Matrix product state decomposition in machine learning and signal processing

Johann Anton Bengua

A thesis submitted for the degree of Doctor of Philosophy at

The University of Technology Sydney in 2016

Faculty of Engineering and Information Technology

Certificate of Original Authorship

I, **Johann Bengua**, certify that the work in this thesis has not previously been submitted for a degree nor has it been submitted as part of the requirements for a degree except as fully acknowledged within the text.

I also certify that the thesis has been written by me. Any help that I have received in my research work and the preparation of the thesis itself has been acknowledged. In addition, I certify that all information sources and literature used are indicated in the thesis.

Signature:

Date:

Author's Publications

The contents of this thesis are based on the following papers that have been published, accepted, or submitted to peer-reviewed journals and conferences.

Journal papers

[J1] Ho N. Phien, Johann A. Bengua, Hoang D. Tuan, Philippe Corboz, and Román Orús. Infinite projected entangled pair states algorithm improved: Fast full update and gauge fixing. *Phys. Rev. B,* 92:035142, July 2015.

[J2] Johann A. Bengua, Ho N. Phien, Hoang D. Tuan, and Minh N. Do. Efficient tensor completion for color image and video recovery: Low-rank tensor train, *IEEE Transactions on Image Processing*, Accepted, August 2016.

[J3] Johann A. Bengua, Ho N. Phien, Hoang D. Tuan, and Minh N. Do. Matrix Product State for Higher-Order Tensor Compression and Classification. *IEEE Transactions on Signal Processing*, Resubmitted, June 2016.

[J4] Johann A. Bengua, Hoang D. Tuan, Trung Q. Duong and H. Vincent Poor. Joint Sensor and Relay Power Control in Tracking Gaussian Mixture Targets by Wireless Sensor Networks, *IEEE Transactions on Signal Processing*, Submitted, July 2016.

Conference papers

[C1] Johann A. Bengua, Ho N. Phien and Hoang D. Tuan. Optimal Feature Extraction and Classification of Tensors via Matrix Product State Decomposition, *Proceedings of the 2015 IEEE International Congress on Big Data*, pp. 669-672, New York, NY, 2015.

[C2] Johann A. Bengua, Hoang D. Tuan, Ho N. Phien and Ha H. Kha. Two-hop Power-Relaying for Linear Wireless Sensor Networks, *Proceedings of the 2016 IEEE Sixth International Conference on Communications and Electronics*, pp. 1-5, Ha Long, Vietnam, 2016.

[C3] Johann A. Bengua, Hoang D. Tuan, Ho N. Phien and Minh N. Do. Concatenated image completion via tensor augmentation and completion, *10th International Conference on Signal Processing and Communication Systems*, Submitted, July 2016.

Acknowledgments

I would like to express my deepest gratitude to my supervisor, Prof. Tuan Hoang, for his unfaltering guidance and supervision. Thank you for your support and great insight. I also want to thank Dr. Phien Ho, who is now at Westpac, for teaching me many things related to the quantum world, for inspiring me and pushing me to try new algorithms and mathematical tools, which greatly increased my knowledge and abilities.

Furthermore, I would like to thank Prof. Minh N. Do (University of Illinois Urbana-Champaign, USA) and Prof. Vincent Poor (Princeton University, USA) for the immense support, input and collaboration in my work.

Additionally, a sincere appreciation to my colleagues in Prof. Tuan Hoang's research group at the University of Technology Sydney: Elong Che, Bao Truong and Tam Ho. Thank you for all the interesting discussions, and providing a fun and friendly research environment.

Finally, a tremendous gratitude to my mother, father and brother, for always encouraging me to seek knowledge. Most importantly, I would like to thank my beloved wife Dolly Sjafrial for always supporting me, for your patience and faith in me all these years, and for your love.

List of Abbreviations

MPS	Matrix product state
TT	Tensor train
CP	Canonical/Parallel factors
TD	Tucker decomposition
PARAFAC	Parallel factors
SiLRTC	Simple low-rank tensor completion
SiLRTC-TT	Simple low-rank tensor completion via tensor train
TMac	Tensor completion by parallel matrix factorization
TMac-TT	Parallel matrix factorization via tensor train
KA	Ket augmentation
ICTAC	Concatenated image completion via tensor augmentation and completion
TTPCA	Principal component analysis via tensor train
HOSVD	Higher-order singular value decomposition
HOOI	Higher-order orthogonal iteration
MPCA	Multilinear principal component analysis
FBCP	Fully Bayesian CP Factorization
STDC	Simultaneous tensor decomposition and completion
LRTC	Low-rank tensor completion
LRMC	Low-rank matrix completion
SVD	Singular value decomposition
ALS	Alternating least squares
PCA	Principal component analysis
LDA	Linear discriminant analysis
KNN	K-nearest neighbours

SPC-QV	Smooth PARAFAC tensor completion with quadratic variation
R-UMLDA	Uncorrelated multilinear discriminant analysis with regularization
WSN	Wireless sensor network
LSN	Linear sensor network
NSN	Nonlinear sensor network
GMM	Gaussian mixture model
MSE	Mean square error
MMSE	Minimum mean square error

Contents

List of Figures

List of Tables

Abstract

There has been a surge of interest in the study of multidimensional arrays, known as tensors. This is due to the fact that many real-world datasets can be represented as tensors. For example, colour images are naturally third-order tensors, which include two indices (or modes) for their spatial index, and one mode for colour. Also, a colour video is a fourth-order tensor comprised of frames, which are colour images, and an additional temporal index. Traditional tools for matrix analysis does not generalise so well in tensor analysis. The main issue is that tensors prescribe a natural structure, which is destroyed when they are vectorised. Many mathematical techniques such as principal component analysis (PCA) or linear discriminant analysis (LDA) used extensively in machine learning rely on vectorised samples of data. Additionally, since tensors may often be large in dimensionality and size, vectorising these samples and applying them to PCA or LDA may not lead to the most efficient results, and the computational time of the algorithms can increase significantly. This problem is known as the so-called curse of dimensionality.

Tensor decompositions and their interesting properties are needed to circumvent this problem. The Tucker (TD) or CANDECOMP/PARAFAC (CP) decompositions have been predominantly used for tensor-based machine learning and signal processing. Both utilise common factor matrices and a core tensor, which retains the dimensionality of the original tensor. A main problem with these type of decompositions is that they essentially rely on an unbalanced matricization scheme, which potentially converts a tensor to a highly unbalanced matrix, where the row size is attributed to always one mode and the column size is the product of the remaining modes. This method is not optimal for problems that rely on retaining as much correlations within the data, which is very important for tensor-based machine learning and signal processing.

In this thesis, we are interested in utilising the matrix product state (MPS) decomposition. MPS has the property that it can retain much of the correlations

within a tensor because it is based on a balanced matricization scheme, which consists of permutations of matrix sizes that can investigate the different correlations amongst all modes of a tensor. Several new algorithms are proposed for tensor object classification, which demonstrate an MPS-based approach as an efficient method against other tensor-based approaches. Additionally, new methods for colour image and video completion are introduced, which outperform the current state-of-the-art tensor completion algorithms.

Chapter 1

Introduction

Statistical procedures such as principal component analysis (PCA) and linear discriminant analysis (LDA) are of utmost importance for analysing numerical data, provided that the input data is structured as a vector. Not all input data can be faithfully represented in this form and vectorising data of higher dimensions could potentially destroy unique correlations [1]. There has been substantial progress into generalising these algorithms to allow for higher-dimensional input data. Tensors are multidimensional arrays that describe the underlying structure of multidimensional data [2, 3]. Specifically, an Nth-order tensor is a tensor product of N vector spaces. Tensors can be decomposed (hence the term *tensor decomposition*) into several forms and are at the core of many efficient algorithms that analyse data of high dimensions. These are but not limited to the Tucker (TD) [4], hierarchical Tucker (HT) [5, 6], canonical decomposition/parallel factors (CP) [7] and tensor-train (TT) [8] decompositions. The latter is also known as the matrix product state (MPS) decomposition, and has been used quite frequently in quantum physics for decades [9, 10, 11].

Prior to elaborating on the concept of tensors decompositions, it will be beneficial to have an insight into the types of data that can be considered multidimensional. At present, multidimensional data is being generated at an incredible rate [12, 13, 14]. Examples of two-dimensional (2D) data may consist of grayscale images utilised in pattern recognition and computer vision problems [15, 16, 17, 18],

and gene expression data analysis [19, 20]. Sources of three-dimensional data (3D) may include brain-computer interface (BCI) Electroencephalography (EEG) measurements in biomedical engineering [21, 22], colour image recognition in computer vision and pattern recognition [23, 24], and hyperspectral imaging in mining [25] and surveillance [26]. Other examples of multidimensional data can also be found in data mining [27], predicting personalised tags in tag recommendation [28], and the study of quantum entanglement via tensor networks [29, 30].

Nearly a century ago the polyadic form of a tensor, i.e. expressing a higher-order tensor as the the summation of first-order tensors, was introduced and is the first account of a tensor decomposition [31, 32]. Later, in 1944 Cattell [33, 34] proposed "parallel proportional profiles" to correspond to psychological traits and the idea of multiple axes for analysis (objects, circumstances and features). These papers pioneered the CP decomposition, but it wasn't until 1970 where it was popularised when Carroll and Chang [35] introduced the canonical decomposition (CANDECOMP), and Harshman [36] proposed parallel factors (PARAFAC), into the psychometrics community. An alternative decomposition, the Tucker decomposition, was introduced by Ledyard Tucker in his seminal work on three-mode factor analysis [4]. TD was subsequently refined through the PhD thesis of Joseph Levin [37] and concurrently with Tucker's later papers [38, 39]. Confusion may arise in regards to TD because there have been many algorithms proposed that utilise the underlying structure of TD. These are, but not limited to, the higher-order singular value decomposition (HOSVD) [40], three-mode principal component analysis (3MPCA) [41], N-mode PCA [42] and N-mode SVD [16].
In comparison to MPS, both CP and TD have been applied to various fields decades earlier. Since the introduction of CP to psychometrics [35] in 1970, it has since been applied to chemometrics [43], statistics [44], sensor-array processing [45], neuroscience [46, 47], data mining [48, 49], image compression and classification [50], scientific computing [51, 52, 53] and applied mathematics [54]. Similarly, some applications of TD are in chemical analysis [55], Wiener filters in signal processing [56], facial recognition and human motion with TensorFaces [57, 58], video processing [59, 60], data mining [61, 62] and machine learning

[63, 64, 65].

Historically, the earliest representation that resembled the matrix product state (MPS) decomposition was in 1941, where the Kramers-Wannier approximation was used to study the two-dimensional classical Ising model [66]. Subsequent papers in statistical mechanics and magnetism also had similar forms to MPS [67, 10, 68], and only in 1992 was the MPS formalism introduced as "finitely correlated states" [69]. Today the MPS decomposition has been used quite extensively in physics research [70, 71, 9, 72, 73, 11, 74, 75, 76, 77, 78] yet only in 2011 had it been introduced to the applied mathematics community by Ivan Oseledets [8] with the name tensor-train (TT) decomposition. Since this inception to the society for industrial and applied mathematics (SIAM), there has been considerable research in this community on utilising MPS/TT for problems in numerical analysis [79, 80], scientific computing [81, 82, 83] and high-dimensional approximation [84, 85].

It is surprising that MPS has had minimal application to fields outside of physics and mathematics. There are unique aspects of MPS that would allow it to be advantageous for tensor-based machine learning and signal processing problems. Specifically, MPS has the ability to approximate an Nth-order tensor that depends only linearly with N and exponentially with its entanglement [9] or TT rank [8]. In comparison, TD/CP decompositions provide a global decomposition of an Nth-order tensor, which means its complexity grows exponentially with N as well as its Tucker rank [86]. Since TD/CP has been used in predominately more applications than MPS, a new approach to many of these TD/CP tensor-based problems via MPS/TT would be an interesting and important direction, which consequently could lead to many new results.

Data completion is the task of filling in missing entries of a partially observed array of data. In computer vision and graphics, the aim of image completion (or image inpainting) is to impute missing pixels of an image based only on a known subset of pixels [87, 86]. For two-dimensional (2D) grayscale images, the low rank matrix constraint could be used to capture the information within an

image, however, this method does not guarantee a globally optimal solution due to it being a nonconvex problem [88]. It was shown theoretically [89, 90, 91] that the trace norm or nuclear norm of a matrix can be used to approximate the matrix rank. Subsequently, efficient algorithms for low rank matrix completion problems based on nuclear norm minimisation were proposed by [92, 91]. Particularly, [91] had also shown that the trace norm is the tightest convex approximation for the rank of matrices.

Low rank tensor completion problems has been under considerable interest by researchers in recent years, this is because the rank of a tensor is not easily understood, and there is no standard method for the generalisation of the matrix rank to tensors. The determination of the CP rank (also known as tensor rank) is an NP-hard problem [93, 2]. An alternative is the Tucker rank, which consists of ranks of matrices that are based on an unbalanced matricization scheme [2]. The major problem of using an unbalanced matricization scheme is that the upper bound of the matrix rank is small, and may not be suitable for capturing the global information of a tensor. Despite these issues, there have been many recent works in utilising Tucker rank or CP rank for low rank tensor completion problems [86, 94, 95, 96, 97, 98], and there has been no approach to tensor completion problems via TT rank.

Tensor objects naturally reside in a high-dimensional vector space, and traditional machine learning classification methods that operate directly on this space can suffer from the so-called curse of dimensionality [99, 17]. Classifier performance tends to perform quite poorly in a high-dimensional space given a small amount of training samples, and handling samples of this size is computationally expensive. Regardless, tensor objects in most real-world applications would be constrained to a subspace or manifold of low dimension because entries of the tensor would often be highly correlated with its surrounding entries [100]. Hence, dimensionality reduction or feature extraction can be used to transform a tensor of high-dimensions to low-dimensions, which retains most of the relevant information of the original tensor space [99, 101].

There have been some recently proposed algorithms for tensor object feature ex-

traction. Discriminant analysis with tensor representation (DATER) [102] and multilinear discriminant analysis (MDA) [15] are dimensionality reduction algorithms. They essentially use an iterative scheme similar to an alternating least squares [103], with the aim of maximising a tensor-based discriminant criterion. The problems of these algorithms is that they are extremely sensitive to parameter adjustments and they also do not converge [104]. Multilinear principal component analysis (MPCA) [17] is a tensor feature extraction algorithm that generalises PCA to higher dimensions. It focuses on producing multilinear projections that can be applied onto the original tensor space, which projects it into a smaller subspace without losing the original order of the tensor. The projections are created based on unbalanced scatter matrices, which may not be efficient in terms of capturing the relevant information from the original tensor space. Uncorrelated multilinear discriminant analysis with regularization (R-UMLDA) [105] is an alternate tensor feature extraction method that projects the original tensor space to a vector space. The aim of R-UMLDA is to extract uncorrelated discriminative features directly from tensors by maximising a scatter ratio criterion. Not surprisingly, there are many other tensor-based methods [106, 107, 108, 109, 110] with the goal of finding a tensor-to-tensor (TTP) or tensor-to-vector (TVP) projection [1] to reduce the dimensionality of tensors, which can eventually be used for classification tasks. Interestingly, the solutions presented have used a CP/TD approach, i.e. the utilisation of CP or Tucker rank and/or decomposition as the core functionality of their algorithms. To the best of our knowledge, there has been no work in approaching tensor object dimensionality reduction and classification via MPS.

In the first part of this thesis, we introduce MPS as a completely new concept to tensor dimensionality reduction, tensor object classification and tensor completion. Firstly, it is shown that MPS can reduces the dimensionality of tensors much more efficiently than TD-based approaches because of the underlying TT rank. Using a thresholding operation by keeping only the relevant singular values from matricizations, MPS is still able to retain better features than TD. Subsequently, two algorithms are proposed to demonstrate the efficiency of an MPS

approach to tensor object classification. The first is called principal component analysis via tensor train (TTPCA), which combines MPS with PCA. The second is simply called MPS, which is a PCA alternative in the higher-dimensional case. Both algorithms provide high classification accuracies against popular tensor-based methods on object, face, gait and neuroscience recognition tasks.

For tensor completion, we show through the von Neumann entropy in quantum information theory [111] that the TT rank is able to capture the global correlations of a tensor much more effectively than the Tucker rank. This is mainly due to the TT rank consisting of ranks of matrices that are formed by a well balanced matricization scheme. Subsequently, we propose three algorithms: simple low-rank tensor completion via tensor train (SiLRTC-TT), parallel matrix factorisation via tensor train (TMac-TT), and concatenated image completion via tensor augmentation and completion (ICTAC). These algorithms demonstrate the superiority of using TT rank for tensor completion problems, which can be seen from their benchmark results against current state-of-the-art tensor completion algorithms in recovering images and videos with missing entries. Additionally, a tensor augmentation scheme is proposed called ket augmentation (KA) that is used to increase the dimensionality of a tensor without changing the total number of its entries. The purpose of KA is to provides a perfect means to obtain a higher-order tensor representation of visual data by maximally exploring the potential of TT rank-based optimization for colour image and video completion. By applying KA as a preprocessing technique prior to using the TMac-TT algorithm, it is shown to provide superb results for colour image and video completion problems. Furthermore, ICTAC is a specialised framework for image completion that uniquely concatenates a single image with missing entries to obtain a new tensor, which contains new structural properties to assist in completing the missing entries. Combining this with KA and TMac-TT, ICTAC provides the best results against present algorithms in image recovery problems.

Another contribution of this thesis is on target tracking with wireless sensor networks. Wireless sensor networks (WSNs) consist of spatially distributed wireless sensors, and are important in many engineering applications such as navigational

and guidance systems, battlefield surveillance, sonar ranging, process monitoring in industrial plants, and internet of things (IoT) [112, 113, 114, 115, 116, 117, 118]. The sensors in a sensor network send their local observations of a target to a fusion center (FC), which is done usually in an amplify-and-forward mode [119]. The purpose of the FC is to provide a global estimate of the position of the target. Sensors can be configured to be linear or nonlinear and depends on the input-output relation, e.g. nonlinear sensors could be ranging and/or bearing sensors for target localisation and tracking [115]. Targets are often assumed to be Gaussian, and in this case, Bayesian filtering can be defined via the first and second order statistical moments of the jointly Gaussian distributed source and observation [120]. Realistically, sensors are limited by their energy resources, and this issue has led to many researchers investigating sensor transmitter power allocation in linear sensor networks (LSNs) [121, 122, 123, 124]. The main aim of these works is to minimise estimate distortion in the FC, with the assumption that the targets are Gaussian and scalar. In [125], a tractable semidefinite program (SDP) for sensor power allocation was introduced to allow the FC to determine the best linear estimate in terms of the mean square error (MSE). This could be achieved for both linear or nonlinear sensor networks (NSNs), and for Gaussian targets that were scalar or vector, static or dynamic.

In the above aforementioned works, the wireless channel was assumed to be strong enough for low sensor transmit power, with path-loss, small-scale fading and shadowing not being taken into consideration [126]. Wireless relay nodes could be used to assist the communication between sensors and the FC, which acts as a wireless bridge. Multi-hop communication/relaying is already a standard in wireless broadband systems [127], yet there has been no exploration of relaying techniques in WSNs. Additionally, Gaussian mixture models (GMMs) have been known to better characterise target priors because they offer more flexibility in describing a target [128]. Recent works in signal processing [129, 130, 131, 132, 133] have shown GMM to be a indispensable tool. However, Bayesian filters with Gaussian mixture targets is computationally intractable in linear models because there is no closed-form solution for the MSE. This problem has been addressed

particularly in [131] via stochastic programming.

In the second part of this thesis, we address the problem of joint sensor and relay power allocation to optimize Bayesian filters. For our case we assume targets, which can be static or dynamic, have Gaussian mixture prior knowledge and the sensor networks can be configured as a LSN or NSN. To the best of our knowledge, this problem has not been considered in previous literature and our results demonstrate a computational tractable scheme based on an iterative algorithm, which converges to a stationary point after only a few iterations. Results demonstrate that utilizing a relay to act as a bridge between the sensors and FC leads to more accurate estimations at the FC compared to the cases with no relay present and/or no optimized power allocation, i.e. equal power is distributed at the sensors and relay.

The plan of the thesis is outlined as follows:

- In *Chapter 2*, a background of tensors is firstly given which introduces the concept of tensors, as well as other necessary mathematical preliminaries. An introduction to the Tucker and the matrix product state decomposition is followed, with an emphasis on MPS. This provides the necessary technical knowledge, including TT and Tucker rank, to understand our work. Lastly, a review on the Schmidt decomposition and the von Neumann entropy is provided.

- In *Chapter 3*, a more detailed overview of current tensor-based feature extraction algorithms is given. Then, a rigorous theoretical analysis of why MPS is more efficient than TD-based approaches for dimensionality reduction and classification is given. The optimised MPS approach is provided with extensive discussion on practical computation. Besides, two MPS-based algorithms are proposed, TTPCA and MPS, and a comparison of computational complexities is given to other tensor-based methods. Extensive experimental comparison on several datasets is provided that shows TTPCA and MPS being excellent approaches to tensor object recognition.

- In *Chapter 4*, a background is firstly presented on existing tensor completion techniques. It is then shown via von Neumann entropy that TT rank is much more efficient than Tucker rank in capturing the relevant correlations in tensors, which is significant for tensor completion. Subsequently, new tensor problem formulations are given based on balanced matricizations, and two new algorithms, SiLRTC-TT and TMac-TT are proposed to solve them. A dimensionality augmentation technique, KA, is also introduced as a preprocessing technique, which enables both algorithms to be competitive against state-of-the-art algorithms. A subsection provides extra work that enhances the results of image completion by establishing a new framework, which is known as ICTAC. ICTAC contains a preprocessing technique that concatenates a single image with missing pixels into a motionless video tensor. Reshaping and permuting this tensor allows for new patterns to arise, and this tensor is then applied with KA and TMac-TT. ICTAC is shown to provide extremely good results for completing images against the most current state-of-the-art algorithms.

- In *Chapter 5*, a contribution to power allocation in wireless sensor networks is presented. First, a background is given on the current methods and problems, then a new efficient algorithm is proposed. It provides the joint transmitter and relay power allocations to optimise a Bayesian filter, which is deployed at an FC for target estimation in a wireless sensor network. The focus is particularly on Gaussian mixture target priors, which can be scalar or vector, and the consideration of both linear and nonlinear sensor networks. The proposed method is the first to consider optimising Bayesian filters with Gaussian mixture target priors for joint sensor and relay power allocation. Experimental results demonstrate its computationally efficiency, and is more accurate in estimating targets compared to WSNs without a relay and/or equal sensor power allocation.

- *Chapter 6* provides the thesis summary and discusses potential applications of the findings as well as the future outlook.

Chapter 2

Background

2.1 Introduction to tensors

For this section we define the mathematical background for tensors.

2.1.1 Notation and preliminaries

A *tensor* is a multidimensional array and the number of dimensions in the array is also known as its *order*, *ways* or *modes*. It is important to highlight that the definition of tensors here is different from the definition of tensors in physics and engineering [134], which is also known as tensor fields [135].

Scalars are zero-order tensors and are denoted by lowercase letters, e.g. x, y, z. Naturally, first-order tensors are vectors which we represent as bold font lower case letters, e.g. $\mathbf{x}, \mathbf{y}, \mathbf{z}$. Matrices are second-order tensors that are denoted as uppercase letters, e.g. X, Y, Z. A higher-order tensor, which is a tensor of order greater than or equal to three, are denoted as calligraphic font uppercase letters, e.g. $\mathcal{X}, \mathcal{Y}, \mathcal{Z}$.

We also denote x_i, x_{ij} and $x_{i_1 \cdots i_N}$ as the ith entry $\mathbf{x}(i)$, (i,j)th entry $\mathbf{X}(i,j)$ and $(i_1, \cdots, i_N)$th entry $\mathcal{X}(i_1, \cdots, i_N)$ of vector $\mathbf{x}$, matrix $\mathbf{X}$ and higher-order tensor $\mathcal{X}$, respectively. The nth element in a sequence of tensors can be denoted as $X^{(n)}$ or X_n.

From the above definitions the general notation for a tensor is

$$\mathcal{X} \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_N}, \tag{2.1}$$

which is the tensor product of N real vector spaces, where $i_n = 1, \ldots, I_n$ represents the index for the nth vector space.

Subarrays can be formed by fixing certain indices in the tensor. For matrices, the row or column index can be fixed and we use a colon to indicate all elements of a mode. Consider a matrix $X \in \mathbb{R}^{I_1 \times I_2}$, then the i_2th column of X is denoted by $x_{:i_2}$, and the i_1th row of X is denoted by $x_{i_1:}$.

The mode-n fiber is a generalisation of the matrix row and columns. A fiber is defined by fixing all the indices of a tensor except one. The subarrays for matrix column and row defined above are known as mode-1 and mode-2 fibers, respectively. Hence, the mode-n fiber of the Nth order tensor in (2.1) is $x_{i_1 \ldots i_{n-1} : i_{n+1} \ldots i_N}$, where all indices are fixed except i_n.

Graphical notation will be used throughout this chapter to support the mathematical concepts of tensors. Fig 2.1 illustrates the simple graphical representation of tensors that is used extensively in physics.

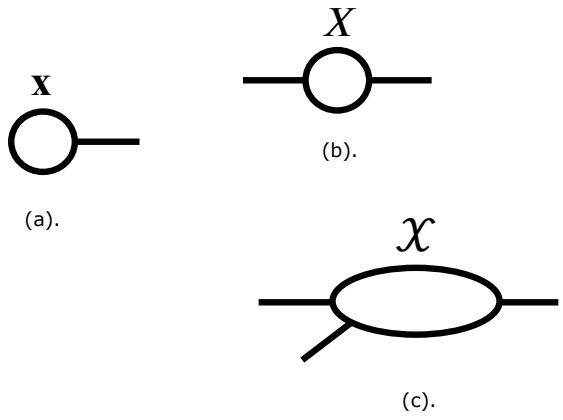


Figure 2.1 – Tensor graphical representation: (a) A vector with a single line protruding to represent its index. (b) A matrix with two protruding lines to represent row and column indices. (c) A third-order tensor.

2.1.2 Matricization

Mode-n matricization (also known as mode-n unfolding or flattening) of the tensor $\mathcal{X}$ in (2.1) is the process of unfolding or reshaping the tensor to a matrix

$$X_{(n)} \in \mathbb{R}^{I_n \times (I_1 \cdots I_{n-1} I_{n+1} \cdots I_N)}. \tag{2.2}$$

This transformation is performed by rearranging the mode-n fibers as the columns of the matrix $X_{(n)}$. Tensor element $(i_1, \ldots, i_{n-1}, i_n, i_{n+1}, \ldots, i_N)$ maps to matrix element (i_n, j) such that

$$j = 1 + \sum_{k=1, k \neq n}^{N} (i_k - 1) J_k \quad \text{with} \quad J_k = \prod_{m=1, m \neq n}^{k-1} I_m. \tag{2.3}$$

For the remainder of this thesis we refer to mode-n matricization as an *unbalanced matricization scheme* because it is comprised of a single mode and the product of the remaining modes for the row and column of the matrix, respectively. Additionally, the matrix rank of $X_{(n)}$ is denoted as $\text{rank}(X_{(n)})$, which is bounded by $\min(I_n, \prod_{l=1}^{N} I_l)$.

Mode-$(1, 2, \ldots, n)$ matricization of a tensor $\mathcal{X}$ is the unfolding of the tensor to a matrix

$$X_{[n]} \in \mathbb{R}^{r \times c} \quad (r = \prod_{l=1}^{n} I_l, c = \prod_{l=n+1}^{N} I_l). \tag{2.4}$$

We refer to mode-$(1, 2, \ldots, n)$ matricization as a *balanced matricization scheme* because for $n = I_N/2$ in (2.4), the matrix is approximately balanced in terms of the dimensions of r and c. The rank of $X_{[n]}$ is bounded by $\min(\prod_{l=1}^{n} I_l, \prod_{l=n+1}^{N} I_l)$.

2.1.3 Tensor multiplication: the n-Mode matrix product

Tensors can be multiplied at each mode n by a matrix. The n-Mode matrix product of a tensor $\mathcal{X} \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_N}$ with a matrix $A \in \mathbb{R}^{J \times I_n}$ results in a new tensor of size $I_1 \times \cdots \times I_{n-1} \times J \times I_{n+1} \times \cdots \times I_N$, which is denoted as $\mathcal{X} \times_n A$. Elementwise, it is described by

$$(\mathcal{X} \times_n A)_{i_1 \cdots i_{n-1} j i_{n+1} \cdots i_N} = \sum_{i_n=1}^{N} x_{i_1 \cdots i_n \cdots i_N} a_{j i_n}. \tag{2.5}$$

The graphical representation of the n-Mode matrix product is shown in Fig. 2.2.

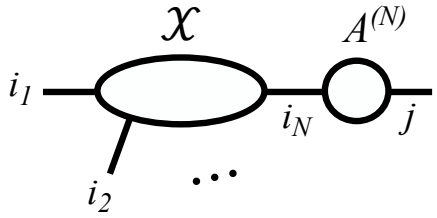


Figure 2.2 – The n-Mode matrix product of a tensor.

2.1.4 Matrix and tensor norms

The Frobenius norm of a tensor $\mathcal{X} \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_N}$ is defined as

$$||\mathcal{X}||_F = \sqrt{\sum_{i_1} \sum_{i_2} \cdots \sum_{i_N} x_{i_1 i_2 \cdots i_N}^2}. \tag{2.6}$$

Schatten p-norms arise when applying the p-norm to the vector of singular values of a matrix. Assuming the singular values of a matrix A are denoted as λ_i, then the Schatten p-norm is defined as

$$||A||_p = \left(\sum_{i=1}^{I} |\lambda_i|^p \right)^{1/p}, \quad 1 \leq p < \infty. \tag{2.7}$$

For this thesis we are interested in the case where $p = 1$, which results in the *trace norm* or nuclear norm of A. This is denoted as

$$||A||_* = \sum_{i=1}^{I} \lambda_i. \tag{2.8}$$

2.2 Tucker decomposition

The Tucker decomposition (TD) decomposes a tensor $\mathcal{X} \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_N}$ into two types of tensors: The first is an Nth-order *core tensor* $\mathcal{Y} \in \mathbb{R}^{P_1 \times P_2 \times \cdots \times P_N}$ with $P_n \leq I_n \; \forall n$. The second is a set of *common factor matrices* $\{A^{(n)}\}$ $(n = 1, \ldots, N)$ with $A^{(n)} \in \mathbb{R}^{I_n \times P_n}$, which can be thought of as the principal component at mode

n. For the case $P_n = I_n \ \forall n$, then TD is an exact decomposition of $\mathcal{X}$, which is denoted as

$$\mathcal{X} = \mathcal{Y} \times_1 A^{(1)} \times_2 A^{(2)} \cdots \times_N A^{(N)}. \tag{2.9}$$

Elementwise, (2.9) is denoted as

$$x_{i_1 i_2 \cdots i_N} = \sum_{i_1=1}^{N} \sum_{i_2=1}^{N} \cdots \sum_{i_N=1}^{N} y_{i_1 i_2 \cdots i_N} a^{(1)}_{i_1 p_1} a^{(2)}_{i_2 p_2} \cdots a^{(N)}_{i_N p_N}. \tag{2.10}$$

Therefore, it is straightforward to see that an approximation of $\mathcal{X}$ is given for $P_n < I_n$. The graphical representation of TD is depicted in Fig. 2.3.

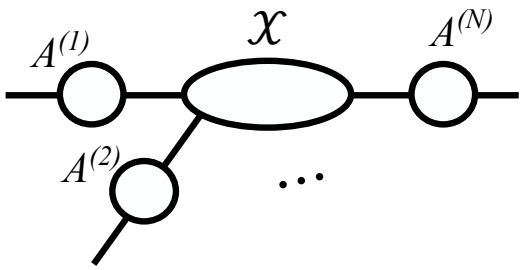


Figure 2.3 – The Tucker decomposition.

Consider a vector $\mathbf{p} = (\text{rank}(X_{(1)}), \text{rank}(X_{(2)}), \ldots, \text{rank}(X_{(N)}))$, then this vector is known as the *Tucker rank* of the tensor $\mathcal{X}$, which consists of unbalanced matrices for each mode in the TD.

Generally, TD is not unique and imposing different constraints will lead to alternate decompositions. By assuming the core tensor $\mathcal{Y}$ in (2.9) contains non-zero elements only on its super diagonal, then we obtain the CP/PARAFAC decomposition. The higher-order singular value decomposition (HOSVD) or higher-order orthogonal iteration (HOOI) algorithms are based on TD with the constraint that the common factor matrices are orthogonal.

2.3 Matrix product state decomposition

The matrix product state decomposition (MPS) is a tensor decomposition that has been used predominantly in physics and mathematics. [9, 72] utilised MPS

for restricting the amount of quantum entanglement in a multipartite quantum mechanical system so that it could be efficiently simulated on a classical computer, which paved the way for many experimental and theoretical works in physics. In this section we define MPS and analyse in detail some useful properties that will be important for the proposed algorithms in this thesis.

2.3.1 MPS formulations

Given a tensor $\mathcal{X} \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_N}$, its general MPS representation in terms of its elements is given as

$$x_{i_1 i_2 \cdots i_N} = A^{(i_1)} A^{(i_2)} \cdots A^{(i_N)}, \tag{2.11}$$

where for each $i_n = 1, \ldots, I_n$, there is a corresponding matrix $A^{(i_n)} \in \mathbb{R}^{R_{n-1} \times R_n}$ ($R_0 = R_{N+1} = 1$). The index i_n is often referred as a *physical index*, and the index r_n is known as the *bond index* in physics [136], or *compression rank* according to [8]. If an assumption is made such that (s.t.) $I_1 = I_2 = \cdots = I_N = I$ and $R_1 = R_2 = \cdots R_N = R$, then the MPS contains approximately $(N-2)R^2 I + 2RI$ parameters, which demonstrates that MPS is represented by a number of parameters that increases only polynomially with N. The *TT rank* of an MPS is a vector $\mathbf{r} = (\text{rank}(X_{[1]}), \text{rank}(X_{[2]}), \ldots, \text{rank}(X_{[N-1]}))$, which consists of the balanced matricizations of $\mathcal{X}$. A graphical representation of MPS is demonstrated in Fig. 2.4.

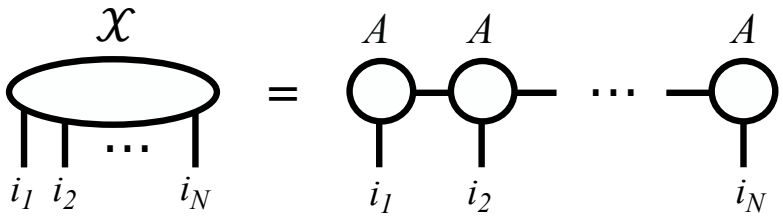


Figure 2.4 – The matrix product state decomposition.

Assuming $\mathbb{I}$ is an identity matrix, the MPS in (2.11) can be defined in terms of a *canonical form* if the component matrices $A^{(n)}$ satisfy a left-orthonormal

constraint

$$\sum_{n=1}^{N}(A^{(i_n)})^{\dagger}A^{(i_n)} = \mathbb{I}, \tag{2.12}$$

in which (2.11) is known as a *left-canonical* MPS, or a right-orthonormal constraint

$$\sum_{n=1}^{N}A^{(i_n)}(A^{(i_n)})^{\dagger} = \mathbb{I}, \tag{2.13}$$

which is called *right-canonical* MPS. Furthermore, a *mixed-canonical* MPS is defined as (2.11) containing both left- and right- orthonormal constraints.

Prior to demonstrating the calculation of MPS forms, it would be beneficial to have a brief review of the singular value decomposition (SVD) because MPS is essentially an iterative SVD algorithm.

Given an arbitrary rectangular matrix with real entries $A \in \mathbb{R}^{I_1 \times I_2}$, then it can have the following form via SVD

$$A = USV^{\dagger}, \tag{2.14}$$

where '$\dagger$' represents complex conjugation and

- $U \in \mathbb{R}^{I_1 \times \min(I_1,I_2)}$ has orthonormal columns s.t. $U^{\dagger}U = I$. If $I_1 \leq I_2$, then U is unitary, which implies $UU^{\dagger} = I$.
- $S \in \mathbb{R}^{\min(I_1,I_2)\times\min(I_1,I_2)}$ is a diagonal matrix with nonnegative entries which are known as *singular values*. The number of non-zero singular values r is the rank of A. For the remainder of this thesis we assume the entries are in descending order s.t. s_{11} is the largest singular value.
- $V^{\dagger} \in \mathbb{R}^{\min(I_1,I_2)\times I_2}$ has orthonormal rows s.t. $V^{\dagger}V = I$. If $I_1 \geq I_2$, then $V^{\dagger}$ is unitary, which implies $VV^{\dagger} = I$.

The remaining subsections will focus on the calculation of the canonical MPS forms and the useful notation of [9]. To make this thesis more self contained, we explain the concepts in a straightforward manner without relating to physics nomenclature.

2.3.2 Left-canonical MPS

The general MPS form in (2.11) can be converted into the left-canonical MPS using SVD's. Consider the Nth-order tensor $\mathcal{X}$, then let $M = X_{[1]} \in \mathbb{R}^{I_1 \times (I_2 \cdots I_N)}$. Applying the SVD on the matrix M gives

$$m_{i_1(i_2 \cdots i_N)} = \sum_{r_1=1}^{R_1} U_{i_1 r_1} S_{r_1 r_1} (V^\dagger)_{r_1(i_2 \cdots i_N)}. \tag{2.15}$$

From the above equation, it is trivial to see that $U \in \mathbb{R}^{I_1 \times R_1}$ fulfils the left-orthonormal constraint in (2.12) that we need for the left-canonical form, therefore we let $A^{(i_1)} = U_{i_1 r_1}$. To form the next component $A^{(i_2)}$, we obtain the matrix $C = SV^\dagger$ from (2.15), and reshape to size $(R_1 I_2) \times (I_3 \cdots I_N)$. Now let $M = C \in \mathbb{R}^{(R_1 I_2) \times (I_3 \times \cdots I_N)}$, and we apply the SVD s.t.

$$m_{(r_1 i_2)(i_3 \cdots i_N)} = \sum_{r_2=1}^{R_2} U_{(r_1 i_2) r_2} S_{r_2 r_2} (V^\dagger)_{r_2(i_3 \cdots i_N)}, \tag{2.16}$$

with $R_2 \leq R_1 I_2$. Reshaping the left-orthogonal matrix U in (2.16) to a third order tensor, then the second component of our MPS is given

$$A^{(i_2)} = U^{(i_2)} \in \mathbb{R}^{R_1 \times R_2}. \tag{2.17}$$

Iterating this procedure, we obtain all component tensors of the MPS. After the full sweep (from left-to-right) of the indices of $\mathcal{X}$, the general state of the MPS in terms of its bond dimensions can be written (element-wise) as

$$x_{i_1 i_2 \cdots i_N} = \sum_{r_1=1}^{R_1} \sum_{r_2=1}^{R_2} \cdots \sum_{r_{N-1}=1}^{R_{N-1}} a_{r_1}^{(i_1)} a_{r_1 r_2}^{(i_2)} \cdots a_{r_{N-1}}^{(i_N)}, \tag{2.18}$$

or in a matrix product representation equivalent to (2.11). Fig. 2.5 shows the graphical representation of the above left-canonical MPS procedure for an arbitrary Nth-order $\mathcal{X}$.

2.3.3 Right-canonical MPS

The right-canonical MPS is obtained similar to the left-canonical MPS, except that we now perform the SVD sweep from right to left of the indices of $\mathcal{X}$. The

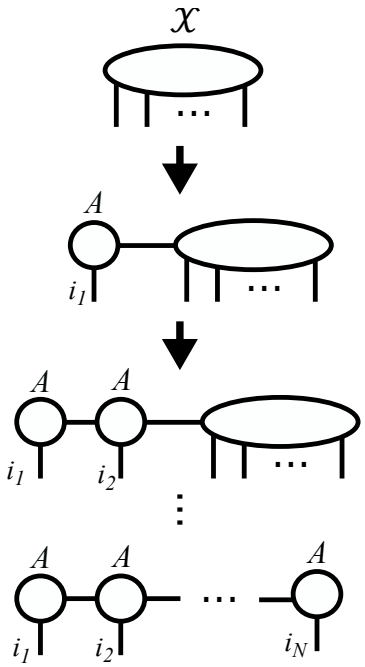


Figure 2.5 – A sequence of SVD's from the first to last index constructs the left-canonical MPS.

goal is to obtain

$$x_{i_1 i_2 \cdots i_N} = B^{(i_1)} B^{(i_2)} \cdots B^{(i_N)} \tag{2.19}$$

or

$$x_{i_1 i_2 \cdots i_N} = \sum_{r_1=1}^{R_1} \sum_{r_2=1}^{R_2} \cdots \sum_{r_{N-1}=1}^{R_{N-1}} b_{r_1}^{(i_1)} b_{r_1 r_2}^{(i_2)} \cdots b_{r_{N-1}}^{(i_N)}, \tag{2.20}$$

where the components of the MPS $B^{(n)}$ satisfy right-orthonormal constraints. The MPS of $\mathcal{X}$ is obtained in the following way. Let $M = X_{[I_{N-1}]} \in \mathbb{R}^{(I_1 \cdots I_{N-1}) \times I_N}$,

then subsequently the MPS representation is as follows:

$$
\begin{aligned}
x_{i_1 i_2 \cdots i_N} &= m_{(i_1 \cdots i_{N-1}) i_N} && (2.21)\\
&= \sum_{r_{N-1}=1}^{R_{N-1}} U_{(i_1 \cdots i_{N-1}) r_{N-1}} S_{r_{N-1} r_{N-1}} (V^\dagger)_{r_{N-1} i_N} && (2.22)\\
&= \sum_{r_{N-1}=1}^{R_{N-1}} M_{(i_1 \cdots i_{N-2})(i_{N-1} r_{N-1})} b^{(i_N)}_{r_{N-1}} && (2.23)\\
&= \sum_{r_{N-1}=1}^{R_{N-1}} \sum_{r_{N-2}=1}^{R_{N-2}} U_{(i_1 \cdots i_{N-2}) r_{N-2}} S_{r_{N-2} r_{N-2}} (V^\dagger)_{r_{N-2}(i_{N-1} r_{N-1})} b^{(i_N)}_{r_{N-1}} && (2.24)\\
&= \sum_{r_{N-1}=1}^{R_{N-1}} \sum_{r_{N-2}=1}^{R_{N-2}} M_{(i_1 \cdots i_{N-3})(i_{N-2} r_{N-2})} b^{(i_{N-1})}_{r_{N-2} r_{N-1}} b^{(i_N)}_{r_{N-1}} && (2.25)
\end{aligned}
$$

and we can see that by iteratively performing this SVD sweep we will obtain the form of (2.20). The graphical representation is illustrated in Fig. 2.6 for $\mathcal{X}$.

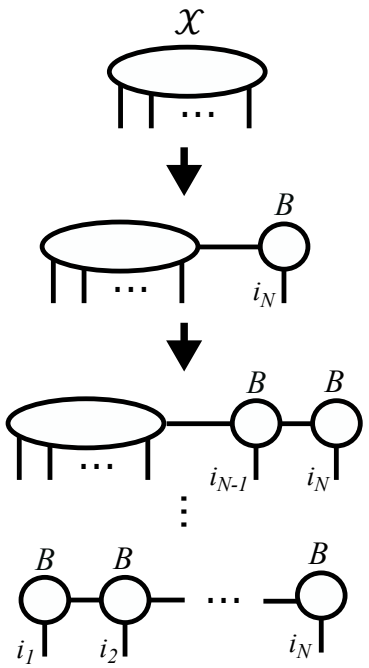


Figure 2.6 – A sequence of SVD's from the last to the first index constructs the right-canonical MPS.

2.3.4 Mixed-canonical MPS

Mixing both left- and right-canonical forms we can obtain the mixed-canonical MPS, which is denoted as

$$x_{i_1 i_2 \cdots i_N} = \sum_{r_1=1}^{R_1} \sum_{r_2=1}^{R_2} \cdots \sum_{r_{N-1}=1}^{R_{N-1}} a_{r_1}^{(i_1)} \cdots a_{r_{k-1} r_k}^{(i_k)} S_{r_k r_k} b_{r_k r_{k+1}}^{(i_{k+1})} \cdots b_{r_{N-1}}^{(i_N)} \tag{2.26}$$

or

$$x_{i_1 i_2 \cdots i_N} = A^{(i_1)} \cdots A^{(i_k)} S B^{(i_{k+1})} \cdots B^{(i_N)}, \tag{2.27}$$

where the set of tensors $\{A^{(i_n)}\}_{n=1}^{k}$ fulfil the left-orthonormal constraint in (2.12) , and the set $\{B^{(i_n)}\}_{n=k+1}^{N}$ fulfil the right-orthonormal constraint in (2.13). The matrix S is diagonal and contains singular values.

The mixed-canonical form is computed in the following manner: Assume the left-canonical sweep has been completed up to k s.t.

$$x_{i_1 i_2 \cdots i_N} = \sum_{r_1=1}^{R_1} \cdots \sum_{r_k=1}^{R_k} (a_{r_1}^{(i_1)} \cdots a_{r_{k-1} r_k}^{(i_k)}) S_{r_k r_k} (V^\dagger)_{r_k (i_{k+1} \cdots i_N)}. \tag{2.28}$$

Then, reshape $V^\dagger$ to a matrix $M \in \mathbb{R}^{(R_k I_{k+1} \cdots I_{N-1}) \times I_N}$ and carry out the right-to-left sweep of successive SVD's until we obtain the final SVD

$$m_{(r_{k+1} i_{k+1})(i_{k+2} r_{k+2})} = \sum_{r_{k+1}=1}^{R_{k+1}} U_{(r_k i_{k+1}) r_{k+1}} S_{r_{k+1} r_{k+1}} (V^\dagger)_{r_{k+1} (i_{k+2} r_{k+2})}, \tag{2.29}$$

where reshaping $(V^\dagger)$ in (2.29) to a third-order tensor obtains $B^{(i_{k+2})} \in \mathbb{R}^{R_{k+1} \times R_{k+2}}$. Finally US in (2.29) is reshaped to a third-order tensor to get $B^{(i_{k+1})} \in \mathbb{R}^{R_k \times R_{k+1}}$. Then from (2.28),

$$\begin{aligned} (V^\dagger)_{r_k (i_{k+1} \cdots i_N)} &= \sum_{r_{k+1}=1}^{R_{k+1}} \cdots \sum_{r_{N-1}=1}^{R_{N-1}} b_{r_k r_{k+1}}^{(i_{k+1})} \cdots b_{r_{N-1}}^{(i_N)} && (2.30) \\ &= B^{(i_{k+1})} \cdots B^{(i_N)}. && (2.31) \end{aligned}$$

It is trivial to see that the component tensors $\{B^{(i_n)}\}$ from $n = k+2, \ldots, N$ follow the right-orthonormal constraint due to the successive SVD's in the right-to-left sweep, however, it is not so straightforward to see this for $B^{(k+1)}$. Following from

the right-orthonormal constraint that $V^\dagger V = I$, then

$$\begin{aligned} I &= V^\dagger V && (2.32)\\ &= (B^{(i_{k+1})} \cdots B^{(i_N)})(B^{\dagger(i_N)} \cdots B^{\dagger(i_{k+1})}) && (2.33)\\ &= B^{(i_{k+1})} \cdots B^{(i_N)} B^{\dagger(i_N)} \cdots B^{\dagger(i_{k+1})} && (2.34)\\ &= B^{(i_{k+1})} B^{\dagger(i_{k+1})}, && (2.35) \end{aligned}$$

where the right-orthonormal constraint of $\{B^{(i_n)}\}$ from $n = k+2, \ldots, N$ is used.

