VIETNAM NATIONAL UNIVERISTY, HANOI

UNIVERSITY OF ENGINEERING AND TECHNOLOGY

LE TRUNG THANH

GMNS-BASED TENSOR DECOMPOSITION

MASTER THESIS: COMMUNICATIONS ENGINEERING

Hanoi, 11/2018

VIETNAM NATIONAL UNIVERISTY, HANOI

UNIVERSITY OF ENGINEERING AND TECHNOLOGY

LE TRUNG THANH

GMNS-BASED TENSOR DECOMPOSITION

Program: Communications Engineering

Major: Electronics and Communications Engineering

Code: 8510302.02

MASTER THESIS: COMMUNICATIONS ENGINEERING

SUPERVISOR: Assoc. Prof. NGUYEN LINH TRUNG

Hanoi – 11/2018

Authorship

"I hereby declare that the work contained in this thesis is of my own and has not been previously submitted for a degree or diploma at this or any other higher education institution. To the best of my knowledge and belief, the thesis contains no materials previously or written by another person except where due reference or acknowledgement is made".

Signature: .....

Supervisor's approval

"I hereby approve that the thesis in its current form is ready for committee examination as a requirement for the Degree of Master in Electronics and Communications Engineering at the University of Engineering and Technology".

Signature: .....

Acknowledgments

This thesis would not have been possible without the guidance and the help of several individuals who contributed and extended their valuable assistance in the preparation and completion of this study.

I am deeply thankful to my family, who have been sacrificing their whole life for me and always supporting me throughout my education process.

I would like to express my sincere gratitude to my supervisor, Prof. Nguyen Linh Trung who introduced me to the interesting research problem of tensor analysis that combines multilinear algebra and signal processing. Under his guidance, I have learned many useful things from him such as passion, patience and academic integrity. I am lucky to have him as my supervisor. To me, he is the best supervisor who a student can ask for. Many thanks to Dr. Nguyen Viet Dung for his support, valuable comments on my work, as well as his professional experience in academic life. My main results in this thesis are inspired directly from his GMNN algorithm for subspace estimation.

I am also thankful to all members of the Signals and Systems Laboratory and my co-authors, Mr. Truong Minh Chinh, Mrs. Nguyen Thi Anh Dao, Mr. Nguyen Thanh Trung, Dr. Nguyen Thi Hong Thinh, Dr. Le Vu Ha and Prof. Karim Abed-Meraim for all their enthusiastic guidance and encouragement during the study and preparation for my thesis.

Finally, I would like to express my great appreciation to all professors of the Faculty of Electronics and Telecommunications for their kind teaching during the two years of my study.

The work presented in this thesis is based on the research and development conducted in Signals and Systems Laboratory (SSL) at University of Engineering and Technology within Vietnam National University, Hanoi (UET-VNU) and is funded by Vietnam National Foundation for Science and Technology Development (NAFOSTED) under grant number 102.02-2015.32.

The work has been presented in the following publication:

[1] **Le Trung Thanh**, Nguyen Viet-Dung, Nguyen Linh-Trung and Karim Abed-Meraim. "Three-Way Tensor Decompositions: A Generalized Minimum Noise Subspace Based Approach." *REV Journal on Electronics and Communications*, vol. 8, no. 1-2, 2018.

Publications in conjunction with my thesis but not included:

[2] **Le Trung Thanh**, Viet-Dung Nguyen, Nguyen Linh-Trung and Karim Abed-Meraim. "Robust Subspace Tracking with Missing Data and Outliers via ADMM ", in*The 44th International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, Brighton-UK, 2019. IEEE. [Submitted]

[3] **Le Trung Thanh**, Nguyen Thi Anh Dao, Viet-Dung Nguyen, Nguyen Linh-Trung, and Karim Abed-Meraim. "Multi-channel EEG epileptic spike detection by a new method of tensor decomposition". *IOP Journal of Neural Engineering*, Oct 2018. [under revision]

[4] Nguyen Thi Anh Dao, **Le Trung Thanh**, Nguyen Linh-Trung, Le Vu Ha. "Nonne-gative Tucker Decomposition for EEG Epileptic Spike Detection", in *2018 NAFOS-TED Conference on Information and Computer Science (NICS),* Ho Chi Minh, 2018, pp.196-201. IEEE.

Table of Contents

List of Figures

Abbreviations

Abbreviation	Definition
EEG	Electroencephalogram
GMNS	Generalized minimum noise subspace
MSA	Minor Subspace Analysis
SVD	Singular Value Decomposition
HOSVD	Higher-order SVD
PCA	Principal Component Analysis
PSA	Principal Subspace Analysis
PARAFAC	Parallel Factor Analysis

Abstract

Tensor decomposition has recently become a popular method of multi-dimensional data analysis in various applications. The main interest in tensor decomposition is for dimensionality reduction, approximation or subspace purposes. However, the emergence of "big data" now gives rise to increased computational complexity for performing tensor decomposition. In this thesis, motivated by the advantages of the generalized minimum noise subspace (GMNS) method, recently proposed for array processing, we proposed two algorithms for principal subspace analysis (PSA) and two algorithms for tensor decomposition using parallel factor analysis (PARAFAC) and higher-order singular value decomposition (HOSVD). The proposed decompositions can preserve several desired properties of PARAFAC and HOSVD while substantially reducing the computational complexity. Performance comparisons of PSA and tensor decompositions between using our proposed methods and the state-of-the-art methods are provided via numerical studies. Experimental results indicated that the proposed methods are of practical values.

Index Terms: Generalized minimum noise subspace, Principal subspace analysis, Tensor decomposition, Parallel factor analysis, Tucker decomposition, High-order singular value decomposition.

Chapter 1

Introduction

Over the last two decades, the number of large-scale datasets have been increasingly collected in various fields and can be smartly mined to discover new valuable information, helping us to obtain deeper understanding of the hidden values [6]. Many examples are seen in physical, biological, social, health and engineering science applications, wherein large-scale multi-dimensional, multi-relational and multi-model data are generated. Therefore, data analysis techniques using tensor decomposition now attract a great deal of attention from researchers and engineers.

A tensor is a multi-dimensional array and often considered as a generalization of a matrix. As a result, tensor representation gives a natural description of multi-dimensional data and hence tensor decomposition becomes a useful tool to analyze high-dimensional data. Moreover, tensor decomposition brings new opportunities for uncovering hidden and new values in the data. As a result, tensor decomposition has been used in various applications. For example, in neuroscience, brain signals are inherently multi-way data in general, and spatio-temporal in particular, due to the fact that they can be monitored through different brain regions at different times. In particular, an electroencephalography (EEG) dataset can be represented by a three-way tensor with three dimensions of time, frequency and electrode, or even by multi-way tensors when extra dimensions such as condition, subject and group are also considered. Tensor decomposition can be used to detect abnormal brain activities such as epileptic

seizures [7], to extract features of Alzheimer's disease [8] or other EEG applications, as reviewed in [9].

1.1 Tensor Decompositions

Two widely used decompositions for tensors are parallel factor analysis (PARAFAC) (also referred to as canonical polyadic decomposition) and Tucker decomposition. PARAFAC decomposes a given tensor into a sum of rank-1 tensors. Tucker decomposition decomposes a given tensor into a core tensor associated with a set of matrices (called factors) which are used to multiply along each mode (way to model a tensor along a particular dimension).

In the literature of tensors, many algorithms have been proposed for tensor decomposition. We can categorize them into three main approaches, respectively based on divide-and-conquer, compression, and optimization. The first approach aims to divide a given tensor into a finite number of sub-tensors, then estimate factors of the sub-tensors and finally combine them together into true factors. The central idea behind the second approach is to reduce the size of a given tensor until it becomes manageable before computing a specific decomposition of the compressed tensor, which retains the main information of the original tensor. In the third approach, tensor decomposition is cast into optimization and is then solved using standard optimization tools. We refer the reader to surveys in [10–12] for further details on the different approaches.

1.2 Objectives

In this thesis, we focus on the divide-and-conquer approach for PARAFAC and high-order singular value decomposition (HOSVD) of three-way tensors. HOSVD is a specific orthogonal form of Tucker decomposition. Examples of three-way tensors are numerous. (Image-row $\times$ image-column $\times$ time) tensors are used in video surveillance, human action recognition and real-time tracking [13–15]. (Spatial-row $\times$ spatial-column $\times$ wavelength) tensors are used for target detection and classification in hyperspectral image applications [16, 17]. (Origin $\times$ destination $\times$ time) tensors are used in transportation networks to discover the spatio-temporal traffic structure [18]. (Time $\times$ frequency $\times$ electrode) tensors are used in EEG analysis [7].

Recently, generalized minimum noise subspace (GMNS) was proposed by Nguyen *et al.* in [19] as a good technique for subspace analysis. This method is highly beneficial in practice because it not only substantially reduces the computational complexity in finding bases for these subspaces, but also provides high estimation accuracy. Several efficient algorithms for principal subspace analysis (PSA), minor subspace analysis (MSA), PCA utilizing the GMNS were proposed and shown to be applicable in various applications. This motivates us to propose in this thesis new implementations for tensor decomposition based on GMNS.

1.3 Contributions

The main contributions of this thesis are summarized as follows. First, by expressing the *right* singular vectors obtained from singular value decomposition (SVD) in terms

of principal subspace, we derive a *modified* GMNS algorithm for PSA with running time faster than the original GMNS, while still retaining the subspace estimation accuracy.

Second, we introduce a *randomized* GMNS algorithm for PSA that can deal with several matrices by performing the randomized SVD.

Third, we propose two algorithms for PARAFAC and HOSVD based on GMNS. The algorithms are highly beneficial and easy to implement in practice, thanks to its parallelized scheme with a low computational complexity. Several applications are studied to illustrate the effectiveness of the proposed algorithms.

1.4 Thesis organization

The structure of the thesis is organized as follows. Chapter 2 provides some background for our study, including two kinds of algorithms for PSA and tensor decomposition. Chapter 3 presents modified and randomized GMNS algorithms for PSA. Chapter 4 presents the GMNS-based algorithms for PARAFAC and HOSVD. Finally, Chapter 5 show experimental results. Chapter 6 gives conclusions on the developed algorithms.

Chapter 2

Preliminaries

In this chapter, we describe a brief review of tensors, related mathematical operators in multilinear algebra (e.g., tensor additions and multiplications). In addition, a divide-and-conquer algorithm for PARAFAC called alternating least-square (ALS) is also provided that is considered as fundamental of our proposed method. Moreover, it is of interest to first explain the central idea of the method before showing how GMNS can be used for tensor decomposition.

2.1 Tensor Notations and Definitions

Follow notations and definitions presented in [1], the mathematical symbols used in this thesis is summarized in the Table 2.1. We use lowercase letters (e.g., a), boldface lowercase letters (e.g., $\mathbf{a}$), boldface capital letters (e.g., $\mathbf{A}$) and bold calligraphic letters (e.g., $\boldsymbol{\mathcal{A}}$) to denote scalars, vectors, matrices and tensors respectively. For operators on a n-order tensor $\boldsymbol{\mathcal{A}}$, $\boldsymbol{\mathcal{A}}_{(k)}$ denotes the mode-k unfolding of $\boldsymbol{\mathcal{A}}$, $k \leq n$. The k-mode product of $\boldsymbol{\mathcal{A}}$ with a matrix $\mathbf{U}$ is denoted by $\boldsymbol{\mathcal{A}} \times_k \mathbf{U}$. The Frobenius norm of $\boldsymbol{\mathcal{A}}$ is denoted by $\|\boldsymbol{\mathcal{A}}\|_F$, meanwhile $\langle \boldsymbol{\mathcal{A}}, \boldsymbol{\mathcal{B}} \rangle$ denotes the inner product of $\boldsymbol{\mathcal{A}}$ and a same-sized tensor $\boldsymbol{\mathcal{B}}$. Specifically, definitions of these operators on $\boldsymbol{\mathcal{A}} \in \mathbb{R}^{I_1 \times I_2 \cdots \times I_n}$ used in this thesis are summarized as follows:

The mode-k unfolding $\boldsymbol{\mathcal{A}}_{(k)}$ of $\boldsymbol{\mathcal{A}}$ is a matrix in vector space $\mathbb{R}^{I_k \times (I_1 \ldots I_{k-1} I_{k+1} \ldots I_n)}$, in

Table 2.1: Mathematical Symbols

$a, \mathbf{a}, \mathbf{A}, \boldsymbol{\mathcal{A}}$	scalar, vector, matrix and tensor
$\mathbf{A}^T$	the transpose of $\mathbf{A}$
$\mathbf{A}^T$	the pseudo-inverse of $\mathbf{A}$
$\boldsymbol{\mathcal{A}}_{(k)}$	the mode-k unfolding of $\boldsymbol{\mathcal{A}}$
$\|\boldsymbol{\mathcal{A}}\|_F$	the Frobenius norm of $\boldsymbol{\mathcal{A}}$
$\mathbf{a} \circ \mathbf{b}$	the outer product of $\mathbf{a}$ and $\mathbf{b}$
$\mathbf{A} \otimes \mathbf{B}$	the Kronecker product of $\mathbf{A}$ and $\mathbf{B}$
$\boldsymbol{\mathcal{A}} \times_k \mathbf{U}$	the k-mode product of the tensor $\boldsymbol{\mathcal{A}}$ with a matrix $\mathbf{U}$
$\langle \boldsymbol{\mathcal{A}}, \boldsymbol{\mathcal{B}} \rangle$	the inner product of $\boldsymbol{\mathcal{A}}$ and $\boldsymbol{\mathcal{B}}$

which each element of $\boldsymbol{\mathcal{A}}_{(k)}$ is defined by

$$\boldsymbol{\mathcal{A}}_{(k)}(i_k, \overline{i_1 \dots i_{k-1} i_{k+1} \dots i_n}) = \boldsymbol{\mathcal{A}}(i_1, i_2, \dots, i_n).$$

where $(\imath_k, \overline{i_1 \dots i_{k-1} i_{k+1} \dots i_n})$ denotes the row and column of the matrix $\boldsymbol{\mathcal{A}}_{(k)}$.

The k-mode product of $\boldsymbol{\mathcal{A}}$ with a matrix $\mathbf{U} \in \mathbb{R}^{r_k \times I_k}$ yields a new tensor $\boldsymbol{\mathcal{B}} \in \mathbb{R}^{I_1 \times \cdots \times I_{k-1} \times r_k \times I_{k+1} \cdots \times I_n}$ such that

$$\boldsymbol{\mathcal{B}} = \boldsymbol{\mathcal{A}} \times_k \mathbf{U} \Leftrightarrow \boldsymbol{\mathcal{B}}_{(k)} = \mathbf{U} \boldsymbol{\mathcal{A}}_{(k)}.$$

As a result, we derive a desired property for the k-mode product as follows

$$\boldsymbol{\mathcal{A}} \times_k \mathbf{U} \times_l \mathbf{V} = \boldsymbol{\mathcal{A}} \times_l \mathbf{V} \times_k \mathbf{U} \text{ for } k \neq l,$$

$$\boldsymbol{\mathcal{A}} \times_k \mathbf{U} \times_k \mathbf{V} = \boldsymbol{\mathcal{A}} \times_k (\mathbf{V}\mathbf{U}).$$

The inner product of two n-order tensors $\boldsymbol{\mathcal{A}}, \boldsymbol{\mathcal{B}} \in \mathbb{R}^{I_1 \times I_2 \cdots \times I_n}$ is defined by

$$\langle \boldsymbol{\mathcal{A}}, \boldsymbol{\mathcal{B}} \rangle = \sum_{i_1=1}^{I_1} \cdots \sum_{i_n=1}^{I_n} \boldsymbol{\mathcal{A}}(i_1, i_2, \dots, i_n) \boldsymbol{\mathcal{B}}(i_1, i_2, \dots, i_n).$$

The Frobenius norm of a tensor $\boldsymbol{\mathcal{A}} \in \mathbb{R}^{I_1 \times I_2 \cdots \times I_n}$ is defined by the inner product of $\boldsymbol{\mathcal{A}}$ with itself

$$\|\boldsymbol{\mathcal{A}}\|_F = \sqrt{\langle \boldsymbol{\mathcal{A}}, \boldsymbol{\mathcal{A}} \rangle}.$$

For operators on a matrix $\mathbf{A} \in \mathbb{R}^{I_1 \times I_2}$, $\mathbf{A}^T$ and $\mathbf{A}^T$ denote the transpose and the pseudo-inverse of $\mathbf{A}$ respectively. The Kronecker product of $\mathbf{A}$ with a matrix $\mathbf{B} \in \mathbb{R}^{J_1 \times J_2}$, denoted by $\mathbf{A} \otimes \mathbf{B}$, yields a matrix $\mathbf{C} \in \mathbb{R}^{I_1 J_1 \times I_2 J_2}$ defined by

$$\mathbf{C} = \mathbf{A} \otimes \mathbf{B} = \begin{bmatrix} a_{1,1}\mathbf{B} & \dots & a_{1,I_2}\mathbf{B} \\ \vdots & \ddots & \vdots \\ a_{I_1,1}\mathbf{B} & \dots & a_{I_1,I_2}\mathbf{B} \end{bmatrix}.$$

For operators on a vector $\mathbf{a} \in \mathbb{R}^{I_1 \times 1}$, the outer product of $\mathbf{a}$ and vector $\mathbf{b} \in \mathbb{R}^{I_2 \times 1}$, denoted by $\mathbf{a} \circ \mathbf{b}$, yields a matrix $\mathbf{C} \in \mathbb{R}^{I_1 \times I_2}$ defined by

$$\mathbf{C} = \mathbf{a} \circ \mathbf{b} = \mathbf{a}\mathbf{b}^T = \begin{bmatrix} b_1\mathbf{a} & b_2\mathbf{a} & \dots & b_{I_2}\mathbf{a} \end{bmatrix}.$$

2.2 PARAFAC based on Alternating Least-Squares

Several divide-and-conquer based algorithms have been proposed for PARAFAC such as [20–25]. The central idea of the approach is to divide a tensor $\boldsymbol{\mathcal{X}}$ into k parallel sub-tensors $\boldsymbol{\mathcal{X}}_i$, then estimate the factors (loading matrices) of the sub-tensors, and then combine them together into the factors of $\boldsymbol{\mathcal{X}}$. In this section, we would like to describe the algorithm proposed by Nguyen *et al.* in [23], namely parallel ALS-based PARAFAC summarized in Algorithm 1, which has motivated us to develop new algorithms in this thesis.

