observed at touchdown. There is considerable frictional heat produced at this instant and particles are generated across the particle size range. Both zirconium and molybdenum are used in high temperature greases as might be used to lubricated bearings that would undergo significant heat stress. The energy deposited in the bearings can be expected to liberate particles from the lubricants.

Factors 4 and 5 have the highest values of S, but in different size fractions. The chemical element S in factor 4 peaks in the middle size ranges. Such sizes are indicative of cloud processed sulfur and a similar factor was observed by Pere-Trepat et al. [102]. There is some intermixing of the sulfate with coarse particle soil. This sulfate is transported to the site given the uniformity of contributions at multiple sites. This factor was an important contributor of particle mass during the summer sampling campaign.

Factor 5 has high sulfur concentrations in the smallest size bins and shows contributions from crustal species in intermediate sized particles. Small size sulfate is usually attributed to homogeneous sulfate formation. Given that off-road diesel fuel has a significantly higher sulfur content than on-road fuel, there may be some contribution from local diesel vehicles such as aircraft tugs and other ground vehicles. It is

not clear what the source of the soil particles might be in size around 1.0 to 2.5 μm. This factor was primarily observed in the April 2009 sampling period with a peak at the Stone House site during the winter. It is likely that there is some admixture of sand or construction material. Diesel construction equipment could also provide small particle sulfate.

	PM_{10} (ng/m^3)	Base Station April	Fire Station January	Stone House January	Fire Station July	Stone House July
Soil	mean	10000	1708	2422	3622	3758
	std dev	6044	1145	2488	2929	3173
	median	10995	1495	1664	2733	2412
Salt	mean	0	7240	8338	0	0
	std dev	0	3883	3782	0	0
	median	0	7703	9040	0	0
Landings	mean	4277	9870	7225	6920	6238
	std dev	4254	4147	4717	2995	3398
	median	2619	10277	7084	7362	6662
Sulfate[a]	mean	350	1210	1322	5695	5565
	std dev	403	1072	958	2543	2611
	median	190	1063	1392	5628	5603
Sulfate[b]	mean	4498	431	3175	759	606
	std dev	3814	498	3545	619	681
	median	4153	259	2379	623	481

Table 7.3: **Apportionment of PM_{10} for each site during each sampling campaign.**
a. Homogeneous Sulfate. b. Local Sulfate.

5 Conclusions

From the analysis of size- and time-resolved particle sample compositional data, five emission sources were identified using a weighted alternating least-squares method: soil, deicing road salt, aircraft landings, transported secondary sulfate, and local sulfate/construction. The largest source associated with the airport operations was aircraft landing that had not been previously considered as a significant source of

particles.

