université PARIS-SACLAY

New perspectives and tools for Tensor Principal Component Analysis and beyond

Nouveaux outils et perspectives pour l'Analyse en Composantes Principales tensorielle et au-delà

Thèse de doctorat de l'université Paris-Saclay

École doctorale n° 564, physique en Île-de-France (PIF)
Spécialité de doctorat : Physique
Graduate School : Physique. Référent : Faculté des sciences d'Orsay

Thèse préparée dans l'unité de recherche **IJCLab (Université Paris-Saclay, CNRS)**, sous la direction de **Vincent RIVASSEAU**, Professeur, et le co-encadrement de **Mohamed TAMAAZOUSTI**, Ingénieur-chercheur

Thèse soutenue à Paris-Saclay, 05 Décembre 2022, par

Mohamed OUERFELLI

Composition du Jury

Membres du jury avec voix délibérative

Alice GUIONNET Professeure, ENS Lyon	Présidente
Gérard BEN AROUS Professeur, Université de New York	Rapporteur & Examinateur
Razvan GURAU Professeur, Université Heidelberg	Rapporteur & Examinateur
Mustapha LEBBAH Professeur, Université Paris-Saclay	Examinateur
Valentina ROS Chercheuse, Université Paris-Saclay	Examinatrice

THESE DE DOCTORAT

NNT : 2022UPASP138

université PARIS-SACLAY | ÉCOLE DOCTORALE Physique en Île-de-France (PIF)

Titre: Nouveaux outils et perspectives pour l'Analyse en Composantes Principales tensorielle et au-delà

Mots clés: ACP tensorielle, Décomposition de tenseur, Ecart théorique algorithmique, Paysages non convexes à hautes dimensions, Tenseurs aléatoires

Résumé: Cette thèse a pour objet l'analyse en composantes principales (APC) tensorielle. L'introduction constate l'intérêt grandissant pour les outils tensoriels dans le domaine de l'Intelligence Artificielle (IA). Trouver de nouvelles méthodes et algorithmes en IA qui soient moins opaques et qui nécessitent moins de données que l'apprentissage profond est crucial pour faciliter l'adoption de l'IA dans de nouveaux domaines d'application. Le deuxième chapitre souligne l'importance plus générale de l'étude de l'APC tensorielle en mettant en lumière sa position clé à l'intersection de sujets de recherche très actifs de trois disciplines différentes, les mathématique appliquées, la physique des systèmes désordonnés, et l'informatique théorique.

La contribution de cette thèse se divise en deux parties. La première consiste en un nouveau cadre théorique adapté aux tenseurs, inspiré par des travaux de recherche récents en physique des hautes énergies. Ce cadre permet d'améliorer les résultats sur des données synthétiques ainsi que dans certaines applications concrètes. Il permet aussi de donner des nouvelles garanties théoriques pour des situations plus générales, comme un tenseur aux dimensions non égales. La deuxième contribution introduit une nouvelle méthode basée sur une approche plus empirique. Elle fournit de nouvelles intuitions qui nous poussent à questionner certaines conjectures telles que celles qui portent sur le comportement et la performance de la méthode de la puissance itérée tensorielle. Ces résultats pourraient apporter de nouveaux éléments essentiels à l'étude de l'écart théorique-algorithmique et le comportement de la descente de gradient dans l'apprentissage profond.

Title: New perspectives and tools for Tensor Principal Component Analysis and beyond

Keywords: Tensor PCA, Tensor decomposition, Statistical-computational gap, High dimensional non convex landscapes, random tensors

Abstract: This thesis deals with tensorial principal component analysis (PCA). The introduction notes the growing interest in tensorial tools in the field of Artificial Intelligence (AI). Finding new AI methods and algorithms that are less opaque and that require less data than deep learning is crucial to facilitate the adoption of AI in new application domains. The second chapter emphasizes the more general importance of the study of tensorial PCA by highlighting its key position at the intersection of very active research subjects from three different disciplines, applied mathematics, the physics of disordered systems, and theoretical computer science.

The contribution of this thesis is divided into two parts. The first consists of a new theoretical framework adapted to tensors, inspired by recent research work in high energy physics. This framework makes it possible to improve the results on synthetic data as well as in certain concrete applications. It also makes it possible to give new theoretical guarantees for more general situations, such as a tensor with unequal dimensions. The second contribution introduces a new method based on a more empirical approach. It provides new insights that lead us to question certain conjectures such as those related to the behavior and performance of the tensor power iteration method. These results could bring new essential elements to the study of the statistical-computational gap and the behavior of gradient descent in deep learning.

Contents

List of Figures

List of Tables

Acknowledgements:

First and foremost I am extremely grateful to my supervisors, Prof. Vincent Rivasseau and Dr. Mohamed Tamaazousti for their invaluable guidance and constant support during these three years. They have always been incredibly available and generous with their time, advice and constructive feedback. Vincent has been an exceptional adviser despite the various difficulties encountered during my PhD studies. He showed me the importance of open-mindedness and curiosity in science by continuously searching and confronting new challenges and novel domains with original approaches and great enthusiasm. Mohamed has been an exceptional mentor, instructing me the importance of having a well thought out strategy and a pedagogical scientific communication in research. He introduced me to research for concrete applications and inspired me by his openness and continuous drive and passion for learning new tools and acquiring knowledge in diverse new scientific fields. I feel very fortunate to be able to continue collaborating with them.

I would also like to express my sincere thanks for the two referees of my thesis: Prof. Gérard Ben Arous and Prof. Razvan Gurau, for the time spent reading my manuscript and their very enriching comments. I would also like to thank Prof. Alice Guionnet, Prof. Mustapha Lebbah and Dr. Valentina Ros to have accepted to be members of my PhD defense committee and for their interesting questions and feedback.

I want to thank my PhD colleagues in CEA who accompanied me during this process, Adrian and Jade. I am also very grateful to my colleagues Vincent, Dine, Vasily, Riccardo, Zakarya, Parham and all the others in LVML.

I would like to extend my sincere thanks to all the members of the Tensor Club that I had a great enjoyment to meet in Bordeaux and Heidelberg, as well as along the seminars of Tensor Club. These meetings and discussions were a very agreeable, fruitful and excellent research experience. Thanks should also go to Virginie who was of a great help for the various and complicated administrative tasks during my PhD.

I want to thank my professors at ENS Cachan, Sorbonnes University and ENS Ulm as well as my supervisors in Aveiro, Lund and Paris for providing me the education and the research experience that were extremely useful for my work.

I am also grateful to my friends and relatives Mikael, François, Melanie, Fahmi, Hanen, Oussema, Wiem, Ali, Anass, Larbi, Tasnime, Ayoub, Aymen, Marwen and many others who encouraged me and made the past 3 years very enjoyable.