Algorithm 1: Parallel ALS-based PARAFAC [23]

Input: Tensor $\boldsymbol{\mathcal{X}} \in \mathbb{R}^{I \times J \times K}$, target rank p, k DSP units

Output: Factors $\mathbf{A} \in \mathbb{R}^{I \times p}, \mathbf{B} \in \mathbb{R}^{J \times p}, \mathbf{C} \in \mathbb{R}^{K \times p}$

function

- Divide $\boldsymbol{\mathcal{X}}$ into k sub-tensors $\boldsymbol{\mathcal{X}}_1, \boldsymbol{\mathcal{X}}_2, \ldots, \boldsymbol{\mathcal{X}}_k$
- Compute $\mathbf{A}_1, \mathbf{B}_1, \mathbf{C}_1$ of $\boldsymbol{\mathcal{X}}_1$ using ALS
- Compute factors of sub-tensors: *// updates can be done in parallel*
- **for** $i = 2 \to k$ **do**
  - Compute $\mathbf{A}_i, \mathbf{B}_i$ and $\mathbf{C}_i$ of $\boldsymbol{\mathcal{X}}_i$ using ALS
  - Rotate $\mathbf{A}_i, \mathbf{B}_i$ and $\mathbf{C}_i$ // (2.4)
- Update $\mathbf{A}, \mathbf{B}, \mathbf{C}$ // (2.5)

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

Without loss of generality, we assume that a tensor $\boldsymbol{\mathcal{X}}$ is divided into k sub-tensors $\boldsymbol{\mathcal{X}}_1, \boldsymbol{\mathcal{X}}_2, \ldots, \boldsymbol{\mathcal{X}}_k$, by splitting the loading matrix $\mathbf{C}$ into $\mathbf{C}_1, \mathbf{C}_2, \ldots, \mathbf{C}_k$ so that the corresponding matrix presentation of the sub-tensor $\boldsymbol{\mathcal{X}}_i$ can be determined by

$$\mathbf{X}_i = (\mathbf{C}_i \odot \mathbf{A})\mathbf{B}^T. \tag{2.1}$$

Here, $\boldsymbol{\mathcal{X}}_i$ is considered as a tensor composed of frontal slices of $\boldsymbol{\mathcal{X}}$, while $\mathbf{X}_i$ is to present the sub-matrix of its matrix representation $\mathbf{X}$ of $\boldsymbol{\mathcal{X}}$.

Exploiting the fact that the two factors $\mathbf{A}$ and $\mathbf{B}$ are unique when decomposing the sub-tensors, thanks to the uniqueness of PARAFAC (see [11, Section IV] and [12, Section III]), gives

$$\boldsymbol{\mathcal{X}}_i = \boldsymbol{\mathcal{I}}_i \times_1 \mathbf{A} \times_2 \mathbf{B} \times_3 \mathbf{C}_i. \tag{2.2}$$

As a result, we here need to look for an updated rule to concatenate the matrices $\mathbf{C}_i$ into the matrix $\mathbf{C}$, while $\mathbf{A}$ and $\mathbf{B}$ can be obtained directly from PARAFAC of $\boldsymbol{\mathcal{X}}_1$.

In particular, the algorithm can be described as follows. First, by performing

PARAFAC of these sub-tensors, the factors $\mathbf{A}_i, \mathbf{B}_i,$ and $\mathbf{C}_i$ can be obtained from decomposing

$$\mathbf{X}_i = (\mathbf{C}_i \odot \mathbf{A}_i)\mathbf{B}_i^T, \tag{2.3}$$

using the Alternative Least-Squares (ALS) algorithm [26]. Then, $\mathbf{A}_i, \mathbf{B}_i, \mathbf{C}_i$ are rotated in the directions of $\boldsymbol{\mathcal{X}}_1$ to yield

$$\mathbf{A}_i \leftarrow \mathbf{A}_i\mathbf{P}_i\mathbf{D}_i^{(\mathbf{A})}, \tag{2.4a}$$

$$\mathbf{B}_i \leftarrow \mathbf{B}_i\mathbf{P}_i\mathbf{D}_i^{(\mathbf{B})}, \tag{2.4b}$$

$$\mathbf{C}_i \leftarrow \mathbf{C}_i\mathbf{P}_i\mathbf{D}_i^{(\mathbf{C})}, \tag{2.4c}$$

where the permutation matrices $\mathbf{P}_i \in \mathbb{R}^{R\times R}$ and scale matrices $\mathbf{D}_i^{(\cdot)} \in \mathbb{R}^{R\times R}$ are computed below

$$\mathbf{P}_i(u,v) = \begin{cases} 1, & \text{for } \max_v \dfrac{|\langle \mathbf{A}_i(:,u), \mathbf{A}_1(:,v)\rangle|}{\|\mathbf{A}_i(:,u)\|\|\mathbf{A}_1(:,v)\|}, \\ 0, & \text{otherwise,} \end{cases}$$

$$\mathbf{D}_i^{(\mathbf{A})}(u,u) = \frac{\|\mathbf{A}_1(:,v)\|}{|\langle \mathbf{A}_i(:,u), \mathbf{A}_1(:,v)\rangle|},$$

$$\mathbf{D}_i^{(\mathbf{B})}(u,u) = \frac{\|\mathbf{B}_1(:,v)\|}{|\langle \mathbf{B}_i(:,u), \mathbf{B}_1(:,v)\rangle|},$$

$$\mathbf{D}_i^{(\mathbf{C})}(u,u) = \left(\mathbf{D}_i^{(\mathbf{A})}(u,u)\mathbf{D}_i^{(\mathbf{B})}(u,u)\right)^{-1}.$$

Finally, we obtain the factors of $\boldsymbol{\mathcal{X}}$

$$\mathbf{A} \leftarrow \mathbf{A}_1, \mathbf{B} \leftarrow \mathbf{B}_1, \tag{2.5a}$$

$$\mathbf{C} \leftarrow \left[\mathbf{C}_1^T\ \mathbf{C}_2^T\ \ldots\ \mathbf{C}_k^T\right]^T. \tag{2.5b}$$

Algorithm 2: GMNS-based PSA [19]

Input: Matrix $\mathbf{X} \in \mathbb{C}^{n\times m}$, target rank p, k DSP units
Output: Principal subspace matrix $\mathbf{W}_{\mathbf{X}} \in \mathbb{R}^{n\times p}$ of $\mathbf{X}$

initilization
  Divide $\mathbf{X}$ into k sub-matrices $\mathbf{X}_i$
  Form covariance matrix $\mathbf{R}_{\mathbf{X}_1} = \frac{1}{m}\mathbf{X}_1\mathbf{X}_1^H$
  Extract principal subspace $\mathbf{W}_1 = \text{eig}(\mathbf{R}_{\mathbf{X}_1}, p)$
  Construct matrix $\mathbf{U}_1 = \mathbf{W}_1^{\#}\mathbf{X}_1$
main estimate PSA : *// updates can be done in parallel*
  for $i = 2 \to k$ **do**
    Form covariance matrix $\mathbf{R}_{\mathbf{X}_i} = \frac{1}{m}\mathbf{X}_i\mathbf{X}_i^H$
    Extract principal subspace $\mathbf{W}_i = \text{eig}(\mathbf{R}_{\mathbf{X}_i}, p)$
    Construct matrix $\mathbf{U}_i = \mathbf{W}_i^{\#}\mathbf{X}_i$
    Construct rotation $\mathbf{T}_i = \mathbf{U}_i\mathbf{U}_1^{\#}$
    Update $\mathbf{W}_i \leftarrow \mathbf{W}_i\mathbf{T}_i$
return $\mathbf{W}_{\mathbf{X}} = [\mathbf{W}_1^T\ \mathbf{W}_2^T\ \ldots\ \mathbf{W}_k^T]^T$

2.3 Principal Subspace Analysis based on GMNS

Consider a low rank matrix $\mathbf{X} = \mathbf{AS} \in \mathbb{C}^{n\times m}$ under the conditions that $\mathbf{A} \in \mathbb{C}^{n\times p}, \mathbf{S} \in \mathbb{C}^{p\times m}$ with $p < \min(n, m)$, and $\mathbf{A}$ is full column rank.

Under the constraint of having only a fixed number k of digital signal processing (DSP) units, the procedure of GMNS for PSA includes: dividing the matrix $\mathbf{X}$ into k sub-matrices $\{\mathbf{X}_1, \mathbf{X}_2, \ldots, \mathbf{X}_k\}$, then estimating each principal subspace matrix $\mathbf{W}_i = \mathbf{A}_i\mathbf{Q}_i$ of $\mathbf{X}_i$, and finally combining them to obtain the principal matrix of $\mathbf{X}$. Clearly, we should choose a number of DSP units so that the size of resulting sub-matrices $\mathbf{X}_i$ must be larger than rank of $\mathbf{X}$, $p \leq n/k$. The algorithm was proposed in [19], summarized in Algorithm 2.

First, the principal subspace matrix $\mathbf{W}_i$ of $\mathbf{X}_i$ can be obtained from the eigenspace

of its corresponding covariance matrix

$$\mathbf{R}_{\mathbf{X}_i} = \mathrm{E}\{\mathbf{X}_i\mathbf{X}_i^H\} = \mathbf{A}_i\mathbf{R}_{\mathbf{S}}\mathbf{A}_i^H \overset{\text{EVD}}{=} \mathbf{W}_i\mathbf{\Lambda}\mathbf{W}_i^H, \tag{2.6}$$

where $\mathbf{W}_i = \mathbf{A}_i\mathbf{Q}_i$ with $\mathbf{Q}_i \in \mathbf{R}^{p\times p}$ is an unknown full rank matrix.

Given the directions of $\mathbf{X}_1$, we look for $(k-1)$ rotation matrices $\mathbf{T}_i$ to align the principal axes of each $\mathbf{X}_i$ with these directions of $\mathbf{X}_1$. Specifically, let

$$\mathbf{U}_i = \mathbf{W}_i^{\#}\mathbf{X}_i, \tag{2.7}$$

$$\mathbf{U}_i = (\mathbf{A}_i\mathbf{Q}_i)^{\#}\mathbf{A}_i\mathbf{S} = \mathbf{Q}_i^{-1}\mathbf{S}. \tag{2.8}$$

On the other hand, combining with (2.6), the signal subspace can be determined by

$$\mathbf{W} = \mathbf{A}\mathbf{Q} = \begin{bmatrix} \mathbf{A}_1\mathbf{Q} \\ \mathbf{A}_2\mathbf{Q} \\ \vdots \\ \mathbf{A}_k\mathbf{Q} \end{bmatrix} = \begin{bmatrix} \mathbf{A}_1\mathbf{Q}_1\mathbf{Q}_1^{-1}\mathbf{Q} \\ \mathbf{A}_2\mathbf{Q}_2\mathbf{Q}_2^{-1}\mathbf{Q} \\ \vdots \\ \mathbf{A}_k\mathbf{Q}_k\mathbf{Q}_k^{-1}\mathbf{Q} \end{bmatrix} = \begin{bmatrix} \mathbf{W}_1\mathbf{T}_1 \\ \mathbf{W}_2\mathbf{T}_2 \\ \vdots \\ \mathbf{W}_k\mathbf{T}_k \end{bmatrix}.$$

It then yields rotation $\mathbf{T}_i$ that can be computed by $\mathbf{T}_i = \mathbf{Q}_i^{-1}\mathbf{Q}_1$. Thus, $\mathbf{T}_i$ can be estimated without knowing $\mathbf{Q}_1$, as

$$\mathbf{T}_i = \mathbf{Q}_i^{-1}\mathbf{Q}_1 = \mathbf{Q}_i^{-1}\mathbf{S}\mathbf{S}^{\#}\mathbf{Q}_1 = \mathbf{Q}_i^{-1}\mathbf{S}(\mathbf{Q}_1^{-1}\mathbf{S})^{\#} = \mathbf{U}_i\mathbf{U}_1^{\#}.$$

where $\mathbf{U}_i$ can be easily computed, as in (2.7).

As a result, the principal subspace matrix of $\mathbf{X}$ can be updated as

$$\mathbf{W} = \left[\mathbf{W}_1^T\ (\mathbf{W}_2\mathbf{T}_2)^T \dots (\mathbf{W}_k\mathbf{T}_k)^T\right]^T = \mathbf{A}\mathbf{Q}_1.$$

Chapter 3

Proposed Modified and Randomized GMNS based PSA Algorithms

In this chapter, we introduce two modifications to the GMNS for PSA. In particular, by expressing the right singular vectors obtained from SVD in terms of principal subspace, we derive a modified GMNS algorithm for PSA with running time faster than the original GMNS, while still retaining the subspace estimation accuracy. In addition, we introduce a randomized GMNS algorithm for PSA that can deal with several matrices by performing the randomized SVD [27].

3.1 Modified GMNS-based Algorithm

Consider again the low rank data matrix $\mathbf{X} = \mathbf{AS} \in \mathbb{R}^{n\times m}$ for measurement, as mentioned in Section 2.3. We first look at the true principal subspace matrix $\mathbf{W}_{\mathbf{X}}$, which is obtained via SVD of $\mathbf{X}$, that is,

$$\mathbf{X} \overset{\text{SVD}}{=} \mathbf{W}\mathbf{\Sigma}\mathbf{V}^H = \mathbf{W}_{\mathbf{X}}\mathbf{U}_{\mathbf{X}},$$

where $\mathbf{W}_{\mathbf{X}}$ and $\mathbf{U}_{\mathbf{X}}$ present the left singular vectors and the right singular vectors of $\mathbf{X}$ respectively.

