Université de Lille

University of Lille

Doctoral School **ED MADIS**

University Department **CRIStAL Laboratory**

Thesis defended by **Ouafae Karmouda**

Defended on **9th December, 2022**

Academic Field **Computer engineering, Automatics and Signal Processing**

Speciality **Signal and Image Processing**

Tensor-based methods for multidimensional data: Learning and rank estimation

Thesis supervised by	Rémy Boyer	Supervisor
	Jérémie Boulanger	Co-Supervisor

Committee members

Referees	David Brie	Professor at Université de Lorraine
	Guillaume Ginolhac	Professor at Université Savoie Mont-Blanc
Examiners	Nadège Thirion-Moreau	Professor at Université de Toulon
	Nicolas Gillis	Professor at Université de Mons
Supervisors	Rémy Boyer	Professor at Université de Lille
	Jérémie Boulanger	Associate Professor at Université de Lille

Université de Lille

University of Lille

Doctoral School **ED MADIS**

University Department **CRIStAL Laboratory**

Thesis defended by **Ouafae Karmouda**

Defended on **9th December, 2022**

Academic Field **Computer engineering, Automatics and Signal Processing**

Speciality **Signal and Image Processing**

Tensor-based methods for multidimensional data: Learning and rank estimation

Thesis supervised by	Rémy Boyer	Supervisor
	Jérémie Boulanger	Co-Supervisor

Committee members

Referees	David Brie	Professor at Université de Lorraine
	Guillaume Ginolhac	Professor at Université Savoie Mont-Blanc
Examiners	Nadège Thirion-Moreau	Professor at Université de Toulon
	Nicolas Gillis	Professor at Université de Mons
Supervisors	Rémy Boyer	Professor at Université de Lille
	Jérémie Boulanger	Associate Professor at Université de Lille

Université de Lille

University of Lille

École doctorale **ED MADIS**

Unité de recherche **CRIStAL Laboratory**

Thèse présentée par **Ouafae Karmouda**

Soutenue le **9 décembre 2022**

Discipline **Génie informatique, automatique et traitement du signal**

Spécialité **Traitement du signal et de l'image**

Méthodes tensorielles pour les données multidimensionnelles : Apprentissage et estimation du rang

Thèse dirigée par	Rémy Boyer	directeur
	Jérémie Boulanger	co-directeur

Composition du jury

Rapporteurs	David Brie	professeur à l'Université de Lorraine
	Guillaume Ginolhac	professeur à l'Université Savoie Mont-Blanc
Examinateurs	Nadège Thirion-Moreau	professeur à l'Université de Toulon
	Nicolas Gillis	professeur à l'Université de Mons
Directeurs de thèse	Rémy Boyer	professeur à l'Université de Lille
	Jérémie Boulanger	mcf à l'Université de Lille

Keywords: kernel, learning, tensor train decomposition, cannonical polyadic decomposition, rank estimation

Mots clés : noyau, décomposition en train de tenseurs, estimation du rang

Tensor-based methods for multidimensional data: Learning and rank estimation

Abstract

Many scenarios involve the representation of data as multidimensional arrays called tensors, thus it is crucial to take this structure into account when analyzing the data. The curse of dimensionality, which refers to the exponential growth of processing and storage costs of large order tensors, is a modern scientific bottleneck. Consequently, tensor factorizations are crucial to reduce the complexity of tensor storage without sacrificing its multidimensionality. The Canonical Polyadic Decomposition (CPD) is the most commonly used tensor decomposition since it permits the representation of tensors as interpretable components. It is however difficult to determine the exact number of components (*i.e.* rank-one tensors) of this model, which constitutes its most significant limitation. For this purpose, the first part of this thesis proposes an optimization-based method called Joint FActors and RANk canonical estimation (FARAC) estimation that jointly estimates both the Canonical Polyadic (CP) factors and the canonical rank. The FARAC method is formulated as a convex optimization problem on each variable in which a sparse promoting constraint is added to the super diagonal of the core tensor of the CPD, whereas the Frobenius norm of the off-diagonal terms is constrained to be bounded. An alternated minimization strategy for the Lagrangian-based cost function is then proposed to solve the optimization problem. The second part of this thesis will focus on machine learning for tensor data. Machine learning algorithms often use similarity measures to perform supervised and unsupervised tasks, such as classification and clustering. Similarity can be determined using kernel methods, which are popular because of their performance in a variety of learning algorithms. Firstly, a method for extracting features from the CPD is analyzed, and it is demonstrated that the scaling ambiguity of the CPD negatively influences its performance. Accordingly, their kernel function cannot theoretically satisfy the properties of a kernel function. Using the extension of Support Vector Machines on tensors for classification, the model shows poor performance on real datasets. The performance of classification can be improved by modifying the choice of the kernel based on Grassmannian geometry. The second contribution in the context of supervised tensor learning aims to break the curse of dimensionality by using the Tensor Train Decomposition (TTD) which enjoys the benefits of good stability properties. As TT-cores are the building blocks for the TTD, a kernel function between two tensors is defined based on their similarities with respect to their respective TT-cores. As TTD is not unique, learning is considered on the subspaces spanned by TT-cores defined using the tensor-linear algebra of third-order tensors. While some of the ambiguities model-based have been mitigated, others still remain, so, another contribution has been proposed to define similarities between TT-cores on the basis of the subspaces spanned by their second unfoldings. In addition to being computationally efficient, this approach will eliminate all ambiguities associated with the TT model.

Keywords: kernel, learning, tensor train decomposition, cannonical polyadic decomposition, rank estimation

CRIStAL Laboratory
UMR CRIStAL – Université de Lille - Campus scientifique – Bâtiment ESPRIT – Avenue Henri Poincaré – 59655 Villeneuve d'Ascq

Méthodes tensorielles pour les données multidimensionnelles : Apprentissage et estimation du rang

Résumé

De nombreux scénarios impliquent la représentation des données sous forme de tableaux multidimensionnels appelés tenseurs, il est donc crucial de prendre en compte cette structure lors de l'analyse des données. La malédiction de la dimensionnalité est un problème associé aux tenseurs d'ordre élevé et qui fait référence à l'augmentation exponentielle des coûts de traitement et de stockage. Par conséquent, les décompositions tensorielles sont cruciales pour réduire la complexité du stockage de tenseurs sans sacrifier leur multidimensionnalité. La décomposition canonique polyadique CP est la décomposition de tenseurs la plus couramment utilisée car elle permet la représentation des tenseurs en tant que composants interprétables (tenseurs de rang 1). Il est cependant difficile de déterminer le nombre exact de composants de ce modèle, ce qui constitue sa limite la plus importante. Pour cela, la première partie de cette thèse propose une méthode basée sur l'optimisation convexe pour estimer conjointement les facteurs CP et le rang canonique appelée FARAC (Joint FActors and RANk canonical estimation). La méthode FARAC est formulée comme un problème d'optimisation convexe dans lequel une contrainte de parcimonie est rajoutée à la superdiagonale du tenseur central de la CP, tandis que la norme de Frobenius des termes hors diagonaux est contrainte d'être bornée. Une stratégie de minimisation alternée du Lagrangien est ensuite proposée pour résoudre le problème d'optimisation. La deuxième partie de cette thèse portera sur l'apprentissage automatique pour les données tensorielles. Les algorithmes d'apprentissage automatique utilisent souvent des mesures de similarité pour effectuer des tâches supervisées et non supervisées, telles que la classification et le regroupement. La similarité peut être déterminée à l'aide des méthodes à noyau, qui sont populaires en raison de leurs performances dans une variété d'algorithmes d'apprentissage. Tout d'abord, une méthode proposant un noyau tensoriel basé sur l'extraction de facteurs à partir de la CP est analysée. Il est ensuite démontré que celle-ci ne satisfait pas théoriquement les propriétés d'une fonction noyau à cause de l'ambiguïté de mise à l'échelle de la CP. Celle-ci affecte négativement ses performances de classification. En modifiant le choix du noyau basé sur la géométrie Grassmannienne, il est démontré que les performances de classification peuvent être améliorées. Pour casser la malédiction de la dimensionnalité, la décomposition en trains de tenseurs TTD qui bénéficie de bonnes propriétés de stabilité serait utilisée pour définir une nouvelle fonction noyau dans l'espace tensoriel. Comme les cœurs TT sont les éléments constitutifs de la TTD, la fonction noyau proposée entre deux tenseurs est définie en fonction des similitudes entre leus coeurs TT respectifs. Comme la TTD n'est pas unique, les sous-espaces engendrés par les cœurs TT seront considérés. Ceux-ci sont définis à l'aide de l'algèbre linéaire des tenseurs d'ordre 3. Alors que certaines des ambiguïtés ont été éliminées, d'autres perdurent. Une autre fonction noyau serait alors proposée qui définit les similitudes entre leurs cœurs TT sur la base des sous-espaces engendrés par leurs deuxième dépliements . En plus d'être efficace numériquement, cette approche éliminera toutes les ambiguïtés associées au modèle TT.

Mots clés : noyau, décomposition en train de tenseurs, estimation du rang

CRIStAL Laboratory
UMR CRIStAL – Université de Lille - Campus scientifique – Bâtiment ESPRIT – Avenue Henri Poincaré – 59655 Villeneuve d'Ascq

Acknoledgments

Thank you, God, for providing me with courage, support, and valuable individuals who have made my thesis journey successful.

My first thanks to the referees, Mr. David Brie, Professor at the University of Lorraine, and Mr. Guillaume Ginolhac, Professor at the University Savoie Mont Blanc, for their thorough reading of this PhD thesis and their helpful suggestions. I would like to express my deepest gratitude to the jury members, Mrs. Nadège Thirion-Moreau, Professor at the University of Toulon, and Mr. Nicolas Gillis, Professor at the University of Mons, for accepting my work for examination.

My supervisor Rémy Boyer has been of significant assistance to me throughout this thesis. It would not have been possible to accomplish this thesis without the expertise, research experience, and advice he provided. I greatly appreciated his enthusiasm and critical approach to science.

My sincere gratitude goes out to my supervisor, Jérémy. This work would not have been possible without his invaluable guidance, assistance, and constant presence. Working with him throughout this thesis has been a pleasure and has provided me with valuable insights.

I would like to thank Mr. Gérard Favier, CNRS Emeritus Research Director and Professor at the University of Nice Sophia Antipolis, Mr. André Lima Férrer de Almeida, Associate Professor with the Teleinformatics Engineering Department of the Federal University of Ceara and Mr. Yassine Zniyed, Associate Professor at the University of Toulon with whom I had the opportunity to contribute.

My gratitude goes out to all Sigma team members with whom I shared my days at CRIStAL laboratory. I am grateful for their mutual assistance, informative advice, enriching discussions, and the pleasant moments that we shared.

The support, encouragement and love of my father Jamal and mother Jamila have been instrumental to my success and motivated me to reach beyond my academic goals. I wish to express my gratitude to them for supporting my aspirations and encouraging me to pursue my dreams and reach for the stars.

I would like to express my gratitude to my mother-in-law Asmahane and my father-in-law Abdeljalil for their continuous and unparalleled love, help and support they have consistently provided to me.

I would like to express my gratitude to my sisters Firdaouss, Faiza and Chaimae for their love, their care, and their support. Their presence in this life is my treasure. I extend my sincere greetings to them.

Finally, my husband Yassine deserves special thanks for his unconditional love, care, and support during these challenging years.

Sommaire

List of Tables

List of Figures

Chapter 1

General Introduction

Outline of the current chapter

1.1 Tensors

The amount of data generated today is enormous, and therefore methods for analyzing this data are always needed. Data is often stored in matrices, and matrix decompositions can be used to extract relevant information. However, the matrix decompositions are not unique unless some additional constraints are imposed, such as orthogonality. Constraints of this type can be unrealistic. Using matrices to represent data has also the disadvantage of obscuring the data structure of tensors due to their inability to exploit the structural information of the tensors's representation. As an example, an Functional magnetic resonance imaging or functional MRI (FMRI) measures the electrical activity of the brain indirectly by measuring changes in blood oxygen levels. In FMRI images, the brain is represented as a volume and its evolution over time is shown as a fourth

spatial dimension. If input data is flattened, FMRI images lose their spatial and temporal representation. This type of data requires four dimensions to be represented. It is possible to do this using a tensor of order 4 without destroying its multidimensional structure. Tensors are higher-order extensions of vectors and matrices. Tensors can represent data in more than one dimension; however, one drawback is the curse of dimensionality, which means that more entries need to be stored as the tensor gets larger. Thus, high-dimensional tensors are decomposed into lower dimensionnal factors using tensor decompositions. Using tensor decompositions, data representation complexity and computation time can be reduced. The CPD is a widespreadly used tensor decomposition in different fields due to its atttractive property of uniqueness under mild conditions and up to some scaling and permutation ambiguities. Its first application in psychology was in [119, 46]. Pioneered the application of CPD in chemistry, [18] uses the CPD for the analysis of fluorescence data. A number of studies have utilized the CPD to analyze time-frequency-transformed Electroencephalography (EEG) data. These studies include [34, 86], and others. Furthermore, the CPD has shown to be useful for sensor array processing in wireless communication [99, 96, 128]. In practice, however, determining CPD's canonical rank poses a major problem. In this context, a new method called FARAC will be proposed to jointly estimate the CP factors as well as the cannonical rank of the CPD.

1.2 Tensorial Kernel Methods

In the machine learning community, kernel methods are widely used due to their high performance in a wide range of learning tasks [17, 95, 132, 78, 62, 94, 134, 12, 71]. Additionally, kernel methods have gained a great deal of attention in the statistics and mathematics communities due to their solid theoretical foundation, in comparison to other methods such as deep learning. Among their greatest advantage, these approaches allow the use of machine learning algorithms such as Support Vector Machines (SVM),Principal Component Analysis (PCA),Linear Discriminant Analysis (LDA), among others, on a variety of data types, such as graphs, text, images, subspaces, and tensors without the need for vectorization. They can be seen as a kind of similarity measure between two

objects. In addition, kernel methods can be used to extend learning algorithms based on the assumption that data is linearly separable to the most common case of nonlinearity. By mapping data to a Hilbert space, linear separation is possible. The advantage of this mapping is that it is implicit since the explicit representation of data in the new Hilbert space is not necessary; only the dot products of the data in the feature space are calculated by a kernel function.
In the second part of the thesis, we will be interested in kernel methods on the tensor space, where the dataset consists mainly of tensors. There are two main reasons why vectorizing tensors is not a suitable approach. The first problem is that the data structure will be lost. Secondly, vectorization results in high-dimensional vectors with high computational costs. To avoid the curse of dimensionality, tensor decompositions will be used. A tensorial kernel function is then defined from the similarities between lower-order factors obtained from the decomposition of a tensor.

1.3 Contributions

The contributions of this thesis can be devided in two parts. Please see Figure 1.1 for an overview.

- In the first part of this thesis and as a first contribution, a new method called FARAC is proposed in chapter 2 that jointly estimates the CP factors and the canonical rank. Since the CPD represents a special case of the Tucker Decomposition (TD), the optimization problem proposed is formulated as a constrained TD in which the number of the Tucker core tensor's superdiagonal elements is minimized as an objective function by applying a sparse promoting constraint. Using the Frobenius norm, a first constraint is added to the error loss, and a second one is added to the offdiagonal terms, making them non-zero but bounded. The proposed optimization problem is then solved using an alternative minimization strategy based on the Lagrangian-based cost function. The validation of FARAC was conducted using synthetic and real data.

- The thesis' second part aims to define tensorial kernel functions that can handle the model ambiguities of the CPD and the TTD.
  - In chapter 4, an analysis of the state-of-the-art method presented in [72] is proposed. This latter proposes a tensorial kernel function that evaluates the similarity between two tensors using their CP factors. The scaling ambiguities associated with the CPD make this method ineffective. As the similarity measure determines the parameters of the learning algorithm, classification performance is adversely affected. The kernel choice can be modified to better handle the scaling ambiguities of the CPD and improve classification accuracy by using Grassmannian geometry. In order to avoid convergence problems during the computation of the CPD, the TTD will be used to determine the CP factors using the equivalence between the CPD and the TTD. This can be done by applying the JIRAFE algorithm recently developed [151] to derive the CP factors from the TT-cores.
  - In chapter 5, the third contribution is described in which a kernel function is defined on the tensor space using the TTD. The similarity between two tensors will be determined by a TT-core comparison. However, the non-unicity of the TT-cores could adversely affect this measure. To overcome this problem, it is proposed to evaluate the similarities between the subspaces spanned by the TT-cores. These subspaces will be characterized by tensor-linear algebra, also known as t-algebra that generalizes the linear algebra concepts for third-order tensors. The proposed kernel function allows thus to minimize the impact of the TTD's non-unicity on the evaluation of the similarity between two tensors.
  - In chapter 6, the 4-th contribution is presented which defines a tensorial kernel function that completely mitigates the ambiguities associated with TTD. It is shown in this context that the subspaces spanned by the second unfoldings of the TT-cores are invariant to the TT-cores' non-unicity. A tensorial kernel function will be defined based on these subspaces. Further, it will be shown that the kernel proposed

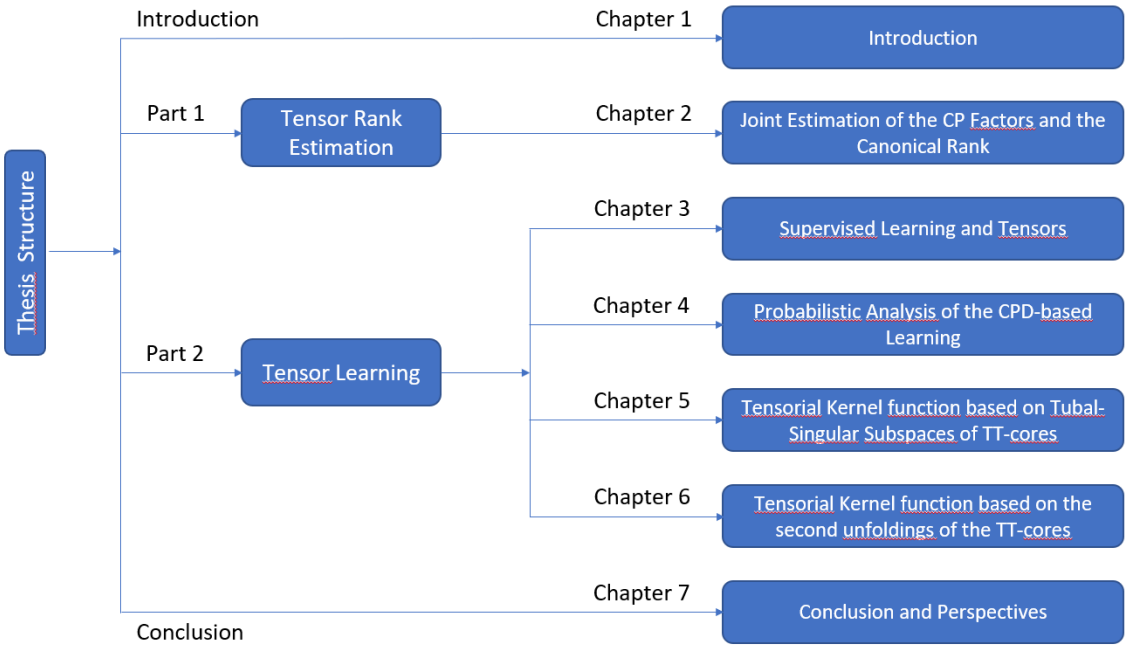


Figure 1.1 – Thesis Structure.

in [90] which is based on the subspaces spanned by the HOSVD factors can be equivalently computed using the proposed kernel in this work. This equivalence occurs because the TD and TTD introduced in JIRAFE are equivalent. There is, however, a curse of dimensionality associated with HOSVD, which appears in [90]. With TTD, the curse of dimensionality can be overcome and the same subspaces can be retrieved.

1.4 List of publications

- International journals:
  - [102] **O. Karmouda**, J. Boulanger and R. Boyer, *Joint Factors and Rank Estimation for the Canonical Polyadic Decomposition Based on Convex Optimization* in: IEEE Access, vol. 10, pp. 82295-82304, 2022.
- International conferences:

  - [103] **O.Karmouda**, J.Boulanger and R.Boyer. *On the analysis of the Canonical Polyadic Decomposition (CPD)-based tensor learning* in: The 56th Asilomar Conference on Signals, Systems, and Computers, 2022.
  - [104] **O.Karmouda**, J.Boulanger and R. Boyer, *Speeding Up of Kernel-Based Learning for High-Order Tensors* in: IEEE International Conference in: The 46th Acoustics, Speech and Signal Processing (ICASSP), 2021.
  - [105] **O.Karmouda**, R.Boyer and J.Boulanger. *High-Dimensional Data Learning Based on Tensorial-Singular Space of Tensor Train Cores* in: The 30th European Signal Processing Conference (EUSIPCO), 2022.

- Book Chapters:

  - [150] Y.Zniyed, **O.Karmouda**, R.Boyer, J.Boulanger, A.De Almeida, G.Favier, *Tensors for Data Processing: Theory, Methods, and Applications. Chap 14: Structured tensor train decomposition for speeding up kernel-based learning*. Academic Press, 2022.

- National conferences:

  - [100] **O.Karmouda**, J.Boulanger, R.Boyer, *Apprentissage supervisé à noyau basé sur la décomposition Canonique Polyadique (CP)*, Gretsi, Nancy, 2022.
  - [101] **O.Karmouda**, J.Boulanger, R.Boyer, *APPRENTISSAGE SUPERVISÉ RAPIDE POUR DES DONNÉES TENSORIELLES*. 2eme Conférence Internationale Francophone sur la Science des Données, Marseille, 2021.

- Seminars

  - Speeding up of kernel-based learning for high-order tensor, Scube seminar CentraleSupélec, 2021.
  - Tensor Learning on Grassmann Manifold, SIGMA seminar, 2020.

Part I

Tensor Rank Estimation

Chapter 2

Joint Factors And Canonical Rank Estimation For The CPD

Outline of the current chapter

In the present study, we propose estimating both the canonical rank and the CP factors from noisy observations using a constrained TD in which a sparse promoting constraint is added to the superdiagonal of the core tensor of the CPD, whereas the Frobenius norm of the offdiagonal terms is constrained to be bounded. We give a formulation of the problem of interest as a convex optimization problem on each variable. The reminder of the paper is organised as follows:
First, we introduce some notations and preliminaries in multilinear algebra in section 2.1. In section 2.2 , we describe our proposed approach FARAC and our algorithm for solving the problem. We then present some existing works for the estimation of the canonical rank in section 2.3. As a final step, we do a number of numerical experiments in section 2.4 to evaluate and compare our proposed approach with CORCONDIA method.

2.1 Tensor Background

Notations

Vectors are denoted by blodface lowercase letter *i.e.* $\boldsymbol{x}$, Matrices are denoted by boldface capital letters *i.e.* $\boldsymbol{X}$, tensors are denoted by caligraphic letters *i.e.* $\mathcal{X}$. The i-th entry of a vector $\boldsymbol{x}$ is denoted by $\boldsymbol{x}(i)$. The (i_1, i_2)-th entry of a matrix $\boldsymbol{X}$ is denoted by $\boldsymbol{X}(i_1, i_2)$. The $(i_1, \ldots, i_N)$-th entry of a tensor $\mathcal{X}$ is denoted by $\mathcal{X}(i_1, \ldots, i_N)$.
Tubes represent higher-order generalization of columns and rows of a matrix. They are defined by fixing all indices except one. Third order tensors have column, row and tube fibers denoted by $\mathcal{X}(:, i_2, i_3)$, $\mathcal{X}(i_1, :, i_3)$, and $\mathcal{X}(i_1, i_2, :)$. In this section, we recall some algebraic definitions on tensor algebra from [135]:

Scalar product and the Frobenius norm of a Tensor

Definition 1. *(Inner product): The inner product $\langle .,. \rangle$ of two N-order tensors $\mathcal{X}, \mathcal{Y} \in \mathbb{R}^{I_1 \times \cdots \times I_N}$ is defined as:*

$$\langle \mathcal{X}, \mathcal{Y} \rangle = \sum_{i_1=1}^{I_1} \sum_{i_2=1}^{I_2} \dots \sum_{i_N=1}^{I_N} \mathcal{X}(i_1, \dots, i_N) \mathcal{Y}(i_1, \dots, i_N).$$

Definition 2. *(Frobenius Norm of a Tensor): The norm Frobenius norm $\|.\|_F$ of a tensor $\mathcal{X}$ is defined as:*

$$\|\mathcal{X}\|_F = \sqrt{\sum_{i_1=1}^{I_1} \sum_{i_2=1}^{I_2} \dots \sum_{i_N=1}^{I_N} \mathcal{X}^2(i_1, \dots, i_N)}.$$

Transforming a Tensor to a Matrix

Definition 3. *(Unfolding) The n-mode unfolding of a tensor $\mathcal{X}$ is a matrix denoted by $\boldsymbol{X}_{(n)}$, whose columns are the n-mode fibers of $\mathcal{X}$.*

Diagonal Tensor

Definition 4. *(Diagonal tensor) A tensor $\mathcal{X} \in \mathbb{R}^{I_1 \times \cdots \times I_N}$ is diagonal if all of its entries are zero except those in its superdiagonal, that is: $\mathcal{X}(i_1, \dots, i_N) \neq 0$ only if $i_1 = i_2 = \cdots = i_N$.*

Products

Definition 5. *(Outer product) The outer product of N vectors $\boldsymbol{u}_1 \circ \boldsymbol{u}_2 \circ \cdots \circ \boldsymbol{u}_N$ with $u_n \in \mathbb{R}^{I_n}$ is a tensor $\mathcal{X} \in \mathbb{R}^{I_1 \times \cdots \times I_N}$ of order N with:*

$$\mathcal{X}(i_1, \cdots, i_N) = \boldsymbol{u}_1(i) \cdots \boldsymbol{u}_N(i_N).$$

Definition 6. *(Kronecker product) The Kronecker product denoted by $\otimes$ of a matrix $\boldsymbol{A}$ of size $I_1 \times I_2$ and a matrix $\boldsymbol{B}$ of size $J_1 \times J_2$ is a matrix of size $(I_1 J_1) \times (I_2 J_2)$ defined as:*

$$\boldsymbol{A} \otimes \boldsymbol{B} = \begin{bmatrix} a_{11}\boldsymbol{B} & \cdots & a_{1I_2}\boldsymbol{B} \\ \vdots & & \vdots \\ & \ddots & \\ a_{I_1 1}\boldsymbol{B} & \cdots & a_{I_1 I_2}\boldsymbol{B} \end{bmatrix}$$
$$= [\boldsymbol{a}_1 \otimes \boldsymbol{b}_1 \quad \boldsymbol{a}_1 \otimes \boldsymbol{b}_2 \quad \cdots \quad \boldsymbol{a}_{I_2} \otimes \boldsymbol{b}_{J_2-1} \quad \boldsymbol{a}_{I_2} \otimes \boldsymbol{b}_{J_2}].$$

Definition 7. *(Khatri-Rao product) The Khatri-Rao product denoted by $\odot$ of a matrix $\boldsymbol{A}$ of size $I_1 \times I_2$ and a matrix $\boldsymbol{B}$ of size $J \times I_2$ is denoted by $\boldsymbol{A} \odot \boldsymbol{B}$ is a matrix of size $(I_1 J) \times I_2$ and is defined as:*

$$\boldsymbol{A} \odot \boldsymbol{B} = [\boldsymbol{a}_1 \otimes \boldsymbol{b}_1 \quad \boldsymbol{a}_2 \otimes \boldsymbol{b}_2 \quad \cdots \quad \boldsymbol{a}_{I_2} \otimes \boldsymbol{b}_{I_2}].$$

Definition 8. *(n-mode multiplication) The n-mode product denoted by $\times_n$ of a tensor $\mathcal{X} \in \mathbb{R}^{I_1 \times \cdots \times I_N}$ with a matrix $\boldsymbol{U} \in \mathbb{R}^{J \times I_n}$ is a tensor of order N and size $I_1 \times \ldots I_{n-1} \times J \times \cdots \times I_N$ and is defined by:*

$$(\mathcal{X} \times_n \boldsymbol{U})(i_1, \ldots, i_{n-1}, j, i_{n+1}, \ldots, i_N) = \sum_{i_n=1}^{I_n} \mathcal{X}(i_1, \ldots, i_N)\boldsymbol{U}(j, i_n).$$

2.1.1 Tensor ranks

The concept of matrix rank can be generalized to tensors in a variety of ways. Their definitions are quite similar but their properties are very different. As a first generalization of the concept of rank for tensors, the n-rank is defined as follows:

Definition 9. *(n-rank) The n-rank R_n of a tensor $\mathcal{X}$ is the generalization of the columm (row) rank of matrices. It is defined as the dimension of the space spanned by the I_n dimensionnal vectors of the n-mode unfolding i.e:*

$$R_n = rank(\boldsymbol{X}_{(n)}).$$

The N-tuple $(R_1, \ldots, R_N)$ of n-ranks is called the multilinear rank of $\mathcal{X}$.

Definition 10. *(Rank-one tensor) In the same way that a rank-one matrix is derived from the outer product of two vectors, a tensor $\mathcal{X} \in \mathbb{R}^{I_1 \times \cdots \times I_N}$ of order N is rank-one if it can be written as the outer product of N vectors:*

$$\mathcal{X} = \boldsymbol{u}_1 \circ \cdots \circ \boldsymbol{u}_N := \bigcirc_{n=1}^{N} \boldsymbol{u}_n, \tag{2.1}$$

where $\boldsymbol{u}_n \in \mathbb{R}^{I_n}$ with $1 \leq n \leq N$.

Definition 11. *(Canonical rank) The canonical rank of a tensor is defined as the minimum number of rank-one tensors necessary to produce $\mathcal{X}$ as their sum.*

Typical, maximal rank

Another difference of the tensor rank from the matrix rank has to deal with typical and maximal ranks. If the entries of a tensor are drawn from a continuous probability distribution, typical ranks are those encoutered with propbability one and the maximam rank is the largest attainable rank. In fact, while the typical rank and the maximum rank are equal to $\min\{I_1, I_2\}$ for $I_1 \times I_2$ matrices, they may be different for tensors. There may also exist more that one typical rank over $\mathbb{R}$. Formulas for specific tensors of specific sizes are given in [135, 111]. For a general third-order tensor $\mathcal{X} \in \mathbb{R}^{I_1 \times I_2 \times I_3}$, the following upper bound holds [51]:

$$R \leq \min\{I_1 I_2, I_1 I_3, I_2 I_3\}. \tag{2.2}$$

2.1.2 The Canonical Polyadic Decomposition

The CPD has gained popularity following its introduction by [119] and [46]. The CPD of a tensor $\mathcal{X}$ of canonical rank-R is expressed as the sum of R rank-one

tensors:

$$\mathcal{X} = \sum_{r=1}^{R} \bigcirc_{n=1}^{N} \boldsymbol{u}_{n,r} = [\![\boldsymbol{U}_1, ..., \boldsymbol{U}_N]\!], \tag{2.3}$$

where $[\![\boldsymbol{U}_1, ..., \boldsymbol{U}_N]\!]$ denotes the CPD of $\mathcal{X}$. Each matrix $\boldsymbol{U}_n$ is of size $I_n \times R$ and the vectors $\boldsymbol{u}_{n,r}$ constitute its columns. $\boldsymbol{U}_n$, $1 \leq n \leq N$ are called the CP factors.