Finally, I would like to thank my family for their backing and encouragement. Special thanks goes out to my young niece Chems and to my parents: Abdallah and Noura for their inestimable support, assistance and guidance during all my studies.

Notations

Tr$(.)$ Trace operator

$\mathbb{E}$ Expectation operator

Var$(.)$ Variance operator

Covar$(.\ ,.)$ Covariance operator

$.^\top$ Transpose operator

$\|.\|$ Norm operator

$\delta_{i,j}$ Kronecker symbol

$\otimes$ Outer product

$\boldsymbol{T}$,$\boldsymbol{M}$,$\boldsymbol{v}$ Tensors, matrices and vectors

$\mathcal{T}_{ijk}, M_{ij}, v_i$ Components of tensors, matrices and vectors

$[p]$ The set $\{1, .., p\}$

A real $p-$th order tensor $\mathbf{T} \in \bigotimes_{i=1}^{p} \mathbb{R}^{n_i}$ is a member of the tensor product of Euclidean spaces $\mathbb{R}^{n_i}, i \in [p]$. It is symmetric if $\mathcal{T}_{i_1 \dots i_k} = \mathcal{T}_{\tau(i_1) \dots \tau(i_k)}\ \forall \tau \in \mathcal{S}_k$ where $\mathcal{S}_k$ is the symmetric group of degree k.

For a vector $\mathbf{v} \in \mathbb{R}^n$, $\boldsymbol{v}^{\otimes p} \equiv \boldsymbol{v} \otimes \mathbf{v} \otimes \cdots \otimes \boldsymbol{v} \in \bigotimes^{p} \mathbb{R}^n$ denotes its p-th tensor power.

$\mathbf{T}(\boldsymbol{v}, \boldsymbol{w}, \boldsymbol{z}) \equiv \langle \mathbf{T}, \boldsymbol{v} \otimes \boldsymbol{w} \otimes \boldsymbol{z} \rangle \equiv \sum_{ijk} \mathcal{T}_{ijk} v_i w_j z_k$ is the euclidean product between the tensors $\mathbf{T}$ and $\boldsymbol{v} \otimes \boldsymbol{w} \otimes \boldsymbol{z}$.

$\mathbf{T}(:, \boldsymbol{w}, \boldsymbol{z})$ is the vector whose i-th entry is $\sum_{jk} \mathcal{T}_{ijk} w_j z_k$ and $\mathbf{T}(:, :, \boldsymbol{z})$ is the matrix whose i, jth element is $\sum_k \mathcal{T}_{ijk} z_k$.

Chapter 1

Introduction

1.1 Artificial Intelligence

Artificial Intelligence (AI) is humanity's attempt to automate/delegate human cognitive abilities through technology. This simulation of intelligence can relate to reasoning and decision (critical reasoning) [Pom97], understanding of natural language [YHPC18], visual perception (interpretation of and deriving meaningful information from images and scenes) [FJY+19], auditory data (understanding of spoken language) [GME11] as well as various other sensors [ŠBP21]. The concept of Artificial intelligence long precedes the arrival of computing. The idea that a created object is capable of producing or obtaining a consciousness similar to that of man is present in Greek mythology, ancient Egypt, Jewish folklore (the Golem) and many other ancient cultures.

The earliest substantial work in the field of artificial intelligence was done in the mid-20th century by the British logician and computer pioneer Alan Turing. His seminal paper "Computing Machinery and Intelligence" [Tur09] crystallizes ideas about the possibility of programming an electronic computer to behave intelligently and how to test its intelligence.

Modern AI is a rich field encompassing diverse major branches such as robotics or expert systems. One important subarea of AI is Machine learning (ML) [Mah20]. It allows machines to learn from data without being programmed explicitly so. ML and data-driven statistical techniques gained momentum in recent years due to an incredible increase in

the number and complexity of data available [BN06, GBC16]. Machine-learning systems are nowadays used for object recognition in images, automatic speech recognition, recommender systems that suggests relevant news items or products to users, and selecting relevant results of search.

There are three main machine learning paradigms: unsupervised, supervised and reinforcement learning [AAJM+20]. Unsupervised learning is a type of machine learning algorithm used to draw inferences from datasets consisting of input data without labelled responses. The most common unsupervised learning method is cluster analysis, which is used for exploratory data analysis to find hidden patterns or grouping in data. Other unsupervised approaches include Association Rules (mainly Apriori algorithm) and Dimensionality reduction (Principal component analysis, Singular Value decomposition, Autoencoder). Supervised learning involves making a prediction based on a set of pre-specified input and output variables. There are a number of statistical tools used for supervised learning. Some examples include traditional statistical prediction methods like regression models (e.g. regression splines, projection pursuit regression, penalized regression) that involve fitting a model to data, evaluating the fit and estimating parameters that are later used in a predictive equation. Other tools include tree-based methods (e.g. classification and regression trees and random forests), which successively partition a data set based on the relationships between predictor variables and a target (outcome) variable. Other examples include neural networks, discriminant functions and linear classifiers, support vector classifiers and machines. Finally, the Reinforcement Learning model learns from the mistakes and the feedback provided on those mistakes.

Artificial intelligence (AI) and machine learning (ML) have demonstrated their potential to revolutionize industries,public services, and society, achieving or even surpassing human levels of performance in terms of accuracy for a range of problems, such as image and speech recognition [MKS+15] and language translation [YHPC18].

1.2 Future challenges for AI

Building on its tremendous potentiality, AI is rapidly gaining influence in people's daily lives and in professional fields like healthcare, education, scientific research, communications, transportation, security, and art. However, at the same time that AI systems are starting to be deployed widely into the economy, multiple issues associated with AI are becoming magnified.

A major problem frequently referred to is the interpretability of the methods. This issue heightened with the diffusion of ML-based technologies in safety-critical domains such as healthcare, finance, law, defense and governance which require accountability for decisions and for how data is used in making decisions. Indeed, such fields require trust of users in a decision which is achieved by having a method that is easily interpretable, relatable to the user, connects the decision with contextual information, known laws and prior experiences and reflects the thinking mechanism of the user in reaching a decision.