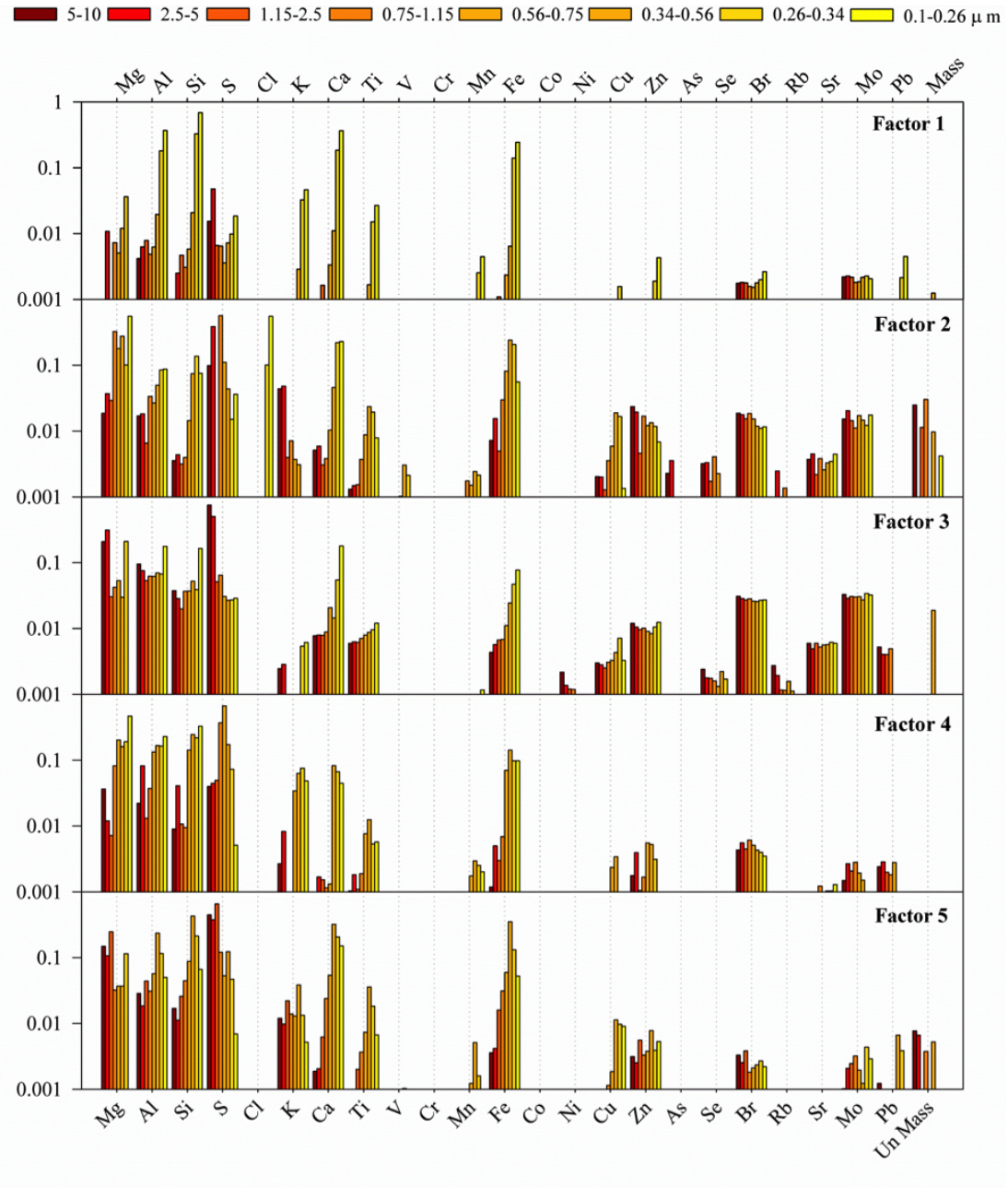


Figure 7.1: **Source profiles for the resolved factors.**

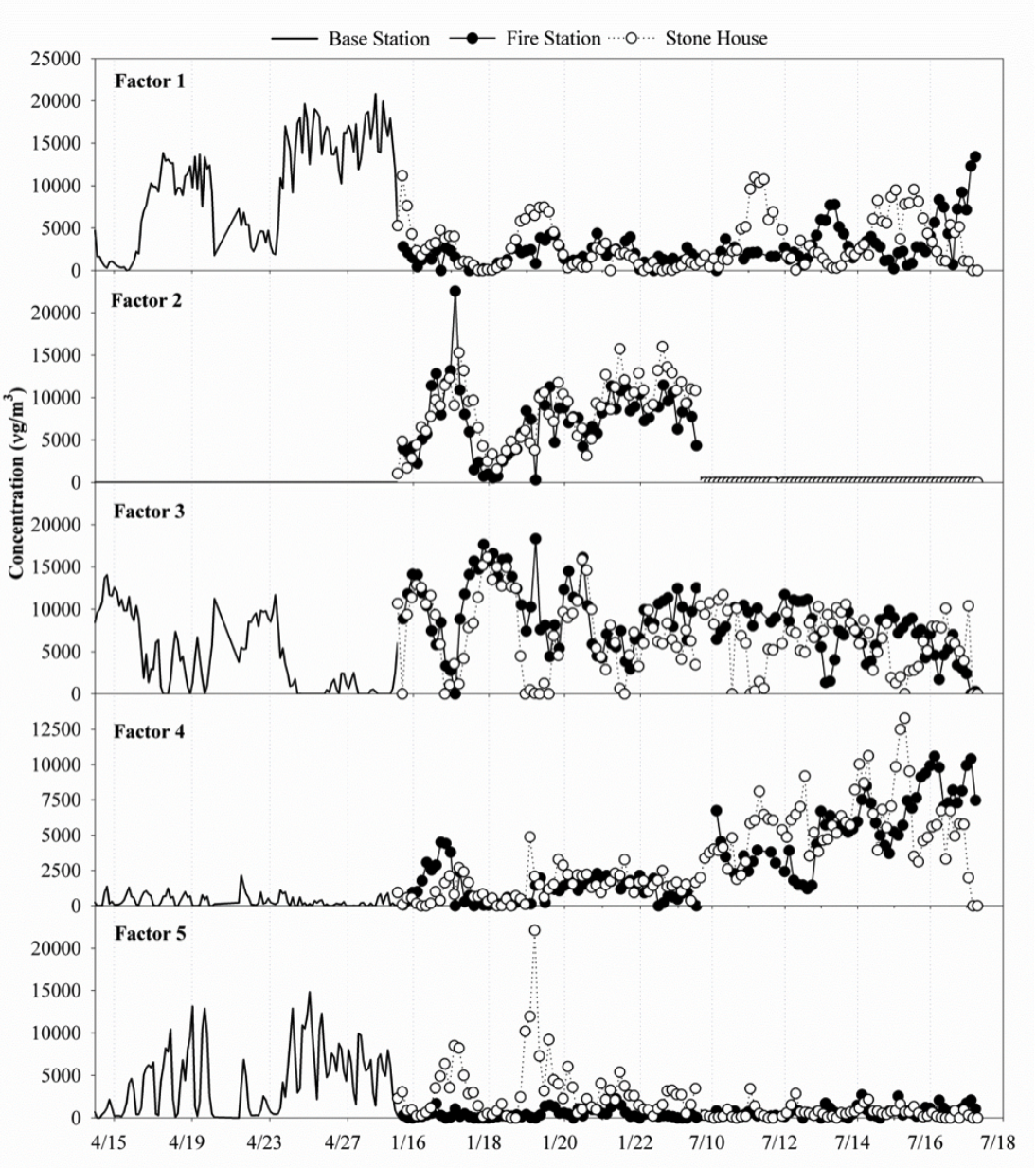


Figure 7.2: **The time series of source contributions.**

Chapter 8

Application II: Nonnegative tensor decomposition with sparseness constraints on sound separation

Tensor decompositions have many applications in the field of signal processing [36, 31, 32, 26]. The CP decomposition has been used to attempt sound source separation, with a focus on multi-channel sound source separation [44]. In this chapter, we provide an application of the tensor CP decomposition with non-negativity and sparseness constraints [82]. It is shown that two-channel mixture of several sound sources separated. In addition, we compare the results from Nonnegative Tensor Factorization (NTF) with the results from NTF using spareness constraints and show that the latter one is more accurate than NTF.

1 Introduction

Sound source separation refers to the problem of synthesizing source signals given a mixture of those same source signals. FitzGerald, etc. [44] provides the background and the theoretical analysis about how the sound mixture can be modeled into a

tensor structure with a CP decomposition. The original tensor $\mathcal{T} \in \mathbb{R}^{I\times J\times K}$ contains the power spectrogram of the multi-channel mixed signals. Such $\mathcal{T}$ has the following CP decomposition:

$$\mathcal{T} = \sum_{r=1}^{R} \mathbf{a}_r \circ \mathbf{s}_r \circ \mathbf{g}_r, \tag{1.1}$$

so the factor matrices are $\mathbf{A} = [\mathbf{a}_1 \ \dots \ \mathbf{a}_R]$, $\mathbf{B} = [\mathbf{b}_1 \ \dots \ \mathbf{b}_R]$ and $\mathbf{G} = [\mathbf{g}_1 \ \dots \ \mathbf{g}_R]$. The entries of matrix $\mathbf{G}$ are the gains of each independent source from the different channels and each matrix $\mathbf{a}_r \circ \mathbf{s}_r^{\mathrm{T}}$ is a power spectrogram of a source signals. The rank R refers to the sum of the number of different notes played by all the instrument sources. According to [44], at present, there is no method for automatically estimating the number of factors (the rank R). Therefore, the data has an inherent tensor structure and hence, tensor decomposition methods are applied.

Our experiment works on a two-channel mixture of the sound source signals from a clarinet, a piano and a steel drum, which is the time-domain input signal. Figure 8.1 shows this. Each channel is a 91139 time sampled signal in 2 seconds and is divided into 89 frames. The power spectrum is calculated within each frame using the discrete Fourier transform (DFT) for each channel (see Figure 8.2a). Then the two spectrograms are stacked into a tensor format (see Figure 8.2b). The two-channel mixture of the signals is a tensor $\mathcal{T}$ of size $1024 \times 89 \times 2$.

For such a tensor $\mathcal{T}$, we want to use the ALS method to factor it, so our problem is

$$\min_{\mathbf{A},\mathbf{S},\mathbf{G}} \quad \|\mathcal{T} - \sum_{r=1}^{R} \mathbf{A}_r \circ \mathbf{S}_r \circ \mathbf{G}_r\|_F^2 \tag{1.2}$$

By ALS method, the problem (1.2) is reformulated into the following three subprob-

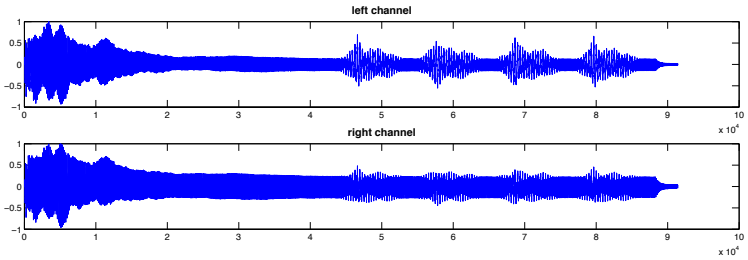
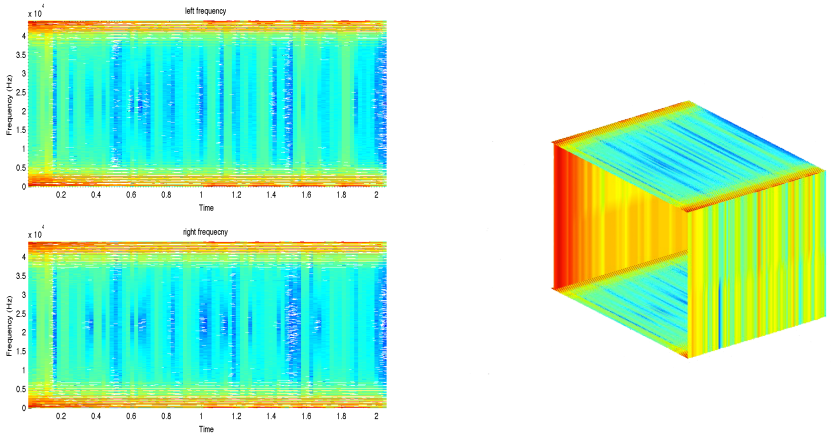