2.3.5 Vidal's decomposition

A popular decomposition introduced in [9] allows for the easy transition between left-, right- and mixed-canonical MPS forms. We follow the notation of Vidal's original work to separate between the other forms.

The essential idea of this decomposition is to let an Nth-order tensor $\mathcal{X}$ be represented in the below form:

$$x_{i_1 i_2 \cdots i_N} = \Gamma^{(1)}_{i_1} \lambda^{(1)} \Gamma^{(2)}_{i_2} \lambda^{(2)} \cdots \lambda^{(N-1)} \Gamma^{(N)}_{i_N}, \tag{2.36}$$

which is graphically depicted in Fig. 2.7. Similarly, each i_n is a matrix $\Gamma^{(n)}_{i_n} \in \mathbb{R}^{I_{n-1} \times I_n}$, and $\lambda^{(n)}$ corresponds to a $R_n \times R_n$ diagonal matrix of singular values.

Vidal's notation in (2.36) can be obtained in the following way. Similar to left-

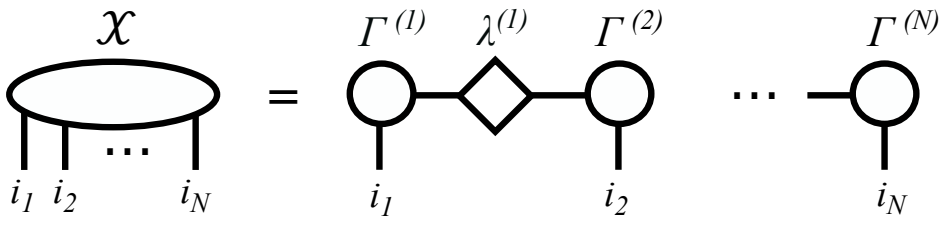


Figure 2.7 – The MPS based on Vidal's decomposition.

canonical MPS, reshape $\mathcal{X}$ to a matrix $M \in \mathbb{R}^{I_1 \times (I_2 \cdots I_N)}$. Then using iterative

SVD's from a left to right sweep we have

$$
\begin{aligned}
x_{i_1 i_2 \cdots i_N} &= m_{i_1 (i_2 \cdots i_N)} && (2.37)\\
&= \sum_{r_1=1}^{R_1} a^{(1)}_{i_1 r_1} \lambda^{(1)}_{r_1 r_1} (V^\dagger)_{r_1 (i_2 \cdots i_N)} && (2.38)\\
&= \sum_{r_1=1}^{R_1} \Gamma^{(1)}_{i_1 r_1} m_{(r_1 i_2)(i_3 \cdots i_N)} && (2.39)\\
&= \sum_{r_1=1}^{R_1} \sum_{r_2=1}^{R_2} \Gamma^{(1)}_{i_1 r_1} a^{(2)}_{r_1 i_2 r_2} \lambda^{(2)}_{r_2 r_2} (V^\dagger)_{r_2 (i_3 \cdots i_N)} && (2.40)\\
&= \sum_{r_1=1}^{R_1} \sum_{r_2=1}^{R_2} \Gamma^{(1)}_{i_1 r_1} \lambda^{(1)}_{r_1 r_1} \Gamma^{(2)}_{r_1 i_2 r_2} m_{(r_2 i_3)(i_4 \cdots i_N)} && (2.41)\\
&= \sum_{r_1=1}^{R_1} \cdots \sum_{r_3=1}^{R_3} \Gamma^{(1)}_{i_1 r_1} \lambda^{(1)}_{r_1 r_1} \Gamma^{(2)}_{r_1 i_2 r_2} a^{(3)}_{r_2 i_3 r_3} \Gamma^{(3)}_{r_3 r_3} (V^\dagger)_{r_3 (i_4 \cdots i_N)} && (2.42)\\
&= \sum_{r_1=1}^{R_1} \cdots \sum_{r_3=1}^{R_3} \Gamma^{(1)}_{i_1 r_1} \lambda^{(1)}_{r_1 r_1} \Gamma^{(2)}_{r_1 i_2 r_2} \lambda^{(2)}_{r_2 r_2} \Gamma^{(3)}_{r_2 i_3 r_3} m_{(r_3 i_4)(i_5 \cdots i_N)} && (2.43)\\
&\vdots && (2.44)\\
&= \sum_{r_1=1}^{R_1} \cdots \sum_{r_{N-1}=1}^{R_{N-1}} \Gamma^{(1)}_{i_1 r_1} \lambda^{(1)}_{r_1 r_1} \Gamma^{(2)}_{r_1 i_2 r_2} \cdots \lambda^{(N-1)}_{r_{N-1} r_{N-1}} \Gamma^{(N)}_{r_{N-1} i_N}. && (2.45)
\end{aligned}
$$

The main difference between this decomposition and left-canonical MPS is that the component tensors $\{A^{(i_n)}\}$ are decomposed with the knowledge of $\lambda^{(n-1)}$ from the previous SVD. Specifically

$$A^{(i_n)}_{r_{n-1} r_n} = \lambda^{(n-1)}_{r_{n-1} r_{n-1}} \Gamma^{(n)}_{r_{n-1} i_n r_n}, \tag{2.46}$$

which implies divisions by the diagonal elements of $\lambda^{(n-1)}$. An alternative way to obtain Vidal's decomposition in (2.45) is to perform the left-canonical MPS, store all the singular value matrices $\{\lambda^{(l)}\}_{l=1}^{N-1}$, then insert afterwards $\lambda^{(l)}(\lambda^{(l)})^{-1}$ between neighbouring A matrices $A^{(i_l)}$ and $A^{(i_{l+1})}$.

Similarly, the decomposition can be obtained via right-orthonormal constraints on the set of B matrices s.t.

$$B^{(i_n)}_{r_{n-1} r_n} = \Gamma^{(n)}_{r_{n-1} i_n r_n} \lambda^{(n)}_{r_n r_n}, \tag{2.47}$$

where for both (2.46) and (2.47), for notational simplification, $\lambda^{(0)}$ and $\lambda^{(N)}$ are both scalars equal to 1. For convenience, a graphical representation of the Vidal

notation and how the left- and right-canonical notation is represented in this form is illustrated in Fig. 2.8.

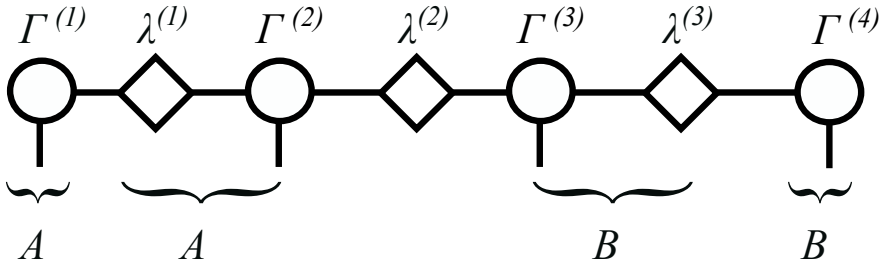


Figure 2.8 – Graphical representation of Vidal's decomposition and the right- and left-canonical conversions. Adjacent Γ and λ can be converted to A or B notation, that have either left- or right-orthonormal constraints.

The sets of A and B tensors can be expressed into the left- and right-orthonormal conditions in terms of the $\Gamma\lambda$ notation. Therefore, the left-orthonormal condition is

$$\mathbb{I} = \sum_{n=1}^{N} (A^{(i_n)})^\dagger A^{(i_n)} = \sum_{n=1}^{N} \Gamma^{\dagger(n)} \lambda^{\dagger(n-1)} \lambda^{(n-1)} \Gamma^{(n)}, \tag{2.48}$$

and the right-orthonormal condition in terms of the B matrices is

$$\mathbb{I} = \sum_{n=1}^{N} B^{(i_n)} (B^{(i_n)})^\dagger = \sum_{n=1}^{N} \Gamma^{(n)} \lambda^{(n)} \lambda^{\dagger(n)} \Gamma^{\dagger(n)}. \tag{2.49}$$

2.4 Measures of entropy

The von Neumann entropy is a useful tool in quantum information theory for describing the uncertainty associated with quantum states. It will be shown that it is also useful for applications outside of quantum physics, which reduces to the well-known Shannon entropy. Prior to introducing von Neumann entropy, we review the Schmidt decomposition.

2.4.1 Schmidt decomposition

Suppose that $\mathbf{x}_{AB}$ is a normalised vector of a composite system of real vector spaces s.t. $\mathbb{R}_{AB} = \mathbb{R}_A \otimes \mathbb{R}_B$, and the dimension of the vector space is $I_{AB} = I_A I_B$.

Then there exists orthonormal bases $\mathbf{u}_A^{(k)} \in \mathbb{R}_A^{I_A}$ and $\mathbf{u}_B^{(k)} \in \mathbb{R}_B^{I_B}$ s.t.

$$\mathbf{x}_{AB} = \sum_{k=1}^{K} s_k \mathbf{u}_A^{(k)} \otimes \mathbf{u}_B^{(k)}, \tag{2.50}$$

where '$\otimes$' denotes the tensor product, $K = \min(\dim(\mathbb{R}_A), \dim(\mathbb{R}_B))$ is the *Schmidt rank*, $\{s_k\}_{k=1}^K$ are *Schmidt coefficients* with the property of being non-negative real numbers satisfying $\sum_{k=1}^K s_k = 1$, and $\mathbf{u}_A^{(k)}$ and $\mathbf{u}_B^{(k)}$ are the *Schmidt bases* for subsystem A and B, respectively.

The proof of the decomposition is quite simple. Let $\mathbf{e}_{i_A}$ and $\mathbf{e}_{i_B}$ be any fixed orthonormal bases for systems A and B, respectively. Then $\mathbf{x}_{AB}$ can be written

$$\mathbf{x}_{AB} = \sum_{i_A=1}^{I_A} \sum_{i_B=1}^{I_B} a_{i_A i_B} \mathbf{e}_{i_A} \otimes \mathbf{e}_{i_B}, \tag{2.51}$$

where $a_{i_A i_B}$ is matrix of real numbers. Using the SVD s.t. $A = USV^\dagger$, then

$$\mathbf{x}_{AB} = \sum_{i_A=1}^{I_A} \sum_{i_B=1}^{I_B} \sum_{k=1}^{R} u_{i_A k} s_{kk} v^*_{k i_B} \mathbf{e}_{i_A} \otimes \mathbf{e}_{i_B}. \tag{2.52}$$

Let $\mathbf{u}_A^{(k)} = \sum_{k=1}^K u_{i_A k} \mathbf{e}_{i_A}$ and $\mathbf{u}_B^{(k)} = \sum_{k=1}^K v_{k i_B} \mathbf{e}_{i_B}$, which are orthonormal because of the orthonormality properties of U and $V^\dagger$. Furthermore, $s_k = s_{kk}$ because s_{kk} is a diagonal matrix, then (2.52) becomes

$$\mathbf{x}_{AB} = \sum_{k=1}^{K} s_k \mathbf{u}_A^{(k)} \otimes \mathbf{u}_B^{(k)}, \tag{2.53}$$

which ends the proof.

2.4.2 The von Neumann entropy

The *von Neumann entropy* is a key concept utilised in quantum information theory. It measures the uncertainty of a quantum state in terms of a density matrix. In classical information theory, the concept of *Shannon entropy* is important for measuring communication capacity.

Formally, the von Neumann entropy is defined in the following way. Consider the matrix X_{AB} being the composite of two subsystems A and B of a real vector

space. Its Schmidt decomposition is denoted as

$$\mathbf{x}_{AB} = \sum_{k=1}^{K} s_k \mathbf{u}_A^{(k)} \otimes \mathbf{u}_B^{(k)}, \tag{2.54}$$

then the correlations between both subsystems can be studied via the von Neumann entropy as

$$S(Z_A) = -\text{Trace}(Z_A \log_2 Z_A), \tag{2.55}$$

where Z_A is the *reduced density matrix* of the composite system and computed by taking the partial trace of the matrix Z_{AB} with respect to B. Specifically,

$$\begin{aligned} Z_{AB} &= X_{AB} \otimes (X_{AB})^T & (2.56)\\ &= (\sum_{k=1}^{K} s_k \mathbf{u}_A^{(k)} \otimes \mathbf{u}_B^{(k)}) \otimes (\sum_{k=1}^{K} s_k \mathbf{u}_A^{(k)} \otimes \mathbf{u}_B^{(k)})^T. & (2.57) \end{aligned}$$

Then Z_A is computed as

$$\begin{aligned} Z_A &= \text{Trace}_B(Z_{AB}) & (2.58)\\ &= \sum_{k=1}^{K} s_k^2 \mathbf{u}_A^{(k)} \otimes (\mathbf{u}_A^{(k)})^T. & (2.59) \end{aligned}$$

Substituting (2.59) to (2.55), we obtain

$$S(Z_A) = -\sum_{k=1}^{K} s_k^2 \log_2 s_k^2. \tag{2.60}$$

Similarly,

$$\begin{aligned} S(Z_B) &= -\text{Trace}(Z_B \log_2 Z_B) & (2.61)\\ &= -\sum_{k=1}^{K} s_k^2 \log_2 s_k^2, & (2.62) \end{aligned}$$

therefore $S(Z_A) = S(Z_B) = S$. This entropy S reflects the amount of *correlations* or *degree of entanglement* between the contiguous subsystems, and is bounded by

$$0 \leq S \leq \log_2 K. \tag{2.63}$$

Therefore, the von Neumann entropy uses the singular values s_k of X_{AB}. There are no correlations when $S = 0$, which is the case when $s_1 = 1$ and the remaining

singular values are zero. There exists maximum correlations when $S = \log_2 K$, which is the case when $s_1 = s_2 \cdots = s_K = 1/\sqrt{s_k}$. Interestingly, in the case that the singular values decay significantly, S' can be the von Neumann entropy based on keeping only a few of the largest singular values. Comparing this to the case where all singular values are kept, in which the von Neumann entropy is S, then $|S - S'|^2$ will be minimal.

Chapter 3

Matrix product states for tensor-based machine learning

There is an increasing need to handle large multidimensional datasets that cannot efficiently be analyzed or processed using modern day computers. Due to the curse of dimensionality it is urgent to develop mathematical tools which can evaluate information beyond the properties of large matrices [1]. The essential goal is to reduce the dimensionality of multidimensional data, represented by tensors, with a minimal information loss by compressing the original tensor space to a lower-dimensional tensor space, also called the feature space [1]. Tensor decomposition is the most natural tool to enable such compressions [2].

Until recently, tensor compression is merely based on Tucker decomposition (TD) [39], also known as higher-order singular value decomposition (HOSVD) when orthogonality constraints on factor matrices are imposed [40]. TD is also an important tool for solving problems related to feature extraction, feature selection and classification of large-scale multidimensional datasets in various research fields. Its well-known application in computer vision was introduced in [16] to analyze some ensembles of facial images represented by fifth-order tensors. In data mining, the HOSVD was also applied to identify handwritten digits [61]. In addition, the HOSVD has been applied in neuroscience, pattern analysis, image classification and signal processing [63, 137, 138]. The higher-order orthogonal iteration

(HOOI) [103] is an alternating least squares (ALS) for finding the TD approximation of a tensor. Its application to independent component analysis (ICA) and simultaneous matrix diagonalization was investigated in [139]. Another TD-based method is multilinear principal component analysis (MPCA) [17], an extension of classical principal component analysis (PCA), which is closely related to HOOI. Meanwhile, TD suffers the following conceptual bottlenecks in tensor compression:

- *Computation.* TD compresses an Nth-order tensor in tensor space $\mathbb{R}^{I_1 \times I_2 \times \cdots I_N}$ of large dimension $I = \prod_{j=1}^{N} I_j$ to its Nth-order core tensor in a tensor space $\mathbb{R}^{\Delta_1 \times \Delta_2 \times \cdots \Delta_N}$ of smaller dimension $N_f = \prod_{j=1}^{N} \Delta_j$ by using N factor matrices of size $I_j \times \Delta_j$. Computation of these N factor matrices is computationally intractable. Instead, each factor matrix is alternatingly optimised with all other $N-1$ factor matrices held fixed, which is still computationally expensive. Practical application of the TD-based compression is normally limited to small-order tensors.

- *Compression quality.* TD is an effective representation of a tensor only when the dimension of its core tensor is fairly large [2]. Restricting dimension $N_f = \prod_{j=1}^{N} \Delta_j$ to a moderate size for tensor classification results in significant lossy compression, making TD-based compression a highly heuristic procedure for classification. It is also almost impossible to tune $\Delta_j \leq I_j$ among $\prod_{j=1}^{N} \Delta_j \leq \bar{N}_f$ for a prescribed $\bar{N}_f$ to have a better compression.

In this chapter, the matrix product state (MPS) decomposition [140, 9, 72, 11] is introduced as a new method to compress tensors, which fundamentally circumvent all the above bottlenecks of TD-based compression. Namely,

- *Computation.* The MPS decomposition is fundamentally different from the TD in terms of its geometric structure as it is made up of local component tensors with maximum order three. Consequently, using the MPS decomposition for large higher-order tensors can potentially avoid the computational

bottleneck of the TD and related algorithms. Computation for orthogonal common factors in MPS is based on successive SVDs without any recursive local optimisation procedure and is very efficient with low-cost.

- *Compression quality.* MPS compresses Nth-order tensors to their core matrices of size $\mathbb{R}^{N_1 \times N_2}$. The dimension $N_f = N_1 N_2$ can be easily tuned to a moderate size with minimum information loss by pre-positioning the core matrix in the MPS decomposition.

MPS has been proposed and applied to study quantum many-body systems with great success, prior to its introduction to the mathematics community under the name tensor-train (TT) decomposition [8]. However, to the best of our knowledge its application to machine learning and pattern analysis has not been proposed.

The main contributions are summarised as follows:

- Propose MPS decomposition as a new and systematic method for compressing tensors of arbitrary order to matrices of moderate dimension, which circumvents all existing bottlenecks in tensor compression;
- Develop a right form of MPS decomposition to optimise the dimensionality of the core matrices. Implementation issues of paramount importance for practical computation are discussed in detail. These include tensor mode permutation, tensor bond dimension control, and positioning the core matrix in MPS;
- Extensive experiments are performed along with comparisons to existing state-of-the-art tensor classification methods to show its advantage.

The chapter is structured as follows. Section 3.1 provides a rigorous mathematical analysis comparing MPS and TD in the context of tensor compression for classification. Section 3.2 is a complete and comprehensive outline of the MPS approach to tensor feature extraction and classification, which includes discussions

of important practical computation as well a computational complexity analysis comparing MPS to HOOI, MPCA and uncorrelated multilinear discriminant analysis with regularization (R-UMLDA) [105]. In Section 3.3, experimental results are shown to benchmark all algorithms in classification performance and training time. Lastly, Section 3.4 concludes the chapter.

3.1 MPS decomposition vs TD decomposition in tensor compression

The problem of tensor compression for supervised learning is the following:

Based on K training Nth-order tensors $\mathcal{X}^{(k)} \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_N}$ $(k = 1, 2, \ldots, K)$, find common factors to compress both training tensor $\mathcal{X}^{(k)}$ and test tensors $\mathcal{Y}^{(\ell)}$ $(\ell = 1, \cdots, L)$ to a feature space of moderate dimension to enable classification.

Until now, only TD has been proposed to address this problem [63]. More specifically, the K training sample tensors are firstly concatenated along the mode $(N+1)$ to form an $(N+1)$th-order tensor $\mathcal{X}$ as

$$\mathcal{X} = [\mathcal{X}^{(1)} \mathcal{X}^{(2)} \cdots \mathcal{X}^{(K)}] \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_N \times K}. \tag{3.1}$$

TD-based compression such as HOOI [103] is then applied to have the approximation

$$\mathcal{X} \approx \mathcal{R} \times_1 \mathbf{U}^{(1)} \times_2 \mathbf{U}^{(2)} \cdots \times_N \mathbf{U}^{(N)}, \tag{3.2}$$

where each matrix $\mathbf{U}^{(j)} \in \mathbb{R}^{I_j \times \Delta_j}$ $(j = 1, 2, \ldots, N)$ is orthogonal, i.e. $\mathbf{U}^{(j)T}\mathbf{U}^{(j)} = \mathbf{I}$ ($\mathbf{I} \in \mathbb{R}^{\Delta_j \times \Delta_j}$ denotes the identity matrix). It is called a *common factor* matrix and can be thought of as the principal components in each mode j. The parameters Δ_j satisfying

$$\Delta_j \leq \text{rank}(\mathbf{X}_{(j)}) \tag{3.3}$$

are referred to as the compression ranks of the TD.

The $(N+1)$th-order core tensor $\mathcal{R}$ and common factor matrices $\mathbf{U}^{(j)} \in \mathbb{R}^{I_j \times \Delta_j}$

are supposed to be found from the following nonlinear least squares

$$\min_{\mathcal{R}\in\mathbb{R}^{\Delta_1\times\cdots\times\Delta_N\times K},\mathbf{U}^{(j)}\in\mathbb{R}^{I_j\times\Delta_j},j=1,...,N} ||\mathcal{X}-\mathcal{R}\times_1\mathbf{U}^{(1)}\times_2\mathbf{U}^{(2)}\cdots\times_N\mathbf{U}^{(N)}||_F^2$$
$$\text{subject to} \quad (\mathbf{U}^{(j)})^T\mathbf{U}^{(j)}=\mathbf{I}, j=1,...,N. \tag{3.4}$$

The optimisation problem (3.4) is computationally intractable, which could be addressed only by alternating least squares (ALS) in each $\mathbf{U}^{(j)}$ (with other $\mathbf{U}^{(\ell)}$, $\ell\neq j$ held fixed) [103]:

$$\min_{\mathcal{R}^{(j)}\in\mathbb{R}^{\Delta_1\times\cdots\times\Delta_N\times K},\mathbf{U}^{(j)}\in\mathbb{R}^{I_j\times\Delta_j}} ||\mathcal{X}-\mathcal{R}^{(j)}\times_1\mathbf{U}^{(1)}\times_2\mathbf{U}^{(2)}\cdots\times_N\mathbf{U}^{(N)}||_F^2$$
$$\text{subject to} \quad (\mathbf{U}^{(j)})^T\mathbf{U}^{(j)}=\mathbf{I}. \tag{3.5}$$

The computation complexity per one iteration consisting of N ALS (3.5) is [141, p. 127]

$$\mathcal{O}(K\Delta I^N+NKI\Delta^{2(N-1)}+NK\Delta^{3(N-1)}) \tag{3.6}$$

for

$$I_j\equiv I \quad \text{and} \quad \Delta_j\equiv\Delta, j=1,2,...,N. \tag{3.7}$$

The optimal $(N+1)$th-order core tensor $\mathcal{R}\in\mathbb{R}^{\Delta_1\times\cdots\times\Delta_N\times K}$ in (3.4) is seen as the concatenation of compressed $\widetilde{\mathcal{X}}^{(k)}\in\mathbb{R}^{\Delta_1\times\cdots\times\Delta_N}$ of the sample tensors $\mathcal{X}^{(k)}\in\mathbb{R}^{I_1\times\cdots\times I_N}$, $k=1,\cdots,K$:

$$\mathcal{R}=[\widetilde{\mathcal{X}}^{(1)}\widetilde{\mathcal{X}}^{(2)}\cdots\widetilde{\mathcal{X}}^{(N)}]=\mathcal{X}\times_1(\mathbf{U}^{(1)})^T\cdots\times_N(\mathbf{U}^{(N)})^T. \tag{3.8}$$

Accordingly, the test tensors $\mathcal{Y}^{(\ell)}$ are compressed to

$$\widetilde{\mathcal{Y}}^{(\ell)}=\mathcal{Y}^{(\ell)}\times_1(\mathbf{U}^{(1)})^T\cdots\times_N(\mathbf{U}^{(N)})^T\in\mathbb{R}^{\Delta_1\times\cdots\times\Delta_N}. \tag{3.9}$$

The number

$$N_f=\prod_{j=1}^{N}\Delta_j \tag{3.10}$$

thus represents the dimension of the feature space $\mathbb{R}^{\Delta_1\times\cdots\times\Delta_N}$.

Putting aside the computational intractability of the optimal factor matrices $\mathbf{U}^{(j)}$ in (3.4), the TD-based tensor compression by (3.8) and (3.9) is a systematic procedure only when the right hand side of (3.2) provides a good approximation

of $\mathcal{X}$, which is impossible for small Δ_j satisfying (3.3) [2]. In other words, the compression of large dimensional tensors to small dimensional tensors results in substantial lossy compression under the TD framework. Furthermore, one can see the value of (3.5) is lower bounded by

$$\sum_{i=1}^{r_j-\Delta_j-1} s_i, \tag{3.11}$$

where $r_j := \text{rank}(\mathcal{X}_{(j)})$ and $\{s_{r_j}, \cdots, s_1\}$ is the set of non-zero eigenvalues of the positive definite matrix $\mathcal{X}_{(j)}(\mathcal{X}_{(j)})^T$ in decreasing order. Since the matrix $\mathcal{X}_{(j)} \in \mathbb{R}^{I_j \times (K \prod_{\ell \neq j} I_\ell)}$ is highly unbalanced as a result of tensor matricization along one mode versus the rest, it is almost full-row (low) rank ($r_j \approx I_j$) and its squared $\mathcal{X}_{(j)}(\mathcal{X}_{(j)})^T$ of size $I_j \times I_j$ is well-conditioned in the sense that its eigenvalues do not decay quickly. As a consequence, (3.11) cannot be small for small Δ_j so the ALS (3.5) cannot result in a good approximation. The information loss with the least square (3.5) is thus more than

$$-\sum_{i=1}^{r_j-\Delta_j-1} \frac{s_i}{\sum_{i=1}^{r_j} s_i} \log_2 \frac{s_i}{\sum_{i=1}^{r_j} s_i}, \tag{3.12}$$

which is really essential in the von Neumann entropy of $\mathcal{X}_{(j)}$:

$$-\sum_{i=1}^{r_j} \frac{s_i}{\sum_{i=1}^{r_j} s_i} \log_2 \frac{s_i}{\sum_{i=1}^{r_j} s_i}. \tag{3.13}$$

Note that each entropy (3.13) is the mean for only local correlation between mode j and the rest [142]. The MPCA [17] aims at (3.4) with

$$\mathcal{X} = [(\mathcal{X}^{(1)} - \bar{\mathcal{X}}) \cdots (\mathcal{X}^{(K)} - \bar{\mathcal{X}})]$$

with $\bar{\mathcal{X}} = \frac{1}{K+L}(\sum_{k=1}^{K} \mathcal{X}^{(k)} + \sum_{\ell=1}^{L} \mathcal{Y}^{(\ell)})$. With such definition of $\mathcal{X}$, $(N+1)$th-order core tensor $\mathcal{X}$ is the concatenation of principal components of $\mathcal{X}^{(k)}$, while principal components of $\mathcal{Y}^{(\ell)}$ is defined by $(\mathcal{Y}^{(\ell)} - \bar{\mathcal{X}}) \times_1 (\mathbf{U}^{(1)})^T \cdots \times_N (\mathbf{U}^{(N)})^T$. Thus, MPCA suffers the similar conceptual drawbacks inherent by TD. Particularly, restricting $N_f = \prod_{j=1}^{N} \Delta_j$ to a moderate size leads to ignoring many important principle components.

We now present a novel approach to extract tensor features, which is based on MPS. Firstly, *permute* mode K of the tensor $\mathcal{X}$ such that

$$\mathcal{X} \in \mathbb{R}^{I_1 \times \cdots I_{n-1} \times K \times I_n \cdots \times I_N}. \tag{3.14}$$

The elements of $\mathcal{X}$ can be presented in the following *mixed-canonical form* [136] of the matrix product state (MPS) or tensor train (TT) decomposition [11, 9, 72, 8]:

$$x_{i_1 \cdots k \cdots i_N} = x^{(k)}_{i_1 \cdots i_n \cdots i_N} \approx \mathbf{B}^{(1)}_{i_1} \cdots \mathbf{B}^{(n-1)}_{i_{n-1}} \mathbf{G}^{(n)}_{k} \mathbf{C}^{(n+1)}_{i_n} \cdots \mathbf{C}^{(N+1)}_{i_N}, \tag{3.15}$$

where matrices $\mathbf{B}^{(j)}_{i_j}$ and $\mathbf{C}^{(j)}_{i_{(j-1)}}$ (the upper index "(j)" denotes the position j of the matrix in the chain) of size $\Delta_{j-1} \times \Delta_j$ ($\Delta_0 = \Delta_{N+1} = 1$), are called "left" and "right" *common factors* which satisfy the following orthogonality conditions:

$$\sum_{i_j=1}^{I_j} (\mathbf{B}^{(j)}_{i_j})^T \mathbf{B}^{(j)}_{i_j} \;=\; \mathbf{I}, \quad (j = 1, \ldots, n-1) \tag{3.16}$$

and

$$\sum_{i_{j-1}=1}^{I_{j-1}} \mathbf{C}^{(j)}_{i_{j-1}} (\mathbf{C}^{(j)}_{i_{j-1}})^T \;=\; \mathbf{I}, \quad (j = n+1, \ldots, N+1) \tag{3.17}$$

respectively, where $\mathbf{I}$ denotes the identity matrix. Each matrix $\mathbf{G}^{(n)}_{k}$ of dimension $\Delta_{n-1} \times \Delta_n$ is the compression of the training tensor $\mathcal{X}^{(k)}$. The parameters Δ_j are called the bond dimensions or compression ranks of the MPS. Using the common factors $\mathbf{B}^{(j)}_{i_j}$ and $\mathbf{C}^{(j)}_{i_{(j-1)}}$, we can extract the core matrices for the test tensors $\mathcal{Y}^{(\ell)}$ as follows. We permute all $\mathcal{Y}^{(\ell)}$, $\ell = 1, \cdots, L$ in such a way that the index ℓ is at the same position as k in the training tensors to ensure the compatibility between the training and test tensors. The compressed matrix $\mathbf{Q}^{(n)}_{\ell} \in \mathbb{R}^{\Delta_{n-1} \times \Delta_n}$ of the test tensor $\mathcal{Y}^{(\ell)}$ is then given by

$$\mathbf{Q}^{(n)}_{\ell} \;=\; \sum_{i_1, \ldots, i_N} (\mathbf{B}^{(1)}_{i_1})^T \cdots (\mathbf{B}^{(n-1)}_{i_{n-1}})^T y^{(\ell)}_{i_1 \ldots\ldots i_N} (\mathbf{C}^{(n+1)}_{i_n})^T \cdots (\mathbf{C}^{(N+1)}_{i_N})^T. \tag{3.18}$$

The dimension

$$N_f = \Delta_{n-1} \Delta_n \tag{3.19}$$

is the number of reduced features.

3.2 Tailored MPS for tensor compression

The advantage of MPS for tensor compression is that the order N of a tensor does not affect directly the feature number N_f in (3.19), which is only determined strictly by the product of the aforementioned bond dimensions Δ_{n-1} and Δ_n. In order to keep Δ_{n-1} and Δ_n to a moderate size, it is important to control the bond dimensions Δ_j, and also to optimise the positions of tensor modes as we address in this section. In what follows, for a matrix $\mathbf{X}$ we denote $\mathbf{X}(i,:)$ ($\mathbf{X}(:,j)$, resp.) as its ith row (jth column, resp.), while for a third-order tensor $\mathcal{X}$ we denote $\mathcal{X}(:,\ell,:)$ as a matrix such that its (i_1,i_3)th entry is $\mathcal{X}(i_1,\ell,i_3)$. For a Nth-order tensor $\mathcal{X} \in \mathbb{R}^{I_1\times\cdots\times I_N}$ we denote $\mathbf{X}_{[j]} \in \mathbb{R}^{(I_1I_2\cdots I_j)\times(I_{j+1}\cdots K\cdots I_N)}$ as its *mode-*$(1,2,\ldots,j)$ *matricization*. It is obvious that $\mathbf{X}_{[1]} = \mathbf{X}_{(1)}$.

3.2.1 Adaptive bond dimension control in MPS

To decompose the training tensor $\mathcal{X}$ into the MPS according to (3.15), we apply two successive sequences of SVDs to the tensor which include left-to-right sweep for computing the left common factors $\mathbf{B}_{i_1}^{(1)},\ldots,\mathbf{B}_{i_{n-1}}^{(n-1)}$, and right-to-left sweep for computing the right common factors $\mathbf{C}_{i_n}^{(n+1)},\ldots,\mathbf{C}_{i_N}^{(N+1)}$ and the core matrix $\mathbf{G}_k^{(n)}$ in (3.15) as follows:

- *Left-to-right sweep for left factor computation:*

The left-to-right sweep involves acquiring matrices $\mathbf{B}_{i_j}^{(j)}$ ($i_j = 1,\ldots,I_j$; $j = 1,\ldots,n-1$) fulfilling orthogonality condition in (3.16). Start by performing the mode-1 matricization of $\mathcal{X}$ to obtain

$$\mathbf{W}^{(1)} := \mathcal{X}_{[1]} = \mathcal{X}_{(1)} \in \mathbb{R}^{I_1\times(I_2\cdots K\cdots I_N)}.$$

For

$$\Delta_1 \leq \operatorname{rank}(\mathbf{X}_{[1]}), \tag{3.20}$$

apply SVD to $\mathbf{W}^{(1)}$ to have the QR-approximation

$$\mathbf{W}^{(1)} \approx \mathbf{U}^{(1)}\mathbf{V}^{(1)} \in \mathbb{R}^{I_1\times(I_2\cdots K\cdots I_N)}, \mathbf{U}^{(1)} \in \mathbb{R}^{I_1\times\Delta_1}, \mathbf{V}^{(1)} \in \mathbb{R}^{\Delta_1\times(I_2\cdots K\cdots I_N)}, \tag{3.21}$$

with $\mathbf{U}^{(1)}$ orthogonal:

$$(\mathbf{U}^{(1)})^T\mathbf{U}^{(1)} = \mathbf{I}. \tag{3.22}$$

Define the the most left common factors by

$$\mathbf{B}_{i_1}^{(1)} = \mathbf{U}^{(1)}(i_1, :) \in \mathbb{R}^{1\times\Delta_1}, i_1 = 1, \cdots, I_1 \tag{3.23}$$

which satisfy the left-canonical constraint in (3.16) due to (3.22).
Next, reshape the matrix $\mathbf{V}^{(1)} \in \mathbb{R}^{\Delta_1\times(I_2\cdots K\cdots I_N)}$ to $\mathbf{W}^{(2)} \in \mathbb{R}^{(\Delta_1 I_2)\times(I_3\cdots K\cdots I_N)}$. For

$$\Delta_2 \leq \text{rank}(\mathbf{W}^{(2)}) \leq \text{rank}(\mathbf{X}_{[2]}), \tag{3.24}$$

apply SVD to $\mathbf{W}^{(2)}$ for the QR-approximation

$$\mathbf{W}^{(2)} \approx \mathbf{U}^{(2)}\mathbf{V}^{(2)} \in \mathbb{R}^{(\Delta_1 I_2)\times(I_3\cdots K\cdots I_N)}, \mathbf{U}^{(2)} \in \mathbb{R}^{(\Delta_1 I_2)\times\Delta_2}, \mathbf{V}^{(2)} \in \mathbb{R}^{\Delta_2\times(I_3\cdots K\cdots I_N)} \tag{3.25}$$

with $\mathbf{U}^{(2)}$ orthogonal

$$(\mathbf{U}^{(2)})^T\mathbf{U}^{(2)} = \mathbf{I}. \tag{3.26}$$

Reshape the matrix $\mathbf{U}^{(2)} \in \mathbb{R}^{(\Delta_1 I_2)\times\Delta_2}$ into a third-order tensor $\mathcal{U} \in \mathbb{R}^{\Delta_1\times I_2\times\Delta_2}$ to define the next common factors

$$\mathbf{B}_{i_2}^{(2)} = \mathcal{U}(:, i_2, :) \in \mathbb{R}^{\Delta_1\times\Delta_2}, i_2 = 1, \cdots, I_2, \tag{3.27}$$

which satisfy the left-canonical constraint due to (3.26).
Applying the same procedure for determining $\mathbf{B}_{i_3}^{(3)}$ by reshaping the matrix $\mathbf{V}^{(2)} \in \mathbb{R}^{\Delta_2\times(I_3\cdots K\cdots I_N)}$ to

$$\mathbf{W}^{(3)} \in \mathbb{R}^{(\Delta_2 I_3)\times(I_4\cdots K\cdots I_N)},$$

performing the SVD, and so on. This procedure is iterated till obtaining the last QR-approximation

$$\begin{aligned} \mathbf{W}^{(n-1)} \;\approx\; & \mathbf{U}^{(n-1)}\mathbf{V}^{(n-1)} \in \mathbb{R}^{(\Delta_{n-2}I_{n-1})\times(KI_n\cdots I_N)}, \\ & \mathbf{U}^{(n-1)} \in \mathbb{R}^{(\Delta_{n-2}I_{n-1})\times\Delta_{n-1}}, \mathbf{V}^{(n-1)} \in \mathbb{R}^{\Delta_{n-1}\times(KI_n\cdots I_N)}, \end{aligned} \tag{3.28}$$

with $\mathbf{U}^{(n-1)}$ orthogonal:

$$\mathbf{U}^{(n-1)}(\mathbf{U}^{(n-1)})^T = \mathbf{I} \tag{3.29}$$

and reshaping $\mathbf{U}^{(n-1)} \in \mathbb{R}^{(\Delta_{n-2}I_{n-1})\times\Delta_{n-1}}$ into a third-order tensor $\mathcal{U} \in \mathbb{R}^{\Delta_{n-2}\times I_{n-1}\times\Delta_{n-1}}$ to define the last left common factors

$$\mathbf{B}_{i_{n-1}}^{(n-1)} = \mathcal{U}(:, i_{n-1}, :) \in \mathbb{R}^{\Delta_{n-2}\times\Delta_{n-1}}, i_{n-1} = 1, \cdots, I_{n-1}, \tag{3.30}$$

which satisfy the left-canonical constraint due to (3.29).

In a nutshell, after completing the left-to-right sweep, the elements of tensor $\mathcal{X}$ are approximated by

$$x_{i_1\cdots i_{n-1}i_n\cdots i_{N+1}}^{(k)} \approx \mathbf{B}_{i_1}^{(1)}\cdots\mathbf{B}_{i_{n-1}}^{(n-1)}\mathbf{V}^{(n-1)}(:, ki_n\cdots i_N). \tag{3.31}$$

The matrix $\mathbf{V}^{(n-1)} \in \mathbb{R}^{\Delta_{n-1}\times(KI_n\cdots I_N)}$ is reshaped to $\mathbf{W}^{(N)} \in \mathbb{R}^{(\Delta_{n-1}K\cdots I_{N-1})\times I_N}$ for the next right-to-left sweeping process.

- *Right-to-left sweep for right factor computation:*

Similar to left-to-right sweep, we perform a sequence of SVDs starting from the right to the left of the MPS to get the matrices $\mathbf{C}_{i_{j-1}}^{(j)}$ $(i_{j-1} = 1, \ldots, I_{j-1};\ j = N+1, \ldots, n+1)$ fulfilling the right-canonical condition in (3.17). To start, we apply the SVD to the matrix $\mathbf{W}^{(N)} \in \mathbb{R}^{(\Delta_{n-1}K\cdots I_{N-1})\times I_N}$ obtained previously in the left-to-right sweep to have the RQ-approximation

$$\mathbf{W}^{(N)} \approx \mathbf{U}^{(N)}\mathbf{V}^{(N)}, \mathbf{U}^{(N)} \in \mathbb{R}^{(\Delta_{n-1}K\cdots I_{N-1})\times\Delta_N}, \mathbf{V}^{(N)} \in \mathbb{R}^{\Delta_N\times I_N}, \tag{3.32}$$

with $\mathbf{V}^{(N)}$ orthogonal:

$$\mathbf{V}^{(N)}(\mathbf{V}^{(N)})^T = \mathbf{I} \tag{3.33}$$

for

$$\Delta_N \leq \text{rank}(\mathbf{W}^{(N)}) \leq \text{rank}(\mathcal{X}_{[N-1]}). \tag{3.34}$$

Define the most right common factors

$$\mathbf{C}_{i_N}^{(N+1)} = \mathbf{V}^{(N)}(:, i_N) \in \mathbb{R}^{\Delta_N\times 1}, i_N = 1, \cdots, I_N,$$

which satisfy the right-canonical constraint (3.17) due to (3.33).
Next, reshape $\mathbf{U}^{(N)} \in \mathbb{R}^{(\Delta_{n-1}K\cdots I_{N-1})\times\Delta_N}$ into $\mathbf{W}^{(N-1)} \in \mathbb{R}^{(\Delta_{n-1}K\cdots I_{N-2})\times(I_{N-1}\Delta_N)}$ and apply the SVD to have the RQ-approximation

$$\mathbf{W}^{(N-1)} \approx \mathbf{U}^{(N-1)}\mathbf{V}^{(N-1)}, \mathbf{U}^{(N-1)} \in \mathbb{R}^{(\Delta_{n-1}K\cdots I_{N-2})\times\Delta_{N-1}}, \mathbf{V}^{(N-1)} \in \mathbb{R}^{\Delta_{N-1}\times(I_{N-1}\Delta_N)} \tag{3.35}$$

with $\mathbf{V}^{(N-1)}$ orthogonal:

$$\mathbf{V}^{(N-1)}(\mathbf{V}^{(N-1)})^T = \mathbf{I} \tag{3.36}$$

for

$$\Delta_{N-1} \leq \text{rank}(\mathbf{W}^{(N-1)}) \leq \text{rank}(\mathcal{X}_{[N-2]}). \tag{3.37}$$

Reshape the matrix $\mathbf{V}^{(N-1)} \in \mathbb{R}^{\Delta_{N-1} \times (I_{N-1}\Delta_N)}$ into a third-order tensor $\mathcal{V} \in \mathbb{R}^{\Delta_{N-1} \times I_{N-1} \times \Delta_N}$ to define the next common factor

$$\mathbf{C}_{i_{N-1}}^{(N)} = \mathcal{V}(:, i_{N-1}, :) \in \mathbb{R}^{\Delta_{N-1} \times \Delta_N} \tag{3.38}$$

which satisfy (3.17) due to (3.36).
This procedure is iterated till obtaining the last RQ-approximation

$$\begin{aligned} \mathbf{W}^{(n)} \quad &\approx \quad \mathbf{U}^{(n)}\mathbf{V}^{(n)} \in \mathbb{R}^{(\Delta_{n-1}K) \times (I_n \Delta_{n+1})}, \\ &\quad\quad \mathbf{U}^{(n)} \in \mathbb{R}^{(\Delta_{n-1}K) \times \Delta_n}, \mathbf{V}^{(n)} \in \mathbb{R}^{\Delta_n \times (I_n \Delta_{n+1})}, \end{aligned} \tag{3.39}$$

with $\mathbf{V}^{(n)}$ orthogonal:

$$\mathbf{V}^{(n)}(\mathbf{V}^{(n)})^T = \mathbf{I} \tag{3.40}$$

for

$$\Delta_n \leq \text{rank}(\mathbf{W}^{(n)}) \leq \text{rank}(\mathcal{X}_{[n-1]}). \tag{3.41}$$

Reshape $\mathbf{V}^{(n)} \in \mathbb{R}^{(\Delta_n) \times (I_n \Delta_{n+1})}$ into a third-order tensor $\mathcal{V} \in \mathbb{R}^{\Delta_n \times I_n \times \Delta_{n+1}}$ to define the last right common factors

$$\mathbf{C}_{i_n}^{(n+1)} = \mathcal{V}(:, i_n, :) \in \mathbb{R}^{\Delta_{n-1} \times \Delta_n}, i_n = 1, \cdots, I_n, \tag{3.42}$$

which satisfy (3.17) due to (3.40).

By reshaping $\mathbf{U}^{(n)} \in \mathbb{R}^{(\Delta_{n-1}K) \times \Delta_n}$ into a third-order tensor $\mathcal{G} \in \mathbb{R}^{\Delta_{n-1} \times K \times \Delta_n}$ to define $\mathbf{G}_k^{(n)} = \mathcal{G}(:, k, :)$, $k = 1, \cdots .K$, we arrive at (3.15).

Note that the MPS decomposition described by (3.15) can be performed exactly or approximately depending on the bond dimensions Δ_j ($j = 1, \ldots, N$). The bond dimension truncation is of crucial importance to control the final feature number $N_f = \Delta_{n-1}\Delta_n$. To this end, we rely on thresholding the singular values

of $\mathbf{W}^{(j)}$. With a threshold ϵ being defined in advance, we control Δ_j such that Δ_j largest singular values $s_1 \geq s_2 \geq ... \geq s_{\Delta_j}$ satisfy

$$\frac{\sum_{i=1}^{\Delta_j} s_i}{\sum_{i=1}^{r_j} s_j} \geq \epsilon, \tag{3.43}$$

for $r_j = \text{rank}(\mathbf{W}^{(j)})$. The information loss from the von Neumann entropy (3.13) of $\mathbf{W}^{(j)}$ by this truncation is given by (3.12). The entropy of each $\mathbf{W}^{(j)}$ provides the correlation degree between two sets of modes $1, \cdots, j$ and $j+1, \cdots, N$ [142]. Therefore, the N entropies $\mathbf{W}^{(j)}$, $j = 1, \cdots, N$ provide the mean of the tensor's global correlation. Furthermore, rank r_j of each $\mathbf{W}^{(j)}$ is upper bounded by

$$\min \{I_1 \cdots I_j, I_{j+1} \cdots I_N\} \tag{3.44}$$

making the truncation (3.43) highly favorable in term of compression loss to matrices of higher rank due to balanced row and column numbers.
A detailed outline of our MPS approach to tensor feature extraction is presented in Algorithm 1.

3.2.2 Tensor mode pre-permutation and pre-positioning mode K for MPS

One can see from (3.44) that the efficiency of controlling the bond dimension Δ_j is dependent on its upper bound (3.44). Particularly, the efficiency of controlling the bond dimensions Δ_{n-1} and Δ_n that define the feature number (3.19) is dependent on

$$\min \{I_1 \cdots I_{n-1}, I_n \cdots I_N\} \tag{3.45}$$

Therefore, it is important to pre-permute the tensors modes such that the ratio

$$\frac{\prod_{i=1}^{n-1} I_i}{\prod_{i=n}^{N} I_i} \tag{3.46}$$

is near to 1 as possible, while $\{I_1, \cdots, I_{n-1}\}$ is in decreasing order

$$I_1 \geq \cdots \geq I_{n-1} \tag{3.47}$$

and $\{I_n, \cdots, I_N\}$ in increasing order

$$I_n \leq \cdots \leq I_N \tag{3.48}$$

Algorithm I: MPS for tensor feature extraction

Input: $\mathcal{X} \in \mathbb{R}^{I_1 \times \cdots \times I_{n-1} \times K \cdots \times I_N}$,
ϵ: SVD threshold

Output: $\mathbf{G}_k^{(n)} \in \mathbb{R}^{\Delta_{n-1} \times \Delta_n}$, $k = 1, \cdots, K$
$\mathbf{B}_{i_j}^{(j)}$ $(i_j = 1, \ldots, I_j, j = 1, \ldots, n-1)$
$\mathbf{C}_{i_{(j-1)}}^{(j)}$ $(i_{(j-1)} = 1, \ldots, I_{(j-1)}, j = n+1, \ldots, N+1)$

1: Set $\mathbf{W}^{(1)} = \mathbf{X}_{(1)}$ % Mode-1 matricization of $\mathcal{X}$
2: **for** $j = 1$ **to** $n-1$ % Left-to-right sweep
3: $\mathbf{W}^{(j)} = \mathbf{USV}$ % SVD of $\mathbf{W}^{(j)}$
4: $\mathbf{W}^{j} \approx \mathbf{U}^{(j)}\mathbf{W}^{(j+1)}$ % Thresholding $\mathbf{S}$ for QR-approximation
5: Reshape $\mathbf{U}^{(j)}$ to $\mathcal{U}$
6: $\mathbf{B}_{i_j}^{(j)} = \mathcal{U}(:, i_j, :)$ % Set common factors
7: **end**
8: Reshape $\mathbf{V}^{(n-1)}$ to $\mathbf{W}^{N} \in \mathbb{R}^{(\Delta_{n-1} K \cdots I_N) \times I_N}$
9: **for** $j = N$ **down to** n % right-to-left sweep
10: $\mathbf{W}^{(j)} = \mathbf{USV}$ % SVD of $\mathbf{W}^{(j)}$
11: $\mathbf{W}^{(j)} \approx \mathbf{W}^{(j-1)}\mathbf{V}^{(j)}$ % Thresholding $\mathbf{S}$ for RQ-approximation
13: Reshape $\mathbf{V}^{(j)}$ to $\mathcal{V}$
14: $\mathbf{C}_{i_{j-1}}^{(j+1)} = \mathcal{V}(:, i_{j-1}, :)$ % Set common factors
15: **end**
16: Reshape $\mathbf{U}^{(n)}$ into $\mathcal{G} \in \mathbb{R}^{\Delta_{n-1} \times K \times \Delta_n}$
17: Set $\mathbf{G}_k^{(n)} = \mathcal{G}(:, k, :)$ % Training core matrix

Texts after symbol "%" are comments.

to improve the ratios

$$\frac{\prod_{i=1}^{j} I_j}{\prod_{i=j+1}^{N} I_i} \tag{3.49}$$

for balancing $\mathbf{W}^{(j)}$.