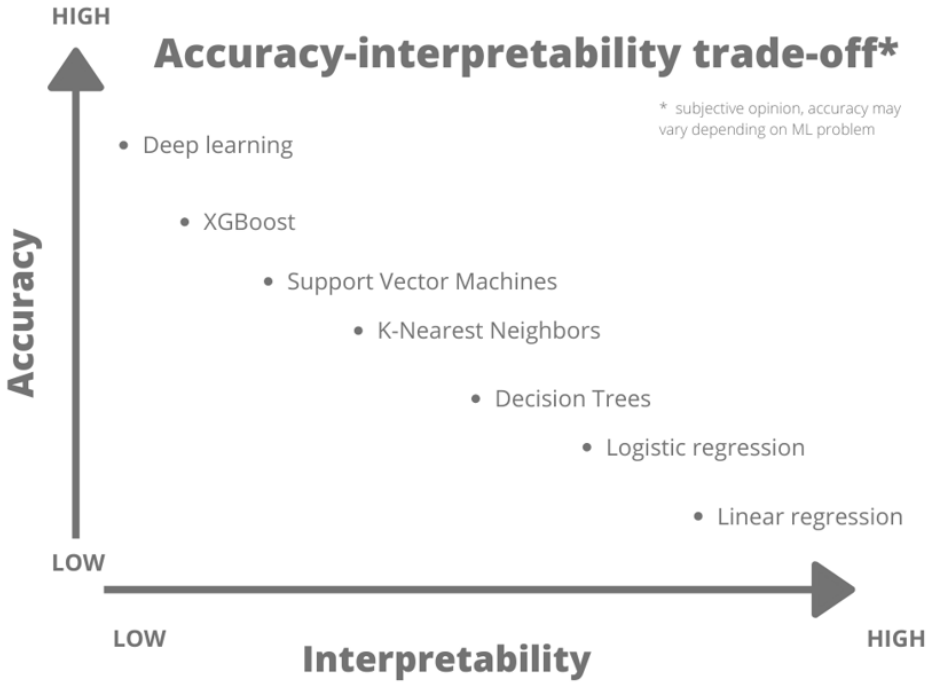


Figure 1.1: Subjective and rough classification of ML methods depending on their Accuracy Interpretability trade-off

Similar concerns are of ethical nature and range from the possibility of criminality, fraud and identity theft to harassment and discriminatory decisions or spreading of disinformation.

Moreover, several of the most-used methods suffer from the scarcity of data in some applications as well as the discrepancy between the training test and the real life data in others. Indeed, in practice, ML models are applied to data in real-world settings that rarely match the training distribution. The reliability of a model depends upon its ability to accommodate underrepresented or imbalanced data material and make relevant decisions in a broad array of scenarios. This requirement is fairly linked to interpretability as understanding how a model works allows to get some intuitions that may be helpful for this purpose.

On the other hand, there have been growing work focusing on building light ML models such as small neural networks (for Internet-of-things (IoT) devices, real-time training, etc.) that require less computing power and are more practically interesting and useful and generally more interpretable.

Limits of current methods

Applying ML techniques without careful consideration of their assumptions and limitations may lead to a dilapidation of valuable resources and incorrect scientific inferences. There are various existent and frequently used ML methods, they all come with their advantages and drawbacks.

Classical statistical methods are widely utilized in most of the day-to-day applications with simple requirements of time and resources. They are well understood by the scientific community. They are thus considered to be transparent and practical tools. However, their performance are less satisfactory when complex data with high number of variables are investigated.

Non-traditional machine learning approaches help in overcoming some of these classical models limitations but bring new drawbacks. Among others, the most important one is their poor interpretability. For example, in a deep neural network, one may determine mathematically which nodes of the neural network were activated, but we lack understanding of how neurons behaved collectively to arrive at the final output. This characteristic gives neural network the property of being a "black box". Figure 1.1 gives a rough and subjective classification of methods based on their accuracy-interpretability trade-off.

These new highly sophisticated methods also commonly necessitate a large amount of data that is densely and uniformly distributed. Indeed, although deep learning models have made exciting progress in vision, language, and other fields [GBC16], the strong performance of such models is generally heavily dependent on having test data drawn from the same distribution as their training set.

It is frequent that these large and complex models, after deployment, fail to achieve the reported high accuracies, lead to unfair decisions, and sometimes provide shocking predictions contradicting the most basic principles of common sense.

1.2.1 Research directions

To address these issues, various research has been done to improve unbiased and impartial decision-making, enhance the generalization ability of models to broader data domains and develop explanations for ML models. These objectives are heavily dependent on each other and the interpretability is a fundamental aspect that improves the two others.

Recent achievements in machine learning have been performed in applications that did not require high interpretability such as online advertisement and research results. Thus, most methods are not very keen in interpretability constraints, as their objectives did not require it. This may lead to the incorrect assumption that the most accurate methods should inherently uninterpretable and complicated.

When researchers gain a deeper understanding on the models they build, it allows them to produce AI systems that are better able to serve the humans who rely upon them, as more interpretable models often become more accurate.

There is two main lines of research pursued to improve interpretability:

1. Developing and improving approaches that are inherently explainable, also known as white-box models, such as decision trees and linear regression models.
2. Providing post-hoc explanations for already trained, so-called ‘black-box’, models.

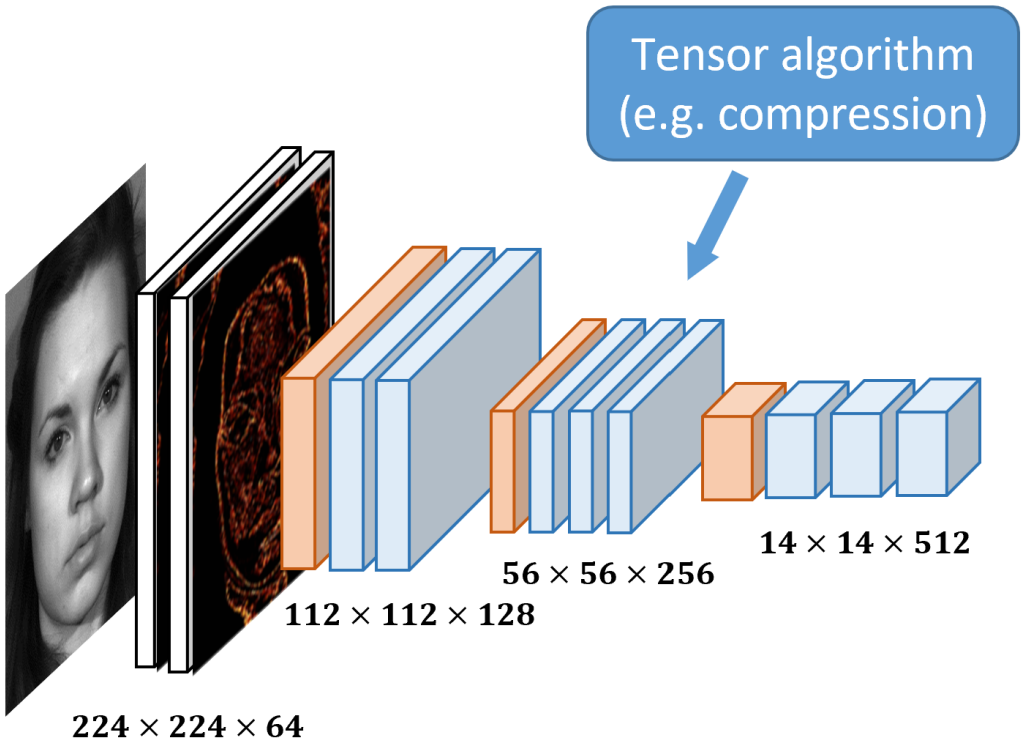


Figure 1.2: Tensors in neural networks.