Unicity

The attractiveness of CPD yields in its uniqueness without any additional constraints contrary to matrix decompositions that are non unique unless some additionnal constraints are added such as orthogonality for the Singular Value Decomposition (SVD). The most well-known result about uniqueness is given by [52] which gives sufficient condition for uniqueness of the CPD. The latter is given as follows for rank-R tensor $\mathcal{X}$:

$$\sum_{n=1}^{N} k_{rank}(\boldsymbol{U}_n) \geq 2R + N - 1, \tag{2.4}$$

Whenever the condition in eq.(2.4) does not hold, uni-mode uniqueness condition (uniqueness of one CP factor) is derived for three-order tensors [143]. Sufficient conditions for uniqueness of the CPD of a fourth-order tensor with one full column rank factor and at most three collinear factors(having one (or more) column(s) proportional to another column) are provided in[26].

- Column permutation : Factors $\boldsymbol{U}_n$ are unique up to a common (over n) column permutation, *i.e.*,

  $$\mathcal{X} = [\![\boldsymbol{U}_1, ..., \boldsymbol{U}_N]\!] = [\![\boldsymbol{U}_1\boldsymbol{\Pi}, ..., \boldsymbol{U}_N\boldsymbol{\Pi}]\!], \tag{2.5}$$

  for any $R \times R$ permutation matrix $\boldsymbol{\Pi}$.

- Column scaling : For a given couple (n, r), the column $\boldsymbol{u}_{n,r}$ is unique up to

a scaling factor denoted $\beta_{n,r} \in \mathbb{R} \setminus \{0\}$ *i.e.*

$$\mathcal{X} = \sum_{r=1}^{R} \bigcirc_{n=1}^{N} (\beta_{n,r} \boldsymbol{u}_{n,r}), \tag{2.6}$$

with $\prod_{n=1}^{N} \beta_{n,r} = 1$.
By normalizing the vectors $\beta_{n,r}\boldsymbol{u}_{n,r}$, the CPD of $\mathcal{X}$ can be expressed as follows:

$$\mathcal{X} = \sum_{r=1}^{R} \boldsymbol{\lambda}(r) \bigcirc_{n=1}^{N} (\mathrm{sgn}(\beta_{n,r})\tilde{\boldsymbol{u}}_{n,r}), \tag{2.7}$$

where $\tilde{\boldsymbol{u}}_{n,r}$ are unit vectors and $\boldsymbol{\lambda}(r) = \prod_{n=1}^{N} \|\beta_{n,r}\boldsymbol{u}_{n,r}\|$, $1 \leq r \leq R$.
From eq.(2.7), the canonical rank can also be defined as the number of strictly positive and ordered values of $\boldsymbol{\lambda}$.
Unless specified otherwise, the signs of the scalars $\beta_{n,r}$ will not appear in the formula of the CPD and considered absorbed into the normalized column factors; therefore, the following expression of the CPD will be considered:

$$\mathcal{X} = \sum_{r=1}^{R} \boldsymbol{\lambda}(r) \bigcirc_{n=1}^{N} \boldsymbol{u}_{n,r}, \tag{2.8}$$

where $\boldsymbol{u}_{n,r}$ are unit vectors (The tilde notation will be omitted for ease of notation).

Computation of the CPD:

To compute the CPD of a rank-R tensor $\mathcal{X}$ from possibly noisy data, a least squares criterion is usually adopted :

$$\min_{\boldsymbol{U}_n, 1 \leq n \leq N} \|\mathcal{X} - [\![\boldsymbol{U}_1, ..., \boldsymbol{U}_N]\!]\|_F^2. \tag{2.9}$$

The most popular algorithm to solve the non convex problem in eq.(2.9) is called ALS [120]. Its idea consists in splitting the problem into smaller linear

square problems that are solved iteratively. At each step, a CP factor is computed conditionnaly to the remaining ones by solving the following linear regression problem:

$$\min_{\boldsymbol{U}_n, 1\leq n\leq N} \left\| \boldsymbol{X}_{(n)} - \boldsymbol{U}_n \boldsymbol{\Lambda} \left(\bigodot_{\substack{l=N \\ l\neq n}}^{l=1} \boldsymbol{U}_l \right)^T \right\|_F^2 ,$$

where $\boldsymbol{\Lambda}$ is a diagonal matrix with $\boldsymbol{\lambda}$ in its diagonal and $\bigodot_{\substack{l=N \\ l\neq n}}^{l=1}$ denotes the kronecker product of the factors $\boldsymbol{U}_l$.

Once all the subproblems are solved, an iteration is completed. To solve the optimization problem in eq.(2.9), the canonical rank R should be first specified at the beginning. If the data is noise free, one attempts of simply comparing the error loss and identifying the number of components that give a zero loss. However, this approach may not be feasible since if the number of rank-one tensors is strictly smaller that the canonical rank R, the optimization problem in eq.(2.9) can be ill-posed [126, 140] contrarly to the matrix case where the best low rank approximation is provided by the truncated SVD [22]. Even it is not the case (the low-rank approximation exists), this procedure does not work in the presence of noise since the decrease of the error loss when more components are extracted can be very small, without a clear jump[24].

2.1.3 Tucker Decomposition

The TD was first introduced in [75, 53, 77, 76] and was popularized by [65] for a particular method to compute it the under the name of HOSVD. Among the applications of this latter are facial expression analysis [92], noise filtering of color images [40]. By providing an orthonormal basis, the HOSVD method can be extended to higher-order tensors extending the SVD subspace method [13, 14].

Definition 12. *(**Tucker Decomposition** **TD**) The* TD *decomposes a tensor* $\mathcal{X} \in \mathbb{R}^{I_1 \times \cdots \times I_N}$ *into a core tensor* $\mathcal{G}$ *multiplied by a factor matrix* $\boldsymbol{T}_n$ *in each mode n:*

$$\mathcal{X} = \mathcal{G} \times_1 \boldsymbol{T}_1 \times_2 ... \times_N \boldsymbol{T}_N, \tag{2.10}$$

where $\boldsymbol{T}_n$ *are of size* $I_n \times R_n$, $1 \leq n \leq N$ *and* $\mathcal{G}$ *is the core tensor of size* $R_1 \times \cdots \times R_N$.

The core tensor $\mathcal{G}$ compresses the tensor and captures interactions between the columns of the factors $\boldsymbol{T}_n$ which can be useful in many data analysis [135, 108].
As opposed to CPD, TDis not unique. It should also be noted that contrarly to the CPD, a low multilinear rank approximation is always well-posed [6]; however, truncation may not be optimal in the least squares sense, although this can usually be determined by taking into account the degree of truncation.
An interesting fact is that the CPD can be seen as a special case of the TD where the core tensor is diagonal with $R_1 = \cdots = R_N = R$.

2.2 Proposed method (FARAC)

The purpose of this section is to present the method FARAC for estimating the canonical rank and the CP factors simultaneously for a rank-R tensor $\mathcal{X}$. Beginning from the fact that the CPD is a special case of the TD, we use this latter with a constrained core tensor $\mathcal{G}$ with all its dimensions set to a large value R_0. We will minimize the number of non-null terms of the superdiagonal of $\mathcal{G}$ under a first constraint on the error loss and a second one on the offdiagonal terms by allowing them to be non-zero but bounded. Our goal is to find a core tensor structured as shown in Fig. 2.1. This can be accomplished by minimizing the superdiagonal of the core tensor using the l_0 norm $\|.\|_0$ as an objective function, using the Frobenius norm for the error loss and on the offdiagonal terms.

2.2.1 Optimization Problem

In the following, $\mathcal{G}$ denotes the CP core tensor, $\boldsymbol{\lambda}$ its superdiagonal and $\tilde{\mathcal{G}}$ is defined according to:

$$\mathcal{G} = \text{diag}(\boldsymbol{\lambda}) + \tilde{\mathcal{G}},$$

where $\text{diag}(\boldsymbol{\lambda})$ is a diagonal tensor whose superdiagonal is $\boldsymbol{\lambda}$.
Our optimization problem can then be expressed mathematicaly as follows:

$$\begin{aligned} &\underset{\mathcal{G},(\boldsymbol{U}_n)_n}{\text{minimize}} \ \|\boldsymbol{\lambda}\|_0, \\ &\text{subject to } \frac{1}{2}\left\|\mathcal{X} - \mathcal{G} \overset{N}{\underset{n=1}{\times}}_n \boldsymbol{U}_n\right\|_F^2 \leq \epsilon_1, \\ &\qquad\qquad \frac{1}{2}\left\|\tilde{\mathcal{G}}\right\|_F^2 \leq \epsilon_2, \end{aligned} \tag{2.11}$$

where ϵ_1 and ϵ_2 are small positive constants. Unlike existing methods that estimate the canonical rank, we do not constrain the factors $\boldsymbol{U}_n$ to be orthogonal [35, 48]. However, since minimizing the l_0 norm is NP-hard [28], we minimize the l_1 norm $\|.\|_1$ of $\boldsymbol{\lambda}$. Eq. (2.11) becomes:

$$\begin{aligned} &\underset{\mathcal{G},(\boldsymbol{U}_n)_n}{\text{minimize}} \ \alpha_1\|\boldsymbol{\lambda}\|_1, \\ &\text{subject to } \frac{1}{2}\left\|\mathcal{X} - \mathcal{G} \overset{N}{\underset{n=1}{\times}}_n \boldsymbol{U}_n\right\|_F^2 \leq \epsilon_1, \\ &\qquad\qquad \frac{1}{2}\left\|\tilde{\mathcal{G}}\right\|_F^2 \leq \epsilon_2, \end{aligned} \tag{2.12}$$

where α_1 is a strictly positive hyper-parameter.

2.2.2 Derivation of FARAC

In this section, we present the proposed approach for solving the optimization problem with constraints in eq. (2.12). Recall that the Lagrangian function is the augmented objective function by the constraint equations using the Lagrangian multipliers. Following this, the Lagrangian function of the problem in eq. (2.12)

Algorithme 1 : Joint **FA**ctors and **RA**nk **C**anonical estimation for the Polyadic decomposition (FARAC)

Input: $\mathcal{X} \in \mathbb{R}^{I_1 \times \ldots I_N}$: CP Tensor, R_0: Upper bound of the rank of $\mathcal{X}$.
Require: $\eta_1, \eta_2 \in [0,1]$: Exponential decay rates; η_3: Stepsize; N_{iter}: Number of iterations, μ: Threshold parameter $\epsilon = 10^{-8}$: Parameter to avoid numerical instabilities.
Initialize: For $1 \leq n \leq N$:

$$\boldsymbol{U}_n^{(0)} = \begin{cases} \text{SVD}(\boldsymbol{X}_{(n)}, R_0) \text{ if } I_n > R_0, \\ conc\Big(\text{SVD}(\boldsymbol{X}_{(n)}, R_0), (R_0 - I_n) \text{ random uniform vectors}\Big) \text{ else.} \end{cases}$$

- Entries of $\mathcal{G}^{(0)}$ of order N and all dimensions equal R_0 are drawn from the standard uniform probability distribution.
- $m_{\tilde{\mathcal{G}}}^{(0)} = v_{\tilde{\mathcal{G}}}^{(0)} = 0$ (Initialize the first and the second moment estimates).

for $t = 1, \ldots, N_{\text{iter}}$:

1: Compute $\boldsymbol{U}_n^{(t)}$ from eq. (2.14).
2: Update biased first moment estimate of the offdiagonals:

$$m_{\tilde{\mathcal{G}}}^{(t)} \leftarrow \eta_1 m_{\tilde{\mathcal{G}}}^{(t-1)} + (1-\eta_1)\nabla_{\tilde{\mathcal{G}}}^{(t)}(r_1, \ldots, r_N).$$

3: Update biased second raw moment estimate of the offdiagonals:

$$v_{\tilde{\mathcal{G}}}^{(t)} \leftarrow \eta_2 v_{\tilde{\mathcal{G}}}^{(t-1)} + (1-\eta_2)\nabla^{2(t)}_{\tilde{\mathcal{G}}}(r_1, \ldots, r_N).$$

4: Compute bias-corrected first and second moment estimates of the offdiagonals:

$$\hat{m}_{\tilde{\mathcal{G}}}^{(t)} \leftarrow \frac{m_{\tilde{\mathcal{G}}}^{(t)}}{1-\eta_1^t}; \quad \hat{v}_{\tilde{\mathcal{G}}}^{(t)} \leftarrow \frac{v_{\tilde{\mathcal{G}}}^{(t)}}{1-\eta_2^t}.$$

5: Update each entry $(r_1, \ldots, r_N)$ of $\tilde{\mathcal{G}}$:

$$\tilde{\mathcal{G}}^{(t)}(r_1, \ldots, r_N) \leftarrow \tilde{\mathcal{G}}^{(t-1)}(r_1, \ldots, r_N) - \eta_3 \frac{\hat{m}_{\tilde{\mathcal{G}}}^{(t)}}{\sqrt{\hat{v}_{\tilde{\mathcal{G}}}^{(t)}} + \epsilon}.$$

6: Update each entry l of the superdiagonal $\boldsymbol{\lambda}$ using eq. (2.16):

$$\boldsymbol{\lambda}^{(t)}(l) = S_\mu\left(-\left\langle \bigcirc_{n=1}^{N} \boldsymbol{u}_{n,l}^{(t)}, \sum_{\substack{r=1 \\ r \neq l}}^{R_0} \boldsymbol{\lambda}^{(t)}(r) \bigcirc_{n=1}^{N} \boldsymbol{u}_{n,r}^{(t)} \right\rangle + \left\langle \mathcal{X} - \tilde{\mathcal{G}} \times_{n=1}^{N} \boldsymbol{U}_n^{(t)}, \bigcirc_{n=1}^{N} \boldsymbol{u}_{n,l}^{(t)} \right\rangle \right),$$

where S_μ is defined in eq. (2.17).
end for
Canonical rank: Number of non-zero values of the superdiagonal of $\mathcal{G}$.
CP factors: Columns of $\boldsymbol{U}_n^{(T)}$ with indices of non-zero values of the superdiagonal of $\mathcal{G}$.
Returns: [Canonical rank, CP factors].

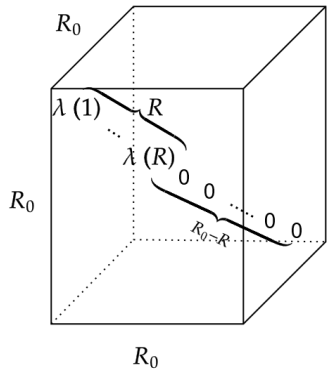


Figure 2.1 – CP core tensor $\mathcal{G}$ of a rank-R tensor of order 3 and size R_0 with $\boldsymbol{\lambda}(1) > \cdots > \boldsymbol{\lambda}(R) > 0$.

is given by:

$$\mathcal{L}_{\mathcal{G},\{\boldsymbol{U}_n\}} = \alpha_1 \|\boldsymbol{\lambda}\|_1 + \frac{\alpha_2}{2}\left(\left\|\mathcal{X} - \mathcal{G} \overset{N}{\underset{n=1}{\times_n}} \boldsymbol{U}_n\right\|_F^2 - \epsilon_1\right) + \frac{\alpha_3}{2}\left(\left\|\tilde{\mathcal{G}}\right\|_F^2 - \epsilon_2\right), \tag{2.13}$$

where α_2 and α_3 are two strictly positive Lagrange multipliers. According to the Lagrangian method [125], $\mathcal{L}_{\mathcal{G},\{\boldsymbol{U}_n\}}$ is minimized with respect to $\{\boldsymbol{U}_n\}_n$, $\tilde{\mathcal{G}}$ and $\boldsymbol{\lambda}$. To do that, we will proceed iteratively. We first minimize $\mathcal{L}_{\mathcal{G},\{\boldsymbol{U}_n\}}$ w.r.t the n-th CP factor $\boldsymbol{U}_n$ at each iteration, assuming the remaining factors and $\mathcal{G}$ are known. This is a classical linear regression problem. Following that, we derive $\mathcal{L}_{\mathcal{G},\{\boldsymbol{U}_n\}}$ w.r.t $\tilde{\mathcal{G}}$ using the factors $\{\boldsymbol{U}_n\}_n$ from the previous step. The convexity w.r.t $\tilde{\mathcal{G}}$ is demonstrated in Appendix .2. Then, we update $\tilde{\mathcal{G}}$ using the Adam optimizer [30]. Finally, we minimize $\mathcal{L}_{\mathcal{G},\{\boldsymbol{U}_n\}}$ w.r.t $\boldsymbol{\lambda}$, which is also a convex optimization problem as shown in Appendix .2. Its expression is given by the soft thresholding operator[123] using the current updates of $\tilde{\mathcal{G}}$ and $\{\boldsymbol{U}_n\}_n$. The final expressions of the solutions of the CP factors, the gradient of the Lagrangian w.r.t the $\tilde{\mathcal{G}}$ and the expression to update the $\boldsymbol{\lambda}$ are given below. Details of the computations can be found in Appendix .1.

- The update of the CP factors $\boldsymbol{U}_n$ is given by:

$$\boldsymbol{U}_n = \boldsymbol{X}_{(n)} \left(\boldsymbol{G}_{(n)} \left(\bigotimes_{\substack{l=N \\ l \neq n}}^{l=1} \boldsymbol{U}_l^T \right) \right)^{\dagger}, \tag{2.14}$$

  where † is the Moore-Penrose inverse.

- Gradient of $\mathcal{L}_{\mathcal{G},\{\boldsymbol{U}_n\}}$ w.r.t the offdiagonals of $\mathcal{G}$ *i.e* $\mathcal{G}(r_1, \ldots, r_N)$ such that $r_1 \neq r_2$ or … or $r_{N-1} \neq r_N$:

$$\left(\nabla_{\tilde{\mathcal{G}}}\left(\mathcal{L}_{\mathcal{G},\{\boldsymbol{U}_n\}}\right)\right)(r_1, \ldots, r_N) = -\alpha_2 \sum_{i_1=1}^{I_1} \cdots \sum_{i_N=1}^{I_N} \mathcal{A}(i_1, \ldots, i_N) \left[\prod_{n=1}^{N} \boldsymbol{U}_n(i_n, r_n) \right] + \alpha_3 \tilde{\mathcal{G}}(r_1, \ldots, r_N), \tag{2.15}$$

  where $\mathcal{A} = \mathcal{X} - \mathcal{G} \times_{n=1}^{N} \boldsymbol{U}_n$.

- Expression for updating the superdiagonal of $\mathcal{G}$:

$$\boldsymbol{\lambda}(l) = S_{\mu}\left(-\left\langle \bigcirc_{n=1}^{N} \boldsymbol{u}_{n,l}, \sum_{\substack{r=1 \\ r \neq l}}^{R_0} \boldsymbol{\lambda}(r) \bigcirc_{n=1}^{N} \boldsymbol{u}_{n,r} \right\rangle + \left\langle \mathcal{X} - \tilde{\mathcal{G}} \times_{n=1}^{N} \boldsymbol{U}_n, \bigcirc_{n=1}^{N} \boldsymbol{u}_{n,l} \right\rangle \right), \tag{2.16}$$

  where S_{μ} is the soft thresholding operator [123] of threshold $\mu > 0$ defined as follows for $x \in \mathbb{R}$:

$$S_{\mu}(x) = \begin{cases} x - \mu \text{ if } x > \mu, \\ 0 \text{ if } |x| \leq \mu, \\ x + \mu \text{ if } x < -\mu. \end{cases} \tag{2.17}$$

2.2.3 Algorithm FARAC

The CP factors are initialised with the left leading singular vectors of the unfoldings as in [135] while the offdiagonals of the TD core tensor $\mathcal{G}$ are randomly initialised. The Adam optimiser [30] is used to update the offdiagonals. Adam is an extended version of gradient descent. It updates exponential moving averages

of gradients and squared gradients using two hyperparameters η_1, η_2 (called exponential decay rates) to control their exponential decay rates averages. Basically, moving averages are estimates of the first moment (the mean) and the second moment of the gradient (the uncentered variance). As mentionned in [30], it is important to note, however, that these moving averages are intialized to 0 leading to moment estimates that are biased to 0 in particular in the first iterations and especially when decay rates are small (*i.e.* when the ηs are close to 1). As a result, bias-corrected data comes from overcoming the initialization bias. The whole algorithm of our derived approach is described in Algorithm 1.

2.2.4 Complexity Analysis

- We evaluate the complexity of Algorithm 1 by taking into consideration the SVDs in the initialization and the main parts of the algorithm, as the computations of the different gradients, the pseudo inverse of factors as well as the shrinkage operator. The complexity is evaluated as follows:

$$\mathcal{O}\left(\left(\sum_{n=1}^{N} I_n\right) R_0^2 + N_{\text{iter}}\left[N + \left(\sum_{n=1}^{N} I_n\right) R_0 + \left(\prod_{n=1}^{N} I_n\right)\left(N + R_0 + N R_0^{N-1}\right)\right]\right), \quad (2.18)$$

  where N_{iter} is the number of iterations.

- If $I_{\max} = \max(I_1, \ldots, I_N)$, then the complexity in eq. (2.18) becomes:

$$\mathcal{O}\left(N I_{\max} R_0^2 + N_{\text{iter}}\left(N + N I_{\max} R_0 + N I_{\max}(N + R_0 + N R_0^{N-1})\right)\right),$$

  which can be simplified to:

$$\mathcal{O}(N R_0^{N-1} + N_{\text{iter}} I_{\max} N^2 R_0^{N-1}).$$

2.3 Related works

Existing approaches for the canonical rank detection include:

- CORCONDIA [122] is a heuristic method for detecting the canonical rank

of a tensor $\mathcal{X}$. As a first step, the CP factors are computed with a fixed number of rank-one tensors. The factors obtained will be used to fit a TD in order to determine whether the tensor $\mathcal{X}$ can be adequately explained by these factors alone or by incorporating their interactions. The principle behind CORCONDIA is to assess the degree of similarity between the implicit superdiagonal tensor and the least squares fitted Tucker tensor using different values of the number of rank-one tensors, beginning with a rank-one tensor. Then, CORCONDIA calculates the similarity between the estimated core tensor and the ideal identity core (known as core consistency) for different CPDs . Based on the gap between the core consistency of different CPDs with different number of rank-one tensors, it determines the canonical rank. The optimal value for the canonical rank is determined by taking the CPD of the largest number of rank-one tensors whose core array remains similar to the ideal diagonal tensor.

- A fast version of CORCONDIA was presented in [107]. It suggests an efficient way to compute the CORCONDIA diagnostic that takes advantage of sparse data and works well as the tensor size grows. In cases where either the tensor, the factors or both are sparse [107], their algorithm significantly outperforms the state-of-the-art baselines and scales well when the tensor size increases. In the fully dense scenario, their proposed algorithm is as good as the state of the art (The CORCONDIA method) for rank estimation.

- Automatic Relevance Determination (ARD) [87] is a Bayesian approach applied to the TD and CPD. In ARD, entries of CP factors are assigned a Gaussian prior. The objective is to find the canonical rank and the CP factors by solving an l_2−regularized CPD. By assigning priors to the CP factors and learning the hyperparameters of these priors, ARD reduces the excess of rank-one tensors and simplifies the core structure.

- Tensor learning models for regression are proposed in [138]. In their regularization process, they suppose that the weight tensor of the regression problem has a low-rank CPD. They constrain the CP factors using the group-sparsity norm. This procedure gives an automatic selection of the

canonical rank during the learning procedure.

- By adding orthogonality constraints on the CP factors, [114] views the superdiagonal of the CP core as analogous to the vector of singular values. To determine the rank of an incomplete tensor, they add l_1 regularization on the superdiagonal of the tensor.

In short, state-of-the-art methods use either Bayesian approaches, propose rank estimation within a learning framework, or use too restricting constraints like factor orthogonality. In contrast to existing methods, the proposed method belongs to the family of deterministic parameter estimators.

2.4 Numerical Experiments

The Tensorflow framework [80] was used for the implementation of the Algorithm 1, where gradient computations are done using automatic differentiation. Numerical experiments involving CORCONDIA were conducted in Python using the CORCONDIA package [24].

SNR [db]	15	20	25	30	35	40
Accuracy [%]	72	77	87	91	96	100

Table 2.1 – Accuracy of FARAC w.r.t SNR for a tensor of a noisy tensor $\mathcal{X}_{\text{noise}}$ of size $5 \times 5 \times 5$. The true rank is $R = 6$ and $R_0 = 7$.

2.4.1 Evaluation metrics

- The convergence of the proposed method will be evaluated in terms of the RSE at each iteration t, which is given by:

$$\frac{\left\|\mathcal{X}_{\text{noise}} - \mathcal{X}^{(t)}\right\|_F}{\|\mathcal{X}_{\text{noise}}\|_F}.$$

- The accuracy of rank estimation for synthetic data will be given by the proportion of realizations where the canonical rank is well estimated.

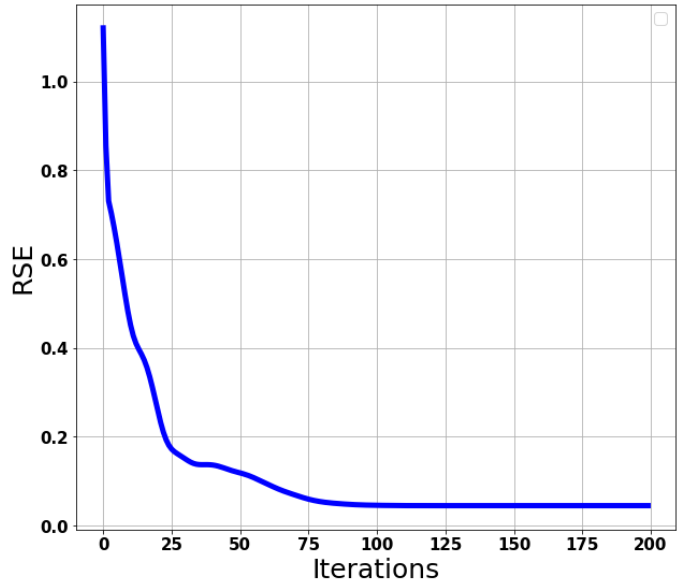


Figure 2.2 – Convergence curve of the mean reconstruction error using the RSE along iterations using FARAC. We used a noisy tensor $\mathcal{X}_{\text{noise}}$ with a size of $5 \times 5 \times 5$ and an SNR of $25db$. The threshold parameter is equal to 0.02.

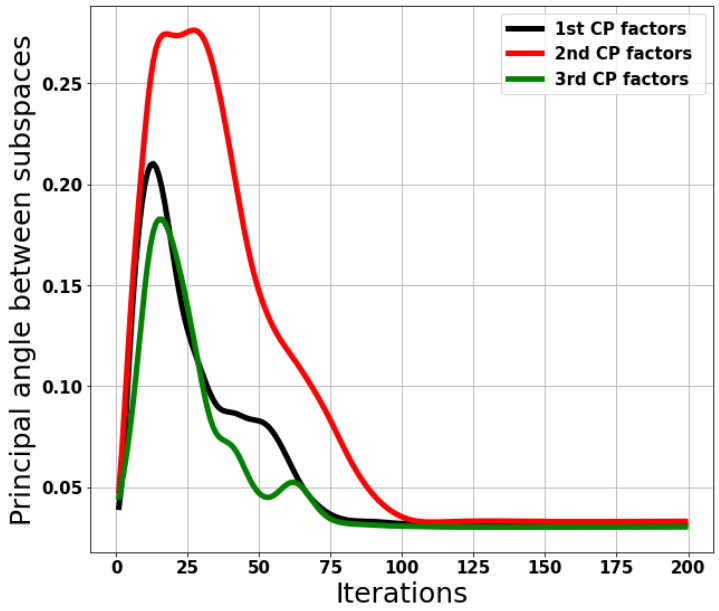


Figure 2.3 – The mean principal angle between the three subspaces spanned by CP factors along iterations using our method. We used a noisy tensor $\mathcal{X}_{\text{noise}}$ of size $5 \times 5 \times 5$ and SNR of $25db$. The threshold parameter is equal to 0.02.

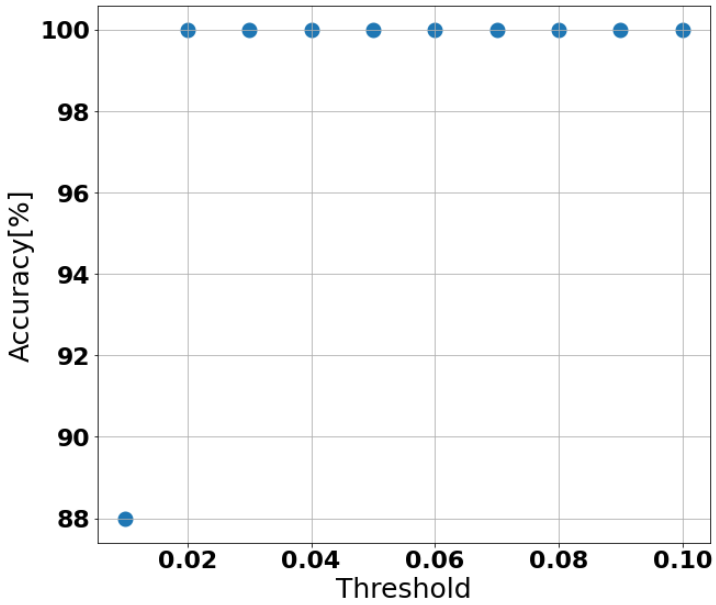


Figure 2.4 – Accuracy of rank estimation of a tensor of size $5 \times 5 \times 5$ w.r.t the threshold parameter. The true rank is $R = 2$ and $R_0 = 5$.

- The recovery of the CP factors will be evaluated by checking the subspace error. This can be done by computing the largest principal angle [127] between the true subspaces and the estimated ones at each iteration t.

$$\theta^{(t)} = \arcsin\left(\left\|\boldsymbol{U}_n\boldsymbol{U}_n^{\dagger} - \hat{\boldsymbol{U}}_n^{(t)}\hat{\boldsymbol{U}}_n^{(t)\dagger}\right\|_2\right),$$

  where $\boldsymbol{U}_n$ and $\hat{\boldsymbol{U}}_n^{(t)}$ are the exact and the estimated factors at iteration t, respectively.