The mode K is then pre-positioned in n-th mode as in (3.14).

3.2.3 Complexity analysis

In the following complexity analysis it is assumed $I_n = I \ \forall n$ for simplicity. The dominant computational complexity of MPS is $\mathcal{O}(KI^{(N+1)})$ due to the first SVD of the matrix obtained from the mode-1 matricization of $\mathcal{X}$. On the other hand,

the computational complexity of HOOI requires several iterations of an ALS method to obtain convergence. In addition, it usually employs the HOSVD to initialize the tensors which involves the cost of order $\mathcal{O}(NKI^{N+1})$, and thus very expensive with large N compared to MPS.

MPCA is computationally upper bounded by $\mathcal{O}(NKI^{N+1})$, however, unlike HOOI, MPCA doesn't require the formation of the $(N+1)$th order core tensor at every iteration and convergence can usually happen in one iteration [17].[1]

The computational complexity of R-UMLDA is approximately $\mathcal{O}(K\sum_{n=2}^{N} I^n + (C+K)I^2 + (p-1)[IK + 2I^2 + (p-1)^2 + (2I(p-1)] + 4I^3)$, where C is the number of classes, p is the number of projections, which determines the core vector size [105]. Therefore, R-UMLDA would perform poorly for many samples and classes.

3.2.4 MPS-based tensor object classification

This subsection presents two methods for tensor objection classification based on Algorithm 1. For each method, an explanation of how to reduce the dimensionality of tensors to core matrices, and subsequently to feature vectors for application to linear classifiers is given.

3.2.4.1 Principal component analysis via tensor-train (TTPCA)

The TTPCA algorithm is an approach where Algorithm 1 is applied directly on the training set, with no preprocessing such as data centering. Specifically, given a set of Nth-order tensor samples $\mathcal{X}^{(k)} \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_N}$, then the core matrices are obtained as

$$\mathbf{G}_k^{(n)} \in \mathbb{R}^{\Delta_{n-1} \times \Delta_n}. \tag{3.50}$$

Vectorizing each k sample results in

$$\mathbf{g}_k^{(n)} \in \mathbb{R}^{\Delta_{n-1}\Delta_n}. \tag{3.51}$$

[1] This does not mean that MPCA is computationally efficient but in contrast this means that alternating iterations of MPCA prematurely terminate, yielding a solution that is far from the optimal one.

Using (3.43), $\Delta_{n-1}\Delta_n$ features of k is significantly less in comparison to $N_f = \prod_{n=1}^{N} I_n$ of $\mathcal{X}^{(k)}$, which allows for PCA to be easily applied.

3.2.4.2 MPS

The second algorithm is simply called MPS, where in this case we first perform data centering on the set of training samples $\{\mathcal{X}^{(k)}\}$, then apply Algorithm 1 to obtain the core matrices

$$\mathbf{G}_k^{(n)} \in \mathbb{R}^{\Delta_{n-1} \times \Delta_n}. \tag{3.52}$$

Vectorizing the K samples results in (3.51), and subsequent linear classifiers such as LDA or nearest neighbors can be utilised. In this method, MPS can be considered a multidimensional analogue to PCA because the tensor samples have been data centered and are projected to a new orthogonal space through Algorithm 1, resulting in the core matrices.

3.3 Experimental results

In this section, we conduct experiments on the proposed TTPCA and MPS algorithms for tensor object classification. An extensive comparison is conducted based on CSR and training time with tensor-based methods MPCA, HOOI, and R-UMLDA.

Four datasets are utilised for the experiment. The Columbia Object Image Libraries (COIL-100) [143, 144], Extended Yale Face Database B (EYFB) [145], BCI Jiaotong dataset (BCI) [146], and the University of South Florida HumanID "gait challenge" dataset (GAIT) version 1.7 [147] . All simulations are conducted in a Matlab environment.

3.3.1 Parameter selection

TTPCA, MPA and HOOI rely on the threshold ϵ defined in (3.43) to reduce the dimensionality of a tensor, while keeping its most relevant features. To demon-

strate how the classification success rate (CSR) varies, we utilise different ϵ for each dataset. It is trivial to see that a larger ϵ would result in a longer training time due to its computational complexity, which was discussed in subsection 3.2.3. Furthermore, TTPCA utilises PCA, and a range of principal components p is used for the experiments. HOOI is implemented with a maximum of 10 ALS iterations. MPCA relies on fixing an initial quality factor Q, which is determined through numerical simulations, and a specified number of elementary multilinear projections (EMP), we denote as m_p, must be initialized prior to using the R-UMLDA algorithm. A range of EMP's is determined through numerical simulations and the regularization parameter is fixed to $\gamma = 10^{-6}$.

3.3.2 Tensor object classification

3.3.2.1 COIL-100

For this dataset we strictly compare MPS and the HOSVD-based algorithm HOOI to analyse how adjusting ϵ affects the approximation of the original tensors, as well as the reliability of the extracted features for classification. The COIL-100 dataset has 7200 color images of 100 objects (72 images per object) with different reflectance and complex geometric characteristics. Each image is initially a 3rd-order tensor of dimension $128 \times 128 \times 3$ and then is downsampled to the one of dimension $32 \times 32 \times 3$. The dataset is divided into training and test sets randomly consisting of K and L images, respectively according to a certain holdout (H/O) ratio r, i.e. $r = \frac{L}{K}$. Hence, the training and test sets are represented by four-order tensors of dimensions $32 \times 32 \times 3 \times K$ and $32 \times 32 \times 3 \times L$, respectively. In Fig. 3.1 we show how a few objects of the training set ($r = 0.5$ is chosen) change after compression by MPS and HOOI with two different values of threshold, $\epsilon = 0.9, 0.65$. We can see that with $\epsilon = 0.9$, the images are not modified significantly due to the fact that many features are preserved. However, in the case that $\epsilon = 0.65$, the images are blurred. That is because fewer features are kept. However, we can observe that the shapes of objects are still preserved. Especially, in most cases MPS seems to preserve the color of the images better than HOOI. This is because

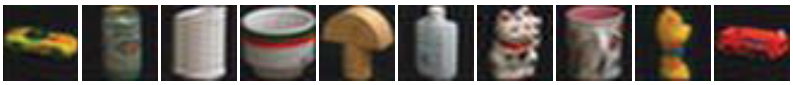
the bond dimension corresponding to the color mode $I_3 = 3$ has a small value, e.g. $\Delta_3 = 1$ for $\epsilon = 0.65$ in HOOI. This problem arises due to the the unbalanced matricization of the tensor corresponding to the color mode. Specifically, if we take a mode-3 matricization of tensor $\mathcal{X} \in \mathbb{R}^{32\times32\times3\times K}$, the resulting matrix of size $3 \times (1024K)$ is extremely unbalanced. Therefore, when taking SVD with some small threshold ϵ, the information corresponding to this color mode may be lost due to dimension reduction. On the contrary, we can efficiently avoid this problem in MPS by permuting the tensor such that $\mathcal{X} \in \mathbb{R}^{32\times K\times3\times32}$ before applying the tensor decomposition.

Table 3.1 – COIL-100 classification results. The best CSR corresponding to different H/O ratios obtained by MPS and HOOI.

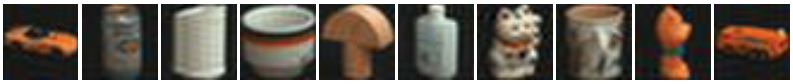
Algorithm	CSR	N_f	ϵ	CSR	N_f	ϵ
$r = 50\%$				$r = 80\%$		
HOOI	98.87 ± 0.19	198	0.80	94.13 ± 0.42	112	0.75
MPS	$\mathbf{99.19 \pm 0.19}$	120	0.80	$\mathbf{95.37 \pm 0.31}$	18	0.65
$r = 90\%$				$r = 95\%$		
HOOI	87.22 ± 0.56	112	0.75	77.76 ± 0.90	112	0.75
MPS	$\mathbf{89.38 \pm 0.40}$	59 ± 5	0.75	$\mathbf{83.17 \pm 1.07}$	18	0.65

(a) Original samples (size $32 \times 32 \times 3$)

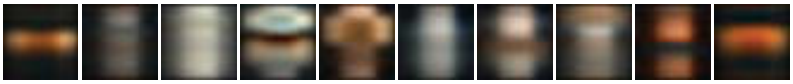
(b) Samples with MPS, $\epsilon = 0.9$ (core size 18×24)



(c) Samples with HOOI, $\epsilon = 0.9$ (core size $18 \times 16 \times 2$)



(d) Samples with MPS, $\epsilon = 0.65$ (core size 6×3)



(e) Samples with HOOI, $\epsilon = 0.65$ (core size $5 \times 4 \times 1$)

Figure 3.1 – Modification of ten objects in the training set of COIL-100 are shown after applying MPS and HOOI corresponding to $\epsilon = 0.9$ and 0.65 to compress tensor objects.

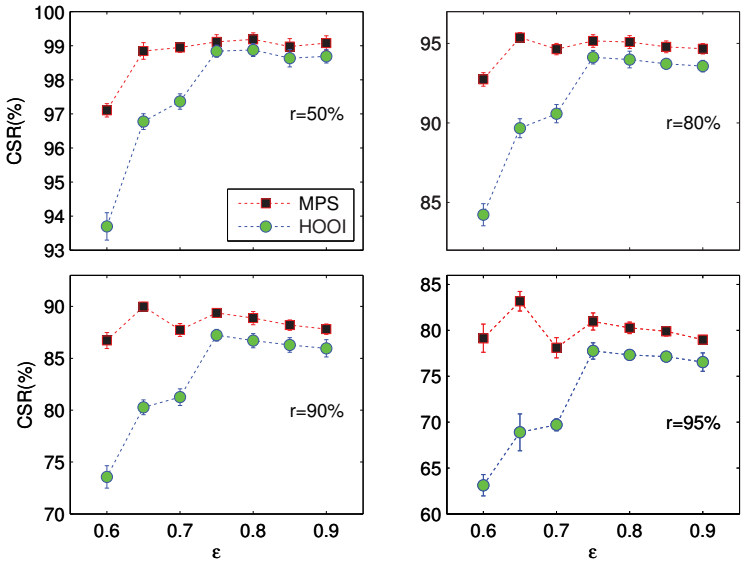


Figure 3.2 – Error bar plots of CSR versus thresholding rate ϵ for different H/O ratios.

K nearest neighbors with K=1 (KNN-1) is used for classification. For each H/O ratio, the CSR is averaged over 10 iterations of randomly splitting the dataset into training and test sets. Comparison of performance between MPS and HOOI is shown in Fig. 3.2 for four different H/O ratios, i.e. $r = (50\%, 80\%, 90\%, 95\%)$. In each plot, we show the CSR with respect to threshold ϵ. We can see that MPS performs quite well when compared to HOOI. Especially, with small ϵ, MPS performs much better than HOOI. Besides, we also show the best CSR corresponding to each H/O ratio obtained by different methods in Table. 3.1. It can be seen that MPS always gives better results than HOOI even in the case of small value of ϵ and number of features N_f defined by (3.10) and (3.19) for HOOI and MPS, respectively.

3.3.2.2 Extended Yale Face Database B

The EYFB dataset contains 16128 grayscale images with 28 human subjects, under 9 poses, where for each pose there is 64 illumination conditions. Similar to [148], to improve computational time each image was cropped to keep only the centre area containing the face, then resized to 73 x 55. The training and test datasets are not selected randomly but partitioned according to poses. More

precisely, the training and test datasets are selected to contain poses 0, 2, 4, 6 and 8 and 1, 3, 5, and 7, respectively. For a single subject the training tensor has size $5 \times 73 \times 55 \times 64$ and $4 \times 73 \times 55 \times 64$ is the size of the test tensor. Hence for all 28 subjects we have fourth-order tensors of sizes $140 \times 73 \times 55 \times 64$ and $112 \times 73 \times 55 \times 64$ for the training and test datasets, respectively.

Table 3.2 – EYFB classification results

Algorithm	CSR ($\epsilon = 0.9$)	CSR ($\epsilon = 0.85$)	CSR ($\epsilon = 0.80$)	CSR ($\epsilon = 0.75$)
KNN-1				
HOOI	90.71 ± 1.49	90.89 ± 1.60	91.61 ± 1.26	88.57 ± 0.80
MPS	$\mathbf{94.29 \pm 0.49}$	$\mathbf{94.29 \pm 0.49}$	$\mathbf{94.29 \pm 0.49}$	$\mathbf{94.29 \pm 0.49}$
TTPCA	86.05 ± 0.44	86.01 ± 0.86	87.33 ± 0.46	86.99 ± 0.53
MPCA	90.89 ± 1.32			
R-UMLDA	71.34 ± 2.86			
LDA				
HOOI	96.07 ± 0.80	95.89 ± 0.49	96.07 ± 0.49	96.07 ± 0.49
MPS	$\mathbf{97.32 \pm 0.89}$	$\mathbf{97.32 \pm 0.89}$	$\mathbf{97.32 \pm 0.89}$	$\mathbf{97.32 \pm 0.89}$
TTPCA	95.15 ± 0.45	95.15 ± 0.45	95.15 ± 0.45	94.86 ± 0.74
MPCA	90.00 ± 2.92			
R-UMLDA	73.38 ± 1.78			

In this experiment, the core tensors remains very large even with a small threshold used, e.g., for $\epsilon = 0.75$, the core size of each sample obtained by TTPCA/MPS and HOOI are $18 \times 201 = 3618$ and $14 \times 15 \times 13 = 2730$, respectively, because of slowly decaying singular values, which make them too large for classification. Therefore, we need to further reduce the sizes of core tensors before feeding them to classifiers for a better performance. In our experiment, we simply apply a further truncation to each core tensor by keeping the first few dimensions of each mode of the tensor. Intuitively, this can be done as we have already known that the space of each mode is orthogonal and ordered in such a way that the first dimension corresponds to the largest singular value, the second one corresponds to the second largest singular value and so on. Subsequently, we can independently truncate the dimension of each mode to a reasonably small value (which can be determined empirically) without changing significantly the meaning of the core tensors. It then gives rise to core tensors of smaller size that can be used directly for classification. More specifically, suppose that the core tensors obtained by

MPS and HOOI have sizes $Q \times \Delta_1 \times \Delta_2$ and $Q \times \Delta_1 \times \Delta_2 \times \Delta_3$, where Q is the number K (L) of training (test) samples, respectively. The core tensors are then truncated to be $Q \times \tilde{\Delta}_1 \times \tilde{\Delta}_2$ and $Q \times \tilde{\Delta}_1 \times \tilde{\Delta}_2 \times \tilde{\Delta}_3$, respectively such that $\tilde{\Delta}_l < \Delta_l$ $(l = 1, 2, 3)$. Note that each $\tilde{\Delta}_l$ is chosen to be the same for both training and test core tensors. In regards to TTPCA, each core matrix is vectorized to have $\Delta_1\Delta_2$ features, then PCA is applied.

Classification results for different threshold values ϵ is shown in Table. 3.2 for TTPCA, MPS and HOOI using two different classifiers, i.e. KNN-1 and LDA. Results from MPCA and R-UMLDA is also included. The core tensors obtained by MPS and HOOI are reduced to have sizes of $Q \times \tilde{\Delta}_1 \times \tilde{\Delta}_2$ and $Q \times \tilde{\Delta}_1 \times \tilde{\Delta}_2 \times \tilde{\Delta}_3$, respectively such that $\tilde{\Delta}_1 = \tilde{\Delta}_2 = \Delta \in (10, 11, 12, 13, 14)$ and $\tilde{\Delta}_3 = 1$. Therefore, the reduced core tensors obtained by both methods have the same size for classification. With MPS and HOOI, each value of CSR in Table. 3.2 is computed by taking the average of the ones obtained from classifying different reduced core tensors due to different Δ. In regards to TTPCA, for each ϵ, a range of principal components $p = \{50, \ldots, 70\}$ is used. We utilise $Q = \{70, 75, 80, 85, 90\}$ for MPCA, and the range $m_p = \{10, \ldots, 20\}$ for R-UMLDA. The average CSR's are computed with TTPCA, MCPA and R-UMLDA according to their respective range of parameters in Table. 3.2. We can see that the MPS gives rise to better results for all threshold values using different classifiers. More importantly, MPS with the smallest ϵ can produce the highest CSR. The LDA classifier gives rise to the best result, i.e. $\mathbf{97.32 \pm 0.89}$.

3.3.2.3 BCI Jiaotong

The BCIJ dataset consists of single trial recognition for BCI electroencephalogram (EEG) data involving left/right motor imagery (MI) movements. The dataset includes five subjects and the paradigm required subjects to control a cursor by imagining the movements of their right or left hand for 2 seconds with a 4 second break between trials. Subjects were required to sit and relax on a chair, looking at a computer monitor approximately 1m from the subject at eye level. For each

Table 3.3 – BCI Jiaotong classification results

Algorithm	CSR ($\epsilon = 0.9$)	CSR ($\epsilon = 0.85$)	CSR ($\epsilon = 0.80$)	CSR ($\epsilon = 0.75$)
Subject 1				
HOOI	84.39 ± 1.12	83.37 ± 0.99	82.04 ± 1.05	84.80 ± 2.21
MPS	$\mathbf{87.24 \pm 1.20}$	$\mathbf{87.55 \pm 1.48}$	$\mathbf{87.24 \pm 1.39}$	$\mathbf{87.65 \pm 1.58}$
TTPCA	78.57 ± 3.95	78.43 ± 3.73	79.43 ± 4.12	79.14 ± 2.78
MPCA	82.14 ± 3.50			
R-UMLDA	63.18 ± 0.37			
CSP	80.14 ± 3.73			
Subject 2				
HOOI	83.16 ± 1.74	82.35 ± 1.92	82.55 ± 1.93	79.39 ± 1.62
MPS	$\mathbf{90.10 \pm 1.12}$	$\mathbf{90.10 \pm 1.12}$	$\mathbf{90.00 \pm 1.09}$	$\mathbf{91.02 \pm 0.70}$
TTPCA	80.57 ± 0.93	81.14 ± 1.86	81.29 ± 1.78	80 ± 2.20
MPCA	81.29 ± 0.78			
R-UMLDA	70.06 ± 0.39			
CSP	81.71 ± 8.96			
Subject 3				
HOOI	60.92 ± 1.83	61.84 ± 1.97	61.12 ± 1.84	60.51 ± 1.47
MPS	61.12 ± 1.36	61.22 ± 1.53	61.12 ± 1.54	60.71 ± 1.54
TTPCA	67.43 ± 2.56	68.29 ± 2.56	67.71 ± 2.28	66.43 ± 2.02
MPCA	56.14 ± 2.40			
R-UMLDA	57.86 ± 0.00			
CSP	$\mathbf{77.14 \pm 2.26}$			
Subject 4				
HOOI	48.27 ± 1.54	47.55 ± 1.36	49.98 ± 1.29	47.96 ± 1.27
MPS	52.35 ± 2.82	52.55 ± 3.40	52.55 ± 3.69	51.84 ± 3.11
TTPCA	50.29 ± 2.97	49.71 ± 3.77	49.14 ± 3.48	52.00 ± 3.48
MPCA	51.00 ± 3.96			
R-UMLDA	46.36 ± 0.93			
CSP	$\mathbf{59.86 \pm 1.98}$			
Subject 5				
HOOI	$\mathbf{60.31 \pm 1.08}$	$\mathbf{60.82 \pm 0.96}$	$\mathbf{59.90 \pm 2.20}$	$\mathbf{60.41 \pm 1.36}$
MPS	59.39 ± 2.08	59.18 ± 2.20	58.57 ± 1.60	59.29 ± 1.17
TTPCA	53.43 ± 2.79	54.29 ± 3.19	53.86 ± 3.83	54.86 ± 2.49
MPCA	50.43 ± 1.48			
R-UMLDA	55.00 ± 0.55			
CSP	59.14 ± 2.11			

subject, data was collected over two sessions with a 15 minute break in between. The first session contained 60 trials (30 trials for left, 30 trials for right) and were used for training. The second session consisted of 140 trials (70 trials for left, 70 trials for right). The EEG signals were sampled at 500Hz and preprocessed with a filter at 8-30Hz, hence for each subject the data consisted of a multidimensional

tensor $channel \times time \times Q$. The common spatial patterns (CSP) algorithm [149] is a popular method for BCI classification that works directly on this tensor, and provides a baseline for the proposed and existing tensor-based methods. For the tensor-based methods, we preprocess the data by transforming the tensor into the time-frequency domain using complex Mortlet wavelets with bandwidth parameter $f_b = 6$Hz (CMOR6-1) to make classification easier [150, 151]. The wavelet center frequency $f_c = 1$Hz is chosen. Hence, the size of the concatenated tensors are 62 $channels \times 23$ $frequency\ bins \times 50$ $time\ frames \times Q$.

We perform the experiment for all subjects. After applying the feature extraction methods MPS and HOOI, the core tensors still have high dimension, so we need to further reduce their sizes before using them for classification. For instance, the reduced core sizes of MPS and HOOI are chosen to be $Q \times 12 \times \Delta$ and $Q \times 12 \times \Delta \times 1$, where $\Delta \in (8, \ldots, 14)$, respectively. With TTPCA, the principal components $p = \{10, 50, 100, 150, 200\}$, $Q = \{70, 75, 80, 85, 90\}$ for MPCA and $m_p = \{10, \ldots, 20\}$ for R-UMLDA. With CSP, we average CSR for a range of spatial components $s_c = \{2, 4, 6, 8, 10\}$.

The LDA classifier is utilised and the results are shown in Table. 3.3 for different threshold values of TTPCA, MPS and HOOI. The results of MPCA, R-UMLDA and CSP are also included. MPS outperforms the other methods for Subjects 1 and 2, and is comparable to HOOI in the results for Subject 5. CSP has the highest CSR for Subjects 3 and 4, followed by MPS or TTPCA, which demonstrates the proposed methods being effective at reducing tensors to relevant features, more precisely than current tensor-based methods.

Table 3.4 – Seven experiments in the USF GAIT dataset

Probe set	A(GAL)	B(GBR)	C(GBL)	D(CAR)	E(CBR)	F(CAL)	G(CBL)
Size	71	41	41	70	44	70	44
Differences	View	Shoe	Shoe, view	Surface	Surface, shoe	Surface, view	Surface, view, shoe

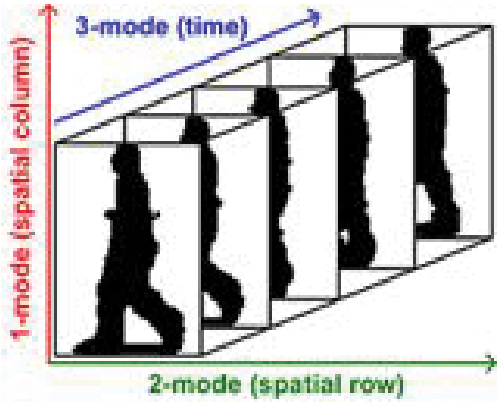


Figure 3.3 – The gait silhouette sequence for a third-order tensor.

3.3.2.4 USF GAIT challenge

The USFG database consists of 452 sequences from 74 subjects who walk in elliptical paths in front of a camera. There are three conditions for each subject: shoe type (two types), viewpoint (left or right), and the surface type (grass or concrete). A gallery set (training set) contains 71 subjects and there are seven types of experiments known as probe sets (test sets) that are designed for human identification. The capturing conditions for the probe sets is summarized in Table 3.4, where G, C, A, B, L and R stand for grass surface, cement surface, shoe type A, shoe type B, left view and right view, respectively. The conditions in which the gallery set was captured is grass surface, shoe type A and right view (GAR). The subjects in the probe and gallery sets are unique and there are no common sequences between the gallery and probe sets. Each sequence is of size 128×88 and the time mode is 20, hence each gait sample is a third-order tensor of size $128 \times 88 \times 20$, as shown in Fig. 3.3. The gallery set contains 731 samples, therefore the training tensor is of size $128 \times 88 \times 20 \times 731$. The test set is of size $128 \times 88 \times 20 \times P_s$, where P_s is the sample size for the probe set that is used for a benchmark, refer to Table 3.4. The difficulty of the classification task increases with the amount and and type of variables, e.g. Probe A only has the viewpoint, whereas Probe F has surface and viewpoint, which is more difficult. For the experiment we perform tensor object classification with Probes A, C, D and F (test sets).

The classification results based on using the LDA classifier is shown in Table 3.5. The threshold ϵ still retains many features in the core tensors of MPS and HOOI. Therefore, further reduction of the core tensors is chosen to be $Q \times 20 \times \Delta$ and $Q \times 20 \times \Delta \times 1$, where $\Delta \in (8, \ldots, 14)$, respectively. The principal components for TTPCA is the range $p = \{150, 200, 250, 300\}$, $Q = \{70, 75, 80, 85\}$ for MPCA and $m_p = \{10, \ldots, 20\}$ for R-UMLDA. The proposed algorithms achieve the highest performance for Probes A, C, and D. MPS and HOOI are similar for the most difficult test set Probe F.

Table 3.5 – GAIT classification results

Algorithm	CSR ($\epsilon = 0.9$)	CSR ($\epsilon = 0.85$)	CSR ($\epsilon = 0.80$)	CSR ($\epsilon = 0.75$)
Probe A				
HOOI	63.71 ± 3.36	63.90 ± 3.40	64.16 ± 3.39	64.33 ± 3.20
MPS	70.03 ± 0.42	70.03 ± 0.38	70.01 ± 0.36	69.99 ± 0.38
TTPCA	$\mathbf{75.31 \pm 0.29}$	$\mathbf{76.03 \pm 0.38}$	$\mathbf{76.38 \pm 0.78}$	$\mathbf{77.75 \pm 0.92}$
MPCA	55.77 ± 1.08			
R-UMLDA	46.62 ± 2.13			
Probe C				
HOOI	36.67 ± 2.84	36.73 ± 2.79	36.70 ± 3.07	36.87 ± 3.68
MPS	$\mathbf{41.46 \pm 0.64}$	$\mathbf{41.36 \pm 0.64}$	$\mathbf{41.29 \pm 0.63}$	41.46 ± 0.59
TTPCA	39.17 ± 0.90	40.83 ± 0.41	41.61 ± 1.02	$\mathbf{44.40 \pm 1.54}$
MPCA	29.35 ± 2.29			
R-UMLDA	20.87 ± 0.76			
Probe D				
HOOI	19.73 ± 0.91	19.96 ± 1.15	20.32 ± 0.93	20.29 ± 1.11
MPS	$\mathbf{23.82 \pm 0.42}$	$\mathbf{23.84 \pm 0.43}$	$\mathbf{23.84 \pm 0.45}$	$\mathbf{23.84 \pm 0.40}$
TTPCA	21.92 ± 0.54	22.14 ± 0.20	22.84 ± 0.42	21.92 ± 0.59
MPCA	21.11 ± 3.43			
R-UMLDA	7.88 ± 1.00			
Probe F				
HOOI	$\mathbf{20.77 \pm 0.92}$	$\mathbf{20.71 \pm 0.72}$	20.15 ± 0.65	19.96 ± 0.67
MPS	20.50 ± 0.40	20.52 ± 0.34	$\mathbf{20.50 \pm 0.29}$	$\mathbf{20.56 \pm 0.46}$
TTPCA	14.78 ± 0.60	14.74 ± 0.77	15.29 ± 0.75	15.40 ± 0.55
MPCA	17.12 ± 2.79			
R-UMLDA	9.67 ± 0.58			

3.3.3 Training time benchmark

An additional experiment on training time for MPS[2], HOOI, MPCA and R-UMLDA is provided to understand the computational complexity of the algorithms. For the COIL-100 dataset, we measure the elapsed training time for the training tensor of size $32 \times 32 \times 3 \times K$ ($K = 720, 3600, 6480$) for H/O= $\{0.9, 0.5, 0.1\}$, according to 10 random partitions of train/test data (*iterations*). MPCA, HOOI and R-UMLDA reduces the tensor to 32 features, and MPS to 36 (due to a fixed dimension Δ^2). In Fig. 3.4, we can see that as the number of training images increases, the MPS algorithms computational time only slightly increases, while MCPA and HOOI increases gradually, with UMLDA having the slowest performance overall.

The EYFB benchmark reduces the training tensor features to 36 (for MPS), 32 (MPCA and HOOI), and 16 (UMLDA, since the elapsed time for 32 features is too long). For this case, Fig. 3.5 demonstrates that MPCA provides the fastest computation time due to its advantage with small sample sizes (SSS). MPS performs the next best, followed by HOOI, then UMLDA with the slowest performance.

The BCI experiment involves reducing the training tensor to 36 (MPS) or 32 (MPS, HOOI and UMLDA) features and the elapsed time is shown for Subject 1 in Fig. 3.6. For this case MPS performs the quickest compared to the other algorithms, with UMLDA again performing the slowest.

Lastly, the USFG benchmark tests Probe A by reducing the MPS training tensor to 36 features, MPCA and HOOI to 32 features, and UMLDA to 16 features. Fig. 3.7 shows that MPCA provides the quickest time to extract the features, followed by MPS, HOOI and lastly UMLDA.

[2]TTPCA would be equivalent in this experiment.

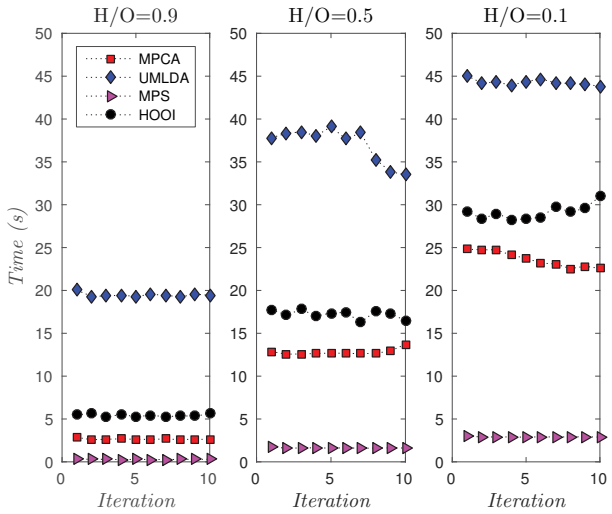


Figure 3.4 – COIL-100 training time comparison.

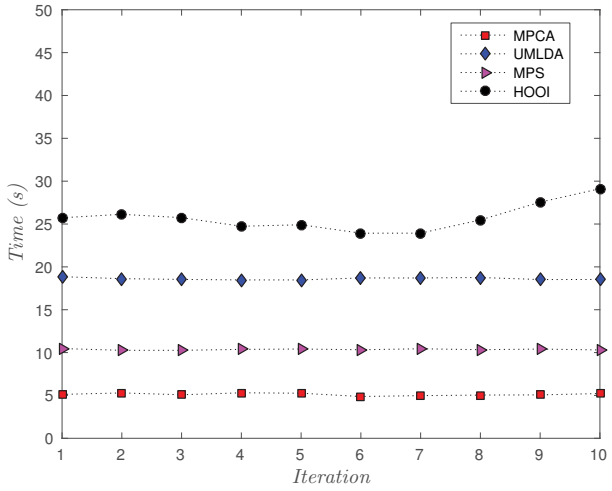


Figure 3.5 – EYFB training time comparison.

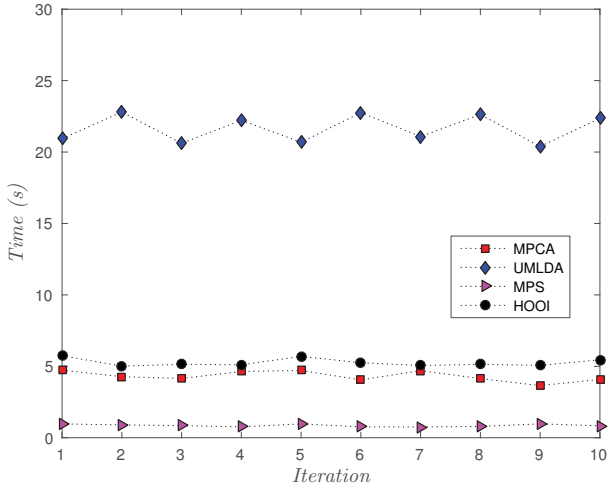


Figure 3.6 – BCI Subject 1 training time comparison.

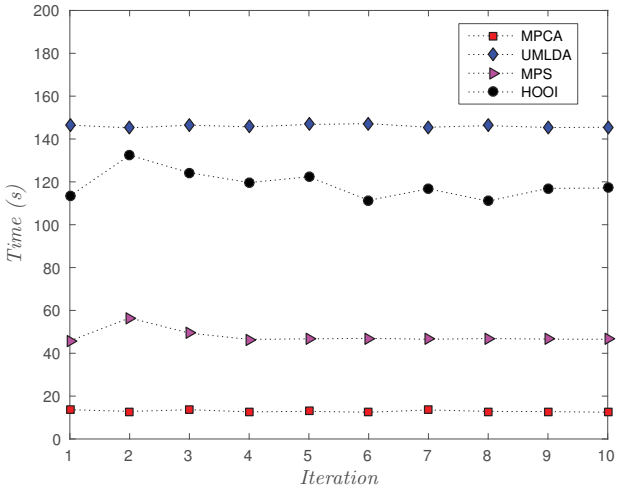


Figure 3.7 – GAIT Probe A training time comparison.

3.4 Conclusion

A a rigorous analysis of MPS and Tucker decomposition proves the efficiency of MPS in terms of retaining relevant correlations and features, which can be used directly for tensor object classification. Subsequently, two new approaches to tensor dimensionality reduction based on compressing tensors to matrices are proposed. One method reduces a tensor to a matrix, which then utilises PCA. And the other is a new multidimensional analogue to PCA known as MPS. Furthermore, a comprehensive discussion on the practical implementation of the MPS-based approach is provided, which emphasizes tensor mode permutation, tensor bond dimension control, and core matrix positioning. Numerical simulations demonstrates the efficiency of the MPS-based algorithms against other popular tensor algorithms for dimensionality reduction and tensor object classification.

Chapter 4

Matrix product states for tensor completion

4.1 Background of tensor completion

Tensors provide a natural way to represent multidimensional data whose entries are indexed by several continuous or discrete variables. For instance, a colour image is a third-order tensor defined by two indices for spatial variables and one index for colour mode. A video comprised of colour images is a fourth-order tensor with an additional index for a temporal variable. Residing in extremely high-dimensional data spaces, the tensors in practical applications are nevertheless often of *low-rank* [2]. Consequently, they can be effectively projected to much smaller subspaces through underlying decompositions such as the CANDECOMP/PARAFAC (CP), Tucker and matrix product state (MPS).

Motivated by the success of low rank matrix completion (LRMC) [92, 152], recent effort has been made to extend its concept to low rank tensor completion (LRTC). In fact, LRTC has found applications in computer vision and graphics, signal processing and machine learning [86, 153, 154, 94, 155, 156, 95]. Additionally, it has been shown in [86] that tensor-based completion algorithms are more efficient in completing tensors than matrix-based completion algorithms. The common target in LRTC is to recover missing entries of a tensor from its

partially observed entities [157, 158, 159]. LRTC remains a grand challenge due to the fact that computation for the tensor rank, defined as CP rank, is already an NP-hard problem [2]. There have been attempts in approaching LRTC via *Tucker rank* [86, 94, 95]. A conceptual drawback of Tucker rank is that its components are ranks of matrices constructed based on an unbalanced matricization scheme (one mode versus the rest). The upper bound of each individual rank is often small and may not be suitable for describing global information of the tensor. In addition, the matrix rank minimizations is only efficient when the matrix is more balanced. As the rank of a matrix is not more than $\min\{n, m\}$, where m and n are the number of rows and columns of the matrix, respectively, the high ratio $\max\{m, n\}/\min\{m, n\}$ would effectively rule out the need of matrix rank minimization. It is not surprising for present state-of-the-art LRMC methods [160, 90, 92, 152] to implicitly assume that the considered matrices are balanced.

4.2 A new approach via TT rank

Recall from Section 2.3 that the TT rank consists of a vector containing the ranks of balanced matricizations of a tensor. Its connection to MPS is based on the number of non-singular values kept (rank) at each SVD iteration (from a balanced matricization) during the calculation of the left-, right- or mixed-canonical forms of MPS. Table 4.1 provides a summary of the advantages and disadvantages of an MPS-based approach.

Low rank tensor analysis via TT rank can be seen in earlier work in physics, specifically in simulations of quantum dynamics [9, 72]. Realizing the computational efficiency of low TT rank tensors, there has been numerous works in applying it to numerical linear algebra [161, 162, 79]. Low TT rank tensors were used for the SVD of large-scale matrices in [163]. The alternating least squares (ALS) algorithms for tensor approximation [84, 82] are also used for solutions of linear equations and eigenvector/eigenvalue approximation. In [164, 165], low TT rank tensors were also used in implementing the steepest descent iteration for large scale least squares problems. The common assumption in all these works is

Table 4.1 – MPS advantages and disadvantages.

Advantages
Easy to compute
Represents each tensor mode locally
Consists of component tensors of at most third-order
Grows polynomially with tensor order
Disadvantages
Depends on SVD, which can be computationally expensive
May be computationally expensive if all singular values are retained

that all the used tensors during the computation processes are of low TT rank for computational practicability. How low TT rank tensors are relevant to real-world problems was not really their concern. Applications of the TT decomposition to fields outside of mathematics and physics has rarely been seen, with only our recent application of TT to machine learning proposed in the previous chapter. As mentioned above, colour image and video are perfect examples of tensors, so their completion can be formulated as tensor completion problems. However, it is still not known if TT rank-based completion is useful for practical solutions. The main purpose of this chapter is to show that TT rank is the right approach for LRTC, which can be addressed by TT rank-based optimization.

In this chapter we propose the following contributions:

1. Using the von Neumann entropy to show that the Tucker rank does not capture the global correlation of tensor entries, and thus is hardly ideal for LRTC. Since TT rank constitutes of ranks of matrices formed by a well-balanced matricization scheme, it is capable of capturing the global correlation of the tensor entries and is thus a promising tool for LRTC.

2. We show that unlike Tucker rank, which is often low and not appropriate for optimization, TT rank optimization is a tractable formulation for LRTC. Two new algorithms are introduced to address the TT rank optimization

based LRTC problems. The first algorithm called simple low-rank tensor completion via tensor train (SiLRTC-TT) solves an optimization problem based on the *TT nuclear norm*. The second algorithm called tensor completion by parallel matrix factorization via tensor train (TMac-TT) uses a mutilinear matrix factorization model to approximate the TT rank of a tensor, bypassing the computationally expensive SVD. Avoiding the direct TT decomposition enables the proposed algorithms to outperform other start-of-the-art tensor completion algorithms.

3. We also introduce a novel technique called *ket augmentation* (KA) to represent a low-order tensor by a higher-order tensor without changing the total number of entries. The KA scheme provides a perfect means to obtain a higher-order tensor representation of visual data by maximally exploring the potential of TT rank-based optimization for colour image and video completion. TMac-TT especially performs well in recovering videos with 95% missing entries.

4.3 Matrix and tensor completion

This section firstly revisits the conventional formulation of LRTC based on the Tucker rank. Then, we propose a new approach to LRTC via TT rank optimization, which leads to two new optimization formulations, one based on nuclear norm minimization, and the other on multilinear matrix factorization.

4.3.1 Conventional tensor completion

As tensor completion is fundamentally based on matrix completion, we give an overview of the latter prior its introduction. Recovering missing entries of a matrix $T \in \mathbb{R}^{m\times n}$ from its partially known entries given by a subset Ω can be studied via the well-known matrix-rank optimization problem [166, 167]:

$$\min_{X} \quad \text{rank}(X) \quad \text{s.t.} \quad X_\Omega = T_\Omega. \tag{4.1}$$

The missing entries of X are completed such that the rank of X is as small as possible, i.e the vector $(\lambda_1, ..., \lambda_{\min\{m,n\}})$ of the singular values λ_k of X is as sparse as possible. The sparsity of $(\lambda_1, ..., \lambda_{\min\{m,n\}})$ leads to the effective representation of X for accurate completion. Due to the combinatorial nature of the function rank$(\cdot)$, the problem (4.1), however, is NP-hard. For the nuclear norm $||X||_* = \sum_{k=1}^{\min\{m,n\}} \lambda_k$, the following convex ℓ^1 optimization problem in $(\lambda_1, ..., \lambda_{\min\{m,n\}})$ has been proved the most effective surrogate for (4.1) [160, 92, 152]:

$$\min_{X} \quad ||X||_* \quad \text{s.t.} \quad X_\Omega = T_\Omega. \tag{4.2}$$

It should be emphasized that the formulation (4.1) is efficient only when X is balanced (square), i.e. $m \approx n$. It is likely that rank$(X) \approx m$ for unbalanced X with $m \ll n$, i.e. there is not much difference between the optimal value of (4.1) and its upper bound m, under which rank optimization problem (4.1) is not interesting. More importantly, one needs at least $Cn^{6/5}\text{rank}(X)\log n \approx Cn^{6/5}m\log n$ sampled entries [90] with a positive constant C to successfully complete X, which is almost the total nm entries of X.

Completing an Nth-order tensor $\mathcal{T} \in \mathbb{R}^{I_1 \times I_2 \cdots \times I_N}$ from its known entries given by an index set Ω is formulated by the following Tucker rank optimization problem [86, 94, 95]:

$$\min_{X_{(k)}} \quad \sum_{k=1}^{N} \alpha_k \text{rank}(X_{(k)}) \quad \text{s.t.} \quad \mathcal{X}_\Omega = \mathcal{T}_\Omega. \tag{4.3}$$

where $\{\alpha_k\}_{k=1}^N$ are defined as weights fulfilling the condition $\sum_{k=1}^N \alpha_k = 1$, which is then addressed by the following ℓ^1 optimization problem [86]:

$$\min_{X_{(k)}} \quad \sum_{k=1}^{N} \alpha_k ||X_{(k)}||_* \quad \text{s.t.} \quad \mathcal{X}_\Omega = \mathcal{T}_\Omega. \tag{4.4}$$

Each matrix $X_{(k)}$ in (4.3) is obtained by matricizing the tensor along one single mode and thus is highly unbalanced. For instance, when all the modes have the same dimension ($I_1 = \cdots = I_N \equiv I$), its dimension is $I \times I^{N-1}$. As a consequence, its rank is low, which makes the matrix rank optimization formulation (4.3) less efficient for completing $\mathcal{T}$. Moreover, as analyzed above, it also makes the ℓ^1 opti-

mization problem (4.4) not efficient in addressing the rank optimization problem (4.3).

Additionally, rank$(X_{(k)})$ is not an appropriate means for capturing the *global correlation* of a tensor as it provides only the mean of the correlation between a single mode (rather than a few modes) and the rest of the tensor.
Recall from *Chapter* 2 that the rank r_k of $X_{(k)}$ is only capable of capturing the correlation between one mode k and the others. Hence, the problem (4.3) does not take into account the correlation between a few modes and the rest of the tensor, and thus may not be sufficient for completing high order tensors ($N > 3$). To overcome this weakness, in the next section, we will approach LRTC problems optimising *TT rank*, which is defined by more balanced matrices and is able to capture the hidden correlation between the modes of the tensor more effectively.

4.3.2 Tensor completion by TT rank optimization

A new approach to the LRTC problem in (4.3) is to address it by the following TT rank optimization

$$\min_{X_{[k]}} \quad \sum_{k=1}^{N-1} \alpha_k \text{rank}(X_{[k]}) \quad \text{s.t.} \quad \mathcal{X}_\Omega = \mathcal{T}_\Omega, \tag{4.5}$$

where α_k denotes the weight that the TT rank of the matrix $X_{[k]}$ contributes to, with the condition $\sum_{k=1}^{N-1} \alpha_k = 1$. Recall that $X_{[k]}$ is obtained by matricizing along k modes and thus its rank captures the correlation between k modes and the other $N-k$ modes. Therefore, $(\text{rank}(X_{[1]}), \text{rank}(X_{[2]}), ..., \text{rank}(X_{[N]}))$ provides a much better means to capture the global information of the tensor.

As the problem (4.5) is still difficult to handle as rank$(\cdot)$ is presumably hard. Therefore, from (4.5), we propose the following two problems.

The first one based on the so-called *TT nuclear norm*, defined as

$$||\mathcal{X}||_* \quad = \quad \sum_{k=1}^{N-1} \alpha_k ||X_{[k]}||_*, \tag{4.6}$$

is given by

$$\min_{\mathcal{X}} \quad \sum_{k=1}^{N-1} \alpha_k ||X_{[k]}||_* \quad \text{s.t.} \quad \mathcal{X}_\Omega = \mathcal{T}_\Omega, \tag{4.7}$$

The concerned matrices in (4.7) are much more balanced than their counterparts in (4.4). As a result, the ℓ^1 optimization problem (4.7) provides an effective means for the matrix rank optimization problem (4.5).
A particular case of (4.7) is the square model [168]

$$\min_{\mathcal{X}} \quad ||X_{[\text{round}(N/2)]}||_* \quad \text{s.t.} \quad \mathcal{X}_\Omega = \mathcal{T}_\Omega. \tag{4.8}$$

by choosing the weights such that $\alpha_k = 1$ if $k = \text{round}(N/2)$, otherwise $\alpha_k = 0$. Although the single matrix $X_{[\text{round}(N/2)]}$ is balanced and thus (4.8) is an effective means for minimizing $\text{rank}(X_{[\text{round}(N/2)]})$, it should be realized that it only captures the local correlation between $\text{round}(N/2)$ modes and other $\text{round}(N/2)$ modes.

The second problem is based on the factorization model $X_{[k]} = UV$ for a matrix $X_{[k]} \in \mathbb{R}^{m \times n}$ of rank r_k, where $U \in \mathbb{R}^{m \times r_k}$ and $V \in \mathbb{R}^{r_k \times n}$. Instead of optimizing the nuclear norm of the unfolding matrices $X_{[k]}$ as in (4.7), the Frobenius norm is minimized:

$$\begin{aligned} \min_{U_k, V_k, \mathcal{X}} \quad & \sum_{k=1}^{N-1} \frac{\alpha_k}{2} ||U_k V_k - X_{[k]}||_F^2 \\ \text{s.t.} \quad & \mathcal{X}_\Omega = \mathcal{T}_\Omega, \end{aligned} \tag{4.9}$$

where $U_k \in \mathbb{R}^{\prod_{j=1}^{k} I_j \times r_k}$ and $V_k \in \mathbb{R}^{r_k \times \prod_{j=k+1}^{N} I_j}$. This model is similar to the one proposed in [156, 95] (which is an extension of the matrix completion model [169]) where the Tucker rank is employed.