A common approach for these two lines of research is based on developing new tensorial tools.

1.2.2 Tensorial approach

Tensors are a type of data structure that generalizes vectors and matrices to dimensions superior or equal to three. They became ubiquitous in modern machine learning given their abilities to retain and capture multidimensional structure that is essential for multiple applications.

Deep neural networks typically map between higher-order tensors through architectures such as convolutional layers, as illustrated in Figure 1.2. In fact, the ability of deep convolutional neural networks to preserve the local structure of the input is considered to be a property that is crucial for the great performances achieved [KPAP16].

1.3 Tensorial tools from theoretical physics perspective

This present work was initiated as part of a collaboration between the IJCLab laboratory (formerly LPT) in Orsay, through my thesis director Vincent Rivasseau and the LVML vision laboratory at the CEA.

Within CEA LIST, the Vision Laboratory for Modeling and Localization is in charge of

a research activity initiated in the 90s on the themes of computer vision and machine learning for applications dealing with localization in the environment, precise location of objects, 3D reconstruction, augmented reality, diminished reality, realistic rendering. Its mission consists in transferring these technologies to the industrial world and is carried out upstream by developing state-of-the-art analysis methods in collaboration with the academic world and downstream by adapting these technologies to the application contexts of our industrial partners, from proof of concept to pre-industrial prototypes.

The identity of IJCLab is centered on the field of "the physics of two infinities" and their applications. The scientific activities of IJCLab are structured in 7 scientific poles: Astroparticles, Astrophysics and Cosmology; Physics of Accelerators; High Energy Physics; Nuclear physics; Theoretical Physics; Energy and environment; Health. The mathematical physics group studies on the one hand algebraic and geometric methods in various fields ranging from non-commutative geometry to quantum field theory, on the other hand classical analysis and functional analysis in quantum mechanics and field theory.

Random Tensor Theory (RTT) provides a set of combinatorial tools dedicated to the study of trace invariant graphs [Gur17]. Trace invariants of a tensor $\mathbf{T} \in \bigotimes_{i=1}^{k} \mathbb{R}^{n_i}$ are tensor networks scalars that are invariant under the following $O(n_1) \times \cdots \times O(n_k)$ transformations:

$$\mathbf{T}_{i_1 \dots i_k} \longrightarrow \mathbf{T}'_{i_1 \dots i_k} = \sum_{j_1 \cdots j_k} O^{(1)}_{i_1 j_1} \dots O^{(k)}_{i_k j_k} \mathbf{T}_{j_1 \dots j_k}$$

RTT allows to obtain important probabilistic results on trace invariants by using simple enumerative combinatorics. In particular, it gives a simple way to compute the moments (expected value, variance, etc.) of the distribution of these scalars for random tensors. In the following, it should be understood from the context that

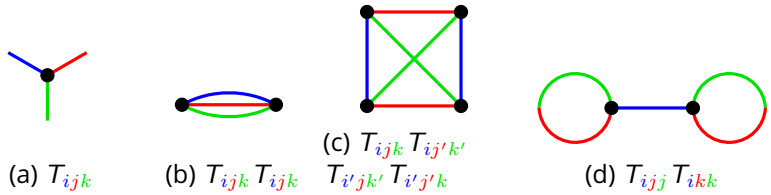
An important concept in problems involving matrices is the spectral theory. It refers to the study of eigenvalues and eigenvectors of a matrix and it is of fundamental importance in numerous areas. Equivalently, the traces of the n first matrix powers

$$\mathrm{Tr}\Big(\boldsymbol{A}\boldsymbol{A}^\top\Big), \mathrm{Tr}\Big((\boldsymbol{A}\boldsymbol{A}^\top)^2\Big), \dots, \mathrm{Tr}\Big((\boldsymbol{A}\boldsymbol{A}^\top)^n\Big)$$

contain the same information as the eigenvalues (in absolute value) since each set can be inferred from the other through some basic algebraic operations.

In the tensor case, the concept of eigenvalue and eigenvector is ill-defined and not practical giving that the number of eigenvalues is exponential with the dimension n [Qi05, CS13] and computing them is very complicated. In contrast, we have a very convenient generalization of the traces of the power matrices for the tensors that we call trace invariants.

Let's give a formal definition of trace invariants. Let $\mathbf{T}$ be a tensor whose entries are $T_{i_1,\dots,i_k}$. Let's define a contraction of a pair of indices as setting them equal to each other and summing over them, as in calculating the trace of a matrix ($\boldsymbol{A}_{ij} \to \sum_{i=1}^{n} \boldsymbol{A}_{ii}$). The trace invariants of the tensor $\mathbf{T}$ correspond to the different ways to contract pairs of indices in a product of an even number of copies of $\mathbf{T}$. The degree of the trace invariants



(a) T_{ijk} (b) $T_{ijk}\,T_{ijk}$ (c) $T_{ijk}\,T_{ij'k'}\,T_{i'jk'}\,T_{i'j'k}$ (d) $T_{ijj}\,T_{ikk}$

Figure 1.3: Example of graphs and their associated invariants.

consists in the number of copies of $\mathbf{T}$ contracted. For example, $\sum_{i_1,i_2,i_3} T_{i_1i_2i_3}T_{i_1i_2i_3}$ and $\sum_{i_1,i_2,i_3} T_{i_1i_2i_2}T_{i_1i_3i_3}$ are trace invariants of degree 2.

A trace invariant of degree d of a tensor $\mathbf{T}$ of order k admits a practical graphical representation as an edge colored graph $\mathcal{G}$ obtained by following two steps: we first draw d vertices representing the d different copies of $\mathbf{T}$. The indices of each copy is represented by k half-edges with a different color in $\{1,\dots,k\}$ for each index position as shown in Figure 1.3a. Then, when two different indices are contracted in the tensor invariant, we connect their corresponding half-edges in $\mathcal{G}$. Reciprocally, to obtain the tensor invariant associated to a graph $\mathcal{G}$ with d vertices, we take d copies of $\mathbf{T}$ (one for each vertex), we associate a color for each index position $\{1,\dots,k\}$, and we contract the indices of the d copies of $\mathbf{T}$ following the coloring of the edges connecting the vertices. We denote this invariant $I_{\mathcal{G}}(\mathbf{T})$. Three important examples of trace invariants worth mentioning are: the melon diagram (Figure 1.3b), the tetrahedral (1.3c) and the tadpole (1.3d). [ABGD20] provides a thorough study about the number of trace invariants for a given degree d.