It is therefore that the column space of $\mathbf{A}$ is exactly the column space of $\mathbf{W}_{\mathbf{X}}$. In

Algorithm 3: Proposed modified GMNS-based PSA

Input: Matrix $\mathbf{X} \in \mathbb{R}^{n \times m}$, target rank p, k DSP units
Output: Principal subspace matrix $\mathbf{W}$ of $\mathbf{X}$

function
  Divide $\mathbf{X}$ into k sub-matrices: $\mathbf{X}_1, \mathbf{X}_2, \ldots, \mathbf{X}_k$
  Compute SVD of $\mathbf{X}_1$ to obtain $[\mathbf{W}_1, \mathbf{U}_1] = \text{svd}(\mathbf{X}_1)$
  // updates can be done in parallel
  for $i = 2 \rightarrow k$ **do**
    Compute $\mathbf{W}_i = \mathbf{X}_i \mathbf{U}_1^{\#}$
return $\mathbf{W}_\mathbf{X} = [\mathbf{W}_1^T \ \mathbf{W}_2^T \ \ldots \ \mathbf{W}_K^T]^T$

particular, $\mathbf{X}$ can be expressed by

$$\mathbf{X} = \mathbf{AS} = \mathbf{AQQ}^{-1}\mathbf{S},$$

where $\mathbf{Q}$ is an unknown full rank matrix such that

$$\mathbf{W}_\mathbf{X} = \mathbf{AQ},$$

$$\mathbf{U}_\mathbf{X} = \mathbf{Q}^{-1}\mathbf{S}.$$

From GMNS, when splitting the original matrix $\mathbf{X}$ into $\mathbf{X}_1, \ldots, \mathbf{X}_k$ sub-matrices, suppose that the principal subspace matrix of each sub-matrix $\mathbf{X}_i$ can be determined as

$$\mathbf{X}_i \overset{\text{SVD}}{=} \mathbf{W}_{\mathbf{X}_i} \mathbf{U}_{\mathbf{X}_i},$$

where $\mathbf{W}_{\mathbf{X}_i} = \mathbf{A}_i\mathbf{Q}_i$ and $\mathbf{U}_{\mathbf{X}_i} = \mathbf{Q}_i^{-1}\mathbf{S}$. We now obtain the following property:

$$\begin{aligned}\mathbf{X} &= \begin{bmatrix}\mathbf{A}_1\mathbf{S}\\ \mathbf{A}_2\mathbf{S}\\ \vdots\\ \mathbf{A}_k\mathbf{S}\end{bmatrix} = \begin{bmatrix}\mathbf{W}_{\mathbf{X}_1}\mathbf{Q}_1^{-1}\\ \mathbf{W}_{\mathbf{X}_2}\mathbf{Q}_2^{-1}\\ \vdots\\ \mathbf{W}_{\mathbf{X}_k}\mathbf{Q}_k^{-1}\end{bmatrix}\mathbf{S}\\ &= \begin{bmatrix}\mathbf{W}_{\mathbf{X}_1}\\ \mathbf{W}_{\mathbf{X}_2}\mathbf{Q}_2^{-1}\mathbf{Q}_1\\ \vdots\\ \mathbf{W}_{\mathbf{X}_k}\mathbf{Q}_k^{-1}\mathbf{Q}_1\end{bmatrix}\mathbf{Q}_1^{-1}\mathbf{S} = \begin{bmatrix}\mathbf{W}_{\mathbf{X}_1}\\ \mathbf{W}_2\\ \vdots\\ \mathbf{W}_k\end{bmatrix}\mathbf{U}_{\mathbf{X}_1}. \end{aligned} \tag{3.1}$$

Hence, the relationship between the sub-matrices $\mathbf{X}_i$ and their corresponding subspace matrices can be given by

$$\mathbf{X}_i = \mathbf{W}_i\mathbf{U}_{\mathbf{X}_1},$$

$$\mathbf{W}_i = \mathbf{X}_i\mathbf{U}_{\mathbf{X}_1}^{\#}.$$

As a result, we derive a new implementation for performing the GMNS algorithm. First, performing SVD of $\mathbf{X}_1$ to obtain

$$\mathbf{X}_1 \overset{\text{SVD}}{=} \mathbf{W}_1\mathbf{U}_1,$$

where $\mathbf{W}_1$ is the left singular vector matrix of $\mathbf{X}_1$ and $\mathbf{U}_1 = \mathbf{\Sigma}_1\mathbf{V}_1$ is to present its right singular vector matrix.

Next, the principal subspace matrices of other sub-matrices $\mathbf{X}_i$, $i = 2, \ldots, k$, can be obtained by projecting these sub-matrices onto the pseudo-inverse right singular vector

matrix of $\mathbf{X}_1$, that is,

$$\mathbf{W}_i = \mathbf{X}_i \mathbf{U}_1^{\#}.$$

Finally, the principal subspace matrix of $\mathbf{X}$ is obtained by concatenating the principal subspace matrices of $\mathbf{X}_i$ as

$$\mathbf{W} = [\mathbf{W}_1^T \ \mathbf{W}_2^T \ \dots \mathbf{W}_k^T]^T,$$

$$\mathbf{X} = \mathbf{W}\mathbf{U}_1.$$

The modified GMNS algorithm for PSA can be summarized in Algorithm 3.

3.2 Randomized GMNS-based Algorithm

Although the original GMNS method in [19] provides an efficient tool for fast subspace estimation with high accuracy, it is only useful for the type of low rank matrices addressed in Section 2.3. This motivates us to look for an improvement on GMNS that can deal with *arbitrary* matrices.

In order to apply GMNS, we here want to produce a good approximation $\widehat{\mathbf{X}} = \mathbf{YZ}$ of the given matrix $\mathbf{X}$ that not only satisfies the required conditions of GMNS, but also must cover the span and preserve important properties of $\mathbf{X}$. Therefore, the matrix $\mathbf{Y} = \mathbf{X}\boldsymbol{\Omega}$ can be a good sketch of $\mathbf{X}$ where $\boldsymbol{\Omega}$ is a sketching matrix like a column selection or random projection matrix. Several studies have been proposed to solve the problem so far; for example, we can apply randomized algorithms and sketching techniques in [27–29] for matrices and data to estimate $\mathbf{Y}$, hence $\mathbf{Z}$.

In this work, we are interested in investigating Gaussian $\mathbf{\Omega}$, with entries being i.i.d. samples from $\mathcal{N}(0,1)$. The Gaussian random matrix has been successfully applied in several matrix analysis methods, such as [27, 30, 31]. It is noted that the Gaussian random matrix has many desired properties, such as the following:

- For all vector $\mathbf{x}$ in the row space of $\mathbf{X}$, its length will not change much if sketching by $\mathbf{\Omega}$: $\|\mathbf{x}\|^2 \approx \|\mathbf{x}\mathbf{\Omega}\|^2$;
- In general, random vectors of $\mathbf{\Omega}$ are likely to be linear position and linearly independent;
- There is no linear combination falling in the null space of $\mathbf{X}$.

As a result, $\mathbf{Y} = \mathbf{X}\mathbf{\Omega}$ is a high quality sketch and can span the range of $\mathbf{X}$.

After finding a good sketch $\mathbf{Y}$ from the Gaussian random matrix $\mathbf{\Omega}$, the next problem is considered as a low rank matrix approximation such that its result has to hold the Frobenius norm error bound with high probability. This leads to the following optimization problem

$$\begin{aligned} \min_{\text{rank}(\mathbf{Z})\leq k} \|\mathbf{X} - \mathbf{Y}\mathbf{Z}\|_F^2 &\leq (1+\epsilon)\|\mathbf{X} - \mathbf{X}_k\|_F^2 \\ &= (1+\epsilon)\sum_{i=k+1}^{N} \sigma_i(\mathbf{X}), \end{aligned} \tag{3.2}$$

where $\sigma_j(\mathbf{X})$ is the j-th singular value of $\mathbf{X}$ and $\mathbf{X}_k$ is the best rank-k approximate of $\mathbf{X}$.

Let $\mathbf{Q_Y}$ contain orthogonal bases of the sketch $\mathbf{Y}$ of $\mathbf{X}$. Clearly, since $\mathbf{Q_Y}$ shares the same the column space with $\mathbf{Y}$, the optimization problem of (3.2) can be rewritten

as

$$\mathbf{Z}^\star = \underset{\text{rank}(\mathbf{Z})\leq k}{\arg\min} \ \|\mathbf{X} - \mathbf{Q}_\mathbf{Y}\mathbf{Z}\|_F^2. \tag{3.3}$$

The solution of (3.3) can be computed more easily as

$$\mathbf{Z}^\star = \mathbf{Q}_\mathbf{Y}^T\mathbf{X}. \tag{3.4}$$

Therefore, with $\mathbf{Q}_\mathbf{Y}$, the Frobenius norm error in the problem (3.2) can be extended to a stronger error measure, that is, the spectral norm error bound (we refer the reader to [29, Section 4.3] for further details), as follows:

$$\|\mathbf{X} - \mathbf{Q}_\mathbf{Y}\mathbf{Q}_\mathbf{Y}^T\mathbf{X}\|_2^2 \leq (1+\epsilon)\|\mathbf{X} - \mathbf{X}_k\|_2^2 = (1+\epsilon)\sigma_{k+1}(\mathbf{X}).$$

From now, we have an approximate basis for the range of $\mathbf{X}$, that is,

$$\mathbf{X} \approx \mathbf{Q}_\mathbf{Y}\mathbf{Q}_\mathbf{Y}^T\mathbf{X}.$$

Let us define $\bar{\mathbf{A}} = \mathbf{Q}_\mathbf{Y}^T\mathbf{X}$, we then have

$$\mathbf{X} \approx \mathbf{Q}_\mathbf{Y}\bar{\mathbf{A}}.$$

Accordingly, the principal subspace matrix $\mathbf{W}_{\bar{\mathbf{A}}}$ of $\bar{\mathbf{A}}$ can be computed by using the original GMNS or the modified GMNS proposed in Section 3.1. Then we can estimate the principal subspace of an arbitrary matrix $\mathbf{X}$ by

$$\mathbf{W}_\mathbf{X} \approx \mathbf{Q}_\mathbf{Y}\mathbf{W}_{\bar{\mathbf{A}}}.$$

This randomized GMNS algorithm for PSA can be summarized in Algorithm 4.

Algorithm 4: Proposed randomized GMNS-based PSA

Input: Matrix $\mathbf{X} \in \mathbb{R}^{n \times m}$, target rank p, k DSP units
Output: Principal subspace matrix $\mathbf{W}$ of $\mathbf{X}$.

function
- Draw a Gaussian random matrix $\mathbf{\Omega} \in \mathbb{R}^{m \times l}$, $l > p$
- Form the sketch $\mathbf{Y} \in \mathbb{R}^{n \times l}$ of $\mathbf{X}$: $\mathbf{Y} = \mathbf{X\Omega}$
- Extract principal subspace $\mathbf{Q}$ from $\mathbf{Y}$ using QR decomposition
- Construct $\bar{\mathbf{A}} = \mathbf{Q}^T\mathbf{X}$
- Estimate $\mathbf{W}_{\bar{\mathbf{A}}}$ of $\bar{\mathbf{A}}$ using GMNS

return $\mathbf{W}_{\mathbf{X}} = \mathbf{Q}\mathbf{W}_{\bar{\mathbf{A}}}$

Remark

Recall that GMNS is with a parallelized computing architecture in practice. Therefore estimating the orthogonal basis of the sketch $\mathbf{Y}$ based on the QR decomposition should be implemented in a parallelization scheme. In this work, we can parallelize the randomized GMNS algorithm by using a distributed QR decomposition, namely TSQR [32].

In particular, we divide $\mathbf{X}$ into k sub-matrices $\mathbf{X}_i$ in the similar way to the original GMNS and the modified GMNS algorithms. First, we find all the sketch $\mathbf{Y}_i$ of sub-matrices $\mathbf{X}_i$ under the sketching $\mathbf{\Omega}$. Next, we perform standard QR decomposition on each sub-matrix $\mathbf{Y}_i$ to obtain $\mathbf{Q}_{1,i}$ and $\mathbf{R}_{1,i}$. Then, the resulting matrices $\mathbf{R}_{1,i}$ are gathered into a single matrix $\mathbf{R}_1$ which is then decomposed into $\mathbf{Q}_{2,:}$ again. As a result, the original factor $\mathbf{Q}$ of $\mathbf{Y}$ can be obtained from multiplying the resulting the $\mathbf{Q}_{1,:}$ with $\mathbf{Q}_{2,:}$ which can be already distributed among the DSP units. Finally, we find the orthogonal basis of the sketch $\bar{\mathbf{A}} = \mathbf{Q}^T\mathbf{Y}$ by using the original GMNS or modified GMNS algorithms, and hence the principal subspace matrix of $\mathbf{X}$. We refer the reader to [32] for further details.

3.3 Computational Complexity

For the sake of simplicity, we assume that standard algorithms for computing matrix multiplication and matrix decompositions (like EVD, SVD, QR) are applied in this work, while costs of transfer and synchronization between the DSP units are ignored. In particular, to decompose a rank-p matrix of size $n \times n$ into factors, the standard EVD requires a cost of $\mathcal{O}(n^2 p)$. Considering a non-square matrix of size $n \times m$, the full Householder QR algorithm is computed in $2nm^2 - 2/3m^3$ flops, while the truncated SVD typically needs nmp flops to derive a rank-p approximation by using the partial QR decomposition. These methods are surveyed in [33]. To multiply a matrix $\mathbf{A}$ of size $n \times p$ with a matrix $\mathbf{B}$ of size $p \times m$, we consider the standard algorithm which is to perform n dot products of rows in $\mathbf{A}$ and columns in $\mathbf{B}$ whose cost is $\mathcal{O}(nmp)$.

Now, we analyse the computational complexity of the modified and randomized GMNS algorithms for PSA. The former consists of two main operations: (i) the truncated SVD of $\mathbf{X}_1$ that is performed in nmp/k flops, and (ii) $(k-1)$ matrix products of sub-matrices $\mathbf{X}_i$ with the right-singular vector matrix of $\mathbf{X}_1$ that requires nmp/k flops. Therefore, the overall complexity is order of $\mathcal{O}(nmp/k)$. Meanwhile, the computational complexity of original GMNS for PSA is order of $\mathcal{O}(n^2(m+p)/k^2)$. Since $m, n \gg p$, the original and the modified GMNS algorithms have lower complexity than that of the well-known method using EVD of the global covariance matrix that costs $\mathcal{O}\left(n^2(m+p)\right)$ flops.

The randomized GMNS algorithm consists of three main operations: (i) estimating a good sketch $\mathbf{Y}$ of $\mathbf{X}$, (ii) orthonormalizing the columns of $\mathbf{Y}$, and (iii) updating its

subspace matrix. In the first operation, deriving a standard Gaussian matrix $\mathbf{\Omega} \in \mathbb{R}^{m \times l}$ and hence a good sketch $\mathbf{Y} \in \mathbb{R}^{n \times l}$ demand a cost of $\mathcal{O}(mnl)$. In the second operation, QR decomposition, used to compute the orthogonal basis of $\mathbf{Y}$, demands a cost of $2nl^2 - 2/3l^3$ flops. In the last operation, two matrix products are used to compute the matrix $\bar{\mathbf{A}}$ and update $\mathbf{W}_{\mathbf{X}}$, demanding a cost of $nl(m+p)$ flops. In addition, the algorithm uses the same order of complexity for estimating the subspace of $\bar{\mathbf{A}}$ using GMNS. Moreover, we can use the structured random matrix $\mathbf{\Omega}$ using the subsampled random FFT instead, to reduce the overall complexity. Specifically, it allows us to compute the product of $\mathbf{X}$ and $\mathbf{\Omega}$ in $nm\log(l)$ flops; and the row-extraction technique to derive $\mathbf{Q}$ with a lower cost of $\mathcal{O}(k^2(n+m))$. We refer the reader to [27, Section 4.6] for further details. In conclusion, the overall complexity of the randomized GMNS algorithm is $\mathcal{O}(nl(m+p)/k)$ using the TSQR algorithm.

Chapter 4

Proposed GMNS-based Tensor Decomposition

4.1 Proposed GMNS-based PARAFAC

In this section, we derive a new implementation based on GMNS approach for performing PARAFAC of three-way tensors.

Considering a three-way tensor $\boldsymbol{\mathcal{X}} \in \mathbb{R}^{I \times J \times K}$, PARAFAC of $\boldsymbol{\mathcal{X}}$ can be expressed as follows:

$$\boldsymbol{\mathcal{X}} = \boldsymbol{\mathcal{I}} \times_1 \mathbf{A} \times_2 \mathbf{B} \times_3 \mathbf{C} = \sum_{i=1}^{R} \mathbf{a}_i \circ \mathbf{b}_i \circ \mathbf{c}_i, \tag{4.1}$$

where R is the rank of the tensor, $\boldsymbol{\mathcal{I}}$ is an identity tensor, $\mathbf{A} \in \mathbb{R}^{I \times R}$, $\mathbf{B} \in \mathbb{R}^{J \times R}$ and $\mathbf{C} \in \mathbb{R}^{K \times R}$ are the factors (loading matrices).

Motivated by the advantages of GMNS and the ALS-based PARAFAC in Section 2.2, we are interested in investigating a parallelization scheme for PARAFAC. The proposed algorithm consists of four steps:

- Step 1: Divide tensor $\boldsymbol{\mathcal{X}}$ into k sub-tensors $\boldsymbol{\mathcal{X}}_1, \boldsymbol{\mathcal{X}}_2, \ldots, \boldsymbol{\mathcal{X}}_k$;
- Step 2: Estimate the principal subspace matrix of each tensors: $\mathbf{W}_i = (\mathbf{C}_i \odot \mathbf{A}_i)\mathbf{Q}_i$ using GMNS;
- Step 3: Obtain the loading matrices $\mathbf{A}, \mathbf{Q}$ and $\mathbf{B}$, thanks to some desired property

of GMNS;

- Step 4: Update the loading matrix $\mathbf{C}$.