2.4.2 Synthetic data

Data generation

We create a synthetic rank-R real valued tensor $\mathcal{X}$ of order N from its CPD. The CP factors are derived from a single realization of the normal standard distribution. $\mathcal{B}$ is a zero mean, unit variance and white noise. Our noisy data tensor is given

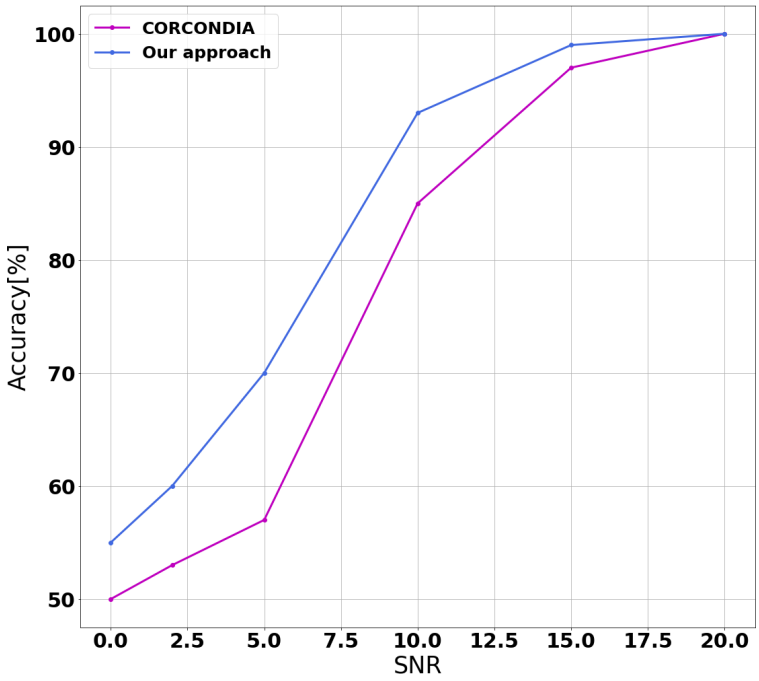


Figure 2.5 – FARAC Vs. CORCONDIA accuracy w.r.t SNR for a tensor of a noisy tensor $\mathcal{X}_{\text{noise}}$ with SNR values ranging from 0db to 20db. $\mathcal{X}_{\text{noise}}$ is of size $5 \times 5 \times 5$. Different CP models with a rank ranging from 1 to 5 are used to fit CORCONDIA. $R = 2$ is the true rank.

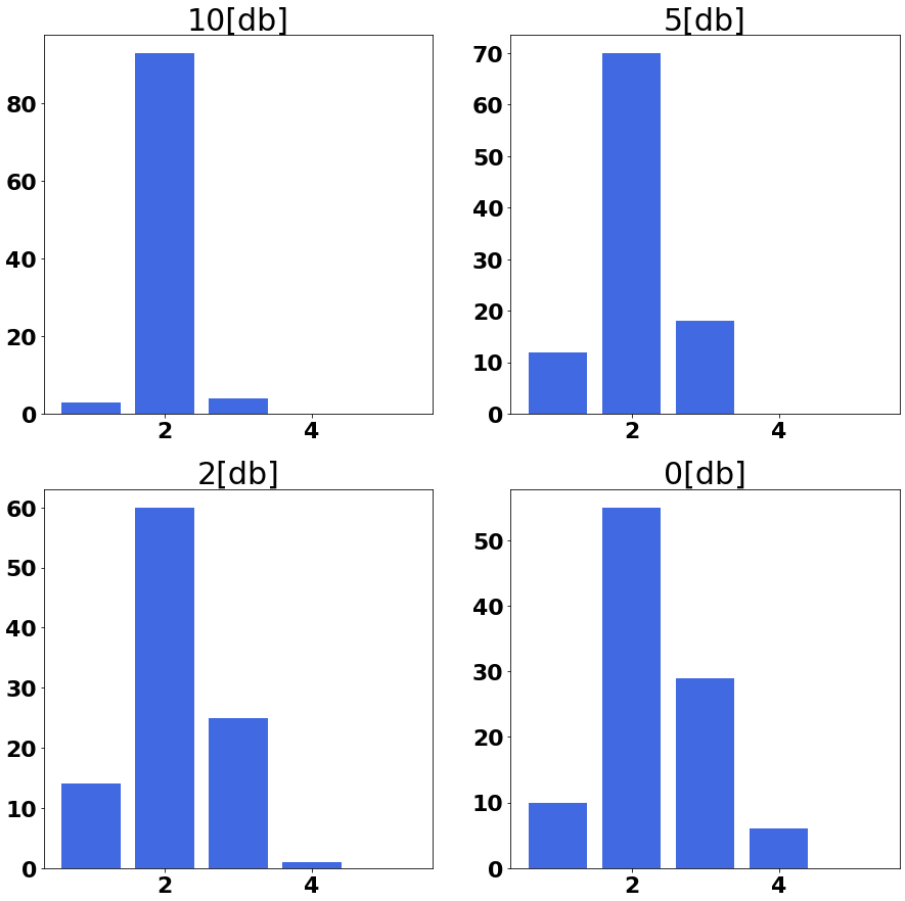


Figure 2.6 – Accuracy of rank estimation of a noisy tensor $\mathcal{X}_{\text{noise}}$ with different low SNR values ranging from 0db to 10db using FARAC. The size of $\mathcal{X}_{\text{noise}}$ is $5 \times 5 \times 5$. The used large bound of rank used is $R_0 = 5$, while the true rank is $R = 2$.

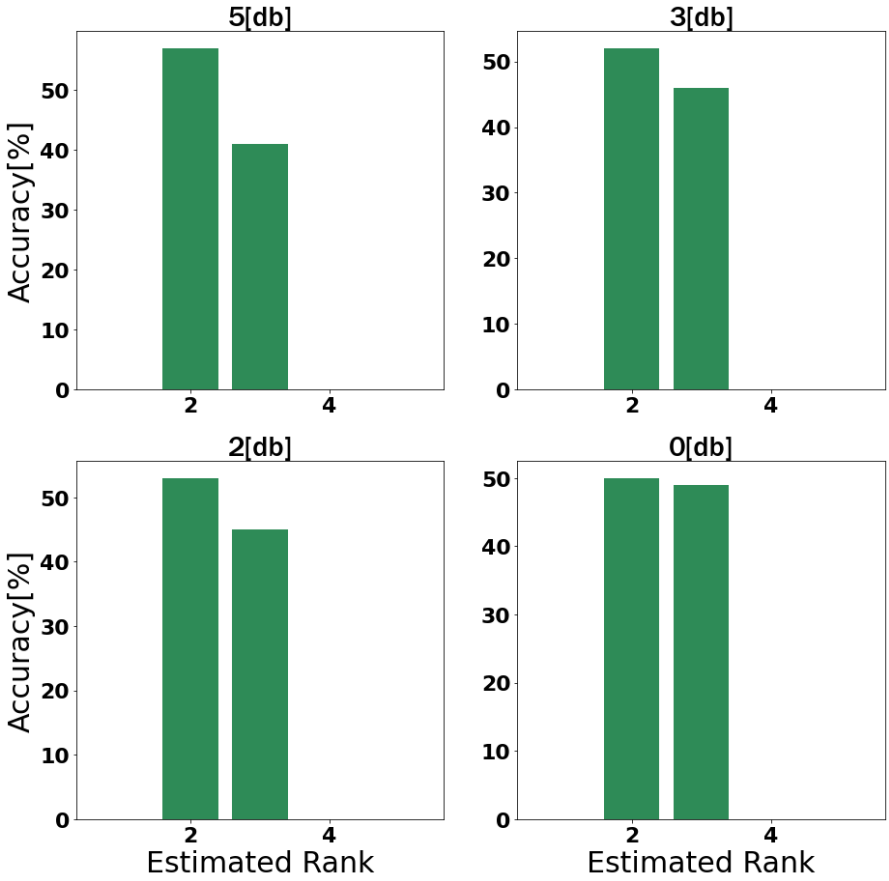


Figure 2.7 – The CORCONDIA approach's accuracy for a noisy tensor $\mathcal{X}_{\text{noise}}$. SNR values range from 0db to 5db. The size of $\mathcal{X}_{\text{noise}}$ is $5 \times 5 \times 5$. Different CP models with a rank ranging from 1 to 5 are used to fit CORCONDIA. $R = 2$ is the true rank.

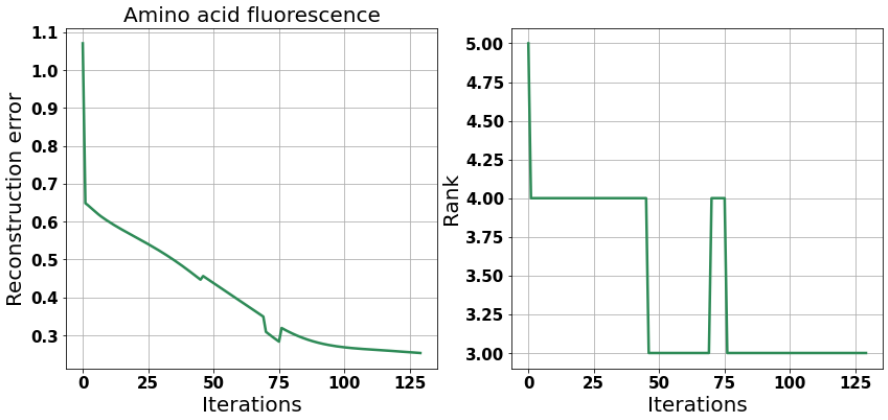


Figure 2.8 – Convergence loss of FARAC (left) and rank estimation (right) on the amino acid dataset over iterations .

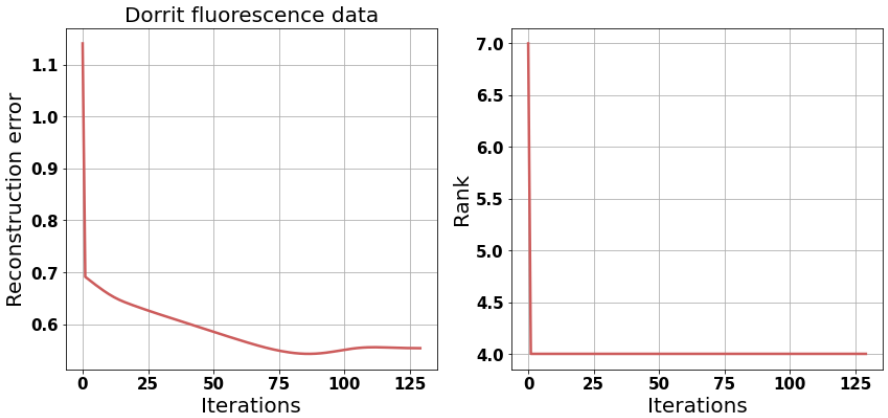


Figure 2.9 – Convergence loss of FARAC (left) and rank estimation (right) on the amino acid dataset over iterations .

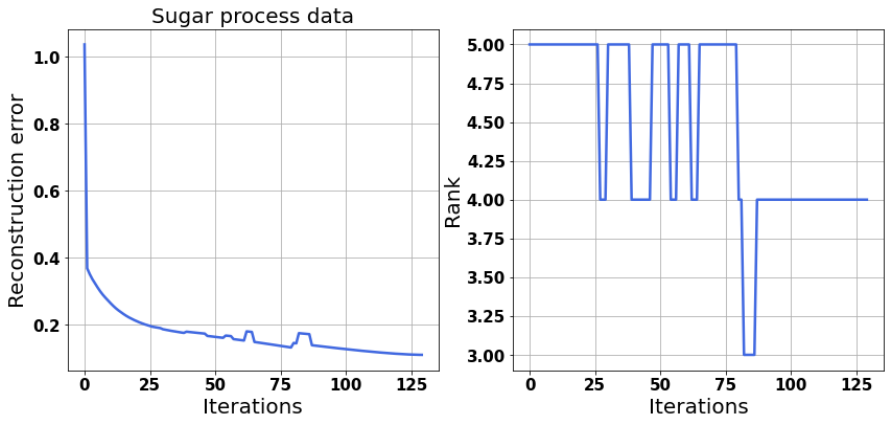


Figure 2.10 – Convergence loss of FARAC (left) and rank estimation (right) on the sugar process dataset over iterations .

by:

$$\mathcal{X}_{\text{noise}} = \mathcal{X}' + \sigma\mathcal{B}', \tag{2.19}$$

where $\mathcal{X}' = \frac{\mathcal{X}}{\|\mathcal{X}\|_F}$ and $\mathcal{B}' = \frac{\mathcal{B}}{\|\mathcal{B}\|_F}$. Hence, the SNR will be calculated using the following formula:

$$SNR = -10\log_{10}\sigma^2 \in [0\,db, 40\,db].$$

FARAC has been run on a rank-2 tensor $\mathcal{X}$ with size $I \times I \times I$ where $R_0 = I = 5$. Experiments are conducted on a tensor with these parameters until other settings are indicated. Similar results are obtained for tensors with other orders and sizes. By using $\mathcal{X}$ and eq. (2.19), we generate 100 noisy realisations of the input tensor.

Accuracy of rank estimation

- The accuracy of rank estimation using the proposed approach is illustrated in Fig. 2.4. We can clearly see that the FARAC approach allows to estimate the exact canonical rank while being robust to the choise of the threshold parameter.

- Fig. 2.2 depicts the convergence curves of the proposed method. We can

see that the reconstruction error quickly decreases to zero w.r.t iterations.

- The recovery of the CP factors is shown in Fig. 2.3 by checking the subspace error. We see that the mean angle between the subspaces of the three CP factors converges to a low value with respect to iterations.

- In Fig. 2.5, we compare the FARAC method with CORCONDIA with respect to SNR values. We can see that FARAC clearly outperforms the state-of-the-art method.

- In Table 2.1, we show the accuracy of the rank estimation using FARAC when the true rank exceeds one of the dimensions of the input tensor. As we can see, FARAC can handle this difficult scenario for a large range of SNR values. The CORCONDIA approach, on the other hand, is ineffective in that situation [60].

- For low SNR values, accuracy of having a rank 3 instead of 2 increases but FARAC still gives the highest accuracy for the true rank according to Fig. 2.6. In contrast, rank estimation using the CORCONDIA method becomes a difficult task at low SNR according to Fig. 2.7.

Compared with the state-of-the-art method, FARAC shows very good performance in terms of the accuracy of rank estimation for large range of values of SNRs while being robust to the choise of the threshold parameter. FARAC also handles the difficult case where the rank is bigger than one of the dimensions of the input tensor.

2.4.3 Real datatsets

Datasets

- **Amino acid fluorescence**: As described in [118], the dataset includes the excitation and emission spectra of five samples of different concentrations of tyrosine, tryptophane and phenylalanine, forming a tensor of 5 (samples) $\times$ 51 (excitation) $\times$ 201 (emission) of canonical rank equal to 3.

- **Sugar process data** [117]: The dataset contains 265 samples that can be arranged in an $I_1 I_2 I_3$ three-order tensor of size $265 \times 571 \times 7$. The first mode relates to samples, the second to emission wavelengths, and the third to excitation wavelengths. The $(i_1 i_2 i_3)$-th element of this tensor, $\mathcal{X}(i_1, i_2, i_3)$, represents the measured emission intensity from sample i_1, excited at wavelength i_3, and measured at wavelength i_2. The canonical rank of the tensor $\mathcal{X}$ is equal to 4.

- **Dorrit fluorescence data** [54]: Fluorescence spectrometer was used to measure 27 synthetic samples containing different concentrations of four analytes (hydroquinone, tryptophan, phenylalanine and dopa). Each fluorescence landscape corresponds to 233 emission wavelengths and 24 excitation wavelengths. This dataset is represented with a tensor of size $27 \times 233 \times 24$ and its cannonical rank is equal to 4.

Accuracy of rank estimation

- In Fig. 2.8, 2.9 and 2.10, we present the convergence loss of FARAC, as well as the canonical rank estimation over iterations on the amino acid fluorescence, the dorrit fluorescence and the sugar process datasets. As shown in these figures, the canonical rank is well estimated for the three real datasets. For the amino and dorrit fluorescence datasets, the threshold parameter μ used is equal to 0.01; for the sugar process dataset, it is equal to 0.001.

Hyperparameters settings

Based on the noise level, one can use a grid search on the threshold parameter μ over the range of values $[0.1, 0.01, 0.001]$. It is important to note that the FARAC method is robust to the choice of the threshold as shown in Fig. 2.4 The learning rate and the exponential decay rates used in the Adam optimizer are all equal to their default values that are good default settings for different optimization problems [30] ($\eta_1 = 0.09, \eta_2 = 0.0999, \eta_3 = 0.001$).

2.5 Conclusion

We have addressed the challenging problem of estimating the canonical rank in this study. With the proposed method, called FARAC, both canonical rank and CP factors are estimated jointly. Our proposed method is shown to be effective in estimating the canonical rank for large values of SNR through different experiments. We have compared FARAC with the well-known CORCONDIA method and found that FARAC is more accurate especially when dealing with high-level noise. In addition, we have demonstrated that FARAC shows strong robustness to the choice of the threshold parameter and that it is capable of handling the difficult case of rank exceeding one dimension of the tensor, unlike CORCONDIA. Lastly, we demonstrate the validity of the FARAC method on real datasets.

Appendices

.1 Detail computations for the derivation of FARAC

Let us recall the Lagrangien of the problem (2.12) that we want to minimize w.r.t $(\boldsymbol{U}_n)_n$, $\tilde{\mathcal{G}}$ and $\boldsymbol{\lambda}$:

$$\mathcal{L}_{\mathcal{G},\{\boldsymbol{U}_n\}} = \alpha_1\|\boldsymbol{\lambda}\|_1 + \frac{\alpha_2}{2}\left(\left\|\boldsymbol{\mathcal{X}} - \mathcal{G}\overset{N}{\underset{n=1}{\times_n}}\boldsymbol{U}_n\right\|_F^2 - \epsilon_1\right) + \frac{\alpha_3}{2}\left(\left\|\tilde{\mathcal{G}}\right\|_F^2 - \epsilon_2\right),$$

- We denote by $\mathcal{L}_{\boldsymbol{U}_n}$, the part of $\mathcal{L}_{\mathcal{G},\{\boldsymbol{U}_n\}}$ which depends only on $\boldsymbol{U}_n$.

$$\mathcal{L}_{\boldsymbol{U}_n} = \frac{\alpha_2}{2}\left\|\mathcal{X} - \mathcal{G}\overset{N}{\underset{n=1}{\times_n}}\boldsymbol{U}_n\right\|_F^2. \tag{20}$$

  The matricized form of eq. (20) is given by:

$$\mathcal{L}_{\boldsymbol{U}_n} = \frac{\alpha_2}{2}\left\|\boldsymbol{X}_{(n)} - \boldsymbol{U}_n\boldsymbol{G}_{(n)}\left(\bigotimes_{\substack{l=N\\ l\neq n}}^{l=1}\boldsymbol{U}_l^T\right)\right\|_F^2.$$

  The optimal solution is given by:

$$\boldsymbol{U}_n = \boldsymbol{X}_{(n)}\left(\boldsymbol{G}_{(n)}\left(\bigotimes_{\substack{l=N\\ l\neq n}}^{l=1}\boldsymbol{U}_l^T\right)\right)^{\dagger}.$$

- The part of $\mathcal{L}_{\mathcal{G},\{\boldsymbol{U}_n\}}$ that depends only on $\tilde{\mathcal{G}}$ is denoted by $\mathcal{L}_{\tilde{\mathcal{G}}}$:

$$\mathcal{L}_{\tilde{\mathcal{G}}} = \frac{\alpha_2}{2}\sum_{i_1,\ldots,i_N}\underbrace{\left(\mathcal{X}(i_1,\ldots,i_N) - \left(\mathcal{G}\overset{N}{\underset{n=1}{\times_n}}\boldsymbol{U}_n\right)(i_1,\ldots,i_N)\right)^2}_{\mathcal{A}^2(i_1,\ldots,i_N)} + \frac{\alpha_3}{2}\sum_{r_1,\ldots,r_N}\tilde{\mathcal{G}}^2(r_1,\ldots,r_N).$$

  Let us derive $\mathcal{L}_{\tilde{\mathcal{G}}}$ with respect to $\tilde{\mathcal{G}}(r_1,\ldots,r_N)$ such that $r_1 \neq r_2$ or … or

$r_{N-1} \neq r_N$ (diagonal elements are excluded since they are equal to 0):

$$[\nabla_{\tilde{\mathcal{G}}}\mathcal{L}_{\tilde{\mathcal{G}}}](r_1,\ldots,r_N) = -\alpha_2 \sum_{i_1,\ldots,i_N} \mathcal{A}(i_1,\ldots,i_N)\left(\prod_{n=1}^{N} \boldsymbol{U}_n(i_n,r_n)\right) + \alpha_3\tilde{\mathcal{G}}(r_1,\ldots,r_N). \tag{21}$$

- We want to derive $\mathcal{L}_{\mathcal{G},\{\boldsymbol{U}_n\}}$ w.r.t $\boldsymbol{\lambda}$. Let us first rewrite it as follows:

$$\begin{aligned}\mathcal{L}_{\mathcal{G},\{\boldsymbol{U}_n\}} &= \alpha_1\|\boldsymbol{\lambda}\|_1 + \frac{\alpha_2}{2}\left\|\mathcal{X} - \left(\tilde{\mathcal{G}} + diag(\boldsymbol{\lambda})\right)\overset{N}{\underset{n=1}{\times}}_n \boldsymbol{U}_n\right\|_F^2 + \frac{\alpha_3}{2}\left\|\tilde{\mathcal{G}}\right\|_F^2,\\ &= \alpha_1\|\boldsymbol{\lambda}\|_1 + \frac{\alpha_2}{2}\left\|\left(\mathcal{X} - \tilde{\mathcal{G}}\overset{N}{\underset{n=1}{\times}}_n \boldsymbol{U}_n\right) - \sum_{r=1}^{R_0}\boldsymbol{\lambda}(r)\overset{N}{\underset{n=1}{\bigcirc}}\boldsymbol{u}_{n,r}\right\|_F^2 + \frac{\alpha_3}{2}\left\|\tilde{\mathcal{G}}\right\|_F^2.\end{aligned}$$

- We denote by $\mathcal{L}_{\boldsymbol{\lambda}}$, the part of $\mathcal{L}_{\mathcal{G},\{\boldsymbol{U}_n\}}$ which depends only on $\boldsymbol{\lambda}$. $\mathcal{L}_{\boldsymbol{\lambda}}$ is given as follows:

$$\begin{aligned}\mathcal{L}_{\boldsymbol{\lambda}} = \alpha_1\|\boldsymbol{\lambda}\|_1 + \frac{\alpha_2}{2}\Bigg(&\left\|\sum_{r=1}^{R_0}\boldsymbol{\lambda}(r)\overset{N}{\underset{n=1}{\bigcirc}}\boldsymbol{u}_{n,r}\right\|_F^2 - 2\left\langle\mathcal{X} - \tilde{\mathcal{G}}\overset{N}{\underset{n=1}{\times}}_n \boldsymbol{U}_n, \sum_{r=1}^{R_0}\boldsymbol{\lambda}(r)\overset{N}{\underset{n=1}{\bigcirc}}\boldsymbol{u}_{n,r}\right\rangle\\ &+ \left\|\mathcal{X} - \tilde{\mathcal{G}}\overset{N}{\underset{n=1}{\times}}_n \boldsymbol{U}_n\right\|_F^2\Bigg),\\ = \alpha_1\left(|\boldsymbol{\lambda}(l)| + \sum_{r\neq l}|\boldsymbol{\lambda}(r)|\right) + \frac{\alpha_2}{2}\Bigg(&\left\|\sum_{\substack{r=1\\r\neq l}}^{R_0}\boldsymbol{\lambda}(r)\overset{N}{\underset{n=1}{\bigcirc}}\boldsymbol{u}_{n,r}\right\|_F^2 + \boldsymbol{\lambda}(l)^2\left\|\overset{N}{\underset{n=1}{\bigcirc}}\boldsymbol{u}_{n,l}\right\|_F^2\\ &+ 2\left\langle\boldsymbol{\lambda}(l)\overset{N}{\underset{n=1}{\bigcirc}}\boldsymbol{u}_{n,l}, \sum_{\substack{r=1\\r\neq l}}^{R_0}\boldsymbol{\lambda}(r)\overset{N}{\underset{n=1}{\bigcirc}}\boldsymbol{u}_{n,r}\right\rangle,\\ - 2\left\langle\mathcal{X} - \tilde{\mathcal{G}}\overset{N}{\underset{n=1}{\times}}_n \boldsymbol{U}_n, \sum_{\substack{r=1\\r\neq l}}^{R_0}\boldsymbol{\lambda}(r)\overset{N}{\underset{n=1}{\bigcirc}}\boldsymbol{u}_{n,r}\right\rangle - 2\left\langle\mathcal{X} - \tilde{\mathcal{G}}\overset{N}{\underset{n=1}{\times}}_n \boldsymbol{U}_n, \boldsymbol{\lambda}(l)\overset{N}{\underset{n=1}{\bigcirc}}\boldsymbol{u}_{n,l}\right\rangle +&\\ \left\|\mathcal{X} - \tilde{\mathcal{G}}\overset{N}{\underset{n=1}{\times}}_n \boldsymbol{U}_n\right\|_F^2\Bigg).&\end{aligned} \tag{22}$$

Let us derive $\mathcal{L}_\lambda$ with respect to $\lambda(l)$ and set it to 0.

$$\begin{aligned}
&\frac{\partial(\mathcal{L}_\lambda)}{\partial\lambda(l)} = 0,\\
&\Leftrightarrow \alpha_1\frac{\partial|\lambda(l)|}{\partial\lambda(l)}+\alpha_2\left(\lambda(l)\left\|\bigcirc_{n=1}^{N}\boldsymbol{u}_{n,l}\right\|_F^2+\left\langle\bigcirc_{n=1}^{N}\boldsymbol{u}_{n,l},\sum_{\substack{r=1\\r\neq l}}^{R_0}\lambda(r)\bigcirc_{n=1}^{N}\boldsymbol{u}_{n,r}\right\rangle-\left\langle\mathcal{X}-\tilde{\mathcal{G}}\times_{n=1}^{N}\boldsymbol{U}_n,\bigcirc_{n=1}^{N}\boldsymbol{u}_{n,l}\right\rangle\right)=0,\\
&\Leftrightarrow \lambda(l)+\frac{\alpha_1}{\alpha_2\|\bigcirc_{n=1}^{N}\boldsymbol{u}_{n,l}\|^2}\frac{\partial|\lambda(l)|}{\partial\lambda(l)} = -\alpha_2\left\langle\bigcirc_{n=1}^{N}\boldsymbol{u}_{n,l},\sum_{\substack{r=1\\r\neq l}}^{R_0}\lambda(r)\bigcirc_{n=1}^{N}\boldsymbol{u}_{n,r}\right\rangle+\alpha_2\left\langle\mathcal{X}-\tilde{\mathcal{G}}\times_{n=1}^{N}\boldsymbol{U}_n,\bigcirc_{n=1}^{N}\boldsymbol{u}_{n,l}\right\rangle.
\end{aligned} \tag{23}$$

Furthermore, we have the following:

$$\left\|\bigcirc_{n=1}^{N}\boldsymbol{u}_{n,l}\right\|^2 = \prod_{n=1}^{N}\left\|\boldsymbol{u}_{n,l}\right\|^2.$$

Since $\boldsymbol{u}_{n,l}$ are unit vectors, we have $\left\|\bigcirc_{n=1}^{N}\boldsymbol{u}_{n,l}\right\|^2 = 1$. Hence we find the following:

$$\lambda(l)+\frac{\alpha_1}{\alpha_2}\frac{\partial|\lambda(l)|}{\partial\lambda(l)} = -\left\langle\bigcirc_{n=1}^{N}\boldsymbol{u}_{n,l},\sum_{\substack{r=1,\\r\neq l}}^{R_0}\lambda(r)\bigcirc_{n=1}^{N}\boldsymbol{u}_{n,r}\right\rangle+\left\langle\mathcal{X}-\tilde{\mathcal{G}}\times_{n=1}^{N}\boldsymbol{U}_n,\bigcirc_{n=1}^{N}\boldsymbol{u}_{n,l}\right\rangle.$$

As a result, the value of $\lambda(l)$ can be computed using the soft thresholding operator S_μ [123]:

$$\lambda(l) = S_\mu\left(-\left\langle\bigcirc_{n=1}^{N}\boldsymbol{u}_{n,l},\sum_{\substack{r=1\\r\neq l}}^{R_0}\lambda(r)\bigcirc_{n=1}^{N}\boldsymbol{u}_{n,r}\right\rangle+\left\langle\mathcal{X}-\tilde{\mathcal{G}}\times_{n=1}^{N}\boldsymbol{U}_n,\bigcirc_{n=1}^{N}\boldsymbol{u}_{n,l}\right\rangle\right).$$

.2 Convexity

In this section, we will demonstrate that the Lagrangien in eq. (22) is convex w.r.t to $\boldsymbol{\lambda}$ and $\tilde{\mathcal{G}}$ so that the global minimum will be reached. To do that, we will show that the hessian w.r.t $\boldsymbol{\lambda}$ and $\tilde{\mathcal{G}}$ are positive semi-definite matrices.

- Convexity w.r.t $\boldsymbol{\lambda}$:
  Let $H_{\boldsymbol{\lambda}}$ be the hessian of $\mathcal{L}_{\mathcal{G},\{\mathbf{U}_n\}}$ w.r.t $\boldsymbol{\lambda}$. Using the gradient computed in eq. (23), we have:

$$H_{\boldsymbol{\lambda}}(s,l) := \frac{\partial^2 \mathcal{L}_{\mathcal{G},\{\mathbf{U}_n\}}}{\partial \boldsymbol{\lambda}(s)\partial \boldsymbol{\lambda}(l)} = \alpha_2 \left\langle \bigcirc_{n=1}^{N} \boldsymbol{u}_{n,l}, \bigcirc_{n=1}^{N} \boldsymbol{u}_{n,s} \right\rangle.$$

  Let $x \in \mathbb{R}^{R_0}$,

$$\begin{aligned}
\boldsymbol{x}^T \boldsymbol{H}_{\boldsymbol{\lambda}} \boldsymbol{x} &= \alpha_2 \sum_{l,s} \boldsymbol{x}(l)\boldsymbol{x}(s) \left\langle \bigcirc_{n=1}^{N} \boldsymbol{u}_{n,l}, \bigcirc_{n=1}^{N} \boldsymbol{u}_{n,s} \right\rangle, \\
&= \alpha_2 \left\langle \sum_{l} \boldsymbol{x}(l) \bigcirc_{n=1}^{N} \boldsymbol{u}_{n,l}, \sum_{s} \boldsymbol{x}(s) \bigcirc_{n=1}^{N} \boldsymbol{u}_{n,s} \right\rangle, \\
&= \alpha_2 \left\| \sum_{l} \boldsymbol{x}(l) \bigcirc_{n=1}^{N} \boldsymbol{u}_{n,l} \right\|^2 \geq 0,
\end{aligned}$$

  since α_2 is a positive Lagrange multiplier.

- Convexity w.r.t $\tilde{\mathcal{G}}$:
  We first place the elements $\tilde{\mathcal{G}}(r_1,\ldots,r_N)$ in a vector $\boldsymbol{x} \in \mathbb{R}^{{R_0}^N - R_0}$ and use the same method as for $\boldsymbol{\lambda}$.
  Let $H_{\tilde{\mathcal{G}}}$ be the hessian of $\mathcal{L}_{\mathcal{G},\{\mathbf{U}_n\}}$ w.r.t $\tilde{\mathcal{G}}$. Using eq.(21), we compute the second derivates of $\mathcal{L}_{\mathcal{G},\{\mathbf{U}_n\}}$. The following derivatives are done only on the offdiagonals of $\tilde{\mathcal{G}}$ since its diagonal is zero.

$$\begin{aligned}
H_{\tilde{\mathcal{G}}}(r_1 \ldots r_N, r'_1 \ldots r'_N) &:= \frac{\partial^2 \mathcal{L}_{\mathcal{G},\{\mathbf{U}_n\}}}{\partial \tilde{\mathcal{G}}(r_1,\ldots,r_N)\partial \tilde{\mathcal{G}}(r'_1,\ldots,r'_N)}, \\
&= \alpha_2 \sum_{i_1,\ldots,i_N} \left[\prod_{n=1}^{N} \mathbf{U}_n(i_n, r'_n) \right] \left[\prod_{n=1}^{N} \mathbf{U}_n(i_n, r_n) \right] + \alpha_3.
\end{aligned}$$

Let $\boldsymbol{x} \in \mathbb{R}^{{R_0}^N - R_0}$,

$$\boldsymbol{x}^T \boldsymbol{H}_{\tilde{\mathcal{G}}} \boldsymbol{x} = \alpha_2 \sum_{i_1,\ldots,i_N} \underbrace{\sum_{r_n} \left[\prod_{n=1}^{N} \boldsymbol{U}_n(i_n, r_n) \right]}_{\mathcal{Z}(i_1,\ldots,i_N)} \underbrace{\sum_{r'_n} \left[\prod_{n=1}^{N} \boldsymbol{U}_n(i_n, r'_n) \right]}_{\mathcal{Z}(i_1,\ldots,i_N)} + \alpha_3 \boldsymbol{x}^T \boldsymbol{x},$$

$$= \alpha_2 \|\mathcal{Z}\|_F^2 + \alpha_3 \|\boldsymbol{x}\|^2 \geq 0,$$

since α_2 and α_3 are strictly positive values.