Figure 8.1: **The two-channel sound signal mixtures**



(a) The frequency of the two channels (b) Tensor

Figure 8.2: **Tensor structure: the left frequency is the top slice and the right frequency is the bottom slice, obtaining a rotated tensor $\mathcal{T} \in \mathbb{R}^{1024\times 89\times 2}$**

lems:

$$\begin{aligned}
\mathbf{A}^{k+1} &= \underset{\widehat{\mathbf{A}}\in\mathbb{R}^{I\times R}}{\operatorname{argmin}} \|\mathbf{T}_{(\mathbf{1})}{}^{I\times JK} - \widehat{\mathbf{A}}(\mathbf{G}^k \odot \mathbf{S}^k)^{\mathrm{T}}\|_F^2, \\
\mathbf{S}^{k+1} &= \underset{\widehat{\mathbf{S}}\in\mathbb{R}^{J\times R}}{\operatorname{argmin}} \|\mathbf{T}_{(\mathbf{2})}{}^{J\times IK} - \widehat{\mathbf{S}}(\mathbf{G}^k \odot \mathbf{A}^{k+1})^{\mathrm{T}}\|_F^2, \\
\mathbf{G}^{k+1} &= \underset{\widehat{\mathbf{G}}\in\mathbb{R}^{K\times R}}{\operatorname{argmin}} \|\mathbf{T}_{(\mathbf{3})}{}^{K\times IJ} - \widehat{\mathbf{G}}(\mathbf{S}^{k+1} \odot \mathbf{A}^{k+1})^{\mathrm{T}}\|_F^2.
\end{aligned} \tag{1.3}$$

For sound separation applications, we require additional constraints on each of the subproblems in the ALS algorithm: all the three factor matrices should be nonnegative and $\mathbf{A}$ also needs be sparse since it represents the signals in the frequency basis.

2 Constrains

2.1 NMF

To obtain nonnegative matrices $\mathbf{S}$ and $\mathbf{G}$, the last two subproblems in (1.3) can be replaced by the optimization problem of finding nonnegative factors $\mathbf{W}$ and $\mathbf{H}$ from a nonnegative matrix $\mathbf{V}$:

$$\text{minimize}\|\mathbf{V}-\mathbf{WH}\|_F^2, \quad \text{subject to } \mathbf{W},\mathbf{H}\succeq 0 \tag{2.1}$$

where $\mathbf{W}\succeq 0$ denotes that all the entries of $\mathbf{W}$ are nonnegative. This is called Nonnegative Matrix Factorization (NMF). Several algorithms have been proposed for NMF, namely, a gradient-based method by Paatero [99] and a multiplicative updating algorithm by Lee and Seung [109].

For the factor matrix $\mathbf{A}$, we have to deal with two constraints: sparsity and nonnegativity. We implement two methods differing in the order of how the constraints are imposed: NMF with sparse constraints [59] (NMF-Sparse) and ℓ_1-minimization with nonnegative constraints (ℓ_1-NMF).

2.2 ℓ_1-NMF

The first subproblem in (1.3) is reformulated by first vectorizing the equation ${\mathbf{T}_{(\mathbf{1})}}^{I\times JK} = \widehat{\mathbf{A}}(\mathbf{G}^k\odot\mathbf{S}^k)^{\mathrm{T}}$ into $\mathbf{t}=\mathbf{Q}\widehat{\mathbf{a}}$ via column stacking with $\mathbf{Q}=\mathbf{I}^{I\times I}\otimes(\mathbf{G}^k\odot\mathbf{S}^k)\in\mathbb{R}^{IJK\times IR}$, $\mathbf{t}\in\mathbb{R}^{IJK}$ and $\widehat{\mathbf{a}}\in\mathbb{R}^{IR}$. Then the least-squares subproblem is replaced by an ℓ_1-minimization with equality constraints [56]:

$$\min\|\widehat{\mathbf{a}}\|_{\ell_1} \quad \text{subject to } \mathbf{t}=\mathbf{Q}\widehat{\mathbf{a}}. \tag{2.2}$$

Note that if $\mathbf{x}\in\mathbb{R}^n$, then $\|\mathbf{x}\|_{\ell_1}=\sum_{i=1}^{n}|x_i|$. The idea behind this model (2.2) is to construct a sparse and exact solution vector $\widehat{\mathbf{a}}$ which matricizes into a sparse factor

matrix $\widehat{\mathbf{A}}$ satisfying the CP decomposition.

The ℓ_1 minimization problem can be recast as linear program [13]:

$$\min \mathbf{1}'\widehat{\mathbf{a}} \quad \text{subject to } \widehat{\mathbf{a}} \geq 0 \text{ and } \mathbf{Q}\widehat{\mathbf{a}} = \mathbf{t} \tag{2.3}$$

with both equality and inequality constraints.

The link between ℓ_1 minimization and linear programs has been known since the 1950's in the paper of [65]. Moreover, numerical techniques for solving linear programs have been well studied. In our codes, we implement Matlab's linear programming algorithm based on the simplex and interior-point methods. To impose the nonnegative constraint on $\widehat{\mathbf{A}}$, the approximation from the linear program is further refined by a nonnegative least-squares method.

2.3 NMF-Sparse

In [59], based on the method proposed in [99] and [109], Hoyer added a sparseness constraint on the NMF algorithm. Below we summarize the algorithm found in [59]: for the nonnegative matrix $\mathbf{V} = \mathbf{T}_{(\mathbf{1})}{}^{k-1}$

1. Initialize $\mathbf{W} = \mathbf{A}^{k-1}$ calculated in a previous ALS iteration and set $\mathbf{H} = (\mathbf{G}^{k-1} \odot \mathbf{S}^{k-1})^{\mathrm{T}}$;

2. $\mathbf{W} \leftarrow \mathbf{W} - \mu_{\mathbf{W}}(\mathbf{WH} - \mathbf{V})\mathbf{H}^{\mathrm{T}}$;

3. Project each column of $\mathbf{W}$ to a vector that is nonnegative with same ℓ_2 norm, but the ℓ_1 norm is set to achieve the desired sparseness.

Here $\mu_{\mathbf{W}} = \frac{\mathbf{W}_{ia}}{(\mathbf{WHH}^T)_{ia}}$ is small positive multiplicative step from [109]. In [59], Hoyer provided the projection operator algorithm find the closest (in the Euclidean sense) nonnegative vector $\mathbf{s}$ for any vector $\mathbf{x}$ constrained to a given the ℓ_1 norm and a given

ℓ_2 norm. The desired sparsity of the factors, S_ω and S_h, are defined as

$$\text{sparseness}(\mathbf{W}_i) = S_\omega, \quad \text{sparseness}(\mathbf{H}_i) = S_h,$$

where

$$\text{sparseness}(\mathbf{x}) = \frac{\sqrt{n} - \frac{\|\mathbf{x}\|_{\ell_1}}{\|\mathbf{x}\|_{\ell_2}}}{\sqrt{n} - 1}.$$

for a vector $\mathbf{x}$.