The main difference between the GMNS-based and ALS-based PARAFAC algorithms is in the way we compute factors $\mathbf{A}_i, \mathbf{B}_i$ and $\mathbf{C}_i$ of each sub-tensor $\boldsymbol{\mathcal{X}}_i$. Specifically, instead of applying ALS for k sub-tensors, these factors can be obtained directly from the principal subspace of each of the sub-tensors $\boldsymbol{\mathcal{X}}_i, i = 2, 3, \ldots, k$. Therefore, we need to apply ALS for only the first sub-tensor $\boldsymbol{\mathcal{X}}_1$. Now, we will describe the algorithm in details.

For the sake of simplicity, we assume that the given tensor $\boldsymbol{\mathcal{X}}$ is divided into k sub-tensors $\boldsymbol{\mathcal{X}}_1, \boldsymbol{\mathcal{X}}_2, \ldots, \boldsymbol{\mathcal{X}}_k$ by splitting the loading matrix $\mathbf{C}$ in the similar way to ALS-based PARAFAC. The corresponding matrix representation of sub-tensors and their subspace matrices are also given by

$$\begin{aligned} \mathbf{X}_i &= (\mathbf{C}_i \odot \mathbf{A})\mathbf{B}^T, \\ \mathbf{W}_i &= (\mathbf{C}_i \odot \mathbf{A})\mathbf{Q}_i, \end{aligned}$$

where $\mathbf{Q}_i \in \mathbb{R}^{R \times R}$ is a full rank matrix.

First, using any specific PARAFAC algorithm, such as the ALS-based PARAFAC, to compute the factors $\mathbf{A}_1$, $\mathbf{B}_1$, and $\mathbf{C}_1$ of $\boldsymbol{\mathcal{X}}_1$, from

$$\mathbf{X}_1 = (\mathbf{C}_1 \odot \mathbf{A}_1)\mathbf{B}_1^T,$$

we obtain the two factors $\mathbf{A} \leftarrow \mathbf{A}_1$ and $\mathbf{B} \leftarrow \mathbf{B}_1$. In addition, the principal subspace

matrix $\mathbf{W}_1$ of $\mathbf{X}_1$ can also be given as

$$\mathbf{W}_1 = (\mathbf{C}_1 \odot \mathbf{A}_1)\mathbf{Q}_1.$$

Therefore, the two rotation matrices $\mathbf{Q}_1$ and $\mathbf{U}_1$ can be obtained as

$$\mathbf{Q}_1 = (\mathbf{C}_1 \odot \mathbf{A}_1)^{\#}\mathbf{W}_1, \tag{4.2a}$$

$$\mathbf{U}_1 = \mathbf{W}_1^{\#}\mathbf{X}_1. \tag{4.2b}$$

From now, the factors of $\boldsymbol{\mathcal{X}}_i$, $i = 2, \ldots, k$, can be derived directly from their principal subspace matrices $\mathbf{W}_i$ of $\mathbf{X}_i$, as

$$\mathbf{W}_i = \mathbf{X}_i\mathbf{U}_1^{\#}, \tag{4.3a}$$

$$\mathbf{C}_i \odot \mathbf{A}_i = \mathbf{W}_i\mathbf{Q}_1^{-1}. \tag{4.3b}$$

The loading matrix $\mathbf{A}_i$ and $\mathbf{C}_i$ are then be easily recovered, thanks to the Khatri-Rao product. In parallel, the loading matrix $\mathbf{B}_i$ can be updated as follows:

$$\mathbf{B}_i = \mathbf{X}_i^T(\mathbf{W}_i^{\#})^T\mathbf{Q}_1^T. \tag{4.4}$$

The next step is to rotate the loading matrix $\mathbf{A}_i, \mathbf{B}_i$ and $\mathbf{C}_i$ according to (2.4). The factors of the overall PARAFAC are then obtained as

$$\begin{aligned} &\mathbf{A} \leftarrow \mathbf{A}_1, \mathbf{B} \leftarrow \mathbf{B}_1, \\ &\mathbf{C} \leftarrow \left[\mathbf{C}_1^T\ \mathbf{C}_2^T\ \ldots\ \mathbf{C}_k^T\right]^T. \end{aligned} \tag{4.5}$$

The proposed GMNS-based PARAFAC algorithm is summarized in Algorithm 5.

Algorithm 5: Proposed GMNS-based PARAFAC

Input: Tensor $\boldsymbol{\mathcal{X}} \in \mathbb{R}^{I \times J \times K}$, target rank R, k DSP units

Output: Factors $\mathbf{A} \in \mathbb{R}^{I \times R}, \mathbf{B} \in \mathbb{R}^{J \times R}$ and $\mathbf{C} \in \mathbb{R}^{K \times R}$

Initilization

- Divide $\boldsymbol{\mathcal{X}}$ into k sub-tensors $\boldsymbol{\mathcal{X}}_1, \boldsymbol{\mathcal{X}}_2, \ldots, \boldsymbol{\mathcal{X}}_k$
- Apply ALS to compute $\mathbf{A}_1, \mathbf{B}_1$ and $\mathbf{C}_1$ of $\boldsymbol{\mathcal{X}}_1$
- Extract principal subspace $\mathbf{W}_1$ of $\mathbf{X}_1$ using SVD
- Compute rotation matrix $\mathbf{Q}_1 = (\mathbf{C}_1 \odot \mathbf{A}_1)^{\#}\mathbf{W}_1$

Main Update factors of other sub-tensors

// updates can be done in parallel

- **for** $i = 2 \to k$ **do**
  - Extract principal subspace $\mathbf{W}_i$ of $\mathbf{X}_i$ using SVD Compute $\mathbf{C}_i$ and $\mathbf{A}_i$ // (4.3)
  - Compute $\mathbf{B}_i$ // (4.4)
  - Rotate $\mathbf{A}_i, \mathbf{C}_i$ and $\mathbf{B}_i$ // (2.4)

return $\mathbf{A}, \mathbf{B}, \mathbf{C}$ // (4.5)

Remark

In the case of tensors with $K \gtrapprox I \times J$, the GMNS-based PARAFAC algorithm can be implemented more efficiently. Matrix representation of the overall tensor and its sub-tensors can be expressed, respectively, as

$$\mathbf{X} = \mathbf{C}(\mathbf{B} \odot \mathbf{A})^T \text{ and } \mathbf{X}_i = \mathbf{C}_i(\mathbf{B} \odot \mathbf{A})^T. \tag{4.6}$$

Therefore, the factors can be computed more easily. Specifically, the principal subspace of $\mathbf{X}_i$ can be given by

$$\mathbf{W}_i = \mathbf{C}_i\mathbf{Q}_i.$$

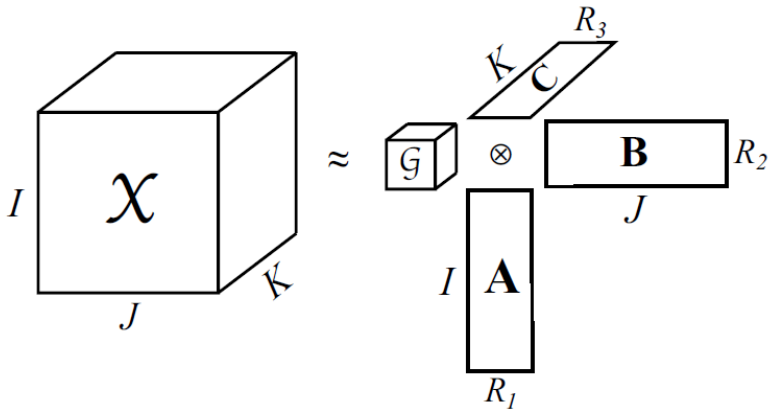


Figure 4.1: Higher-order singular value decomposition.

Meanwhile, the rotation matrices are updated in a way similar to the above, as

$$\mathbf{U}_1 = \mathbf{W}_1^{\#}\mathbf{X}_1,$$

$$\mathbf{Q}_1 = \mathbf{C}_1^{\#}\mathbf{W}_1.$$

Therefore, the sub-factors $\mathbf{C}_i$ are obtained by

$$\mathbf{W}_i = \mathbf{X}_i\mathbf{U}_1^{\#},$$

$$\mathbf{C}_i = \mathbf{W}_i\mathbf{Q}_1^{-1}.$$

As a result, the loading matrix $\mathbf{C}$ is updated while $\mathbf{A}$ and $\mathbf{B}$ are computed from $\mathbf{X}_1$.

4.2 Proposed GMNS-based HOSVD

In this section, we investigate a parallelization scheme for HOSVD of three-way tensors based on GMNS.

Let us consider a three-way tensor $\boldsymbol{\mathcal{X}} \in \mathbb{R}^{I\times J\times K}$. Tucker decomposition of $\boldsymbol{\mathcal{X}}$ can

be expressed as

$$\begin{aligned}\boldsymbol{\mathcal{X}} &= \boldsymbol{\mathcal{G}} \times_1 \mathbf{A} \times_2 \mathbf{B} \times_3 \mathbf{C} \\ &= \sum_{i=1}^{R_1}\sum_{j=1}^{R_2}\sum_{k=1}^{R_3} \boldsymbol{\mathcal{G}}_{i,j,k}\mathbf{a}_i \circ \mathbf{b}_j \circ \mathbf{c}_k,\end{aligned}$$

where $\mathbf{A} \in \mathbb{R}^{I\times R_1}$, $\mathbf{B} \in \mathbb{R}^{J\times R_2}$ and $\mathbf{C} \in \mathbb{R}^{K\times R_3}$ are loading factors, $\boldsymbol{\mathcal{G}} \in \mathbb{R}^{R_1\times R_2\times R_3}$ is the core of $\boldsymbol{\mathcal{X}}$ with $R_1 \leq I, R_2 \leq J$ and $R_3 \leq K$.

HOSVD, also called Tucker1, is a specific Tucker decomposition with orthogonal factors which are derived from singular vectors of the three matrices unfolding $\boldsymbol{\mathcal{X}}$ according to its three modes. In general, Tucker decomposition is not unique (see [11, Section V] or [12, Section IV]). Fortunately, the subspaces spanned by the factors $\mathbf{A}$, $\mathbf{B}$ and $\mathbf{C}$ are physically unique. It means that these factors can be rotated by any full rank matrix $\mathbf{Q}$. In turn, this multiplies the core tensor by its inverse. It is of interest here that GMNS may be used to find multilinear subspaces of tensors, hence used for HOSVD.

Similar to GMNS-based PARAFAC, tensor $\boldsymbol{\mathcal{X}}$ is divided into k sub-tensors $\boldsymbol{\mathcal{X}}_1$, $\boldsymbol{\mathcal{X}}_2$, $\ldots, \boldsymbol{\mathcal{X}}_k$ whose corresponding matrix representations are

$$\mathbf{X}_i = \mathbf{C}_i\mathbf{G}(\mathbf{B} \otimes \mathbf{A})^T.$$

We exploit the fact that factors are derived from the principal components of three modes. Thus, to estimate subspaces for $\mathbf{A}, \mathbf{B}$ and $\mathbf{C}$, we can apply the method based

on calculating the covariance matrix of the tensor, that is,

$$\begin{aligned}\mathbf{R}_{\mathbf{X}} &= \mathrm{E}\{\mathbf{X}\mathbf{X}^T\} \\ &= \mathrm{E}\{\mathbf{C}\mathbf{G}(\mathbf{B}\otimes\mathbf{A})^T(\mathbf{B}\otimes\mathbf{A})\mathbf{G}^T\mathbf{C}^T\} \\ &= \mathrm{E}\{\mathbf{C}\mathbf{G}\mathbf{G}^T\mathbf{C}^T\} = \mathbf{C}\mathbf{R}_{\mathbf{G}}\mathbf{C}^T \\ &\overset{\text{EVD}}{=} \mathbf{W}\mathbf{\Lambda}\mathbf{W}^T.\end{aligned}$$

It is therefore essential to demonstrate that the principal subspace matrix carries information of these factors, that is,

$$\mathbf{W} = \mathbf{C}\mathbf{Q},$$

where $\mathbf{Q} \in \mathbb{R}^{R_3 \times R_3}$ is unknown full rank matrix.

We can derive all these factors by using the original GMNS algorithm for PSA or the modified and randomized GMNS algorithms proposed in this thesis. In this thesis, we illustrate this by using the proposed modified GMNS algorithm. Specifically, assume that we first obtain the factors $\mathbf{A}_1, \mathbf{B}_1$, and $\mathbf{C}_1$ of the sub-tensor $\boldsymbol{\mathcal{X}}_1$ which can be derived from the original HOSVD, or alternatively the original higher order orthogonal iteration of tensors (HOOI) decomposition.

Then, by using GMNS to estimate the principal subspace matrices of the sub-tensors, we can obtain the decomposition. Specifically,

$$\mathbf{U}_1 = \mathbf{C}_1^{\#}\mathbf{X}_1, \tag{4.7}$$

where the matrix $\mathbf{U}_1$ presents the right singular vectors of $\mathbf{X}_1$. As shown in Section 4.1, we have to rotate sub-factors $\mathbf{C}_i$ to follow the direction of $\mathbf{C}_1$. Instead of computing

Algorithm 6: Proposed GMNS-based HOSVD

Input: Tensor $\boldsymbol{\mathcal{X}} \in \mathbb{R}^{I \times J \times K}$, target rank R, k DSP units

Output: Factors $\mathbf{A} \in \mathbb{R}^{I \times R}$, $\mathbf{B} \in \mathbb{R}^{J \times R}$, $\mathbf{C} \in \mathbb{R}^{K \times R}$

function

- Divide $\boldsymbol{\mathcal{X}}$ into k sub-tensors $\boldsymbol{\mathcal{X}}_1, \boldsymbol{\mathcal{X}}_2, \ldots, \boldsymbol{\mathcal{X}}_k$
- Compute factors of $\boldsymbol{\mathcal{X}}_1$'s using HOSVD $\{\mathbf{A}_1, \mathbf{B}_1, \mathbf{C}_1\} = \text{HOSVD}(\boldsymbol{\mathcal{X}}_1)$
- Compute rotation matrix: $\mathbf{U}_1 = \mathbf{C}_1^{\#}\mathbf{X}_1$
- Update factors using modified GMNS algorithm
- *// updates can be done in parallel*
- **for** $i = 2 \rightarrow k$ **do**
  - Compute $\mathbf{C}_i = \boldsymbol{\mathcal{X}}_i^{(3)}\mathbf{U}_1^{\#}$

return $\mathbf{A} \leftarrow \mathbf{A}_1, \mathbf{B} \leftarrow \mathbf{B}_1$ and $\mathbf{C} \leftarrow [\mathbf{C}_1^T, \mathbf{C}_2^T, \ldots, \mathbf{C}_k^T]^T$

the rotation matrices $\mathbf{T}_i$, we dedicate the work to projecting matrices $\mathbf{X}_i$ into the row space $\mathbf{U}_1$ of $\mathbf{X}_1$, as

$$\mathbf{C}_j = \mathbf{X}_j\mathbf{U}_1^{\#}. \tag{4.8}$$

As a result, the subspace generated by the loading factors $\mathbf{A}_i$ and $\mathbf{B}_i$ remains constant. The overall loading matrices can be updated as

$$\mathbf{A} \leftarrow \mathbf{A}_1, \mathbf{B} \leftarrow \mathbf{B}_1, \tag{4.9a}$$

$$\mathbf{C} \leftarrow [\mathbf{C}_1^T\ \mathbf{C}_2^T\ \ldots\ \mathbf{C}_k^T]^T. \tag{4.9b}$$

The core tensor $\mathbf{G}$ can be also computed as

$$\boldsymbol{\mathcal{G}} = \boldsymbol{\mathcal{X}} \times_1 \mathbf{A}^T \times_2 \mathbf{B}^T \times_3 \mathbf{C}^T. \tag{4.10}$$

The implementation of the proposed GMNS-based HOSVD is summarized in Algorithm 6.

Chapter 5

Results and Discussions

In this chapter, numerical simulation is done to compare the performance of the proposed GMNS-based algorithms for PSA and tensor decomposition with the state-of-the-art methods. Some application-based scenarios are also presented to illustrate the effectiveness of the proposed methods.