1.4 Tensor PCA model and its motivations.

Tensor PCA was introduced in the pioneer work of [RM14] and consists in recovering a signal spike $\boldsymbol{v}_0^{\otimes k}$ that has been corrupted by a noise tensor $\mathbf{Z}$: $\mathbf{T} = \mathbf{Z} + \beta \boldsymbol{v}_0^{\otimes k}$ where $\boldsymbol{v}_0$ is a unitary vector and β the Signal-to-Noise Ratio (SNR). The motivation for Tensor PCA is three-fold:

1) Tensor PCA could be considered as a simple case of Tensor decomposition. However, it has a different motivation which is the theoretical study of the computational limitations in the very low SNR regime while the Tensor decomposition literature mainly address practical applications, often in a large SNR. Yet, algorithms developed for Tensor PCA could be generalised to address Tensor decomposition as in [WA16].

2) In addition to that, Tensor PCA is also often used as a prototypical inference problem for the theoretical study of the computational hardness of optimization in high-dimensional non-convex landscapes, in particular using the well spread gradient descent algorithm and its variants ([BAGJ+20, MKUZ19, MBC+19, MBC+20]). Indeed, these algorithms are used with great empirical success in many ML areas such as Deep Learning, but unfortunately they are generally devoid of theoretical guarantees. Understanding the dynamics of gradient descent methods in specific landscapes such as Tensor PCA could bring new insights.

3) One of the main characteristic of Tensor PCA is its conjectured statistical algorithmic gap: while information theory shows that it is theoretically possible to recover the signal for $\beta \sim O(1)$, all existent algorithms have been shown or conjectured to have an algorithmic threshold for $k \geq 3$ of at least $\beta \sim O(n^{(k-2)/4})$. Thus Tensor PCA is considered as an interesting study case of such a gap that appears in various other problems (see references in [BAGJ+20] and [LZ20]).

Simple and symmetrized power iteration The simple power iteration consists in performing the following operation $\boldsymbol{v} \leftarrow \frac{\mathbf{T}(:,\boldsymbol{v},\boldsymbol{v})}{\|\mathbf{T}(:,\boldsymbol{v},\boldsymbol{v})\|}$ Given a non-symmetrical tensor, we define the symmetrized power iteration as:

$$\boldsymbol{v} \leftarrow \frac{\mathbf{T}(:,\boldsymbol{v},\boldsymbol{v}) + \mathbf{T}(\boldsymbol{v},:,\boldsymbol{v}) + \mathbf{T}(\boldsymbol{v},\boldsymbol{v},:)}{\|\mathbf{T}(:,\boldsymbol{v},\boldsymbol{v}) + \mathbf{T}(\boldsymbol{v},:,\boldsymbol{v}) + \mathbf{T}(\boldsymbol{v},\boldsymbol{v},:)\|} \tag{1.1}$$

Performing a symmetrized power iteration on a tensor $\mathbf{T}$ amounts to perform a simple power iteration on the symmetrized tensor $\mathbf{T}_{\text{sym}} \equiv \sum_{\sigma \in \mathcal{S}_k} \mathbf{T}_{\sigma(i_1)\sigma(i_2)\ldots\sigma(i_k)}$. Unless specified otherwise, we restrict ourselves to a symmetrical tensor and $k = 3$, that can easily be generalized to an asymmetrical tensor (by symmetrizing the tensor) and $k \geq 4$.

Chapter 2

Tensor PCA : Intersection of fundamentally different approaches

2.1 From Tensor PCA to the CANDECOMP/PARAFAC Tensor Decomposition

2.1.1 Matrix PCA

Matrix Principal Component Analysis (PCA) is a statistical technique for multivariate analysis first introduced in 1901 [Pea01] and has since become one of the most used statistical methods thanks to its efficiency, simplicity and large number of fields of application. The basic idea of PCA is to perform a dimensionality reduction on a large dataset while keeping most of the statistical information, leading to low-dimensional representations of the datasets in an adaptive and insightful way.

Matrix PCA does so by creating new uncorrelated variables that successively maximize variance. Finding such new variables, the principal components, reduces to solving an eigenvalue/ eigenvector problem. Indeed, a set of observations is given in the form of p $n-$dimensional vectors $\{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_p\}$, or equivalently a $n \times p$ matrix $\boldsymbol{X} \equiv (\boldsymbol{x}_1, \ldots, \boldsymbol{x}_p)$. The objective of this method is to seek new uncorrelated variables consisting in linear combinations of the vectors $\boldsymbol{x}_{i1\leq i\leq p}$ that maximize the variance. These linear combinations could be written as $\sum_{i=1}^{p} a_i x_i \equiv \boldsymbol{X}\boldsymbol{a}$ where $\boldsymbol{a} = (a_1, \ldots, a_p)$ with the additional constraint $\|\boldsymbol{a}\| = 1$. The variances of this new variable is equal to $\text{Var}(\boldsymbol{X}\boldsymbol{a}) = \boldsymbol{a}^T\boldsymbol{S}\boldsymbol{a}$ where $\boldsymbol{S}$ is the sample covariance matrix associated to the dataset $\boldsymbol{X}$ defined as

$$\boldsymbol{S} = \frac{1}{n-1}\sum_{i=1}^{n}(\boldsymbol{x}_i - \bar{\boldsymbol{x}})(\boldsymbol{x}_i - \bar{\boldsymbol{x}})^T \tag{2.1}$$

where $\bar{\boldsymbol{x}} = \frac{1}{n}\sum_{i=1}^{n}\boldsymbol{x}_i$ is the sample mean vector. Given that $\boldsymbol{S}$ is a $p \times p$ real symmetric matrix, it has exactly p eigenvalues λ_k associated to eigenvectors that form an orthonormal set of vectors. The eigenvectors associated to the largest eigenvalue of $\boldsymbol{S}$ solves the problem. They are uncorrelated given that for two such linear combinations $\boldsymbol{X}\boldsymbol{a}_k$ and $\boldsymbol{X}\boldsymbol{a}_{k'}$, $\text{Covar}(\boldsymbol{X}\boldsymbol{a}_k, \boldsymbol{X}\boldsymbol{a}_{k'}) = \boldsymbol{a}_{k'}^T\boldsymbol{S}\boldsymbol{a}_k = \lambda_k\boldsymbol{a}_{k'}.\boldsymbol{a}_{k'} = \delta_{k,k'}$. $\boldsymbol{X}\boldsymbol{a}_k$ are thus called the principal components.

$$\boldsymbol{S}\boldsymbol{a} - \lambda\boldsymbol{a} = 0 \Longleftrightarrow \boldsymbol{S}\boldsymbol{a} = \lambda\boldsymbol{a} \tag{2.2}$$

PCA has an intuitive geometric interpretation as searching the eigendecomposition of the covariance matrix $\boldsymbol{S}$ amounts to performing a singular value decomposition (SVD) on the centred data matrix defined as $\boldsymbol{X}^* \equiv (\boldsymbol{x}_1 - \bar{\boldsymbol{x}}, \ldots, \boldsymbol{x}_p - \bar{\boldsymbol{x}})$ given that

$$(n-1)\boldsymbol{S} = (\boldsymbol{X}^*)^T\boldsymbol{X}^* \tag{2.3}$$

Several adaptations of PCA have been developed and tailored for various objectives and data types in different disciplines such as Robust Principle Component Analysis (RPCA) [CLMW11] which is an adaptation less sensitive to outliers, Multiple Correspondence Analysis (MCA) for categorical variables [GB06], Multiple Factor Analysis (MFA) [AV+07] for variables structured by sets, Functional principal component analysis for continuous variables [RD91], PCA for interval type data [CCDS97]. etc.