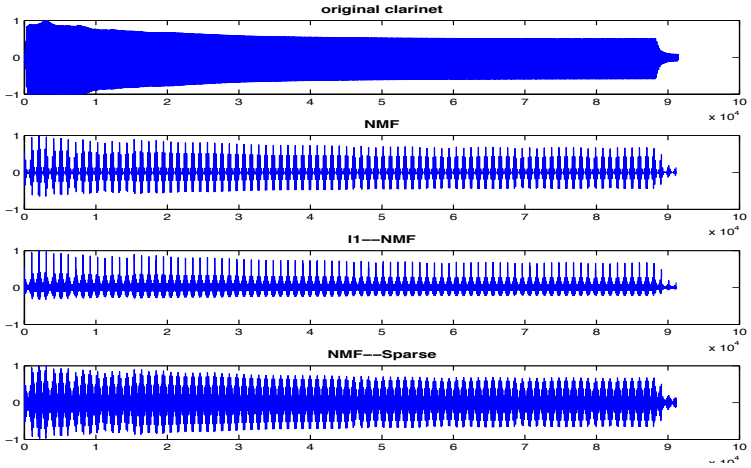
3 Numerical results

Now, we can decompose the tensor $\mathcal{T}$ by using the ALS method with those constraints. We want to compare the results from different constraints and methods for the factor matrices $\mathbf{A}$. The three are:

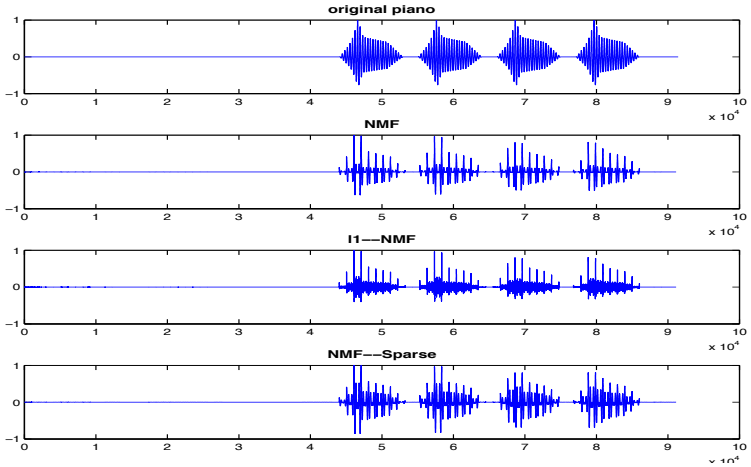
- Use NMF on $\mathbf{A}$, so there is no sparseness constraint here;
- Use ℓ_1-NMF to solve for $\mathbf{A}$;
- Use NMF-Sparse on $\mathbf{A}$.

So, we describe the algorithm as followings:

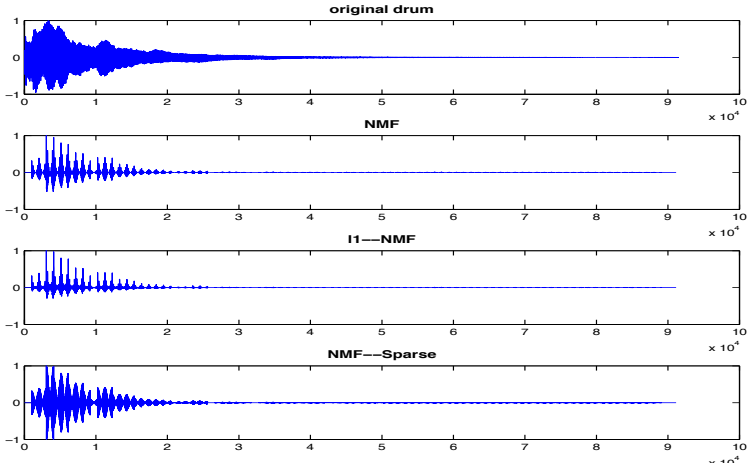
1. CP decomposition is applied to $\mathcal{T}$ via ALS (1.3) to obtain $\mathbf{A}$, $\mathbf{S}$ and $\mathbf{G}$. Then use one of the following three methods for factor $\mathbf{A}$ and apply the nonnegative matrix factorization method for factors $\mathbf{S}$ and $\mathbf{G}$.
    - (NMF)–For nonnegative $\mathbf{A}$, solve the first subproblem of (1.3) by NMF.
    - (ℓ_1-NMF)–For nonnegative sparse matrix $\mathbf{A}$, use ℓ_1 minimization with nonnegative constraints (2.2) as the method for the first least-squares subproblem.



(a) Separated waveforms for clarinet



(b) Separated waveforms for piano



(c) Separated waveforms for steel drum

Figure 8.3: **Waveforms of original signal, NMF, ℓ_1 nonnegative minimization and NMF with sparseness**

- (NMF-Sparse)–For nonnegative sparse $\mathbf{A}$, use NMF with sparseness constraints as a method by using the algorithm in Section (2.3).
- Nonnegative factor matrices $\mathbf{S}$ and $\mathbf{G}$ are solved by NMF.

2. For each $1 \leq r \leq R$, construct the source spectrogram, $\mathbf{F}_r = \mathbf{A}(:, r)\mathbf{S}(:, r)^T$.

3. The matrix $\mathbf{G}$ gives the ratios of each signal of the instrument in two channels. let a vector of size R, $H = \log \frac{\mathbf{G}(1,:)}{\mathbf{G}(2,:)}$. Using k-means cluster method [41], H is divided into 3 clusters, where each is from an instrument. So, H_r corresponds to $\mathbf{F}_r$ which should be in the same cluster.

4. Apply phase information [8] for the spectrogram obtained in 2 where the corresponding source signal up to the clusters is dominant to $\mathbf{F}_r$. Invert the spectrogram to obtain the time domain waveforms. See Figures 8.3a–8.3c.

Figure 8.1 shows the two channel sound signal mixture input formed into a tensor. We compared these methods (NMF-Sparse, ℓ_1-NMF) and NMF to the original waveforms in terms of the time domain waveform plots and frequency plots. Figures 8.3a, 8.3b and 8.3c show the comparison results, the waveform (top) in each figure is the original sound signal waveform. The rest of the waveforms (second from the top to bottom) in Figures 8.3a, 8.3b and 8.3c are NMF, ℓ_1-NMF and NMF-Sparse, respectively. It can be seen that all three methods can capture the main characteristics of the sources.