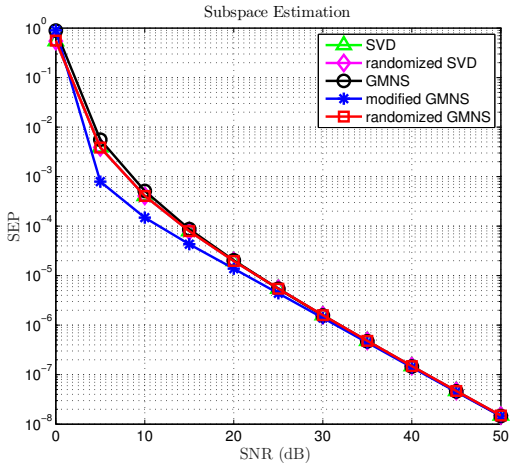
5.1 GMNS-based PSA

We follow experiments and evaluation metrics used in [19]. Specifically, the measurement data $\mathbf{X} = \mathbf{AS}$ are generated by a random system matrix $\mathbf{A}$ and a signal matrix $\mathbf{S}$. The received data are then normalized by its Frobenius norm. The impact of noise on algorithm performance is also investigated by adding noise $\mathbf{N}$ derived from the white Gaussian noise $\mathcal{N}(0, \sigma^2)$, by

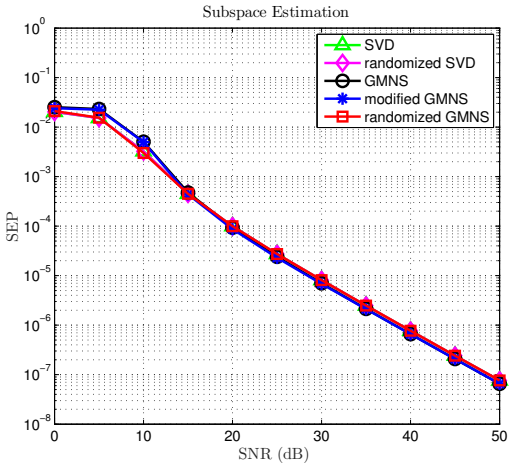
$$\mathbf{X} = \frac{\mathbf{X}}{\|\mathbf{X}\|} + \sigma \frac{\mathbf{N}}{\|\mathbf{N}\|}.$$

The SNR is defined as

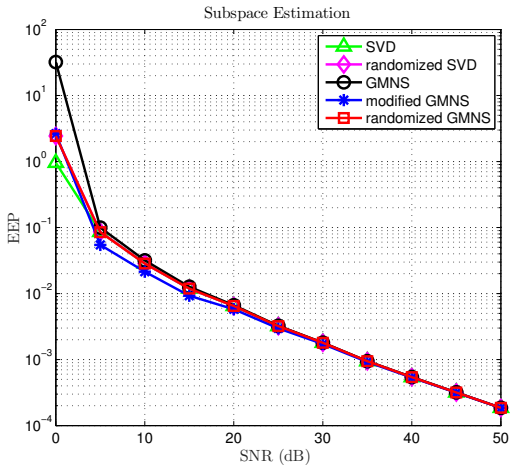
$$\text{SNR} = -10 \log_{10} \sigma^2.$$



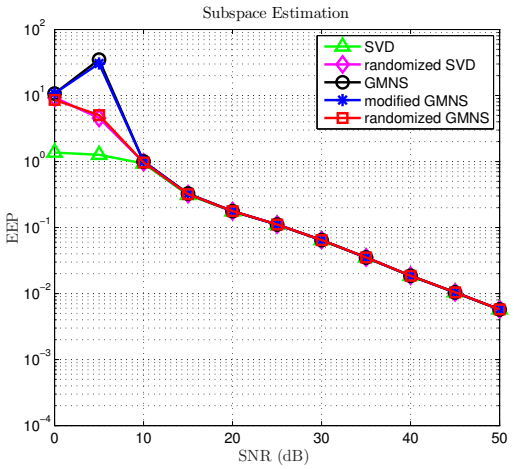
(a) SEP vs. SNR: $p = 2$



(b) SEP vs. SNR: $p = 50$



(c) EEP vs. SNR: $p = 2$



(d) EEP vs. SNR: $p = 50$

Figure 5.1: Effect of number of sources, p, on performance of PSA algorithms; $n = 200$, $m = 500$, $k = 2$.

To evaluate the subspace estimation accuracy, we use the subspace estimation performance (SEP) metric

$$\text{SEP} = \frac{1}{L} \sum_{1}^{L} \frac{\text{tr}\{\mathbf{W}_i^H (\mathbf{I} - \mathbf{W}_{\text{ex}} \mathbf{W}_{\text{ex}}^H) \mathbf{W}_i\}}{\text{tr}\{\mathbf{W}_i^H (\mathbf{W}_{\text{ex}} \mathbf{W}_{\text{ex}}^H) \mathbf{W}_i\}}, \tag{5.1}$$

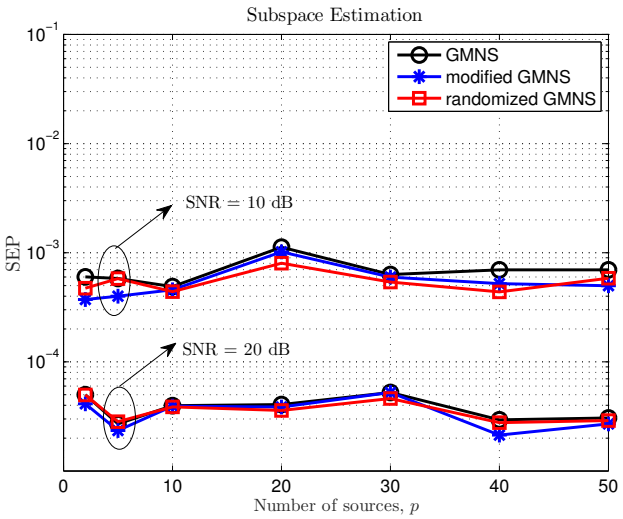


Figure 5.2: Performance of the proposed GMNS algorithms for PSA versus the number of sources p, with $n = 200, m = 500$ and $k = 2$.

and the eigenvector estimation performance (EEP) metric, also referred to as Root Means Square Error,

$$\text{EEP} = \frac{1}{L}\sum_{1}^{L}\|\mathbf{U}_i - \mathbf{U}_{\text{ex}}\|_F^2, \tag{5.2}$$

where L is the number of Monte Carlo runs, $\mathbf{W}_i$ and $\mathbf{U}_i$ denote the estimated subspace and eigenvector matrix at i^{th} run, $\mathbf{W}_{ex}$ and $\mathbf{U}_{ex}$ denote the true subspace and eigenvector matrix, respectively. We note that the lower SEP and EEP are, the better performance algorithms provide.

In this work, we compare and analyze the performance of the state-of-the-art methods for PSA, based on: SVD, randomized SVD [27], original GMNS, modified GMNS and randomized GMNS. The number of Monte Carlo run is fixed at $L = 100$.

5.1.1 Effect of the number of sources, p

We change the number of source p while fixing the number of sensors, number of time observations and number of DSP units at $n = 200$, $m = 500$ and $k = 2$, respectively.

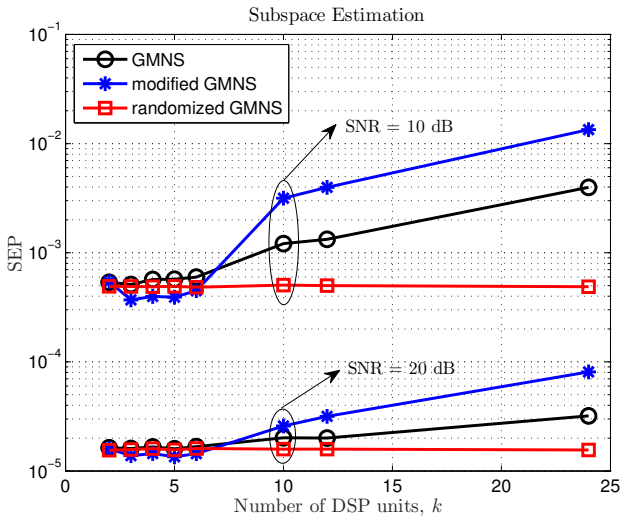
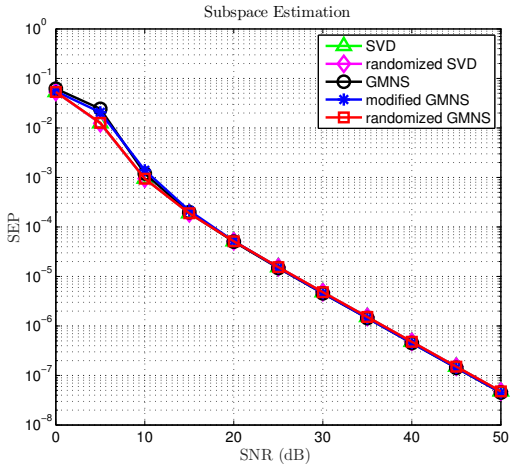


Figure 5.3: Performance of the proposed GMNS algorithms for PSA versus the number of DSP units k, SEP vs. SNR with $n = 240, m = 600$ and $p = 2$.

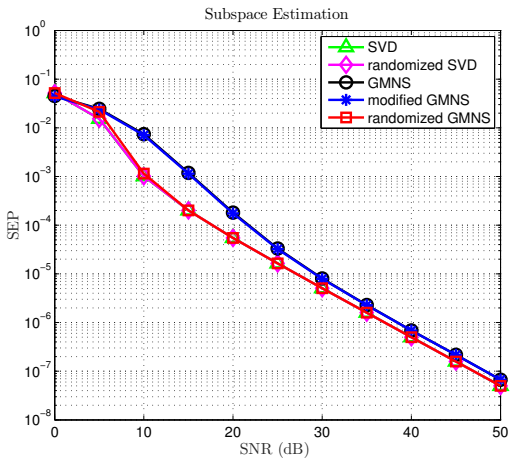
As shown in Figure 5.1, when dealing with a specific p, the modified and randomized GMNS-based algorithms provided similar performance compared to algorithms based on original GMNS, SVD and randomized SVD, in terms of SEP and EEP. In particular, at low SNRs ($\leq$ 10 dB), the SVD-based algorithm yielded the better subspace estimation accuracy than GMNS-based ones, but only slightly. Meanwhile, at high SNRs ($>$ 10 dB), when the impact of noise is reduced, all methods performed the same. As shown in Figure 5.2, there is little difference in subspace estimation accuracy among the original, modified and randomized GMNS-based algorithms when changing the number of sources p, excepting the case of the modified GMNS-based one with small p at SNR = 10 dB. However, the result is still reasonable when compared to the conventional SVD-based algorithms.

5.1.2 Effect of the number of DSP units, k

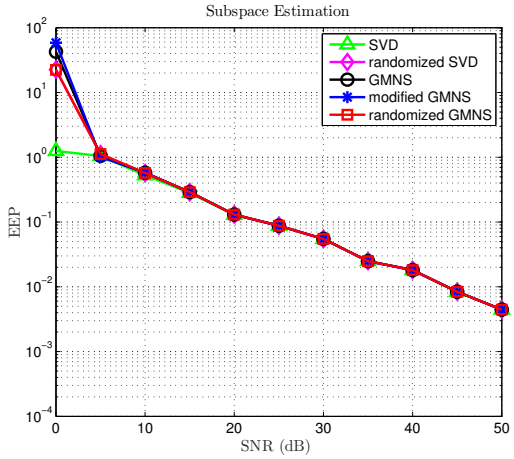
In the similar way, we consider how the number of DSP units affects algorithm performance of the methods by fixing $n = 240$, $m = 600$, and $p = 2$ while changing k.



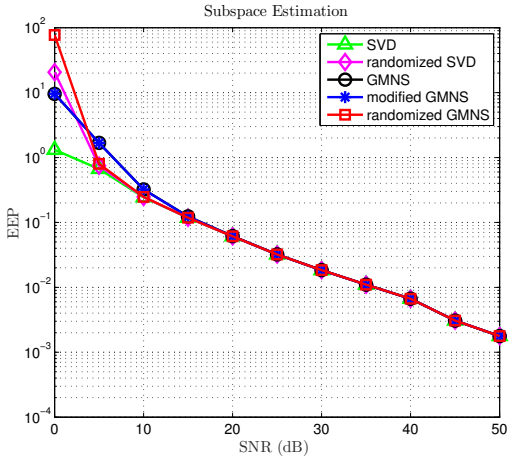
(a) SEP vs. SNR: $k = 2$



(b) SEP vs. SNR: $k = 10$



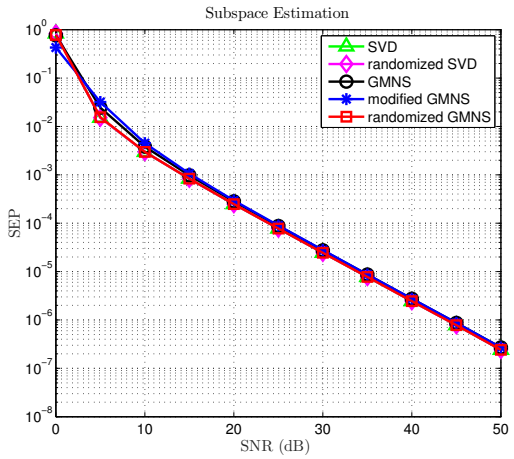
(c) EEP vs. SNR: $k = 2$



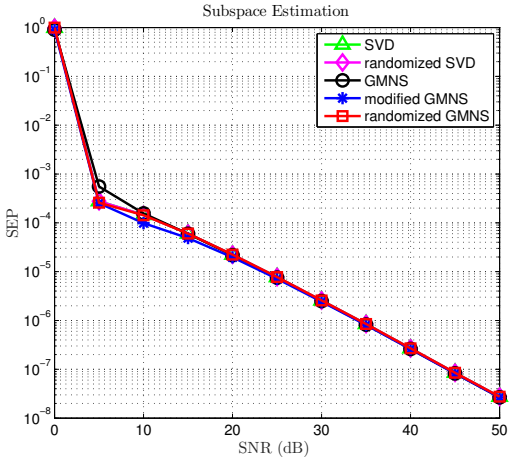
(d) EEP vs. SNR: $k = 10$

Figure 5.4: Effect of number of DSP units, k, on performance of PSA algorithms; $n = 240, m = 600, p = 20$.

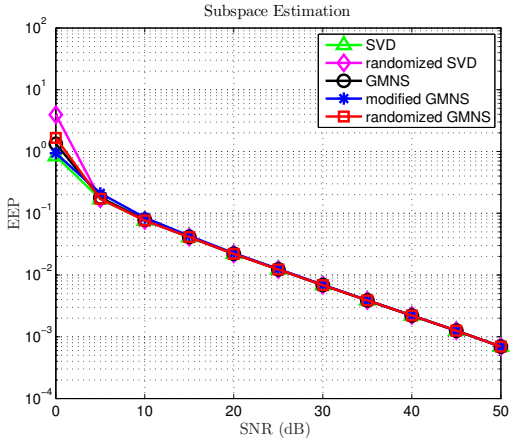
The experimental results indicated that increasing k resulted in slightly reduced SEP. In particular, when the system $\mathbf{A}$ is divided into a small number of sub-systems with $k < 10$, all algorithms provided almost same subspace estimation accuracy, as shown in Figure 5.3. When k is large, the randomized GMNS-based algorithm yielded a same result as the SVD-based and randomized SVD-based algorithms, while a slightly better than that of the original and modified GMNS-based algorithms, as shown in Figure 5.4.



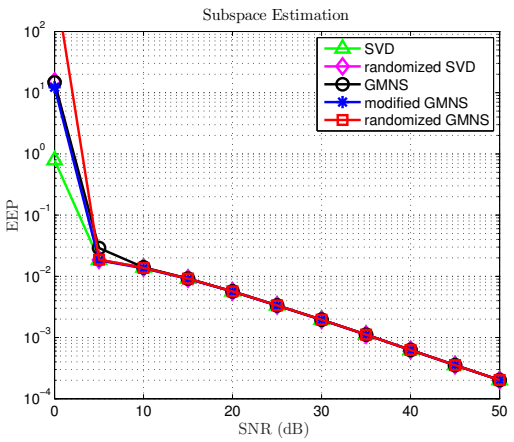
(a) SEP vs. SNR: $n = 50$, $m = 100$



(b) SEP vs. SNR: $n = 200$, $m = 1000$



(c) EEP vs. SNR: $n = 50$, $m = 100$



(d) EEP vs. SNR: $n = 200$, $m = 1000$

Figure 5.5: Effect of matrix size, (m, n), on performance of PSA algorithms; $p = 2$, $k = 2$.