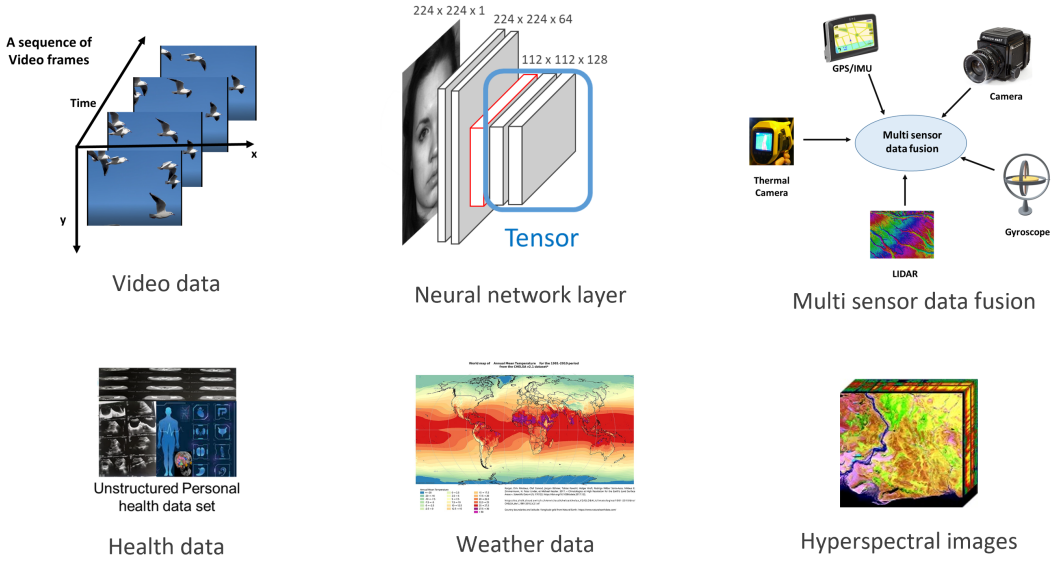


Figure 2.1: Examples of tensorial data

2.1.2 Tensor decomposition

Advances in data acquisition and storage technology have enabled the acquisition of massive amounts of data in a wide range of emerging applications. In particular, numerous applications across the physical, biological, social and engineering sciences generate large multidimensional, multi-relational and/or multi-modal data (see Figure 2.1). Efficient analysis of this data requires dimensionality reduction techniques. However, traditionally employed matrix decompositions techniques such as the singular value decomposition (SVD) and principal component analysis (PCA) can become inadequate when dealing with multidimensional data. This is because reshaping multi-modal data into matrices, or data flattening, can fail to reveal important structures in the data.

The research on multinilear generalizations of linear algebra tools has been very extensive and rich. Tensor decompositions overcome the information loss from flattening. These "tensor methods" have found applications in many fields, including quantitative biology [YAD19], computer graphics [VT04], Hyperspectral analysis [MDDN18], Outlier Detection Methods [DJPM18], Data Recover [SCZL18], Image classification [FH08], face recognition [VT02], quantum computing [PKYA21] and wireless communication [DLG20, CAVP21], among other areas. Thus, tensor generalizations to the standard algorithms of linear algebra have the potential to substantially enlarge the arsenal of core tools in numerical computation.

Tensor decomposition is a fundamental unsupervised machine learning method in data science, with applications including network analysis and sensor data processing. From the various generalizations of matrix SVD to tensors, there is two main tensor decompositions that have been successfully used in numerous applications: (i) CANDECOMP/PARAFAC

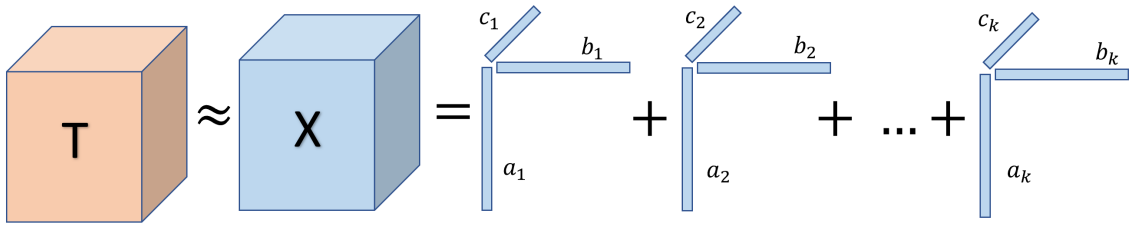


Figure 2.2: CP decomposition illustration.

(CP) decomposition that consists in approximating a tensor with a sum of rank-one tensors. [AGH+14, AGJ15, WA16]. One of the most used algorithms is Alternative Least Squares (ALS) [CLDA09]. (ii) Tucker decomposition that approximates the initial tensor with one small core tensor and a set of matrices [ZX18]. HOSVD [DLDMV00a] and HOOI [DLDMV00b] are the most popular algorithms for this model and their statistical limits have been studied in [ZX18].

CP decomposition

The CANDECOMP/PARAFAC (CP) tensor decomposition is a popular dimensionality-reduction method for multiway data. The canonical CP tensor decomposition expresses an N-way tensor as a sum of rank-one tensors to extract multi-modal structure as illustrated in 2.2. Structural features in the dataset are represented as rank-1 tensors, which reduces the size and complexity of the data. An important interest in resorting to CP tensor decomposition, compared to more standard matrix-based approaches, lies in the uniqueness of the decomposition. This form of dimensionality reduction has many applications including data decomposition into explanatory factors, dimensionality reduction, filling in missing data, and data compression. It has been used to analyze multiway datasets in a variety of domains including neuroscience [NLK+20], quantum chemistry [HTG17], cybersecurity [BSBE+16], latent variable modeling such as hidden Markov models [HKZ12], independent component analysis [BS05] and topic models [AHK12] and so on.

More precisely, CP Tensor Decomposition of a tensor consists in expressing a tensor as a sum of r rank-one tensors:

$$\mathbf{T} \equiv \sum_{i=1}^{r} \lambda_i (\boldsymbol{v}_1^{(i)} \otimes \dots \boldsymbol{v}_k^{(i)}) \tag{2.4}$$

with $\lambda_i \in \mathbb{R}, \boldsymbol{v}_j^{(i)} \in \mathbb{R}^{n_j}, j \in \{1, \dots, k\}$ and r is the rank of the tensor which is the minimal number of rank-one components that can be summed to express $\mathbf{T}$.