Part II

Tensors and Learning

Chapter 3

Supervised Learning and Tensors

Outline of the current chapter

3.1 Introduction

Many machine learning tasks are carried out using supervised learning on labeled data. Classification is a major task in supervised learning. The solid foundation of SVM makes it suitable for a wide range of applications. However, SVM cannot be used for nonlinear classification. In real-life scenarios, kernels are

often helpful in dealing with such cases.
This chapter provides some preliminary information regarding kernel methods, beginning with kernel SVM. By mapping input data to high-dimensional Hilbert spaces, these kernel methods appear to extend SVM to non-linearly separable data. Then, we will discuss kernels based on Grassmann manifolds, and finally, present how SVM were extended to tensorial data, along with some methods that propose kernels based on tensor decompositions.

3.2 Support Vector Machines (SVMs)

3.2.1 Linear SVMs classifier

Problem setting

Let consider the following set of training data composed of M vectors $\boldsymbol{x}_m \in \mathbb{R}^I$ with labels $y_m \in \{1, -1\}$:

$$D_{\text{train}} = \{(\boldsymbol{x}_1, y_1), (\boldsymbol{x}_2, y_1), \ldots, (\boldsymbol{x}_M, y_M),\ \ \boldsymbol{x}_m,\ \ y_m \in \{1, -1\}\}.$$

Assume that D_{train} is linearly separable, meaning that there is a hyperplane that separates the two classes of data (Vectors with labels 1 and vectors with labels -1). The equation for this hyperplane is:

$$h(\boldsymbol{x}) := \boldsymbol{w}^T \boldsymbol{x} + b = 0, \tag{3.1}$$

where $\boldsymbol{w}$ is a normal vector to the hyperplane and $\frac{b}{\|\boldsymbol{w}\|}$ is the offset of the hyperplane from the origin.
Based on the assumption that all points with the label $y_m = 1$ lie on the positive side of the hyperplane ($h(\boldsymbol{x}) \geq 0$) and all points with the label $y_m = -1$ lie on the negative side of the hyperplane, it follows that:

$$y_m h(\boldsymbol{x}_m) \geq 0, \quad 1 \leq m \leq M. \tag{3.2}$$

Maximizing the margin

The basic idea behind SVM is to choose the hyperplane that maximizes the margin, which is defined as the minimum distance between the hyperplane and the training points. The support vectors are those located at this minimal distance. Choosing a hyperplane maximizing margins is the best choice for predicting good classification while dealing with unseen data [133].
Even though the optimal hyperplane is unique, there exist infinite couples $(\boldsymbol{w}, b)$ that describe it. Thus, $(\boldsymbol{w}, b)$ are determined classically such that the distance between the support vectors and the hyperplane is equal to 1. As a result, for any point that is not a support vector, $y_m h(\boldsymbol{x}_m) > 1$, since it is, by definition, further from the hyperplane, and the margin is equal to $\frac{1}{\|\boldsymbol{w}\|}$ [1]. Thus, we have the following inequalities over all points in the dataset:

$$y_m h(\boldsymbol{x}_m) \geq 1, \text{ for all } \boldsymbol{x}_m \in D_{\text{train}}. \tag{3.3}$$

The margin can also be seen as the distance between the hyperplanes of equations $h(\boldsymbol{x}) = 1$ and $h(\boldsymbol{x}) = -1$.

Optimization problem of the SVM:

The SVM's goal is to find the hyperplane that maximizes the margin $\frac{1}{\|\boldsymbol{w}\|}$ under the constraints (3.3). Since maximizing the margin is equivalent to minimizing $\|\boldsymbol{w}\|$ or $\frac{1}{2}\|\boldsymbol{w}\|^2$, the standard formulation of the optimization problem of SVM is given by [94]:

$$\begin{cases} \underline{\text{Objective function}} : \min\limits_{\boldsymbol{w},b} \dfrac{1}{2}\|\boldsymbol{w}\|^2, \\ \underline{\text{Linear constraints}} : \; y_m h(\boldsymbol{x}_m) \geq 1, \;\; \forall \boldsymbol{x}_m \in D_{\text{train}}. \end{cases} \tag{3.4}$$

To solve the optimization problem in eq. (3.4), it is very common to solve its dual. The main idea is to introduce the lagrangian multiplier α_m using the

Karush-Kuhn-Tucker (KKT) conditions:

$$\alpha_m \left(y_m h(\boldsymbol{x}_m) - 1\right) = 0, \tag{3.5}$$

$$\alpha_m \geq 0. \tag{3.6}$$

By minimizing the lagrangian of eq.(3.4) according to $\boldsymbol{w}, b$, the following equations can be derived [94]:

$$\boldsymbol{w} = \sum_{m=1}^{M} \alpha_m y_m \boldsymbol{x}_m, \tag{3.7}$$

$$\sum_{m=1}^{M} \alpha_m y_m = 0. \tag{3.8}$$

The lagrangian variables are then obtained by solving the following maximization problem known as the dual problem [94]:

$$\begin{cases} \underline{\text{Objective function}} : \max\limits_{\alpha_m} \sum\limits_{m=1}^{M} \alpha_m - \dfrac{1}{2} \sum\limits_{m_1=1}^{M} \sum\limits_{m_2=1}^{M} \alpha_{m_1} \alpha_{m_2} y_{m_1} y_{m_2} \boldsymbol{x}_{m_1}^T \boldsymbol{x}_{m_2}, \\ \underline{\text{Linear constraints}} : \sum\limits_{m=1}^{M} \alpha_m y_m = 0. \end{cases} \tag{3.9}$$

Once the α_m are computed, $\boldsymbol{w}$ can be determined using eq.(3.7) . Using eq.(3.5) for a training point $(\boldsymbol{x}_{m_0}, y_{m_0}) \in D_{\text{train}}$, b will be equal to :

$$b = y_{m_0} - \sum_{\alpha_m > 0} \alpha_m y_m \langle \boldsymbol{x}_m, \boldsymbol{x}_{m_0} \rangle. \tag{3.10}$$

Decision function

In order to classify a new data point $\boldsymbol{z}$, its label is calculated based on the decision function defined as follows:

$$f(\boldsymbol{z}) := \text{sgn}(h(\boldsymbol{z})). \tag{3.11}$$

Slack variables

Due to noise, a perfectly linear separation may not be possible. It is possible that some outliers violate the margin of error. In such a case, SVM will allow some training points to violate the constraints, but these violations will be penalized. Thus, SVM introduces a slack variable ξ_m for each training point $\boldsymbol{x}_m$ and relaxes the constraints associated with it [19]:

$$y_m(\boldsymbol{w}^T \boldsymbol{x}_m + b) \geq 1 - \xi_m,$$

where $\xi_m \geq 0$. In order to minimize these violations, an additional term of slack variables will be introduced in the SVM optimization problem, which can have a quadratic loss on the slack variables. The new optimization problem for SVM is as follows [19]:

$$\begin{cases} \underline{\text{Objective function}} : \min\limits_{\boldsymbol{w},b} \ \dfrac{1}{2}\|\boldsymbol{w}\|^2 + C \sum\limits_{m=1}^{M} \xi_m^2, \\ \underline{\text{Linear constraints}} : \ y_m h(\boldsymbol{x}_m) \geq 1 - \xi_m, \ \ \forall \boldsymbol{x}_m \in D_{\text{train}}, \end{cases} \tag{3.12}$$

where C is a penalty parameter trading off the size of the margin and the number of missclassifications. Larger C leads to a small number of misclassifications but with a small margin and vice versa.

3.2.2 Kernel SVM: Non linear case

The majority of real-world machine learning problems cannot be linearly separated, therefore learning algorithms based on linear decision functions, such as SVM, cannot identify appropriate boundaries for classifying data. For such a challenging situation, kernel methods are an effective solution. In general, the idea is to project data using a nonlinear map ϕ called the feature map into a high dimensionnal Hilbert space $\mathbb{H}$, where data become linearly separable. The

map ϕ can be defined as follows [134]:

$$\phi : \mathbb{R}^I \to \mathbb{H} \tag{3.13}$$

$$\boldsymbol{x} \to \phi(\boldsymbol{x}) \tag{3.14}$$

In practice, the vector $\boldsymbol{x}$ can lie in a high-dimensional space getting its transformation using the feature map ϕ very complicated involving many polynomial combinations of its components. This will lead to impractical and extremely high computationnal cost. To overcome this problem, the concept of the kernel trick is introduced [12]. In fact, instead of computing explicitely the transformed outputs $\phi(\boldsymbol{x})$ in the feature space $\mathbb{H}$, we only need to compute the inner products of the transformed vectors $\phi(\boldsymbol{x})$ in the feature space. This inner products are given via a kernel function defined as follows [129]:

Definition 13. *(Kernel function in $\mathbb{R}^I$) For $\boldsymbol{x}, \boldsymbol{y} \in \mathbb{R}^I$ and $\phi : \mathbb{R}^I \to \mathbb{H}$, then*

$$k(\boldsymbol{x}, \boldsymbol{y}) = \langle \phi(\boldsymbol{x}), \phi(\boldsymbol{y}) \rangle, \tag{3.15}$$

is a kernel function.

It is important to note that using the kernel function implicitely defines a feature map and a feature space that are not explicitly constructed in practice. The kernel can then be constructed without explicitly constructing the feature space. For a function k to be considered as a valid kernel, it must be a real-valued, positive definite function [12] as follows:

Definition 14. *A function $k : \mathbb{R}^I \times \mathbb{R}^I \to \mathbb{R}$ is called a kernel if it is a symmetric and positive definite function i.e.:*

$$\forall M \in \mathbb{N}^*, \forall \boldsymbol{x}_1, \ldots, \boldsymbol{x}_M \in \mathbb{R}^I, \forall c_1, \ldots c_M \in \mathbb{R}, \sum_{m_1=1}^{M} \sum_{m_2=1}^{M} c_{m_1} c_{m_2} k(\boldsymbol{x}_{m_1}, \boldsymbol{x}_{m_2}) \geq 0. \tag{3.16}$$

Examples of kernels on $\mathbb{R}^I$:

- **Polynomial kernel**:
  The Polynomial kernel represents the similarity between two vectors as

follows:

$$k_P(\boldsymbol{x}, \boldsymbol{x}) = (\langle \boldsymbol{x}, \boldsymbol{x} \rangle + 1)^n,$$

where n is the degree of the polynomial. This kernel consists of all monomials with degrees less than or equal to n and their combinations. If $n = 1$, the constant is omitted, and k_{P} corresponds to the linear kernel generated by the dot product, which corresponds to the angle between the two vectors (that have been normalized). The linear kernel does not perform well for challenging classification problems, particularly when the dataset cannot be separated linearly. In most cases, $n = 2$ is used, since large values of n can result in an overfitting of the data (the score on the training set is high and low given new data).

- **Radial Basis Function kernel**:
  The Radial Basis Function kernel (or Gaussian-Euclidean (GE)) is a kernel that has the form of a Gaussian function. GE represents the similarity between two vectors as a function of their euclidean distance (the squared norm of the distance between them). According to this definition, k_{GE} is calculated as follows:

$$k_{\mathrm{GE}}(\boldsymbol{x}, \boldsymbol{y}) = \exp\left(-\gamma \|\boldsymbol{x} - \boldsymbol{y}\|^2\right),$$

  where the parameter γ determines the influence of each data point. Depending on γ, this can give a good fit or an over fit: If γ is too small compared to the distance between classes, this means discriminant surfaces will be flat. If γ is too large, it may be overfitting. Thus, the parameter γ is crucial to have good performance. Selecting a proper value is necessary and worth to do in practice.

Kernel matrix:

Based on the definition of the kernel function, we can define an important quantity during learning, known as the kernel matrix $\boldsymbol{K}$. For a set of data

$\{\boldsymbol{x}_1,\dots,\boldsymbol{x}_M\}$, the kernel matrix (also called Gram matrix) is the $M\times M$ defined as follows:

$$\boldsymbol{K}=\begin{pmatrix} k(\boldsymbol{x}_1,\boldsymbol{x}_1) & \dots & k(\boldsymbol{x}_1,\boldsymbol{x}_M)\\ \vdots & \dots & \vdots \\ k(\boldsymbol{x}_M,\boldsymbol{x}_1) & \dots & k(\boldsymbol{x}_M,\boldsymbol{x}_M)\end{pmatrix}.$$

We will see that in dual forms of SVMs utilizing kernel methods, the algorithm receives information about the training set from the kernel matrix and the labels associated with it.

Optimization problem of SVM using kernels:

By mapping data using a feature map ϕ, the optimization problem in eq. (3.12) is given in the feature space by the following [134]:

$$\begin{cases} \underline{\text{Objective function}} : \min\limits_{\boldsymbol{w},b} \dfrac{1}{2}\|\boldsymbol{w}\|^2 + C\sum\limits_{m=1}^{M}\xi_m^2, \\ \underline{\text{Linear constraints}} : \ y_m(\phi(\boldsymbol{x}_m)^T\boldsymbol{w}+b)\geq 1-\xi_m, \ \ \forall \boldsymbol{x}_m\in D_{\text{train}}, \end{cases} \tag{3.17}$$

where the weight vector $\boldsymbol{w}$ and the slack variables are all defined in the feature space.
The dual problem of eq.(3.17) is then given by [134]:

$$\begin{cases} \underline{\text{Objective function}} : \max\limits_{\alpha} \sum\limits_{m=1}^{M}\alpha_m - \dfrac{1}{2}\sum\limits_{m_1=1}^{M}\sum\limits_{m_2=1}^{n}\alpha_{m_1}\alpha_{m_2}y_{m_1}y_{m_2}\langle\phi(\boldsymbol{x}_{m_1}),\phi(\boldsymbol{x}_{m_2})\rangle, \\ \underline{\text{Linear constraints}} : \ 0\leq\alpha_m\leq C, \sum\limits_{m=1}^{M}\alpha_m y_m = 0. \end{cases} \tag{3.18}$$

It should be noted that the dual problem in eq. (3.18) depends only on dot products between data in the feature space and since this dot products verify $\langle\phi(\boldsymbol{x}_{m_1}),\phi(\boldsymbol{x}_{m_2})\rangle = k(\boldsymbol{x}_{m_1},\boldsymbol{x}_{m_2})$, the optimization problem in eq.(3.18) will be solved using the kernel matrix $\boldsymbol{K}=\{(\boldsymbol{x}_{m_1},\boldsymbol{x}_{m_2})\}_{m_1,m_2=1}^{M}$.

Weight vector and bias:

The solution $(\boldsymbol{w}, b)$ of eq.(3.18) is given by [94]:

$$\boldsymbol{w} = \sum_{\alpha_M > 0} \alpha_m y_m \phi(\boldsymbol{x}_m), \quad b = y_{m_0} \alpha_{m_0} - \sum_{\alpha_m > 0} y_m k(\boldsymbol{x}_m, \boldsymbol{x}_{m_0}), \;\; m_0 \in \{1, \ldots, M\}. \tag{3.19}$$

Decision function:

The predicted class for a new point $\boldsymbol{z}$ is given by:

$$y = \operatorname{sgn}\Big(\boldsymbol{w}^T \phi(\boldsymbol{z}) + b\Big), \tag{3.20}$$

$$= \operatorname{sgn}\left(\sum_{\alpha_m > 0} \alpha_m y_m \phi(\boldsymbol{x}_m)^T \phi(\boldsymbol{z}) + b\right), \tag{3.21}$$

$$= \operatorname{sgn}\left(\sum_{\alpha_m > 0} \alpha_m y_m k(\boldsymbol{x}_m, \boldsymbol{z})\right). \tag{3.22}$$

In the above equation, only dot products are evaluated on the feature space specified by the non-linear kernel function k, so it is not necessary to calculate $\phi(\boldsymbol{x}_m)$ explicitly. The choice of the kernel function will have a significant impact on the performance of the classification process. Although there is no theory on how to select a kernel based on data, the Gaussian-Euclidean kernel has been found to be very effective for a wide variety of classification problems [16].

Kernels as similarity measures:

Because kernels are inner products in the feature space, they naturally induce similarity measures that quantify similarity between objects [12]. Unlike distance metrics, kernel functions yield large values for similar objects and small values for dissimilar objects. It is possible to define kernels on a variety of objects, such as subspaces, graphs, images, tensors, texts, and so on. Based on the context, the structure of the data, and the knowledge of the application, a kernel function may be defined as a comparison between two input objects.

3.3 Kernels on Grassmann manifold

In many applications such as human activity modeling, face recognition, video based face recognition, learning is involved on cetain non-euclidean spaces called Grassmann manifolds [124, 139, 62, 39]. Due to the non-euclidean geometry of these spaces, learning algorithms cannot be directly used. To overcome this problem, some techniques are used to be able to use Euclidean tools. One of this tools lies on the use of kernels. We will present how in this section. but first we will review some basic Grassmannian geometry concepts.

3.3.1 Grassmann manifold

Definition 15. *(Grassmann manifold) The Grassmann manifold $\mathbb{G}(I,R)$ with integers $I \geq R$ is the space formed by all R-dimensional linear subspaces in an I-dimensional euclidean space. An element of $\mathbb{G}(I,R)$ is represented by an arbitrarly $I \times R$ orthonormal matrix $\boldsymbol{U}$ whose columns span the corresponding subspace. $\mathbb{G}(I,R)$ can be written as [106]:*

$$\mathbb{G}(I,R) = \{span(\boldsymbol{U}), \boldsymbol{U} \in \mathbb{R}^{I\times R}, \boldsymbol{U}^T\boldsymbol{U} = \boldsymbol{I}_R\}, \tag{3.23}$$

where $span(\boldsymbol{U})$ denotes the subspace spanned by the columns of $\boldsymbol{U}$. Since elements in $\mathbb{G}(I,R)$ are subspaces, they are invariant to rotations, this means that $\boldsymbol{U}$ and $\boldsymbol{UR}$ define the same elements in $\mathbb{G}(I,R)$ for any $R \times R$ invertible matrix $\boldsymbol{R}$.

Each element of $\mathbb{G}(I,R)$ can be identified with an equivalence class of orthonormal basis that span the same subspace. Theses orthonormal basis are elements of the so-called Stiefel manifold defined by:

$$\mathbb{S}(I,R) = \{\boldsymbol{U} \in \mathbb{R}^{I\times R}, \boldsymbol{U}^T\boldsymbol{U} = \boldsymbol{I}_R\}. \tag{3.24}$$

3.3.2 Principal angles

Given two subspaces $span(\boldsymbol{U}), span(\boldsymbol{V}) \in \mathbb{G}(I,R)$ there exist R angles $\{\theta_r\}_{r=1}^R$ defined by [3, 50]:

$$\begin{cases} \cos(\theta_r) = \max\limits_{\boldsymbol{u}_r \in span(\boldsymbol{U}), \boldsymbol{v}_r \in span(\boldsymbol{V})} \boldsymbol{u}_r^T \boldsymbol{v}_r, \\ \boldsymbol{u}_r^T \boldsymbol{u}_r = 1, \ \boldsymbol{v}_r^T \boldsymbol{v}_r = 1 \ \ 1 \leq r \leq R, \\ \boldsymbol{u}_r^T \boldsymbol{u}_l = 0, \ \boldsymbol{u}_r^T \boldsymbol{u}_l = 0 \ \forall l < r, \ r \geq 1. \end{cases} \tag{3.25}$$

The principal angles $\{\theta_r\}_{r=1}^R$ form a set of minimal angles between all possible bases of $span(\boldsymbol{X})$ and $span(\boldsymbol{V})$. The vectors $\{\boldsymbol{u}_r\}_{r=1}^R$ and $\{\boldsymbol{v}_r\}_{r=1}^R$ are called principal vectors of the pair of subspaces. In practice, principal angles can be computed from the SVD, where the singular values s_r of $\boldsymbol{U}^T\boldsymbol{V}$ are the cosines of the principal angles. Thus, θ_r are computed by the following:

$$\theta_r = \arccos(s_r), \ \ 1 \leq r \leq R. \tag{3.26}$$

3.3.3 Subspace Distance on Grassmann manifold

In Riemannien manifolds, smooth curves connect points. A Riemannien manifold's geodesic distance between two points is equal to the length of the shortest curve connecting them. In the Grassmannien manifold, the geodesic distance is obtained from the 2-norm of the principal angle vector between two points[8].

$$d_G(span(\boldsymbol{U}), span(\boldsymbol{V})) = \|\theta\|_2. \tag{3.27}$$

There are still many other distances that can be defined on a Grassmann manifold [63, 83]. This is not an exhaustive list. But, we will give some examples w.r.t to the principal angles $\{\theta_r\}_{r=1}^R$ and the orthonormal matrices $\boldsymbol{U}$ and $\boldsymbol{V}$ spanning $span(\boldsymbol{U})$ and $span(\boldsymbol{V})$.

- Chordal distance [8]:

$$d_C(span(\boldsymbol{U}), span(\boldsymbol{V})) = \|\boldsymbol{U}\boldsymbol{U}^T - \boldsymbol{V}\boldsymbol{V}^T\|_F = \sqrt{2}\left(\sum_{r=1}^{R} \sin^2(\theta_r)\right)^{1/2}. \tag{3.28}$$

To understand the construction of the chordal distance in eq. (3.28), recall that Grassmann manifold is not an Euclidean space, therefore, tools of Euclidean space cannot be used. To address this issue, one way is to embed elements of the grassmann manifold onto an Euclidean space [50]. By taking the space of $I \times I$ matrices $Sym(I)$ and taking the Frobenius norm as a distance in the embedding space, the projection metric can be derived [83]. Due to the fact that singular values of $\boldsymbol{U}^T\boldsymbol{V}$ are the cosines of principal angles, the formula with principal angles can be derived.

- Projection metric [147]:

$$d_P(span(\boldsymbol{U}), span(\boldsymbol{V})) = \sin(\theta_R) = \|\boldsymbol{U}\boldsymbol{U}^T - \boldsymbol{V}\boldsymbol{V}^T\|_2. \tag{3.29}$$

As with the chordal distance metric, the projection metric is derived by the same embedding using the 2-norm.

3.3.4 Grassmann kernel

It is also possible to define kernel methods on Grassmann manifolds. The kernel must verify the following properties in order to be considered valid on the Grassmann manifold:

Definition 16. *A function $k : \mathbb{G}(I,R) \times \mathbb{G}(I,R) \to \mathbb{R}$ is a Grassmanine kernel if it is positive definite and invariant to the choice of basis to represent subspaces in $\mathbb{G}(I,R)$ [83]. This means that*

$$k(span(\boldsymbol{U}), span(\boldsymbol{V})) = k(span(\boldsymbol{U}\boldsymbol{R}_1), span(\boldsymbol{Y}\boldsymbol{R}_2)), \tag{3.30}$$

for all $span(\boldsymbol{U}), span(\boldsymbol{V}) \in \mathbb{G}(I,R)$ and for all $R \times R$ invertible matrices $\boldsymbol{R}_1, \boldsymbol{R}_2$.

3.3.5 Examples of Grassmannien kernels

It is possible to generalize Gaussian kernels on Grassmannien manifolds by replacing Euclidean distance with a Grassmannien one. Such a kernel will be

called a Grassmannien kernel that follows this form [124]:

$$k(span(\boldsymbol{U}), span(\boldsymbol{V})) = \exp\left(-\gamma\left(d^2\left(span(\boldsymbol{U}), span(\boldsymbol{V})\right)\right)\right), \tag{3.31}$$

where d can be any Grassmannian distance that allows the Grassmann kernel to be a valid Gaussian kernel, such as the chordal as well as the projection distance. Several Grassmannien metrics do not result in valid Gaussian kernels, such as the geodesic distance, which does not allow the Gaussian kernel to be positive definite.
The major focus of the work will be the use of Gaussian kernels because they are universal kernels [57], i.e., they provide good results in approximating any arbitrarily continuous function, provided there is sufficient training data. Besides these kernels, there are several other Grassmannien kernels, including:

- Binet-Cauchy kernel [78]:

$$k(span(\boldsymbol{U}), span(\boldsymbol{V})) = det(\boldsymbol{U}\boldsymbol{U}^T\boldsymbol{V}\boldsymbol{V}^T). \tag{3.32}$$

- Polynomial kernel [83]:

$$k(span(\boldsymbol{U}), span(\boldsymbol{V})) = (\beta + \|\boldsymbol{U}^T\boldsymbol{V}\|_F^2)^n, \tag{3.33}$$

  where $\beta > 0$ and n is the polynomial degree.

In the following section, we introduce a number of methods that extend SVM to handle tensor data. As a next step, we will review some literature that presents kernel methods for tensor data.

3.4 Support Tensor Machines

Traditional supervised learning methods based on flattening tensors into vectors or matrices will not be effective for two reasons. First, tensors will lose their structure [72]. In general, an entry in a data tensor is correlated with its surrounding entries, much like the arrangement of voxels in a 3D FMRI image,

where adjacent voxels typically exhibit similar characteristics, or the arrangement of pixels in an image.
Furthermore, tensors have large dimensions. As an example, consider the video of the UCF11 dataset [74], which is a tensor of order 4 and size $240\times240\times320\times3$. If we flatten this tensor, we get 55×10^6 features, which leads to overfitting (The learning algorithm seems working well on the training data set, but does not perform well on data that has not been seen before), particularly if there are only a few data points [47]. Moreover, learning with such large vectors is computationally inefficient.
For all these reasons, [21, 144, 70, 42, 20, 32] propose not to flatten the data tensor. To avoid the curse of dimensionality problem related to tensors, they propose to retain information about the structure of the tensor using tensor decompositions. Based on these decompositions, they propose algorithms for supervised learning on data tensors that is called supervised tensor learning. In this context, the following is a description of the problem statement of Support Tensor Machines (STM) as the extension of SVM to tensors.

Given a training set of input tensors $\{(\mathcal{X}_m, y_m)\}_{m=1}^{M}$, we would like to solve a binary classification problem where $\mathcal{X}_m \in \mathbb{R}^{I_1\times\cdots\times I_N}$ and $y_m \in \{-1,1\}$ are the labels of $\mathcal{X}_m$. The extension of the standard linear SVM in eq.(3.12) is given by [72]:

$$\begin{cases} \min\limits_{\mathcal{W},b,\xi} \ \dfrac{1}{2}\|\mathcal{W}\|_F^2 + C\sum\limits_{m=1}^{M}\xi_m, \\ \underline{\text{subject to}}: \ \ y_m(\langle\mathcal{W},\mathcal{X}_m\rangle + b) \geq 1-\xi_m, \\ \xi_m \geq 0, 1\leq m\leq M, \end{cases} \tag{3.34}$$

where $\mathcal{W} \in \mathbb{R}^{I_1\times\cdots\times I_N}$ is a tensor weight parameter.
The multiway correlation in $\mathcal{W}$ can be captured by imposing a low rank constraint on $\mathcal{W}$ [32]. In [32], for example, Supervised Tensor Learning (STL) is proposed, and $\mathcal{W}$ is assumed to be rank-1. However, it involves nonconvex optimization problems and requires iterative methods. As a result, it is very time-consuming and may lead to local minima. For better model expressive power, rank-1 weight tensors of STM are generalized using the CPD in [42]. In

addition, they took into account the spread of training data along the different tensor modes. The approach described in [41] called Support Tucker machines (STuMs) assumes that the weight tensor follows the Tucker model. The size of the weight tensor, $\mathcal{W}$, however, scales exponentially with the number of entries, so it is computationally prohibitive to learn. This problem can be addressed by [20] by proposing Support Tensor Train Machines (STTM) . This method replaces the TD with the TTD that can approximate any tensor with a scalable number of parameters. As an alternative to imposing constraints on the weight parameter, [153] proposes a linear support tensor machine that constructs a hyperplane in the tensor space based on inner products between input tensors. This approach yields linear bounds for classification, which is extremely limited.

Using the approaches cited, the problem can be solved in eq.(3.34) in its primal form. However, solving the dual form will allow kernel methods to be applied easily to more challenging datasets in order to detect nonlinear boundaries. As a result, [72] extends eq. 3.34 to the nonlinear case using a nonlinear map ϕ described below:

$$\phi : \mathbb{R}^{I_1 \times \cdots \times I_N} \rightarrow \mathbb{R}^{H_1 \times \cdots \times H_N}. \tag{3.35}$$

The projected tensor $\phi(\mathcal{X})$ may be of a different order from $\mathcal{X}$, and its dimensions may be larger than those of $\mathcal{X}$, or be infinite. The tensor feature space is the space in which the tensor $\mathcal{X}$ is projected. In this space, 3.34 can be represented by the following:

$$\begin{cases} \min\limits_{\mathcal{W},b,\xi} \ \dfrac{1}{2}\|\mathcal{W}\|_F^2 + C\sum\limits_{m=1}^{M}\xi_m, \\ \underline{\text{subject to}}: \ \ y_m(\langle \mathcal{W}, \phi(\mathcal{X}_m)\rangle + b) \geq 1 - \xi_m, \\ \xi_m \geq 0, \ \ 1 \leq m \leq M, \end{cases} \tag{3.36}$$

The Dual problem of 3.36 is expressed as [72]:

$$\begin{cases} \underline{\text{Objective function}} : \max\limits_{\alpha_m} \sum\limits_{m=1}^{M} \alpha_m - \frac{1}{2} \sum\limits_{m_1=1}^{M} \sum\limits_{m_2=1}^{M} \alpha_{m_1} \alpha_{m_2} y_{m_1} y_{m_2} \langle \phi(\mathcal{X}_{m_1}), \phi(\mathcal{X}_{m_2}) \rangle, \\ \underline{\text{Linear constraints}} : \ 0 \leq \alpha_m \leq C, \sum\limits_{m=1}^{M} \alpha_m y_m = 0, 1 \leq m \leq M. \end{cases} \tag{3.37}$$

Based on the fundamental principe of kernel methods, the inner products $\langle \phi(\mathcal{X}_{m_1}), \phi(\mathcal{X}_{m_2}) \rangle$ is given by a tensorial kernel $k(\mathcal{X}_{m_1}, \mathcal{X}_{m_2})$. The predicted class for a new tensor $\mathcal{Z}$ is then given by:

$$f(\mathcal{Z}) = \text{sgn}\left(\sum_{m=1}^{M} \alpha_m y_m k(\mathcal{X}_m, \mathcal{Z}) + b \right), \tag{3.38}$$

where k is a kernel function on the tensor space.
In order to classify a new tensor point $\mathcal{Z}$, the evaluation of similarities between $\mathcal{Z}$ and all the training points should be computed, therefore, the choice of k is crucial. In this context, in their publication [90], Signoretto et al. propose a Grassmannien tensor-based kernel. The kernel function is defined by considering the matrix-based subspaces spanned by factors of the HOSVD. Recently, [21] Kernelized Support Tensor Train Machines (KSTTM) proposed a kernelized support tensor train machine. The authors propose a kernel metric based on kernel mappings on the different fibers of TT-cores based on the TTD. It should be noted, however that due to the fact that TTD is not unique, the kernel function expression in [21] may be negatively impacted. The work described in [71] proposes a kernelized tensor factorization method which can be understood as performing CPD in a high-dimensional space implicitly defined by a kernel function. In [70], this approach was generalized using the TD and the parameters of the classification model were estimated jointly. The DuSK method [72] proposes a scheme to design structure-preserving kernels for supervised tensor learning. First, the CPD is used to extract a more compact and informative representation of the original data, and then a new kernel function is defined in the tensor space

based on the columns of the CP factors. It is important to note, however, that the ambiguities inherent in the CPD will affect the performance classification of this method. In the following chapter, we will provide a detailed analysis of this method, as well as analyze how scaling impacts the intrinsic similarity measure.