5.1.3 Effect of number of sensors, n, and time observations, m

We fixed the number of DSP units and sources at $k = 2, p = 2$ and varied the size of the data matrix. The results, as shown in Figure 5.5, indicated that all methods provided the same subspace estimation accuracy. However, in terms of runtime, as mentioned previously in Section 3.3, it can be seen from the Figure 5.6 that, when the data matrix is small $(n, m \leq 1000)$, all the GMNS-based algorithms took the similar amount time

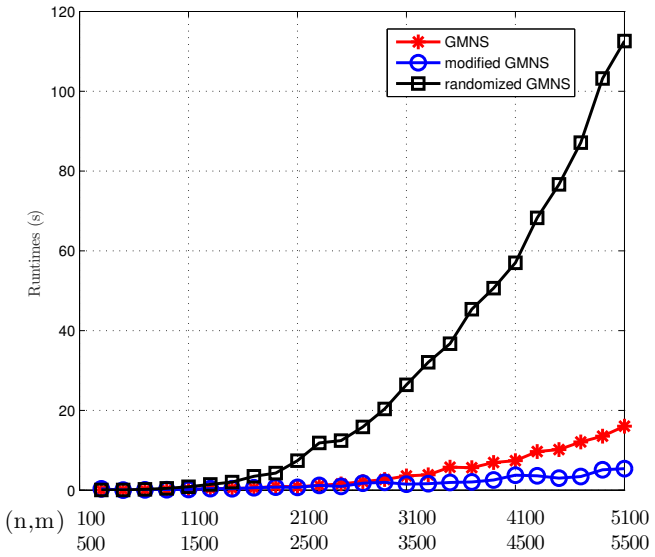


Figure 5.6: Effect of data matrix size, (n, m), on runtime of GMNS-based PSA algorithms; $p = 20$, $k = 5$.

to obtain the same accuracy. When dealing with matrices of higher dimension, the modified GMNS-based algorithm was faster.

5.1.4 Effect of the relationship between the number of sensors, sources and the number of DSP units

As mentioned above, the GMNS and modified GMNS-based PSA may be useful for only data measurements under the condition $p < n/k$, meanwhile the randomized GMNS-based approach was proposed to handle the rest. The key idea is (1) to choose the number of random vectors so that $n < k.p \leq l$, so the problem will return the original setup, or (2) to structure the random matrix using the subsampled random FFT, thanks to advantages of spectral domain. We fixed the size of the data matrix at $n = 150, m = 500$, and $k = 2$. The number of random vectors are set at $l = 2p$. As we can see from Figure 5.7, the randomized GMNS can be useful for the problem, as shown via the green line.

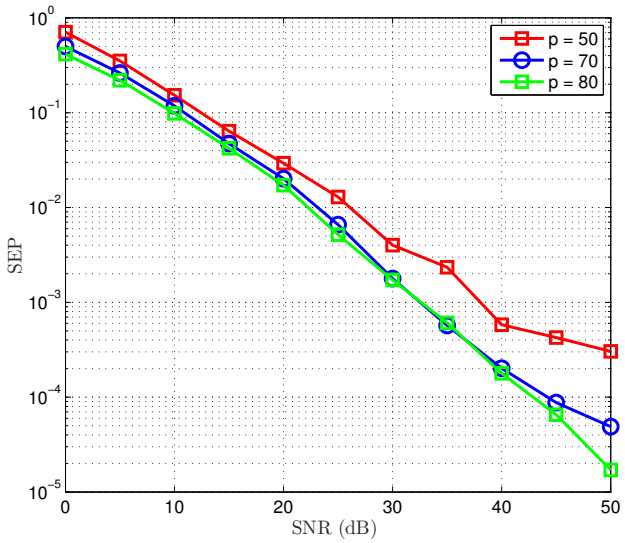


Figure 5.7: Performance of the randomized GMNS algorithm on data matrices with $k.p > n$, $k = 2$.

5.2 GMNS-based PARAFAC

We simulated tensors $\boldsymbol{\mathcal{X}} \in \mathbb{R}^{I \times J \times K}$, derived from the Gaussian distribution $\mathcal{N}(0, 1)$. The tensors were then normalized and added by a random noise $\boldsymbol{\mathcal{N}}$ with a parameter σ to control the noise level

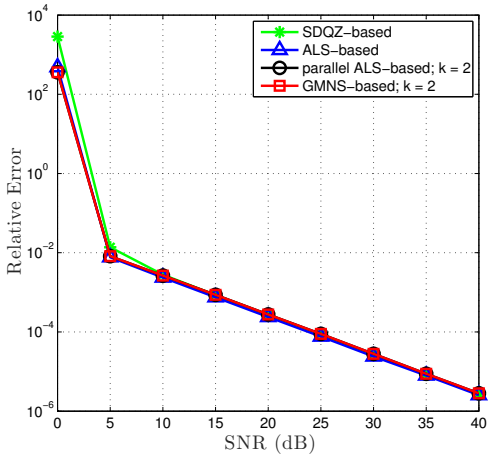
$$\boldsymbol{\mathcal{Y}} = \frac{\boldsymbol{\mathcal{X}}}{\|\boldsymbol{\mathcal{X}}\|} + \sigma \frac{\boldsymbol{\mathcal{N}}}{\|\boldsymbol{\mathcal{N}}\|}.$$

To assess the estimated factors, we use the metric of relative error, ρ, as given by

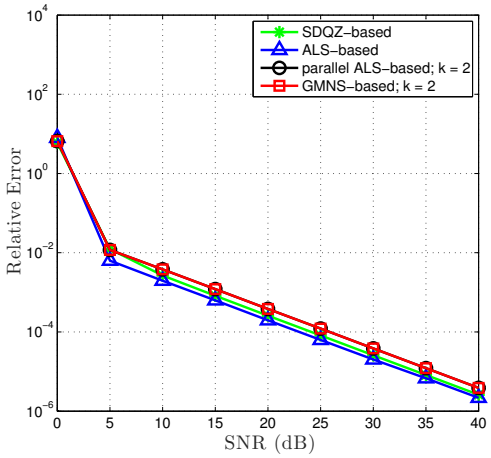
$$\rho(\mathbf{H}) = \frac{1}{L} \sum_{i=1}^{L} \frac{\|\mathbf{H}_i - \mathbf{H}_{ex}\|}{\|\mathbf{H}_{ex}\|}, \tag{5.3}$$

where L is the Monte Carlo run, $\mathbf{H}_i$ and $\mathbf{H}_{ex}$ are estimated and true factors respectively.

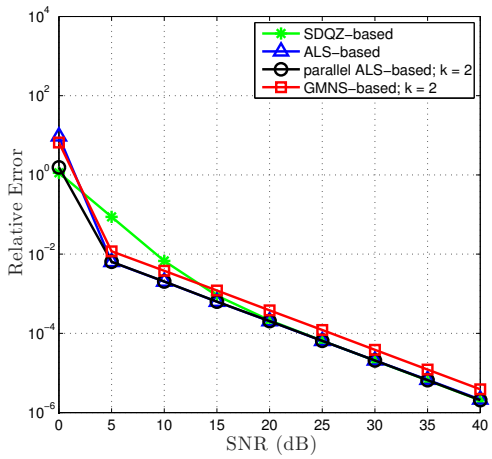
Our experiments are implemented in MATLAB 2015b on Intel core i7 processor and 8G RAM machine using the tensor toolbox [34]. Four kinds of PARAFAC algorithms are compared: Simultaneous Diagonalization computed by QR iteration-based



(a) Loading matrix **A**



(b) Loading matrix **B**



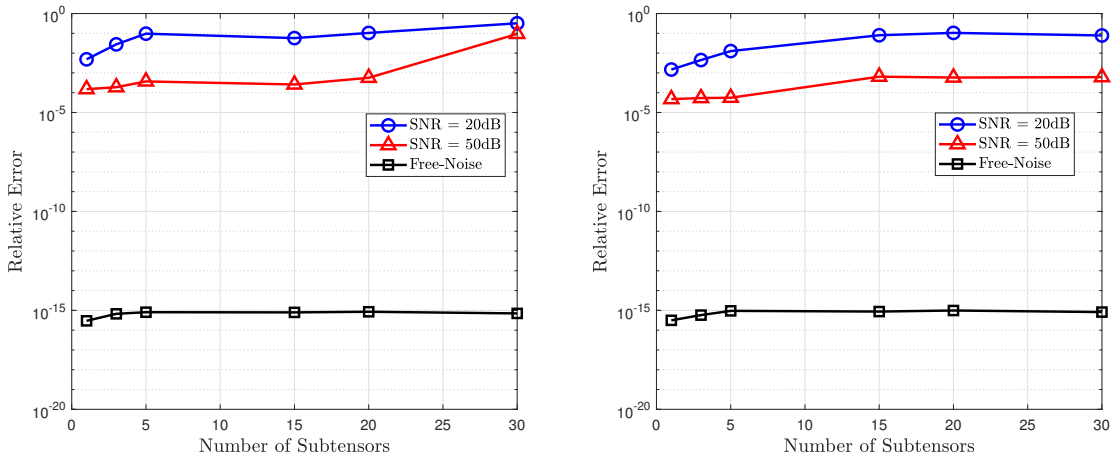
(c) Loading matrix **C**

Figure 5.8: Effect of noise on performance of PARAFAC algorithms; tensor size = $50 \times 50 \times 60$, rank $R = 5$.

PARAFAC, namely SDQZ-based PARAFAC [35], original ALS-based PARAFAC [26], parallel ALS-based PARAFAC developed in [23] and described in Section 2.2 and our proposed GMNS-based PARAFAC. The number of Monte Carlo run is fixed at $L = 100$.

5.2.1 Effect of Noise

We study the effect of noise on the performance of the PARAFAC algorithms at different values of SNR. The tested tensor has size of $100 \times 100 \times 120$ and rank of 10.



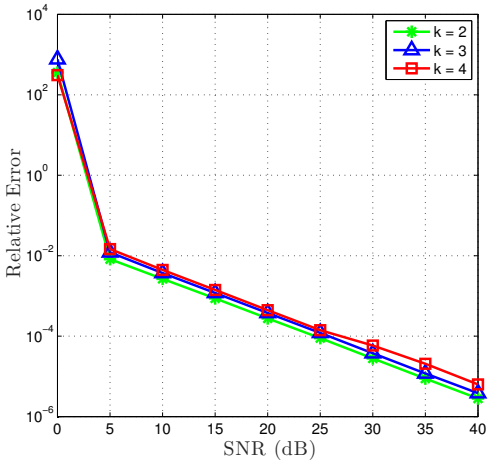
(a) Loading matrix $\mathbf{C}$ for $\boldsymbol{\mathcal{X}}_1$ of size $50 \times 50 \times 60$ (b) Loading matrix $\mathbf{C}$ for $\boldsymbol{\mathcal{X}}_2$ of size $100 \times 100 \times 120$

Figure 5.9: Effect of number of sub-tensors on performance of GMNS-based PARAFAC algorithm; tensor rank $R = 5$.

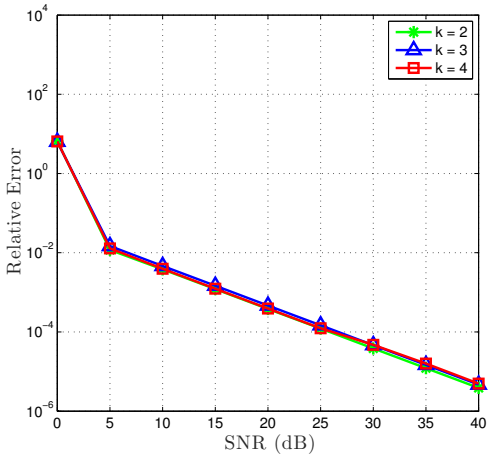
As shown in Figure 5.8, our GMNS-based PARAFAC algorithm performed similarly to the other ALS-based PARAFAC algorithms. At low SNR ($\leq$ 15dB), they were all better than SDQZ-based PARAFAC. At high SNR, all algorithms yielded almost the same results in terms of relative estimation error.

5.2.2 Effect of the number of sub-tensors, k

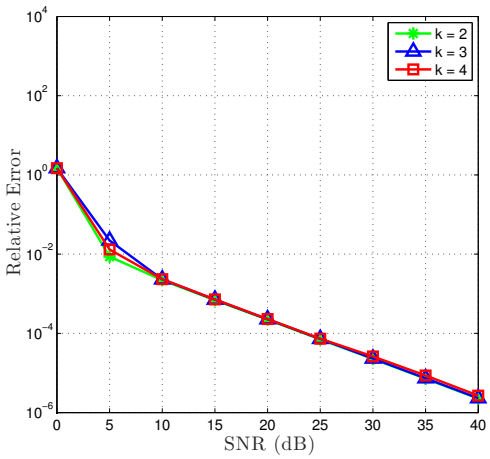
Consider two tensors with size of $50 \times 50 \times 60$ and $100 \times 100 \times 120$. The SNRs are fixed to 20 dB and 50 dB. Assume that the tensors are divided into sub-tensors by splitting the loading matrix $\mathbf{C}$. The number of sub-tensors varied in the range $[1, k/\operatorname{rank}(\boldsymbol{\mathcal{X}})]$, while still being required to maintain the conditions of uniqueness of PARAFAC. The experimental results are shown in Figure 5.9 and Figure 5.10. It can be seen that, in general, the higher the number of sub-tensors was, the lower the performance of the GMNS-based PARAFAC algorithm, with or without noise. Intuitively, this is a trade-off between complexity and accuracy over the number of DSP units. However,



(a) Loading matrix **A**



(b) Loading matrix **B**



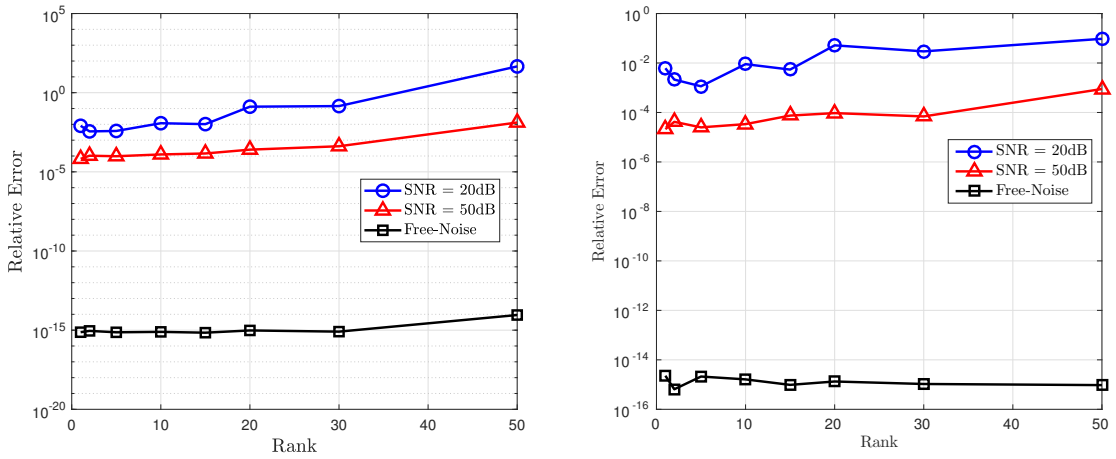
(c) Loading matrix **C**

Figure 5.10: Effect of number of sub-tensors on performance of GMNS-based PARAFAC algorithm; tensor size = $50 \times 50 \times 60$, rank $R = 5$.

the difference was little.

5.2.3 Effect of tensor rank, R

Consider two tensors with size of $50 \times 50 \times 60$ and $100 \times 100 \times 120$. The number of sub-tensors is fixed $k = 2$. The results are shown in Figure 5.11. Generally, the higher the rank of the tensor was, the lower the performance of the GMNS-based PARAFAC algorithm resulted. Under the effect of noise, the algorithm still provided a reasonable



(a) Loading matrix $\mathbf{C}$ for $\boldsymbol{\mathcal{X}}_1$ of size $50 \times 50 \times 60$ (b) Loading matrix $\mathbf{C}$ for $\boldsymbol{\mathcal{X}}_2$ of size $100 \times 100 \times 120$

Figure 5.11: Effect of tensor rank, R, on performance of GMNS-based PARAFAC algorithm.

estimation accuracy for tensors of small rank; $R(\boldsymbol{\mathcal{X}}_1) < 30$ or $R(\boldsymbol{\mathcal{X}}_2) < 50$. However, there was an unprecedented rise in error if the tensor rank became greater a specific threshold of n/k. Therefore, choosing k plays a vital role in decomposing a tensor with a given rank.

5.3 GMNS-based HOSVD

To study the performance of the proposed GMNS-based HOSVD, we investigate three main application-based scenarios: best low-rank tensor approximation, tensor-based principal subspace estimation, and tensor-based dimensionality reduction.

5.3.1 Application 1: Best low-rank tensor approximation

A performance comparison of Tucker decomposition with different initialization methods is provided via simulation study. In particular, we consider three algorithms to initialize loading factors: original HOSVD, GMNS-based HOSVD and a method that

factors are chosen randomly (legend = RAND) [12]. After that, the alternating least square (ALS) algorithm is applied to obtain the best low-rank approximation of tensors.