4.4 Proposed Algorithms

This section is devoted to the algorithmic development for solutions of two optimization problems (4.7) and (4.9).

4.4.1 SiLRTC-TT

To address the problem (4.7) we further convert it to the following problem:

$$\begin{aligned} \min_{\mathcal{X}, M_k} \quad & \sum_{k=1}^{N-1} \alpha_k ||M_k||_* + \frac{\beta_k}{2} ||X_{[k]} - M_k||_F^2 \\ \text{s.t.} \quad & \mathcal{X}_\Omega = \mathcal{T}_\Omega, \end{aligned} \tag{4.10}$$

where β_k are positive numbers. The central concept is based on the BCD method to alternatively optimize a group of variables while the other groups remain fixed. More specifically, the variables are divided into two main groups. The first one contains the unfolding matrices $M_1, M_2, \ldots, M_{N-1}$ and the other is tensor $\mathcal{X}$. Computing each matrix M_k is related to solving the following optimization problem:

$$\min_{M_k} \quad \alpha_k ||M_k||_* + \frac{\beta_k}{2} ||X_{[k]} - M_k||_F^2, \tag{4.11}$$

with fixed $X_{[k]}$. The optimal solution for this problem has the closed form [152] which is determined by

$$M_k = \mathbf{D}_{\gamma_k}(X_{[k]}), \tag{4.12}$$

where $\gamma_k = \frac{\alpha_k}{\beta_k}$ and $\mathbf{D}_{\gamma_k}(X_{[k]})$ denotes the thresholding SVD of $X_{[k]}$ [92]. Specifically, if the SVD of $X_{[k]} = U\lambda V^T$, its thresholding SVD is defined as:

$$\mathbf{D}_{\gamma_k}(X_{[k]}) = U\lambda_{\gamma_k} V^T, \tag{4.13}$$

where $\lambda_{\gamma_k} = diag(\max(\lambda_l - \gamma_k, 0))$. After updating all the M_k matrices, we turn into another block to compute the tensor $\mathcal{X}$ which elements are given by

$$x_{i_1 \cdots i_N} = \begin{cases} \left(\frac{\sum_{k=1}^N \beta_k \text{fold}(M_k)}{\sum_{k=1}^N \beta_k} \right)_{i_1 \cdots i_N} & (i_1 \cdots i_N) \notin \Omega \\ t_{i_1 \cdots i_N} & (i_1 \cdots i_N) \in \Omega \end{cases} \tag{4.14}$$

The pseudocode of this algorithm is given in Algorithm 4.2. We call it simple low-rank tensor completion via tensor train (SiLRTC-TT) as it is an enhancement of SiLRTC [86]. The convergence condition is reached when the relative error between two successive tensors $\mathcal{X}$ is smaller than a threshold. The algorithm is

guaranteed to be converged and gives rise to a global solution since the objective in (4.10) is a convex and the nonsmooth term is separable. We can also apply this algorithm for the square model [168] by simply choosing the weights such that $\alpha_k = 1$ if $k = \text{round}(N/2)$ otherwise $\alpha_k = 0$. For this particular case, the algorithm is defined as SiLRTC-Square.

Algorithm 4.2 – SiLRTC-TT

Input: The observed data $\mathcal{T} \in \mathbb{R}^{I_1 \times I_2 \cdots \times I_N}$, index set Ω.

Parameters: $\alpha_k, \beta_k, k = 1, \ldots, N-1$.

1: **Initialization:** $\mathcal{X}^0$, with $\mathcal{X}^0_\Omega = \mathcal{T}_\Omega$, $l = 0$.
2: **While not converged do:**
3: **for** $k = 1$ **to** $N-1$ **do**
4: Unfold the tensor $\mathcal{X}^l$ to get $X^l_{[k]}$
5: $M^{l+1}_k = \mathbf{D}_{\frac{\alpha_k}{\beta_k}}(X^l_{[k]})$
6: **end for**
7: Update $\mathcal{X}^{l+1}$ from M^{l+1}_k by (4.14)
8: **End while**

Output: The recovered tensor $\mathcal{X}$ as an approximation of $\mathcal{T}$

4.4.2 TMac-TT

To solve the problem given by (4.9), following TMac and TC-MLFM in [95] and [156], we apply the BCD method to alternatively optimize different groups of variables. Specifically, we focus on the following problem:

$$\min_{U_k, V_k, X_{[k]}} \quad ||U_k V_k - X_{[k]}||_F^2, \tag{4.15}$$

for $k = 1, 2, \ldots, N-1$. This problem is convex when each variable U_k, V_k and $X_{[k]}$ is modified while keeping the other two fixed. To update each variable, perform

the following steps:

$$U_k^{l+1} = X_{[k]}^l (V_k^l)^T (V_k^l (V_k^l)^T)^\dagger, \tag{4.16}$$

$$V_k^{l+1} = ((U_k^{l+1})^T U_k^{l+1})^\dagger (U_k^{l+1})^T) X_{[k]}^l \tag{4.17}$$

$$X_{[k]}^{l+1} = U_k^{l+1} V_k^{l+1}, \tag{4.18}$$

where "†"denotes the Moore-Penrose pseudoinverse. It was shown in [95] that we can replace (4.16) by the following:

$$U_k^{l+1} = X_{[k]}^l (V_k^l)^T, \tag{4.19}$$

to avoid computing the Moore-Penrose pseudoinverse $(V_k^l (V_k^l)^T)^\dagger$. The rationale behind this is that we only need the product $U_k^{l+1} V_k^{l+1}$ to compute $X_{[k]}^{l+1}$ in (4.18), which is the same when either (4.16) or (4.19) is used. After updating U_k^{l+1}, V_k^{l+1} and $X_{[k]}^{l+1}$ for all $k = 1, 2, \ldots, N-1$, we compute elements of the tensor $\mathcal{X}^{l+1}$ as follows:

$$x_{i_1 \cdots}^{l+1} = \begin{cases} \Big(\sum_{k=1}^{N-1} \alpha_k \text{fold}(X_{[k]}^{l+1}) \Big)_{i_1 \cdots} & (i_1 \cdots) \notin \Omega \\ t_{i_1 \cdots} & (i_1 \cdots) \in \Omega \end{cases} \tag{4.20}$$

This algorithm is defined as tensor completion by parallel matrix factorization in the concept of tensor train (TMac-TT), and its pseudocode is summarized in Algorithm 4.3. The essential advantage of this algorithm is that it avoids a lot of SVDs, and hence it can substantially save computational time.

The algorithm can also be applied for the square model [168] by choosing the weights such that $\alpha_k = 1$ if $k = \text{round}(N/2)$, otherwise $\alpha_k = 0$. For this case, we define the algorithm TMac-Square.

4.4.3 Computational complexity of algorithms

The computational complexity of the algorithms are given in Table 4.4 to complete a tensor $\mathcal{X} \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_N}$, where we assume that $I_1 = I_2 = \cdots = I_N = I$. The Tucker rank and TT rank are assumed to be equal, i.e. $r_1 = r_2 = \cdots = r_N = r$.

Algorithm 4.3 – TMac-TT

Input: The observed data $\mathcal{T} \in \mathbb{R}^{I_1 \times I_2 \cdots \times I_N}$, index set Ω.

Parameters: $\alpha_i, r_i, i = 1, \ldots, N-1$.

1: **Initialization:** $U^0, V^0, \mathcal{X}^0$, with $\mathcal{X}^0_\Omega = \mathcal{T}_\Omega$, $l = 0$.

While not converged do:

2: **for** $k = 1$ **to** $N - 1$ **do**

3: Unfold the tensor $\mathcal{X}^l$ to get $X^l_{[k]}$

4: $U^{l+1}_i = X^l_{[k]}(V^l_k)^T$

5: $V^{l+1}_k = ((U^{l+1}_k)^T U^{l+1}_k)^\dagger (U^{l+1}_k)^T X^l_{[k]}$

6: $X^{l+1}_{[k]} = U^{l+1}_k V^{l+1}_k$

7: **end**

8: Update the tensor $\mathcal{X}^{l+1}$ using (4.20)

End while

Output: The recovered tensor $\mathcal{X}$ as an approximation of $\mathcal{T}$

Table 4.4 – Computational complexity of algorithms for one iteration.

Algorithm	Computational complexity
SiLRTC	$O(NI^{N+1})$
SiLRTC-TT	$O(I^{3N/2} + I^{3N/2-1})$
TMac	$O(3NI^N r)$
TMac-TT	$O(3(N-1)I^N r)$

4.5 Tensor augmentation

In this section, we introduce *ket augmentation* (KA) to represent a low-order tensor by a higher-order one, i.e. to cast an Nth-order tensor $\mathcal{T} \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_N}$

into a Kth-order tensor $\tilde{\mathcal{T}} \in \mathbb{R}^{J_1 \times J_2 \times \cdots \times J_K}$, where $K \geq N$ and $\prod_{l=1}^{N} I_l = \prod_{l=1}^{K} J_l$. A higher-order representation of the tensor offers some important advantages. For instance, the TT decomposition is more efficient for the augmented tensor because the local structure of the data can be exploited effectively in terms of computational resources. Actually, if the tensor is slightly correlated, its augmented tensor can be represented by a low-rank TT [8, 170].

The concept of KA was originally introduced in [170] for casting a grayscale image into *real ket state* of a Hilbert space, which is simply a higher-order tensor, using an appropriate block structured addressing.

We define KA as a generalization of the original scheme to third-order tensors $\mathcal{T} \in \mathbb{R}^{I_1 \times I_2 \times I_3}$ that represent colour images, where $I_1 \times I_2 = 2^n \times 2^n$ $(n \geq 1 \in \mathbb{Z})$ is the number of pixels in the image and $I_3 = 3$ is the number of colors (red, green and blue). Let us start with an initial block, labeled as i_1, of 2×2 pixels corresponding to a single colour j (assume that the colour is indexed by j where $j = 1, 2, 3$ corresponding to red, green and blue colors, respectively). This block can be represented as

$$\mathcal{T}_{[2^1 \times 2^1 \times 1]} = \sum_{i_1=1}^{4} c_{i_1 j} \mathbf{e}_{i_1}, \tag{4.21}$$

where $c_{i_1 j}$ is the pixel value corresponding to colour j and $\mathbf{e}_{i_1}$ is the orthonormal base which is defined as $\mathbf{e}_1 = (1, 0, 0, 0)$, $\mathbf{e}_2 = (0, 1, 0, 0)$, $\mathbf{e}_3 = (0, 0, 1, 0)$ and $\mathbf{e}_4 = (0, 0, 0, 1)$. The value $i_1 = 1, 2, 3$ and 4 can be considered as labeling the up-left, up-right, down-left and down-right pixels, respectively. For all three colors, we have three blocks which are represented by

$$\mathcal{T}_{[2^1 \times 2^1 \times 3]} = \sum_{i_1=1}^{4} \sum_{j=1}^{3} c_{i_1 j} \mathbf{e}_{i_1} \otimes \mathbf{u}_j, \tag{4.22}$$

where $\mathbf{u}_j$ is also an orthonormal base which is defined as $\mathbf{u}_1 = (1, 0, 0)$, $\mathbf{u}_2 = (0, 1, 0)$, $\mathbf{u}_3 = (0, 0, 1)$. We now consider a larger block labeled as i_2 make up of four inner sub-blocks for each colour j as shown in Fig. 4.1. In total, the new block is represented by

$$\mathcal{T}_{[2^2 \times 2^2 \times 3]} = \sum_{i_2=1}^{4} \sum_{i_1=1}^{4} \sum_{j=1}^{3} c_{i_2 i_1 j} \mathbf{e}_{i_2} \otimes \mathbf{e}_{i_1} \otimes \mathbf{u}_j. \tag{4.23}$$

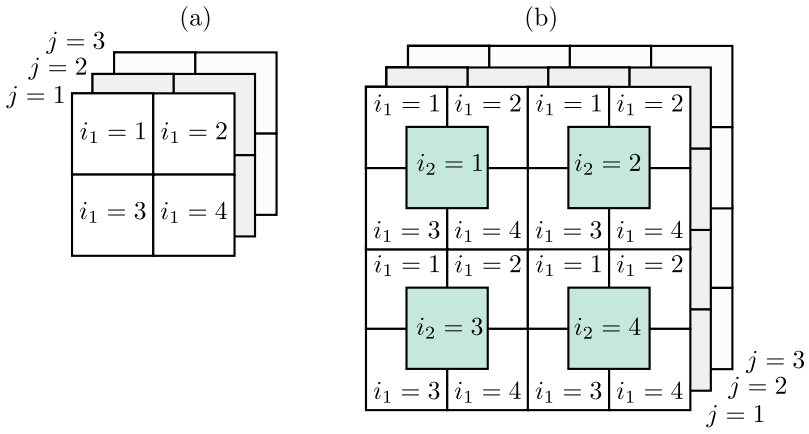


Figure 4.1 – A structured block addressing procedure to cast an image into a higher-order tensor. (a) Example for an image of size $2 \times 2 \times 3$ represented by (4.22). (b) Illustration for an image of size $2^2 \times 2^2 \times 3$ represented by (4.23).

Generally, this block structure can be extended to a size of $2^n \times 2^n \times 3$ after several steps until it can present all the values of pixels in the image. Finally, the image can be cast into an $(n+1)$th-order tensor $\mathcal{C} \in \mathbb{R}^{4\times 4\times \cdots \times 4\times 3}$ containing all the pixel values as follows,

$$\mathcal{T}_{[2^n \times 2^n \times 3]} = \sum_{i_n,\ldots,i_1=1}^{4} \sum_{j=1}^{3} c_{i_n \cdots i_1 j} \mathbf{e}_{i_n} \otimes \cdots \otimes \mathbf{e}_{i_1} \otimes \mathbf{u}_j. \tag{4.24}$$

This presentation is suitable for image processing as it not only preserves the pixels values, but also rearranges them in a higher-order tensor such that the richness of textures in the image can be studied via the correlation between modes of the tensor [170]. Therefore, due to the flexibility of the TT-rank, our proposed algorithms would ideally take advantage of KA.

4.6 Tensor completion simulations

Extensive experiments are conducted with synthetic data, colour images and videos. The proposed algorithms are benchmarked against TMac, TMac-Square, SiLRTC, SiLRTC-Square and state-of-the-art tensor completion methods FBCP [97] and STDC [171][1]. Additionally, we also benchmark the TT-rank based optimization algorithm, ALS [84, 82].

[1]Applicable only for tensors of order $N = 3$.

The simulations for the algorithms are tested with respect to different missing ratios (mr) of the test data, with mr defined as

$$mr = \frac{p}{\prod_{k=1}^{N} I_k}, \tag{4.25}$$

where p is the number of missing entries, which is chosen randomly from a tensor $\mathcal{T}$ based on a uniform distribution.

To measure performance of a LRTC algorithm, the relative square error (RSE) between the approximately recovered tensor $\mathcal{X}$ and the original one $\mathcal{T}$ is used, which is defined as,

$$RSE = ||\mathcal{X} - \mathcal{T}||_F / ||\mathcal{T}||_F. \tag{4.26}$$

The convergence criterion of our proposed algorithms is defined by computing the relative error of the tensor $\mathcal{X}$ between two successive iterations as follows:

$$\epsilon = \frac{||\mathcal{X}^{l+1} - \mathcal{X}^{l}||_F}{||\mathcal{T}||_F} \leq tol, \tag{4.27}$$

where $tol = 10^{-4}$ and the maximum number of iterations $maxiter = 1000$. These simulations are implemented under a Matlab environment.

4.6.1 Initial parameters

In the experiments there are three parameters that must be initialized: the weighting parameters α and β, and the initial TT ranks $(r_i, i = 1, \ldots, N-1)$ for TMac, TMac-TT and TMac-Square. Firstly, the weights α_k are defined as follows:

$$\alpha_k = \frac{\delta_k}{\sum_{k=1}^{N-1} \delta_k} \text{ with } \delta_k = \min(\prod_{l=1}^{k} I_l, \prod_{l=k+1}^{N} I_l), \tag{4.28}$$

where $k = 1, \ldots, N-1$. In this way, we assign the large weights to the more balanced matrices. The positive parameters are chosen by $\beta_k = f\alpha_k$, where f is empirically chosen from one of the following values in $[0.01, 0.05, 0.1, 0.5, 1]$ in such a way that the algorithm performs the best. Similarly, for SiLRTC and

TMac, the weights are chosen as follows:

$$\alpha_k = \frac{I_k}{\sum_{k=1}^{N} I_k}, \tag{4.29}$$

where $k = 1, \ldots, N$. The positive parameters are chosen such that $\beta_k = f\alpha_k$, where f is empirically chosen from one of the following values in $[0.01, 0.05, 0.1, 0.5, 1]$ which gives the best performance.

To obtain the initial TT ranks for TMac, TMac-TT and TMac-Square, each rank r_i is bounded by keeping only the singular values that satisfy the following inequality:

$$\frac{\lambda_j^{[i]}}{\lambda_1^{[i]}} > th, \tag{4.30}$$

with $j = 1, \ldots, r_i$, threshold th, and $\{\lambda_j^{[i]}\}$ is assumed to be in descending order. This condition is chosen such that the matricizations with low-rank (small correlation) will have more singular values truncated. We also choose th empirically based on the algorithms performance.

It is important to highlight that these initial parameters can affect the performance of the proposed algorithms. Consequently, the proposed algorithms performance may not necessarily be optimal and future work will need to be considered in determining the optimal TT rank and weights via automatic [97] and/or adaptive methods [95].

4.6.2 Synthetic data completion

We firstly perform the simulation on two different types of low-rank tensors which are generated synthetically in such a way that the Tucker and TT rank are known in advance.

4.6.2.1 Completion of low TT rank tensor

The Nth-order tensors $\mathcal{T} \in \mathbb{R}^{I_1 \times I_2 \cdots \times I_N}$ of TT rank $(r_1, r_2, \ldots, r_{N-1})$ are generated such that its elements is represented by a TT format [8]. Specifically, its elements

is $t_{i_1 i_2 \ldots i_N} = A^{[1]}_{i_1} A^{[2]}_{i_2} \cdots A^{[N]}_{i_N}$, where $A^{[1]} \in \mathbb{R}^{I_1 \times r_1}$, $A^{[N]} \in \mathbb{R}^{r_N \times I_N}$ and $\mathcal{A}^{[k]} \in \mathbb{R}^{r_{k-1} \times I_k \times r_k}$ with $k = 2, \ldots, N-1$ are generated randomly with respect to the standard Gaussian distribution $\mathcal{N}(0, 1)$. For simplicity, we set all components of the TT rank the same, as well as the dimension of each mode, i.e. $r_1 = r_2 = \cdots = r_{N-1} = r$ and $I_1 = I_2 = \cdots = I_N = I$.

The plots of RSE with respect to mr are shown in the Figure. 4.2 for tensors of different sizes, $40 \times 40 \times 40 \times 40$ (4D), $20 \times 20 \times 20 \times 20 \times 20$ (5D), $10 \times 10 \times 10 \times 10 \times 10 \times 10$ (6D) and $10 \times 10 \times 10 \times 10 \times 10 \times 10 \times 10$ (7D) and the corresponding TT rank tuples are $(10, 10, 10)$ (4D), $(5, 5, 5, 5)$ (5D), $(4, 4, 4, 4, 4)$ (6D) and $(4, 4, 4, 4, 4, 4)$ (7D). From the plots we can see that TMac-TT shows best performance in most cases. Particularly, TMac-TT can recover the tensor successfully despite the high missing ratios, where in most cases with high missing ratios, e.g. $mr = 0.9$, it can recover the tensor with $RSE \approx 10^{-4}$. More importantly, the proposed algorithms SiLRTC-TT and TMac-TT often performs better than their corresponding counterparts, i.e. SiLRTC and TMac in most cases. FBCP and ALS have the worst results with random synthetic data, so for the remaining synthetic data experiments, only SiLRTC, SiLRTC-Square, SiLRTC-TT, TMac, TMac-Square and TMac-TT are compared.

For a better comparison on the performance of different LRTC algorithms, we present the phase diagrams using the grayscale colour to estimate how successfully a tensor can be recovered for a range of different TT rank and missing ratios. If $RSE \leq \epsilon$ where ϵ is a small threshold, we say that the tensor is recovered successfully and is represented by a white block in the phase diagram. Otherwise, if $RSE > \epsilon$, the tensor is recovered partially with a relative error and the block colour is gray. Especially the recovery is completely failed if $RSE = 1$. Concretely, we show in Fig. 4.3 the phase diagrams for different algorithms applied to complete a 5D tensor of size $20 \times 20 \times 20 \times 20 \times 20$ where the TT rank r varies from 2 to 16 and $\epsilon = 10^{-2}$. We can see that our LRTC algorithms outperform the others. Especially, TMac-TT always recovers successfully the tensor with any TT rank and missing ratio.

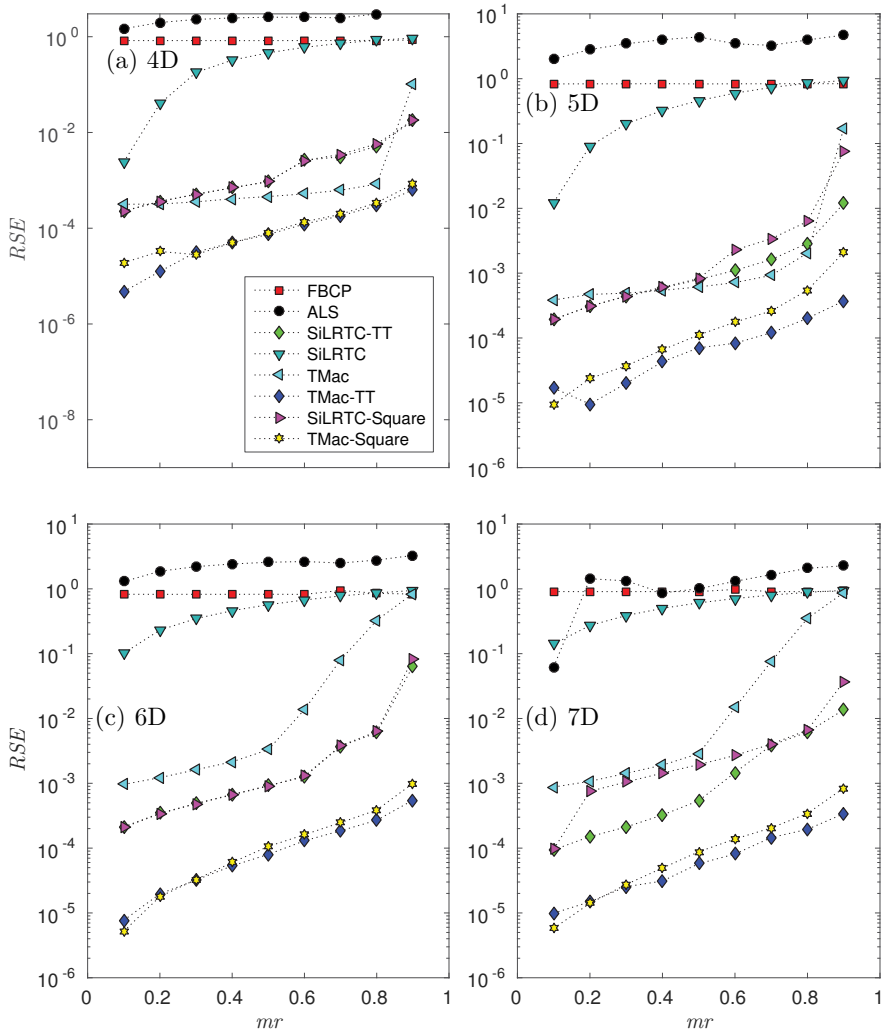


Figure 4.2 – The RSE comparison when applying different LRTC algorithms to synthetic random tensors of low TT rank. Simulation results are shown for different tensor dimensions, 4D, 5D, 6D and 7D.

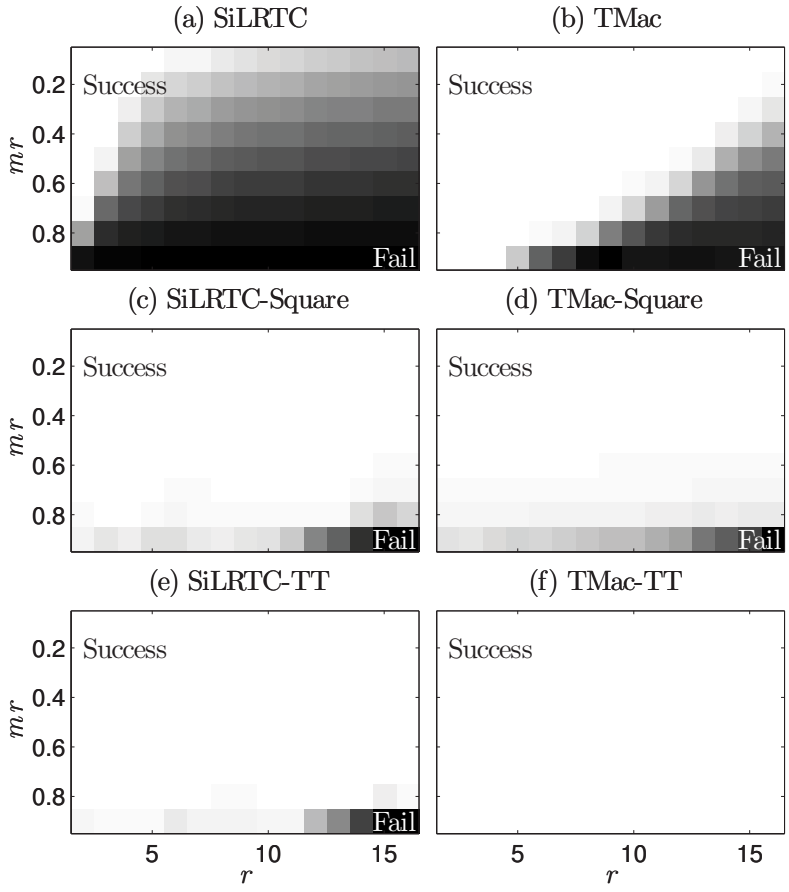


Figure 4.3 – Phase diagrams for low TT rank tensor completion when applying different algorithms to a 5D tensor.

4.6.2.2 Completion of low Tucker rank tensor

Let us now apply our proposed algorithms to synthetic random tensors of low Tucker rank. The Nth-order tensor $\mathcal{T} \in \mathbb{R}^{I_1 \times I_2 \cdots \times I_N}$ of Tucker rank $(r_1, r_2, \ldots, r_N)$ is constructed by $\mathcal{T} = \mathcal{G} \times_1 A^{(1)} \times_2 A^{(2)} \cdots \times_N A^{(N)}$, where the core tensor $\mathcal{G} \in \mathbb{R}^{r_1 \times r_2 \cdots \times r_N}$ and the factor matrices $A^{(k)} \in \mathbb{R}^{r_k \times I_k}, k = 1, \ldots, N$ are generated randomly by using the standard Gaussian distribution $\mathcal{N}(0, 1)$. Here, we choose $r_1 = r_2 = \cdots = r_N = r$ and $I_1 = I_2 = \cdots = I_N = I$ for simplicity. To compare the performance between the algorithms, we show in the Fig. 4.5 the phase diagrams for different algorithms applied to complete a 5D tensor of size $20 \times 20 \times 20 \times 20 \times 20$ where the Tucker rank r varies from 2 to 16 and $\epsilon = 10^{-2}$. We can see that both TMac and TMac-TT perform much better than the others. Besides, SiLRTC-TT shows better performance when compared to SiLRTC and SiLRTC-Square.

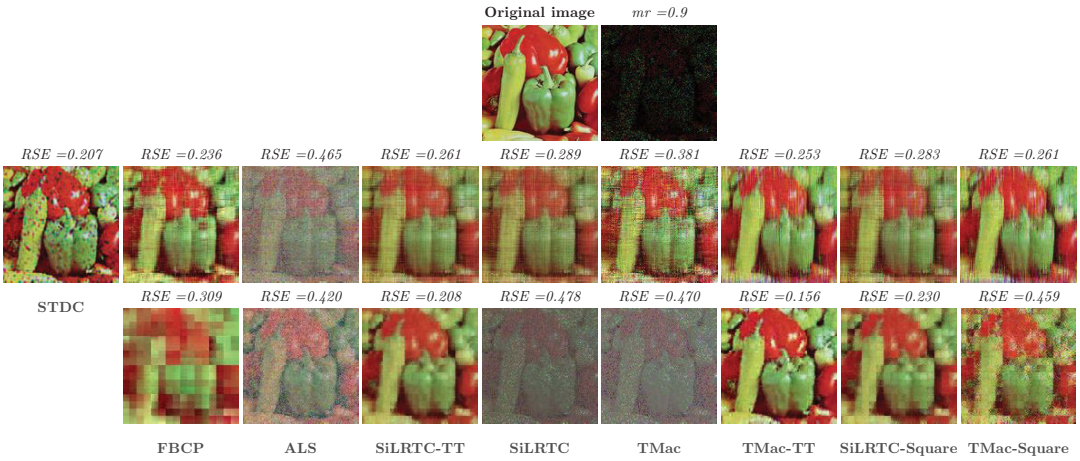


Figure 4.4 – Recover the Peppers image with 90% of missing entries using different algorithms. Top row from left to right: the original image and its copy with 90% of missing entries. Second and third rows represent the recovery results of third-order (no order augmentation) and ninth-order tensors (KA augmentation), using different algorithms: STDC (only on second row), FBCP, ALS, SiLRTC-TT, SiLRTC, TMac, TMac-TT, SiLRTC-Square and TMac-Square from the left to the right, respectively. The Tucker-based SiLRTC and TMac perform considerably worse when using KA because they are based on an unbalanced matricization scheme, which is unable to take the full advantage of KA.

Similarly, TMac-TT is better than its particular case TMac-Square.

In summary, we can see that although the tensors are generated synthetically to have low Tucker ranks, the proposed algorithms are still capable of producing results which are as good as those obtained by the Tucker-based algorithms.

The synthetic data experiments were performed to initially test the proposed algorithms. In order to have a better comparison between the algorithms we benchmark the methods against real world data such as colour images and videos, where the ranks of the tensors are not known in advance. These will be seen in the subsequent subsections.

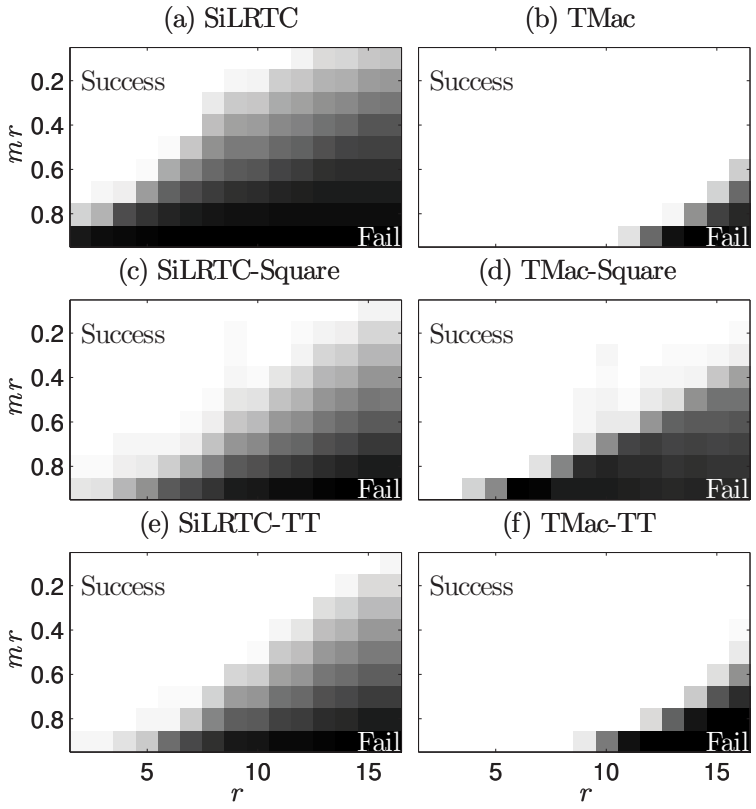


Figure 4.5 – Phase diagrams for low Tucker rank tensor completion when applying different algorithms to a 5D tensor.

Figure 4.6 – Recover the Lena image with 90% of missing entries using different algorithms. Top row from left to right: the original image and its copy with 90% of missing entries. Second and third rows represent the recovery results of third-order (no order augmentation) and ninth-order tensors (KA augmentation), using different algorithms: STDC (only on second row), FBCP, ALS, SiLRTC-TT, SiLRTC, TMac, TMac-TT, SiLRTC-Square and TMac-Square from the left to the right, respectively.

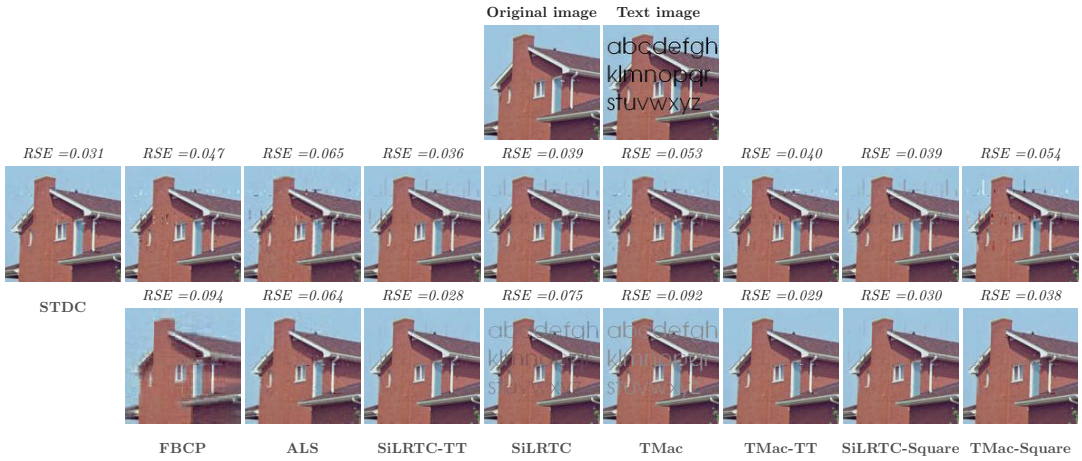


Figure 4.7 – Recover the House image with missing entries described by the white letters using different algorithms. Top row from left to right: the original image and its copy with white letters. Second and third rows represent the recovery results of third-order (no order augmentation) and ninth-order tensors (KA augmentation), using different algorithms: STDC (only on second row), FBCP, ALS, SiLRTC-TT, SiLRTC, TMac, TMac-TT, SiLRTC-Square and TMac-Square from the left to the right, respectively.

Figure 4.8 – The first frame of the bus video.

Figure 4.9 – The first frame of the city video.

Figure 4.10 – Bus video sequence tensor for the combined rows 20000:20700.

Figure 4.11 – City video sequence tensor for the combined rows 20000:20700.

4.6.3 Image completion

The colour images known as Peppers, Lena and House are employed to test the algorithms. All the images are initially represented by third-order tensors which have same sizes of $256 \times 256 \times 3$.

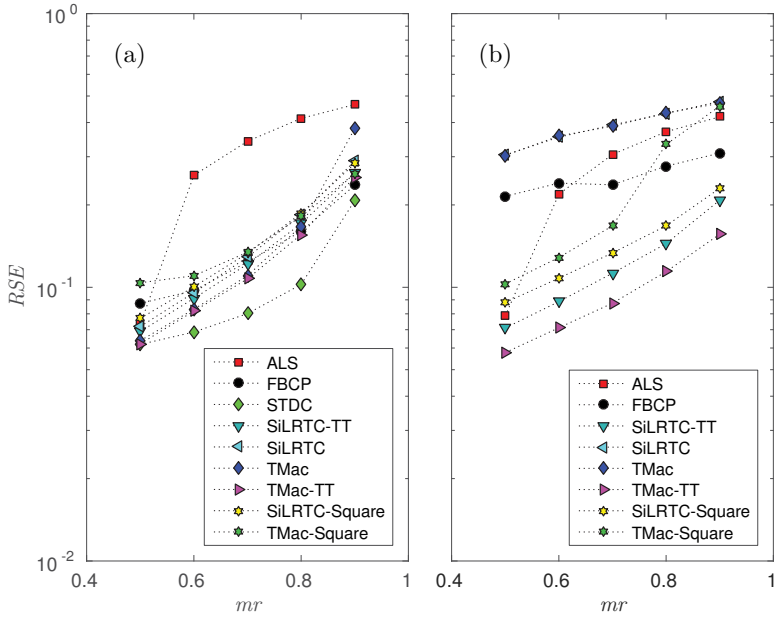


Figure 4.12 – Performance comparison between different tensor completion algorithms based on the RSE vs the missing rate when applied to the Peppers image. (a) Original tensor (no order augmentation). (b) Augmented tensor using KA scheme.

Note that when completing the third-order tensors, we do not expect the proposed methods to prevail against the conventional ones due to the fact that the TT rank of the tensor is a special case of the Tucker rank. Thus, performance of the algorithms should be mutually comparable. However, for the purpose of comparing the performance between different algorithms for real data (images) represented in terms of higher-order tensors, we apply the tensor augmentation scheme KA mentioned above to reshape third-order tensors to higher-order ones without changing the number of entries in the tensor. Specifically, we start our simulation by casting a third-order tensor $\mathcal{T} \in \mathbb{R}^{256\times256\times3}$ into a ninth-order $\tilde{\mathcal{T}} \in \mathbb{R}^{4\times4\times4\times4\times4\times4\times4\times4\times3}$ and then apply the tensor completion algorithms to impute its missing entries. We perform the simulation for the Peppers and Lena

images where missing entries of each image are chosen randomly according to a uniform distribution, the missing ratio mr varies from 0.5 to 0.9.

In Fig. 4.12, performance of the algorithms on completing the Peppers image is shown. When the image is represented by a third-order tensor, the STDC algorithm performs very well against all methods, with the ALS algorithm performing poorly, and the remaining algorithms having similar performance. However, for the case of the ninth-order tensors, the performance of the algorithms are rigorously distinguished. Specifically, our proposed algorithms (especially TMac-TT) prevails against all other methods, and this is demonstrated in Fig. 4.4 for $mr = 0.9$. Furthermore, using the KA scheme to increase the tensor order, SiLRTC-TT and TMac-TT are *at least* comparable to STDC, with TMac-TT having the lowest RSE for $mr = 0.9$. More precisely, TMac-TT gives the best result of $RSE \approx 0.156$ when using the KA scheme. The great results can be attributed to the capability of TT rank-based optimization algorithms in taking full advantage of the correlations between modes of the tensor, which represent the richness of textures in the image.

The results for the experiment performed on the Lena image for $mr = 0.9$ and recovery results are shown in Fig. 4.6 and Fig. 4.13, respectively. These figures show that again TMac-TT gives the lowest RSE for each mr thanks to its ability to utilise KA effectively.

For the House image, the missing entries are now chosen as white text, and hence the missing rate is fixed. The result is shown in Fig. 4.7. STDC provides the best performance without augmentation, while all other algorithms are comparable. However, the outlines of text can still be clearly seen on the STDC image. Using tensor augmentation, TMac-TT and TMac-Square provides the best performance, where the text is almost completely removed using TMac-TT.

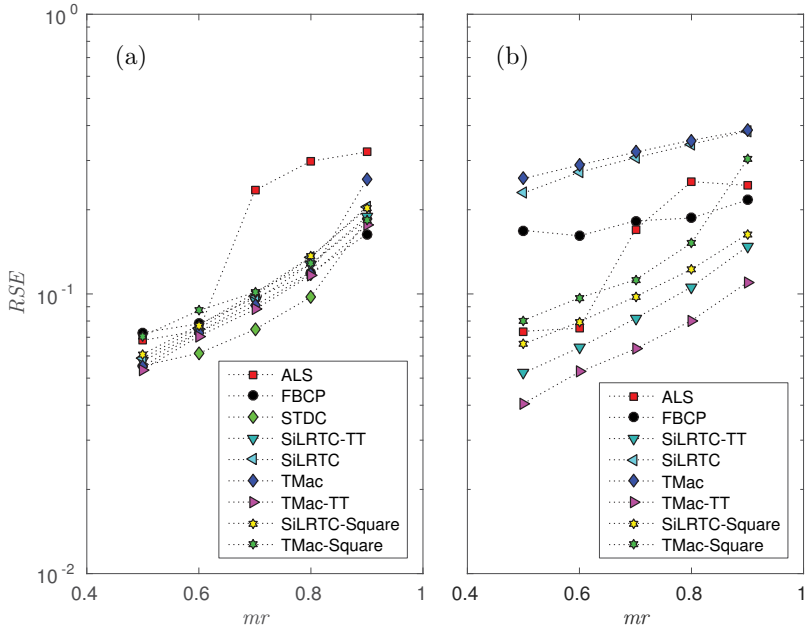


Figure 4.13 – Performance comparison between different tensor completion algorithms based on the RSE vs the missing rate when applied to the Lena image. (a) Original tensor (no order augmentation). (b) Augmented tensor using KA scheme.

4.6.4 Video completion with ket augmentation

In colour video completion we benchmark FBCP, ALS, TMac, TMac-TT and TMac-Square against two videos, *New York City (NYC)* and *bus*[2]. The other methods are computationally intractable or not applicable for $N \geq 4$ in this experiment. For each video, the following preprocessing is performed: Resize the video to a tensor of size $81 \times 729 \times 1024 \times 3$ ($frame \times image\ row \times image\ column \times RGB$). The first frame of each video can be seen in Figs. 4.9 and 4.8. The $frame$ mode is merged with the *image row* mode to form a third-order tensor, which we define here as a *video sequence tensor (VST)*, of size $59,049 \times 1024 \times 3$ ($combined\ row \times image\ column \times RGB$). Examples of the VST can be seen in the range 20000:20700 for *combined row* in Figs. 4.11 and 4.10. Hence, rather than performing an image completion on each frame, we perform our tensor completion benchmark on the *entire video*. It is important to highlight that we only benchmark with a ket augmented (not the third-order) VST due to

[2]Videos available at https://engineering.purdue.edu/~reibman/ece634/

computational intractability for high-dimensional low-order tensors.

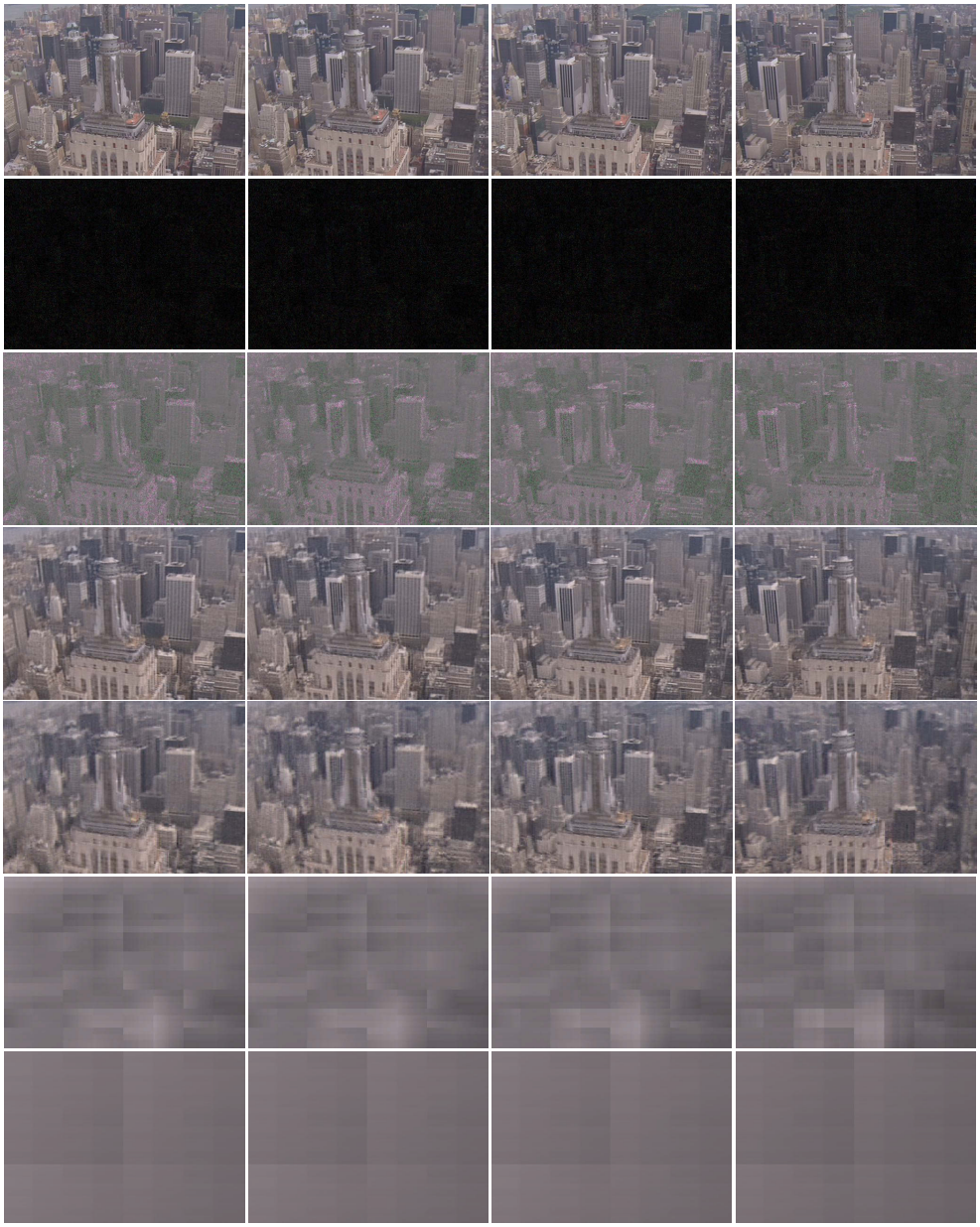


Figure 4.14 – The 7th, 21st, 33rd and 70th frames (from left to right column) in the NYC video, with each row (from top to bottom) representing the original frames, original frames with 95% missing entries, TMac, TMac-TT, TMac-Square, ALS and FBCP.

Using KA, reshape the VST to a low-dimensional high-order tensor of size $6 \times 6 \times 6 \times 6 \times 6 \times 6 \times 6 \times 6 \times 6 \times 6 \times 3$. The eleventh-order VST is directly used for the tensor completion algorithms.