In practice, tensors describe data that is corrupted by noise so one resorts to the approximate decomposition for a given rank r :

$$\underset{\lambda_i, \left\|\boldsymbol{v}_j^{(i)}\right\|=1}{\arg\min} \left\| \mathbf{T} - \sum_{i=1}^{r} \lambda_i (\boldsymbol{v}_1^{(i)} \otimes \dots \otimes \boldsymbol{v}_k^{(i)}) \right\| \tag{2.5}$$

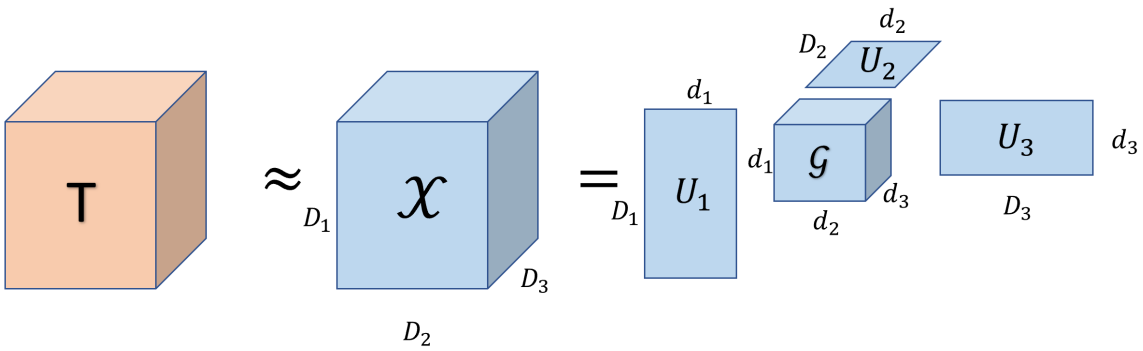


Figure 2.3: Tucker decomposition illustration.

Many algorithms for CP tensor decomposition have been developed in the last years, and many of them are based on methods for best rank-one approximation [dSCdA15a]. The basic idea is to compute successive rank-one approximations of and subtracting it at each step in order to compute the full CP decomposition.

2.1.3 Tucker decomposition

The Tucker decomposition is a generalization of the SVD to higher-order tensors (ie. arrays with more than two indices). This decomposition plays an important role in various domains, such as quantum chemistry [BLHG10], signal processing [MB05] among many others (the multiple applications are detailed in reviews like [KB09]).

In this case we write the tensor as

$$\mathbf{T} = \mathcal{G} \times_1 \boldsymbol{U}_1 \times_2 \boldsymbol{U}_2 \times_3 \boldsymbol{U}_3 \tag{2.6}$$

where $\boldsymbol{U}_1 \in \mathbb{R}^{D_1 \times d_1}, \boldsymbol{B} \in \mathbb{R}^{D_2 \times d_2}, \boldsymbol{C} \in \mathbb{R}^{D_3 \times d_3}$ are the factor matrices, $\times_i$ stands for the mode-i product defined in [KB09] and $\mathcal{G} \in \mathbb{R}^{d_1 \times d_2 \times d_3}$ is denoted the core tensor and characterizes the interactions between the different components of the factor matrices. This is illustrated in Figure 2.3.

HOSVD HOSVD method consists for the factor matrices to be

$$\begin{aligned} \boldsymbol{U}_1 &= \text{first } r_1 \text{ left singular vectors of } \mathbf{T}_{(1)} \\ \boldsymbol{U}_2 &= \text{first } r_2 \text{ left singular vectors of } \mathbf{T}_{(2)} \\ \boldsymbol{U}_3 &= \text{first } r_3 \text{ left singular vectors of } \mathbf{T}_{(3)} \end{aligned} \tag{2.7}$$

and the core tensor $\mathcal{G}$

$$\mathcal{G} = \mathbf{T} \times_1 \boldsymbol{U}_1^T \times_2 \boldsymbol{U}_2^T \times_3 \boldsymbol{U}_3^T \tag{2.8}$$

The n-rank Given a Tucker decomposition of the tensor **T**, the n-rank corresponds to the tuple of the dimensions (d_1, d_2, d_3) of the core tensor.

2.1.4 Tensor PCA and algorithms

Tensor PCA could be seen as a special case that is common to both CP and Tucker decomposition, for a rank of one and a n-rank of (1,1,1) respectively. Indeed, Tensor PCA algorithms aim to find the best rank-one approximation for the input tensor.

Multiple Cp tensor decomposition methods based on been proposed that are based on successive rank-one approximation in [dSCdA15b, PTC15]. These methods consist in careful and subtle deflations as the standard procedure of computing successive rank-one approximations followed by subtractions is not as performant for tensors as for matrices.

Also, trying and modelizing new theoretical models of algorithms in this simple model could be beneficial as it could be easy to generalize it to the tensor decomposition case. One can take as example the deflation method.

Thus, given that Tensor PCA algorithms aim to compute the best rank-one tensor approximation, they could prove useful for Tensor decomposition in general.

which has various applications such as topic modelling [AGJ15], community detection [AGHK13], etc. CP Tensor decomposition also proved useful in the context of deep neural networks, particularly in compressing Convolutional Neural Networks to reduce the memory and the computational cost [AL17, WY+20], Tensor faces [VT02], Hyperspectral imagery [LB13], chemical materials [SB20], multimodal data fusion [LAJ15], data mining [PFS16], etc.

Tensor PCA algorithms

Several algorithms have been developed to tackle the tensor PCA problem. [RM14] analyzed many algorithms theoretically and empirically (in a range of $25 \leq n \leq 800$). The tensor unfolding algorithm showed an empirical threshold of $\beta \sim n^{1/4}$ while naive power iteration with a random initialization performed much worse with an empirical threshold of $n^{1/2}$. [HSSS16] provided an algorithm based on sum-of-squares, which was the first with theoretical guarantees whose threshold matches $n^{1/4}$. Other studied methods have been inspired by different perspectives like homotopy in [ADGM17a], statistical physics ([BAGJ+20], [WEAM19] and [BCRT20]), quantum computing [Has20], low-degree polynomials [KWB19], statistical query [DH21], random tensor theory [OTR22] as well as renormalization group [LOST21].

Power iteration is a simple method that has been extensively used in multiple tensor problems [AGH+12, AGHK13]. [RM14] investigated the empirical performance of power iteration with a random initialization in the range of $n \in [50, 800]$ and observed an empirical threshold of $n^{1/2}$. Through an improved noise analysis, [WA16] showed that for a symmetrical tensor, power iteration is indeed able to recover the signal for a SNR β above $n^{1/2}$ with a constant number of initialization and a number of iterations logarithmic on n. Their experiments in the range of $n \in [25, \ldots, 250]$ suggested that this threshold is tight.