Chapter 4

Probabilistic analysis of the CPD-based tensor learning

Outline of the current chapter

In this chapter, the tensor-based kernel approach DuSK proposed in [72] is analyzed. This method aims to design a tensorial kernel for tensors. Based on the CPD, [72] uses a GE kernel between CP factors in order to perform classification

using STM. However, their choice of kernel function is unable to address the scaling ambiguities inherent in the CPD. It is demonstrated that the DuSK method fails theoretically to satisfy the intrinsic property of similarity of a kernel function. In particular, it fails to the evaluation of auto-similarity, which is the similarity between the tensor and itself. In addition, it will be shown that the scaling ambiguities of the CPD corrupt the STM algorithm's decision function. The difficulty of DuSK to achieve high classification performance on real datasets can be explained by these two weaknesses. By modifying the kernel choice based on Grassmannian geometry, the scaling ambiguities of the CPD can be effectively managed and the classification performance will be improved.

4.1 Analysis of the DuSK's scheme

Let $(\mathcal{X}, \mathcal{X}')$ a couple of rank-R tensors, the tensorial kernel function proposed in DuSK [72] is based on their CPDs given by:

$$\mathcal{X} = \sum_{r=1}^{R} \bigcirc_{n=1}^{N} \boldsymbol{u}_{n,r}, \quad \mathcal{X}' = \sum_{r=1}^{R} \bigcirc_{n=1}^{N} \boldsymbol{u}'_{n,r}. \tag{4.1}$$

The DuSK's kernel k_{DuSK} is defined as follows:

$$k_{\text{DuSK}}(\mathcal{X}, \mathcal{X}') := \frac{1}{R} \sum_{r=1}^{R} \sum_{r'=1}^{R} \prod_{n=1}^{N} k\left(\boldsymbol{u}_{n,r}, \boldsymbol{u}'_{n,r'}\right), \tag{4.2}$$

where k_{GE} is the standard Gaussian kernel. Thus, for every pair of rank-one tensors of $\mathcal{X}$ and $\mathcal{X}'$, the product of subkernels k_{GE} between the respective columns of the CP factors is calculated.

As stated in section 2.1.2, the CPD is unique under mild conditions. These conditions are supposed to be verified here up to the ambiguities of the CPD. The permutation ambiguity will not affect the expression of the k_{DuSK} kernel since the order of computing similarities between rank-one tensors is irrelevant(they will be added at the end via the summation over r and r' in eq.(4.2)). Nevertheless, the scaling ambiguity will affect the similarity of the kernel function even if the

column factors of CP are normalized. This normalization is an important step before calculating the subkernel k_{GE} to ensure that no column factors affect the kernel value because of its range. Even though normalization was performed, sign ambiguity remains, which will negatively affect classification performance. For measuring these impacts, we recall the notion of the congruent set $\mathfrak{X}_k$ which contains tensors that are similar to $\mathcal{X}$ with respect to a kernel function k.

Definition 1. *[89] With a normalized kernel function k that assigns a value of 1 to similar tensors, the congruent set $\mathfrak{X}_k$ associated with a tensor $\mathcal{X}$ is the set of similar tensors to $\mathcal{X}$ and defined as follows:*

$$\mathfrak{X}_k = \{\mathcal{X}' \in \mathbb{R}^{I_1 \times \cdots \times I_N} : k(\mathcal{X}, \mathcal{X}') = 1\}.$$

From the definition of the congruent set above, $\mathfrak{X}_k$ should at least contain the tensor $\mathcal{X}$ itself. However, we will demonstrate that the auto-similarity (the similarity between a tensor ant itself) is prone to zero value.

4.1.1 Probability of nonemptiness of the congruent set

We will demonstrate that in the general case of rank-R tensors, the probability of $\mathfrak{X}_{k_{DuSK}}$ not being empty tends to zero with the order N.
Let us consider two CPDs of a rank-R tensor $\mathcal{X}$ from eq.(2.7):

$$\mathcal{X} = \sum_{r=1}^{R} \lambda(r) \bigcirc_{n=1}^{N} (\operatorname{sgn}(\beta_{n,r})\boldsymbol{u}_{n,r}), \quad \mathcal{X} = \sum_{r=1}^{R} \lambda'(r) \bigcirc_{n=1}^{N} \operatorname{sgn}\left(\left(\beta'_{n,r}\right)\boldsymbol{u}_{n,r}\right). \tag{4.3}$$

Thus, the kernel $k_{\mathrm{DuSK}}(\mathcal{X}, \mathcal{X})$ can be written as:

$$k_{\mathrm{DuSK}}(\mathcal{X}, \mathcal{X}) = \frac{1}{R}\left(\sum_{r=1}^{R} \exp\left(-\gamma \sum_{n=1}^{N} (1 - \operatorname{sgn}(\alpha_{n,r}))^2\right) + T_{N,R}\right),$$

where

$$\alpha_{n,r} = \frac{\beta_{n,r}}{\beta'_{n,r}}, \tag{4.4}$$

and

$$T_{N,R} := \sum_{r=1}^{R} \sum_{\substack{r'\neq r \\ r'=1}}^{R} \prod_{n=1}^{N} \exp\left(-\gamma \left\| \operatorname{sgn}(\beta_{n,r})\boldsymbol{u}_{n,r} - \operatorname{sgn}(\beta'_{n,r'})\boldsymbol{u}_{n,r'} \right\|^2\right). \tag{4.5}$$

In eq.(4.5), the ambiguity of permutation of the CPD is considered equal to the identity matrix because of the double summation over the rank.
The constraints of the scaling factors of the CPD are given by (see section 2.1.2):

$$\prod_{n=1}^{N} \beta_{n,r} = 1, \quad \prod_{n=1}^{N} \beta'_{n,r} = 1, \ \ 1 \leq r \leq R, \tag{4.6}$$

Thus, we have :

$$\prod_{n=1}^{N} \alpha_{n,r} = 1. \tag{4.7}$$

It is clear from eq. (4.7) that the number of negative values of $\alpha_{n,r}$ defined by $Q_r = \#\{\alpha_{n,r} = -1\}_n$ must be an even number that belongs to $\{0, \cdots, N\}$. This quantity will be crucial when examining the non-emptiness of $\mathfrak{X}_{k_{DuSK}}$.
An example may be given as follows: for a given tensor of order $N = 3$, if the set of signs of $\beta_{n,1}$ is $\{1, -1, 1\}$ and the set of signs $\beta'_{n,1}$ is $\{-1, 1, 1\}$, then the set of signs of $\alpha_{n,1}$ are $\{-1, -1, 1\}$ and Q_1 is equal to 2.
The terms $\alpha_{n,r}$ are then considered discrete random variables and Q_r will be modeled as a binomial random variable belonging to a truncated support, which leaves out odd values. Based on these assumptions, the following theorem gives the probability of nonemptiness for rank-R tensors.

Theorem 1. *Let $\mathcal{X}$ be a rank-R tensor. We can show that:*

$$\mathbb{P}(\mathfrak{X}_{k_{DuSK}} \neq \varnothing) \leq \mathbb{P}\left(\prod_{r=1}^{R} Q_r = C_R\right) \underset{N\to+\infty}{\longrightarrow} 0, \tag{4.8}$$

where $C_R = \left[\dfrac{\ln(R)}{(4\gamma)^R}\right]$.

Proof. See Appendix .1. □

As shown by Theorem 1, computing the auto-similarity using k_{DuSK} results in an empty congruent set. Consequently, two identical tensors can not even be considered as elements from the same class. Thus, as N increases, k_{DuSK} will be unable to verify the kernel properties.
The following specific results are obtained for rank-one tensors:

Theorem 2. *Let $\mathcal{X}$ be a rank-1 tensor. We can show that:*

- *The probability to have a non-empty congruent set is given by:*

$$\mathbb{P}(\mathfrak{X}_{k_{DuSK}} \neq \varnothing) = \mathbb{P}(Q_1 = 0) = \frac{1}{2^{N-1}}. \tag{4.9}$$

- *The expectation $\mathbb{E}\Big[k_{DuSK}(\mathcal{X},\mathcal{X})\Big]$ converges towards 0 with respect to the order N.*

Proof. See Appendix .2. □

Theorem 2 provides the exact probability of non-emptiness of $\mathfrak{X}_{k_{\text{DuSK}}}$ for rank-one tensors as well as the expectation of k_{DuSK} and shows that the DuSK kernel fails to measure the autosimilarity and thus the congruent set is empty with probability 1 when the tensor's order N tends to infinity.
To validate the choice of modelization of the random variable N_1, the following numerical experiment is conducted. A single rank-one deterministic tensor of different orders where all dimensions are equal to 5 is generated. The theoritical value of $\mathbb{P}(Q_1 = 0)$ given by the model considered is compared with the empirical value given by 1000 CPDs of a fixed tensor of a specific order. We present in Figure 4.1 a fitting of the numerical and theoretical probabilities. We note that the model chosen for Q_1 is well suited.

4.1.2 Effect of scaling ambiguity on the kernel matrix

This section demonstrates how the failure of k_{DuSK} to verify the similarity property impacts the kernel matrix, which is the key to parameter learning. We

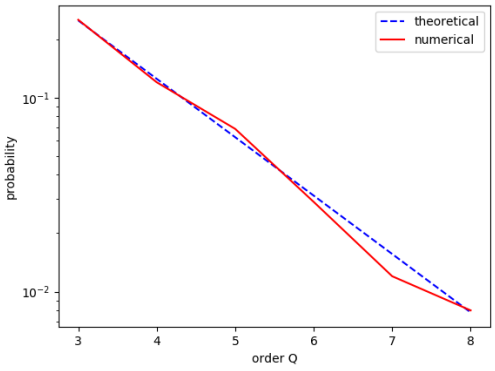


Figure 4.1 – Illustration of the probability of $\mathfrak{X}_{k_{\text{DuSK}}}$ to have a non empty congruent set for different values of N. Theoretical and Numerical values of $\mathbb{P}(\mathfrak{X}_{k_{\text{DuSK}}} \neq \varnothing)$ are presented. Theoretical values are computed using the probability from Theorem 2.

consider rank-one tensors since ambiguity permutation is not an issue with k_{DuSK} as explained in section 4.1 and thus, the rank-one case is considered here just to see the impact of the scaling ambiguity. Consider the following toy example from a binary classification problem:

$$\mathcal{D}_{train} = \{ \underbrace{\mathcal{X}_0, \mathcal{X}_0}_{\text{1st class}} , \underbrace{\mathcal{X}_1, \mathcal{X}_2}_{\text{2nd class}} \}. \tag{4.10}$$

Theoretically, the kernel matrix should have the following form:

$$\boldsymbol{K} = \begin{pmatrix} 1 & \boldsymbol{K}(1,2) & O(\epsilon) & O(\epsilon) \\ \boldsymbol{K}(1,2) & 1 & O(\epsilon) & O(\epsilon) \\ O(\epsilon) & O(\epsilon) & 1 & \boldsymbol{K}(3,4) \\ O(\epsilon) & O(\epsilon) & \boldsymbol{K}(3,4) & 1 \end{pmatrix},$$

where $k_{\text{DuSK}}(\mathcal{X}_m, \mathcal{X}_m) = 1$, $1 \leq m \leq 4$, $O(\epsilon)$ is a small constant and

$$\boldsymbol{K}(1,2) = k_{\text{DuSK}}(\mathcal{X}_0, \mathcal{X}_0),$$
$$\boldsymbol{K}(3,4) = k_{\text{DuSK}}(\mathcal{X}_1, \mathcal{X}_2).$$

As $\mathcal{X}_0$ appears twice in the dataset, the CPD of $\mathcal{X}_0$ will be computed twice, resulting in differing scaling factors. Thus, $\boldsymbol{K}(1,2)$ will tend towards 0 thanks to Theorem 2. In this case, the submatrix $M := \begin{pmatrix} 1 & \boldsymbol{K}(1,2) \\ \boldsymbol{K}(1,2) & 1 \end{pmatrix}$ in the kernel matrix $\boldsymbol{K}$, will be equal to $\begin{pmatrix} 1 & O(\epsilon) \\ O(\epsilon) & 1 \end{pmatrix}$ instead of $\begin{pmatrix} 1 & 1 \\ 1 & 1 \end{pmatrix}$. Based on the experiments in section 4.3, the learning process may fail. The failure of k_{DuSK} will also negatively affect the decision function as we will see in the next section.

4.1.3 Effect of scaling ambiguity on the decision function

In this section, we show how the DuSK method fails to accurately predict the right labels due to the scaling ambiguity of the CPD of input data tensors. This can be illustrated by considering the classification of a new data tensor $\mathcal{Z}$ with label 1 without any loss of generality. In such case, the decision function f is given by:

$$f(\mathcal{Z}) = \operatorname{sgn}(T_1 - T_2) + b, \tag{4.11}$$

where

$$T_1 = \sum_{m \in S} \alpha_m k_{GE}(\mathcal{X}_i, \mathcal{Z}),$$

and

$$T_2 = \sum_{m \notin S} \alpha_m k_{GE}(\mathcal{X}_m, \mathcal{Z}),$$

where b is the parameter of SVM and α_m are the dual lagrangien variables of the primal problem of SVM discussed in chapter 3 and S represents the set of indices of the same class as $\mathcal{Z}$. A numerical value of T_1 should be larger than that of T_2 because T_1 correponds to kernels of similar tensors whereas T_2 correponds to the other class. However, we verify that it is not always the case.

In order to achieve this, 4th-order rank-one tensors were generated where all dimensions are equal to 20. The columns of CP factors of tensors of each class

	1	2	3	4	5	6	7
T_1	$\mathbf{10^{-34}}$	10^{-4}	$\mathbf{10^{-33}}$	10^{-4}	$\mathbf{10^{-27}}$	$\mathbf{10^{-33}}$	$\mathbf{10^{-28}}$
T_2	$\mathbf{10^{-11}}$	10^{-13}	$\mathbf{10^{-4}}$	10^{-14}	$\mathbf{10^{-9}}$	$\mathbf{10^{-10}}$	$\mathbf{10^{-4}}$
$f(\mathcal{Z})$	**-1**	1	**-1**	1	**-1**	**-1**	**-1**

Table 4.1 – Different realisations and comparisons of T_1 (kernel value of similar tensors) and T_2 (kernel value of different tensors) using the DuSK kernel

are respectively generated from a Gaussian distribution of standard deviation 0.01 and mean 0.2 and (resp. 0.3). α_m are found equal to 1 and b equal to 0. Table 4.1 shows the repetition of the numerical experiment described. We note that in realisations 1,3,5,6,7 in Table 4.1, T_2 is greater than T_1. This means that the similarity between tensors within the same class is smaller than that between tensors of different classes. As a result, whenever the test point $\mathcal{Z}$ has label 1, the decision function in eq. (4.11) will return the false prediction if $T_1 < T_2$.

4.2 Proposed scheme

A possible solution to the scaling ambiguity due to the CPD discussed previously is to use the Grassmann geometry. Thus, the Grassmann geometry can be used in our context by considering in eq. (4.2) the subspaces spanned by the column factors and by considering the Gaussian-Grassmann kernel defined in section 3.3.4. In this case, the chordal distance between the two subspaces defined in eq.(3.28) will be used. Due to the fact that the projectors are scale-invariant to any non-zero multiplicative scaling, we can avoid the scaling ambiguity inherent in CPD using the Grassmann kernel. We call this approach Tensor Learning on Grassmann Manifold (TL-OGA).
As the ALS optimization strategy used in [72] to compute the CPD suffers greatly from the curse of dimensionality in the sense that its complexity grows exponentially with the order of the input tensor, we use the JIRAFE [151] as an efficient method to mitigate it for the computation of the CP factors wich is based on the exploitation of an algebraic relation between the CPD and the Tensor-Train Decomposition TTD. When the CPD is derived using the JIRAFE algorithm, we

call this approach Fast Tensor Learning on Grassmann Manifold (FTL-OGA).

4.3 Numerical Experiments

This section compares the state-of-the-art method DuSK [72] with the two proposed approaches (TL-OGA and FTL-OGA).

4.3.1 Datasets

- **UCF11 dataset** [74]: In this dataset, there are 1600 video clips some of which show diving, trampoling jumping, walking, shooting... . We randomly chose two human actions, "Trampoling jumping" and "Walking" for human activity recognition. This is a binary classification problem in which the 2 classes are given by the two human actions considered. The clips have a frame rate of 29.97 Frames Per Second (FPS) and each video has only one action associated with it. From each video, we consider a sequence that contains the first 240 frames from each clip video where the resolution of each Red-Green-Blue (RGB) frame is 320×240. The clip videos are organized as tensors of order 4 with dimensions $240 \times 240 \times 320 \times 3$.

- **Extended Yale dataset B** [9]: There are 28 human subjects in this dataset. For each subject, there are 576 images of size 480×640 taken under 9 poses and each pose is taken under 64 different illuminations. The images of three subjects are randomly chosen for subject recognition. This is a multiclass classification problem where the classes are given by the three subjects considered. We can arrange the images of each subject into a 4-th order tensor with dimensions $9 \times 480 \times 640 \times 64$ where 9 is the number of poses and 64 is the number of illuminations. The training and the test set are constructed by breaking the tensor of each subject into 16 tensors and considering each 4 illuminations in a tensor of size $9 \times 480 \times 640 \times 4$.

4.3.2 Classification performance

- The number of rank-one tensors used to compute the CPD will be denoted by R.
- To begin with, the dataset is divided into a training set and a test set. If it is not stated otherwise, the training set contains 60% of the data and the test set contains the remainder. Each training and test set was randomly sampled ten times, and the accuracy scores were calculated using the accuracy score, a classical metric for evaluating classification performance, which is a measure of the percentage of well classified data in a set of test data.
- We remark that both TL-OGA and FTL-OGA offer superior performance for various possible values of R for the two real-world datasets considered (see Tables 4.2 and 4.3). Our approaches are successful because the Grassmann kernel allows for circumvention of scaling ambiguities due to the CPD.
- Using FTL-OGA in Table 4.2, accuracy scores are slightly better that TL-OGA. This is because TL-OGA may suffer from some convergence difficulties problems related to ALS. This is not the case of the FTL-OGA that uses the JIRAFE algorithm instead.
- In Table 4.4, we note that TL-OGA and FTL-OGA are robust to a small training dataset such that, respectively, they reach high accuracy scores of 97% and 95% when only 20% of the dataset are used as training data, compared to 55% obtained using the DuSK method.

4.3.3 Hyperparameters settings

The couple of hyperparameters of STM are tuned using grid search from the grid $\{2^{-9}, 2^{-8}, ..., 2^{8}, 2^{9}\}$ by a 5-fold cross-validation. We thus divided the training set into 5 parts. Each pair of hyperparameters is used to learn on one subset

each time, and the accuracy score is calculated based on the remaining subsets. Based on the average score, we can establish a prediction of the model's performance. The pair of hyperparameters chosen are the ones with the highest average validation score.

4.3.4 Parameter sensitivity

For the computation of the CPD, different small values (values smaller than the smallest dimension of the tensor) of R were tested. When R is small, the factors may not be able to model the given tensor with enough accuracy. However, when it is too large, the factors can be poorly estimated. This can explain the variation in scores of TL-OGA in Table 4.2 w.r.t R if we assume that a low-rank approximation exists. On the other hand, the difficulty of DuSK to reach high scores on the UCF11 dataset can be explained by the impact of the scaling ambiguities of CPD on finding accurate boundaries for classification. In Table. 4.3, we remark that TL-OGA and FTL-OGA are both robust to the choice of the rank.

4.3.5 Time computation of the CPD

The major part of time computation of DuSK and TL-OGA is due to the computation of the CPD. Time computation of these two methods are reported in tables 4.5 and 4.6. This illustrates the benefit of using the JIRAFE method. For example, the gain is around a factor of 200 when $R = 2$ and a factor of 400 when $R = 3$ on the UCF11 dataset while on the Extended Yale dataset, the gain is around 125 when $R = 3$ and a factor of 75 when $R = 4$. With FTL-OGA, similar scores can be reached while computing CPD takes less time.

R	1	2	3
DuSK	$0.65(10^{-2})$	$0.57(10^{-2})$	$0.63(10^{-2})$
TL-OGA	$\mathbf{0.82}10^{-2})$	$\mathbf{0.84}(10^{-2})$	$\mathbf{0.72}(10^{-2})$
FTL-OGA	$\mathbf{0.86}(10^{-2})$	$\mathbf{0.86}(10^{-2})$	$\mathbf{0.81}(10^{-2})$

Table 4.2 – Average accuracy scores for the different methods on the UCF11 dataset according to the rank of input tensors: mean(standard deviation)

R	1	2	3	4
DuSK	0.88(0.12)	$0.9(10^{-2})$	$0.77(10^{-2})$	$0.93(10^{-2})$
TL-OGA	$\mathbf{1}(0)$	$\mathbf{1}(0)$	$\mathbf{1}(0)$	$\mathbf{1}(0)$
FTL-OGA	$\mathbf{1}(0)$	$\mathbf{0.99}(10^{-2})$	$\mathbf{0.99}(10^{-2})$	$\mathbf{1}(0)$

Table 4.3 – Average accuracy scores using different methods on the Extended yale dataset B with respect to R: mean(standard deviation)

% Train	20 %	40%	60%	80%
DuSK	$0.55(10^{-2})$	$0.69(10^{-2})$	$0.93(10^{-2})$	$0.88(10^{-2})$
TL-OGA	$\mathbf{0.97}(10^{-2})$	$\mathbf{0.99}(10^{-3})$	$\mathbf{1}(0)$	$\mathbf{1}(0)$
FTL-OGA	$\mathbf{0.95}(10^{-2})$	$\mathbf{0.99}(10^{-3})$	$\mathbf{1}(0)$	$\mathbf{1}(0)$

Table 4.4 – Average accuracy scores using different methods on the Extended yale dataset B with respect to the percentage of the training set: mean(standard deviation)

R	1	2	3
ALS(s)	470	6264	8887
JIRAFE(s)	**14**	**22**	**30**
Gain	33	283	444

Table 4.5 – Computational time in seconds for computeing the CPD of the UCF11 dataset using the JIRAFE algorithm compared with the ALS algorithm and the gain in time using JIRAFE

R	1	2	3	4
ALS(s)	49	760	1251	1455
JIRAFE(s)	**3**	**9**	**11**	**20**
Gain	16	84	113	73

Table 4.6 – Computational time in seconds for computing the CPD for the Extended yale dataset B using JIRAFE algorithm compared with ALS algorithm and the gain in time using JIRAFE

4.4 Conclusion

A statistical analysis of the popular DuSK method for supervised learning of high-order tensors is derived. By using the ALS optimization method, the DuSK method derives the factors of the CPD of the data tensor. Then, the GE is exploited as a similarity metric between the columns of the CP factors in the context of STM. The present work shows that the DuSK approach fails to verify the intrinsic property of similarity of a kernel function. Due to scaling ambiguities in CPD, the GE kernel was unable to achieve high accuracy classification scores on real datasets. Therefore, the approach TL-OGA is proposed in this work which uses a Gaussian-Grassmann kernel between the subspaces spanned by the CP factors which is invariant to the scaling ambiguities of the CPD. The FTL-OGA proposed relies on the JIRAFE method to alleviate the "curse of dimensionality" for the computation of the CP factors. Finally, it is shown on two real tensor datasets the ability of the proposed method to reach high classification accuracy scores while consuming much less computational power.

Appendices

.1 Proof of Theorem 1

The two following lemmas are required for the proofs of Theorem 1 and Theorem 2.

Lemma 1. *If L is a random variable with an (N,p)-binomial distribution, then:*

$$\mathbb{P}(L \equiv 0[2]) = \frac{1+(1-2p)^n}{2}.$$

Proof. Using the classical binomial sums:

$$\begin{aligned}
((1-p)+p)^N &= \sum_{n=1}^{N} \binom{N}{n} p^l (1-p)^{N-n}, \\
&= \sum_{n\equiv 0[2]} \mathbb{P}(L=n) + \sum_{n\equiv 1[2]} \mathbb{P}(L=n). \\
((1-p)-p)^N &= \sum_{n=1}^{N} \binom{N}{n} (-p)^n (1-p)^{N-n}, \\
&= \sum_{n\equiv 0[2]} \binom{N}{n} p^l (1-p)^{N-n} - \sum_{n\equiv 1[2]} \binom{N}{n} p^n (1-p)^{N-n}, \\
&= \sum_{n\equiv 0[2]} \mathbb{P}(L=n) - \sum_{n\equiv 1[2]} \mathbb{P}(L=n).
\end{aligned}$$

By summing the two expressions, we have:

$$2 \sum_{n\equiv 0[2]} \mathbb{P}(L=n) = \frac{1+(1-2p)^N}{2}.$$

Hence,

$$\begin{aligned}
\mathbb{P}(L \equiv 0[2]) &= \sum_{n\equiv 0[2]} \mathbb{P}(L=n), \\
&= \frac{1+(1-2p)^N}{2}.
\end{aligned}$$

Recall that Q_r was modelized as an $(N,1/2)$ binomial random variable belonging

to a truncated support, which leaves out odd values. Thus, by normalizing the truncated distribution, we have:

$$\mathbb{P}(Q_r = l) = \frac{2}{1+(1-2p)^N}\binom{N}{l}p^N(1-p)^{N-l}. \tag{12}$$

□

Lemma 2. *Let $N \in \mathbb{N}$, we have:*

$$\binom{N}{n}2^{n-1} \leq N^n \text{ for } n \in \{1,\cdots,N\}.$$

Proof. Let us show recursively the property :

$$\mathcal{P}(n) : \binom{N}{n}2^{n-1} \leq N^n \text{ for } n \in \{1,\cdots,N\}.$$

- $\mathcal{P}(0)$ is true for $n = 0$ since we have:

$$\binom{N}{0}2^{-1} = 1 \times \frac{1}{2} = \frac{1}{2} < 1 = N^0.$$

- $\mathcal{P}(n) \Rightarrow \mathcal{P}(n+1)$: Assume that $\mathcal{P}(n)$ is true for $n \in \{1,\cdots,N-1\}$. Hence,

$$\begin{aligned}\binom{N}{n+1}2^n &= \underbrace{\frac{N!}{(n+1)!(N-n-1)!}2^{n-1}}_{\leq N^n} \times \underbrace{\frac{2(N-n)}{n+1}}_{\leq N}, \\ &\leq N^{n+1}.\end{aligned}$$

□

Proof of Theorem 1:
Let $\mathcal{X}$ be a rank-R tensor. Consider two CPDs of the same tensor $\mathcal{X}$ with respective scaling factors $\{\beta_{n,r}\}_{n,r}$ and $\{\beta'_{n,r}\}_{n,r}$. Recall that $\alpha_{n,r} = \operatorname{sgn}\left(\frac{\beta_{n,r}}{\beta_{n,r'}}\right)$, we will first

show that $\mathbb{P}(k_{\text{DuSK}}(\mathcal{X},\mathcal{X}) = 1)$ tends toward 0 by demonstrating that $\exists C_R > 0$:

$$\mathbb{P}(k_{\text{DuSK}}(\mathcal{X},\mathcal{X}) = 1) \leq \mathbb{P}\left(\prod_{r=1}^{R} Q_r = C_R\right) \underset{N\to+\infty}{\longrightarrow} 0. \tag{13}$$

Proof.

$$k_{\text{DuSK}}(\mathcal{X},\mathcal{X}) = 1, \tag{14}$$

$$\iff \exp\left(\prod_{r=1}^{R}\left(-\gamma \sum_{n=1}^{N} \gamma(1-\beta_{n,r})^2\right)\right) = R - T_{N,R}, \tag{15}$$

$$\iff \prod_{r=1}^{R} Q_r = \frac{ln(R-T_{N,R})}{(-1)^R(4\gamma)^R}. \tag{16}$$

Since $\prod_{r=1}^{R} Q_r$ is a positive quantity, the term $\ln(R-T_{N,R})(-1)^R(4\gamma)^R$ should be positive, if it is not the case, $\mathbb{P}(k_{\text{DuSK}}(\mathcal{X},\mathcal{X}) = 1) = 0$. We suppose then that $\frac{\ln(R-T_{N,R})}{(-1)^R(4\gamma)^R} \geq 0$. Hence, we have two cases: $R-T_{N,R} > 1$ and R is even or $R-T_{N,R} < 1$ and R is odd. In both cases, we will be able to have an upper bound of $\frac{ln(R-T_{N,R})}{(-1)^R(4\gamma)^R}$ that depends on R. Let us focus on the first case and the same reasoning will be the same for the second. From eq.(16), we deduce that:

$$\mathbb{P}(k_{\text{DuSK}}(\mathcal{X},\mathcal{X}) = 1) = \mathbb{P}\left(\prod_{r=1}^{R} N_r = \frac{ln(R-T_{N,R})}{(4\gamma)^R}\right). \tag{17}$$

Let $C_R = [\frac{\ln(R)}{(4\gamma)^R}]$. Since $T_{N,R} > 0$, we have the following inclusion property:

$$\frac{ln(R-T_{N,R})}{(4\gamma)^R} \leq C_R. \tag{18}$$

Thus, using eq. (18), we have the following inclusion property:

$$\underbrace{\prod_{r=1}^{R} Q_r = \frac{ln(R-T_{N,R})}{(4\gamma)^R}}_{\text{event } E} \subset \underbrace{\prod_{r=1}^{R} Q_r \leq C_R}_{\text{event } F}. \tag{19}$$

Let E_c be the complementary event of E in F such as $\mathbb{P}(F) = \mathbb{P}(E \cup E_c)$. As E and E_c are mutually exclusive events, the intersection is null and we have $\mathbb{P}(F) = \mathbb{P}(E) + \mathbb{P}(E_c) \geq \mathbb{P}(E)$ or equivalently:

$$\mathbb{P}\left(\prod_{r=1}^{R} N_r = \frac{ln(R - T_{N,R})}{(4\gamma)^R}\right) \leq \mathbb{P}\left(\prod_{r=1}^{R} Q_r \leq M_R\right). \tag{20}$$

- Let us first assume that $\prod_{r=1}^{R} Q_r \geq 1$ i.e $Q_r \geq 1$ for $1 \leq r \leq R$, and the case $\prod_{r=1}^{R} Q_r = 0$ will be treated just after. Hence,

$$\prod_{r=1}^{R} Q_r \leq C_R \subset \forall r : Q_r \leq C_R. \tag{21}$$

Using the same methodology for deriving eq.(20) from eq.(19), we obtain:

$$\mathbb{P}\left(\prod_{r=1}^{R} Q_r \leq C_R\right) \leq \mathbb{P}\left(\forall r : Q_r \leq C_R\right). \tag{22}$$

As $(Q_r)_{1 \leq r \leq R}$ are assumed to be independent and identically distributed, hence,

$$\mathbb{P}\left(\forall r : Q_r \leq C_R\right) = \prod_{r=1}^{R} \mathbb{P}\left(Q_r \leq C_R\right). \tag{23}$$

From eq. (17), (20)-(23), we deduce that:

$$\mathbb{P}\left(k_{\mathrm{DuSK}}(\mathcal{X}, \mathcal{X}) = 1\right) \leq \prod_{r=1}^{R} \mathbb{P}(N_r \leq M_R). \tag{24}$$

Assume that Q_r is an $(N, \frac{1}{2})$ binomial distribution with truncated odd values, so we have from eq. (12) of lemma 1:

$$\mathbb{P}(Q_r \leq C_R) = \sum_{l=1}^{C_R} \mathbb{P}\left(Q_r = l\right) = 2 \sum_{l=1}^{C_R} \binom{N}{l} \frac{1}{2^N}. \tag{25}$$

Using lemma 2, we have from (25):

$$\mathbb{P}(Q_r \leq C_R) \leq \sum_{k=1}^{C_R} \frac{N^l}{2^l} \frac{1}{2^N} \leq C'_R N^{C_R} \frac{1}{2^N}, \tag{26}$$

where $C'_R = \sum_{l=1}^{C_R} \frac{1}{2^l}$.
Hence $\mathbb{P}(Q_r \leq C_R)$ converges toward 0 with N.