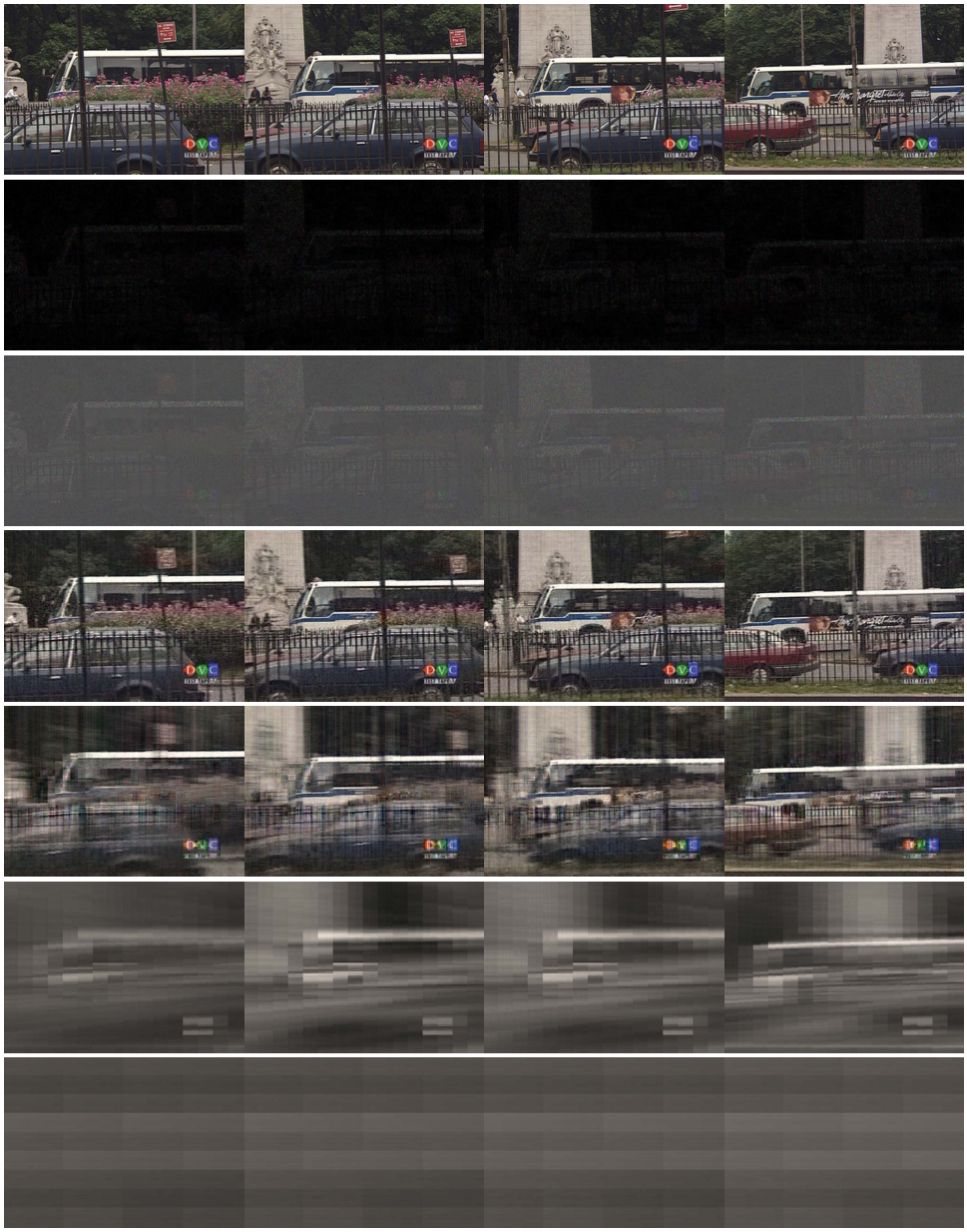


Figure 4.15 – The 7th, 21st, 33rd and 70th frames (from left to right column) in the bus video, with each row (from top to bottom) representing the original frames, original frames with 95% missing entries, TMac, TMac-TT, TMac-Square, ALS and FBCP.

Table 4.5 – RSE and SSIM tensor completion results for 95%, 90% and 70% missing entries from the NYC video.

	$mr = 0.95$		$mr = 0.9$		$mr = 0.7$	
Algorithm	RSE	SSIM	RSE	SSIM	RSE	SSIM
FBCP	0.210	0.395	0.210	0395	0.210	0.396
ALS	0.193	0.397	0.189	0.398	0.168	0.429
TMac	0.185	0.605	0.143	0.750	0.055	**0.967**
TMac-TT	**0.072**	**0.876**	**0.066**	**0.902**	**0.053**	0.949
TMac-Square	0.111	0.722	0.076	0.901	0.056	0.946

For the case of 95% missing entries, results of the benchmark can be seen in Figs. 4.14 and 4.15. The NYC results in Fig. 4.14 shows that FBCP and ALS are completely incomprehensible, whereas only the TMac-based algorithms can successfully complete the video. Moreover, in this case, TMac-TT outperforms all algorithms, which can be seen with the RSE and mean structural similarity index ($SSIM$) [172] (over all 81 *frames*) in Table 4.5 for $mr = 0.95$. For the bus results in Fig. 4.15, TMac-TT outperforms all algorithms. The other TT rank-based algorithm ALS can only manage a simple structure of the bus, and FBCP cannot produce any resemblence to the original video. Table 4.6 summarizes the RSE and mean $SSIM$ results. With 90% missing entries, the results are similar to those of 95% missing entries, however, TMac-TT and TMac-Square are now comparable in performance for the NYC video. For the NYC video with $mr = 0.7$, Table 4.5 shows that all TMac-based algorithms are comparable, with FBCP and ALS unable to reproduce a sufficient approximation. In the bus video, TMac-TT and TMac-Square provide comparable RSE and SSIM.

In summary, the bus video includes more vibrant colours and textures compared to the NYC video, which can be clearly seen from the overall $SSIM$ performance in Tables 4.5 and 4.6. It is important to highlight that TMac-TT still provides a high quality ($SSIM = 0.807$) approximation for the high missing ratio ($mr = 0.95$) test, where the next best result of TMac-Square had only $SSIM = 0.582$.

Table 4.6 – RSE and SSIM tensor completion results for 95%, 90% and 70% missing entries from the bus video.

	$mr = 0.95$		$mr = 0.9$		$mr = 0.7$	
Algorithm	RSE	SSIM	RSE	SSIM	RSE	SSIM
FBCP	0.527	0.269	0.527	0.269	0.504	0.271
ALS	0.447	0.323	0.342	0.387	0.271	0.513
TMac	0.518	0.316	0.496	0.374	0.402	0.598
TMac-TT	**0.154**	**0.807**	**0.092**	**0.932**	**0.062**	**0.974**
TMac-Square	0.267	0.582	0.196	0.781	0.077	0.968

This demonstrates the superiority of using TMac-TT over the other algorithms for high missing ratio video completion problems.

4.7 Image concatenation for colour image completion

The proposed TT rank-based algorithm TMac-TT outperformed SiLRTC and many state-of-the-art tensor completion algorithms such as FBCP and STDC in both colour image and video recovery problems. The advantage of TMac-TT is attributed to the utilization of the novel preprocessing tensor augmentation scheme known as ket augmentation (KA) that creates a structured block addressing of a tensor, which is advantageous only for TT rank-based methods. However, the disadvantage of using KA directly on an image is that block-artifacts [173] are created due to the TT rank optimization from TMac-TT. These artifacts can be seen in the result for completion of the Lena image with 90% missing entries using TMac-TT with KA (KA+TMac-TT) in Fig. 4.16. This effect is minimised for the results on colour video completion because the initial fourth-order tensor to complete is reshaped to a third-order tensor by combining the row and temporal indices. This provides different structural properties than completing each frame of the video individually, and gives potential for new patterns to assist in

Figure 4.16 – Completed Lena image using KA+TMac-TT that previously had 90% missing entries.

completing the missing entries.

In this section, we address the problem of block-artifacts caused by the tensor augmentation of a colour image. A novel framework for image completion is introduced that firstly concatenates copies of an image containing missing elements into a third-order tensor, which is inspired by the previous result in colour video completion. Then, KA is applied on the tensor, followed by the TMac-TT algorithm for tensor completion, and lastly the recovered image is extracted from the tensor. For the remainder of the section, we refer to this framework as concatenated Image Completion via Tensor Augmentation and Completion (ICTAC).

4.7.1 Modified KA

For this section we modify the KA algorithm such that

$$\tilde{\mathcal{T}}_{[3^n \times 2^n \times 3]} = \sum_{i_n,\ldots,i_1=1}^{6} \sum_{j=1}^{3} c_{i_n \cdots i_1 j} \mathbf{e}_{i_n} \otimes \cdots \otimes \mathbf{e}_{i_1} \otimes \mathbf{u}_j, \tag{4.31}$$

where each mode is $i_n = 1, \ldots, 6$, rather than $i_n = 1, \ldots, 4$ in Section 4.5. This modification is used for image concatenation, which produces a tensor with one mode significantly larger than the others, e.g. $I_1 \gg I_2$ for $\mathcal{T}$, hence the modified KA in (4.31) caters for rectangular matrices in the subspace $I_1 \times I_2$.

4.7.2 A concatenated image completion framework

The concatenated image completion via tensor augmentation and completion (ICTAC) framework is outlined in this section. The framework is divided into three main steps:

1. *Image concatenation*: The concatenation of a single image with missing entries into a third-order tensor to prepare it for KA, as discussed in Subsection 4.7.2.1.

2. *KA+TMac-TT*: The application of KA then TMac-TT on the concatenated third-order tensor to recover missing entries, as discussed in Subsection 4.7.2.2.

3. *Image extraction*: Extracting a single recovered image from the recovered concatenated tensor, which is discussed in Subsection 4.7.2.3.

4.7.2.1 Concatenating images for tensor augmentation

Consider an Nth-order colour image tensor $\mathcal{X} \in \mathbb{R}^{I_1 \times I_2 \times 3}$ that consists of partially known entries given by a subset Ω. Applying directly the KA scheme to $\mathcal{X}$, then subsequently the TMac-TT algorithm for completion, will result in blocking-artifacts as demonstrated in Fig. 4.16. To circumvent this problem, an initial preprocessing step prior to KA is added that concatenates identical copies of $\mathcal{X}$ to form a fourth-order tensor $\mathcal{X}_{ci} \in \mathbb{R}^{I_1 \times I_2 \times 3 \times C}$, where $C > 1$ is the number of copies of the tensor $\mathcal{X}$. In fact, subsection 4.6.4 had naturally formed a tensor similar to $\mathcal{X}_{ci}$ for colour video recovery, however, rather than have C for the fourth mode, the label T is used to represent the time frames of a colour video. Therefore $\mathcal{X}_{ci}$ can be considered a motionless colour video with C frames.

The next step is to permute and reshape $\mathcal{X}_{ci}$ to a third-order tensor $\mathcal{X}_{vst} \in \mathbb{R}^{\tilde{I}_1 \times I_2 \times 3}$, where $\tilde{I}_1 = CI_1$ is the *combined row mode*. Displaying $\mathcal{X}_{vst}$ would result in a distorted continuous stream of the original image, hence, its structural

properties have completely changed. The motivation to form this type of tensor is that the repetition of the image allows for more potential correlations, symmetry and/or continuity than a single image would exhibit. Additionally, applying KA on $\mathcal{X}_{vst}$ would result with obvious block-artifacts only on $\mathcal{X}_{vst}$, however, when the recovered image is extracted in the final step of ICTAC, these artifacts are minimised substantially. Fig. 4.17 demonstrates $\mathcal{X}_{vst}$ for the original Lena image with no missing entries.

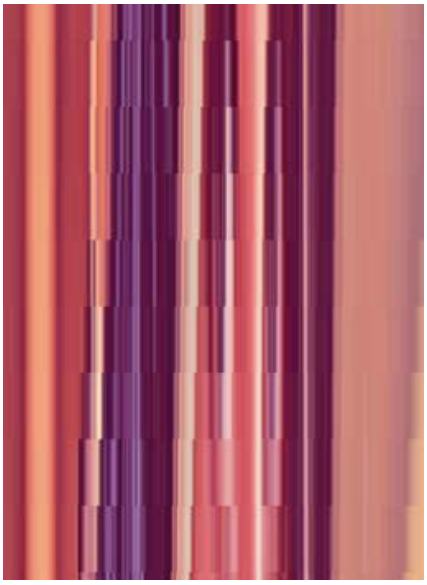


Figure 4.17 – An example of a third-order concatenated tensor $\mathcal{X}_{vst}$ of the Lena image.

We can see that there are considerably more patterns in the image compared to the original image itself. Therefore by converting an image to essentially a motionless video allows for new insights and potential for image completion tasks.

4.7.2.2 KA and TMac-TT

For the next step we apply the modified KA scheme in (4.31) to augment the third-order tensor $\mathcal{X}_{vst}$ of size $\tilde{I}_1 \times I_2 \times 3$ to a higher-order tensor $\tilde{\mathcal{X}} \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_K}$, with $K \geq N$. The modified KA scheme is needed due to subspace $\tilde{I}_1 \times I_2$ of $\mathcal{X}_{vst}$ being a large non-square matrix, and the original KA scheme only catered for

tensors of the form $I_1 \times I_2 \times 3$, with $I_1 = I_2$, hence the subspace $I_1 \times I_2$ is a square matrix.

After a block structured addressing via KA has been applied on $\mathcal{X}_{vst}$, then it is ready to be transferred to a tensor completion algorithm. The TMac-TT algorithm can now be utilised to recover the missing entries of $\tilde{\mathcal{X}}$.

4.7.2.3 Recovered image extraction

After $\tilde{\mathcal{X}}$ has been recovered via KA+TMac-TT, to recover a single image, an inverse KA scheme is utilised to obtain a third-order recovered concatenated tensor $\tilde{\mathcal{X}}_{vst} \in \mathbb{R}^{\tilde{I}_1 \times I_2 \times 3}$. $\tilde{\mathcal{X}}_{vst}$ is subsequently reshaped and permuted back to a fourth-order tensor $\tilde{\mathcal{X}}_{ci} \in \mathbb{R}^{I_1 \times I_2 \times 3 \times C}$, and from this tensor we can extract a single recovered image, e.g. the image at $\tilde{\mathcal{X}}_{ci}(:, :, :, 1)$.

4.7.3 Image recovery experiments

The experiments are conducted for image completion tasks of various missing ratios for the Lena and Peppers colour images. The proposed ICTAC framework is benchmarked against current state-of-the-art tensor completion algorithms KA+TMac-TT and SPC-QV [174].

The missing ratio (mr) as a percentage of a test image is defined as

$$mr = \frac{p}{\prod_{l=1}^{N} I_l} \times 100\%, \tag{4.32}$$

with p being the number of missing entries, which is chosen randomly based on a uniform distribution.

Performance measures include the relative square error (RSE) between an approximately recovered tensor and the original one, which is defined as,

$$RSE = ||\mathcal{X} - \mathcal{T}||_F / ||\mathcal{T}||_F, \tag{4.33}$$

and the structural similarity index ($SSIM$) [172].

Figure 4.18 – Recovery of the Lena image for 90% missing entries and 80% missing entries. Top row from left to right: the original image, the original image with 90% missing entries, and the subsequent recovery results for ICTAC, KA+TMac-TT and SPC-QV. Similarly for the bottom row from left to right: the original image with 80% missing entries, then recovery results for ICTAC, KA+TMac-TT and SPC-QV.

The algorithms ICTAC and SPC-QV recover colour images represented by third-order tensors of size $243 \times 512 \times 3$ to cater for the modified KA scheme of ICTAC as discussed in the previous sections, whereas KA+TMac-TT uses images of size $256 \times 256 \times 3$ because the traditional KA scheme works only on the condition $I_1 = I_2$. To compare the algorithms in a fair manner, the RSE and $SSIM$ is calculated based on the same image sizes for the initial tensor with missing entries and the final recovered tensor, and there is no image distortion which can change the mr throughout the runtime of the completion algorithms. The only distortion that may happen is after the algorithms have completed their calculations, where the final recovered tensors of size $243 \times 512 \times 3$ from ICTAC and SPC-QV is reshaped to $256 \times 256 \times 3$ so that a visual comparison can be made to the results of KA+TMacTT. The number of copied images $C = 81$ for all benchmarks, and simulations are conducted using a Matlab environment.

The ICTAC framework contains several tensor size transformations. For clarity,

an outline of these changes is given for an initial tensor $\mathcal{X} \in \mathbb{R}^{243\times512\times3}$ to the higher-order tensor $\tilde{\mathcal{X}}$ in Subsection 4.7.2.2:

1. *Image concatenation*: $\mathcal{X} \in \mathbb{R}^{243\times512\times3} \to \mathcal{X}_{ci} \in \mathbb{R}^{243\times512\times3\times81}$.
2. *Obtaining a VST*: $\mathcal{X}_{ci} \in \mathbb{R}^{243\times512\times3\times81} \to \mathcal{X}_{vst} \in \mathbb{R}^{19683\times512\times3}$.
3. *Applying the modified KA*: $\mathcal{X}_{vst} \in \mathbb{R}^{19683\times512\times3} \to \tilde{\mathcal{X}} \in \mathbb{R}^{6\times6\times6\times6\times6\times6\times6\times6\times6\times3}$, i.e. $n = 9$ in (4.31).

Fig. 4.18 presents the results for the Lena completed images using ICTAC, KA+TMac-TT and SPC-QV. In the case of 90% missing entries, the KA+TMac-TT algorithm performs the worst in terms of RSE and $SSIM$ compared to the other two algorithms. This is a significant result because it was shown previously that KA+TMac-TT outperformed state-of-the-art algorithms in colour image recovery. Comparing ICTAC and SPC-QV, it is interesting to see that although both algorithms had similar performance in RSE and $SSIM$, with ICTAC slightly better in both, there are quite striking visual differences in their image recovery results. Specifically, SPC-QV tends to have a smoother uniform textures of the Lena image, especially around edges, however, a slight blur and fading can be seen that affects detail such as the hair strands when comparing it to the original image. The recovered ICTAC image demonstrates an attempt to detail the finer textures of the original image, which can be clearly seen on the hair strands, however, some block-artifacts and errors from the image recovery can be seen. For 80% missing entries, ICTAC provides a recovered image almost completely similar to the original image with an SSIM of 0.983 and RSE of 0.048. SPC-QV has a slight image blur and still does not provide detail on more complicated parts of the image such as the hair and lips. The KA+TMac-TT results still has the worst performance with an RSE of 0.08, and block-artifacts can still be easily observed.

Results for the Peppers image completion task is presented in Fig. 4.19. Similar to the Lena image results for 80% and 90% missing entries, KA+TMac-TT has

the lowest RSE and $SSIM$ for both cases, with obvious block-artifacts and less detail than ICTAC and SPC-QV. The results of SPC-QV is shown to still have blurring and fading effects, which decreases the detail of the image regardless of the RSE or $SSIM$. ICTAC is shown to have slightly better RSE and $SSIM$ against SPC-QV for both cases and attempts to recover fine details of the original image, but with some minimal block-artifacts that can still be observed.

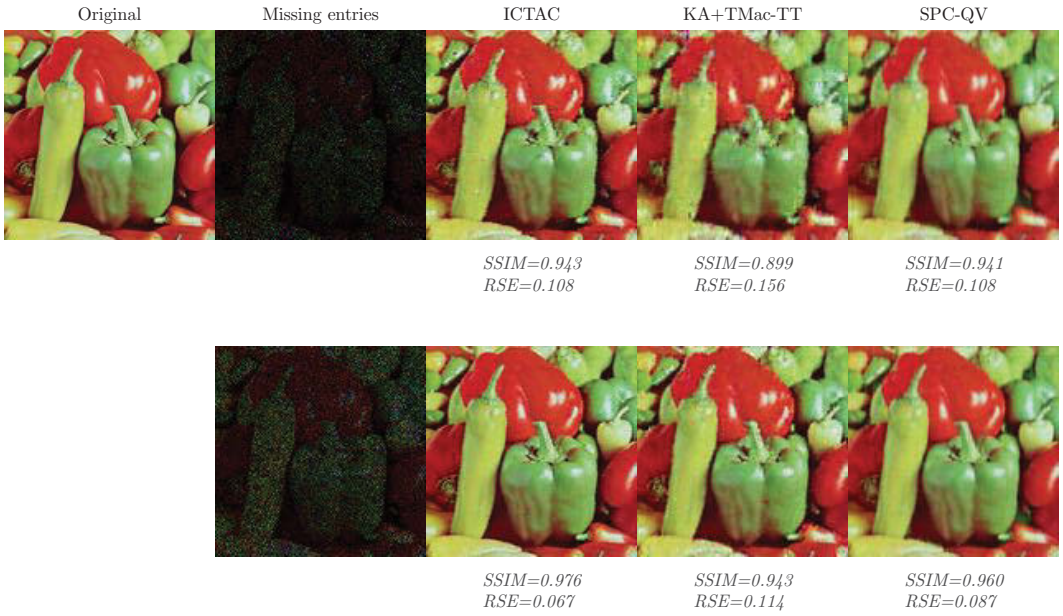


Figure 4.19 – Recovery of the Peppers image for 90% missing entries and 80% missing entries. Top row from left to right: the original image, the original image with 90% missing entries, and the subsequent recovery results for ICTAC, KA+TMac-TT and SPC-QV. Similarly for the bottom row from left to right: the original image with 80% missing entries, then recovery results for ICTAC, KA+TMac-TT and SPC-QV.

4.8 Conclusion

A novel approach to the LRTC problem based on TT rank was introduced along with corresponding algorithms for its solution. The SiLRTC-TT algorithm was defined to minimize the TT rank of a tensor by TT nuclear norm optimization. Meanwhile, TMac-TT was proposed, which is based on the multilinear matrix factorization model to minimize the TT-rank. The latter is more computationally efficient due to the fact that it does not need the SVD. The proposed algorithms are employed to simulate both synthetic and real world data represented by higher-order tensors. For synthetic data, our algorithms prevails against the others when the tensors have low TT rank. Their performance is comparable in the case of low Tucker rank tensors. The TT-based algorithms are quite promising and reliable when applied to real world data. To validate this, image and video completion problems were studied. Benchmark results show that when applied to original tensors without tensor augmentation, our algorithms are comparable to STDC in image completion. However, in the case of augmented tensors, the proposed algorithms not only outperform the others, but also provide better recovery results when compared to the case without tensor order augmentation in both image and video completion experiments.

Additionally, a novel framework known as concatenated image completion via tensor augmentation and completion (ICTAC) is proposed. The framework formulates a tensor from a concatenation of identical copies of a single colour image with missing entires, which provides additional patterns to support image completion algorithms. It then utilises tensor augmentation based on modified KA, a TT rank-based tensor completion algorithm TMac-TT to impute the missing entries, and finally, an image extraction method to recover the completed image. Our method was shown to outperform recently proposed state-of-the-art tensor completion algorithms KA+TMac-TT and SPC-QV for colour image completion tasks.

Chapter 5

Wireless sensor networks

5.1 Background

Wireless sensor networks (WSNs), which consist of spatially distributed wireless sensors, play a key role in many applications such as process monitoring in industrial plants, navigational and guidance systems, radar tracking, sonar ranging, environment monitoring, battlefield surveillance, health care and Internet of Things (IoT) [112, 175, 114, 115, 116, 176, 177, 117, 118, 178, 179, 180]. Each sensor in the network often operates in an amplify-and-forward mode [181, 182] in delivering its local observation on a target to a central system, known as the fusion center (FC). The FC filters these local observations for a global estimate of the target. The sensors may be linear or nonlinear depending upon their input-output relations. For instance, the ranging and/or bearing sensors [115] for target localisation and tracking are nonlinear. The target is often assumed to be prior Gaussian, in which case the Bayesian filter is defined via the first and second order statistical moments of the jointly Gaussian distributed source and observation [120, p. 155]. As the sensors are limited by energy resources, sensor transmitter power allocation in linear sensor networks (LSNs) via minimizing estimate distortion at the FC for scalar Gaussian targets has been a subject of considerable interest [121, 183, 184, 123, 124, 185]. Provided that the target is prior characterised by a Gaussian random variable, our previous work [125]

derived a tractable semi-definite program (SDP) for sensor power allocation in both LSNs and nonlinear sensor networks (NSNs). The SDP allows the FC to determine the best linear estimate in terms of the mean squared error (MSE) irrespective of targets that are scalar or vector, static or dynamic. The wireless communication channels between the sensors and FC have been assumed strong enough in all aforementioned works to compensate the sensor's low transmitter power. As all wireless channels suffer the common impairments such as path-loss, shadowing and small-scale fading, this assumption implicitly implies that the sensors must be in a good position relative to the FC, which is not always possible. It is known that wireless relay nodes can be deployed to act as wireless bridges to effectively assist the communication between the sensors and the FC. Multi-hop communication/relaying has been accepted as a standard to provide high capacity coverage area in next generation wireless broadband systems [186]. However, the relaying techniques for wireless transmission have not been explored in wireless sensor networks.

Meanwhile, Gaussian mixture models (GMMs) have been widely acknowledged as a better means for characterizing the target priors since they offer the flexibility of target description [128]. Indeed, GMMs have been shown to provide powerful tools in signal processing (see e.g. [187, 130, 131] and references therein). However, Bayesian filters for Gaussian mixture targets already causes computational intractability in linear models, simply because there is no closed-form of the MSE function. A particular problem has been addressed in [131] by stochastic programming.

In this chapter we will address the joint sensor and relay power allocation to optimise Bayesian filters in estimating static or dynamic targets with Gaussian mixture prior knowledge by LSNs or nonlinear sensor networks (NSNs), which non-trivially changes the nature of the power allocation and requires a different approach to the solution. To the authors's best knowledge, this problem has not been considered in literature. Our contribution is to show that this computationally intractable problem can be addressed by an iterative scalable procedure of

very low computational complexity, which converges to a stationary point after only a few iterations.

5.2 Mathematical preliminaries

Bold lower-case and upper-case symbols are used to represent vectors and matrices, respectively. By $\mathbf{A} \succeq \mathbf{B}$ it means $\mathbf{A} - \mathbf{B} \succeq 0$, i.e. $\mathbf{A} - \mathbf{B}$ is a positive definite matrix. $\mathbf{x} > 0$ for a vector $\mathbf{x}$ is component-wise understood. $\mathsf{diag}[a_i]_1^N$ or $\mathsf{diag}[a_i]_{i=1,\ldots,N}$ is a diagonal matrix with ordered diagonal entries $a_1, a_2, \ldots, a_N$, which may be scalars or matrices. The trace of a square matrix $\mathbf{A}$ is expressed by $\mathrm{tr}(\mathbf{A})$. $\mathbb{E}[.]$ is the expectation operator. $\mathbf{X} \sim p_{\mathbf{X}}(.)$ is referred to a random variable (RV) $\mathbf{X}$ with probability density function (PDF) $p_{\mathbf{X}}(.)$. $\mathbf{m}_{\mathbf{X}}$ is its expectation $\mathbb{E}[\mathbf{X}]$, while $\mathbf{C}_{\mathbf{X}}$ is its auto-covariance matrix $\mathbb{E}[(\mathbf{X} - \mathbf{m}_{\mathbf{X}})(\mathbf{X} - \mathbf{m}_{\mathbf{X}})^T]$ and $\mathbf{C}_{\mathbf{XY}}$ is its cross-covariance matrix $\mathbb{E}[(\mathbf{X} - \mathbf{m}_{\mathbf{X}})(\mathbf{Y} - \mathbf{m}_{\mathbf{Y}})^T]$ with another RV $\mathbf{Y}$. Similarly $\mathbf{R}_{\mathbf{X}}$ is its auto-correlation matrix $\mathbb{E}[\mathbf{X}\mathbf{X}^T] = \mathbf{C}_{\mathbf{X}} + \mathbf{m}_{\mathbf{X}}(\mathbf{m}_{\mathbf{X}})^T$ and $\mathbf{R}_{\mathbf{XY}}$ is its cross-correlation matrix $\mathbb{E}[\mathbf{X}\mathbf{Y}^T] = \mathbf{C}_{\mathbf{XY}} + \mathbf{m}_{\mathbf{X}}(\mathbf{m}_{\mathbf{Y}})^T$ with another RV $\mathbf{Y}$. $\mathbf{X}|\mathbf{Y}$ is a random variable $\mathbf{X}$ restricted by a realization of the conditioning random variable $\mathbf{Y}$ and accordingly $\mathbf{X}|\mathbf{Y} = \mathbf{y}$ is a random variable restricted by the value $\mathbf{Y} = \mathbf{y}$ of $\mathbf{Y}$. $\mathcal{N}(\mathbf{x}; \mathbf{m}_{\mathbf{X}}, \mathbf{C}_{\mathbf{X}}) := \frac{1}{\sqrt{2\pi \det(\mathbf{C}_{\mathbf{X}})}} \exp\left(-\frac{1}{2}(\mathbf{x} - \mathbf{m}_{\mathbf{X}})^T \mathbf{C}_{\mathbf{X}}^{-1}(\mathbf{x} - \mathbf{m}_{\mathbf{X}})\right)$ is a Gaussian distribution so $\mathbf{X} \sim \mathcal{N}(.; \mathbf{m}_{\mathbf{X}}, \mathbf{C}_{\mathbf{X}})$ means that $\mathbf{X}$ is Gaussian random varibale (RV) with expectation $\mathbf{m}_{\mathbf{X}}$ and covariance $\mathbf{C}_{\mathbf{X}}$.

5.3 Fundamental matrix inequalities for GMM

A Gaussian mixture random variable is characterized by a PDF in the form,

$$p_{\mathbf{X}}(\mathbf{x}) = \sum_{i=1}^{L} \lambda_i \mathcal{N}(\mathbf{x}; \mathbf{m}_{\mathbf{X}}{}^{(i)}, \mathbf{C}_{\mathbf{X}}{}^{(i)}), \sum_{i=1}^{L} \lambda_i = 1, \lambda_i > 0. \tag{5.1}$$

This PDF is a weighted sum of L component Gaussian PDFs

$$p_{\mathbf{X}^{(i)}}(\mathbf{x}) = \mathcal{N}(\mathbf{x}; \mathbf{m}_{\mathbf{X}}{}^{(i)}, \mathbf{C}_{\mathbf{X}}{}^{(i)}).$$

Here $\mathbf{m}_{\mathbf{X}}^{(i)}$ is the statistical mean and $\mathbf{C}_{\mathbf{X}}{}^{(i)}$ is the covariance matrix defined by $\mathbf{C}_{\mathbf{X}}^{(i)} = \mathbb{E}[(\mathbf{X}-\mathbf{m}_{\mathbf{X}}^{(i)})(\mathbf{X}-\mathbf{m}_{\mathbf{X}}^{(i)})^T]$. By straightforward calculation, the mean vector and auto-covariance matrix of such a random variable is

$$\begin{aligned}
\mathbf{m}_{\mathbf{X}} &= \sum_{i=1}^{L} \lambda_i \mathbf{m}_{\mathbf{X}}{}^{(i)} && (5.2)\\
\mathbf{C}_{\mathbf{X}} &= \sum_{i=1}^{L} \lambda_i \left[\mathbf{C}_{\mathbf{X}}^{(i)} + \mathbf{m}_{\mathbf{X}}{}^{(i)}(\mathbf{m}_{\mathbf{X}}{}^{(i)})^T\right] \\
&\quad -\mathbf{m}_{\mathbf{X}}(\mathbf{m}_{\mathbf{X}})^T && (5.3)\\
&= \sum_{i=1}^{L} \lambda_i \mathbf{R}_{\mathbf{X}}^{(i)} - \mathbf{m}_{\mathbf{X}}(\mathbf{m}_{\mathbf{X}})^T. && (5.4)
\end{aligned}$$

The last equality implies

$$\mathbf{R}_{\mathbf{X}} = \sum_{i=1}^{L} \lambda_i \mathbf{R}_{\mathbf{X}}^{(i)}. \tag{5.5}$$

It is worth noticing the following convex matrix equality

$$\begin{aligned}
&\sum_{i=1}^{L} \lambda_i \mathbf{m}_{\mathbf{X}}{}^{(i)}(\mathbf{m}_{\mathbf{X}}{}^{(i)})^T - \mathbf{m}_{\mathbf{X}}(\mathbf{m}_{\mathbf{X}})^T = \\
&\sum_{i=1}^{L} \lambda_i(1-\lambda_i)\mathbf{m}_{\mathbf{X}}^{(i)}(\mathbf{m}_{\mathbf{X}}^{(i)})^T - \sum_{i\neq j} \lambda_i\lambda_j \mathbf{m}_{\mathbf{X}}^{(i)}(\mathbf{m}_{\mathbf{X}}^{(j)})^T \\
&= \sum_{i\neq j} \lambda_i\lambda_j(\mathbf{m}_{\mathbf{X}}^{(i)} - \mathbf{m}_{\mathbf{X}}^{(j)})(\mathbf{m}_{\mathbf{X}}^{(i)} - \mathbf{m}_{\mathbf{X}}^{(j)})^T && (5.6)\\
&\succeq 0, && (5.7)
\end{aligned}$$

which together with (5.3) gives the following bound

$$\mathbf{C}_{\mathbf{X}} \succeq \sum_{i=1}^{L} \lambda_i \mathbf{C}_{\mathbf{X}}^{(i)}. \tag{5.8}$$

One of the most important features of a Gaussian PDF is its factorized representation (see e.g. [[188, Th. 2.1]])

$$\mathcal{N}((\mathbf{x},\mathbf{y});\mathbf{m}_{\mathbf{X},\mathbf{Y}},\mathbf{C}) = \mathcal{N}(\mathbf{x};\mathbf{m}_{\mathbf{X}|\mathbf{Y}},\mathbf{C}_{\mathbf{X}|\mathbf{Y}})\mathcal{N}(\mathbf{y};\mathbf{m}_{\mathbf{Y}},\mathbf{C}_{\mathbf{Y}})$$

for $\mathbf{m}_{\mathbf{X},\mathbf{Y}} := \begin{pmatrix} \mathbf{m}_{\mathbf{X}} \\ \mathbf{m}_{\mathbf{Y}} \end{pmatrix}$, $\mathbf{C} = \begin{pmatrix} \mathbf{C}_{\mathbf{X}} & \mathbf{C}_{\mathbf{XY}} \\ \mathbf{C}_{\mathbf{YX}} & \mathbf{C}_{\mathbf{Y}} \end{pmatrix}$, $\mathbf{m}_{\mathbf{X}|\mathbf{Y}} = \mathbf{m}_{\mathbf{X}} + \mathbf{C}_{\mathbf{XY}}(\mathbf{C}_{\mathbf{Y}})^{-1}(\mathbf{y} - \mathbf{m}_{\mathbf{Y}})$, $\mathbf{C}_{\mathbf{X}|\mathbf{Y}} = \mathbf{C}_{\mathbf{X}} - \mathbf{C}_{\mathbf{XY}}(\mathbf{C}_{\mathbf{Y}})^{-1}\mathbf{C}_{\mathbf{XY}}^T$. This also means

$$f_{\mathbf{X}|\mathbf{Y}}(\mathbf{x},\mathbf{y}) = \mathcal{N}(\mathbf{x},\mathbf{m}_{\mathbf{X}|\mathbf{Y}},\mathbf{C}_{\mathbf{X}|\mathbf{Y}}),$$

i.e. the conditional $\mathbf{X}|\mathbf{Y}$ of two jointly Gaussian RVs $\mathbf{X}$ and $\mathbf{Y}$ is still jointly Gaussian. Using this we can state the following result (see e.g. [[187]]):

Theorem 1: *Suppose* $(\mathbf{X}, \mathbf{Y})$ *is a jointly Gaussian mixture RV characterized by (5.28). Then*

$$f_{\mathbf{X},\mathbf{Y}}(\mathbf{x}, \mathbf{y}) = f_{\mathbf{Y}}(\mathbf{y}) \sum_{i=1}^{L} \lambda_i(\mathbf{y}) \mathcal{N}(\mathbf{x}, \mathbf{m}_{\mathbf{X}^{(i)}|\mathbf{Y}^{(i)}}, \mathbf{C}_{\mathbf{X}^{(i)}|\mathbf{Y}^{(i)}}) \tag{5.9}$$

for

$$\begin{aligned}
\mathbf{m}_{\mathbf{X}^{(i)}|\mathbf{Y}^{(i)}} &:= \mathbf{m}_{\mathbf{X}}{}^{(i)} + \mathbf{C}_{\mathbf{XY}}^{(i)} (\mathbf{C}_{\mathbf{Y}}^{(i)})^{-1} (\mathbf{y} - \mathbf{m}_{\mathbf{Y}}{}^{(i)}), \\
\mathbf{C}_{\mathbf{X}^{(i)}|\mathbf{Y}^{(i)}} &:= \mathbf{C}_{\mathbf{X}}{}^{(i)} - \mathbf{C}_{\mathbf{XY}}^{(i)} (\mathbf{C}_{\mathbf{Y}}^{(i)})^{-1} \mathbf{C}_{\mathbf{XY}}^{(i)T}, \\
f_{\mathbf{Y}}(\mathbf{y}) &:= \sum_{i=1}^{L} \lambda_i \mathcal{N}(\mathbf{y}; \mathbf{m}_{\mathbf{Y}}^{(i)}, \mathbf{C}_{\mathbf{Y}}^{(i)}),
\end{aligned}$$

and

$$\lambda_i(\mathbf{y}) := \lambda_i \mathcal{N}(\mathbf{y}; \mathbf{m}_{\mathbf{Y}}^{(i)}, \mathbf{C}_{\mathbf{Y}}^{(i)}) / f_{\mathbf{Y}}(\mathbf{y}).$$

□

It follows from the above Theorem that

$$f_{\mathbf{X}|\mathbf{Y}}(\mathbf{x}, \mathbf{y}) = \sum_{i=1}^{L} \lambda_i(\mathbf{y}) \mathcal{N}(\mathbf{x}, \mathbf{m}_{\mathbf{X}^{(i)}|\mathbf{Y}^{(i)}}, \mathbf{C}_{\mathbf{X}^{(i)}|\mathbf{Y}^{(i)}}),$$

which is Gaussian mixture in $\mathbf{x}$. The MMSE estimate for $\mathbf{X}$ based on the measurement $\mathbf{Y} = \mathbf{y}$ is

$$\hat{\mathbf{x}}_{mmse} = \mathbb{E}[\mathbf{X}|\mathbf{Y} = \mathbf{y}] := \int \mathbf{x} f_{\mathbf{X}|\mathbf{Y}=\mathbf{y}}(\mathbf{x}) d\mathbf{x} \tag{5.10}$$

$$= \sum_{i=1}^{L} \lambda_i(\mathbf{y}) \mathbf{m}_{\mathbf{X}^{(i)}|\mathbf{Y}^{(i)}}. \tag{5.11}$$

The covariance matrix $\mathbf{C}_{mmse}(\mathbf{y})$ of the estimation error is equal to the conditional covariance matrix of $\mathbf{C}_{\mathbf{X}|\mathbf{Y}=\mathbf{y}}$ of RV $\mathbf{X}|\mathbf{Y} = \mathbf{y}$:

$$\begin{aligned}
\mathbf{C}_{mmse}(\mathbf{y}) &= \mathbf{C}_{\mathbf{X}|\mathbf{Y}=\mathbf{y}} \\
&:= \int (\mathbf{x} - \hat{\mathbf{x}}_{mmse})(\mathbf{x} - \hat{\mathbf{x}}_{mmse})^T f_{\mathbf{X}|\mathbf{Y}=\mathbf{y}}(\mathbf{x}) d\mathbf{x} \qquad (5.12) \\
&= \sum_{i=1}^{L} \lambda_i(\mathbf{y}) (\mathbf{C}_{\mathbf{X}^{(i)}|\mathbf{Y}^{(i)}} + \mathbf{m}_{\mathbf{X}^{(i)}|\mathbf{Y}^{(i)}} (\mathbf{m}_{\mathbf{X}^{(i)}|\mathbf{Y}^{(i)}})^T) \\
&\quad - \hat{\mathbf{x}}_{mmse} (\hat{\mathbf{x}}_{mmse})^T. \qquad (5.13)
\end{aligned}$$

MMSE of Bayesian estimate $\mathbb{E}[||\mathbf{X}-\mathbf{X}|\mathbf{Y}=y||^2]$ for $\mathbf{X}$ based on observation $\mathbf{Y}=\mathbf{y}$ is thus

$$\begin{aligned}\epsilon^2_{mmse}(\mathbf{y}) &= \mathrm{tr}(\mathbf{C}_{mmse}(\mathbf{y}))\\ &= \sum_{i=1}^{L}\lambda_i(\mathbf{y})\left[\mathrm{tr}(\mathbf{C}_{\mathbf{X}^{(i)}|\mathbf{Y}^{(i)}})\right.\\ &\quad \left.+||\mathbf{m}_{\mathbf{X}^{(i)}|\mathbf{Y}^{(i)}}||^2\right]-||\hat{\mathbf{x}}_{mmse}||^2\Big]. \qquad (5.14)\end{aligned}$$

On the other hand, by [[189, Theorem 1]], the LMMSE estimate for $\mathbf{X}$ based on observation $\mathbf{Y}=\mathbf{y}$ is

$$\hat{\mathbf{x}}_{lmse}=\mathbf{m}_{\mathbf{X}}+\mathbf{C}^T_{\mathbf{YX}}\mathbf{C}^{-1}_{\mathbf{Y}}(\mathbf{y}-\mathbf{m}_{\mathbf{Y}}) \qquad (5.15)$$

with MSE covariance

$$\begin{aligned}\mathbf{C}_{lmse} &= \int(\mathbf{x}-\hat{\mathbf{x}}_{lmse})(\mathbf{x}-\hat{\mathbf{x}}_{lmse})^T f_{\mathbf{X}|\mathbf{Y}=\mathbf{y}}(\mathbf{x})d\mathbf{x} \qquad (5.16)\\ &= \mathbf{C}_{\mathbf{X}}-\mathbf{C}_{\mathbf{XY}}\mathbf{C}^{-1}_{\mathbf{Y}}\mathbf{C}_{\mathbf{YX}}, \qquad (5.17)\end{aligned}$$

and MSE $\epsilon^2_{lmse}=\mathrm{tr}(\mathbf{C}_{lmse})$ for all $\mathbf{y}$. Here, according to (5.2), (5.3) and (5.7)

$$\begin{aligned}\mathbf{m}_{\mathbf{X},\mathbf{Y}} &:= \begin{pmatrix}\mathbf{m}_{\mathbf{X}}\\ \mathbf{m}_{\mathbf{Y}}\end{pmatrix}=\sum_{i=1}^{L}\lambda_i\mathbf{m}^{(i)}_{\mathbf{X},\mathbf{Y}}\\ &\quad \text{for}\quad \mathbf{m}^{(i)}_{\mathbf{X},\mathbf{Y}}:=\begin{pmatrix}\mathbf{m}^{(i)}_{\mathbf{X}}\\ \mathbf{m}^{(i)}_{\mathbf{Y}}\end{pmatrix}, \qquad (5.18)\\ \begin{pmatrix}\mathbf{C}_{\mathbf{X}} & \mathbf{C}_{\mathbf{XY}}\\ \mathbf{C}_{\mathbf{YX}} & \mathbf{C}_{\mathbf{Y}}\end{pmatrix} &= \sum_{i=1}^{L}\lambda_i(\mathbf{C}^{(i)}+\mathbf{m}^{(i)}_{\mathbf{X},\mathbf{Y}}(\mathbf{m}^{(i)}_{\mathbf{X},\mathbf{Y}})^T)\\ &\quad -\mathbf{m}_{\mathbf{X},\mathbf{Y}}(\mathbf{m}_{\mathbf{X},\mathbf{Y}})^T \qquad (5.19)\\ &\succeq \mathbf{C}(\lambda), \qquad (5.20)\end{aligned}$$

for

$$\mathbf{C}(\lambda):=\begin{pmatrix}\mathbf{C}_{\mathbf{X}}(\lambda) & \mathbf{C}_{\mathbf{XY}}(\lambda)\\ \mathbf{C}_{\mathbf{YX}}(\lambda) & \mathbf{C}_{\mathbf{Y}}(\lambda)\end{pmatrix}=\sum_{i=1}^{L}\lambda_i\mathbf{C}^{(i)}.$$

The following lemma is a direct consequence of [[190, Appendix]].

Lemma 1: *(Shur's convex and monotonic inequalities) For all matrices* $\mathbf{C}^{(i)} \succeq 0$ *and* $\sum_{i=1}^{L} \lambda_i = 1, \lambda_i \geq 0$ *the following convex matrix inequality and monotonic matrix inequality hold true*

$$\mathbf{C}_{\mathbf{XY}}(\lambda)(\mathbf{C}_{\mathbf{Y}}(\lambda))^{-1}\mathbf{C}_{\mathbf{YX}}(\lambda) \preceq \sum_{i=1}^{L} \lambda_i \mathbf{C}_{\mathbf{XY}}^{(i)})(\mathbf{C}_{\mathbf{Y}}^{(i)})^{-1}\mathbf{C}_{\mathbf{YX}}^{(i)}), \tag{5.21}$$

$$\sum_{i=1}^{L}(\mathbf{C}_{\mathbf{X}}^{(i)} - \mathbf{C}_{\mathbf{XY}}^{(i)}(\mathbf{C}_{\mathbf{Y}}^{(i)})^{-1}\mathbf{C}_{\mathbf{YX}}^{(i)}) \preceq \sum_{i=1}^{L} \mathbf{C}_{\mathbf{X}}^{(i)} - (\sum_{i=1}^{L} \mathbf{C}_{\mathbf{XY}}^{(i)})(\sum_{i=1}^{L} \mathbf{C}_{\mathbf{Y}}^{(i)})^{-1}(\sum_{i=1}^{L} \mathbf{C}_{\mathbf{YX}}^{(i)}). \tag{5.22}$$

Particularly,

$$\mathbf{C}_{\mathbf{X}} - \mathbf{C}_{\mathbf{XY}}(\mathbf{C}_{\mathbf{Y}})^{-1}\mathbf{C}_{\mathbf{YX}} \preceq \mathbf{C}'_{\mathbf{X}} - \mathbf{C}'_{\mathbf{XY}}(\mathbf{C}'_{\mathbf{Y}})^{-1}\mathbf{C}'_{\mathbf{YX}} \tag{5.23}$$

for all

$$0 \preceq \begin{pmatrix} \mathbf{C}_{\mathbf{X}} & \mathbf{C}_{\mathbf{XY}} \\ \mathbf{C}_{\mathbf{YX}} & \mathbf{C}_{\mathbf{Y}} \end{pmatrix} := \mathbf{C} \preceq \mathbf{C}' := \begin{pmatrix} \mathbf{C}'_{\mathbf{X}} & \mathbf{C}'_{\mathbf{XY}} \\ \mathbf{C}'_{\mathbf{YX}} & \mathbf{C}'_{\mathbf{Y}} \end{pmatrix}.$$

Important matrix inequalities for covariances are summarized in the following theorem.

Theorem 2: *For a jointly GM characterized by equation (5.28), the following matrix inequalities hold true*

$$\mathbf{C}_{mmse}(\mathbf{y}) \preceq \mathbf{C}_{lmse} \quad \forall\, \mathbf{y}, \tag{5.24}$$

$$\sum_{i=1}^{L} \lambda_i \mathbf{C}_{\mathbf{X}^{(i)}|\mathbf{Y}^{(i)}} \preceq \mathbb{E}_{\mathbf{y}}(\mathbf{C}_{mmse}(\mathbf{y})) \preceq \mathbf{C}_{lmse}, \tag{5.25}$$

$$\sum_{i=1}^{L} \lambda_i \mathbf{C}_{\mathbf{X}^{(i)}|\mathbf{Y}^{(i)}} \preceq \mathbf{C}_{\mathbf{X}|\mathbf{Y}}(\lambda) \preceq \mathbf{C}_{lmse}, \tag{5.26}$$

where $\mathbf{C}_{\mathbf{X}|\mathbf{Y}}(\lambda) = \mathbf{C}_{\mathbf{X}}(\lambda) - \mathbf{C}_{\mathbf{XY}}(\lambda)\mathbf{C}_{\mathbf{Y}}^{-1}(\lambda)\mathbf{C}_{\mathbf{YX}}(\lambda)$.