We measure the distance between two signals w and v in a least squared error sense [8] by the following equation:

$$E = \sum_{i=1}^{N} (|w_i| - |v_i|)^2 \tag{3.1}$$

where N is the length of the signals. In Figure 8.4, set 1 is the comparison of the separated clarinet signals, set 2 and set 3 are the separated piano and steel drum

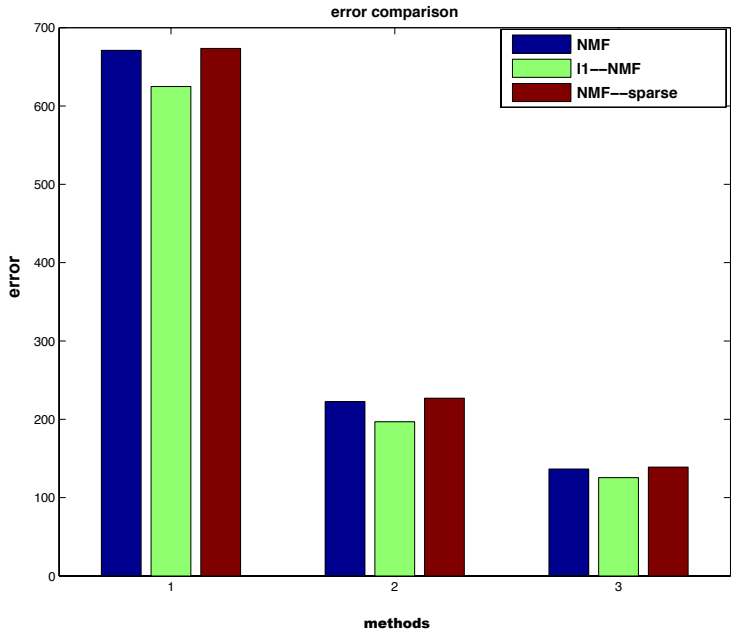


Figure 8.4: **Error comparison of the different methods**

signals, respectively. It shows that ℓ_1-NMF (the green bar) is better than the other two methods in all three sets.

Although both of ℓ_1-NMF and NMF-Sparse obtain a sparse and nonnegative factor matrix $\mathbf{A}$, we have seen that ℓ_1-NMF is better in the least squared error sense. Let us look at the clarinet spectrogram plots of the separated and original signals. Figure 8.5 shows that the NMF-Sparse method does not capture the spectral density at frequency between 0 to 0.5×10^4 and over 4×10^4 while the higher densities appear in the spectrograms of the NMF and ℓ_1-NMF methods.

4 Conclusion

Two methods for hybrid tensor nonnegative decomposition are presented in the application of sound source separation. The two methods, the ℓ_1-NMF and NMF-Sparse, are proposed to implement the hybrid tensor decomposition to obtain a sparse and nonnegative factor due to the sparsity of power spectral. The numerical examples

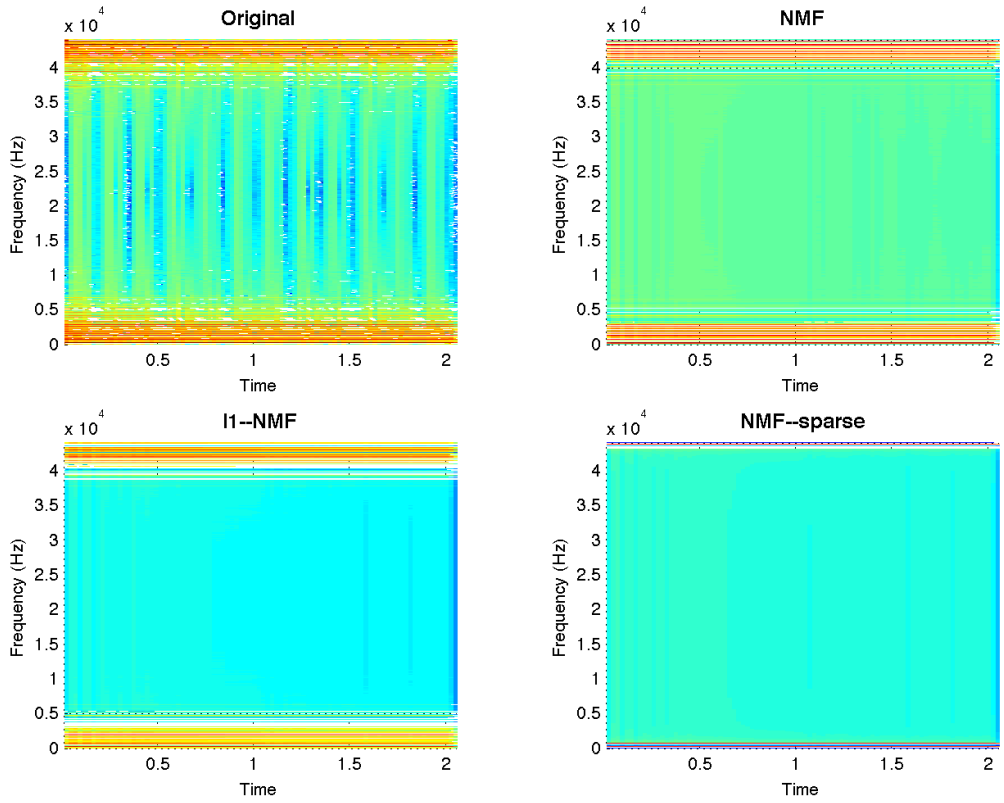


Figure 8.5: **Spectrograms of the clarinet for the varying methods, the top left one is the original spectrogram**

show the effectiveness of these techniques. Moreover, the error comparison plots show the ℓ_1 nonnegative minimization (ℓ_1-NMF) is better than NMF. The results of ℓ_1-NMF shows that it performs well with respect to the error and the resulting spectrogram plots of the separated signals which coincide with the actual sound. The NMF-Sparse method is the worst according to the error comparison. More study is needed to compare the two methods. Furthermore, we plan to develop more efficient techniques for sound source separation with better clustering methods.

Conclusion

This thesis studied the variants of alternating least-squares method for different tensor decompositions and two applications.

The classical alternating least-squares (ALS) method is proposed to solve the CANDECOMP/PARAFAC (CP) decomposition for a given Nth order tensor. However, a phenomenal called swamp that the objective function stays in a value and does not decrease for a long time happens in the computation of ALS. Then the regularized alternating least-squares (RALS) method is introduced to reduce the swamp. This thesis studied the RALS algorithm and proved the convergence property of RALS. In addition, we analyzed the ALS and provided numerical examples of the comparison of ALS and RALS.

For the partially symmetric tensor, the decomposition that factors it into a summation of rank-one partially symmetric tensors is studied in this thesis. We proposed an alternating method called Partial Column-Wise alternating least-squares (PCW-ALS) to compute the decomposition for the third-order case and two types of fourth-order partially symmetric tensor. Numerical examples are provided to compare the PCW-ALS algorithm with the ALS method. It has been shown that the PCW-ALS is more efficient than ALS.

The three-way receptor model is for the study of source apportionment of air pollutants. It is a relaxed version of BTD-$(L, L, 1)$ model. We provided one solution of this model. Furthermore, based on new formulation, we showed that the three-way

receptor model does not have unique solution.

Two applications were studied in this thesis. The source apportionment of time and size resolved ambient particulate matter studied the real data set collected from Washington-Dulles airport. The three-way weighted receptor model was applied with non-negativity constraints and a new method called weighted alternating least-squares (WALS) is used to solve the model. We successfully identified five major emission sources: soil, road salt, aircraft landings, transported secondary sulfate and local sulfate/construction.

The nonnegative tensor decomposition with sparseness constraints on sound separation studied two-channel mixture of sound source signals. The CP decomposition with non-negativity and sparseness constraints was applied. We proposed two methods to implement the CP to obtain a sparse and non-negative factor and made comparison of these methods.

Bibliography

[1] Evrim Acar, Canan Aykut-Bingol, Haluk Bingol, Rasmus Bro, and Bülent Yener. Multiway analysis of epilepsy tensors. *Bioinformatics*, 23(13):i10–i18, 2007.

[2] Claus A Andersson and Rasmus Bro. The n-way toolbox for matlab. *Chemometrics and Intelligent Laboratory Systems*, 52(1):1–4, 2000.