- In the case where $\prod_{r=1}^{R} Q_r = 0$ then (17) becomes:

$$\mathbb{P}\left(k_{\text{DuSK}}(\mathcal{X}, \mathcal{X}) = 1\right) = \mathbb{P}\left(\prod_{r=1}^{R} Q_r = 0\right). \tag{27}$$

Since we have,

$$\prod_{r=1}^{R} Q_r = 0 \subset \exists r_0 : Q_{r_0} = 0, \tag{28}$$

Using (28) and lemma 1, we have:

$$\mathbb{P}\left(\prod_{r=1}^{R} Q_r = 0\right) \leq \mathbb{P}\left(Q_{r_0} = 0\right) = \frac{1}{2^{N-1}},$$

which also converges towards 0 with N.

So far, we have demonstrated that $\mathbb{P}\left(k_{\text{DuSK}}(\mathcal{X}, \mathcal{X}) = 1\right)$ tends toward 0. To see that $\mathbb{P}(\mathfrak{X}_{k_{\text{DuSK}}} \neq \varnothing)$ tends toward 0, consider a tensor $\mathcal{Y}$ that is similar to $\mathcal{X}$ but different from $\mathcal{X}$. From the properties of a kernel function, we can derive:

$$k(\mathcal{X}, \mathcal{Y}) \leq k(\mathcal{X}, \mathcal{X}). \tag{29}$$

Since $\mathbb{P}\left(k_{\text{DuSK}}(\mathcal{X}, \mathcal{X}) = 1\right)$ tends towards 0, the quantity $\mathbb{P}\left(k_{\text{DuSK}}(\mathcal{X}, \mathcal{Y}) = 1\right)$ will also tend towards 0. Hence, $\mathbb{P}(\mathfrak{X}_{k_{\text{DuSK}}} \neq \varnothing)$ tends towards 0. □

.2 Proof of Theorem 2

Let us demonstrate the first part of for Theorem 2 given by:

$$\mathbb{P}(\mathfrak{X}_{k_{\text{DuSK}}} \neq \varnothing) = \mathbb{P}(Q_1 = 0) = \frac{1}{2^{N-1}}. \tag{30}$$

Proof. Similarly to .1, we show that $\mathbb{P}(k_{\text{DuSK}}(\mathcal{X},\mathcal{X}) = 1)$ tends towards 0. Let $\mathcal{X}$ a rank-one tensor where two CPDs are computed with respective scaling factors $\{\beta_{n,1}\}_{n,1}$ and $\{\beta'_{n,1}\}_{n,1}$. Then, we have:

$$\mathbb{P}(k_{\text{DuSK}}(\mathcal{X},\mathcal{X}) = 1) = \mathbb{P}\left(\sum_{n=1}^{N} \frac{(1-\text{sgn}(\alpha_n))^2}{4} = 0\right).$$

Assuming that Q_1 is a binomial distribution with parameter $p = 1/2$ with truncated support by removing all odd values, we have from eq.(12):

$$\mathbb{P}(Q_1 = 0) = \frac{1}{2^{N-1}}. \tag{31}$$

The second part of this proof will demonstrate that the expectation $\mathbb{E}\left[k_{\text{DuSK}}(\mathcal{X},\mathcal{X})\right]$ converges towards 0 with respect to the order N.
The expectation for k_{DuSK} can be written as:

$$\begin{aligned}
\mathbb{E}[k_{\mathrm{DuSK}}(\mathcal{X},\mathcal{X})] &= \mathbb{E}\left[\exp\left(-\gamma \sum_{n=1}^{N} (1-\operatorname{sgn}(\alpha_n))^2\right)\right], \\
&= \sum_{k=0}^{N} e^{-4\gamma k} \mathbb{P}\left(\sum_{n=1}^{N} \frac{(1-\operatorname{sgn}(\alpha_n))^2}{4} = k\right), \\
&= \sum_{k\equiv 0[2]} e^{-4\gamma k} \frac{1}{2^{n-1}} \binom{N}{k}, \\
&\leq \sum_{k=0}^{N} e^{-4\gamma k} \frac{1}{2^{N-1}} \binom{N}{k}, \\
&= 2\left(\frac{1+e^{-4\gamma}}{2}\right)^{N}.
\end{aligned}$$

Therefore, for any fixed $\gamma > 0$, the expectation converges towards 0 with respect to N.

□

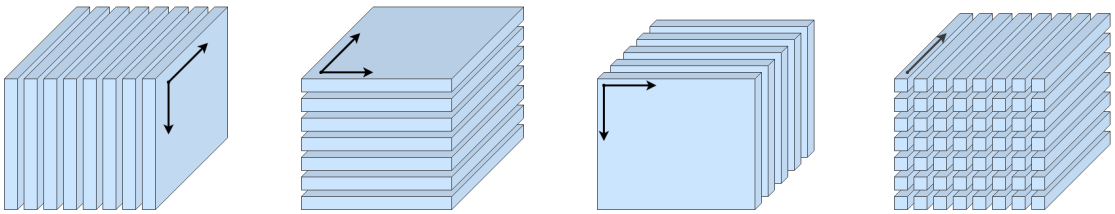
Chapter 5

Kernel based on Tubal-Singular Space of Tensor Train Cores

Outline of the current chapter

TTD is one of the simplest tensor networks that is capable of mitigating the curse of dimensionality. It has been introduced in the numerical mathematics community in [43] and under the name 'Matrix Product States' in the particle physics community [121]. TTD decomposes a tensor into lower-order tensors called TT-cores.

In this chapter, a kernel function on the tensor space will be defined based on the TTD. The similarity between two tensors will be determined by evaluating the similarities between their respective TT-cores. The non-unicity of the TT-cores could, however, adversely affects this measure. In order to overcome this problem, similarities between the subspaces spanned by the TT-cores will be



Vertical: $\mathcal{X}(:,j,:)$ Horizontal: $\mathcal{X}(i,:,:)$ Frontal: $\mathcal{X}(:,:,k)$ Tubal: $\mathcal{X}(i,j,:)$

Figure 5.1 – Slices of a tensor $\mathcal{X}$ of order-3.

considered. To characterize these subspaces, tools from the algebra of third order tensors known as tensor-linear algebra (t-algebra) will be used. This chapter is organized as follows. To begin with, we present some preliminary remarks concerning t-algebra in section 5.1. We then present the TTD in section 5.2. Lastly, in section 5.3, the proposed method is described, and its validity is demonstrated in section 5.4.

5.1 Background in t-algebra

A number of algebraic tools are presented in this section that generalize linear algebra for tensors of order 3 proposed in [84, 85]. First of all, some notations are introduced. Tensor slices are two-dimensional sections of a tensor. They are defined by all indices except two must. A third order tensor $\mathcal{X}$ has vertical, horizontal, and frontal slices, which are denoted by $\mathcal{X}(i_1,:,:)$, $\mathcal{X}(:,i_2,:)$, and $\mathcal{X}(:,:,i_3)$. We will refer to this later as $\boldsymbol{X}_{i_3}$.
Figure 5.1 illustrates the different slices and tubes of a 3-order tensor.
To define the tensor product between two tensors of order 3 which preserves the order of the tensor, we will first define some neccesary tools.
The block circulant matrix $\text{circ}(\mathcal{X})$ of a tensor $\mathcal{X}$ is of size $I_1 \times I_2 \times I_3$ and is defined

using its frontal slices $\boldsymbol{X}_{i_3}$:

$$\text{circ}(\mathcal{X}) = \begin{bmatrix} \boldsymbol{X}_1 & \boldsymbol{X}_{I_3} & \boldsymbol{X}_{I_3-1} & \cdots & \boldsymbol{X}_2 \\ \boldsymbol{X}_2 & \boldsymbol{X}_1 & \boldsymbol{X}_{I_3} & \cdots & \boldsymbol{X}_3 \\ \vdots & \ddots & \ddots & \ddots & \vdots \\ \boldsymbol{X}_{I_3} & \boldsymbol{X}_{I_3-1} & \ddots & \boldsymbol{X}_2 & \boldsymbol{X}_1 \end{bmatrix}.$$

The MatVec operation takes a 3-rd order tensor as input $\mathcal{X}$ of size $I_1 \times I_2 \times I_3$ and returns a block matrix of size $I_1 I_3 \times I_2$:

$$\text{MatVec}(\mathcal{X}) = \begin{bmatrix} \boldsymbol{X}_1 \\ \boldsymbol{X}_2 \\ \vdots \\ \boldsymbol{X}_{I_3} \end{bmatrix}, \quad \text{fold}(\text{MatVec}(\mathcal{X})) = \mathcal{X},$$

where the fold is the inverse operation of MatVec($\mathcal{X}$) that takes back to the tensor $\mathcal{X}$. Finally, the t-product is defined by the following:

Definition 17. *Let $\mathcal{X}$ be a tensor of size $I_1 \times I_2 \times I_3$ and $\mathcal{Y}$ of size $I_2 \times I \times I_3$. Then, the t-product $\mathcal{X} * \mathcal{Y}$ is a tensor of size $I_1 \times I \times I_3$ defined as:*

$$\mathcal{X} * \mathcal{Y} = fold(circ(\mathcal{X}) \,.\, MatVec(\mathcal{Y})). \tag{5.1}$$

It shall be noted that the computation of the t-product in (5.1) demands $\mathcal{O}(I_1 I_2 I_3^2 I)$ operations. In practice, the t-product is realised in the Fourier domain.

Computation of the t-product in the Fourier domain

To illustrate how computations are performed in the Fourier domain, we first define the Discret Fourier Transform (DFT) of a tensor.

Definition 18. *(*DFT *of a tensor) For $\mathcal{X} \in \mathbb{R}^{I_1 \times I_2 \times I_3}$, its* DFT *is denoted by $\bar{\mathcal{X}}$ and it is the result of applying the Fast Fourier Tansform (*FFT*) on $\mathcal{X}$ on the tube fibers of $\mathcal{X}$.*

As stated in [84], block circulant matrices can be block diagonalized just as circulant matrices can be diagonalized by the DFT. Specifically, if $\boldsymbol{F}_{I_3}$ is the $I_3 \times I_3$

DFT matrix, then:

$$(\boldsymbol{F}_{I_3} \otimes \boldsymbol{I}_{I_1}) . \text{circ}(\mathcal{X}) . (\boldsymbol{F}^*_{I_3} \otimes \boldsymbol{I}_{I_2}) = \bar{\boldsymbol{X}}, \tag{5.2}$$

where $\bar{\boldsymbol{X}}$ is a block diagonal matrix where its i-th block on the diagonal is the frontal slice $\boldsymbol{X}_{i_3}$:

$$\bar{\boldsymbol{X}} = \text{bdiag}(\bar{\mathcal{X}}) = \begin{bmatrix} \bar{\boldsymbol{X}}_1 & & & \\ & \bar{\boldsymbol{X}}_2 & & \\ & & \ddots & \\ & & & \bar{\boldsymbol{X}}_{I_3} \end{bmatrix}. \tag{5.3}$$

To compute the t-product in eq. (5.1) in the Fourier domain, $\bar{\mathcal{X}}$ and $\bar{\mathcal{Y}}$ which are respectively the DFTs of $\mathcal{X}$ and $\mathcal{Y}$ are computed then it remains to multiply each frontal slice of $\bar{\mathcal{X}}$ with each frontal slice of $\bar{\mathcal{Y}}$. Specifically, we have [84]:

$$\mathcal{Z} = \mathcal{X} * \mathcal{Y} \iff \bar{Z} = \bar{\boldsymbol{X}}\bar{\boldsymbol{Y}}. \tag{5.4}$$

In the Fourier domain, the t-product will cost $\mathcal{O}(I_1 I_2 I_3 I \log_2(I_3))$ [84].

Other definitions from tensor linear algebra

Definition 19. ***(t-transpose)*** *For a tensor $\mathcal{X}$ of size $I_1 \times I_2 \times I_3$, its transpose $\mathcal{X}^T$ is a tensor of size $I_2 \times I_1 \times \boldsymbol{I}_3$ obtained by transposing the frontal slices $\boldsymbol{X}_{i_3}$ and reversing their order from 2 through I_3.*

Definition 20. ***(Identity tensor)****: The identity tensor $\mathcal{I}_{I_1 I_1 I_3}$ is a tensor whose first frontal slice is the identity matrix I_{I_1} and whose all other frontal slices are zeros.*

Definition (Tensor inverse) *: A tensor $\mathcal{X}$ of size $I_1 \times I_1 \times I_3$ has an inverse $\mathcal{X}^{-1}$ if:*

$$\mathcal{X} * \mathcal{X}^{-1} = \mathcal{X}^{-1} * \mathcal{X} = \mathcal{I}_{I_1 I_1 I_3}.$$

Definition 21. ***(Orthogonal tensor)****: A real tensor of size $I_1 \times I_1 \times I_3$ is orthogonal if:*

$$\mathcal{X}^T * \mathcal{X} = \mathcal{X} * \mathcal{X}^T = \mathcal{I}_{I_1 I_1 I_3}.$$

Definition 22. *(**f-diagonal tensor**): A tensor $\mathcal{X}$ of size $I_1 \times I_1 \times I_3$ is f-diagonal if each of its frontal slices $\boldsymbol{X}_{i_3}$ is a diagonal matrix.*

Definition 23. *(**Tubal scalar**): An element $c \in \mathbb{R}^{1\times 1\times I_3}$ is called a tubal scalar of length I_3.*

Definition 24. *(**Range**) The range of a tensor $\mathcal{X} \in \mathbb{R}^{I_1\times I_2\times I_3}$ denoted by $span(\mathcal{X})$ is the t-linear span of its lateral slices:*

$$span(\mathcal{X}) = \left\{ \mathcal{X} * \mathcal{C} = \sum_{i_2=1}^{I_2} \mathcal{X}(:, i_2, :) * c_{i_2},\ \mathcal{C} \in \mathbb{R}^{I_2\times 1\times I_3} \right\}, \tag{5.5}$$

where $c_{i_2} = \mathcal{C}(i_2, i_2, :)$ are the tube fibers of $\mathcal{C}$.

Tensor Singluar Value Decomponistion (t-SVD):

Based on the framework of t-linear algebra, the SVD has been generalized to a tensor of order 3 as follows:

Theorem 3. *[84] Let $\mathcal{X}$ a $I_1 \times I_2 \times I_3$ be a real-tensor. Then, $\mathcal{X}$ can be decomposed as:*

$$\mathcal{X} = \mathcal{U} * \mathcal{S} * \mathcal{V}^T, \tag{5.6}$$

where $\mathcal{U}$ and $\mathcal{V}$ are orthogonal tensors of size $I_1 \times I_1 \times I_3$ and $I_2 \times I_2 \times I_3$ respectively and $\mathcal{S}$ is a tensor f-diagonal. The entries of the diagonal of the first frontal slice $\mathcal{S}(:, :, 1)$ have the decreasing property:

$$\mathcal{S}(1, 1, 1) \geq \mathcal{S}(2, 2, 1) \geq \cdots \geq \mathcal{S}(I_{min}, I_{min}, 1) \geq 0. \tag{5.7}$$

where $I_{min} = \min(I_1, I_2)$.
The entries of the diagonal od $\mathcal{S}(:, :, 1)$ are called the singular values $\mathcal{X}$.

Computation of the t-SVD:

In order to compute the t-SVD of $\mathcal{X}$, $\bar{\boldsymbol{X}}$ is first computed, followed by SVDs of its frontal slices $\bar{\boldsymbol{X}}_{i_3} = \bar{\boldsymbol{U}}_{i_3} \bar{\boldsymbol{S}}_{i_3} \bar{\boldsymbol{V}}_{i_3}$, each of which constitutes the frontal slices of $\bar{\mathcal{U}}$,

$\bar{\mathcal{S}}$ and $\bar{\mathcal{V}}$. The tensors $\mathcal{U}$, $\mathcal{S}$, and $\mathcal{V}$ are obtained by computing the inverse fourier transform along the third dimension of $\bar{\mathcal{U}}$, $\bar{\mathcal{S}}$, and $\bar{\mathcal{V}}$, respectively.

If $I_2 < I_1$, The reduced t-SVD can be computed rather than the full t-SVD, in which case, $\bar{\boldsymbol{U}}_{i_3}$ are no longer orthognal but has $I_2 < I_1$ orthonormal columns. As a result, $\mathcal{U}$ will be partially orthogonal of size $I_1 \times I_2 \times I_3$, rather than orthogonal.

Definition 25. *(Tensor tubal rank) For $\mathcal{X} \in \mathbb{R}^{I_1 \times I_2 \times I_3}$, the tensor tubal rank denoted by R_t is defined as the number of non-zero tubes of $\mathcal{S}$ and can be determined by the first frontal slice $\mathcal{S}(:,:,1)$, i.e.,*

$$R_t = \#\{i, \mathcal{S}(i,i,1) \neq 0 \quad 1 \leq i \leq \min(I_1, I_2)\}. \tag{5.8}$$

Orthonormal basis for the range of a tensor:

The t-SVD in eq. (5.6) can also be written as:

$$\mathcal{X} = \sum_{i=1}^{I_{\min}} \mathcal{U}(:,i,:) * \mathcal{S}(i,i,:) * \mathcal{V}(:,i,:)^T. \tag{5.9}$$

In this sense, the t-SVD in eq. (5.9) can be viewed as a sum of outer products of matrices that generalizes the SVD obtained through a sum of outer products of vectors.
Furthermore, using eq. (5.9), for every $\mathcal{C} \in \mathbb{R}^{I_2 \times 1 \times I_3}$, we have:

$$\mathcal{X} * \mathcal{C} = \sum_{i=1}^{I_{\min}} \mathcal{U}(:,i,:) * \Big(\mathcal{S}(i,i,:) * \mathcal{V}(:,i,:)^T * \mathcal{C} \Big). \tag{5.10}$$

The term in parenthesis is a tubal scalar, thus the lateral slices of $\mathcal{U}$ provide an orthonormal basis for $span(\mathcal{X})$ and its dimension is equal to the tubal rank.
One of the most interesting features of the t-SVD is that it can be used to find an optimal approximation of a tensor in the following sense:

Theorem 4. *[84] Let $\mathcal{X}$ be given by its t-*SVD *form eq.*(5.9). *Then, for $r_t < min(I_1, I_2)$,*

define:

$$\mathcal{X}_{r_t} = \sum_{i=1}^{r_t} \mathcal{U}(:,i,:) * \mathcal{S}(i,i,:) * \mathcal{V}(:,i,:)^T,$$

Then,

$$\mathcal{X}_{r_t} = argmin_{\tilde{\mathcal{X}} \in S} \|\mathcal{X} - \tilde{\mathcal{X}}\|_F, \tag{5.11}$$

where $S = \{\mathcal{C} = \mathcal{X} * \mathcal{Y},\ \mathcal{X} \in \mathbb{R}^{I_1 \times I \times I_3},\ \mathcal{Y} \in \mathbb{R}^{I \times I_2 \times I_3}\}$.

$\mathcal{X}_{r_t}$ is then the *best tubal-rank-r_t* approximation to $\mathcal{X}$.

Definition 26. *(Pseudo-inverse) $\mathcal{P}$ is a projector if $\mathcal{P}^2 = \mathcal{P} * \mathcal{P} = \mathcal{P}$. If $\mathcal{X} \in span(\mathcal{P})$, then $\mathcal{P} * \mathcal{X} = \mathcal{X}$.*

For a tensor $\mathcal{X} \in \mathbb{R}^{I_1 \times I_2 \times I_3}$ whose frontal slices are full column rank with $I_2 < I_1$, its pseudo-inverse is defined as :

$$\mathcal{X}^\dagger = (\mathcal{X}^T * \mathcal{X})^{-1} * \mathcal{X}^T.$$

In this case, $\mathcal{P} = \mathcal{X} * \mathcal{X}^\dagger$ is an orthogonal projector onto the range of $\mathcal{X}$. Using the reduced t-SVD, $\mathcal{X} = \mathcal{U} * \mathcal{S} * \mathcal{V}$, where $\mathcal{U} \in \mathbb{R}^{I_1 \times I_2 \times I_3}$, then $\mathcal{P} = \mathcal{U} * \mathcal{U}^T$.

Next, we present generalization of Stiefel and Grassmann manifold for third order tensors under the t-product defined in [59]. First, we begin with a definition of the t-orthogonal group.

Definition 27. *(t-Orthogonal Group) The t-orthogonal group of tubal rank R_t is defined as:*

$$\mathbb{O}(R_t, R_t, I_3) = \left\{\mathcal{R} \in \mathbb{R}^{R_t \times R_t \times I_3} : \mathcal{R}^T * \mathcal{R} = \mathcal{R} * \mathcal{R}^T = \mathcal{I}_{R_t R_t I_3}\right\}. \tag{5.12}$$

Definition 28. *(t-Grassmann manifold) The t-Grassmann manifold $\mathbb{G}(I_1, R_t, I_3)$ is defined as follows:*

$$\mathbb{G}(I_1, R_t, I_3) = \left\{span(\mathcal{U}) : \mathcal{U} \in \mathbb{R}^{I_1 \times R_t \times I_3}, \mathcal{U}^T * \mathcal{U} = \mathcal{I}_{R_t R_t I_3}\right\}, \tag{5.13}$$

Each element of $\mathbb{G}(I_1, R_t, I_3)$ can be identified with an equivalence class of orthonormal basis that span the same subspace. Theses orthonormal basis are elements of the so-called t-Stiefel manifold defined by:

$$\mathbb{S}(I_1, R_t, I_3) = \left\{ \mathcal{U} \in \mathbb{R}^{I_1 \times R_t \times I_3} : \mathcal{U}^T * \mathcal{U} = \mathcal{I}_{R_t R_t I_3} \right\}. \tag{5.14}$$

5.2 Tensor train decomposition (TTD)

We first begin with the definition of the tensor contraction product that will be used in the reprsentation of the TTD. The contraction product $\times_q^n$ of two tensors $\mathcal{X}$ of size $I_1 \times \cdots \times I_N$ and $\mathcal{Y}$ of size $J_1 \times \cdots \times J_Q$ with $I_n = J_q$ is a tensor of order $Q + N - 2$ defined by [4]:

$$\begin{aligned}(\mathcal{X} \times_n^q \mathcal{Y})(i_1, \ldots, i_{n-1}, i_{n+1}, \ldots, i_{N_1}, j_1, \ldots, j_{n-1}, j_{n+1}, \ldots, j_Q) \\ = \sum_{i=1}^{I_n} \mathcal{X}(i_1, \ldots, i_{n-1}, i, i_{n+1}, \ldots, i_N) \mathcal{Y}(j_1, \ldots, j_{n-1}, i, j_{n+1}, \ldots, j_Q).\end{aligned} \tag{5.15}$$

Definition 1. *A tensor $\mathcal{X}$ admits a* TTD *with TT-ranks $(R'_1, \ldots, R'_{N-1})$ if it can be expressed as:*

$$\mathcal{X}(i_1, \cdots, i_N) = \sum_{r_1, \cdots, r_N}^{R'_1, \ldots, R'_{N-1}} \boldsymbol{G}_1(i_1, r_1) \mathcal{G}_2(r_1, i_2, r_2) \cdots \mathcal{G}_{N-1}(r_{n-1}, i_n, r_n) \boldsymbol{G}_N(r_{N-1}, i_N), \tag{5.16}$$

where the size of each core is:

- $\boldsymbol{G}_1 \in \mathbb{R}^{I_1 \times R'_1}$,
- $\mathcal{G}_n \in \mathbb{R}^{R'_{n-1} \times I_n \times R'_n}$, $\forall n : 2 \leq n \leq N$,
- $\boldsymbol{G}_N \in \mathbb{R}^{R'_{N-1} \times I_N}$.

A tensor entry is evaluated by the product of the core tensors at the indices (r_{n-1}, i_n, r_n), for $n \geq 1$. Thereafter, the summation over the TT-ranks is performed.

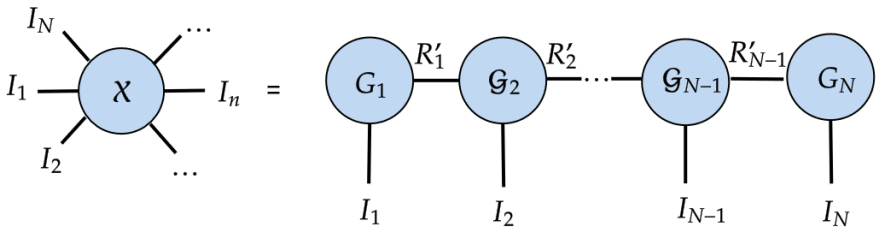


Figure 5.2 – TTD of a Q-order tensor with TT-ranks $(R'_1, \cdots, R'_{Q-1})$.

Eq. (1) *can be expressed in a compact tensor form using the contraction product as follows:*

$$\mathcal{X} = \boldsymbol{G}_1 \times_2^1 \mathcal{G}_2 \times_3^1 \times \cdots \times_{N-1}^1 \mathcal{G}_{N-1} \times_N^1 \boldsymbol{G}_N. \tag{5.17}$$

The number of parameters of the TTD is estimated as $O(NIR'^2_{\max} + (N-2)R'^3_{\max})$ where $R'_{\max}$ is the maximal TT-rank. The complexity of TTD is linear in N as well as the CPD. The advantage of TTD is that it has stable (non iterative) algorithm as we will describe thereafter.

Computation of TTD:

The TTD can be computed through a sequence of SVDs using the TT-SVD algorithm [43]. First, the R'_1-truncated SVD of $\boldsymbol{X}_{(1)}$: $\boldsymbol{X}_{(1)} = \boldsymbol{U}_1 \boldsymbol{S}_1 \boldsymbol{V}_1^T$. The core $\mathcal{G}_1$ is obtained by reshaping the matrix $\boldsymbol{U}_1$. Next, the matrix $\boldsymbol{S}_1 \boldsymbol{V}_1^T$ is reshaped and its R'_2-truncated SVD is calculated. Following this, the left singular matrix of the last SVD will be reshaped in order to obtain $\mathcal{G}_2$, and the product of its right singular vector by the matrix of its singular values will be reshaped for the next step. The computation of the next cores is carried out in a similar manner until all cores have been computed.
It shall be noted that there is always a best approximation to a tensor $\mathcal{X}$ in the Frobenius norm with TT-ranks bounded by R'_k, and the TT-approximation using the TT-SVD is quasi-optimal [43].

Non uniqueness of TTD:

It shall be noted that TTD is not unique. In fact, $\mathcal{X}$ can be written in a TTD format using different cores than those in eq. (5.17) as follows [151]:

$$\mathcal{X} = \boldsymbol{A}_1 \times_2^1 \mathcal{A}_2 \times_3^1 \cdots \times_{N-1}^1 \mathcal{A}_{N-1} \times_N^1 \boldsymbol{A}_N, \tag{5.18}$$

whith,

$$\boldsymbol{A}_1 = \boldsymbol{G}_1 M_1^{-1}, \tag{5.19}$$
$$\boldsymbol{A}_N = \boldsymbol{G}_{N-1}^{-1} \boldsymbol{G}_N, \tag{5.20}$$
$$\mathcal{A}_n = \boldsymbol{M}_{n-1} \times_2^1 \mathcal{G}_n \times_3^1 \boldsymbol{M}_n^{-1}, \tag{5.21}$$

where $\boldsymbol{M}_n$ are nonsingular matrices of dimension $R'_n \times R'_n$.

5.3 Proposed method

In this section, we describe the approach considered to define a tensorial kernel using TTD. This later will be given based on sub-kernels defined on TT-cores. Sepecifically, the similarity between two tensors will be given based on similarities between TT-cores. However, as TTD is not unique (see eq. (5.18)), comparing two tensors via their non-unique decomposition will lead to compare cores that are not similar. To overcome this problem, learning on the subspaces spanned by the TT-cores will be considered.
Let $\mathcal{X}, \mathcal{X}' \in \mathbb{R}^{I_1 \times \cdots \times I_N}$ with $\{\boldsymbol{G}_1, \mathcal{G}_2 \ldots \mathcal{G}_{N-1}, \boldsymbol{G}_N\}$ and $\{\boldsymbol{G}'_1, \mathcal{G}'_2 \ldots \mathcal{G}'_{N-1}, \boldsymbol{G}'_N\}$ being respectively the sets of TT-cores of $\mathcal{X}$ and $\mathcal{X}'$.
Since subspaces spanned by the first core and the transpose of the last core are invariant to the non-unicity involved by the TTD in eq. (5.19) and (5.20), learning on Grassmann manifold will be considered using the following gaussian

subkernels:

$$k_1(span(\boldsymbol{G}_1), span(\boldsymbol{G}'_1)) = \exp\left(-\gamma \|\boldsymbol{U}_1\boldsymbol{U}_1^T - \boldsymbol{U'}_1\boldsymbol{U'}_1^T\|_F^2\right), \tag{5.22}$$

$$k_N\left(span(\boldsymbol{G}_N^T), span(\boldsymbol{G'}_N^T)\right) = \exp\left(-\gamma \|\boldsymbol{U}_N\boldsymbol{U}_N^T - \boldsymbol{U'}_N\boldsymbol{U'}_N^T\|_F^2\right), \tag{5.23}$$

where,

- $span(\boldsymbol{G}_1), span(\boldsymbol{G}'_1) \in \mathbb{G}(I_1, R'_1)$,
- $span(\boldsymbol{G}_N^T), span(\boldsymbol{G}_N'^T) \in \mathbb{G}(I_N, R'_N)$,
- $\boldsymbol{U}_1, \boldsymbol{U}'_1$ are othonormal bases of $span(\boldsymbol{G}_1)$, $span(\boldsymbol{G}'_1)$,
- $\boldsymbol{U}_N, \boldsymbol{U}'_N$ are othonormal bases of $span(\boldsymbol{G}_N)$, $span(\boldsymbol{G}'_N)$.