Proof. (5.24) follows directly from the definitions (5.10), (5.12) and (5.16):

$$\begin{aligned}
\mathbf{C}_{lmse}(\mathbf{y}) - \mathbf{C}_{mmse} &= \int \mathbf{x}\mathbf{x}^T f_{\mathbf{X}|\mathbf{Y}=\mathbf{y}}(\mathbf{x})d\mathbf{x} \\
&\quad -\hat{\mathbf{x}}_{lmse}\int \mathbf{x}^T f_{\mathbf{X}|\mathbf{Y}=\mathbf{y}}(\mathbf{x})d\mathbf{x} \\
&\quad -\int \mathbf{x} f_{\mathbf{X}|\mathbf{Y}=\mathbf{y}}(\mathbf{x})d\mathbf{x} \\
&\quad +\mathbf{x}_{lmse}\mathbf{x}_{lmse}^T - \mathbf{C}_{mmse} \\
&= \mathbf{C}_{mmse} + \mathbf{x}_{mmse}\mathbf{x}_{mmse}^T - \hat{\mathbf{x}}_{lmse}\hat{\mathbf{x}}_{mmse}^T \\
&\quad -\hat{\mathbf{x}}_{mmse}\hat{\mathbf{x}}_{lmse}^T + \mathbf{x}_{lmse}\mathbf{x}_{lmse}^T - \mathbf{C}_{mmse} \\
&= (\hat{\mathbf{x}}_{lmse} - \hat{\mathbf{x}}_{mmse})(\hat{\mathbf{x}}_{lmse} - \hat{\mathbf{x}}_{mmse})^T \\
&\succeq 0 \qquad (5.27)
\end{aligned}$$

Also, by (5.7) $\sum_{i=1}^{L}\lambda_i(\mathbf{y})\mathbf{m}_{\mathbf{X}^{(i)}|\mathbf{Y}^{(i)}}(\mathbf{m}_{\mathbf{X}^{(i)}|\mathbf{Y}^{(i)}})^T \succeq \hat{\mathbf{x}}_{mmse}(\hat{\mathbf{x}}_{mmse})^T$, so (5.25) is shown as follows:

$$\begin{aligned}
\mathbb{E}_{\mathbf{y}}(\mathbf{C}_{mmse}(\mathbf{y})) &\succeq \int \sum_{i=1}^{L}\lambda_i(\mathbf{y})\mathbf{C}_{\mathbf{X}^{(i)}|\mathbf{Y}^{(i)}} f_{\mathbf{Y}}(\mathbf{y})d\mathbf{y} \\
&= \int \sum_{i=1}^{L}\lambda_i \mathcal{N}(\mathbf{y}; \mathbf{m}_{\mathbf{Y}}^{(i)}, \mathbf{C}_{\mathbf{Y}}^{(i)})\mathbf{C}_{\mathbf{X}^{(i)}|\mathbf{Y}^{(i)}}d\mathbf{y} \\
&= \sum_{i=1}^{L}\lambda_i \mathbf{C}_{\mathbf{X}^{(i)}|\mathbf{Y}^{(i)}}.
\end{aligned}$$

Finally, (5.26) is a direct consequence of Lemma 1. □

To our best knowledge, the matrix inequalities (5.24)-(5.26) have not been known in literature. Particularly, (5.25) implies the main result of [[191]]:

$$\sum_{i=1}^{L}\lambda_i \mathrm{tr}(\mathbf{C}_{\mathbf{X}^{(i)}|\mathbf{Y}^{(i)}}) \leq \mathbb{E}_{\mathbf{y}}(\epsilon_{mmse}^2(\mathbf{y})) \leq \epsilon_{lmse}^2,$$

which was proved by many involved calculations.

5.4 Joint GMM relayed equations

In statistical signal processing, detection and estimation for an object is based on the knowledge of its statistics along with noisy observations [120]. We start the

section by introducing the following joint GMM for the N-dimensional target $\mathbf{X}$ and M-sensor noisy observation $\mathbf{Y}$

$$(\mathbf{X}, \mathbf{Y}) \sim \sum_{i=1}^{L} \lambda_i \mathcal{N}\left((.,.); \mathbf{m}_{\mathbf{X},\mathbf{Y}}^{(i)}, \mathbf{C}^{(i)}\right) \tag{5.28}$$

with $\lambda_i > 0, \sum_{i=1}^{L} \lambda_i = 1$ and $\mathbf{m}_{\mathbf{X},\mathbf{Y}}^{(i)} = \begin{pmatrix} \mathbf{m}_{\mathbf{X}}{}^{(i)} \\ \mathbf{m}_{\mathbf{Y}}^{(i)} \end{pmatrix}$, $\mathbf{C}^{(i)} = \begin{pmatrix} \mathbf{C}_{\mathbf{X}}^{(i)} & \mathbf{C}_{\mathbf{XY}}^{(i)} \\ \mathbf{C}_{\mathbf{YX}}^{(i)} & \mathbf{C}_{\mathbf{Y}}^{(i)} \end{pmatrix}$.
It is well known (see e.g. [120, Chapter III], [189], [125]) that almost all results for Gaussian target estimation are based on the derivation of the joint Gaussian distribution of the target and its observation. We will see later that the joint GM distribution (5.28) facilitates unified framework for Bayesian and Kalman filters in both linear and nonlinear models.
In accordance to GMM (5.28), $\mathbf{X}$ is a Gaussian mixture (GM) $\sum_{i=1}^{L} \lambda_i \mathcal{N}(\mathbf{x}; \mathbf{m}_{\mathbf{X}}^{(i)}, \mathbf{C}_{\mathbf{X}}^{(i)})$ and $\mathbf{Y} = (\mathbf{Y}_1, ..., \mathbf{Y}_M)^T$ is a GM $\sum_{i=1}^{L} \lambda_i \mathcal{N}(\mathbf{y}; \mathbf{m}_{\mathbf{Y}}^{(i)}, \mathbf{C}_{\mathbf{Y}}^{(i)})$. The sensor observations are instantaneously sampled value $\mathbf{y} = (y_1, y_2, ..., y_M)^T$ of $\mathbf{Y}$. One can define

$$||y_j||^2 = \mathbf{C}_{\mathbf{Y}}(j,j) + \mathbf{m}_{\mathbf{Y}}^2(j), \tag{5.29}$$

where $\mathbf{C}_{\mathbf{Y}}$ is the covariance matrix of $\mathbf{Y}$ and $\mathbf{m}_{\mathbf{Y}} = \mathbb{E}[\mathbf{Y}]$. As illustrated by Figure 5.1, the sensors send these observations y_j to the relay over wireless time-orthogonal communication channels [121]. The analog signals received at the relay can thus be written as

$$z_{jR} = \sqrt{h_{jR}\alpha_j} y_j + w_{jR}, j = 1, 2, ..., M, \tag{5.30}$$

where $\sqrt{h_{jR}}$ is the channel gain between sensor j and the relay, w_{jR} is a corrupt noise, which can be assumed white with power σ_{jR} and independent from z_{jR} and $\sqrt{\alpha_j}$ controls the transmitter power $P_j = \alpha_j ||y_j||^2 = (\mathbf{C}_{\mathbf{Y}}(j,j) + \mathbf{m}_{\mathbf{Y}}^2(j))\alpha_j$ of sensor j, which is subject to a fixed sum power budget $P_T > 0$, defined as

$$\sum_{j=1}^{M} P_j = \sum_{j=1}^{M} ||y_j||^2 \alpha_j \leq P_T. \tag{5.31}$$

According to Figure 5.1, the relay will then amplify these received signals z_{jR} to power level β_j before forwarding them to the FC over wireless time-orthogonal

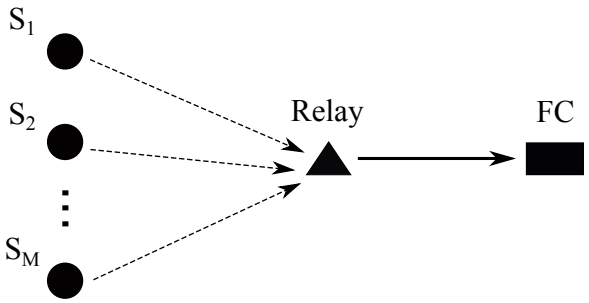


Figure 5.1 – System model

communication channels so the analog signals received at the FC are

$$\begin{aligned} z_j &= \sqrt{h_{jD}}\sqrt{\beta_j/||z_{jR}||^2}z_{jR} + w_{jD} \\ &= \sqrt{h_{jD}h_{jR}\beta_j\alpha_j/(h_{jR}||y_j||^2\alpha_j + \sigma_{jR})}y_j + w_j, \end{aligned} \tag{5.32}$$

where $\sqrt{h_{jD}}$ is the channel gain between the relay and the FC, w_{jD} is the background noise at the FC, which can be assumed to be noise with power σ_{jD} and independent from z_j. Accordingly,

$$w_j = \sqrt{h_{jD}\beta_j/(h_{jR}||y_j||^2\alpha_j + \sigma_{jR})}w_{jR} + w_{jD}$$

is white noise with power $\sigma_{jR}h_{jD}\beta_j/(h_{jR}||y_j||^2\alpha_j + \sigma_{jR}) + \sigma_{jD}$. The power levels β_j are constrained by the relay power budget P_R, defined as

$$\sum_{i=1}^{M}\beta_j \leq P_R. \tag{5.33}$$

Thus, the signals received at the FC can be written in a vector form by

$$\mathbf{Z} = \mathbf{H}_{\alpha,\beta}\mathbf{Y} + \mathbf{W}_{\alpha,\beta}, \tag{5.34}$$

where $\mathbf{H}_{\alpha,\beta} \in \mathbb{R}^{M\times M}$ is defined by

$$\mathbf{H}_{\alpha,\boldsymbol{\beta}} = \mathsf{diag}\left[\sqrt{h_{jD}h_{jR}\beta_j\alpha_j/(h_{jR}||y_j||^2\alpha_j + \sigma_{jR})}\right]_1^M$$

and the total noise $\mathbf{W}_{\alpha,\beta} \sim \mathcal{N}(.;0,\mathbf{C}_{\alpha,\beta})$ with diagonal matrix

$$\mathbf{C}_{\alpha,\beta} = \mathsf{diag}[\sigma_{jR}h_{jD}\beta_j/(h_{jR}||y_j||^2\alpha_j + \sigma_{jR}) + \sigma_{jD}]_1^M.$$

Based on the joint GM (5.28) for the target $\mathbf{X}$ and sensor noisy observation $\mathbf{Y}$ and the output equation (5.34) for relayed observation $\mathbf{Z}$, one can write the joint

distribution of the target $\mathbf{X}$ and its relayed observations $\mathbf{Z}$ at FC as

$$\begin{aligned}(\mathbf{X},\mathbf{Z}) \;\sim\; & \sum_{i=1}^{L}\lambda_i\mathcal{N}\left((.,.);\begin{pmatrix}\mathbf{m_X}^{(i)}\\ \mathbf{H}_{\boldsymbol{\alpha},\boldsymbol{\beta}}\mathbf{m}_{\mathbf{Y}}^{(i)}\end{pmatrix},\right.\\ & \left.\begin{pmatrix}\mathbf{C}_{\mathbf{X}}^{(i)} & \mathbf{C}_{\mathbf{XY}}^{(i)}\mathbf{H}_{\boldsymbol{\alpha},\boldsymbol{\beta}}\\ \mathbf{H}_{\boldsymbol{\alpha},\boldsymbol{\beta}}\mathbf{C}_{\mathbf{YX}}^{(i)} & \mathbf{H}_{\boldsymbol{\alpha},\boldsymbol{\beta}}\mathbf{C}_{\mathbf{Y}}^{(i)}\mathbf{H}_{\boldsymbol{\alpha},\boldsymbol{\beta}}+\mathbf{C}_{\boldsymbol{\alpha},\boldsymbol{\beta}}\end{pmatrix}\right)\end{aligned} \tag{5.35}$$

Accordingly,

$$\{\mathbf{X}|\mathbf{Z}=\mathbf{z}\}\sim\sum_{i=1}^{L}\lambda_i(\mathbf{z},\boldsymbol{\alpha},\boldsymbol{\beta})\mathcal{N}(.,\mathbf{m}_{\mathbf{X}^{(i)}|\mathbf{Z}^{(i)}},\mathbf{C}_{\mathbf{X}^{(i)}|\mathbf{Z}^{(i)}}), \tag{5.36}$$

where

$$\begin{aligned}\mathbf{m}_{\mathbf{X}^{(i)}|\mathbf{Z}^{(i)}} &= \mathbf{m}_{\mathbf{X}}^{(i)}+\mathbf{C}_{\mathbf{YX}}^{(i)T}\mathbf{H}_{\boldsymbol{\alpha},\boldsymbol{\beta}}(\mathbf{H}_{\boldsymbol{\alpha},\boldsymbol{\beta}}\mathbf{C}_{\mathbf{Y}}^{(i)}\mathbf{H}_{\boldsymbol{\alpha},\boldsymbol{\beta}}+\mathbf{C}_{\boldsymbol{\alpha},\boldsymbol{\beta}})^{-1}\\ &\quad\times(\mathbf{z}-\mathbf{H}_{\boldsymbol{\alpha},\boldsymbol{\beta}}\mathbf{m}_{\mathbf{Y}}^{(i)}),\end{aligned} \tag{5.37}$$

$$\lambda_i(\mathbf{z},\boldsymbol{\alpha},\boldsymbol{\beta})=\frac{\lambda_i\mathcal{N}(\mathbf{z};\mathbf{H}_{\boldsymbol{\alpha},\boldsymbol{\beta}}\mathbf{m}_{\mathbf{Y}}^{(i)},\mathbf{H}_{\boldsymbol{\alpha},\boldsymbol{\beta}}\mathbf{C}_{\mathbf{Y}}^{(i)}\mathbf{H}_{\boldsymbol{\alpha},\boldsymbol{\beta}}+\mathbf{C}_{\boldsymbol{\alpha},\boldsymbol{\beta}})}{\sum_{i=1}^{L}\lambda_i\mathcal{N}(\mathbf{z};\mathbf{H}_{\boldsymbol{\alpha},\boldsymbol{\beta}}\mathbf{m}_{\mathbf{Y}}^{(i)},\mathbf{H}_{\boldsymbol{\alpha},\boldsymbol{\beta}}\mathbf{C}_{\mathbf{Y}}^{(i)}\mathbf{H}_{\boldsymbol{\alpha},\boldsymbol{\beta}}+\mathbf{C}_{\boldsymbol{\alpha},\boldsymbol{\beta}})}, \tag{5.38}$$

$$\begin{aligned}\mathbf{C}_{\mathbf{X}^{(i)}|\mathbf{Z}^{(i)}} &= \mathbf{C}_{\mathbf{X}}^{(i)}-\mathbf{C}_{\mathbf{YX}}^{(i)T}\mathbf{H}_{\boldsymbol{\alpha},\boldsymbol{\beta}}(\mathbf{H}_{\boldsymbol{\alpha},\boldsymbol{\beta}}\mathbf{C}_{\mathbf{Y}}^{(i)}\mathbf{H}_{\boldsymbol{\alpha},\boldsymbol{\beta}}+\mathbf{C}_{\boldsymbol{\alpha},\boldsymbol{\beta}})^{-1}\\ &\quad\times\mathbf{H}_{\boldsymbol{\alpha},\boldsymbol{\beta}}\mathbf{C}_{\mathbf{YX}}^{(i)}.\end{aligned} \tag{5.39}$$

The Bayesian estimate $\hat{\mathbf{x}}(\mathbf{z})$ based on FC output $\mathbf{Z}=\mathbf{z}$ is

$$\hat{\mathbf{x}}(\mathbf{z})\triangleq\mathbb{E}[\mathbf{X}|\mathbf{Z}=\mathbf{z}]=\sum_{i=1}^{L}\lambda_i(\mathbf{z},\boldsymbol{\alpha},\boldsymbol{\beta})\mathbf{m}_{\mathbf{X}^{(i)}|\mathbf{Z}^{(i)}} \tag{5.40}$$

with the mean squared error

$$\mathbb{E}(||\hat{\mathbf{x}}(\mathbf{z})-\mathbf{x}||^2)=\mathrm{tr}(\mathbf{C}_{\mathbf{z}}(\boldsymbol{\alpha},\boldsymbol{\beta})), \tag{5.41}$$

where

$$\begin{aligned}\mathbf{C}_{\mathbf{z}}(\boldsymbol{\alpha},\boldsymbol{\beta}) &= \sum_{i=1}^{L}\lambda_i(\mathbf{z},\boldsymbol{\alpha},\boldsymbol{\beta})[\mathbf{C}_{\mathbf{X}^{(i)}|\mathbf{Z}^{(i)}}\\ &\quad+\mathbf{m}_{\mathbf{X}^{(i)}|\mathbf{Z}^{(i)}}(\mathbf{m}_{\mathbf{X}^{(i)}|\mathbf{Z}_{\boldsymbol{\alpha},\boldsymbol{\beta}}^{(i)}})^T]-\\ &\quad(\sum_{i=1}^{L}\lambda_i(\mathbf{z},\boldsymbol{\alpha},\boldsymbol{\beta})\mathbf{m}_{\mathbf{X}^{(i)}|\mathbf{Z}^{(i)}})\\ &\quad\times(\sum_{i=1}^{L}\lambda_i(\mathbf{z},\boldsymbol{\alpha},\boldsymbol{\beta})\mathbf{m}_{\mathbf{X}^{(i)}|\mathbf{Z}^{(i)}})^T\end{aligned} \tag{5.42}$$

(see (5.13)-(5.14) in the Appendix A). By defining

$$g(\alpha,\beta)=\mathbb{E}_{\mathbf{z}}(\mathbb{E}(||\hat{\mathbf{x}}(\mathbf{z})-\mathbf{x}||^2))=\mathbb{E}_{\mathbf{z}}(\mathrm{tr}(\mathbf{C}_{\mathbf{z}}(\boldsymbol{\alpha},\boldsymbol{\beta}))),$$

where $\mathbb{E}_{\mathbf{z}}$ is the expectation with respect to random variable $\mathbf{z}$, the joint sensor and relay power allocation to minimise the mean squared error of Bayesian filtering is formulated by

$$\min_{\alpha,\beta}\ g(\boldsymbol{\alpha},\boldsymbol{\beta})\quad \text{s.t.}\quad (5.31),(5.33). \tag{5.43}$$

Unfortunately, there is no closed-form for function $g(\boldsymbol{\alpha},\boldsymbol{\beta})$, making the optimisation (5.43) computationally intractable.

We now use a surrogate function for $g(\boldsymbol{\alpha},\boldsymbol{\beta})$ such that the optimisation for the former leads to the optimisation for the latter. It follows from Theorem 2 in the Appendix A that

$$\mathbf{C}_{\mathbf{z}}(\boldsymbol{\alpha},\boldsymbol{\beta})\preceq\mathbf{C}_{lmse}(\boldsymbol{\alpha},\boldsymbol{\beta})\quad\forall\ \mathbf{z}, \tag{5.44}$$

where

$$\begin{aligned}\mathbf{C}_{lmse}(\boldsymbol{\alpha},\boldsymbol{\beta}) &= \mathbf{C}_{\mathbf{X}}-\mathbf{C}_{\mathbf{YX}}^T\mathbf{H}_{\boldsymbol{\alpha},\boldsymbol{\beta}}(\mathbf{H}_{\boldsymbol{\alpha},\boldsymbol{\beta}}\mathbf{C}_{\mathbf{Y}}\mathbf{H}_{\boldsymbol{\alpha},\boldsymbol{\beta}}+\mathbf{C}_{\boldsymbol{\alpha},\boldsymbol{\beta}})^{-1}\\ &\quad\times\mathbf{H}_{\boldsymbol{\alpha},\boldsymbol{\beta}}\mathbf{C}_{\mathbf{YX}}\\ &= \mathbf{C}_{\mathbf{X}}-\mathbf{C}_{\mathbf{YX}}(\mathbf{C}_{\mathbf{Y}})^{-1}\mathbf{C}_{\mathbf{YX}}\\ &\quad+\mathbf{C}_{\mathbf{YX}}^T(\mathbf{C}_{\mathbf{Y}})^{-1}((\mathbf{C}_{\mathbf{Y}})^{-1}\\ &\quad+\mathsf{diag}[\varphi_j(\alpha_j,\beta_j)]_1^M)^{-1}(\mathbf{C}_{\mathbf{Y}})^{-1}\mathbf{C}_{\mathbf{YX}}\end{aligned} \tag{5.45}$$

with $\begin{pmatrix}\mathbf{C}_{\mathbf{X}} & \mathbf{C}_{\mathbf{XY}}\\ \mathbf{C}_{\mathbf{YX}} & \mathbf{C}_{\mathbf{Y}}\end{pmatrix}=\sum_{i=1}^{L}\lambda_i(\mathbf{C}^{(i)}+\mathbf{m}_{\mathbf{X},\mathbf{Y}}^{(i)}(\mathbf{m}_{\mathbf{X},\mathbf{Y}}^{(i)})^T)-\mathbf{m}_{\mathbf{X},\mathbf{Y}}(\mathbf{m}_{\mathbf{X},\mathbf{Y}})^T$, which is the covariance matrix of $(\mathbf{X},\mathbf{Y})$, and $\varphi_j(\alpha_j,\beta_j)=p_j\alpha_j\beta_j/(q_j\alpha_j+r_j\beta_j+\sigma_j)$, $p_j=h_{jR}h_{jD}$, $q_j=h_{jR}\sigma_{jD}||y_j||^2$, $r_j=\sigma_{jR}h_{jD}$, $\sigma_j=\sigma_{jD}\sigma_{jR}$. In fact, $\mathrm{tr}(\mathbf{C}_{lmse}(\boldsymbol{\alpha},\boldsymbol{\beta}))$ is the minimum MSE (MMSE) by linear estimator for $\mathbf{X}$ [189].

Therefore, it is true that $g(\boldsymbol{\alpha},\boldsymbol{\beta})\leq\mathrm{tr}(\mathbf{C}_{lmse}(\boldsymbol{\alpha},\boldsymbol{\beta}))\ \forall\ \boldsymbol{\alpha},\boldsymbol{\beta}$ and we seek a suboptimal solution of the computationally intractable optimisation problem (5.43) by solving its following majorant minimisation

$$\min_{\boldsymbol{\alpha}>0,\boldsymbol{\beta}>0}\ \mathrm{tr}(\mathbf{C}_{lmse}(\boldsymbol{\alpha},\boldsymbol{\beta}))\quad\text{s.t.}\quad(5.31),(5.33), \tag{5.46}$$

which by (5.45) is equivalent to the following program

$$\begin{aligned}\min_{\boldsymbol{\alpha}>0,\boldsymbol{\beta}>0} \varphi(\boldsymbol{\alpha},\boldsymbol{\beta}) &:= \mathrm{tr}(\boldsymbol{\Psi}^H(\boldsymbol{\Phi}+\mathsf{diag}[\varphi_j(\alpha_j,\beta_j)]_1^M)^{-1}\boldsymbol{\Psi}) \\ &\quad \text{subject to} \quad (5.31),(5.33), \end{aligned} \tag{5.47}$$

where $\boldsymbol{\Psi} = \mathbf{C}_{\mathbf{Y}}^{-1}\mathbf{C}_{\mathbf{YX}}$, $\boldsymbol{\Phi} = \mathbf{C}_{\mathbf{Y}}^{-1}$. Being closed-form, the objective function $\mathrm{tr}(\mathbf{C}_{lmse}(\boldsymbol{\alpha},\boldsymbol{\beta}))$ is easily computed for every $(\boldsymbol{\alpha},\boldsymbol{\beta})$. However, its optimisation is still difficult and now we develop its computational solution.
Given $(\alpha^{(\kappa)},\beta^{(\kappa)})$ feasible to (5.31), (5.33) we process the following successive approximations. Define

$$\begin{aligned}\varphi_j^{(\kappa)} &= \varphi_j(\alpha_j^{(\kappa)},\beta_j^{(\kappa)}), \\ \boldsymbol{\Theta}^{(\kappa)} &= \mathsf{diag}[\varphi_j^{(\kappa)}]_1^M(\boldsymbol{\Phi}+\mathsf{diag}[\varphi_j^{(\kappa)}]_1^M)^{-1}\boldsymbol{\Psi}\boldsymbol{\Psi}^H \\ &\quad \times(\boldsymbol{\Phi}+\mathsf{diag}[\varphi_j^{(\kappa)}]_1^M)^{-1}\mathsf{diag}[\varphi_j^{(\kappa)}]_1^M \succ 0, \\ \rho_j^{(\kappa)} &= \boldsymbol{\Theta}^{(\kappa)}(j,j) > 0, \end{aligned} \tag{5.48}$$

where $\boldsymbol{\Theta}^{(\kappa)}(j,j)$ is the j-th diagonal entry of $\Theta^{(\kappa)}$.

Theorem 3: *The following inequalities hold true for all* $\boldsymbol{\alpha}>0$ *and* $\boldsymbol{\beta}>0$,

$$\begin{aligned}\varphi(\boldsymbol{\alpha},\boldsymbol{\beta}) &\le \varphi(\alpha^{(\kappa)},\beta^{(\kappa)}) + \\ &\quad \sum_{j=1}^M \rho_j^{(\kappa)}\left(\frac{r_j}{p_j\alpha_j}+\frac{q_j}{p_j\beta_j}+\frac{\sigma_j}{p_j\alpha_j\beta_j}-\frac{1}{\varphi_j^{(\kappa)}}\right) \end{aligned} \tag{5.49}$$

$$\begin{aligned}&\le \varphi^{(\kappa)}(\boldsymbol{\alpha},\boldsymbol{\beta}) := \varphi(\alpha^{(\kappa)},\beta^{(\kappa)}) + \\ &\quad + \sum_{j=1}^M \rho_j^{(\kappa)}\left[\frac{r_j}{p_j\alpha_j}+\frac{q_j}{p_j\beta_j}+\right. \\ &\quad \left.\frac{\sigma_j}{2p_j}\left(\frac{\alpha_j^{(\kappa)}}{\beta_j^{(\kappa)}\alpha_j^2}+\frac{\beta_j^{(\kappa)}}{\alpha_j^{(\kappa)}\beta_j^2}\right)-\frac{1}{\varphi_j^{(\kappa)}}\right] \end{aligned} \tag{5.50}$$

Proof. Define a function $\chi(\phi) = \mathrm{tr}(\boldsymbol{\Psi}^H(\boldsymbol{\Phi}+\mathsf{diag}[1/\phi_j]_1^M)^{-1}\boldsymbol{\Psi})$, which by the Matrix Inverse Lemma [192] is seen as $\mathrm{tr}(\boldsymbol{\Psi}^H\mathsf{diag}[\phi_j]_1^M)\boldsymbol{\Psi})-\mathrm{tr}(\boldsymbol{\Psi}^H(\mathsf{diag}[\phi_j]_1^M(\mathsf{diag}[\phi_j]_1^M+\boldsymbol{\Phi})^{-1}\mathsf{diag}[\phi_j]_1^M)\boldsymbol{\Psi})$. The function $\chi(\phi)$ is thus concave in $\phi=(\phi_1,..,\phi_M)^T>0$ because the first term is obviously linear while the second term is convex [190,

Appendix C]. Therefore, for all $\phi > 0$ and $\phi^{(\kappa)} > 0$, it is true that [193]

$$\begin{aligned}\chi(\phi) &\leq \chi(\phi^{(\kappa)}) + \langle \nabla\chi(\phi^{(\kappa)}), \phi - \phi^{(\kappa)}\rangle \\ &= \chi(\phi^{(\kappa)}) + \langle \mathsf{diag}[1/\phi_j^{(\kappa)}]_1^M(\boldsymbol{\Phi} + \mathsf{diag}[1/\phi_j^{(\kappa)}]_1^M)^{-1} \\ &\quad \times \boldsymbol{\Psi}\boldsymbol{\Psi}^H(\boldsymbol{\Phi} + \mathsf{diag}[1/\phi_j^{(\kappa)}]_1^M)^{-1}\mathsf{diag}[1/\phi_j^{(\kappa)}]_1^M, \\ &\quad \mathsf{diag}[\phi_j]_1^M - \mathsf{diag}[\phi_j^{(\kappa)}]_1^M\rangle. \end{aligned} \tag{5.51}$$

The inequality (5.49) is obtained by replacing $\phi_j = 1/\varphi_j(\alpha_j, \beta_j) = r_j/p_j\alpha_j + q_j/p_j\beta_j + \sigma_j/p_j\alpha_j\beta_j$ and $\phi_j^{(\kappa)} = 1/\varphi_j(\alpha_j^{(\kappa)}, \beta_j^{(\kappa)})$ into the above inequality (5.51). The inequality (5.50) follows from the inequality $\dfrac{1}{\alpha_j\beta_j} \leq \dfrac{1}{2}(\dfrac{\alpha_j^{(\kappa)}}{\beta_j^{(\kappa)}}\dfrac{1}{\alpha_j^2} + \dfrac{\beta_j^{(\kappa)}}{\alpha_j^{(\kappa)}}\dfrac{1}{\beta_j^2})$. □

Thus function $\varphi^{(\kappa)}$ is a convex majorant of the highly nonconvex function φ. Accordingly, we consider the following majorant minimisation

$$\min_{\boldsymbol{\alpha},\boldsymbol{\beta}} \ \varphi^{(\kappa)}(\boldsymbol{\alpha}, \boldsymbol{\beta}) \quad \text{subject to} \quad (5.31), (5.33). \tag{5.52}$$

Proposition 1: *Whenever $(\alpha^{(\kappa)}, \beta^{(\kappa)})$ is feasible to (5.31), (5.33), the optimal solution $(\alpha^{(\kappa+1)}, \beta^{(\kappa+1)})$ of the convex program (5.52) is a feasible point of nonconvex program (5.47), which is better than $(\alpha^{(\kappa)}, \beta^{(\kappa)})$, i.e.*

$$\varphi(\alpha^{(\kappa+1)}, \beta^{(\kappa+1)}) < \varphi(\alpha^{(\kappa)}, \beta^{(\kappa)}) \tag{5.53}$$

as far as $(\alpha^{(\kappa+1)}, \beta^{(\kappa+1)}) \neq (\alpha^{(\kappa)}, \beta^{(\kappa)})$.

Proof. Note that the convex function $\varphi^{(\kappa)}$ agrees with the nonconvex function φ at $(\alpha^{(\kappa)}, \beta^{(\kappa)})$, which is also feasible to (5.31), (5.33). Therefore

$$\begin{aligned}\varphi(\alpha^{(\kappa+1)}, \beta^{(\kappa+1)}) &\leq \varphi^{(\kappa)}(\alpha^{(\kappa+1)}, \beta^{(\kappa+1)}) \\ &< \varphi^{(\kappa)}(\alpha^{(\kappa)}, \beta^{(\kappa)}) \\ &= \varphi(\alpha^{(\kappa)}, \beta^{(\kappa)}),\end{aligned}$$

showing (5.53). □

We now show that the convex program (5.52) admits the optimal solution in closed-form. Indeed, (5.52) boils down to

$$\min_{\boldsymbol{\alpha},\boldsymbol{\beta}} \ \sum_{j=1}^{M}(\frac{a_j^{(\kappa)}}{\alpha_j} + \frac{b_j^{(\kappa)}}{\beta_j} + \frac{c_j^{(\kappa)}}{2\alpha_j^2} + \frac{d_j^{(\kappa)}}{2\beta_j^2}) \quad \text{subject to} \quad (5.31), (5.33) \tag{5.54}$$

with

$$\begin{aligned} a_j^{(\kappa)} = \rho_j^{(\kappa)} r_j/p_j, b_j^{(\kappa)} = \rho_j^{(\kappa)} q_j/p_j, \\ c_j^{(\kappa)} = \rho_j^{(\kappa)} \sigma_j \alpha_j^{(\kappa)}/(p_j \beta_j^{(\kappa)}), d_j^{(\kappa)} = \rho_j^{(\kappa)} \sigma_j \beta_j^{(\kappa)}/(p_j \alpha_j^{(\kappa)}) \end{aligned} \tag{5.55}$$

By using the Lagrangian multiplier method, it can be shown that the optimal α_j and β_j are the unique positive roots of the following compressed cubic equations

$$a_j^{(\kappa)} \alpha_j + c_j^{(\kappa)} = \lambda_T ||y_j||^2 \alpha_j^3, j = 1, 2, ..., M, \tag{5.56}$$

$$b_j^{(\kappa)} \beta_j + d_j^{(\kappa)} = \lambda_R \beta_j^3, j = 1, 2, ..., M, \tag{5.57}$$

where $\lambda_T > 0$ and $\lambda_R > 0$ such that α_j and β_j satisfy the power constraints (5.31) and (5.33) at equality sign. Accordingly,[1]

$$\begin{aligned} \alpha_j^{(\kappa+1)} &= \left\{ \frac{c_j^{(\kappa)}}{2\lambda_T ||y_j||^2} + \left[(\frac{c_j^{(\kappa)}}{2\lambda_T ||y_j||^2})^2 + (\frac{a_j^{(\kappa)}}{3\lambda_T ||y_j||^2})^2 \right]^{1/2} \right\}^{1/3} \\ &\quad + \left\{ \frac{c_j^{(\kappa)}}{2\lambda_T ||y_j||^2} - \left[(\frac{c_j^{(\kappa)}}{2\lambda_T ||y_j||^2})^2 + (\frac{a_j^{(\kappa)}}{3\lambda_T ||y_j||^2})^2 \right]^{1/2} \right\}^{1/3}, \end{aligned} \tag{5.58}$$

$$\begin{aligned} \beta_j^{(\kappa+1)} &= \left\{ \frac{d_j^{(\kappa)}}{2\lambda_R} + \left[(\frac{d_j^{(\kappa)}}{2\lambda_R})^2 + (\frac{b_j^{(\kappa)}}{3\lambda_R})^2 \right]^{1/2} \right\}^{1/3} \\ &\quad + \left\{ \frac{d_j^{(\kappa)}}{2\lambda_R} - \left[(\frac{d_j^{(\kappa)}}{2\lambda_R})^2 + (\frac{b_j^{(\kappa)}}{3\lambda_R})^2 \right]^{1/2} \right\}^{1/3} \end{aligned} \tag{5.59}$$

where $\lambda_T > 0$ and λ_R are chosen so that such α_j and β_j satisfy the power constraints (5.31) and (5.33) at equality sign, which can be located by the golden search.

Algorithm 1 is a pseudocode for solving the nonconvex optimisation problem (5.47), which yields a suboptimal solution of the computationally intractable problem (5.43). The following result is a consequence of Proposition 1 and [194].

Proposition 2: *Algorithm 1 generates a sequence* $\{(\alpha^{(\kappa)}, \beta^{(\kappa)})\}$ *of improved points, which converges to an optimal solution of the nonconvex problem (5.47).*

Proof. We have shown in (5.53) that $\{(\alpha^{(\kappa)}, \beta^{(\kappa)})\}$ is a sequence of improved points to (5.47). Due to the constraints (5.31) and (5.33), the convergence of

[1] the unique positive root of cubic equation $ax^3 - cx - d = 0$ with $a > 0$, $c > 0$, $d > 0$ is $[(d/2a) + \sqrt{(d/2a)^2 + (c/3a)^2}]^{1/3} + [(d/2a) - \sqrt{(d/2a)^2 + (c/3a)^2}]^{1/3}$

Algorithm 1 Fast iterative procedure for two-hop (2H) power allocation

1: Initialize $\kappa := 0$ and $(\alpha^{(0)}, \beta^{(0)})$ feasible to (5.31) and (5.33).

2: **repeat** Generate a feasible solution $(\alpha^{(\kappa+1)}, \beta^{(\kappa+1)})$ according to formula (5.58) and (5.59).

3: **until**

$$\frac{\varphi(\alpha^{(\kappa)}, \beta^{(\kappa)}) - \varphi(\alpha^{(\kappa+1)}, \beta^{(\kappa+1)})}{\varphi(\alpha^{(\kappa)}, \beta^{(\kappa)})} \leq \epsilon, \quad \text{for a given tolerance} \quad \epsilon. \tag{5.60}$$

4: Extract $(\alpha^*, \beta^*) = (\alpha^{(\kappa)}, \beta^{(\kappa)})$ as an suboptimal solution of the computationally intractable problem (5.43).

$\{(\alpha^{(\kappa)}, \beta^{(\kappa)})\}$ can be easily shown by using Cauchy's theorem. According to [194], the limit point of $\{(\alpha^{(\kappa)}, \beta^{(\kappa)})\}$ satisfies the KKT conditions for optimality of (5.47). □

5.4.1 One-hop communication

In one-hop communication between the sensors and FC, the relay plays the role of the FC, so

$$\begin{aligned} \mathbf{C}_{\mathbf{X}|\mathbf{Z}_{\boldsymbol{\alpha}}} &= (\mathbf{C}_{\mathbf{X}} - \mathbf{C}_{\mathbf{YX}}^T(\mathbf{C}_{\mathbf{Y}})^{-1}\mathbf{C}_{\mathbf{YX}}) \\ &\quad + \mathbf{C}_{\mathbf{YX}}^T(\mathbf{C}_{\mathbf{Y}})^{-1}((\mathbf{C}_{\mathbf{Y}})^{-1} \\ &\quad + \mathsf{diag}[\varphi_j(\alpha_j)]_1^M)^{-1}(\mathbf{C}_{\mathbf{Y}})^{-1}\mathbf{C}_{\mathbf{YX}} \end{aligned} \tag{5.61}$$

with $\varphi_j(\alpha_j, \beta_j) = h_{jR}\dfrac{\alpha_j}{\sigma_j}$. Recall that h_{jR} is the channel gain from the sensor j to the FC, and σ_{jR} is the power of the background noise at FC/relay.

Consider $\min_{\boldsymbol{\alpha}>0}\ \mathrm{tr}(\mathbf{C}_{lmse}(\boldsymbol{\alpha}))$ s.t. (5.31), which is equivalent to the following program

$$\min_{\boldsymbol{\alpha}>0} \varphi(\boldsymbol{\alpha}) := \mathrm{tr}(\boldsymbol{\Psi}^H(\boldsymbol{\Phi} + \mathsf{diag}[\varphi_j(\alpha_j)]_1^M)^{-1}\boldsymbol{\Psi}) \quad \text{s.t} \quad (5.31), \tag{5.62}$$

where $\boldsymbol{\Psi} = \mathbf{C}_{\mathbf{Y}}^{-1}\mathbf{C}_{\mathbf{YX}}, \boldsymbol{\Phi} = \mathbf{C}_{\mathbf{Y}}^{-1}$. Unlike (5.47), the program (5.62) is convex, which has been solved in [[125]] by semi-definite programming (SDP). The complexity of SDP is still high for online applications and more importantly, it is not

scalable. We now develop a path-following scalable procedure for the computational solution of (5.62).

Given $\alpha^{(\kappa)}$ feasible to (5.31), we now process the following successive approximation. Define

$$\begin{aligned}
\varphi_j^{(\kappa)} &= \varphi_j(\alpha_j^{(\kappa)}), \\
\boldsymbol{\Theta}^{(\kappa)} &= \mathsf{diag}[\varphi_j^{(\kappa)}]_1^M(\boldsymbol{\Phi}+\mathsf{diag}[\varphi_j^{(\kappa)}]_1^M)^{-1}\boldsymbol{\Psi}\boldsymbol{\Psi}^H \\
&\quad \times(\boldsymbol{\Phi}+\mathsf{diag}[\varphi_j^{(\kappa)}]_1^M)^{-1}\mathsf{diag}[\varphi_j^{(\kappa)}]_1^M \succ 0, \\
\rho_j^{(\kappa)} &= \boldsymbol{\Theta}^{(\kappa)}(j,j) > 0,
\end{aligned} \tag{5.63}$$

where $\boldsymbol{\Theta}^{(\kappa)}(j,j)$ is the j-th diagonal entry of $\Theta^{(\kappa)}$. Analogously to Theorem 3 we can show that

$$\varphi(\boldsymbol{\alpha}) \leq \varphi^{(\kappa)}(\boldsymbol{\alpha}) := \varphi(\alpha^{(\kappa)},\beta^{(\kappa)}) + \sum_{j=1}^{M}\rho_j^{(\kappa)}(\frac{\sigma_{jR}}{h_{jR}\alpha_j} - \frac{1}{\varphi_j^{(\kappa)}}).$$

Accordingly, we consider the majorant minimization

$$\min_{\boldsymbol{\alpha}} \varphi^{(\kappa)}(\boldsymbol{\alpha}) \quad \text{s.t.} \quad (5.31),$$

which admits the optimal solution in closed-form

$$\alpha_j^{(\kappa+1)} = \lambda_T\sqrt{\rho_j^{(\kappa)}\sigma_{jR}/h_{jR}} \tag{5.64}$$

where $\lambda_T > 0$ such that $\alpha_j^{(\kappa+1)}$ satisfies the power constraint (5.31), i.e.

$$\lambda_T = P_T/\sum_{j=1}^{M}||y_j||^2\sqrt{\rho_j^{(\kappa)}\sigma_{jR}/h_{jR}}.$$

Algorithm 2 is a pseudocode for solving (5.62). The limit point by Algorithm 2 is the global optimal solution of (5.62) because it satisfies KKT conditions of the convex program (5.62).

5.5 Applications to static target localization

Let us emphasise that the receive equations (5.30) and (5.34) for the relay and FC and the power budget constraints (5.31) and (5.33) are generic for whatever sensor

Algorithm 2 Fast iterative procedure for 1H power allocation

1: Initialize $\kappa := 0$ and $\alpha^{(0)}$ feasible to (5.31).

2: **repeat** Generate a feasible solution $\alpha^{(\kappa+1)}$ according to formula (5.64) .

3: **until**

$$\frac{\varphi(\alpha^{(\kappa)}) - \varphi(\alpha^{(\kappa+1)})}{\varphi(\alpha^{(\kappa)})} \leq \epsilon \quad \text{for a given tolerance} \quad \epsilon. \tag{5.65}$$

4: Extract $\alpha^* = \alpha^{(\kappa)}$ as the solution of 1H power allocation.

networks. In this section we show how linear and nonlinear sensor input-output equations facilitate the joint GM distribution (5.28) and thus utilise Algorithm 1 for sensor and relay power allocation in locating a GM target.

5.5.1 Linear sensor networks

For M linear sensors observing a GM target $\mathbf{X} \sim \sum_{i=1}^{L} \lambda_i \mathcal{N}(.; \mathbf{m_X}^{(i)}, \mathbf{C_X}^{(i)})$, the input-output equation is [125]

$$\mathbf{Y} = \mathbf{G}\mathbf{X} + \mathbf{N}_s, \tag{5.66}$$

where the noise $\mathbf{N}_s \sim \mathcal{N}(.; 0, \mathbf{R_n})$ is independent from $\mathbf{x}$. Here $\mathbf{G} \in R^{M \times N}$, so the observation $\mathbf{y}$ is the noisy linear combination of the target $\mathbf{x}$. Then it is obvious that $(\mathbf{X}, \mathbf{Y})$ follows the joint distribution (5.28) with

$$\begin{aligned}
\mathbf{m}_{\mathbf{X},\mathbf{Y}}^{(i)} &= \begin{pmatrix} \mathbf{m}_{\mathbf{X}}^{(i)} \\ \mathbf{m}_{\mathbf{Y}}^{(i)} \end{pmatrix} = \begin{pmatrix} \mathbf{m}_{\mathbf{X}}^{(i)} \\ \mathbf{G}\mathbf{m}_{\mathbf{X}}^{(i)} \end{pmatrix}, \\
\mathbf{C}^{(i)} &= \begin{pmatrix} \mathbf{C}_{\mathbf{X}}^{(i)} & \mathbf{C}_{\mathbf{XY}}^{(i)} \\ \mathbf{C}_{\mathbf{YX}}^{(i)} & \mathbf{C}_{\mathbf{Y}}^{(i)} \end{pmatrix}, \\
\mathbf{C}_{\mathbf{XY}}^{(i)} &= (\mathbf{C}_{\mathbf{X}}^{(i)} + \mathbf{m}_{\mathbf{X}}^{(i)}(\mathbf{m}_{\mathbf{X}}^{(i)})^T - \mathbf{m}_{\mathbf{X}}(\mathbf{m}_{\mathbf{X}})^T)\mathbf{G}^T, \\
\mathbf{C}_{\mathbf{YX}}^{(i)} &= (\mathbf{C}_{\mathbf{XY}}^{(i)})^T, \\
\mathbf{C}_{\mathbf{Y}}^{(i)} &= \mathbf{G}(\mathbf{C}_{\mathbf{X}}^{(i)} + \mathbf{m}_{\mathbf{X}}^{(i)}(\mathbf{m}_{\mathbf{X}}^{(i)})^T - \mathbf{m}_{\mathbf{X}}(\mathbf{m}_{\mathbf{X}})^T)\mathbf{G}^T + \mathbf{R}_{\mathbf{n}}, \\
\mathbf{C}_{\mathbf{Y}} &= \mathbf{G}\mathbf{C}_{\mathbf{X}}\mathbf{G}^T + \mathbf{R}_{\mathbf{n}}, \mathbf{C}_{\mathbf{YX}} = \mathbf{G}\mathbf{C}_{\mathbf{X}}, \\
&\text{and} \quad \mathbf{C}_{\mathbf{XY}} = \mathbf{C}_{\mathbf{X}}\mathbf{G}^T.
\end{aligned} \tag{5.67}$$

We first consider a static target $\mathbf{X}$ in a two-dimensional field where the target is positioned at location $(\mathbf{X}_1, \mathbf{X}_2)m$. Specifically $\mathbf{X}$ has the following prior proba-

bility distribution

$$\begin{aligned}\mathbf{X} \;\sim\; & \frac{1}{3}\mathcal{N}(.;(0,0)^T,\mathbf{I}_2)+\frac{1}{3}\mathcal{N}(.;(5,5)^T,\mathbf{I}_2)\\ & +\frac{1}{3}\mathcal{N}(.;(-5-5),\mathbf{I}_2),\end{aligned} \tag{5.68}$$

i.e. the target can be either located at $(0,0)m$, $(5,5)m$ or $(-5,-5)m$ and uncertainty in its position is characterized by a variance of $1m$. The sensor measuring parameter $\mathbf{G}$ in (5.66) is determined by linearizing the following nonlinear ranging and bearing function at the target mean $\mathbf{m_X}$

$$g_j(\mathbf{X})=\left(\sqrt{(s_{j,x}-\mathbf{X}_1)^2+(s_{j,y}-\mathbf{X}_2)^2},\frac{s_{j,y}-\mathbf{X}_2}{s_{j,x}-\mathbf{X}_1}\right)^T \tag{5.69}$$

with sensor position $(s_{j,x},s_{j,y})^T$, with $j=1,\ldots,M$.

We let the relay and FC be positioned at $(100,0)m$ and $(200,0)m$ but $M=10$ sensors be positioned randomly surrounding the mean $\mathbf{m_X}$ of the target. The channel gains h_{jR} and h_{jD} are determined according to the free-space path gain [126] $h=G_tG_r(\lambda/4\pi d)^2$, with the distance between two ends d, signal wavelength λ and antenna gains $G_t=2$dB, $G_r=5$dB. The covariance matrices are defined as $\mathbf{R_n}=\mathbf{R_{w_R}}=\mathbf{R_{w_D}}=0.5\mathbf{I}$ and the sensor transmit power budget varies as $P_T=0.1\ell$, $\ell=1,2,...,10$ but the relay power budget is fixed at $P_R=5$. The simulation is validated via $N_{mc}=10000$ Monte Carlo channel realizations. To show the viability of our proposed suboptimal solution by Algorithm 1, in Fig. 5.2 the normalized mean squared error (NMSE) is benchmarked with the proposed suboptimal power allocation (2H-Algorithm 1), only sensor power allocation in one-hop (1H) communication between the sensors and FC (1H-Algorithm 2), which is based on Algorithm 2) given in Appendix B, and equal power allocation schemes for one-hop (1H) and two-hop (2H) communication environments (1H and 2H-equal power) Overall, 2H-Algorithm 1 provides the lowest NMSE for all power budgets P_T. The average iterations for 2H Algorithm 1 and 1H Algorithm 2 under error tolerance $\epsilon=10^{-3}$ in the stopping condition are shown in Table 5.1.