[3] Claus A Andersson and Rene Henrion. A general algorithm for obtaining simple structure of core arrays in n-way pca with application to fluorometric data. *Computational statistics & data analysis*, 31(3):255–278, 1999.

[4] Carl J Appellof and ER Davidson. Strategies for analyzing data from video fluorometric monitoring of liquid chromatographic effluents. *Analytical Chemistry*, 53(13):2053–2056, 1981.

[5] A Auslender. Asymptotic properties of the fenchel dual functional and applications to decomposition problems. *Journal of optimization theory and applications*, 73(3):427–449, 1992.

[6] Brett W Bader and Tamara G Kolda. Matlab tensor toolbox version 2.4, march 2010. *URL http://csmr. ca. sandia. gov/˜ tgkolda/TensorToolbox*.

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

[8] Dan Barry, Eugene Coyle, and Bob Lawlor. Comparison of signal reconstruction methods for the azimuth discrimination and resynthesis algorithm. In *Audio Engineering Society Convention 118*, 2005.

[9] Dimitri P Bertsekas. Nonlinear programming. 1999.

[10] Dimitri P Bertsekas and Paul Tseng. Partial proximal minimization algorithms for convex pprogramming. *SIAM Journal on Optimization*, 4(3):551–572, 1994.

[11] Gregory Beylkin and Martin J Mohlenkamp. Numerical operator calculus in higher dimensions. *Proceedings of the National Academy of Sciences*, 99(16):10246–10251, 2002.

[12] Gregory Beylkin and Martin J Mohlenkamp. Algorithms for numerical analysis in high dimensions. *SIAM Journal on Scientific Computing*, 26(6):2133–2159, 2005.

[13] Stephen Boyd and Lieven Vandenberghe. *Convex optimization*. Cambridge university press, 2004.

[14] Jerome Brachat, Pierre Comon, Bernard Mourrain, and Elias Tsigaridas. Symmetric tensor decomposition. *Linear Algebra and its Applications*, 433(11):1851–1872, 2010.

[15] Michael Brazell, Na Li, Carmeliza Navasca, and Christino Tamon. Tensor and matrix inversions with applications. *arXiv preprint arXiv:1109.3830*, 2011.

[16] Rasmus Bro. *Multi-way analysis in the food industry: models, algorithms, and applications*. PhD thesis, Kbenhavns UniversitetKbenhavns Universitet, Det

Biovidenskabelige Fakultet for Fdevarer, VeteFaculty of Life Sciences, Institut for FdevarevidenskabDepartment of Food Science, Kvalitet og TeknologiQuality & Technology, 1998.

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

[18] Beverly S Cohen, Arline L Bronzaft, Maire Heikkinen, Jerome Goodman, and Arthur Nádas. Airport-related air pollution and noise. *Journal of occupational and environmental hygiene*, 5(2):119–129, 2007.

[19] P Comon and P Chevalier. Blind source separation: Models, concepts, algorithms and performance. *Unsupervised adaptive filtering*, 1:191–237, 2000.

[20] Pierre Comon. Independent component analysis, a new concept? *Signal processing*, 36(3):287–314, 1994.

[21] Pierre Comon, Gene Golub, Lek-Heng Lim, and Bernard Mourrain. Symmetric tensors and symmetric tensor rank. *SIAM Journal on Matrix Analysis and Applications*, 30(3):1254–1279, 2008.

[22] Pierre Comon and Christian Jutten. *Handbook of Blind Source Separation: Independent component analysis and applications*. Academic press, 2010.

[23] Pierre Comon and Bernard Mourrain. Decomposition of quantics in sums of powers of linear forms. *Signal Processing*, 53(2):93–107, 1996.

[24] Pierre Comon, Jos MF ten Berge, Lieven De Lathauwer, and Josephine Castaing. Generic and typical ranks of multi-way arrays. *Linear Algebra and its Applications*, 430(11):2997–3007, 2009.

[25] André LF De Almeida, Gérard Favier, João CM Mota, et al. The constrained block-parafac decomposition. In *TRICAP'06*, 2006.

[26] Lieven De Lathauwer. *Signal processing based on multilinear algebra*. Katholieke Universiteit Leuven, 1997.

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

[28] Lieven De Lathauwer. Decompositions of a higher-order tensor in block terms-part i: Lemmas for partitioned matrices. *SIAM Journal on Matrix Analysis and Applications*, 30(3):1022–1032, 2008.

[29] Lieven De Lathauwer. Decompositions of a higher-order tensor in block terms-part ii: Definitions and uniqueness. *SIAM Journal on Matrix Analysis and Applications*, 30(3):1033–1066, 2008.

[30] Lieven De Lathauwer. A survey of tensor methods. In *Circuits and Systems, 2009. ISCAS 2009. IEEE International Symposium on*, pages 2773–2776. IEEE, 2009.

[31] Lieven De Lathauwer, Joséphine Castaing, and J-F Cardoso. Fourth-order cumulant-based blind identification of underdetermined mixtures. *Signal Processing, IEEE Transactions on*, 55(6):2965–2973, 2007.

[32] Lieven De Lathauwer and Alexandre de Baynast. Blind deconvolution of ds-cdma signals by means of decomposition in rank-$(1, l, l)$ terms. *Signal Processing, IEEE Transactions on*, 56(4):1562–1571, 2008.

[33] Lieven De Lathauwer, Bart De Moor, and Joos Vandewalle. A multilinear singular value decomposition. *SIAM journal on Matrix Analysis and Applications*, 21(4):1253–1278, 2000.

[34] Lieven De Lathauwer, Bart De Moor, and Joos Vandewalle. On the best rank-1 and rank-(r1, r 2,..., rn) approximation of higher-order tensors. *SIAM Journal on Matrix Analysis and Applications*, 21(4):1324–1342, 2000.

[35] Lieven De Lathauwer, Bart De Moor, and Joos Vandewalle. Computation of the canonical decomposition by means of a simultaneous generalized schur decomposition. *SIAM Journal on Matrix Analysis and Applications*, 26(2):295–327, 2004.

[36] Lieven De Lathauwer and Dimitri Nion. Decompositions of a higher-order tensor in block terms-part iii: Alternating least squares algorithms. *SIAM Journal on Matrix Analysis and Applications*, 30(3):1067–1083, 2008.

[37] Lieven De Lathauwer and Joos Vandewalle. Dimensionality reduction in higher-order signal processing and rank-(r1, r 2,..., rn) reduction in multilinear algebra. *Linear Algebra and its Applications*, 391:31–55, 2004.

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

[39] Maarten De Vos, A Vergult, Lieven De Lathauwer, Wim De Clercq, Sabine Van Huffel, Patrick Dupont, A Palmini, and Wim Van Paesschen. Canonical decomposition of ictal scalp eeg reliably detects the seizure onset zone. *NeuroImage*, 37(3):844–854, 2007.

[40] Jeffrey A Dodd, John M Ondov, Gurdal Tuncel, Thomas G Dzubay, and Robert K Stevens. Multimodal size spectra of submicrometer particles bearing various elements in rural air. *Environmental science & technology*, 25(5):890–903, 1991.

[41] Lars Eldén. *Matrix methods in data mining and pattern recognition*, volume 4. Society for Industrial and Applied Mathematics, 2007.

[42] Heinz Werner Engl, Martin Hanke, and Andreas Neubauer. *Regularization of inverse problems*, volume 375. Kluwer Academic Pub, 1996.