For the third-order TT-cores, if the matrix $\boldsymbol{M}_n$ defined in eq.(5.21)) is reduced to the identity matrix, $\mathcal{A}_n$ and $\mathcal{G}_n$ will define the same projectors and hence span the same subspaces. In this case, subspaces spanned by the TT-cores of order 3 will be invariant to the multiplication of $\mathcal{G}_n$ from the left by $\boldsymbol{M}_{n-1}$ as involved in eq.(5.21). Hence for each $2 \leq n \leq N-1$, a subkernel k_n will be defined on the t-Grassmannien manifold $\mathbb{G}(I_n, R'_{n-1}, R'_n)$. The subkernel k_n can be considered as a generalization of a Grassmannien kernel. To define a Gaussian kernel on the t-Grassmann manifold, a metric between elements of this space should be defined. To do this, elements of the t-Grassmann manifold are first embedded in the space of $I_1 \times I_1 \times I_3$ symmetric tensors and the Frobenius norm is taken as the distance in the embedding space. Using this metric, the Gaussian kernel can be generalized on t-Grassmannien manifolds as follows:

$$k_n(\text{span}(\mathcal{G}_n), \text{span}(\mathcal{G}'_n)) = \exp\left(-\gamma \left\|\mathcal{U}_n * \mathcal{U}_n^T - \mathcal{U}'_n * \mathcal{U}_n'^T\right\|_F^2\right) \; 2 \leq n \leq N-1, \tag{5.24}$$

where $\mathcal{U}_n$, $\mathcal{U}'_n \in \mathbb{R}^{I_n \times R'_{n-1} \times R'_n}$ are orthonormal basis of $span(\mathcal{G}_n)$ and $span(\mathcal{G}'_n)$ that can be found using the t-SVD.
The kernel k_n is a valid Gaussian-Grassmann (GG) kernel since it is equal to a traditional Gaussian kernel with a Euclidean distance on the flattened projectors. Finally, the tensorial kernel proposed will be defined as the product of subkernels

defined on subspaces spanned by the TT-cores. Specifically, the kernel between $\mathcal{X}$ and $\mathcal{X}'$ is defined as follows:

$$k(\mathcal{X},\mathcal{Y}) = \prod_{n=1}^{N-1} \exp\left(-\gamma \left\|\mathcal{U}_n * \mathcal{U}_n^T - \mathcal{U'}_n * \mathcal{U'}_n^T\right\|_F^2\right). \\ \exp\left(-\gamma \left\|\boldsymbol{U}_1\boldsymbol{U}_1^T - \boldsymbol{U'}_1\boldsymbol{U'}_1^T\right\|_F^2\right). \exp\left(-\gamma \left\|\boldsymbol{U}_N\boldsymbol{U}_N^T - \boldsymbol{U'}_N\boldsymbol{U'}_N^T\right\|_F^2\right), \quad (5.25)$$

where $\gamma > 0$.

5.4 Experiments

In this section we conduct numerical experiments to validate the proposed approach which will be compared to the KSTTM approach [21]. This approach defines the similarity between two tensors by evaluating the similarities between the row fibers of their respective TT-cores.

Datasets:

We evaluate the proposed approach on three real datasets. We use the UCF11 and Extended Yale datasets defined in section 4.3 as well as the Faces96 dataset. This latter contains images of 119 subjects in Joint Photographic Group (JPG) format. There are 119 persons. Three subjects are randomly chosen for image recognition. This is a classification problem with three classes being the subjects. For every subject, there are 48 images of size $196 \times 196 \times 3$ taken under different positions from the camera. We can arrange the images of each subject into a 4^{th} order tensor with dimensions $196 \times 196 \times 3 \times 48$. The training and the test sets are constructed by breaking the tensor of each subject into tensors of size $196 \times 196 \times 3 \times 16$.

Classification performance:

- The procedure for fitting the hyperparameters of the STM, the split of the dataset as well as the accuracy score metric used in this section are the same as the ones described in section 4.3.

- Table 5.1 shows the classification results of the considered approaches. Here, a grid search has been realized using different values for the TT-ranks. It is clear that the proposed approach achieves the highest performance. This can be explained by the fact that the kernel proposed reduces the ambiguities caused by the non-unicity of TTD.

Parameter sensitivity:

Across different values of TT-ranks, the proposed approach consistently achieves high accuracy scores. Thus, small TT-ranks can be used for reducing calculation costs while remaining effective in terms of classification performance. We note, however, that the choice of TT-ranks has a significant impact on the performance of KSTTM. The highest accuracy scores of this latter are achieved for higher TT-ranks as shown in table 5.1 .

Computation time:

Table 5.2 shows the computation time of the kernel matrix for of the KSTTM method and for the proposed approach. KSTTM approach [21] gives the lowest cost while the proposed approach achieves reasonable results in terms of complexity because of additionnal cost of the projectors.

Dataset	TT-ranks	Our approach	[21]
UCF11	[1,1,1,1,1]	$\mathbf{0.98(10^{-2})}$	$0.67(10^{-1})$
	[1,1,1,2,1]	$\mathbf{0.99(10^{-2})}$	$0.68(10^{-1})$
	[1,1,1,3,1]	$\mathbf{0.98(10^{-2})}$	$0.86(10^{-1})$
	[1,1,2,2,1]	$\mathbf{0.99(10^{-2})}$	$0.88(10^{-1})$
	[1,2,2,2,1]	**1(0)**	$0.96(10^{-2})$
Faces96	[1,1,1,1,1]	$\mathbf{0.99(10^{-2})}$	$0.73(10^{-1})$
	[1,1,2,1,1]	$\mathbf{0.86(10^{-2})}$	$0.71(10^{-2})$
	[1,1,2,2,1]	$\mathbf{0.87(10^{-2})}$	$0.75(10^{-2})$
	[1,2,2,2,1]	$\mathbf{0.97(10^{-2})}$	$0.91(10^{-2})$

Table 5.1 – Accuracy scores (Mean accuracy (standard deviation)) for different TT-ranks.

Dataset/Method	KSTTM	Our approach
UCF11	**130**	148
Faces96	**0.32**	**0.32**

Table 5.2 – Computational time on seconds of different methods on the three real-world datasets considered.

5.5 Conclusion

A new kernel function on the tensor space is proposed in this chapter to address non linear classification problems in the context of supervised learning of higher order tensors. In particular, the similarity of two tensors is defined by evaluating similarities between their respective TT-cores using TTD. The kernel function proposed in this chapter is defined on the t-Grassmann manifold of the span of TT-cores in order to overcome the non-unicity of the TT-cores. In addition, the proposed approach improves the classification performance, even for small TT-ranks, on the different real-world datasets considered. The approach proposed in this chapter mitigates only a part of the ambiguities associated with TTD, and in the following chapter, a different approach is proposed to deal with the remaining ambiguities.

Chapter 6

Speeding Up Of Kernel-Based Learning For High-Order Tensors

Outline of the current chapter

The objective of this chapter is to propose a tensorial kernel function that mitigates the ambiguities associated with TTD. In this context, it will be demonstrated that the subspaces spanned by the second unfoldings of the TT-cores are invariant to the non-unicity of the TT-cores. Based on these subspaces, a tensorial kernel function will be defined. In addition, it will be shown that the kernel proposed in [90] which is based on the subspaces spanned by the HOSVD

factors can be equivalently computed using the proposed kernel in this work. It is due to the equivalence between the TD and the TTD introduced in [151] that this equivalence occurs. The HOSVD used in [90], however, suffers from the curse of dimensionality. Using TTD, the curse of dimensionality can be broken and the same subspaces can be retrieved.

6.1 Kernel on Grassmann manifold

6.1.1 Invariant subspaces to the non-unicity of the TTD

Recall from the previous chapter in section 5.2 that the TTD of a tensor $\mathcal{X}$ could be derived from different sets of TT-cores as follows:

$$\mathcal{X} = \boldsymbol{G}_1 \times_2^1 \mathcal{G}_2 \times_3^1 \cdots \times_{N-1}^1 \mathcal{G}_{N-1} \times_N^1 \boldsymbol{G}_N,$$
$$\mathcal{X} = \boldsymbol{A}_1 \times_2^1 \mathcal{A}_2 \times_3^1 \cdots \times_{N-1}^1 \mathcal{A}_{N-1} \times_N^1 \boldsymbol{A}_N,$$

with,

$$\boldsymbol{A}_1 = \boldsymbol{G}_1 \boldsymbol{M}_1^{-1}, \tag{6.1}$$
$$\boldsymbol{A}_N = \boldsymbol{M}_{N-1}^{-1} \boldsymbol{G}_N, \tag{6.2}$$
$$\mathcal{A}_n = \boldsymbol{M}_{n-1} \times_2^1 \mathcal{G}_n \times_3^1 \boldsymbol{M}_n^{-1}. \tag{6.3}$$

To see that the second unfoldings of $\mathcal{A}_n$ and $\mathcal{G}_n$ span the same subspaces, one can write eq.(6.3) in its following equivalent form :

$$\mathcal{A}_n = \mathcal{G}_n \times_1 \boldsymbol{M}_{n-1} \times_2 \boldsymbol{I}_2 \times_3 (\boldsymbol{M}_n^T)^{-1}. \tag{6.4}$$

Thus,

$$\boldsymbol{A}_{n(2)} = \boldsymbol{G}_{n(2)} \left(\boldsymbol{M}_{n-1} \otimes \boldsymbol{M}_n^{-1} \right)^T. \tag{6.5}$$

Therefore, $\boldsymbol{A}_{n(2)}$ and $\boldsymbol{G}_{n(2)}$ span the same subspaces which will be invariant to any pre-multiplication and post-multiplication ambiguities involved in eq. (6.3).

6.1.2 Tensorial kernel-based on TTD

Consider the TTD of two tensors $\mathcal{X}, \mathcal{X}' \in \mathbb{R}^{I_1 \times \cdots \times I_N}$:

$$\mathcal{X} = \boldsymbol{G}_1 \times_2^1 \mathcal{G}_2 \times_3^1 \cdots \times_{N-1}^1 \mathcal{G}_{N-1} \times_N^1 \boldsymbol{G}_N, \tag{6.6}$$

$$\mathcal{X}' = \boldsymbol{G}_1' \times_2^1 \mathcal{G}_2' \times_3^1 \cdots \times_{N-1}^1 \mathcal{G}_{N-1}' \times_N^1 \boldsymbol{G}_N', \tag{6.7}$$

Since the subspaces of the second unfoldings are invariant to the non-unicity of the TT-cores as shown in setion 6.1.1, learning on these subspaces can be considered. The following Grassmannien Gaussian kernel can be used as a similarity measure with the chordal distance defined in eq. (3.28) as follows:

$$k_n\left(span\left(\boldsymbol{G}_{n(2)}\right), span\left(\boldsymbol{G}_{n(2)}'\right)\right) = \exp\left(-\gamma \left\|\sin(\theta)\right\|^2\right), \ \ 2 \leq n \leq N-1, \tag{6.8}$$

where k_n is defined on the Grasmmann manifold $\mathbb{G}(I_2, R_n' R_{n-1}')$ given sufficient small TT-ranks and θ is the vector of principal angles bewteen $span(\boldsymbol{G}_{n(2)})$ between $span(\boldsymbol{G}_{n(2)}')$.

The kernel projection metric in the kernel k_n can also be computed using the projectors as follows (see eq. (3.28)):

$$k_n\left(span\left(\boldsymbol{G}_{n(2)}\right), span\left(\boldsymbol{G}_{n(2)}'\right)\right) = \exp\left(-\gamma \left\|\boldsymbol{U}_n \boldsymbol{U}_n^T - \boldsymbol{U'}_n \boldsymbol{U'}_n^T\right\|_F^2\right), \ \ 2 \leq n \leq N-1, \tag{6.9}$$

where $\boldsymbol{U}_n, \boldsymbol{U}_n' \in \mathbb{R}^{I_n \times R_n' R_{n-1}'}$ are respectively the left singular matrices of $\boldsymbol{G}_{n(2)}$ and $\boldsymbol{G}_{n(2)}'$.
For the first and the last cores, the subspaces spanned by $\boldsymbol{G}_1$ and $\boldsymbol{G}_N^T$ are invariant to the post-multiplication by any non singular matrix and thus the following subkernels will be used:

$$k_1\left(span(\boldsymbol{G}_1), span(\boldsymbol{G}_1')\right) = \exp\left(-\gamma \|\boldsymbol{U}_1 \boldsymbol{U}_1^T - \boldsymbol{U'}_1 \boldsymbol{U'}_1^T\|_F^2\right), \tag{6.10}$$

$$k_N\left(span(\boldsymbol{G}_N^T), span(\boldsymbol{G'}_N^T)\right) = \exp\left(-\gamma \|\boldsymbol{U}_N \boldsymbol{U}_N^T - \boldsymbol{U'}_N \boldsymbol{U'}_N^T\|_F^2\right), \tag{6.11}$$

where,

- $span(\boldsymbol{G}_1), span(\boldsymbol{G}'_1) \in \mathbb{G}(I_1, R'_1)$,
- $span(\boldsymbol{G}_N^T), span(\boldsymbol{G}_N'^T) \in \mathbb{G}(I_N, R'_N)$,
- $\boldsymbol{U}_1, \boldsymbol{U}'_1 \in \mathbb{R}^{I_1 \times R'_1}$ and $\boldsymbol{U}_N, \boldsymbol{U}'_N \in \mathbb{R}^{I_N \times R'_N}$ are respectively the left singular matrices of $\boldsymbol{G}_1, \boldsymbol{G}'_1, \boldsymbol{G}_N^T, \boldsymbol{G}_N'^T$.

The final expression of the kernel function is then given by:

$$k(\mathcal{X}, \mathcal{X}') = \prod_{n=1}^{N} \exp\left(-\gamma \left\| \boldsymbol{U}_n \boldsymbol{U}_n^T - \boldsymbol{U}'_n \boldsymbol{U}'^T_n \right\|_F^2\right). \tag{6.12}$$

6.2 Kernel based on HOSVD factors

In this section, we present the tensorial kernel proposed in eq.(6.12) which is based on HOSVD and then it will be shown that it can be derived using the second unfoldings of the TT-cores. We first define the TD and the HOSVD decompositions.

6.2.1 Higher-order Singular Value Decomposition (HOSVD)

The TD has a constrained format known as HOSVD. In the latter, the factors $\boldsymbol{T}_n$ are orthonormal and the core tensor $\mathcal{G}$ is all-orthogonal [65]. To compute the n-th HOSVD factor, [90] considers the R_n left dominant singular vectors of the n-th unfolding of $\mathcal{X}$. The complexity of the HOSVD for a cubic N-order tensor of size $I_1 \times \cdots \times I_N$ is evaluated to $O(N R_{\max} I^N)$ where $I_{\max} = \max_n\{I_n\}$ and $R_{\max} = \max_n\{R_n\}$ is the maximal multilinear rank. We can see that the HOSVD complexity grows linearly and exponentially with respect to the order N. For low-order tensor [136, 61], this complexity remains acceptable but this limitation becomes rapidly severe for high-order tensors ($N > 3$).

6.2.2 Tensor based Kernel on HOSVD factors

Consider the HOSVD of two tensors $\mathcal{X}, \mathcal{X}' \in \mathbb{R}^{I_1 \times \cdots \times I_N}$ with multilinear ranks $(R_1, \ldots, R_N)$:

$$\mathcal{X} = \mathcal{G} \times_1 \boldsymbol{T}_1 \times_2 \ldots \times_N \boldsymbol{T}_N, \tag{6.13}$$

$$\mathcal{X}' = \mathcal{H} \times_1 \boldsymbol{T}_1' \times_2 \ldots \times_N \boldsymbol{T}_N', \tag{6.14}$$

where $\boldsymbol{T}_n, \boldsymbol{T}_n'$ are of size $I_n \times R_n$, $1 \leq n \leq N$ and $\mathcal{G}, \mathcal{H}$ are the core tensors of size $R_1 \times \cdots \times R_N$.
The kernel-based part of the proposed method in [90] is given by:

$$k(\mathcal{X}, \mathcal{X}') = \prod_{n=1}^{N} k_n \left(span(\boldsymbol{T}_n), span(\boldsymbol{T}_n') \right), \tag{6.15}$$

where k_n is a Grassmann kernel defined on $\mathbb{G}(I_n, R_n)$.

6.2.3 Equivalent tensorial kernels

Recall the equivalence between TD and TTD presented in [151]. In fact, each tensor core extracted from the TTD follows a 3-order Tucker model with two latent matrices in its first and third dimensions. In the second dimension, there is the interesting property that the R_n left dominant singular vectors from the second unfolding span the same subspace as $\boldsymbol{T}_n$. Furthermore, $\text{span}(\boldsymbol{T}_1) = \text{span}(\boldsymbol{G}_1)$, $\text{span}(\boldsymbol{T}_N) = \text{span}(\boldsymbol{G}_N^T)$.
In addition, the TT-ranks are related to the multlinear ranks by the following relation:

$$R_n' = \min \left(\prod_{p=1}^{n} R_p, \prod_{p=n+1}^{N} R_p \right).$$

Thus, the TTD allows to recover the same subspaces spanned by the HOSVD factors. As a result, the kernel in eq.(6.15) could be derived using the kernel proposed in eq.(6.12) when considering the R_n left dominant singular basis vectors from the second unfoldings of TT-cores. However, while the complexity of TTD

is linear with the order of the tensor, the complexity of HOSVD is exponential. This kernel will speed up the approach based on HOSVD so that it will be designed by Fast Kernel Subspace Estimation based on Tensor Train decomposition (FAKSETT). The pseudocode is given by Algorithm 2 for use in the context of classification using STM.

Algorithme 2 : FAKSETT Algorithm

Input: Training dataset $\{\mathcal{X}_m \in \mathbb{R}^{I_1 \times \cdots \times I_N}, y_m \in \{-1, 1\}\}_{m=1}^{M}$, TT-ranks $\{R'_1, \cdots, R'_N\}$, performance trade-off C, width parameter γ.
Output: The learning parameters of the decision function in eq.(3.38).

1: Compute the TTD of training samples $\{\mathcal{X}_m\}_{m=1}^{M}$:

$$[\boldsymbol{G}_1^{(m)}, \mathcal{G}_2^{(m)}, \cdots, \mathcal{G}_{N-1}^{(m)}, \boldsymbol{G}_N^{(m)}] = TT - SVD(\mathcal{X}_m; R'_1, \cdots, R'_N).$$

2: Compute the orthonormal factors for each training sample $\mathcal{X}_m$:
3: **for** $n = 2, \cdots, N-1$ **do**
4: $\quad \boldsymbol{U}_n^{(m)} \leftarrow$ Matrix of the left singular vectors of $\mathcal{G}_n^{(m)}{}_{(2)}$.
5: **end for**
6: Construct the kernel matrix $\boldsymbol{K}(m_1, m_2) = k(\mathcal{X}_{m_1}, \mathcal{X}_{m_2})$ using eq.(6.12).
7: Determine the decision function by solving the dual of the optimisation problem in eq.(3.37) using the kernel matrix $\boldsymbol{K}$.

6.3 Numerical Experiments

In the context of classification using STM, we will first be comparing FAKSETT and the proposed approach in the previous chapter [105] to see the effect of completely mitigating the non-unicity of TT-cores on the classification performance. Next, we compare FAKSETT and [90] to see if they have equivalent performances.

Datasets:

The UCF11 and Extended Yale datasets defined in section 4.3 are used to perform binary classification and multiclass classification, respectively.

The procedure for fitting the hyperparameters of the STM, the split of the dataset as well as the accuracy score metric used in this section are the same as the ones described in section 4.3.

Classification performance:

- Table 6.2 shows the accuracy scores of FAKSETT as well as the one proposed in the precedent chapter [105] using the kernel defined in eq.(5.25). FAKSETT ameliorated the one proposed in [105]. This can be explained by the fact that FAKSETT completely mitigates the non-unicity of the TT-cores and thus the kernel proposed reflects well the similarity between the TT-cores. Both FAKSETT and [105] outperform [21] which does note take into account the non-uniqueness of the TT-cores into account in the kernel function.
- Table 6.1 show very close accuracy scores between FAKSETT and the method of [90] for classification tasks on both real datasets. This indicates that the FAKSETT method operates as efficiently as the state-of-art method.
- From table 6.1, it is clear that FAKSETT and [105] are robust to the choice of the TT-ranks while KSTTM achieves its highest scores for high-values of TT-ranks.

Computation time:

It is noticeable from Table 6.3 that FAKSETT reduces significantly the running time for the computation of the factors, despite working with only $Q = 4$ order tensors. Higher orders would lead to an even higher running time gain between the two methods.

6.4 Conclusion

Based on TTD, a tensorial kernel function is proposed which considers similarities between the subspaces spanned by the second unfoldings of the TT-cores. As a result, the non-unicity of the TT-cores is completely mitigated. By using this

Dataset	n-ranks	TT-ranks	[104]	[90]
UCF11	[1,1,1,1]	[1,1,1,1,1]	$\mathbf{0.98(10^{-2})}$	$\mathbf{0.98(10^{-2})}$
	[2,2,2,2]	[1,2,4,2,1]	$\mathbf{0.99(10^{-2})}$	$\mathbf{0.99(10^{-2})}$
	[3,3,3,3]	[1,3,9,3,1]	$\mathbf{0.99(10^{-2})}$	$\mathbf{0.99(10^{-2})}$
Extended Yale	[1,1,1,1]	[1,1,1,1,1]	$\mathbf{0.99(10^{-2})}$	$\mathbf{0.99(10^{-2})}$
	[2,2,2,2]	[1,2,4,2,1]	**1(0)**	**1(0)**
	[3,3,3,3]	[1,3,9,3,1]	**1(0)**	**1(0)**

Table 6.1 – Accuracy scores (Mean accuracy (standard deviation)) for different multi-linear ranks varying the size of the training set on the UCF11 database.

Dataset	TT-ranks	[105]	[104]	[21]
UCF11	[1,1,1,1,1]	$0.98(10^{-2})$	$\mathbf{0.98(10^{-2})}$	$0.67(10^{-1})$
	[1,1,1,2,1]	$0.99(10^{-2})$	$\mathbf{0.99(10^{-3})}$	$0.68(10^{-1})$
	[1,1,1,3,1]	$0.98(10^{-2})$	$\mathbf{0.99(10^{-2})}$	$0.86(10^{-1})$
	[1,1,2,2,1]	$0.99(10^{-2})$	$\mathbf{0.99(10^{-3})}$	$0.88(10^{-1})$
	[1,2,2,2,1]	**1(0)**	**1(0)**	$0.96(10^{-2})$
Faces96	[1,1,1,1,1]	$\mathbf{0.99(10^{-2})}$	$0.97(10^{-2})$	$0.73(10^{-1})$
	[1,1,2,1,1]	$0.86(10^{-2})$	$\mathbf{0.95(10^{-2})}$	$0.71(10^{-2})$
	[1,1,2,2,1]	$0.87(10^{-2})$	$\mathbf{0.96(10^{-2})}$	$0.75(10^{-2})$
	[1,2,2,2,1]	$\mathbf{0.97(10^{-2})}$	$\mathbf{0.97(10^{-2})}$	$0.91(10^{-2})$

Table 6.2 – Accuracy scores (Mean accuracy (standard deviation)) for different TT-ranks.

Database	n-ranks	TT-ranks	FAKSETT	[90]
UCF11	[1,1,1,1]	[1,1,1,1,1]	**24**	63
	[2,2,2,2]	[1,2,4,2,1]	**14**	69
	[3,3,3,3]	[1,3,9,3,1]	**15**	104
Extended Yale	[1,1,1,1]	[1,1,1,1,1]	**3**	9
	[1,2,2,1]	[1,2,4,2,1]	**2.56**	9.47
	[1,2,3,1]	[1,3,9,3,1]	**2.58**	9.34

Table 6.3 – Computational time in seconds for computing the factors using the HOSVD and TTD on the UCF11 and Extended Yale datasets.

kernel, it is possible to speed up the kernel-based approach on HOSVD factors which has been proposed in [90]. This is because the TTD allows to recover the subspaces spanned by the HOSVD. As a result, the curse of dimensionnality can be avoided, particularly for datasets associated with N-order tensors when $N > 3$. The two methods have been compared numerically. As a result, both methods exhibit approximately the same classification scores, but the method based on the TTD method mitigates the curse of dimensionnality and thus is more computationally efficient.

Chapter 7

Conclusion and Perspectives

Outline of the current chapter

7.1 Tensor canonical rank estimation

In the first part of this thesis, we addressed the joint estimation of the canonical rank and the CP factors for high-order tensors. Since the CPD is a special case of the TD, our proposed approach is modeled as a constrained TD with a first constraint to minimize the number of superdialgonal elements, and a 2-norm on the offdiagonals is used to find the diagonal structure of the CP core. Based on different simulations and on three real data sets, the proposed approach is found to be efficient in estimating the cannonical rank.

Perspectives

- Although the proposed approach estimates the true canonical rank , it is subject to the curse of dimensionnality in its computation. The curse of dimensionnality may be broken by designing an equivalent method which

estimates the canonical rank efficiently while breaking the curse of dimensionnality. This can be achieved by applying the TTD-CPD equivalence to a tensor, and then estimating the canonical rank from the TT-cores, since, as demonstrated in [151], the TT-cores have cannonical rank equal to R.

- FARAC assumes that a rank-R tensor is disturbed by a normally distributed noise, but from another perspective, other cost functions could be used when specific noise affects the rank-R tensor.
- Tensors that are real nonnegative may require rank-one tensors in their CPD to also be nonnegative. Further, for many signal processing applications, it is required that the CP factors have a particular structure [49]. The tensor rank in such cases may increase, just as in the nonnegative case [110], so estimating the canonical rank in these cases may be of interest.
- FARAC is robust to thresholding parameter choice, but it was not demonstrated how to specify it clearly. Thus, it is possible to deal with this issue. Since the Tucker core elements show interactions between the factors, using the norm of the offdiagonals is a promising method for estimating the goodness of the threshold parameter.

7.2 Tensor kernels and ambiguities of tensor decompositions

The second part of this thesis discusses kernel methods for tensorial data. Specifically, it is discussed how the scaling ambiguity of the CPD and the TTD affect the evaluation of the Gaussian radial basis function (RBF) kernel between factors of the decomposition used. Chapter 3 demonstrated that evaluating similarities between CP factors degraded the properties of the tensorial kernel function used in the state-of-the-art method. Due to the ambiguities of scaling of the CPD, the inherent similarity measure of the kernel function does not hold. It was shown theoretically that the kernel value of two CPDs of the same tensor tends toward zero as the order of the tensor increases. Therefore, the kernel matrix constructed based on similarities between data is affected, resulting in

difficulties when predicting labels for new data. Chapter 4 provides a tensorial kernel function using TTD. This study aims to assess the similarity between two tensors by comparing the TT cores of the two tensors. To minimize the impact of TTD's non-unicity on the evaluation of the kernel function, similarity between subspaces spanned by TT-cores is considered. These subspaces lie on a t-Grassmann manifold that can be seen as a generalization of the Grassmann manifold for third-order tensors. The tensor algebra of third-order tensors was used to define the metric that enables the use of RBF kernels in this space.

Since the subspaces spanned by the TT-cores are not invariant to right ambiguity, so the approach proposed in Chapter 4 minimizes the effect of ambiguities associated with TTD but does not mitigate all of them. This problem can be mitigated by taking into account the subspaces spanned by the second unfolding of the TT-cores which are completely invariant to the non-unicity of the TT-cores. As a result, the similarity of two tensors is defined as the similarity between the subspaces of their second unfoldings. Moreover, by using this kernel, the tensorial kernel function proposed in [90], which is based on factors of HOSVD, can be speeded up. When TTD is used, the same subspaces can be recovered as with HOSVD, while being computationally more efficient because TTD breaks the curse of dimensionality.

Perspectives

- In spite of the fact that kernel methods are very efficient, their complexity depends on the number of training examples and therefore can be extremely high when training data are large. In light of this, it would be beneficial to see how these methods can be sped up for large datasets

- Computing low-rank decompositions using the CPD in section 4.3 can be ill-posed. It is known that adding constraints such as nonnegativity converts ill-posed optimization problems into well-posed ones [141]. Thus, non-negative tensor factorizations can be considered since the datsets used naturally have nonnegative entries. Algorithms for computing non-

negative CPD can be found in [55].

- Computation of the tensor decompositions can be very expensive especially for large datasets with higher-order, one way to deal with that is to consider methods for reducing this cost, randomized SVDs for the TT-svd, HOSVD and T-SVD may be used.
- Another perspective could be to see how can the proposed algorithms proposed be adapted for online learning, *i.e.*, how can we avoid having to retrain all datasets when new data is obtained.

Bibliography

[1] A.Ben-Hur and J.Weston. "A user's guide to support vector machines". In: *Data mining techniques for the life sciences*. Springer, 2010, pp. 223–239.

[2] A.Bhaskara, M.Charikar, and A.Vijayaraghavan. "Uniqueness of tensor decompositions with applications to polynomial identifiability". In: *Conference on Learning Theory*. PMLR. 2014, pp. 742–778.

[3] A.Björck and G.Golub. "Numerical Methods for Computing Angles Between Linear Subspaces". In: *Mathematics of Computation* 27.123 (1973), pp. 579–594. ISSN: 00255718, 10886842.

[4] A.Cichocki. "Era of big data processing: A new approach via tensor networks and tensor decompositions". In: *arXiv preprint arXiv:1403.2048* (2014).

[5] A.Cichocki, A.Phan, Q.Zhao, N.Lee, I.Oseledets, M.Sugiyama, and D.Mandic. "Tensor networks for dimensionality reduction and large-scale optimizations: Part 2 applications and future perspectives". In: *arXiv preprint arXiv:1708.09165* (2017).

[6] A.Cichocki, D.Mandic, L.De Lathauwer, G.Zhou, Q.Zhao, C.Caiafa, and H.A.Phan. "Tensor Decompositions for Signal Processing Applications: From two-way to multiway component analysis". In: *IEEE Signal Processing Magazine* 32.2 (2015), pp. 145–163.

[7] A.Cichocki, N.Lee, I.Oseledets, H.A.Phan, Q.Zhao, and D.Mandic. "Tensor Networks for Dimensionality Reduction and Large-Scale Optimization: Part 1 Low-Rank Tensor Decompositions". In: *Found. Trends Mach. Learn.* 9.4–5 (2016), 249–429.

[8] A.Delman, T.Arias, and S.Smith. *The Geometry of Algorithms with Orthogonality Constraints*. 1998. arXiv: physics/9806030 [physics.comp-ph].

[9] A.Georghiades, P.Belhumeur, and D.Kriegman. "From Few to Many: Illumination Cone Models for Face Recognition under Variable Lighting and Pose". In: *IEEE Trans. Pattern Anal. Mach. Intelligence* 23.6 (2001), pp. 643–660.

[10] A.Gupta, M.Ayhan, and A.Maida. "Natural Image Bases to Represent Neuroimaging Data". In: *Proceedings of the 30th International Conference on International Conference on Machine Learning - Volume 28*. ICML'13. Atlanta, GA, USA: JMLR.org, 2013, III–987–III–994.