Additionally, Figs 5.3 demonstrates the value of the surrogate function $\varphi(\alpha^{(\kappa)},\beta^{(\kappa)})$ in (5.46), and the Fig. 5.4 shows the corresponding MSE (5.41). Of course, from

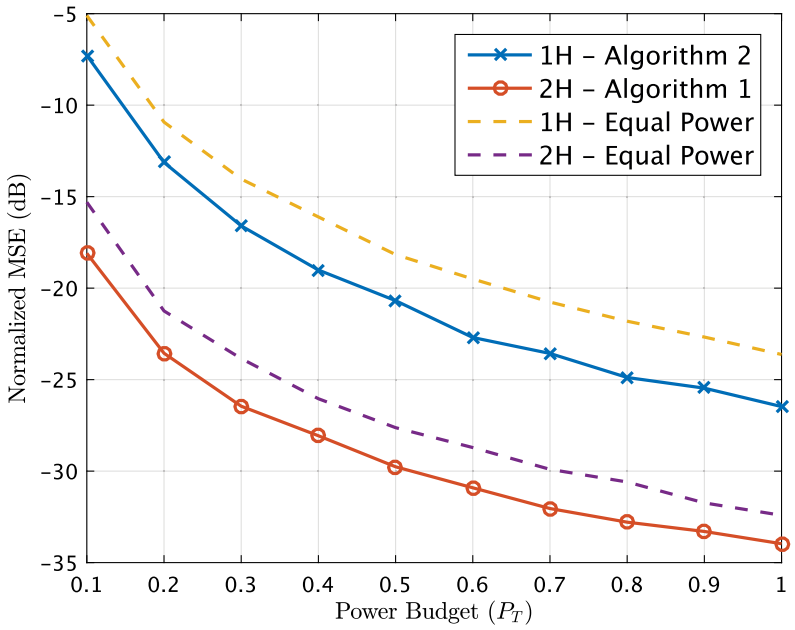


Figure 5.2 – Normalized MSE of LSN by different power schemes

Table 5.1 – Average iterations of two algorithms for LSN.

	Power Budget P_T									
Algorithm	*0.1*	*0.2*	*0.3*	*0.4*	*0.5*	*0.6*	*0.7*	*0.8*	*0.9*	*1*
1H	4.06	4.20	4.16	4.03	3.86	3.63	3.38	3.19	3.09	3.02
2H	5.18	4.40	4.02	3.83	3.63	3.51	3.46	3.40	3.36	3.36

(5.44), the improvements of the former and the latter are not necessarily parallel and that's why the optimal solution of the former is only a suboptimal solution of the latter.

5.5.2 Nonlinear sensor networks

Rather than using (5.66), the input-output equation of a NSN is given as

$$\mathbf{Y} = g(\mathbf{X}) + \mathbf{N}_s. \tag{5.70}$$

Then, $(\mathbf{X}, \mathbf{Y})$ approximately follows the joint GMM (5.28) with $\mathbf{m}_{\mathbf{Y}}^{(i)}$, $\mathbf{C}_{\mathbf{XY}}^{(i)}$ and $\mathbf{C}_{\mathbf{Y}}^{(i)}$ calculated through the unscented transformation [195] as follows. For each

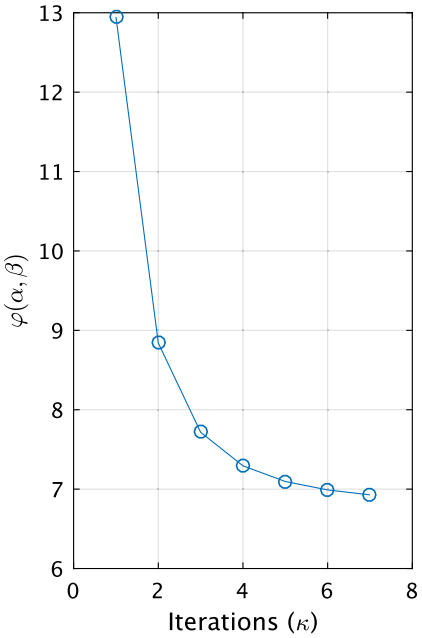


Figure 5.3 – The function $\varphi(\alpha^{(\kappa)}, \beta^{(\kappa)})$ for power allocation at each iteration κ.

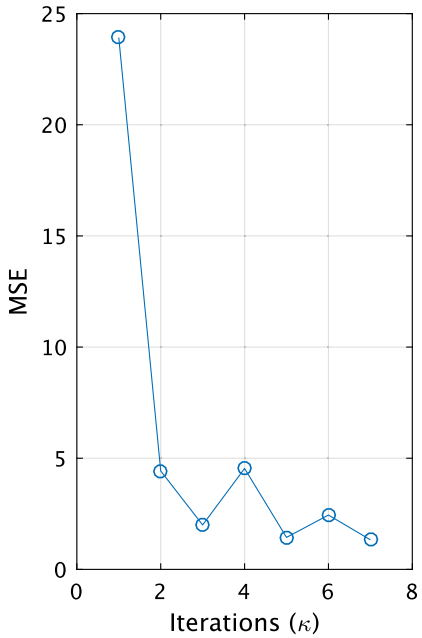


Figure 5.4 – The MSE calculated for each estimated target $\tilde{\mathbf{X}}^{(\kappa)}$ at iteration κ.

i, take the Cholesky decomposition[2]

$$\mathbf{C}_{\mathbf{X}}^{(i)} = \sum_{r=1}^{N} \tilde{\mathbf{x}}^{(r)} (\tilde{\mathbf{x}}^{(r)})^T. \tag{5.71}$$

Accordingly, $2N+1$ regression points $\mathbf{x}^{(r)}, r = 0, 1, \ldots, 2N$ are defined by

$$\begin{aligned} \mathbf{x}^{(0)} &= \mathbf{m}_{\mathbf{X}}^{(i)}, \\ \mathbf{x}^{(r)} &= \mathbf{m}_{\mathbf{X}}{}^{(i)} + \sqrt{\frac{2N+1}{2}} \tilde{\mathbf{x}}^{(r)}, \\ \mathbf{x}^{(N+r)} &= \mathbf{m}_{\mathbf{X}}^{(i)} - \sqrt{\frac{2N+1}{2}} \tilde{\mathbf{x}}^{(r)}, r = 1, 2, ..., N. \end{aligned} \tag{5.72}$$

Clearly,

$$\begin{aligned} \mathbf{m}_{\mathbf{X}}{}^{(i)} &= \frac{1}{2N+1} \sum_{r=0}^{2N} \mathbf{x}^{(r)}, \\ \mathbf{C}_{\mathbf{x}}^{(i)} &= \frac{1}{2N+1} \sum_{r=0}^{2N} (\mathbf{x}^{(r)} - \mathbf{m}_{\mathbf{X}}{}^{(i)})(\mathbf{x}^{(r)} - \mathbf{m}_{\mathbf{X}}{}^{(i)})^T, \end{aligned}$$

and thereby transform $\mathbf{y}^{(r)} := g(\mathbf{x}^{(r)}),\ r = 0, 1, ..., 2N$ for approximations

$$\begin{aligned} \mathbf{m}_{\mathbf{Y}}^{(i)} &= \frac{1}{2N+1} \sum_{r=0}^{2N} \mathbf{y}^{(r)} \\ \mathbf{C}_{\mathbf{Y}}^{(i)} &= \frac{1}{2N+1} \sum_{r=0}^{2N} (\mathbf{y}^{(r)} - \mathbf{m}_{\mathbf{Y}}^{(i)})(\mathbf{y}^{(r)} - \mathbf{m}_{\mathbf{Y}}^{(i)})^T \\ \mathbf{C}_{\mathbf{XY}}^{(i)} &= \frac{1}{2N+1} \sum_{r=0}^{2N} (\mathbf{x}^{(r)} - \mathbf{m}_{\mathbf{X}}{}^{(i)})(\mathbf{y}^{(r)} - \mathbf{m}_{\mathbf{Y}}^{(i)})^T. \end{aligned} \tag{5.73}$$

We use nonlinear ranging and bearing functions (5.69) with sensor position $(s_{j,x}, s_{j,y})^T$ for $g_j(\mathbf{x}) = (g_1(\mathbf{x}), g_2(\mathbf{x}))^T$ in (5.70), while the target is prior characterized by (5.68). The simulation environment is the same as that in the previous LSN simulation. Table 5.2 shows the average iterations for 2H Algorithm 1 and 1H Algorithm 2.

---

[2]For the SVD (singular value decomposition): $\mathbf{C}_{\mathbf{X}}^{(i)} = \sum_{r=1}^{N} \lambda_r x^{(r)} (x^{(r)})^T$, it is obvious that $\tilde{\mathbf{x}}^{(r)} = \sqrt{\lambda_r} x^{(r)}$. For notational simplicity we omit the index i in x^r, i.e. rigorously speaking, it should be $x^{(i,r)}$.

Table 5.2 – Average iterations of two algorithms in NSN.

	Power Budget P_T									
Algorithm	*0.1*	*0.2*	*0.3*	*0.4*	*0.5*	*0.6*	*0.7*	*0.8*	*0.9*	*1*
1H	4.49	5.56	6.16	6.42	6.71	6.86	7.11	7.17	7.21	7.22
2H	5.83	6.56	6.77	6.86	6.89	6.88	6.90	6.88	6.87	6.77

Fig. 5.5 shows the NMSE curves. Like Fig. 5.2, 2H provides a notable difference in dB, especially in lower fixed power budgets.

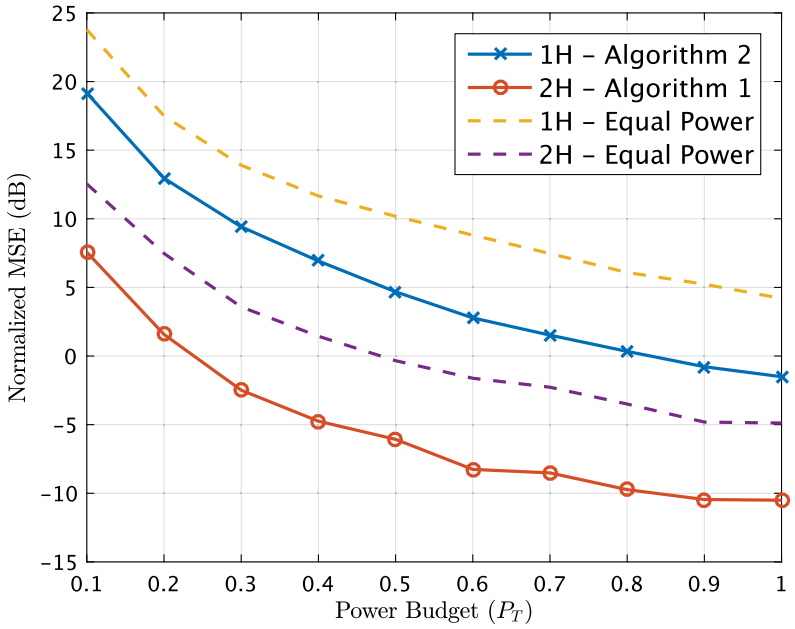


Figure 5.5 – Normalized MSE Performance of NSN by different power schemes.

5.6 Dynamic target tracking by WSN

In this section, we consider the tracking of a dynamic target, which is moving in a surveillance region. The sensor nodes are distributed geographically to take independent measurements of a target's position and send these measurements to the FC via the relay node. These relayed observations are processed at each time step to update and predict the target state. The following set of equations

model the considered scenario:

$$\begin{aligned}
\mathbf{X}_{k+1} &= \mathbf{F}_k\mathbf{X}_k + \mathbf{V}_k, && (5.74)\\
\mathbf{Y}_k &= g_k(\mathbf{X}_k) + \mathbf{N}_k, && (5.75)\\
\mathbf{Z}_{k,R} &= \mathsf{diag}\left[\sqrt{\alpha_k}\sqrt{h_{jR}}\right]_{j=1,2...,M}\mathbf{Y}_k + \mathbf{W}_{k,R}, && (5.76)\\
\mathbf{Z}_k &= \mathbf{H}_{k,(\alpha,\beta)}\mathbf{Y}_k + \mathbf{W}_{k,(\alpha,\beta)}, && (5.77)
\end{aligned}$$

At time step $k = 0, 1, ...,$ (5.74) represents a linear Gaussian dynamical model of a target with the state transition matrix $\mathbf{F}_k$ and (5.75) represents the sensor measurement, where $\mathbf{V}_k \sim \mathcal{N}(.; 0, \mathbf{R}_v)$ is process noise and $\mathbf{N}_k \sim \mathcal{N}(.; 0, \mathbf{R}_n)$ with diagonal $\mathbf{R}_n$, is the sensor measurement noise. Similarly to (5.30), equation (5.76) represents the signal received by the relay at time k, where $\sqrt{h_{jR}}$ is the channel gain between sensor j and the relay, $\mathbf{W}_{k,R} \sim \mathcal{N}(.; 0, \mathbf{R}_R)$ with $\mathbf{R}_R = \mathsf{diag}[\sigma_{jR}]_1^M$ is a corrupt noise at relay, and $\sqrt{\alpha_j}$ controls the transmitter power $P_j = \alpha_j||y_{k,j}||^2 = (\mathbf{C}_{\mathbf{Y}_k}(j,j) + \mathbf{m}^2_{\mathbf{Y}_k}(j))\alpha_j$ of sensor j to satisfy the fixed sum power budget $P_T > 0$, which is defined similarly to (5.31) as

$$\sum_{j=1}^{M} P_j = \sum_{j=1}^{M} ||y_{k,j}||^2\alpha_j \leq P_T. \quad (5.78)$$

Similar to (5.34), equation (5.77) with

$$\mathbf{H}_{k,(\alpha,\beta)} = \mathsf{diag}[\sqrt{h_{jD}h_{jR}\beta_j\alpha_j/(h_{jR}||y_{k,j}||^2\alpha_j + \sigma_{jR})}]_1^M$$

represents the signal received at the FC at time k, where $\sqrt{h_{jD}}$ is the channel gain between the relay and the FC. Note that

$$\mathbf{W}_{k,(\alpha,\beta)} = \mathsf{diag}[\sqrt{h_{jD}\beta_j/(h_{jR}||y_{k,j}||^2\alpha_j + \sigma_{jR})}]_1^M\mathbf{W}_R + \mathbf{W}_D$$

is the total noise, where $\mathbf{W}_D \sim \mathcal{N}(.; 0, \mathsf{diag}[\sigma_{jD}]_1^M)$ is the background noise, which is independent with $\mathbf{Z}_R$. Accordingly, $\mathbf{W}_{k,(\alpha,\beta)} \sim \mathcal{N}(.; 0, \mathbf{C}_{k,(\alpha,\beta)})$ with

$$\mathbf{C}_{k,(\alpha,\beta)} = \mathsf{diag}[\sigma_{jR}h_{jD}\beta_j/(h_{jR}||y_{k,j}||^2\alpha_j + \sigma_{jR}) + \sigma_{jD}]_1^M.$$

The power levels β_j are constrained by the relay power budget P_R in (5.33).

Given the initial information $\mathbf{X}_{0|-1} \sim p_{\mathbf{X}_{0|-1}}(\mathbf{x}) = \sum_{i=1}^{L}\lambda_i(-1)\mathcal{N}(\mathbf{x}, \mathbf{m}^{(i)}_{\mathbf{X}_{0|-1}}, \mathbf{C}^{(i)}_{\mathbf{X}_{0|-1}})$, the FC iterates at time $k = 0, 1, ...,$ as followings.

- *Power constrained filtering.* Execute the unscented transformation in Subsection III.B with input $\{\lambda_i(k-1), \mathbf{m}^{(i)}_{\mathbf{X}_{k|k-1}}, \mathbf{C}^{(i)}_{\mathbf{X}_{k|k-1}}\}_{i=1}^{L}$, $\mathbf{R}_n$ and $g = g_k$ to write the approximate joint GM distribution

$$\begin{aligned}(\mathbf{X}_{k|k-1}, \mathbf{Y}_k) \;\sim\; & \sum_{i=1}^{L} \lambda_i(k-1)\mathcal{N}((.,.); \\ & (\mathbf{m}^{(i)}_{\mathbf{X}_{k|k-1}}, \mathbf{m}^{(i)}_{\mathbf{Y}_k}), \mathbf{C}^{(i)}_{k,R}), \end{aligned} \tag{5.79}$$

where $\mathbf{C}^{(i)}_{k,R}$ is in sub-block form

$$\mathbf{C}^{(i)}_{k,R} = \begin{pmatrix} \mathbf{C}^{(i)}_{\mathbf{X}_{k|k-1}} & \mathbf{C}^{(i)}_{\mathbf{X}_{k|k-1}\mathbf{Y}_k} \\ \mathbf{C}^{(i)}_{\mathbf{Y}_k\mathbf{X}_{k|k-1}} & \mathbf{C}^{(i)}_{\mathbf{Y}_k} \end{pmatrix}.$$

Execute Algorithm 1 to output the suboptimal power allocation (α^*_k, β^*_k) and then update $\mathbf{X}_{k|k} = \mathbf{X}_{k|k-1}|\mathbf{Z}_k = \mathbf{z}_k$ as

$$\begin{aligned}\lambda_i(k) \;=\; & \lambda_i(k-1) \times \\ & \mathcal{N}(\mathbf{z}_k; \mathbf{H}_{\alpha^*_k,\beta^*_k}\mathbf{m}^{(i)}_{\mathbf{Y}_k}, \mathbf{H}_{\alpha^*_k,\beta^*_k}\mathbf{C}^{(i)}_{\mathbf{Y}_k}\mathbf{H}_{\alpha^*_k,\beta^*_k} + \mathbf{C}_{\alpha^*_k,\beta^*_k}) \\ & / \sum_{i=1}^{L} \lambda_i(k-1)\mathcal{N}(\mathbf{z}_k; \mathbf{H}_{\alpha^*_k,\beta^*_k}\mathbf{m}^{(i)}_{\mathbf{Y}_k}, \\ & \mathbf{H}_{\alpha^*_k,\beta^*_k}\mathbf{C}^{(i)}_{\mathbf{Y}_k}\mathbf{H}_{\alpha^*_k,\beta^*_k} + \mathbf{C}_{\alpha^*_k,\beta^*_k}), \end{aligned} \tag{5.80}$$

$$\begin{aligned}\mathbf{m}^{(i)}_{\mathbf{X}_{k|k}} \;=\; & \mathbf{m}^{(i)}_{\mathbf{X}_{k|k-1}} + \mathbf{C}^{(i)T}_{\mathbf{Y}_k\mathbf{X}_{k|k-1}}\mathbf{H}_{\alpha^*_k,\beta^*_k} \times \\ & (\mathbf{H}_{\alpha^*_k,\beta^*_k}\mathbf{C}^{(i)}_{\mathbf{Y}_k}\mathbf{H}_{\alpha^*_k,\beta^*_k} + \mathbf{C}_{\alpha^*_k,\beta^*_k})^{-1} \times \\ & (\mathbf{z}_k - \mathbf{H}_{\alpha^*_k,\beta^*_k}\mathbf{m}^{(i)}_{\mathbf{Y}_k}), i = 1, 2, ..., L, \end{aligned} \tag{5.81}$$

$$\begin{aligned}\mathbf{C}^{(i)}_{\mathbf{X}_{k|k}} \;=\; & \mathbf{C}^{(i)}_{\mathbf{X}_{k|k-1}} - \mathbf{C}^{(i)}_{\mathbf{X}_{k|k-1}\mathbf{Y}_k}\mathbf{H}_{\alpha^*_k,\beta^*_k} \times \\ & (\mathbf{H}_{\alpha^*_k,\beta^*_k}\mathbf{C}^{(i)}_{\mathbf{Y}_k}\mathbf{H}_{\alpha^*_k,\beta^*_k} + \mathbf{C}_{\alpha^*_k,\beta^*_k})^{-1} \times \\ & \mathbf{H}_{\alpha^*_k,\beta^*_k}(\mathbf{C}^{(i)}_{\mathbf{X}_{k|k-1}\mathbf{Y}_k})^T, i = 1, 2, ..., L, \end{aligned} \tag{5.82}$$

$$\mathbf{X}_{k|k} \;\sim\; \sum_{i=1}^{L} \lambda_i(k)\mathcal{N}(.; \mathbf{m}^{(i)}_{\mathbf{X}_{k|k}}, \mathbf{C}^{(i)}_{\mathbf{X}_{k|k}}), \tag{5.83}$$

$$\mathbf{x}_{k|k} \;=\; \sum_{i=1}^{L} \lambda_i(k)\mathbf{m}^{(i)}_{\mathbf{X}_{k|k}}. \tag{5.84}$$

- *State distribution prediction.* Write the joint GM distribution

$$(\mathbf{X}_{k|k}, \mathbf{X}_{k+1|k}) \sim \sum_{i=1}^{L} \lambda_i(k)\mathcal{N}\left((.,.); \begin{pmatrix} \mathbf{m}^{(i)}_{\mathbf{X}_{k|k}} \\ \mathbf{m}^{(i)}_{\mathbf{X}_{k+1|k}} \end{pmatrix}, \begin{pmatrix} \mathbf{C}^{(i)}_{\mathbf{X}_{k|k}} & \mathbf{C}^{(i)}_{\mathbf{X}_{k|k}\mathbf{X}_{k+1|k}} \\ \mathbf{C}^{(i)}_{\mathbf{X}_{k+1|k}\mathbf{X}_{k|k}} & \mathbf{C}^{(i)}_{\mathbf{X}_{k+1|k}} \end{pmatrix}\right), \tag{5.85}$$

with

$$\begin{aligned} \mathbf{m}^{(i)}_{\mathbf{X}_{k+1|k}} &= \mathbf{F}_k \mathbf{m}^{(i)}_{\mathbf{X}_{k|k}}, \\ \mathbf{C}^{(i)}_{\mathbf{X}_{k+1|k}\mathbf{X}_{k|k}} &= (\mathbf{C}^{(i)}_{\mathbf{X}_{k|k}\mathbf{X}_{k+1|k}})^T, \\ \mathbf{C}^{(i)}_{\mathbf{X}_{k|k}\mathbf{X}_{k+1|k}} &= (\mathbf{C}^{(i)}_{\mathbf{X}_{k|k}} + \mathbf{m}^{(i)}_{\mathbf{X}_{k|k}}(\mathbf{m}^{(i)}_{\mathbf{X}_{k|k}})^T \\ &\quad -\mathbf{m}_{\mathbf{X}_{k|k}}(\mathbf{m}_{\mathbf{X}_{k|k}})^T)\mathbf{F}_k^T, \\ \mathbf{C}^{(i)}_{\mathbf{X}_{k+1|k}} &= \mathbf{F}_k(\mathbf{C}^{(i)}_{\mathbf{X}_{k|k}} + \mathbf{m}^{(i)}_{\mathbf{X}_{k|k}}(\mathbf{m}^{(i)}_{\mathbf{X}_{k|k}})^T \\ &\quad -\mathbf{m}_{\mathbf{X}_{k|k}}(\mathbf{m}_{\mathbf{X}|_{k|k}})^T)\mathbf{F}_k^T + \mathbf{R}_{\mathbf{v}} \end{aligned}$$

to update

$$\mathbf{X}_{k+1|k} \sim \sum_{i=1}^{L} \lambda_i(k)\mathcal{N}(.; \mathbf{m}^{(i)}_{\mathbf{X}_{k+1|k}}, \mathbf{C}^{(i)}_{\mathbf{X}_{k+1|k}}). \tag{5.86}$$

Thus, the track of $\mathbf{X}_k$ is $\mathbf{x}_{k|k}$ defined by equation (5.84), while the track of $\mathbf{X}_k$ by LMMSE estimate is defined by (5.15) (in Appendix B) with $\mathbf{X} \to \mathbf{X}_{k|k}$ and $\mathbf{Y} \to \mathbf{Z}_k$.

5.6.1 Linear Sensor Networks

Consider a scenario with a 2D dynamic target moving in the surveillance region $[-80, 40] \times [0, 500]m^2$, where the relay is at $[100, 0]m$, FC is at $[200, 0]m$ and $M = 10$ sensors are randomly distributed within a region $[-100, 100] \times [-100, 100]m^2$. The state $\mathbf{X}_k = (p_{xk}, p_{yk}, \dot{p}_{xk}, \dot{p}_{yk})^T$ of the target consists of position (p_{xk}, p_{yk}) and velocity $(\dot{p}_{xk}, \dot{p}_{xk})$, while the measurement is a noise corrupted version of the position. The target's dynamics follows the linear Gaussian dynamical model

(5.74) with

$$F_k = \begin{pmatrix} I_2 & TI_2 \\ 0_2 & I_2 \end{pmatrix},$$

and

$$\mathbf{R_v} = \begin{pmatrix} \frac{T^4}{4} I_2 & \frac{T^3}{2} I_2 \\ \frac{T^3}{2} I_2 & T^2 I_2 \end{pmatrix},$$

where $T = 1s$ is the sampling period. All the sensors are linear, so (5.75) is $g_k(\mathbf{X}_k) = \begin{pmatrix} I_2 & 0_{2\times 2} \end{pmatrix} \mathbf{X}_k$. We assume that the initial state of the target is

$$\begin{aligned} \mathbf{X}_{0|-1} \sim & \frac{1}{3}\mathcal{N}(x; \mathbf{m}^{(1)}_{\mathbf{X}_{0|-1}}, \mathbf{C}^{(1)}_{\mathbf{X}_{0|-1}}) \\ & + \frac{1}{3}\mathcal{N}(x; \mathbf{m}^{(2)}_{\mathbf{X}_{0|-1}}, \mathbf{C}^{(2)}_{\mathbf{X}_{0|-1}}) \\ & + \frac{1}{3}\mathcal{N}(x; \mathbf{m}^{(3)}_{\mathbf{X}_{0|-1}}, \mathbf{C}_{\mathbf{x_0}}{}^{(3)}) \end{aligned}$$

where

$$\begin{aligned} &\mathbf{m}^{(1)}_{\mathbf{X}_{0|-1}} = [0,0,0,0]^T, \mathbf{m}^{(2)}_{\mathbf{X}_{0|-1}} = [50,50,0,0]^T, \\ &\mathbf{m}^{(3)}_{\mathbf{X}_{0|-1}} = [-50,-50,0,0]^T, \mathbf{C}^{(1)}_{\mathbf{X}_{0|-1}} = \mathsf{diag}([2,2,5,5]^T), \\ &\mathbf{C}^{(2)}_{\mathbf{X}_{0|-1}} = \mathsf{diag}([3,4,5,6]^T), \mathbf{C}^{(3)}_{\mathbf{X}_{0|-1}} = \mathsf{diag}([5,6,7,8]^T). \end{aligned}$$

Fig. 5.6 shows the motion of the target. Overall, all the algorithms 1 and 2 help track the true path considerably well because the blue line (true track) is hidden by the estimated tracks. This implies the algorithms can accurately recognize the true track, however at larger distances from the Relay (or FC), 2H algorithm 1 is more robust in tracking the path of the target, followed by 2H using equal power allocation (2H-EP), 1H and 1H with equal power allocation (1H-EP).

The MSE can be seen for the x-coordinate and y-coordinate in Figs. 5.7 and 5.8, respectively. In the majority of time steps, the 2H Algorithm 1 has less MSE in comparison to the other methods in both x and y-coordinates. Furthermore, it can be seen that 1H outperforms 1H-EP in the majority of time steps in the y-coordinate and about half the time-steps in the x-coordinate. Thus it can be concluded that 2H completely outperforms 1H for this example.

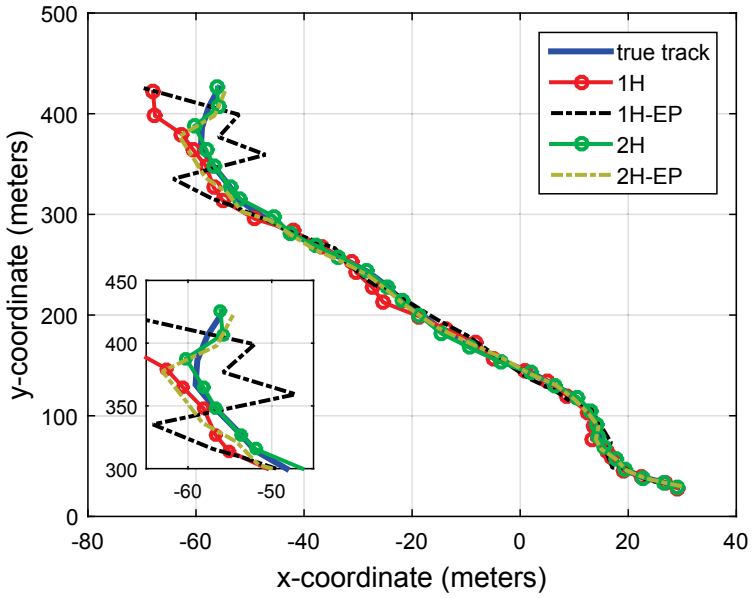


Figure 5.6 – Path of a constant velocity target and the estimated tracks.

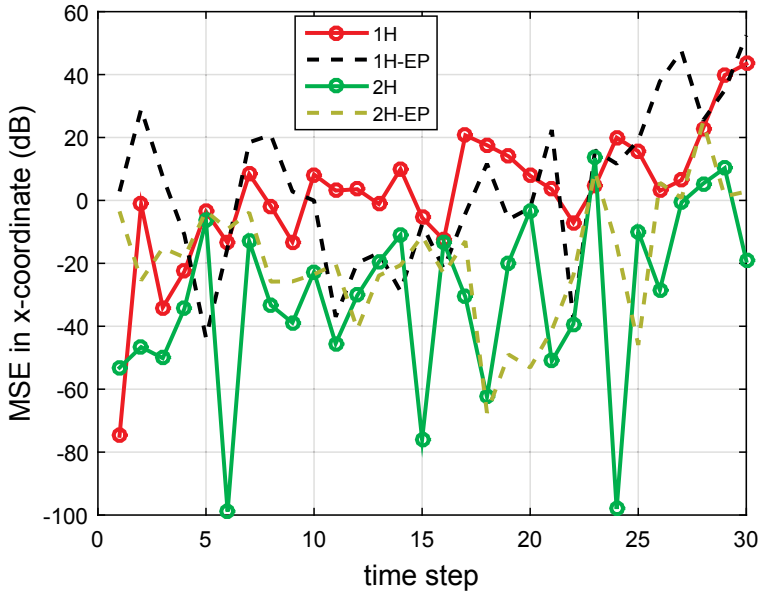


Figure 5.7 – Comparison of MSE performance for the x-coordinate.

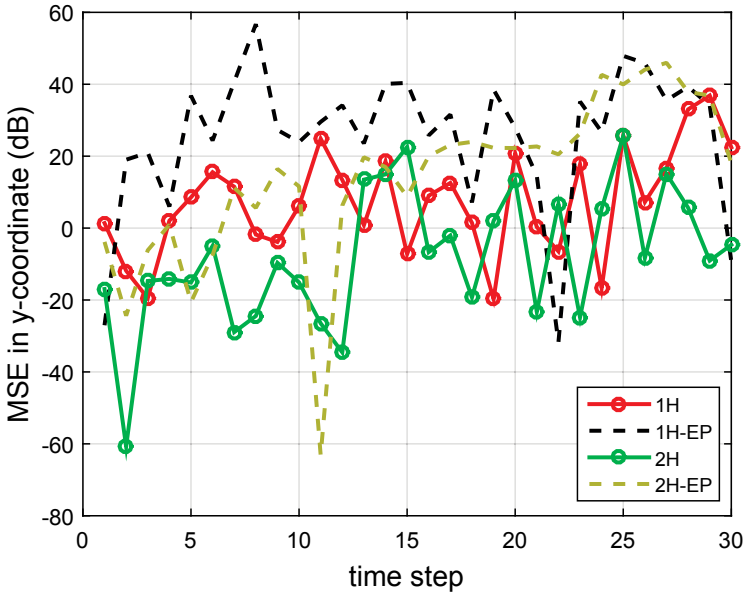


Figure 5.8 – Comparison of MSE performance for the y-coordinate.

5.6.2 Nonlinear Sensor Networks

In this example, we study the tracking performance of a maneuvering target in a region $[0, 600] \times [-50, 300] m^2$. The sensors are distributed geographically over a surveillance region $[-150, 150] \times [-150, 150] m^2$ to take maximum advantage of estimation diversity. The relay and FC are positioned at $[200, 0] m$ and $[400, 0] m$, respectively. A coordinated turn model (see e.g. [196]) characterizes the dynamics of the target. The target kinematic state $\mathbf{X}_k = (p_{x_k}, \dot{p}_{x_k}, p_{y_k}, \dot{p}_{y_k})^T$ consists of the target position (p_x, p_y) and its velocity $(\dot{p}_x, \dot{p}_y)$. The state dynamical model of the target is assumed to be linear Gaussian, which is mathematically expressed by (5.74) with

$$\mathbf{F}_k = \begin{pmatrix} 1 & \frac{\sin \omega T}{\omega} & 0 & -\frac{1-\cos \omega T}{\omega} \\ 0 & \cos \omega T & 0 & -\sin \omega T \\ 0 & \frac{1-\cos \omega T}{\omega} & 1 & \frac{\sin \omega T}{\omega} \\ 0 & \sin \omega T & 0 & \cos \omega T \end{pmatrix},$$

$$\mathbf{R_v} = \begin{pmatrix} \frac{T^4}{4} & \frac{T^3}{2} & 0 & 0 \\ \frac{T^3}{2} & T^2 & 0 & 0 \\ 0 & 0 & \frac{T^4}{4} & \frac{T^3}{2} \\ 0 & 0 & \frac{T^3}{2} & T^2 \end{pmatrix},$$

where T is the sampling period and ω is the turn rate of the maneuvering target. The sensor nonlinear measurements include range and bearing information of the vehicle, which is represented by (5.75) with

$$g_k(\mathbf{X}) = \left(\begin{pmatrix} \sqrt{(s_{1,x} - \mathbf{X}_k(1))^2 + (s_{1,y} - \mathbf{X}_k(3))^2} \\ \frac{s_{1,y} - \mathbf{X}_k(3)}{s_{1,x} - \mathbf{X}_k(1)} \end{pmatrix}, ..., \begin{pmatrix} \sqrt{(s_{M,x} - \mathbf{X}_k(1))^2 + (s_{M,y} - \mathbf{X}_k(3))^2} \\ \frac{s_{M,y} - \mathbf{X}_k(3)}{s_{M,x} - \mathbf{X}_k(1)} \end{pmatrix} \right),$$

where $(s_{j,x}, s_{j,y})$ is the position of sensor j. The initial state of target is assumed that

$$\begin{aligned} \mathbf{X}_{\mathbf{X}_{0|-1}} \sim\ & 0.5\mathcal{N}(x; \mathbf{m}^{(1)}_{\mathbf{X}_{0|-1}}, \mathbf{C}^{(1)}_{\mathbf{X}_{0|-1}}) \\ & +0.5\mathcal{N}(x; \mathbf{m}^{(2)}_{\mathbf{X}_{0|-1}}, \mathbf{C}^{(2)}_{\mathbf{X}_{0|-1}}) \end{aligned}$$

where

$$\begin{aligned} \mathbf{m}^{(1)}_{\mathbf{X}_{0|-1}} &= [50, 5, 10, 5]^T, \\ \mathbf{m}^{(2)}_{\mathbf{X}_{0|-1}} &= [90, 5, 10, 5]^T, \\ \mathbf{C}^{(i)}_{\mathbf{X}_{0|-1}} &\equiv \mathsf{diag}([3, 3, 2, 2]^T). \end{aligned}$$

Fig. 5.9 shows the two-dimensional motion of a maneuvering target. At $k = 1s$, the target begins its motions from a position $[94, 10]m$ at a constant velocity of $5m/s$ and after $14s$ performs a counterclockwise turn for $11s$ at a turn rate of $\omega = 0.2rad/s$. It then takes a clockwise turn after $25s$ with a turn rate of $\omega = -0.1rad/s$ until it reaches its final position at $[550, 175]m$. The 2H algorithm estimates the true path of the target very accurately in comparison to the other algorithms. It can be seen that 1H and 1H-EP perform poorly since at several time steps their estimated paths diverge from the original path of the target.

A comparison of MSE for both x and y-coordinates can be seen in Figs. 5.10 and 5.11. Both 1H and 1H-EP have a significantly higher error overall in comparison to the 2H algorithms. The 2H algorithm has quite lower MSE in comparison to 2H-EP and the difference in MSE increases as the target moves further away from the sensors.

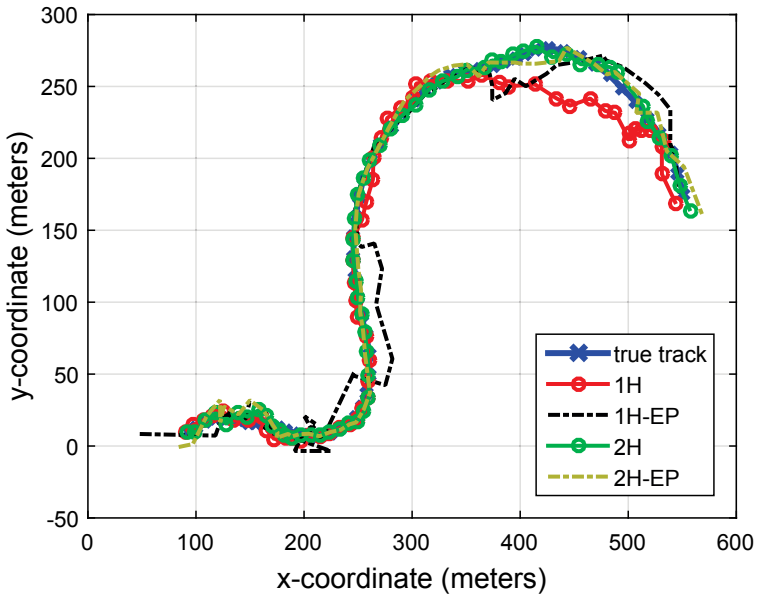


Figure 5.9 – Path of a maneuvering target and the estimated tracks.

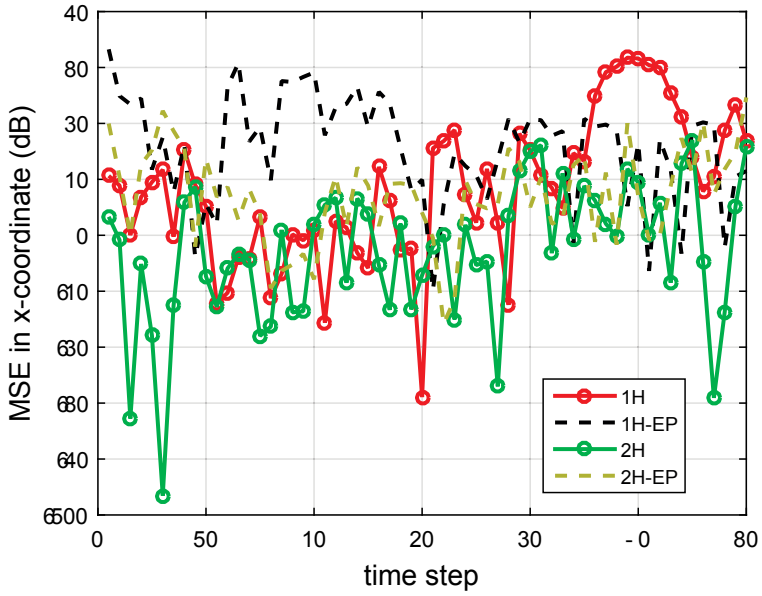


Figure 5.10 – Comparison of MSE performance for the x-coordinate.

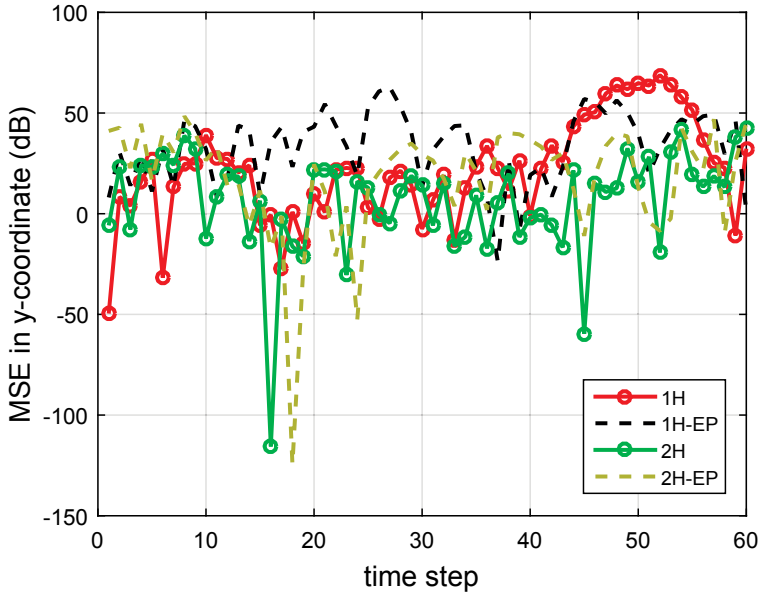


Figure 5.11 – Comparison of MSE performance for the y-coordinate.

5.6.3 LSN for nonlinear dynamics

The unscented transformation in subsection III.B can be applied to a target with nonlinear dynamics to have an approximated joint GM distribution (5.85) for state prediction, which however may not lead to tracking a target's true path. Consider a typical third-order nonlinear autoregressive model described mathematically by [189, 125] as

$$\mathbf{q}_{k+2} = -0.1\mathbf{q}_{k+1} - \mathbf{q}_k^3 + \mathbf{v}_k$$

with the noise corrupted observations $\mathbf{y}_k^{(i)} = g^{(i)}\mathbf{q}_k + \mathbf{n}_k^{(i)}$, where $\mathbf{v}_k \sim \mathcal{N}(.,0,0.04)$, $\mathbf{n}_k^{(i)} \sim \mathcal{N}(.,0,0.1)$ and $g^{(i)} = 1 + 0.11(\ell - 1)$, $\ell = 1, 2, ..., 10$.
By choosing the state $\mathbf{X}_k = (\mathbf{q}_k, \mathbf{q}_{k+1})^T \in R^2$, the state dynamic and measurement equations are

$$\mathbf{X}_{k+1} = f(\mathbf{X}_k) + \mathbf{V}, \mathbf{Y}_k = \mathbf{G}\mathbf{X}_k + \mathbf{N}_s \tag{5.87}$$

where

$$f(\mathbf{X}_k) = \begin{pmatrix} \mathbf{X}_k(2) \\ -\mathbf{X}_k(1)^3 - 0.1\mathbf{X}_k(2) \end{pmatrix}, \mathbf{V} = \begin{pmatrix} 0 \\ 1 \end{pmatrix} \mathbf{v}_k,$$

$$\mathbf{G} = \begin{pmatrix} g^{(1)} & 0 \\ g^{(2)} & 0 \\ ... & ... \\ g^{(M)} & 0 \end{pmatrix}, \mathbf{n} = (\mathbf{n}^{(1)}, ..., \mathbf{n}^{(M)})^T.$$

The initial state of target is given by

$$\mathbf{X}_{0|-1} \sim 0.5\mathcal{N}(x; (0.1, 0.1)^T, \mathbf{C}^{(1)}_{\mathbf{X}_{0|-1}}) + 0.5\mathcal{N}(x; (-0.1, -0.1)^T, \mathbf{C}^{(2)}_{\mathbf{X}_{0|-1}}),$$

where $\mathbf{C}^{(i)}_{\mathbf{X}_{0|-1}} = \begin{pmatrix} 1 & \rho^{(i)} \\ \rho^{(i)} & 1 \end{pmatrix}, \rho^{(1)} = 0.75, \rho^{(2)} = 0.8$. In this example, using the unscented transformation for updating the joint GM distribution of $\mathbf{X}_{k|k}$ and $\mathbf{X}_{k+1|k}$ for state prediction will not track the target. Following [189], we represent

$$\mathbf{X}_{k+1} = (F + B(I_2 - \Delta(\mathbf{X}_k)D)^{-1}\Delta(\mathbf{X}_k)C)\mathbf{X}_k$$

for

$$F = \begin{pmatrix} 0 & 1 \\ 0 & -0.1 \end{pmatrix}, B = \begin{pmatrix} 0 & 0 \\ 0 & -1 \end{pmatrix}, D = \begin{pmatrix} 0 & 0 \\ 1 & 0 \end{pmatrix},$$

$$C = \begin{pmatrix} 1 & 0 \\ 0 & 0 \end{pmatrix}, \Delta(\mathbf{x}_k) = \mathbf{x}_k(1)I_2$$

and use the following procedure for updating $\mathbf{C}^{(i)}_{\mathbf{X}_{k+1|k}}$ and $\mathbf{m}^{(i)}_{\mathbf{X}_{k+1|k}}$ from $\mathbf{C}^{(i)}_{\mathbf{X}_{k|k}}$ and $\mathbf{m}^{(i)}_{\mathbf{X}_{k|k}}$.

- Take the Cholesky decomposition $\mathbf{C}^{(i)}_{\mathbf{X}_{k|k}} = \tilde{x}^{(1)}(\tilde{x}^{(1)})^T + \tilde{x}^{(2)}(\tilde{x}^{(2)})^T$ and set $x^{(0)} = \mathbf{m}^{(i)}_{\mathbf{X}_{k|k}}$, $x^{(r)} = x^{(0)} + \sqrt{5/2}\tilde{x}^{(r)}$ and $x^{(r+2)} = x^{(0)} - \sqrt{5/2}\tilde{x}^{(r)}$, $r = 1, 2$.

- Set $\bar{w}_\Delta = \frac{1}{5}(I_2 - \Delta(x^{(0)})D)^{-1}\sum_{r=0}^{4}\Delta(x^{(r)})Cx^{(r)}$ and then $w_{\Delta r} = \Delta(x^{(r)})(Cx^{(r)} + D\bar{w}_\Delta)$, $r = 0, 1, .., 5$.

- Set $R_\Delta = \frac{1}{5}\sum_{r=0}^{4}(w_{\Delta r} - \bar{w}_\Delta)(w_{\Delta r} - \bar{w}_\Delta)^T$ and take

$$\begin{aligned} \mathbf{m}^{(i)}_{\mathbf{X}_{k+1|k}} &= F\mathbf{m}^{(i)}_{\mathbf{X}_{k|k}} + B\bar{w}_\Delta, \\ \mathbf{C}^{(i)}_{\mathbf{X}_{k+1|k}} &= F\mathbf{C}^{(i)}_{\mathbf{X}_{k|k}}F^T + BR_\Delta B^T \\ &\quad + \begin{pmatrix} 0 & 0 \\ 0 & \sigma_{\mathbf{v}_k} \end{pmatrix}. \end{aligned}$$

The trajectory of $\mathbf{X}_k(1)$ for 50 time steps along with the estimated tracks are shown in Fig. 5.12 and the MSE is plotted in Fig. 5.13. The results suggest that the 2H algorithm outperforms the other algorithms for the measured state estimation. Also, the optimised power allocation algorithm offers less MSE compared to the equal power allocation techniques.

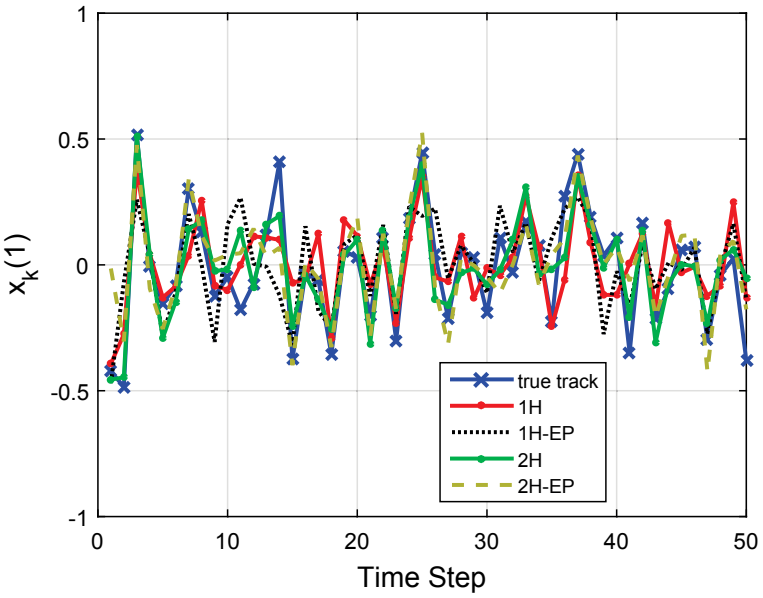


Figure 5.12 – The true and estimated trajectory of the state $\mathbf{x}_k$.