[43] Nicolaas Klaas M Faber, Rasmus Bro, and Philip K Hopke. Recent developments in candecomp/parafac algorithms: a critical review. *Chemometrics and Intelligent Laboratory Systems*, 65(1):119–137, 2003.

[44] Derry FitzGerald, Matt Cranitch, and Eugene Coyle. Non-negative tensor factorisation for sound source separation. 2005.

[45] Feng Gan and Philip K Hopke. New convergence criterion for multi-variable curve resolution. *Analytica chimica acta*, 495(1):195–203, 2003.

[46] L Grippo and M Sciandrone. On the convergence of the block nonlinear gauss–seidel method under convex constraints. *Operations Research Letters*, 26(3):127–136, 2000.

[47] Luigi Grippof and Marco Sciandrone. Globally convergent block-coordinate techniques for unconstrained optimization. *Optimization methods and software*, 10(4):587–637, 1999.

[48] Per Christian Hansen. Analysis of discrete ill-posed problems by means of the l-curve. *SIAM review*, 34(4):561–580, 1992.

[49] R. A. Harshman. Foundations of the PARAFAC procedure: Model and conditions for an 'explanatory' multi-mode factor analysis. *UCLA Working Papers Phonetics*, 16, 1970.

[50] R. A. Harshman. Determination and proof of minimum uniqueness conditions for PARAFAC1. *UCLA Working Papers in Phonetics*, 22, 1972.

[51] Johan Håstad. Tensor rank is np-complete. *Journal of Algorithms*, 11(4):644–654, 1990.

[52] René Henrion. Body diagonalization of core matrices in three-way principal components analysis: Theoretical bounds and simulation. *Journal of chemometrics*, 7(6):477–494, 1993.

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

[54] FL Hitchcock. Multiple invariants and generalized rank of a p-way matrix or tensor. *J. Math. Phys*, 7(1):39–79, 1927.

[55] Frank Lauren Hitchcock. *The expression of a tensor or a polyadic as a sum of products*. Inst. of Technology, 1927.

[56] Jason R Holloway and Carmeliza Navasca. Recovering tensor data from incomplete measurement via compressive sampling. In *Signals, Systems and Computers, 2009 Conference Record of the Forty-Third Asilomar Conference on*, pages 1310–1314. IEEE, 2009.

[57] Philip K Hopke et al. *Receptor modeling in environmental chemistry.* John Wiley & Sons, 1985.

[58] Philip K Hopke, Robert E Lamb, and David FS Natusch. Multielemental characterization of urban roadway dust. *Environmental science & technology*, 14(2):164–172, 1980.

[59] Patrik O Hoyer. Non-negative matrix factorization with sparseness constraints. *The Journal of Machine Learning Research*, 5:1457–1469, 2004.

[60] Aapo Hyvarinen, Juha Karhunen, and Erkki Oja. Independent component analysis. *STUDIES IN INFORMATICS AND CONTROL*, 11(2):205–207, 2002.

[61] Tao Jiang and Nicholas D Sidiropoulos. Kruskal's permutation lemma and the identification of candecomp/parafac and bilinear models with constant modulus constraints. *Signal Processing, IEEE Transactions on*, 52(9):2625–2636, 2004.

[62] Ten Berge J.M.F. and Kiers H.A.L. Simplicity of core arrays in three-way principal component analysis and the typical rank of pxqx2 arrays. *Linear Algebra and its Applications*, 294(1):169–179, 1999.

[63] Ten Berge J.M.F., Sidiropoulos N.D., and Rocci R. Typical rank and indscal dimensionality for symmetric three-way arrays of order ix2x2 or ix3x3. *Linear Algebra Appl.*, 388, 2004.

[64] Arie Kapteyn, Heinz Neudecker, and Tom Wansbeek. An approach to n-mode components analysis. *Psychometrika*, 51(2):269–275, 1986.

[65] Narendra Karmarkar. A new polynomial-time algorithm for linear programming. In *Proceedings of the sixteenth annual ACM symposium on Theory of computing*, pages 302–311. ACM, 1984.

[66] Henk AL Kiers. Towards a standardized notation and terminology in multiway analysis. *Journal of chemometrics*, 14(3):105–122, 2000.

[67] Brian Y Kim. *Guidance for Quantifying the Contribution of Airport Emissions to Local Air Quality*, volume 71. Transportation Research Board, 2012.

[68] A Knöchel, W Petersen, and G Tolkiehn. X-ray fluorescence analysis with synchrotron radiation. *Nuclear Instruments and Methods in Physics Research*, 208(1):659–663, 1983.

[69] Eleftherios Kofidis and Phillip A Regalia. On the best rank-1 approximation of higher-order supersymmetric tensors. *SIAM Journal on Matrix Analysis and Applications*, 23(3):863–884, 2002.

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

[71] Tamara Gibson Kolda. *Multilinear operators for higher-order decompositions*. United States. Department of Energy, 2006.

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

[73] Pieter M Kroonenberg. *Applied multiway data analysis*, volume 702. Wiley-Interscience, 2008.

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

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

[76] Joseph B Kruskal. Statement of some current results about three-way arrays. In *Unpublished manuscript*. AT&T Bell Labs, Murray Hill, NC, 1983.

[77] Joseph B Kruskal. Rank, decomposition, and uniqueness for 3-way and n-way arrays. *Multiway data analysis*, pages 7–18, 1989.

[78] Joseph PS Kung and Gian-Carlo Rota. The invariant theory of binary forms. *Bulletin of the American Mathematical Society*, 10(1):27–85, 1984.

[79] David C Lay. Linear algebra and its applications. *Addision-Wesley Publishing Company*, 1994.

[80] Na Li, Philip K Hopke, Pramod Kumar, Steven S Cliff, Yongjing Zhao, and Carmeliza Navasca. Source apportionment of time and size resolved ambient particulate matter. *Chemometrics and Intelligent Laboratory Systems*, 2013.

[81] Na Li, Stefan Kindermann, and Carmeliza Navasca. Some convergence results on the regularized alternating least-squares method for tensor decomposition. *Linear Algebra and its Applications*, 438(2):796–812, 2013.

[82] Na Li and Carmeliza Navasca. Sparseness constraints on nonnegative tensor decomposition. In *Signals, Systems and Computers (ASILOMAR), 2010 Conference Record of the Forty Fourth Asilomar Conference on*, pages 613–617. IEEE, 2010.

[83] Lek-Heng Lim. Singular values and eigenvalues of tensors: a variational approach. In *Computational Advances in Multi-Sensor Adaptive Processing, 2005 1st IEEE International Workshop on*, pages 129–132. IEEE, 2005.

[84] Lek-Heng Lim and Pierre Comon. Nonnegative approximations of nonnegative tensors. *Journal of Chemometrics*, 23(7-8):432–441, 2009.

[85] Ning Liu, Benyu Zhang, Jun Yan, Zheng Chen, Wenyin Liu, Fengshan Bai, and Leefeng Chien. Text representation: From vector to tensor. In *Data Mining, Fifth IEEE International Conference on*, pages 4–pp. IEEE, 2005.

[86] Charles F Van Loan. The ubiquitous kronecker product. *Journal of Computational and Applied Mathematics*, 123(1):85–100, 2000.

[87] Zhi-Quan Luo and Paul Tseng. On the convergence of the coordinate descent method for convex differentiable minimization. *Journal of Optimization Theory and Applications*, 72(1):7–35, 1992.

[88] Vladimir Alekseevich Morozov and Michael Stessin. *Regularization methods for ill-posed problems*. CRC Press Boca Raton, FL:, 1993.