Two performance metrics are used: tensor core relative change (TCRC) and subspace relative change (SRC). They are defined as

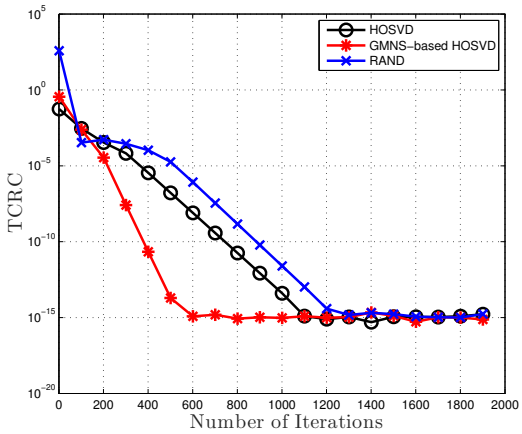
$$\text{TCRC}(k) = \frac{\|\boldsymbol{\mathcal{G}}^{(k)} - \boldsymbol{\mathcal{G}}^{(k-1)}\|}{\|\boldsymbol{\mathcal{G}}^{(k-1)}\|}, \tag{5.4}$$

$$\text{SRC}(k) = \frac{\sum_{i=1}^{N} \|\mathbf{U}_i^{(k)}(\mathbf{U}_i^{(k)})^T - \mathbf{U}_i^{(k-1)}(\mathbf{U}_i^{(k-1)})^T\|}{\sum_{i=1}^{N} \|\mathbf{U}_i^{(k-1)}(\mathbf{U}_i^{(k-1)})^T\|}, \tag{5.5}$$

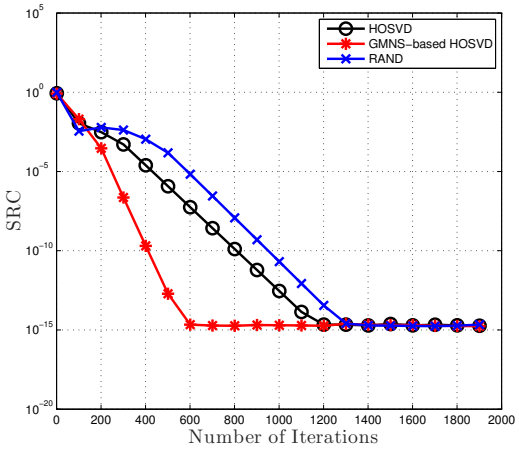
where N is the number of modes (fibers), $\boldsymbol{\mathcal{G}}_{(k)}$ and $\mathbf{U}_i^{(k)}$ are the estimated tensor core and factors at the k-th iteration step.

We use three tensors to assess algorithm performance: two synthetic tensors and one real tensor from the Coil20 database [5]. The two synthetic tensors $\boldsymbol{\mathcal{X}}_1$ of size $50 \times 50 \times 50$ and $\boldsymbol{\mathcal{X}}_2$ of size $400 \times 400 \times 400$ were randomly generated from the zero-mean, unit-variance Gaussian distribution. They were then compressed into a tensor core $\boldsymbol{\mathcal{G}}_1$ of $5 \times 5 \times 5$. The Coil20 database is composed of 9 subjects with 72 different images. We formed a real tensor $\boldsymbol{\mathcal{X}}_3$ of size $128 \times 128 \times 648$, associated with a tensor core $\boldsymbol{\mathcal{G}}_2$ of size $64 \times 64 \times 100$.

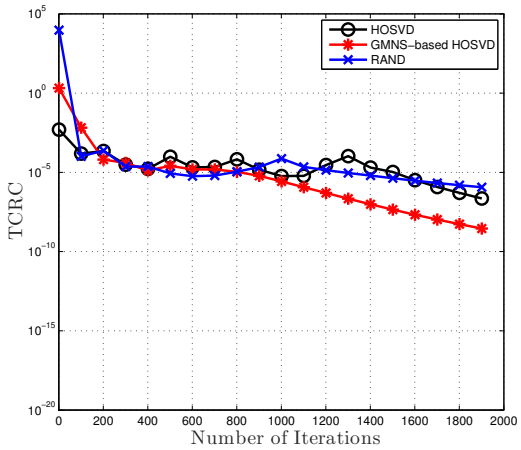
The convergence results are shown in Figure 5.12 and Figure 5.13. As we can see that, for the small synthetic tensor, the GMNS-based HOSVD algorithm converged fastest, while still providing a good performance, in terms of TCRC and SRC ($\approx 10^{-15}$). For the big synthetic tensor, all algorithms provided similar performance, but the GMNS-based algorithm yielded better performance with a faster convergence than that of the original HOSVD and the random-based algorithms, as shown clearly in Figure 5.12 (c) and (d) . In the case of the real data, all algorithms provided the same



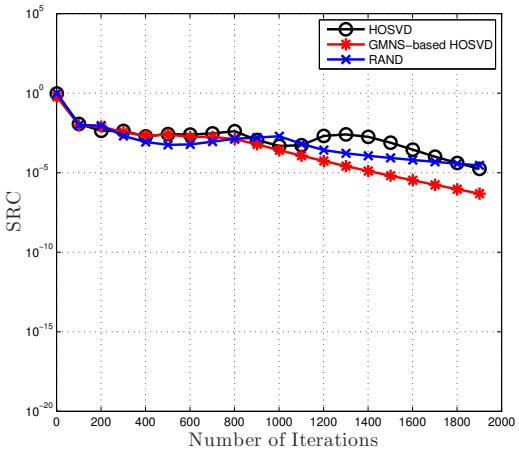
(a) TCRC for $\boldsymbol{\mathcal{X}}_1$ of size $50 \times 50 \times 50$



(b) SRC for $\boldsymbol{\mathcal{X}}_1$ of size $50 \times 50 \times 50$



(c) TCRC for $\boldsymbol{\mathcal{X}}_2$ of size $400 \times 400 \times 400$



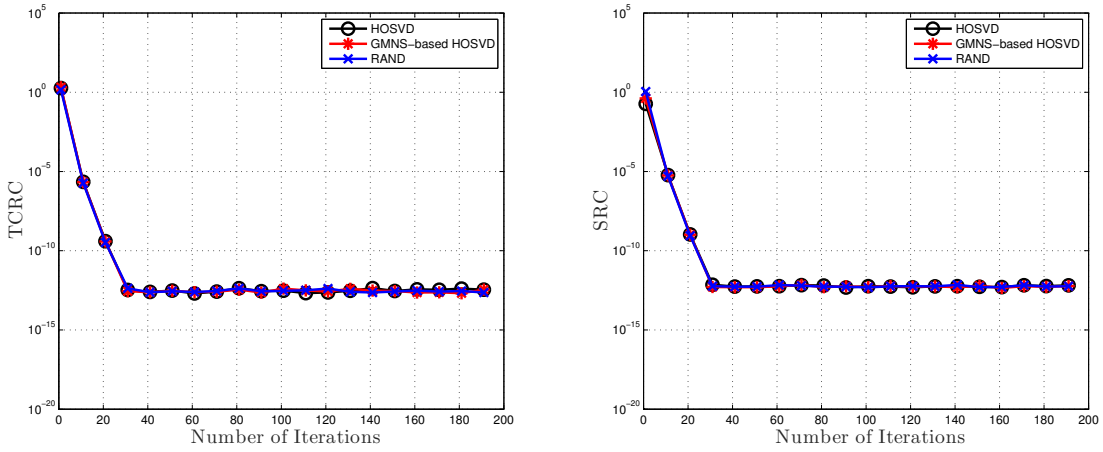
(d) SRC for $\boldsymbol{\mathcal{X}}_2$ of size $400 \times 400 \times 400$

Figure 5.12: Performance of Tucker decomposition algorithms on random tensors, $\boldsymbol{\mathcal{X}}_1$ and $\boldsymbol{\mathcal{X}}_2$, associated with a core tensor $\boldsymbol{\mathcal{G}}_1$ size of $5 \times 5 \times 5$.

performance in term of TCRC and SRC with a fast convergence.

5.3.2 Application 2: Tensor-based principal subspace estimation

We investigate the use of GMNS-based HOSVD, original HOSVD, modified GMNS, and SVD for principal subspace estimation. Tensor-based subspace estimation was



(a) TCRC for $\boldsymbol{\mathcal{X}}$ of size $128 \times 128 \times 648$ (b) SRC for $\boldsymbol{\mathcal{X}}$ of size $128 \times 128 \times 648$

Figure 5.13: Performance of Tucker decomposition algorithms on real tensor obtained from Coil20 database [5]; $\boldsymbol{\mathcal{X}}$ of size $128 \times 128 \times 648$ associated with tensor core $\boldsymbol{\mathcal{G}}_2$ of size $64 \times 64 \times 100$.

introduced in [36], wherein it was proved that the HOSVD-based approach improved subspace estimation accuracy and was better than conventional methods, like SVD, if the steering matrix $\mathbf{A}$ satisfies some specific conditions. Inspired by this work, we would like to see how the proposed GMNS-based HOSVD algorithms work for principal subspace estimation.

For the sake of simplicity, we assume that the measurement $\mathbf{X}$ can be expressed by matrix and tensor representations as

$$\mathbf{X} = \mathbf{A}\mathbf{S} + \sigma\mathbf{N},$$

$$\boldsymbol{\mathcal{X}} = \boldsymbol{\mathcal{A}} \times_{R+1} \mathbf{S}^T + \sigma\boldsymbol{\mathcal{N}},$$

where the steering matrix $\mathbf{A}$ and tensor $\boldsymbol{\mathcal{A}}$ can be expressed by two sub-systems $\mathbf{A}_1$

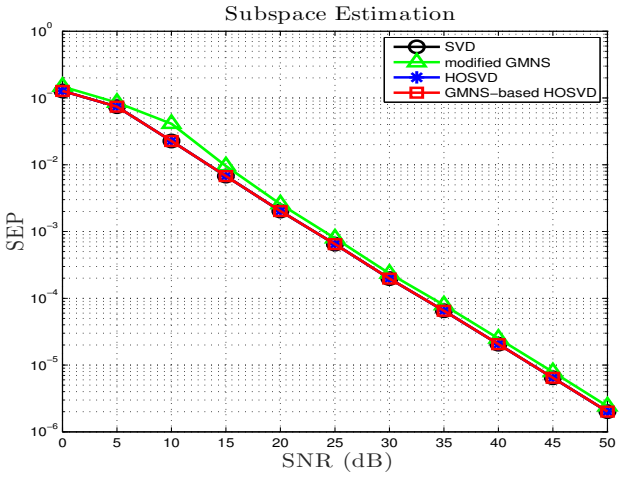


Figure 5.14: HOSVD for PSA

and $\mathbf{A}_2$ as

$$\mathbf{A} = \mathbf{A}_1 \odot \mathbf{A}_2,$$

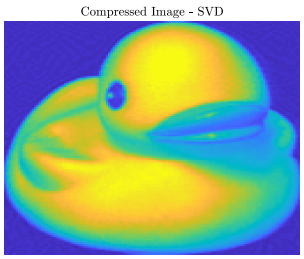
$$\boldsymbol{\mathcal{A}} = \boldsymbol{\mathcal{I}} \times_1 \mathbf{A}_1 \times_2 \mathbf{A}_2.$$

The multidimensional version of the true subspace $\mathbf{W}$ in the matrix case can be defined as

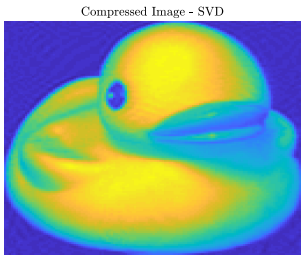
$$\boldsymbol{\mathcal{U}} = \boldsymbol{\mathcal{G}} \times_1 \mathbf{U}_1 \times_2 \mathbf{U}_2, \tag{5.6}$$

where $\boldsymbol{\mathcal{G}}$ denotes the core of tensor $\boldsymbol{\mathcal{X}}$, $\mathbf{U}_1$ and $\mathbf{U}_2$ are two (truncated) loading factors derived by a specific algorithm for Tucker decomposition, such as the original HOSVD, the GMNS-based HOSVD, and the HOOI algorithms.

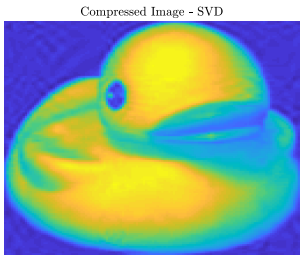
In this work, we follow experiments set up in [36]. The array steering tensor $\boldsymbol{\mathcal{A}}$ and the signal $\mathbf{S}$ were derived from the random zeros-mean, unit-variance Gaussian distribution in the same way presented in Section 5.1. The experimental results are shown in Figure 5.14. It can be seen that the GMNS-based HOSVD algorithm provided



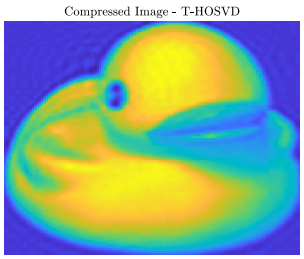
(a) SVD: $n = 40$, RMSE = 0.0036



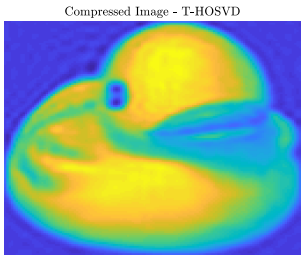
(b) SVD: $n = 30$, RMSE = 0.0059



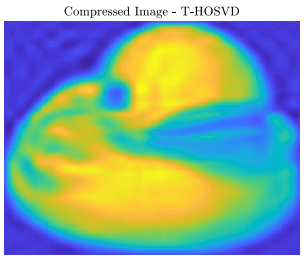
(c) SVD: $n = 20$, RMSE = 0.01



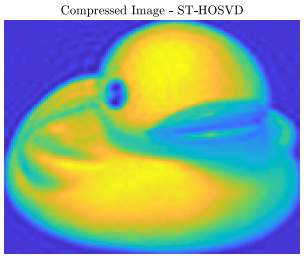
(d) T-HOSVD: $n = 40$, RMSE = 0.0129



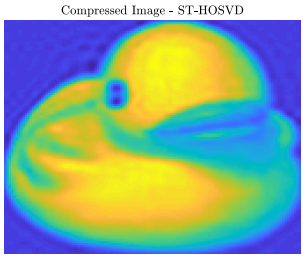
(e) T-HOSVD: $n = 30$, RMSE = 0.0166



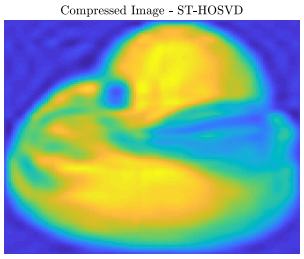
(f) T-HOSVD: $n = 20$, RMSE = 0.0328



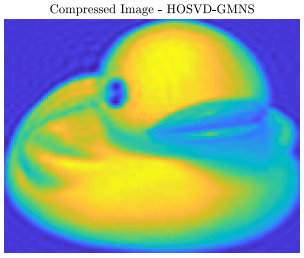
(g) ST-HOSVD: $n = 40$, RMSE = 0.0127



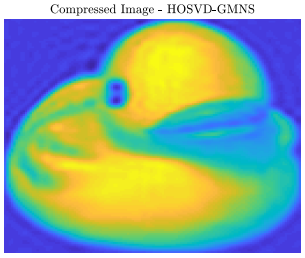
(h) ST-HOSVD: $n = 30$, RMSE = 0.0164



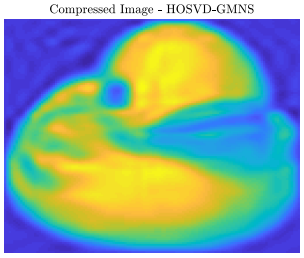
(i) ST-HOSVD: $n = 20$, RMSE = 0.0326



(j) GMNS-based HOSVD: $n = 40$, RMSE = 0.0129



(k) GMNS-based HOSVD: $n = 30$, RMSE = 0.0166



(l) GMNS-based HOSVD: $n = 20$, RMSE = 0.0328

Figure 5.15: Image compression using SVD and different Tucker decomposition algorithms.

almost the same subspace estimation accuracy in terms of SEP as the HOSVD-based, SVD-based and GMNS-based algorithms. Thus, the proposed GMNS-based HOSVD algorithm can be useful for subspace-based parameter estimation.