5.7 Conclusion

The chapter addressed the problem of joint sensor and relay power allocation for locating a stationary Gaussian mixture target or for tracking a dynamic Gaussian mixture target by either linear sensor networks or nonlinear sensor networks. We considered scenarios where the sensors noisy observations are transmitted

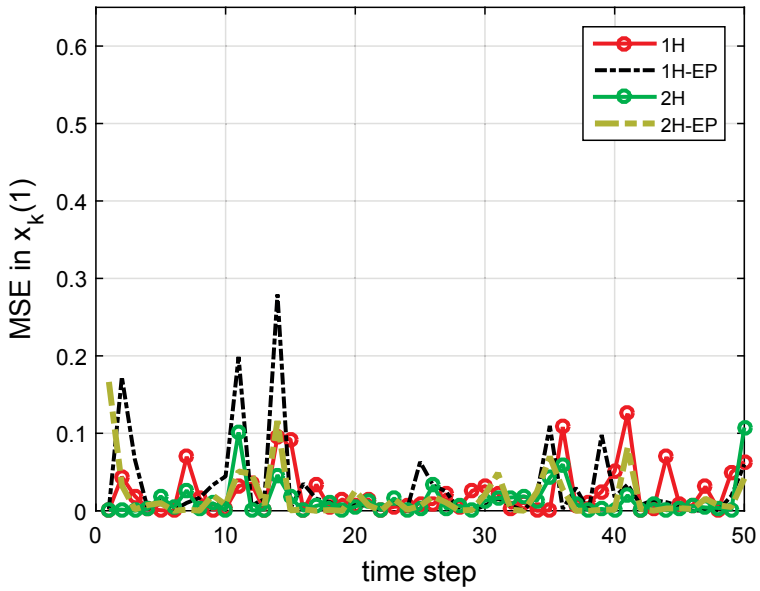


Figure 5.13 – Comparison of MSE at each time step.

to the relay, who amplifies and then forwards them to the FC. To arrive at an accurate estimate of a targets state, a novel technique based on tractable and scalable optimisation was proposed to optimise Bayesian filtering under low sensor transmitter and relay power budgets. Numerical examples have confirmed the merits of our proposed technique.

Chapter 6

Summary and outlook

6.1 Thesis summary

In the first part of this thesis, a historical and pedagogical background of tensor decompositions is provided with an emphasis on MPS, tensor completion and tensor-based machine learning. Additionally, a review of wireless sensor networks is given to highlight another contribution of the thesis. *Chapter 2* provides mathematical notation for *Chapters 3-4*, a preliminary introduction to tensors that includes the definition and notation of the TD and MPS, as well as the important measure of entropy known as the von Neumann entropy.

In *Chapter 3*, a new novel approach to machine learning dimensionality reduction and classification based on the concept of MPS is given. It rigorously shows that MPS is extremely efficient in retaining the relevant features of a tensor as well as being able to project it to a matrix of moderate size. This is theoretically justified using the von Neumann entropy, which shows that MPS is able to capture more correlations because it consists of matrices constructed from a balanced matricization scheme, whereas TD-based methods consists of matrices constructed from an unbalanced matricization scheme. A comprehensive outline of an optimised MPS-approach to tensor compression is given, with extensive discussions on practical computations. The TTPCA and MPS algorithms are proposed to demonstrate two different methods in which to apply the core algorithm for tensor

object classification. The benchmark results demonstrate excellent performance in tensor object recognition against several popular tensor-based methods. The results of this chapter have been published in [C1] and [J3].

Chapter 4 introduces MPS to tensor completion, where it shows that an MPS approach is substantially more efficient for estimating missing entries of data than present state-of-the-art methods. A theoretical analysis via the von Neumann entropy proves that correlations within the modes of a tensor are more efficiently captured using TT rank than Tucker rank, which is advantageous for tensor completion. From this analysis, several new problems based on TT rank for tensor completion are formulated, along with new algorithms SiLRTC-TT and TMac-TT to provide solutions. To supplement the TT rank-based algorithms, a novel tensor augmentation technique known as ket augmentation creates a new tensor structure that maps levels of textures in images and videos within the modes of the tensor. Additionally, the ICTAC framework improves the results of image completion significantly by utilising a new approach to image completion by transforming an image to a video sequence tensor. Therefore, both ICTAC and KA with TMac-TT can be considered state-of-the-art algorithms for colour image and video completion, respectively, which is clearly seen from the impressive results in the experiments. The results of this chapter have been published in [C3] and [J2].

Finally, in *Chapter 5*, a new novel approach to target tracking in wireless sensor networks is proposed. Specifically, the problem of joint sensor and relay power allocation for locating a stationary Gaussian mixture target or for tracking a dynamic Gaussian mixture target by either linear sensor networks or nonlinear sensor networks is considered. This approach has not been considered previously and a rigorous mathematical proof is given to theoretically justify the proposed algorithm. The scenario considers when sensors transmit their noisy observations to a relay, who amplifies and then forwards them to a fusion center. To obtain an accurate estimate of a targets state, a novel technique based on tractable and scalable optimisation is proposed to optimise Bayesian filtering under low sensor

transmitter and relay power budgets. The extensive numerical simulations considered combinations of scenarios that included static and dynamic targets, and linear and nonlinear sensor networks. In all simulations, the proposed algorithm outperformed all other methods. The results of this chapter have been published in [C2] and [J4].

6.2 Future Outlook

The introduction of ket augmentation as a novel tensor augmentation technique supplements TT rank based algorithms. The structure of a tensor that has been applied by KA is fixed in terms of the size of the block structured addressing intended. Investigation into different block sizes, or even optimal block sizes for tensor completion provides potential for future work. Additionally, KA enables TMac-TT and SiLRTC-TT to perform very well for tensor completion problems. Applying KA to other problems where the analysis of correlations between modes of data is important could provide new sights and results. For example, in lossy compression, where only significant features are retained in order to compress the size of an image or video.

In quantum physics, tensor networks [[30]] is another name for using tensor decompositions to analyse quantum mechanical systems. There are decompositions in tensor networks that have not been utilised outside of physics, one is known as project entangled pair states (PEPS), which decomposes a large multidimensional tensor into a two-dimensional grid of component tensors. Future research and investigation into these types of decompositions for applications in computer vision, machine learning and signal processing is an exciting prospect.

Bibliography

[1] H. Lu, K. N. Plataniotis, and A. N. Venetsanopoulos, "A survey of multilinear subspace learning for tensor data," *Pattern Recognition*, vol. 44, no. 7, pp. 1540 – 1551, 2011.

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

[3] L. Grasedyck, D. Kressner, and C. Tobler, "A literature survey of low-rank tensor approximation techniques," *GAMM-Mitteilungen*, vol. 36, no. 1, pp. 53–78, 2013.

[4] L. R. Tucker, "Implications of factor analysis of three-way matrices for measurement of change," in *Problems in measuring change.*, C. W. Harris, Ed. Madison WI: University of Wisconsin Press, 1963, pp. 122–137.

[5] L. Grasedyck, "Hierarchical singular value decomposition of tensors," *SIAM Journal on Matrix Analysis and Applications*, vol. 31, no. 4, pp. 2029–2054, 2010.

[6] W. Hackbusch and S. Kühn, "A new scheme for the tensor representation," *Journal of Fourier Analysis and Applications*, vol. 15, no. 5, pp. 706–722, 2009.

[7] H. A. L. Kiers, "Towards a standardized notation and terminology in multiway analysis," *Journal of Chemometrics*, vol. 14, no. 3, pp. 105–122, 2000.

[8] I. V. Oseledets, "Tensor-train decomposition," *SIAM Journal on Scientific Computing*, vol. 33, no. 5, pp. 2295–2317, 2011.

[9] G. Vidal, "Efficient classical simulation of slightly entangled quantum computations," *Phys. Rev. Lett.*, vol. 91, p. 147902, Oct 2003.

[10] I. Affleck, T. Kennedy, E. H. Lieb, and H. Tasaki, "Rigorous results on valence-bond ground states in antiferromagnets," *Phys. Rev. Lett.*, vol. 59, pp. 799–802, Aug 1987.

[11] D. Pérez-García, F. Verstraete, M. M. Wolf, and J. I. Cirac, "Matrix product state representations," *Quantum Information & Computation*, vol. 7, no. 5, pp. 401–430, 2007.

[12] C. E. Cook, M. T. Bergman, R. D. Finn, G. Cochrane, E. Birney, and R. Apweiler, "The european bioinformatics institute in 2016: Data growth and integration," *Nucleic Acids Research*, vol. 44, no. D1, pp. D20–D26, 2016.

[13] I. A. T. Hashem, I. Yaqoob, N. B. Anuar, S. Mokhtar, A. Gani, and S. U. Khan, "The rise of "big data" on cloud computing: Review and open research issues," *Information Systems*, vol. 47, pp. 98 – 115, 2015.

[14] P. Mehrotra, L. H. Pryor, F. R. Bailey, and M. Cotnoir, "Supporting "big data" analysis and analytics at the nasa advanced supercomputing (nas) facility," NASA Advanced Supercomputing (NAS) Division, Tech. Rep., January 2014.

[15] S. Yan, D. Xu, Q. Yang, L. Zhang, X. Tang, and H. J. Zhang, "Multilinear discriminant analysis for face recognition," *IEEE Transactions on Image Processing*, vol. 16, no. 1, pp. 212–220, Jan 2007.

[16] M. A. O. Vasilescu and D. Terzopoulos, *Computer Vision — ECCV 2002: 7th European Conference on Computer Vision Copenhagen, Denmark, May 28–31, 2002 Proceedings, Part I.* Berlin, Heidelberg: Springer Berlin Heidelberg, 2002, ch. Multilinear Analysis of Image Ensembles: TensorFaces, pp. 447–460.

[17] H. Lu, K. N. Plataniotis, and A. N. Venetsanopoulos, "Mpca: Multilinear principal component analysis of tensor objects," *IEEE Transactions on Neural Networks*, vol. 19, no. 1, pp. 18–39, Jan 2008.

[18] S. Belongie, J. Malik, and J. Puzicha, "Shape matching and object recognition using shape contexts," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 24, no. 4, pp. 509–522, Apr 2002.

[19] D. K. Slonim, "Review: From patterns to pathways: gene expression data analysis comes of age," *Nature Genetics*, vol. 32, pp. 502–508, 2002.

[20] W. Huber, A. von Heydebreck, and M. Vingron, *Handbook of Statistical Genetics*. John Wiley & Sons, Ltd, 2004, ch. Analysis of Microarray Gene Expression Data.

[21] F. Lotte, M. Congedo, A. Lcuyer, F. Lamarche, and B. Arnaldi, "A review of classification algorithms for eeg-based braincomputer interfaces," *Journal of Neural Engineering*, vol. 4, no. 2, p. R1, 2007.

[22] B. Blankertz, K. R. Muller, G. Curio, T. M. Vaughan, G. Schalk, J. R. Wolpaw, A. Schlogl, C. Neuper, G. Pfurtscheller, T. Hinterberger, M. Schroder, and N. Birbaumer, "The bci competition 2003: progress and perspectives in detection and discrimination of eeg single trials," *IEEE Transactions on Biomedical Engineering*, vol. 51, no. 6, pp. 1044–1051, June 2004.

[23] K. N. Plataniotis and A. N. Venetsanopoulos, *Color Image Processing and Applications*. New York, NY, USA: Springer-Verlag New York, Inc., 2000.

[24] L. Shao, L. Liu, and X. Li, "Feature learning for image classification via multiobjective genetic programming," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 25, no. 7, pp. 1359–1371, July 2014.

[25] J. Ellis, H. Davis, and J. Zamudio, "Exploring for onshore oil seeps with hyperspectral imaging," *Oil and Gas Journal*, vol. 99, no. 37, pp. 49–58, 2001.

[26] P. W. Yuen and M. Richardson, "An introduction to hyperspectral imaging and its application for security, surveillance and target acquisition," *The Imaging Science Journal*, vol. 58, no. 5, pp. 241–253, 2010.

[27] T. G. Kolda and J. Sun, "Scalable tensor decompositions for multi-aspect data mining," in *Proceedings of the 2008 Eighth IEEE International Conference on Data Mining*, ser. ICDM '08, Washington, DC, USA, 2008, pp. 363–372.

[28] S. Rendle, L. Balby Marinho, A. Nanopoulos, and L. Schmidt-Thieme, "Learning optimal ranking with tensor factorization for tag recommendation," in *Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, ser. KDD '09. New York, NY, USA: ACM, 2009, pp. 727–736.

[29] F. Verstraete, V. Murg, and J. Cirac, "Matrix product states, projected entangled pair states, and variational renormalization group methods for quantum spin systems," *Advances in Physics*, vol. 57, no. 2, pp. 143–224, 2008.

[30] R. Orús, "A practical introduction to tensor networks: Matrix product states and projected entangled pair states," *Annals of Physics*, vol. 349, pp. 117 – 158, 2014.

[31] F. L. Hitchcock, "The expression of a tensor or a polyadic as a sum of products," *Journal of Mathematics and Physics*, vol. 6, no. 1-4, pp. 164–189, 1927.

[32] ——, "Multiple invariants and generalized rank of a p-way matrix or tensor," *Journal of Mathematics and Physics*, vol. 7, no. 1-4, pp. 39–79, 1928.

[33] R. B. Cattell, ""parallel proportional profiles" and other principles for determining the choice of factors by rotation," *Psychometrika*, vol. 9, no. 4, pp. 267–283, 1944.

[34] ——, "The three basic factor-analytic research designs?their interrelations and derivatives." *Psychological bulletin*, vol. 49, no. 5, p. 499, 1952.

[35] 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.

[36] R. A. Harshman, "Foundations of the PARAFAC procedure: Models and conditions for an" explanatory" multi-modal factor analysis," *UCLA Working Papers in Phonetics*, vol. 16, no. 1, p. 84, 1970.

[37] J. Levin, "Three-Mode Factor Analysis," Ph.D. dissertation, University of Illinois, Urbana, 1963.

[38] L. R. Tucker, "The extension of factor analysis to three-dimensional matrices," in *Contributions to mathematical psychology.*, H. Gulliksen and N. Frederiksen, Eds. New York: Holt, Rinehart and Winston, 1964, pp. 110–127.

[39] ——, "Some mathematical notes on three-mode factor analysis," *Psychometrika*, vol. 31, no. 3, pp. 279–311, 1966.

[40] L. D. Lathauwer, B. D. Moor, and J. Vandewalle, "A multilinear singular value decomposition," *SIAM Journal on Matrix Analysis and Applications*, vol. 21, no. 4, pp. 1253–1278, 2000.

[41] P. M. Kroonenberg and J. de Leeuw, "Principal component analysis of three-mode data by means of alternating least squares algorithms," *Psychometrika*, vol. 45, no. 1, pp. 69–97, 1980.

[42] A. Kapteyn, H. Neudecker, and T. Wansbeek, "An approach ton-mode components analysis," *Psychometrika*, vol. 51, no. 2, pp. 269–275, 1986.

[43] C. J. Appellof and E. R. Davidson, "Strategies for analyzing data from video fluorometric monitoring of liquid chromatographic effluents," *Analytical Chemistry*, vol. 53, no. 13, pp. 2053–2056, 1981.

[44] L. D. Lathauwer and J. Castaing, "Blind identification of underdetermined mixtures by simultaneous matrix diagonalization," *IEEE Transactions on Signal Processing*, vol. 56, no. 3, pp. 1096–1105, March 2008.

[45] N. D. Sidiropoulos, R. Bro, and G. B. Giannakis, "Parallel factor analysis in sensor array processing," *IEEE Transactions on Signal Processing*, vol. 48, no. 8, pp. 2377–2388, Aug 2000.

[46] J. Mocks, "Topographic components model for event-related potentials and some biophysical considerations," *IEEE Transactions on Biomedical Engineering*, vol. 35, no. 6, pp. 482–484, June 1988.

[47] A. H. Andersen and W. S. Rayens, "Structure-seeking multilinear methods for the analysis of fmri data," *NeuroImage*, vol. 22, no. 2, pp. 728 – 739, 2004.

[48] E. Acar, S. A. Çamtepe, M. S. Krishnamoorthy, and B. Yener, *Intelligence and Security Informatics: IEEE International Conference on Intelligence and Security Informatics, ISI 2005, Atlanta, GA, USA, May 19-20, 2005. Proceedings*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2005, ch. Modeling and Multiway Analysis of Chatroom Tensors, pp. 256–268.

[49] M. Mrup, "Applications of tensor (multiway array) factorizations and decompositions in data mining," *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, vol. 1, no. 1, pp. 24–40, 2011.

[50] A. Shashua and A. Levin, "Linear image coding for regression and classification using the tensor-rank principle," in *Computer Vision and Pattern Recognition, 2001. CVPR 2001. Proceedings of the 2001 IEEE Computer Society Conference on*, vol. 1, 2001, pp. I–42–I–49.

[51] M. J. M. Gregory Beylkin, "Numerical operator calculus in higher dimensions," *Proceedings of the National Academy of Sciences of the United States of America*, vol. 99, no. 16, pp. 10 246–10 251, 2002.

[52] H. D. Sterck, "A nonlinear gmres optimization algorithm for canonical tensor decomposition," *SIAM Journal on Scientific Computing*, vol. 34, no. 3, pp. A1351–A1379, 2012.

[53] S. Friedland, V. Mehrmann, R. Pajarola, and S. Suter, "On best rank one approximation of tensors," *Numerical Linear Algebra with Applications*, vol. 20, no. 6, pp. 942–955, 2013.

[54] E. Zander and H. G. Matthies, "Tensor product methods for stochastic problems," *PAMM*, vol. 7, no. 1, pp. 2 040 067–2 040 068, 2007.

[55] R. Henrion, "N-way principal component analysis theory, algorithms and applications," *Chemometrics and Intelligent Laboratory Systems*, vol. 25, no. 1, pp. 1 – 23, 1994.

[56] D. Muti and S. Bourennane, "Multidimensional filtering based on a tensor approach," *Signal Processing*, vol. 85, no. 12, pp. 2338 – 2353, 2005.

[57] M. A. O. Vasilescu and D. Terzopoulos, "Multilinear subspace analysis of image ensembles," in *Computer Vision and Pattern Recognition, 2003. Proceedings. 2003 IEEE Computer Society Conference on*, vol. 2, June 2003, pp. 93–99.

[58] M. A. O. Vasilescu, "Human motion signatures: analysis, synthesis, recognition," in *Pattern Recognition, 2002. Proceedings. 16th International Conference on*, vol. 3, 2002, pp. 456–460.

[59] E. E. Abdallah, A. B. Hamza, and P. Bhattacharya, *Image Analysis and Recognition: 4th International Conference, ICIAR 2007, Montreal, Canada, August 22-24, 2007. Proceedings.* Berlin, Heidelberg: Springer Berlin Heidelberg, 2007, ch. MPEG Video Watermarking Using Tensor Singular Value Decomposition, pp. 772–783.

[60] S. R., S. Sharma, M. Thakur, and P. K. Bora, "Perceptual video hashing based on tucker decomposition with application to indexing and retrieval of near-identical videos," *Multimedia Tools and Applications*, pp. 1–19, 2015.

[61] B. Savas and L. Eldén, "Handwritten digit classification using higher order singular value decomposition," *Pattern Recognition*, vol. 40, no. 3, pp. 993 – 1003, 2007.

[62] J.-T. Sun, H.-J. Zeng, H. Liu, Y. Lu, and Z. Chen, "Cubesvd: A novel approach to personalized web search," in *Proceedings of the 14th International Conference on World Wide Web*, ser. WWW '05, New York, NY, USA, 2005, pp. 382–390.

[63] A. H. Phan and A. Cichocki, "Tensor decompositions for feature extraction and classification of high dimensional datasets," *Nonlinear Theory and Its Applications, IEICE*, vol. 1, no. 1, pp. 37–68, 2010.

[64] B. Romera-paredes, H. Aung, N. Bianchi-berthouze, and M. Pontil, "Multilinear multitask learning," in *Proceedings of the 30th International Conference on Machine Learning (ICML-13)*, S. Dasgupta and D. Mcallester, Eds. JMLR Workshop and Conference Proceedings, May 2013, pp. 1444–1452.

[65] B. Cyganek, *Hybrid Artificial Intelligent Systems: 7th International Conference, HAIS 2012, Salamanca, Spain, March 28-30th, 2012. Proceedings, Part II.* Berlin, Heidelberg: Springer Berlin Heidelberg, 2012, ch. Ensemble of Tensor Classifiers Based on the Higher-Order Singular Value Decomposition, pp. 578–589.

[66] H. A. Kramers and G. H. Wannier, "Statistics of the two-dimensional ferromagnet. part ii," *Phys. Rev.*, vol. 60, pp. 263–276, Aug 1941.

[67] R. J. Baxter, "Dimers on a rectangular lattice," *Journal of Mathematical Physics*, vol. 9, no. 4, pp. 650–654, Apr. 1968.

[68] I. Affleck, T. Kennedy, E. Lieb, and H. Tasaki, "Valence bond ground states in isotropic quantum antiferromagnets," *Communications in Mathematical Physics*, vol. 115, pp. 477–528, 1988.

[69] M. Fannes, B. Nachtergaele, and R. Werner, "Finitely correlated states on quantum spin chains," *Communications in Mathematical Physics*, vol. 144, no. 3, pp. 443–490, Mar. 1992.

[70] A. Klümper, A. Schadschneider, and J. Zittartz, "Matrix product ground states for one-dimensional spin-1 quantum antiferromagnets," *EPL (Europhysics Letters)*, vol. 24, no. 4, p. 293, 1993.

[71] A. K. Kolezhuk, H.-J. Mikeska, and S. Yamamoto, "Matrix-product-states approach to heisenberg ferrimagnetic spin chains," *Phys. Rev. B*, vol. 55, pp. R3336–R3339, Feb 1997.

[72] G. Vidal, "Efficient simulation of one-dimensional quantum many-body systems," *Phys. Rev. Lett.*, vol. 93, p. 040502, Jul 2004.

[73] ——, "Classical simulation of infinite-size quantum lattice systems in one spatial dimension," *Phys. Rev. Lett.*, vol. 98, p. 070201, Feb 2007.

[74] R. Orús and G. Vidal, "Infinite time-evolving block decimation algorithm beyond unitary evolution," *Phys. Rev. B*, vol. 78, p. 155117, Oct 2008.

[75] M. C. Bañuls, M. B. Hastings, F. Verstraete, and J. I. Cirac, "Matrix product states for dynamical simulation of infinite chains," *Phys. Rev. Lett.*, vol. 102, p. 240603, Jun 2009.

[76] F. Verstraete and J. I. Cirac, "Continuous matrix product states for quantum fields," *Phys. Rev. Lett.*, vol. 104, p. 190405, May 2010.

[77] H. N. Phien, G. Vidal, and I. P. McCulloch, "Infinite boundary conditions for matrix product state calculations," *Phys. Rev. B*, vol. 86, p. 245107, Dec 2012.

[78] ——, "Dynamical windows for real-time evolution with matrix product states," *Phys. Rev. B*, vol. 88, p. 035103, Jul 2013.

[79] I. V. Oseledets and S. V. Dolgov, "Solution of linear systems and matrix inversion in the tt-format," *SIAM Journal on Scientific Computing*, vol. 34, no. 5, pp. A2718–A2739, 2012.

[80] C. Lubich, I. V. Oseledets, and B. Vandereycken, "Time integration of tensor trains," *SIAM Journal on Numerical Analysis*, vol. 53, no. 2, pp. 917–941, 2015.

[81] D. Savostyanov and I. Oseledets, "Fast adaptive interpolation of multi-dimensional arrays in tensor train format," in *Multidimensional (nD) Systems (nDs), 2011 7th International Workshop on*, Sept 2011, pp. 1–8.

[82] L. Grasedyck, M. Kluge, and S. Kramer, "Variants of alternating least squares tensor completion in the tensor train format," *SIAM J. Sci. Comput.*, vol. 37, no. 5, pp. A2424–A2450, Jan 2015.

[83] K. Kormann, "A semi-lagrangian vlasov solver in tensor train format," *SIAM Journal on Scientific Computing*, vol. 37, no. 4, pp. B613–B632, 2015.

[84] S. Holtz, T. Rohwedder, and R. Schneider, "The alternating linear scheme for tensor optimization in the tensor train format," *SIAM Journal on Scientific Computing*, vol. 34, no. 2, pp. A683–A713, 2012.

[85] M. Bachmayr and W. Dahmen, "Adaptive near-optimal rank tensor approximation for high-dimensional operator equations," *Foundations of Computational Mathematics*, vol. 15, no. 4, pp. 839–898, 2015.

[86] J. Liu, P. Musialski, P. Wonka, and J. Ye, "Tensor completion for estimating missing values in visual data," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 35, no. 1, pp. 208–220, Jan 2013.

[87] J. Sun, L. Yuan, J. Jia, and H.-Y. Shum, "Image completion with structure propagation," *ACM Trans. Graph.*, vol. 24, no. 3, pp. 861–868, Jul. 2005.

[88] M. Fazel, H. Hindi, and S. P. Boyd, "A rank minimization heuristic with application to minimum order system approximation," in *American Control Conference, 2001. Proceedings of the 2001*, vol. 6, 2001, pp. 4734–4739.

[89] E. J. Candes and T. Tao, "The power of convex relaxation: Near-optimal matrix completion," *IEEE Transactions on Information Theory*, vol. 56, no. 5, pp. 2053–2080, May 2010.

[90] E. J. Candès and B. Recht, "Exact matrix completion via convex optimization," *Foundations of Computational Mathematics*, vol. 9, no. 6, pp. 717–772, 2009.

[91] B. Recht, M. Fazel, and P. A. Parrilo, "Guaranteed minimum-rank solutions of linear matrix equations via nuclear norm minimization," *SIAM Review*, vol. 52, no. 3, pp. 471–501, 2010.

[92] J.-F. Cai, E. J. Cands, and Z. Shen, "A singular value thresholding algorithm for matrix completion," *SIAM Journal on Optimization*, vol. 20, no. 4, pp. 1956–1982, 2010.

[93] C. J. Hillar and L.-H. Lim, "Most tensor problems are np-hard," *J. ACM*, vol. 60, no. 6, pp. 1–39, Nov. 2013.

[94] S. Gandy, B. Recht, and I. Yamada, "Tensor completion and low-n-rank tensor recovery via convex optimization," *Inverse Problems*, vol. 27, no. 2, p. 025010, 2011.

[95] Y. Xu, R. Hao, W. Yin, and Z. Su, "Parallel matrix factorization for low-rank tensor completion," *Inverse Problems and Imaging*, vol. 9, no. 2, pp. 601–624, 2015.

[96] M. Filipović and A. Jukić, "Tucker factorization with missing data with application to low-n-rank tensor completion," *Multidimensional Systems and Signal Processing*, vol. 26, no. 3, pp. 677–692, 2015.

[97] Q. Zhao, L. Zhang, and A. Cichocki, "Bayesian cp factorization of incomplete tensors with automatic rank determination," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 37, no. 9, pp. 1751–1763, Sept 2015.

[98] S. Pouryazdian, S. Beheshti, and S. Krishnan, "Candecomp/parafac model order selection based on reconstruction error in the presence of kronecker structured colored noise," *Digital Signal Processing*, vol. 48, pp. 12 – 26, 2016.

[99] A. K. Jain, R. P. W. Duin, and J. Mao, "Statistical pattern recognition: a review," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 22, no. 1, pp. 4–37, Jan 2000.

[100] J. Zhang, S. Z. Li, and J. Wang, *Manifold Learning and Applications in Recognition*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2005, pp. 281–300.

[101] M. H. C. Law and A. K. Jain, "Incremental nonlinear dimensionality reduction by manifold learning," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 28, no. 3, pp. 377–391, March 2006.

[102] S. Yan, D. Xu, Q. Yang, L. Zhang, X. Tang, and H.-J. Zhang, "Discriminant analysis with tensor representation," in *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*, vol. 1, June 2005, pp. 526–532 vol. 1.

[103] L. D. Lathauwer, B. D. Moor, and J. Vandewalle, "On the best rank-1 and rank-(r1 ,r2 ,. . .,rn) approximation of higher-order tensors," *SIAM Journal on Matrix Analysis and Applications*, vol. 21, no. 4, pp. 1324–1342, 2000.

[104] D. Xu, S. Yan, D. Tao, L. Zhang, X. Li, and H.-J. Zhang, "Human gait recognition with matrix representation," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 16, no. 7, pp. 896–903, July 2006.

[105] H. Lu, K. N. Plataniotis, and A. N. Venetsanopoulos, "Uncorrelated multilinear discriminant analysis with regularization and aggregation for tensor object recognition," *IEEE Transactions on Neural Networks*, vol. 20, no. 1, pp. 103–123, Jan 2009.

[106] D. Tao, X. Li, X. Wu, and S. J. Maybank, "General tensor discriminant analysis and gabor features for gait recognition," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 29, no. 10, pp. 1700–1715, Oct 2007.

[107] D. Tao, M. Song, X. Li, J. Shen, J. Sun, X. Wu, C. Faloutsos, and S. J. Maybank, "Bayesian tensor approach for 3-d face modeling," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 18, no. 10, pp. 1397–1410, Oct 2008.

[108] D. Xu, S. Yan, L. Zhang, S. Lin, H. J. Zhang, and T. S. Huang, "Reconstruction and recognition of tensor-based objects with concurrent subspaces analysis," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 18, no. 1, pp. 36–47, Jan 2008.

[109] J. Sun, D. Tao, S. Papadimitriou, P. S. Yu, and C. Faloutsos, "Incremental tensor analysis: Theory and applications," *ACM Trans. Knowl. Discov. Data*, vol. 2, no. 3, pp. 1–37, Oct. 2008.

[110] Y. Panagakis, C. Kotropoulos, and G. R. Arce, "Non-negative multilinear principal component analysis of auditory temporal modulations for music genre classification," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 18, no. 3, pp. 576–588, March 2010.

[111] M. A. Nielsen and I. L. Chuang, *Quantum computation and quantum information*. Cambridge, England: Cambridge University Press, 2000.

[112] I. F. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci, "Wireless sensor networks: A survey," *Comput. Netw.*, vol. 38, no. 4, pp. 393–422, Mar. 2002.

[113] D. Nguyen and M. J. Bagajewicz, "Design of nonlinear sensor networks for process plants," *Industrial & Engineering Chemistry Research*, vol. 47, no. 15, pp. 5529–5542, 2008.

[114] S. Kim, B. Ku, W. Hong, and H. Ko, "Performance comparison of target localization for active sonar systems," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 44, no. 4, pp. 1371–1380, Oct 2008.

[115] K. C. Ho and L. M. Vicente, "Sensor allocation for source localization with decoupled range and bearing estimation," *IEEE Transactions on Signal Processing*, vol. 56, no. 12, pp. 5773–5789, Dec 2008.

[116] X. Zhang, H. V. Poor, and M. Chiang, "Optimal power allocation for distributed detection over MIMO channnels in wireless sensor networks," *IEEE Trans. Signal Process.*, vol. 56, no. 9, pp. 4124–4141, 2008.

[117] M. Arik and O. Akan, "Collaborative mobile target imaging in UWB wireless radar sensor networks," *IEEE J. Sel. Areas Commun.*, vol. 28, no. 6, pp. 950–961, 2010.

[118] L. Liu, X. Zhang, and H. Ma, "Optimal node selection for target localization in wireless camera sensor networks," *IEEE Transactions on Vehicular Technology*, vol. 59, no. 7, pp. 3562–3576, Sept 2010.

[119] M. Gastpar and M. Vetterli, *Source-Channel Communication in Sensor Networks*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2003, pp. 162–177.

[120] H. V. Poor, *An Introduction to Signal Detection and Estimation (second edition)*. New York: Springer-Verlag, 1994.

[121] S. Cui, J. J. Xiao, A. J. Goldsmith, Z. Q. Luo, and H. V. Poor, "Estimation diversity and energy efficiency in distributed sensing," *IEEE Transactions on Signal Processing*, vol. 55, no. 9, pp. 4683–4695, Sept 2007.

[122] G. Thatte and U. Mitra, "Sensor selection and power allocation for distributed estimation in sensor networks: Beyond the star topology," *IEEE Transactions on Signal Processing*, vol. 56, no. 7, pp. 2649–2661, July 2008.

[123] J. Fang and H. Li, "Power constrained distributed estimation with correlated sensor data," *IEEE Trans. Signal Process.*, vol. 57, no. 8, pp. 3292 –3297, 2009.

[124] M. H. Chaudhary and L. Vandendorpe, "Power constrained linear estimation in wireless sensor networks with correlated data and digital modulation," *IEEE Trans. Signal Process.*, vol. 60, no. 2, pp. 570–584, 2012.

[125] U. Rashid, H. D. Tuan, P. Apkarian, and H. H. Kha, "Globally optimized power allocation in multiple sensor fusion for linear and nonlinear networks," *IEEE Trans. Signal Process.*, vol. 60, no. 2, pp. 903 –915, 2012.

[126] A. Goldsmith, *Wireless Communications*. New York, NY, USA: Cambridge University Press, 2005.

[127] R. Pabst, B. H. Walke, D. C. Schultz, P. Herhold, H. Yanikomeroglu, S. Mukherjee, H. Viswanathan, M. Lott, W. Zirwas, M. Dohler, H. Aghvami, D. D. Falconer, and G. P. Fettweis, "Relay-based deployment concepts for wireless and mobile broadband radio," *IEEE Communications Magazine*, vol. 42, no. 9, pp. 80–89, Sept 2004.

[128] G. McLachlan and K. Basford, *Mixture Models: Inference and Applications to Clustering*. Marcel Dekker, New York, 1988.

[129] D. Persson, T. Eriksson, and P. Hedelin, "Packet video error concealment with gaussian mixture models," *IEEE Transactions on Image Processing*, vol. 17, no. 2, pp. 145–154, Feb 2008.

[130] G. Yu, G. Sapiro, and S. Mallat, "Solving inverse problems with piecewise linear estimators: From Gaussian mixture models to structured sparsity," *IEEE Trans. Image Process.*, vol. 21, no. 5, pp. 2481–2499, 2012.

[131] J. T. Flam, D. Zachariah, M. Vehkaper, and S. Chatterjee, "The linear model under mixed gaussian inputs: Designing the transfer matrix," *IEEE Transactions on Signal Processing*, vol. 61, no. 21, pp. 5247–5259, Nov 2013.

[132] G. Yu and G. Sapiro, "Statistical compressed sensing of gaussian mixture models," *IEEE Transactions on Signal Processing*, vol. 59, no. 12, pp. 5842–5858, Dec 2011.

[133] J. Yang, X. Liao, X. Yuan, P. Llull, D. J. Brady, G. Sapiro, and L. Carin, "Compressive sensing by learning a gaussian mixture model from measurements," *IEEE Transactions on Image Processing*, vol. 24, no. 1, pp. 106–119, Jan 2015.

[134] M. N. Narasimhan, *Principles of continuum mechanics*. Wiley-Interscience, 1993.

[135] J. A. Schouten, *Ricci-calculus: an introduction to tensor analysis and its geometrical applications*. Springer Science & Business Media, 2013, vol. 10.

[136] U. Schollwöck, "The density-matrix renormalization group in the age of matrix product states," *Annals of Physics*, vol. 326, no. 1, pp. 96 – 192, 2011.

[137] L. Kuang, F. Hao, L. Yang, M. Lin, C. Luo, and G. Min, "A tensor-based approach for big data representation and dimensionality reduction," *IEEE Trans. Emerging Topics in Computing*, vol. 2, no. 3, pp. 280–291, Sept 2014.

[138] A. Cichocki, D. Mandic, L. De Lathauwer, G. Zhou, Q. Zhao, C. Caiafa, and A. H. Phan, "Tensor decompositions for signal processing applications: From two-way to multiway component analysis," *IEEE Signal Processing Magazine*, vol. 32, no. 2, pp. 145–163, March 2015.

[139] L. D. Lathauwer and J. Vandewalle, "Dimensionality reduction in higher-order signal processing and rank-(R1,R2,...,RN) reduction in multilinear algebra," *Linear Algebra and its Applications*, vol. 391, no. 0, pp. 31 – 55, 2004.

[140] F. Verstraete, D. Porras, and J. I. Cirac, "Density matrix renormalization group and periodic boundary conditions: A quantum information

perspective," *Phys. Rev. Lett.*, vol. 93, p. 227205, Nov 2004. [Online]. Available: http://link.aps.org/doi/10.1103/PhysRevLett.93.227205

[141] M. Ishteva, P.-A. Absil, S. van Huffel, and L. de Lathauwer, "Best low multilinear rank approximation of higher-order tensors, based on the Riemannian trust-region scheme." *SIAM J. Matrix Anal. Appl.*, vol. 32, no. 1, pp. 115–135, 2011.

[142] J. A. Bengua, H. N. Phien, H. D. Tuan, and M. N. Do, "Efficient tensor completion for color image and video recovery: Low-rank tensor train," *arXiv preprint arXiv:1606.01500*, 2016.

[143] S. A. Nene, S. K. Nayar, and H. Murase, "Columbia object image library (coil-100)," *Technical Report CUCS-005-96*, Feb 1996.

[144] M. Pontil and A. Verri, "Support vector machines for 3d object recognition," *IEEE Trans. Patt. Anal. and Mach. Intell.*, vol. 20, no. 6, pp. 637–646, Jun 1998.

[145] A. Georghiades, P. Belhumeur, and D. Kriegman, "From few to many: illumination cone models for face recognition under variable lighting and pose," *IEEE Trans. Pattern Anal. and Mach. Intell.*, vol. 23, no. 6, pp. 643–660, Jun 2001.

[146] (2013) Data set for single trial 64-channels EEG classification in BCI. [Online]. Available: http://bcmi.sjtu.edu.cn/resource.html

[147] S. Sarkar, P. J. Phillips, Z. Liu, I. R. Vega, P. Grother, and K. W. Bowyer, "The humanid gait challenge problem: data sets, performance, and analysis," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 27, no. 2, pp. 162–177, Feb 2005.

[148] Q. Li and D. Schonfeld, "Multilinear discriminant analysis for higher-order tensor data classification," *IEEE Trans. Patt. Anal. and Mach. Intell.*, vol. 36, no. 12, pp. 2524–2537, Dec 2014.

[149] Y. Wang, S. Gao, and X. Gao, "Common spatial pattern method for channel selelction in motor imagery based brain-computer interface," in *2005 IEEE Engineering in Medicine and Biology 27th Annual Conference*, Jan 2005, pp. 5392–5395.

[150] Q. Zhao and L. Zhang, "Temporal and spatial features of single-trial eeg for brain-computer interface," *Computational Intelligence and Neuroscience*, vol. 2007, pp. 1–14, Jun 2007.

[151] A. H. Phan, "NFEA: Tensor toolbox for feature extraction and application," Lab for Advanced Brain Signal Processing, BSI, RIKEN, Tech. Rep., 2011.

[152] S. Ma, D. Goldfarb, and L. Chen, "Fixed point and Bregman iterative methods for matrix rank minimization," *Math. Programm.*, vol. 128, no. 1-2, pp. 321–353, Sep 2009.

[153] M. Signoretto, L. De Lathauwer, and J. A. K. Suykens†, "Nuclear norms for tensors and their use for convex multilinear estimation," ESAT-SISTA, K. U. Leuven (Leuven, Belgium), Tech. Rep., 2010.

[154] M. Signoretto, R. Van de Plas, B. De Moor, and J. Suykens, "Tensor versus matrix completion: A comparison with application to spectral data," *IEEE Signal Process. Lett.*, vol. 18, no. 7, pp. 403–406, Jul 2011.

[155] T. Ryota, S. Taiji, K. Hayashi, and H. Kashima, "Statistical performance of convex tensor decomposition," in *Proc. Adv. Neural Inf. Process. Syst.*, Dec 2011, pp. 972–980.

[156] H. Tan, B. Cheng, W. Wang, Y.-J. Zhang, and B. Ran, "Tensor completion via a multi-linear low-n-rank factorization model," *Neurocomputing*, vol. 133, pp. 161–169, Jun 2014.

[157] M. Bertalmio, G. Sapiro, V. Caselles, and C. Ballester, "Image inpainting," in *Proc. ACM Conf. Comp. Graph. (SIGGRAPH ', 00)*, 2000, pp. 417–424.

[158] N. Komodakis, "Image completion using global optimization," in *Proc. IEEE Comput. Vis. Pattern Recognit.*, vol. 1, 2006, pp. 442–452.

[159] T. Korah and C. Rasmussen, "Spatiotemporal inpainting for recovering texture maps of occluded building facades," *IEEE Trans. Image Processing*, vol. 16, no. 9, pp. 2262–2271, Sep 2007.

[160] F. R. Bach, "Consistency of trace norm minimization," *J. Mach. Learn. Res.*, vol. 9, pp. 1019–1048, Jun 2008.

[161] I. V. Oseledets, E. E. Tyrtyshnikov, and N. L. Zamarashkin, "Tensor-train ranks of matrices and their inverses," *Comput. Meth. Appl. Math*, vol. 11, no. 3, pp. 394–403, 2011.

[162] E. Corona, A. Rahimian, and D. Zorin, "A tensor-train accelerated solver for integral equations in complex geometries," *arXiv*, vol. abs/quant-ph/1511.06029, 2015.

[163] T. Mach, "Computing inner eigenvalues of matrices in tensor train matrix format," in *Proc. 9th European Conference on Numerical Mathematics and Advanced Applications, Leicester*, 2011, pp. 781–788.

[164] C. D. Silva and F. J. Herrmann, "Optimization on the hierarchical tucker manifold – applications to tensor completion," *Linear Algebra Appl.*, vol. 481, pp. 131–173, Sep 2015.

[165] H. Rauhut, R. Schneider, and Z. Stojanac, "Tensor completion in hierarchical tensor representations," in *Compressed Sensing Appl.* Springer, 2015, pp. 419–450.

[166] M. Fazel, "Matrix rank minimization with applications," Ph.D. dissertation, PhD thesis, Stanford University, 2002.

[167] M. Kurucz, A. A. Benczur, and K. Csalogany, "Methods for large scale SVD with missing values," in *Proc. 13th ACM SIGKDD Int. Conf. Knowl. Discov. Data Mining*, 2007.

[168] C. Mu, B. Huang, J. Wright, and D. Goldfarb, "Square deal: Lower bounds and improved relaxations for tensor recovery," in *Proc. 31th Int'l Conf. Machine Learning, ICML 2014*, vol. 32, 2014, pp. 73–81.

[169] Z. Wen, W. Yin, and Y. Zhang, "Solving a low-rank factorization model for matrix completion by a nonlinear successive over-relaxation algorithm," *Math. Programm. Comput.*, vol. 4, no. 4, pp. 333–361, Jul 2012.

[170] J. I. Latorre, "Image compression and entanglement," *arxiv*, vol. abs/quant-ph/0510031, 2005.

[171] Y. L. Chen, C. T. Hsu, and H. Y. M. Liao, "Simultaneous tensor decomposition and completion using factor priors," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 36, no. 3, pp. 577–591, March 2014.

[172] Z. Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli, "Image quality assessment: from error visibility to structural similarity," *IEEE Transactions on Image Processing*, vol. 13, no. 4, pp. 600–612, April 2004.

[173] Y. Luo and R. K. Ward, "Removing the blocking artifacts of block-based dct compressed images," *IEEE Transactions on Image Processing*, vol. 12, no. 7, pp. 838–842, July 2003.

[174] T. Yokota, Q. Zhao, and A. Cichocki, "Smooth parafac decomposition for tensor completion," *arXiv:1505.06611*, 2016.

[175] D. Nguyen and M. Bagajewicz, "Design of nonlinear sensor networks for process plants," *Industrial & Engineering Chemistry Research*, vol. 47, no. 15, pp. 5529–5542, 2008.

[176] J. Majchrzak, M. Michalski, and G. Wiczynski, "Distance estimation with a long-range ultrasonic sensor system," *IEEE Sens. J.*, vol. 9, no. 7, pp. 767–773, 2009.

[177] S. Kar and J. Moura, "A mixed time-scale algorithm for distributed parameter estimation: Nonlinear observation models and imperfect communication," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process. (ICASSP) 2009*, 2009, pp. 3669–3672.

[178] L. Liu, X. Zhang, and H. Ma, "Percolation theory-based explosure-apth prevention for wireless sensor networks coverage in internet of things," *IEEE Sensor J.*, vol. 13, no. 10, pp. 3625–3636, 2013.

[179] Y. Zhou, C. Huang, T. Jang, and S. Cui, "Wireless sensor networks and internet of things: optimal estimation with nonuniform quantization and bandwith allocation," *IEEE Sensor J.*, vol. 13, no. 10, pp. 3568–3574, 2013.

[180] G. Alirezaei, M. Reyer, and R. Mathar, "Optimum power allocation in sensor networks for passive radar applications," *IEEE Trans. Wireless Commun.*, vol. 13, no. 6, pp. 3222–3231, 2014.

[181] M. Gastpar and M. Vetterli, "Source-channel communication in sensor neworks," *Lecture Notes in Computer Science*, vol. 2634, pp. 162–177, 2003.

[182] ——, "Power, spatio-temporal bandwidth, and distortion in large sensor network," *IEEE J. Sel. Areas Commun.*, vol. 23, no. 4, pp. 745–754, 2005.

[183] G. Thatte and U. Mitra, "Sensor selection and power allocation for distributed estimation in sensor networks: Beyond the star topology," *IEEE Trans. Signal Process.*, vol. 56, no. 7, pp. 2649 –2661, 2008.

[184] I. Bahceci and A. Khandani, "Linear estimation of correlated data in wireless sensor networks with optimum power allocation and analog modulation," *IEEE Trans. Commun.*, vol. 56, no. 7, pp. 1146 –1156, 2008.

[185] M. H. Chaudhary and L. Vandendorpe, "Performance of power-constrained estimation in hierarchical wireless sensor networks," *IEEE Trans. Signal Process.*, vol. 61, no. 2, pp. 724–739, 2013.

[186] R. Pabst et al, "Relay-based deployment concepts for wireless and mobile broadband radio," *IEEE Commun. Magazine*, vol. 42, no. 9, pp. 80–89, 2004.

[187] D. Persson, T. Eriksson, and P. Hedelin, "Packet video error concealment with Gaussin mixture models," *IEEE Trans. Image Processing*, vol. 17, no. 2, pp. 145–154, 2008.

[188] S. M. Kay, *Fundamentals of Statistical Signal Processing: Estimation Theory*. Upper Saddle River, NJ:Prentice Hall, 1993.

[189] S. A. Pasha, H. D. Tuan, and B. Vo, "Nonlinear Bayesian filtering using the unscented linear fractional transformation model," *IEEE Trans. Signal Process.*, vol. 58, no. 2, pp. 477–489, 2010.

[190] U. Rashid, H. D. Tuan, and H. H. K. nd H. H. Nguyen, "Joint optimization of source precoding and relay beamforming in wireless MIMO relay networks," *IEEE Trans. Commun.*, vol. 62, no. 2, pp. 488–499, 2014.

[191] J. Flam, S. Chatterjee, K. Kansanen, and T. Ekman, "On MMSE estimation: A linear model under Gaussian mixture statistics," *IEEE Trans. Signal Process.*, vol. 60, no. 7, pp. 3840 –3845, 2012.

[192] K. Zhou, J. C. Doyle, and K. Glover, *Robust and Optimal Control*. Englewood Cliffs, NJ: Prentice-Hall, 1996.

[193] H. Tuy, *Convex Analysis and Global Optimization*. Kluwer Academic, 1998.

[194] B. R. Marks and G. P. Wright, "A general inner approximation algorithm for nonconvex mathematical programms," *Operations Research*, vol. 26, no. 4, pp. 681–683, Jul. 1978.

[195] S. Julier, J. Uhlmann, and H. Durrant-Whyte, "A new method for the nonlinear transformation of means and covariances in filters and estimators," *IEEE Trans. Autom. Control*, vol. 45, no. 3, pp. 477 –482, 2000.

[196] Y. Bat-Shalom, X.-R Li, and T. Kirubajan, *Multi-target-multi-sensor tracking: principles and techniques*. New Your: Wiley, 2001.