[89] Takashi Murakami, Jos MF Ten Berge, and Henk AL Kiers. A case of extreme simplicity of the core matrix in three-mode principal components analysis. *Psychometrika*, 63(3):255–261, 1998.

[90] Damien Muti and Salah Bourennane. Multidimensional filtering based on a tensor approach. *Signal Processing*, 85(12):2338–2353, 2005.

[91] Carmeliza Navasca, Lieven De Lathauwer, and Stefan Kindermann. Swamp reducing technique for tensor decomposition. *Submitted for publication*, 2008.

[92] Dimitri Nion and Lieven De Lathauwer. An enhanced line search scheme for complex-valued tensor decompositions. application in ds-cdma. *Signal Processing*, 88(3):749–755, 2008.

[93] Pentti Paatero. A weighted non-negative least squares algorithm for three-way parafacfactor analysis. *Chemometrics and Intelligent Laboratory Systems*, 38(2):223–242, 1997.

[94] Pentti Paatero. The multilinear enginea table-driven, least squares program for solving multilinear problems, including the n-way parallel factor analysis model. *Journal of Computational and Graphical Statistics*, 8(4):854–888, 1999.

[95] Pentti Paatero. Construction and analysis of degenerate parafac models. *Journal of Chemometrics*, 14(3):285–299, 2000.

[96] Pentti Paatero. Construction and analysis of degenerate parafac models. *Journal of Chemometrics*, 14(3):285–299, 2000.

[97] Pentti Paatero and Philip K Hopke. Discarding or downweighting high-noise variables in factor analytic models. *Analytica Chimica Acta*, 490(1):277–289, 2003.

[98] Pentti Paatero, Philip K Hopke, Xin-Hua Song, and Ziad Ramadan. Understanding and controlling rotations in factor analytic models. *Chemometrics and intelligent laboratory systems*, 60(1):253–264, 2002.

[99] Pentti Paatero and Unto Tapper. Positive matrix factorization: A non-negative factor model with optimal utilization of error estimates of data values. *Environmetrics*, 5(2):111–126, 1994.

[100] Michael Patriksson. Decomposition methods for differentiable optimization problems over cartesian product sets. *Computational optimization and applications*, 9(1):5–42, 1998.

[101] Hazel Peace, Janet Maughan, Bethan Owen, and David Raper. Identifying the contribution of different airport related sources to local urban air quality. *Environmental modelling & software*, 21(4):532–538, 2006.

[102] Emma Peré-Trepat, Eugene Kim, Pentti Paatero, and Philip K Hopke. Source apportionment of time and size resolved ambient particulate matter measured with a rotating drum impactor. *Atmospheric Environment*, 41(28):5921–5933, 2007.

[103] Alexandr V Polissar, Philip K Hopke, Pentti Paatero, William C Malm, and James F Sisler. Atmospheric aerosol over alaska: 2. elemental composition and sources. *Journal of Geophysical Research: Atmospheres (1984–2012)*, 103(D15):19045–19057, 1998.

[104] Michael JD Powell. On search directions for minimization algorithms. *Mathematical Programming*, 4(1):193–201, 1973.

[105] Liqun Qi. Eigenvalues of a real supersymmetric tensor. *Journal of Symbolic Computation*, 40(6):1302–1324, 2005.

[106] Myriam Rajih, Pierre Comon, and Richard A Harshman. Enhanced line search: A novel method to accelerate parafac. *SIAM Journal on Matrix Analysis and Applications*, 30(3):1128–1147, 2008.

[107] C Radhakrishna Rao and Sujit Kumar Mitra. Generalized inverse of a matrix and its applications. *J. Wiley, New York*, 1971.

[108] Berkant Savas and Lars Eldén. Handwritten digit classification using higher order singular value decomposition. *Pattern recognition*, 40(3):993–1003, 2007.

[109] D Seung and L Lee. Algorithms for non-negative matrix factorization. *Advances in neural information processing systems*, 13:556–562, 2001.

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

[111] Nicholas D Sidiropoulos, Rasmus Bro, and Georgios B Giannakis. Parallel factor analysis in sensor array processing. *Signal Processing, IEEE Transactions on*, 48(8):2377–2388, 2000.

[112] Nicholas D Sidiropoulos, Georgios B Giannakis, and Rasmus Bro. Blind parafac receivers for ds-cdma systems. *Signal Processing, IEEE Transactions on*, 48(3):810–823, 2000.

[113] Age Smilde, Rasmus Bro, and Paul Geladi. *Multi-way analysis: applications in the chemical sciences*. Wiley, 2005.

[114] Alwin Stegeman. On uniqueness of the n th order tensor decomposition into rank-1 terms with linear independence in one mode. *SIAM Journal on Matrix Analysis and Applications*, 31(5):2498–2516, 2010.

[115] Alwin Stegeman. On uniqueness of the canonical tensor decomposition with some form of symmetry. *SIAM Journal on Matrix Analysis and Applications*, 32(2):561–583, 2011.

[116] Alwin Stegeman, Jos MF Ten Berge, and Lieven De Lathauwer. Sufficient conditions for uniqueness in candecomp/parafac and indscal with random component matrices. *Psychometrika*, 71(2):219–229, 2006.

[117] Jimeng Sun, Spiros Papadimitriou, and Philip S Yu. Window-based tensor analysis on high-dimensional and multi-aspect streams. In *Proceedings of the Sixth International Conference on Data Mining*, pages 1076–1080. IEEE Computer Society, 2006.

[118] Jos MF ten Berge. Kruskal's polynomial for 2x2x2 arrays and a generalization to 2xnxn arrays. *Psychometrika*, 56:631–636, 1991.

[119] Jos MF ten Berge. The typical rank of tall three-way arrays. *Psychometrika*, 65(4):525–532, 2000.

[120] AN Tikhonov and V Ya Arsenin. *Methods for solving ill-posed problems*, volume 15. Nauka, Moscow, 1979.

[121] Giorgio Tomasi. *Practical and computational aspects in chemometric data analysis*. KVL-PhD. KVL-PhD., 2006.

[122] Giorgio Tomasi and Rasmus Bro. A comparison of algorithms for fitting the parafac model. *Computational Statistics & Data Analysis*, 50(7):1700–1734, 2006.

[123] Ledyard R Tucker. Implications of factor analysis of three-way matrices for measurement of change. *Problems in measuring change*, pages 122–137, 1963.

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

[125] M Alex O Vasilescu and Demetri Terzopoulos. Multilinear analysis of image ensembles: Tensorfaces. In *Computer VisionECCV 2002*, pages 447–460. Springer, 2002.

[126] Peter Wentzell, Tobias Karakach, Sushmita Roy, M Juanita Martinez, Christopher Allen, and Margaret Werner-Washburne. Multivariate curve resolution of time course microarray data. *BMC bioinformatics*, 7(1):343, 2006.

[127] Dane Westerdahl, Scott A Fruin, Phillip L Fine, and Constantinos Sioutas. The los angeles international airport as a source of ultrafine particles and other pollutants to nearby communities. *Atmospheric Environment*, 42(13):3143–3155, 2008.

[128] JP Wold, Rasmus Bro, A Veberg, F Lundby, AN Nilsen, and J Moan. Active photosensitizers in butter detected by fluorescence spectroscopy and multivariate curve resolution. *Journal of agricultural and food chemistry*, 54(26):10197–10204, 2006.