[11] A.Smola and B.Schölkopf. "A tutorial on support vector regression". In: *Statistics and computing* 14.3 (2004), pp. 199–222.

[12] A.Smola and B.Schölkopf. *Learning with kernels*. Vol. 4. Citeseer, 1998.

[13] N.Le Bihan and G.Ginolhac. "Subspace methods for 3D arrays". In: *Workshop on Physics in Signal and Image Processing (PSIP)*. 2001, pp. 359–364.

[14] N.Le Bihan and G.Ginolhac. "Three-mode data set analysis using higher order subspace method: application to sonar and seismo-acoustic signal processing". In: *Signal Processing* 84.5 (2004), pp. 919–942.

[15] B.Savas and L.Eldén. "Handwritten digit classification using higher order singular value decomposition". In: *Pattern recognition* 40.3 (2007), pp. 993–1003.

[16] B.Scholkopf, K.Sung, C.Burges, F.Girosi, P.Niyogi, T.Poggio, and V.Vapnik. "Comparing support vector machines with Gaussian kernels to radial basis function classifiers". In: *IEEE Transactions on Signal Processing* 45.11 (1997), pp. 2758–2765.

[17] B.Schölkopfa, A.Smola, and F.Bach. *Learning with kernels: support vector machines, regularization, optimization, and beyond*. MIT press, 2002.

[18] C.Appellof and E.Davidson. "Strategies for analyzing data from video fluorometric monitoring of liquid chromatographic effluents". In: *Analytical Chemistry* 53.13 (1981), pp. 2053–2056.

[19] C.Burges. "A tutorial on support vector machines for pattern recognition". In: *Data mining and knowledge discovery* 2.2 (1998), pp. 121–167.

[20] C.Chen, K.Batselier, C.Ko, and N.Wong. "A support tensor train machine". In: *2019 International Joint Conference on Neural Networks (IJCNN)*. IEEE. 2019, pp. 1–8.

[21] C.Chen, K.Batselier, W.Yu, and N.Wong. "Kernelized support tensor train machines". In: *Pattern Recognition* 122 (2022), p. 108337.

[22] C.Eckart and G.Young. "The approximation of one matrix by another of lower rank". In: *Psychometrika* 1.3 (1936), pp. 211–218.

[23] C.Hillar and L.Lim. "Most Tensor Problems Are NP-Hard". In: *Journal of the ACM (JACM)* 60.6 (Nov. 2013).

[24] *CORCONDIA (Core Consistency Diagnostic) implementation in Python*. URL: https://pypi.org/project/corcondia/ (visited on 03/28/2021).

[25] C.Sun, Z.Zhang, X.Luo, T.Guo, J.Qu, and B.Li. "Support vector machine-based Grassmann manifold distance for health monitoring of viscoelastic sandwich structure with material ageing". In: *Journal of Sound and Vibration* 368 (2016), pp. 249 –263.

[26] D.Brie, S.Miron, F.Caland, and C.Mustin. "An uniqueness condition for the 4-way CANDECOMP/PARAFAC model with collinear loadings in three modes". In: *2011 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE. 2011, pp. 4108–4111.

[27] D.Cheng, R.Peng, Y.Liu, and I.Perros. "SPALS: Fast Alternating Least Squares via Implicit Leverage Scores Sampling". In: *Advances in Neural Information Processing Systems 29: Annual Conference on Neural Information Processing Systems 2016, December 5-10, 2016, Barcelona, Spain*. 2016, pp. 721–729.

[28] D.Geand, X.Jiang, and Y.Ye. "A Note on the Complexity of Lp Minimization". In: *Math. Program.* 129.2 (2011), 285–299. ISSN: 0025-5610.

[29] D.Guimaraes and R.De Souza. "Simple and efficient algorithm for improving the MDL estimator of the number of sources". In: *Sensors* 14.10 (2014), pp. 19477–19492.

[30] D.Kingma and J.Ba. "Adam: A Method for Stochastic Optimization". In: *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*. Ed. by Yoshua Bengio and Yann LeCun. 2015.

[31] D.Kressner, M.Steinlechner, and B.Vandereycken. "Low-rank tensor completion by Riemannian optimization". In: *BIT Numerical Mathematics* 54.2 (2014), pp. 447–468.

[32] D.Tao, X.Li, W.Hu, S.Maybank, and X.Wu. "Supervised tensor learning". In: *Fifth IEEE International Conference on Data Mining (ICDM'05)*. IEEE. 2005, 8–pp.

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

[34] F.Miwakeichi, E.Martınez-Montes, P.Valdés-Sosa, N.Nishiyama, H.Mizuhara, and Y.Yamaguchi. "Decomposing EEG data into space–time–frequency components using parallel factor analysis". In: *NeuroImage* 22.3 (2004), pp. 1035–1045.

[35] G.Bergqvist and E.Larsson. "The Higher-Order Singular Value Decomposition: Theory and an Application [Lecture Notes]". In: *IEEE Signal Processing Magazine* 27.3 (2010), pp. 151–154.

[36] G.Boros and V.Moll. *Irresistible Integrals: Symbolics, Analysis and Experiments in the Evaluation of Integrals*. Cambridge University Press, 2004.

[37] H.Çetingül and R.Vidal. "Intrinsic mean shift for clustering on Stiefel and Grassmann manifolds". In: *2009 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR 2009), 20-25 June 2009, Miami, Florida, USA*. IEEE Computer Society, 2009, pp. 1896–1902.

[38] H.Kim, P.Howland, H. Park, and N.Christianini. "Dimension reduction in text classification with support vector machines". In: *Journal of machine learning research* 6.Jan (2005), pp. 37–53.

[39] H.Tan, Z.Ma, S.Zhang, Z.Zhan, B.Zhang, and C.Zhang. "Grassmann manifold for nearest points image set classification". In: *Pattern Recognition Letters* 68 (2015), pp. 190–196.

[40] H.Wang and N.Ahuja. "Facial expression decomposition". In: *Proceedings ninth IEEE international conference on computer vision*. IEEE. 2003, pp. 958–965.

[41] I.Kotsia and I.Patras. "Support tucker machines". In: *CVPR 2011*. IEEE. 2011, pp. 633–640.

[42] I.Kotsia, W.Guo, and I.Patras. "Higher rank support tensor machines for visual recognition". In: *Pattern Recognition* 45.12 (2012), pp. 4192–4203.

[43] I.Oseledets. "Tensor-Train Decomposition". In: *SIAM J. Sci. Comput.* 33.5 (Sept. 2011), 2295–2317. ISSN: 1064-8275.

[44] I.Oseledets and E.Tyrtyshnikov. "Breaking the curse of dimensionality, or how to use SVD in many dimensions". In: *SIAM Journal on Scientific Computing* 31.5 (2009), pp. 3744–3759.

[45] I.Tsang, J.Kwok, P.Cheung, and N.Christianini. "Core vector machines: Fast SVM training on very large data sets". In: *Journal of Machine Learning Research* 6.Apr (2005), pp. 363–392.

[46] J.Carroll and J. Chang. "Analysis of individual differences in multidimensional scaling via an N-way generalization of "Eckart-Young" decomposition". In: *Psychometrika* 35.3 (1970), pp. 283–319.

[47] J.Davis and I.Dhillon. "Structured metric learning for high dimensional problems". In: *Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining*. 2008, pp. 195–203.

[48] J.Denis and T.Dhorne. "Orthogonal tensor decomposition of 3-way tables." In: *In Multiway Data Analysis* (1989), pp. 31–37.

[49] J.Goulart, M.Boizard, R.Boyer, G.Favier, and P.Comon. "Tensor CP decomposition with structured factor matrices: Algorithms and performance". In: *IEEE Journal of Selected Topics in Signal Processing* 10.4 (2015), pp. 757–769.

[50] J.Hamm and D.Lee. "Grassmann Discriminant Analysis: A Unifying View on Subspace-Based Learning". In: *Proceedings of the 25th International Conference on Machine Learning*. ICML '08. Helsinki, Finland: Association for Computing Machinery, 2008, 376–383. ISBN: 9781605582054.

[51] J.Kruskal. "Rank, decomposition, and uniqueness for 3-way and N-way arrays". In: *Multiway data analysis*. 1989, pp. 7–18.

[52] J.Kruskal. "Three-way arrays: rank and uniqueness of trilinear decompositions, with application to arithmetic complexity and statistics". In: *Linear Algebra and its Applications* 18.2 (1977), pp. 95 –138.

[53] J.Levin. "Three-mode factor analysis." In: *Psychological Bulletin* 64.6 (1965), p. 442.

[54] J.Riu and R.Bro. "Jack-knife technique for outlier detection and estimation of standard errors in PARAFAC models". In: *Chemometrics and Intelligent Laboratory Systems* 65.1 (2003), pp. 35–49.

[55] J.Royer, N.Thirion-Moreau, and P.Comon. "Computing the polyadic decomposition of nonnegative third order tensors". In: *Signal Processing* 91.9 (2011), pp. 2159–2171. ISSN: 0165-1684. DOI: https://doi.org/10.1016/j.sigpro.2011.03.006. URL: https://www.sciencedirect.com/science/article/pii/S0165168411000818.

[56] J.Sadecka and J.Tothova. "Fluorescence Spectroscopy and Chemometrics in the Food Classification - A Review". In: *Czech Journal of Food Sciences* 25 (Jan. 2007), pp. 159–173.

[57] J.Wang, Q.Chen, and Y.Chen. "RBF kernel based support vector machine with universal approximation and its application". In: *International symposium on neural networks*. Springer. 2004, pp. 512–517.

[58] J.Zhang, G.Zhu, R. Heath Jr, and K.Huang. "Grassmannian Learning: Embedding Geometry Awareness in Shallow and Deep Learning". In: *CoRR* abs/1808.02229 (2018). arXiv: 1808.02229.

[59] K.Gilman, D.Tarzanagh, and L.Balzano. "Grassmannian optimization for online tensor completion and tracking with the t-svd". In: *IEEE Transactions on Signal Processing* 70 (2022), pp. 2152–2167.

[60] K.Liu, JPCL.Da.Costa, HC.So, L.Huang, and J.Ye. "Detection of number of components in CANDECOMP/PARAFAC models via minimum description length". In: *Digital Signal Processing* 51 (2016), pp. 110–123.

[61] K.Makantasis, A.Doulamis, N.Doulamis, and A.Nikitakis. "Tensor-Based Classification Models for Hyperspectral Data Analysis". In: *IEEE Transactions on Geoscience and Remote Sensing* 56.12 (2018), pp. 6884–6898.

[62] K.Sharma and R.Rameshan. "Image set classification using a distance-based kernel over affine Grassmann manifold". In: *IEEE Transactions on Neural Networks and Learning Systems* 32.3 (2020), pp. 1082–1095.

[63] K.Ye and L.Lim. "Schubert varieties and distances between subspaces of different dimensions". In: *SIAM Journal on Matrix Analysis and Applications* 37.3 (2016), pp. 1176–1197.

[64] L.De Lathauwer and J.Vandewalle. "Dimensionality reduction in higher-order signal processing and rank-(R1, R2,. . . , RN) reduction in multilinear algebra". In: *Linear Algebra and its Applications* 391 (2004), pp. 31–55.

[65] L.De Lathauwer, B.De Moor, and J.Vandewalle. "A multilinear singular value decomposition". In: *SIAM journal on Matrix Analysis and Applications* 21.4 (2000), pp. 1253–1278.

[66] L.Cheng, X.Tong, and Y.Wu. "Distributed Nonnegative Tensor Canonical Polyadic Decomposition With Automatic Rank Determination". In: *2020 IEEE 11th Sensor Array and Multichannel Signal Processing Workshop (SAM)*. IEEE. 2020, pp. 1–5.

[67] L.Cheng, Y.Wu, and H.Poor. "Scaling probabilistic tensor canonical polyadic decomposition to massive data". In: *IEEE Transactions on Signal Processing* 66.21 (2018), pp. 5534–5548.

[68] L.Cheng, Z.Chen, Q.Shi, Y.Wu, and S.Theodoridis. "Towards Flexible Sparsity-Aware Modeling: Automatic Tensor Rank Learning Using the Generalized Hyperbolic Prior". In: *IEEE Transactions on Signal Processing* 70 (2022), pp. 1834–1849.

[69] L.Grasedyck, D.Kressner, and C.Tobler. "A literature survey of low-rank tensor approximation techniques". In: *GAMM-Mitteilungen* 36.1 (2013), pp. 53–78.

[70] L.He, C.Lu, G.Ma, S.Wang, L.Shen, S.Philip, and A.Ragin. "Kernelized support tensor machines". In: *International Conference on Machine Learning*. PMLR. 2017, pp. 1442–1451.

[71] L.He, C.Lu, H.Ding, S.Wang, L.Shen, P.Yu, and A.Ragin. "Multi-way multi-level kernel modeling for neuroimaging classification". In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2017, pp. 356–364.

[72] L.He, X.Kong, P.Yu, X.Yang, A.Ragin, and Z.Hao. "Dusk: A dual structure-preserving kernel for supervised tensor learning with applications to neuroimages". In: *Proceedings of the 2014 SIAM International Conference on Data Mining*. SIAM. 2014, pp. 127–135.

[73] N. Li, S. Kindermann, and C. Navasca. "Some convergence results on the regularized alternating least-squares method for tensor decomposition". In: *Linear Algebra and its Applications* 438.2 (2013), pp. 796–812.

[74] J. Liu, J.Luo, and M. Shah. "Recognizing realistic actions from videos "in the wild"". In: *2009 IEEE Conference on Computer Vision and Pattern Recognition*. 2009, pp. 1996–2003.

[75] L.Tucker. "Implications of factor analysis of three-way matrices for measurement of change". In: *Problems in measuring change* 15.122-137 (1963), p. 3.

[76] L.Tucker. "Some mathematical notes on three-mode factor analysis". In: *Psychometrika* 31.3 (1966), pp. 279–311.

[77] L.Tucker. "The extension of factor analysis to three-dimensional matrices". In: *Contributions to mathematical psychology*. Ed. by H. Gulliksen and N. Frederiksen. New York: Holt, Rinehart and Winston, 1964, pp. 110–127.

[78] L.Wolf and A.Shashua. "Learning over sets using kernel principal angles". In: *Journal of Machine Learning Research* 4.Oct (2003), pp. 913–931.

[79] L.Yuan, C.Li, D.Mandic, J.Cao, and Q.Zhao. "Tensor ring decomposition with rank minimization on latent space: An efficient approach for tensor completion". In: *Proceedings of the AAAI Conference on Artificial Intelligence*. Vol. 33. 01. 2019, pp. 9151–9158.

[80] M.Abadi, P.Barham, J.Chen, Z.Chen, A.Davis, J.Dean, M.Devin, S.Ghemawat, G.Irving, M.Isard, M.Kudlur, J.Levenberg, R.Monga, S.Moore, D.Murray, B.Steiner, P.Tucker, V.Vasudevan, P.Warden, M.Wicke, Y.Yu, and X.Zheng. *TensorFlow: Large-Scale Machine Learning on Heterogeneous Systems*. Software available from tensorflow.org. 2015. URL: `https://www.tensorflow.org/`.

[81] M.Bousse, O.Debals, and L.De Lathauwer. "A tensor-based method for large-scale blind source separation using segmentation". In: *IEEE Transactions on Signal Processing* 65.2 (2016), pp. 346–358.

[82] M.Harandi, C.Sanderson, S.Shirazi, and B.Lovell. "Graph embedding discriminant analysis on Grassmannian manifolds for improved image set matching". In: *CVPR 2011*. IEEE. 2011, pp. 2705–2712.

[83] M.Harandi, M.Salzmann, S.Jayasumana, R.Hartley, and H.Li. "Expanding the family of grassmannian kernels: An embedding perspective". In: *European conference on computer vision*. Springer. 2014, pp. 408–423.

[84] M.Kilmer and C.Martin. "Factorization strategies for third-order tensors". In: *Linear Algebra and its Applications* 435.3 (2011). Special Issue: Dedication to Pete Stewart on the occasion of his 70th birthday, pp. 641–658. ISSN: 0024-3795.

[85] M.Kilmer, K.Braman, H.Karen, H.Ning, and R.Hoover. "Third-Order Tensors as Operators on Matrices: A Theoretical and Computational Framework with Applications in Imaging". In: *SIAM Journal on Matrix Analysis and Applications* 34 (Jan. 2013).

[86] M.Mørup, L.Hansen, C.Herrmann, J.Parnas, and S.Arnfred. "Parallel factor analysis as an exploratory tool for wavelet transformed event-related EEG". In: *NeuroImage* 29.3 (2006), pp. 938–947.

[87] M.Mørup and L.K.Hansen. "Automatic relevance determination for multi-way models". In: *Journal of Chemometrics* 23 (July 2009), pp. 352 –363. DOI: 10.1002/cem.1223.

[88] M.Signoretto. "Kernels and Tensors for Structured Data Modelling". PhD thesis. Katholieke Universiteit Leuven – Faculty of Engineering, Dec. 2011.

[89] M.Signoretto, Q.Tran Dinh, L. De Lathauwer, and J.Suykens. "Learning with tensors: a framework based on convex optimization and spectral regularization". In: *Machine Learning* 94 (2013), pp. 303–351.

[90] M.Signoretto, L.De Lathauwer, and J.Suykens. "A kernel-based framework to tensorial data analysis". In: *Neural networks : the official journal of the International Neural Network Society* 24 8 (2011), pp. 861–74.

[91] M.Sørensen and L.De Lathauwer. "Blind signal separation via tensor decomposition with Vandermonde factor: Canonical polyadic decomposition". In: *IEEE Transactions on Signal Processing* 61.22 (2013), pp. 5507–5519.

[92] M.Vasilescu and D.Terzopoulos. "Multilinear analysis of image ensembles: Tensorfaces". In: *European conference on computer vision*. Springer. 2002, pp. 447–460.

[93] M.Zhou, Y.Liu, Z.Long, L.Chen, and C.Zhu. "Tensor rank learning in CP decomposition via convolutional neural network". In: *Signal Processing: Image Communication* 73 (2019), pp. 12–21.

[94] N.Christianini and J.Shawe-Taylor. "Support vector machines and other kernel-based learning methods". In: *Cambridge UP* (2000).

[95] N.Cristianini, C.Campbell, and C.Burges. "Editorial: Kernel Methods: Current Research and Future Directions". In: *Mach. Learn.* 46.1–3 (Mar. 2002), 5–9. ISSN: 0885-6125.

[96] N.Sidiropoulos, G.Giannakis, and R.Bro. "Blind PARAFAC receivers for DS-CDMA systems". In: *IEEE Transactions on Signal Processing* 48.3 (2000), pp. 810–823.

[97] N.Sidiropoulos, L.De Lathauwer, X.Fu, K.Huang, E.Papalexakis, and C.Faloutsos. "Tensor decomposition for signal processing and machine learning". In: *IEEE Transactions on Signal Processing* 65.13 (2017), pp. 3551–3582.

[98] N.Sidiropoulos and R.Bro. "On the uniqueness of multilinear decomposition ofN-way arrays". eng. In: *Journal of chemometrics* 14.3 (2000), pp. 229–239. ISSN: 0886-9383.

[99] N.Sidiropoulos, R.Bro, and G.B.Giannakis. "Parallel factor analysis in sensor array processing". In: *IEEE Transactions on Signal Processing* 48.8 (2000), pp. 2377–2388.

[100] O.Karmouda, J.Boulanger, and R.Boyer. "Apprentissage supervisé à noyau basé sur la décomposition Canonique Polyadique (CP)". In: *Gretsi 2022*. Nancy, France, Sept. 2022.

[101] O.Karmouda, J.Boulanger, and R.Boyer. "APPRENTISSAGE SUPERVISÉ RAPIDE POUR DES DONNÉES TENSORIELLES". In: *2eme Conférence Internationale Francophone sur la Science des Données*. Marseille, France, June 2021.

[102] O.Karmouda, J.Boulanger, and R.Boyer. "Joint Factors and Rank Estimation for the Canonical Polyadic Decomposition Based on Convex Optimization". In: *IEEE Access* 10 (2022), pp. 82295–82304.

[103] O.Karmouda, J.Boulanger, and R.Boyer. "On the analysis of the Canonical Polyadic Decomposition (CPD)-based tensor learning." In: *2022 56th Asilomar Conference on Signals, Systems, and Computers*. IEEE. 2022.

[104] O.Karmouda, J.Boulanger, and R.Boyer. "Speeding Up of Kernel-Based Learning for High-Order Tensors". In: *ICASSP 2021 - 2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. 2021, pp. 2905–2909.

[105] O.Karmouda, R.Boyer, and J.Boulanger. "High-Dimensional Data Learning Based on Tensorial-Singular Space of Tensor Train Cores". In: *30th European Signal Processing Conference (EUSIPCO)*. 2022.

[106] P.Absil, R.Mahony, and R.Sepulchre. "Riemannian geometry of Grassmann manifolds with a view on algorithmic computation". In: *Acta Applicandae Mathematica* 80.2 (2004), pp. 199–220.

[107] E. Papalexakis and C.Faloutsos. "Fast efficient and scalable Core Consistency Diagnostic for the parafac decomposition for big sparse tensors". In: *2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. 2015, pp. 5441–5445.

[108] E. Papalexakis, C.Faloutsos, and N.Sidiropoulos. "Tensors for Data Mining and Data Fusion: Models, Applications, and Scalable Algorithms". In: *ACM Trans. Intell. Syst. Technol.* 8.2 (Oct. 2016).

[109] P.Comon. "Tensor decompositions". In: *Mathematics in signal processing V* (2002), pp. 1–24.

[110] P.Comon. "Tensors: a brief introduction". In: *IEEE Signal Processing Magazine* 31.3 (2014), pp. 44–53.

[111] P.Comon, X.Luciani, and A.De Almeida. "Tensor decompositions, alternating least squares and other tales". In: *Journal of Chemometrics: A Journal of the Chemometrics Society* 23.7-8 (2009), pp. 393–405.

[112] P.Kroonenberg and J.De Leeuw. "Principal component analysis of three-mode data by means of alternating least squares algorithms". In: *Psychometrika* 45.1 (1980), pp. 69–97.

[113] Q.Shi, H.Lu, and Y. Cheung. "Tensor Rank Estimation and Completion via CP-Based Nuclear Norm". In: *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management*. CIKM '17. Singapore, Singapore: Association for Computing Machinery, 2017, 949–958.

[114] Q.Shi, H.Lu, and Y.Cheung. "Tensor rank estimation and completion via CP-based nuclear norm". In: *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management*. 2017, pp. 949–958.

[115] Q.Zhao, G.Zhou, T.Adalı, L.Zhang, and A.Cichocki. "Kernel-based tensor partial least squares for reconstruction of limb movements". In: *2013 IEEE International Conference on Acoustics, Speech and Signal Processing*. IEEE. 2013, pp. 3577–3581.

[116] R.Bhatia. *Positive Definite Matrices*. Princeton University Press, 2009.

[117] R.Bro. "Exploratory study of sugar production using fluorescence spectroscopy and multi-way analysis". In: *Chemometrics and Intelligent Laboratory Systems* 46.2 (1999), pp. 133–147.

[118] R.Bro. "PARAFAC. Tutorial and applications". In: *Chemometrics and intelligent laboratory systems* 38.2 (1997), pp. 149–171.

[119] R.Harshman. "Foundations of the PARAFAC procedure: Models and conditions for an "explanatory" multi-modal factor analysis". In: *UCLA Working Papers in Phonetics* 16 (1970).

[120] R.Harshman. "Foundations of the PARAFAC procedure: Models and conditions for an "explanatory" multi-modal factor analysis". In: *UCLA Working Papers in Phonetics* 16 (1970), pp. 1–84.

[121] R.Orús. "A practical introduction to tensor networks: Matrix product states and projected entangled pair states". In: *Annals of Physics* 349 (2014), pp. 117 –158.

[122] R.Rasmus and H.Kiers. "A new efficient method for determining the number of components in PARAFAC models". In: *Journal of Chemometrics* 17.5 (May 2003), pp. 274–286.

[123] R.Tibshirani. "Regression Shrinkage and Selection via the Lasso". In: *Journal of the Royal Statistical Society. Series B (Methodological)* 58.1 (1996), pp. 267–288.

[124] R.Vemulapalli, J.Pillai, and R.Chellappa. "Kernel learning for extrinsic classification of manifold features". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2013, pp. 1782–1789.

[125] S.Boyd and L.Vandenberghe. *Convex optimization*. Cambridge university press, 2004.

[126] V.De Silva and L.Lim. "Tensor rank and the ill-posedness of the best low-rank approximation problem". In: *SIAM Journal on Matrix Analysis and Applications* 30.3 (2008), pp. 1084–1127.

[127] S.Jayasumana, R.Hartley, M.Salzmann, H.Li, and M.Harandi. "Kernel methods on Riemannian manifolds with Gaussian RBF kernels". In: *IEEE transactions on pattern analysis and machine intelligence* 37.12 (2015), pp. 2464–2477.

[128] S.Miron, Y.Zniyed, R.Boyer, A. De Almeida, G.Favier, D.Brie, and P.Comon. "Tensor methods for multisensor signal processing". In: *IET signal processing* 14.10 (2020), pp. 693–709.

[129] S.Theodoridis and K.Koutroumbas. *Pattern recognition*. Elsevier, 2006.

[130] S.Tong and E.Chang. "Support vector machine active learning for image retrieval". In: *Proceedings of the ninth ACM international conference on Multimedia*. 2001, pp. 107–118.

[131] T.Giorgio. *Practical and Computational Aspects in Chemometric Data Analysis: Ph. D. Dissertation*. Department of Food Science, Royal Veterinary and Agricultural University, 2006.

[132] T.Gärtner, J.Lloyd, and P.Flach. "Kernels and Distances for Structured Data". In: *Machine Learning*. 2004.

[133] T.Hastie, R.Tibshirani, and J.Friedman. *The elements of statistical learning: data mining, inference, and prediction*. Vol. 2. Springer, 2009.

[134] T.Huang, V.Kecman, and I.Kopriva. *Kernel based algorithms for mining huge data sets*. Vol. 1. Springer, 2006.

[135] T.Kolda and B.Bader. "Tensor Decompositions and Applications". In: *SIAM Review* 51.3 (2009), pp. 455–500.

[136] T.Papastergiou, E.Zacharaki, and V.Megalooikonomou. "Tensor Decomposition for Multiple-Instance Classification of High-Order Medical Data". In: *Complexity* 2018 (Dec. 2018), pp. 1–13.

[137] P. Turaga, A. Veeraraghavan, A. Srivastava, and R. Chellappa. "Statistical Computations on Grassmann and Stiefel Manifolds for Image and Video-Based Recognition". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 33.11 (2011), pp. 2273–2286.

[138] W.Guo, I.Kotsia, and I.Patras. "Tensor Learning for Regression". In: *IEEE Transactions on Image Processing* (2012).

[139] W.Kim and M.Crawford. "A novel adaptive classification method for hyperspectral data using manifold regularization kernel machines". In: *2009 First Workshop on Hyperspectral Image and Signal Processing: Evolution in Remote Sensing*. IEEE. 2009, pp. 1–4.

[140] W.Krijnen, T. Dijkstra, and A.Stegeman. "On the non-existence of optimal solutions and the occurrence of "degeneracy" in the Candecomp/Parafac model". In: *Psychometrika* 73.3 (2008), pp. 431–439.

[141] X.Fu, N.Vervliet, L.De Lathauwer, K.Huang, and N.Gillis. "Computing large-scale matrix and tensor decomposition with structured factors: A unified nonconvex optimization perspective". In: *IEEE Signal Processing Magazine* 37.5 (2020), pp. 78–94.

[142] X.Gong, Q.Lin, F.Cong, and L.De Lathauwer. "Double coupled canonical polyadic decomposition for joint blind source separation". In: *IEEE Transactions on Signal Processing* 66.13 (2018), pp. 3475–3490.

[143] X.Guo, S.Miron, D.Brie, and A.Stegeman. "Uni-mode and partial uniqueness conditions for CANDECOMP/PARAFAC of three-way arrays with linearly dependent loadings". In: *SIAM Journal on Matrix Analysis and Applications* 33.1 (2012), pp. 111–129.

[144] X.Guo, X.Huang, L.Zhang, L.Zhang, A.Plaza, and J.Benediktsson. "Support tensor machines for classification of hyperspectral remote sensing imagery". In: *IEEE Transactions on Geoscience and Remote Sensing* 54.6 (2016), pp. 3248–3264.

[145] X.Han, L.Albera, A.Kachenoura, L.Senhadji, and H.Shu. "Low rank canonical polyadic decomposition of tensors based on group sparsity". In: *2017 25th European Signal Processing Conference (EUSIPCO)*. 2017, pp. 668–672. DOI: 10.23919/EUSIPCO.2017.8081291.

[146] X.Kong and P.Yu. "Brain network analysis: a data mining perspective". In: *ACM SIGKDD Explorations Newsletter* 15.2 (2014), pp. 30–38.

[147] Y.Chikuse. *Statistics on special manifolds*. Vol. 174. Springer Science & Business Media, 2003.

[148] Y.Ji, Q.Wang, X.Li, and J.Liu. "A Survey on Tensor Techniques and Applications in Machine Learning". In: *IEEE Access* 7 (2019), pp. 162950–162990.

[149] Y.Wang, H.Tung, A.Smola, and A.Anandkumar. "Fast and guaranteed tensor decomposition via sketching". In: *Appeared in Proceedings of Advances in Neural Information Processing Systems (NIPS)* (2015).

[150] Y.Zniyed, O.Karmouda, R.Boyer, J.Boulanger, A.De Almeida, and G.Favier. "Chapter 14 - Structured tensor train decomposition for speeding up kernel-based learning". In: *Tensors for Data Processing*. Ed. by Y.Liu. Academic Press, 2022, pp. 537–563. ISBN: 978-0-12-824447-0.

[151] Y.Zniyed, R.Boyer, A. De Almeida, and G.Favier. "High-order tensor estimation via trains of coupled third-order CP and Tucker decompositions". In: *Linear Algebra and its Applications* 588 (2020), pp. 304 –337. ISSN: 0024-3795.

[152] Y.Zniyed, R.Boyer, A.De Almeida, and G.Favier. "A TT-based hierarchical framework for decomposing high-order tensors". In: *SIAM Journal on Scientific Computing* 42.2 (2020), A822–A848.

[153] Z.Hao, L.He, B.Chen, and X.Yang. "A linear support higher-order tensor machine for classification". In: *IEEE Transactions on Image Processing* 22.7 (2013), pp. 2911–2920.

[154] Q. Zhao, G. Zhou, T. Adali, L. Zhang, and A. Cichocki. "Kernelization of Tensor-Based Models for Multiway Data Analysis: Processing of Multi-dimensional Structured Data". In: *IEEE Signal Processing Magazine* 30.4 (2013), pp. 137–148.

[155] Z.Huang, R. Wang, S.Shan, and X.Chen. "Projection Metric Learning on Grassmann Manifold with Application to Video based Face Recognition". In: *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2015, pp. 140–149.

[156] Y. Zniyed, S. Miron, R. Boyer, and D. Brie. "Uniqueness of Tensor Train Decomposition with Linear Dependencies". In: *2019 IEEE 8th International Workshop on Computational Advances in Multi-Sensor Adaptive Processing (CAMSAP)*. 2019, pp. 460–464.