5.3.3 Application 3: Tensor based dimensionality reduction

We investigate the use of GMNS-based HOSVD, original truncated HOSVD (legend = T-HOSVD), another truncated HOSVD [37] (legend = ST-HOSVD), and SVD for compression of an image tensor with a fixed rank. The image tensor was obtained from the Coil20 database.

The root mean square error (RMSR) is used as the performance metric and is defined as

$$\text{RMSE} = \frac{\|\mathbf{A}_{\text{re}} - \mathbf{A}_{\text{ex}}\|}{\|\mathbf{A}_{\text{ex}}\|}, \tag{5.7}$$

where $\mathbf{A}_{\text{ex}}$ and $\mathbf{A}_{\text{re}}$ are the true and reconstructed images, respectively.

The results are shown in Figure 5.15. Clearly, GMNS-based HOSVD provided the same performance as truncated-HOSVD but slightly worse ($\backsimeq 0.2\%$ in terms of RMSE) than ST-HOSVD. The tensor-based approach for dimensionality reduction was much worse than SVD-based approach on each single image.

Chapter 6

Conclusions

In this thesis, motivated by advantages of the GMNS method, we proposed several new algorithms for principal subspace analysis and tensor decompositions. We first introduced modified and randomized GMNS-based algorithms for PSA with reasonable subspace estimation accuracy. Then, based on these, we proposed two GMNS-based algorithms for PARAFAC and HOSVD. Numerical experiments indicate that our proposed algorithms may be a suitable alternative to their counterparts, as they can significantly reduce the computational complexity while preserving reasonable performance.

References

[1] L. T. Thanh, V.-D. Nguyen, N. Linh-Trung, and K. Abed-Meraim, "Three-way tensor decompositions: A generalized minimum noise subspace based approach," *REV Journal on Electronics and Communications*, vol. 8, no. 1-2, 2018.

[2] ——, "Robust subspace tracking for incomplete data with outliers," in *The 44th IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. Brighton, UK: IEEE, May 2019 [Submitted].

[3] L. T. Thanh, A. D. Nguyen Thi, N. Viet-Dung, L.-T. Nguyen, and A.-M. Karim, "Multi-channel eeg epileptic spike detection by a new method of tensor decomposition," *IOP Journal of Neural Engineering*, Oct. 2018 [Submitted].

[4] N. T. Anh-Dao, L. T. Thanh, and N. Linh-Trung, "Nonnegative tensor decomposition for eeg epileptic spike detection," in *the 5th NAFOSTED Conference on Information and Computer Science (NICS)*. IEEE, Nov. 2018, pp. 196–201.

[5] S. A. Nene, S. K. Nayar, and H. Murase, "Columbia University Image Library (COIL-20)," 1996. [Online]. Available: http://www.cs.columbia.edu/CAVE/software/softlib/coil-20.php

[6] M. Chen, S. Mao, and Y. Liu, "Big data: A survey," *Mobile networks and applications*, vol. 19, no. 2, pp. 171–209, 2014.

[7] E. Acar, C. Aykut-Bingol, H. Bingol, R. Bro, and B. Yener, "Multiway analysis of epilepsy tensors," *Bioinformatics*, vol. 23, no. 13, pp. i10–i18, 2007.

[8] C.-F. V. Latchoumane, F.-B. Vialatte, J. Solé-Casals, M. Maurice, S. R. Wimalaratna, N. Hudson, J. Jeong, and A. Cichocki, "Multiway array decomposition analysis of EEGs in Alzheimer's disease," *Journal of neuroscience methods*, vol. 207, no. 1, pp. 41–50, 2012.

[9] F. Cong, Q.-H. Lin, L.-D. Kuang, X.-F. Gong, P. Astikainen, and T. Ristaniemi, "Tensor decomposition of EEG signals: a brief review," *Journal of neuroscience methods*, vol. 248, pp. 59–69, 2015.

[10] V. D. Nguyen, K. Abed-Meraim, and N. Linh-Trung, "Fast tensor decompositions for big data processing," in *2016 International Conference on Advanced Technologies for Communications (ATC)*, Oct 2016, pp. 215–221.

[11] N. D. Sidiropoulos, L. D. Lathauwer, X. Fu, K. Huang, E. E. Papalexakis, and C. Faloutsos, "Tensor decomposition for signal processing and machine learning," *IEEE Transactions on Signal Processing*, vol. 65, no. 13, pp. 3551–3582, July 2017.

[12] T. G. Kolda and B. W. Bader, "Tensor decompositions and applications," *SIAM review*, vol. 51, no. 3, pp. 455–500, 2009.

[13] L. Tran, C. Navasca, and J. Luo, "Video detection anomaly via low-rank and sparse decompositions," in *2012 Western New York Image Processing Workshop (WNYIPW)*. IEEE, 2012, pp. 17–20.

[14] X. Zhang, X. Shi, W. Hu, X. Li, and S. Maybank, "Visual tracking via dynamic

tensor analysis with mean update," *Neurocomputing*, vol. 74, no. 17, pp. 3277–3285, 2011.

[15] H. Li, Y. Wei, L. Li, and Y. Y. Tang, "Infrared moving target detection and tracking based on tensor locality preserving projection," *Infrared Physics & Technology*, vol. 53, no. 2, pp. 77–83, 2010.

[16] S. Bourennane, C. Fossati, and A. Cailly, "Improvement of classification for hyperspectral images based on tensor modeling," *IEEE Geoscience and Remote Sensing Letters*, vol. 7, no. 4, pp. 801–805, 2010.

[17] N. Renard and S. Bourennane, "Dimensionality reduction based on tensor modeling for classification methods," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 47, no. 4, pp. 1123–1131, 2009.

[18] H. Fanaee-T and J. Gama, "Event detection from traffic tensors: A hybrid model," *Neurocomputing*, vol. 203, pp. 22–33, 2016.

[19] V. D. Nguyen, K. Abed-Meraim, N. Linh-Trung, and R. Weber, "Generalized minimum noise subspace for array processing," *IEEE Transactions on Signal Processing*, vol. 65, no. 14, pp. 3789–3802, July 2017.

[20] A. H. Phan and A. Cichocki, "PARAFAC algorithms for large-scale problems," *Neurocomputing*, vol. 74, no. 11, pp. 1970–1984, 2011.

[21] A. L. de Almeida and A. Y. Kibangou, "Distributed computation of tensor decompositions in collaborative networks," in *2013 IEEE 5th International Workshop on Computational Advances in Multi-Sensor Adaptive Processing (CAMSAP)*. IEEE, 2013, pp. 232–235.

[22] A. L. De Almeida and A. Y. Kibangou, "Distributed large-scale tensor decomposition," in *2014 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2014, pp. 26–30.

[23] V. D. Nguyen, K. Abed-Meraim, and L. T. Nguyen, "Parallelizable PARAFAC decomposition of 3-way tensors," in *2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, April 2015, pp. 5505–5509.

[24] K. Shin, L. Sael, and U. Kang, "Fully scalable methods for distributed tensor factorization," *IEEE Transactions on Knowledge and Data Engineering*, vol. 29, no. 1, pp. 100–113, Jan 2017.

[25] D. Chen, Y. Hu, L. Wang, A. Y. Zomaya, and X. Li, "H-PARAFAC: Hierarchical parallel factor analysis of multidimensional big data," *IEEE Transactions on Parallel and Distributed Systems*, vol. 28, no. 4, pp. 1091–1104, April 2017.

[26] J. D. Carroll and J.-J. Chang, "Analysis of individual differences in multidimensional scaling via an N-way generalization of "Eckart-Young" decomposition," *Psychometrika*, vol. 35, no. 3, pp. 283–319, 1970.

[27] N. Halko, P.-G. Martinsson, and J. A. Tropp, "Finding structure with randomness: Probabilistic algorithms for constructing approximate matrix decompositions," *SIAM Review*, vol. 53, no. 2, pp. 217–288, 2011.

[28] M. W. Mahoney, "Randomized algorithms for matrices and data," *Foundations and Trends® in Machine Learning*, vol. 3, no. 2, pp. 123–224, 2011.

[29] D. P. Woodruff, "Sketching as a Tool for Numerical Linear Algebra," *Foundations and Trends® in Theoretical Computer Science*, vol. 10, no. 1–2, pp. 1–157, 2014.

[30] V. Rokhlin, A. Szlam, and M. Tygert, "A randomized algorithm for principal component analysis," *SIAM Journal on Matrix Analysis and Applications*, vol. 31, no. 3, pp. 1100–1124, 2009.

[31] C. Boutsidis, P. Drineas, and M. Magdon-Ismail, "Near-optimal column-based matrix reconstruction," *SIAM Journal on Computing*, vol. 43, no. 2, pp. 687–717, 2014.

[32] A. R. Benson, D. F. Gleich, and J. Demmel, "Direct QR factorizations for tall-and-skinny matrices in MapReduce architectures," in *2013 IEEE International Conference on Big Data*, Oct 2013, pp. 264–272.

[33] N. Kishore Kumar and J. Schneider, "Literature survey on low rank approximation of matrices," *Linear and Multilinear Algebra*, vol. 65, no. 11, pp. 2212–2244, 2017.

[34] B. W. Bader, T. G. Kolda *et al.*, "MATLAB Tensor Toolbox Version 2.6," Available online, February 2015. [Online]. Available: http://www.sandia.gov/~tgkolda/TensorToolbox/

[35] L. De Lathauwer, "A link between the canonical decomposition in multilinear algebra and simultaneous matrix diagonalization," *SIAM Journal on Matrix Analysis and Applications*, vol. 28, no. 3, pp. 642–666, 2006.

[36] M. Haardt, F. Roemer, and G. Del Galdo, "Higher-order SVD-based subspace estimation to improve the parameter estimation accuracy in multidimensional harmonic retrieval problems," *IEEE Transactions on Signal Processing*, vol. 56, no. 7, pp. 3198–3213, 2008.

[37] N. Vannieuwenhoven, R. Vandebril, and K. Meerbergen, "A new truncation strategy for the higher-order singular value decomposition," *SIAM Journal on Scientific Computing*, vol. 34, no. 2, pp. A1027–A1052, 2012.

Le Trung Thanh

Personal Information

Advanced Institute of Engineering and Technology
VNU University of Engineering and Technology
707 Room, E3 Building, 144 Xuan Thuy, Hanoi, Vietnam

(+84) 853 008 712
thanhletrung@vnu.edu.vn
ResearchGate

Research Interests

Signal Processing

Education

VNU University of Engineering and Technology, Hanoi, Vietnam

M.S., Communications Engineering, (12/2016 – 12/2018)

- Topic: *GMNS-based Tensor Decomposition*
- Advisor: Assoc. Prof. Nguyen Linh Trung

B.S., Electronics and Communications, (8/2012 – 7/2016)
International Standard Program, instructed in English

- Topic: *EEG Epileptic Spike Detection Using Deep Belief Networks*
- Advisor: Assoc. Prof. Nguyen Linh Trung

Professional Experience

Research Assistant

- Advanced Institute of Engineering and Technology (AVITECH) 1/2018 – present
  VNU University of Engineering and Technology
- Faculty of Electronics and Telecommunications, 7/2016 – 12/2017
  VNU University of Engineering and Technology
  Supervisor: Prof. Nguyen Linh Trung
  Research Topics:
  - Network Coding: *Implementation of OFDM system over Software Defined Radio*
  - Deep Learning: *EEG Epileptic Spike Detection Using Deep Learning*
  - Tensor Decomposition: *GMNS-based Tensor Decomposition.*
  - Graph Signal Processing: *Vertex-Frequency Processing Tools for GSP.* (ongoing)
  - Subspace Tracking: *Robust Subspace Tracking for Missing Data with Outliers.* (ongoing)

Teaching Assistant

- Faculty of Electronics and Telecommunications, 8/2017 – present
  VNU University of Engineering and Technology
  Instructor: Prof. Nguyen Linh Trung
  - ELT 2029 – Engineering Mathematics
  - ELT 3144 – Digital Signal Processing

Refereed Journal Publications

1. **Le Trung Thanh**, Nguyen Linh Trung, Nguyen Viet Dung and Karim Abed-Meraim. "Windowed Graph Fourier Transform for Directed Graph". *IEEE Transactions on Signal Processing*, [to submit Nov 2018].
2. **Le Trung Thanh**, Nguyen Thi Anh Dao, Viet-Dung Nguyen, Nguyen Linh-Trung, and Karim Abed-Meraim. "Multi-channel EEG epileptic spike detection by a new method of tensor decomposition". *IOP Journal of Neural Engineering*, Oct 2018. [under revision].
3. **Le Trung Thanh**, Nguyen Viet-Dung, Nguyen Linh-Trung and Karim Abed-Meraim. "Three-Way Tensor Decompositions: A Generalized Minimum Noise Subspace Based Approach." *REV Journal on Electronics and Communications*, vol. 8, no. 1-2, 2018.

4. Le Thanh Xuyen, **Le Trung Thanh**, Dinh Van Viet, Tran Quoc Long, Nguyen Linh-Trung and Nguyen Duc Thuan. "Deep Learning for Epileptic Spike Detection" *VNU Journal of Science: Computer Science and Communication Engineering*, vol. 33, no. 2, 2018.

Conference Publications

1. **Le Trung Thanh**, Viet-Dung Nguyen, Nguyen Linh-Trung and Karim Abed-Meraim. 'Robust Subspace Tracking with Missing Data and Outliers via ADMM ", in *The 44th International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, Brighton-UK, 2019. IEEE. [Submitted]
2. Nguyen Thi Anh Dao, **Le Trung Thanh**, Nguyen Linh-Trung, Le Vu Ha. "Nonnegative Tucker Decomposition for EEG Epileptic Spike Detection", in *2018 NAFOSTED Conference on Information and Computer Science (NICS),* Ho Chi Minh, 2018, pp.196-201. IEEE.
3. **Le Trung Thanh**, Nguyen Linh-Trung, Viet-Dung Nguyen and Karim Abed-Meraim. "A New Windowed Graph Fourier Transform", in *2017 NAFOSTED Conference on Information and Computer Science (NICS),* Hanoi, 2017, pp.150-155. IEEE.
4. **Le Trung Thanh**, Nguyen Thi Anh Dao, Nguyen Linh-Trung and Ha Vu Le, "On the overall ROC of multistage systems," in *2017 International Conference on Advanced Technologies for Communications (ATC)*, Quy Nhon, 2017, pp. 229-234. IEEE.
5. Nguyen Thi Hoai Thu, **Le Trung Thanh**, Chu Thi Phuong Dung, Nguyen Linh-Trung and Ha Vu Le."Multi-source data analysis for bike sharing systems", in *2017 International Conference on Advanced Technologies for Communications (ATC),* Quynhon, 2017, pp. 235-240. IEEE.

Technical Skills

Computer Programming:

- C/C++, Python, R, Matlab, GNURadio

Awards and Honors

Student Awards — VNU University of Engineering and Technology

1. **Excellent Undergraduate Thesis Award**, VNU-UET 2016

Contest Awards

1. **Third Prize** in National Physic Olympiad for Undergraduates, Vietnam Physical Society 2015
2. **Second Prize** in Provincial Excellent Physics Students Contest, Nam Dinh Department of Eduction and Training, Vietnam 2011–12
3. **Third Prize** in Provincial Excellent Informatics Students Contest Nam Dinh Department of Eduction and Training, Vietnam 2010–11

Scholarships

1. **Toshiba Scholarship**, Toshiba Scholarship Foundation, Japan 2017-18
2. **Yamada Scholarship**, Yamada Foundation, Japan 2016
3. **Odon Vallet Scholarship**, Rencontres du Vietnam 2015
4. **Tharal-InSEWA Scholarship** Tharal-In sewa Foundation, Singapore 2015
5. **Pony Chung Scholarship**, Pony Chung Foudation, Korea 2014

Additional Activities

1. Training Workshop on "Advanced Technologies for 5G and Beyond" 5-6/2018
   University of Danang.
2. The 3^{nd} International Summer School 8/2017
   Duy Tan University and British Council, Da Nang, Vietnam
3. Mini-course "Introduction to Data Science" 5/2017
   Vietnam Institute for Advanced Study in Mathematics, Hanoi, Vietnam.
4. The 1^{st} Research School on "Advanced Technologies for IoT Applications" 3/2017
   University of Technology Sydney (UTS)
   and VNU University of Engineering and Technology (VNU-UET).
5. Summer school on statistical machine learning 8/2015
   Vietnam Institute for Advanced Study in Mathematic, Hanoi, Vietnam.
6. Intensive English Training C1 CEFR 2012–13
   International Standard Program
   VNU University of Languages and International Studies (VNU-ULIS).

References

Assoc. Prof. Nguyen Linh Trung
Advanced Institute of Engineering and Technology, (+84) 4 3754 9271
VNU University of Engineering and Technology linhtrung@vnu.edu.vn