A recent paper [HHYC20] investigates the simple power iteration for a non-symmetric tensor for Tensor PCA, they prove that the algorithmic threshold is strictly equal to $n^{1/2}$. The results of their experiments for $n \in [200, \dots, 800]$ match their theoretical results. In this paper, we aim to draw attention to a surprising observation that contrasts with previous work: if we impose five essential features for an algorithm based on power iteration or gradient descent (use a symmetrized power iteration, impose a polynomial number of initializations and iterations, etc.), we observe that a novel powerful mechanism for the convergence towards the signal takes place, leading to a fundamentally different performance. In fact, for $n \in [50, 1000]$, SMPI is the first algorithm to exhibit an empirical threshold corresponding to $O(1)$ and whose results matches the theoretically-optimal correlation at large n.

2.2 Glassy systems and rough landscapes

2.2.1 Glasses

Standard condensed matter has generally tended to focus on ordered and equilibrium physics. However, since a few decades (the 1970s), the scientific community has started to take more and more interest in disordered and non-equilibrium physics, which has become a big part of statistical physics. The reason for this development is the incredible wealth of behaviors in these systems and the multitude of their applications in materials science, as well as the variety of conceptual problems involved in understanding these behaviors.

A system is considered out of equilibrium if it undergoes a continuous change of its mechanical and statistical properties. Interestingly, the vast majority of daily physical processes, from biology to industry, are out of equilibrium. Thus, the theoretical tools developed to address non-equilibrium dynamics can prove useful in a wide variety of contexts, from mechanics of granular systems to artificial intelligence.

There may be various reasons for not reaching equilibrium. Generally, this is because the time required for the system to attain equilibrium is larger than the observation time scale. In other cases, the action of external forces keep the system out of equilibrium.

Within non-equilibrium physics, one subject has in particular been the subject of very rich and fruitful research: glassy systems. Glasses are amorphous structures (a solid that lacks the long-range order) which experience a so-called vitreous transition (the reverse is called a glass transition). When the temperature of a liquid is rapidly reduced (the system is then said to be quenched), there comes a time when the viscosity highly drops and becomes considered as a glass. The glass-transition temperature T_g is the range of temperature at which this transition occurs. Below it, the relaxation time becomes exceedingly long which prevents the system to reach equilibrium in laboratory or even geological time scales. There are several examples of such material and they vary greatly in size, from macroscopic to microscopic, such as heated metals that are supercooled to form glasses instead of crystals, plastics, colloidal dispersion, and even many biological tissues .

The nature of the glass transition is one of the major unsolved questions of condensed matter science. Below the glass transition temperature T_g, the system remain stuck for a very long time and seems to be frozen in amorphous configurations and the number of these configurations is exponentially large in the system size. The extremely slow relaxation to equilibrium is characterized by the time elapsed so far, sometimes called "age" [BA03, CKR94].

2.2.2 Spin glass and Mean field

The modern theory of spin glasses began with the work of Edwards and Anderson [EA75] on the simple model associated to equation 2.9. Spin glasses are metals with several scattered magnetized defects that exhibits both quenched disorder and frustration (competing interactions). A quenched disorder describes the fact that the J_{ij} in equation 2.9 are random scalars that are constant in the timescales associated to the fluctuations of the σ_i. The frustration property is due to the fact that the interactions are competing and no ground state is able to satisfy them all, which is the reason of the huge number of ground states. Although, the interactions that define the spin glass are in principle quantum mechanical, a classical statistical mechanics approach is enough to recover many of the important phenomena observed down to very low temperatures in a wide variety of spin glasses, provided that we are able to address the complications that arises from the quenched randomness inherent to spin glasses. Indeed, this quenched disorder leads to the absence of spatial symmetries that usually extensively simplify the mathematical study of homogeneous systems such as crystals. Although many basic questions remain open, the study of spin glasses already allowed to uncover numerous important new ideas and techniques that could have wide applicability.

$$\mathcal{H} = -\sum_{\langle x,y \rangle} J_{xy}\sigma_x\sigma_y \tag{2.9}$$

Spin glasses models can also have an addition applying an external magnetic field.

$$\mathcal{H} = -\sum_{\langle x,y \rangle} J_{xy}\sigma_x\sigma_y - h\sum_x \sigma_x \tag{2.10}$$

Strong phenomenological analogies has been unveiled between glassy systems and spin glasses such as glassy dynamics and the aging phenomenon. Glassy systems provide thus simpler models that can be studied analytically. In particular, mean- field models have been most valuable in clarifying some of the basic theoretical issues of glassy systems. Mean field theory of spin glass consists in assuming a large number of spins (thus the absence of space limits) and a equivalent interaction coupling between them which facilitate mathematical analysis. It provided the first quantitative analysis of rough high-dimensional landscapes, in particular of the number and the properties of the critical points, and of the associated dynamics. Yet, the applicability of the results of mean field models to finite dimensional systems is still not very clear.

$$\mathcal{H} = \sum_{i,j,k} J_{ijk}\sigma_i\sigma_j\sigma_k \tag{2.11}$$

2.2.3 Spherical p-spin glass model

The p-spin spherical spin glass model, has attracted a lot of interest in the study of spin glasses. It is a simpler than the Sherrington-Kirkpatrick model and its static and dynamic properties could be studied. In particular, the metastable states could be approximately counted and studied, and the Langevin dynamics shows the aging behavior.

Although the models on which we concentrate are simple in the sense discussed above one needs to master many analytical methods to extract all the richness of their behavior. The model is defined by the Hamiltonian

$$\mathcal{H} = -\sum_{i_1 i_2 \ldots i_p} \mathsf{T}_{i_1 i_2 \ldots i_p}\sigma_{i_1}\sigma_{i_2}\ldots\sigma_{i_p} \tag{2.12}$$

2.2.4 Tensor PCA landscape as a prototypical inference problem and exploration of complex landscapes

Different complex physical systems are characterized by energy functions. Their landscapes possess multiple local minima that are associated to configurations of the system and are separated by barriers, an example is illustrated in Figure 2.4. Complex physical systems are generally characterized by very rough landscapes, which means a large number of such local minima with large barriers to traverse in order to pass from one to the other. Spin glasses, and in particular spherical p-spin glass are typical examples for rough landscapes and their analytical properties have been mathematically studied. This could be essential for the understanding of important physical phenomena such as aging. A graphical example of such landscape is in Figure 2.4.

Deriving new insights on rough landscapes is not only important for the understanding of glassy phenomenon, but is relevant for several areas such as protein folding in biology, string theory in physics, and neural network in machine learning. It could be useful to create and improve algorithms and methods.

Studying the energy landscape aims to have a better understanding of the dynamical behavior of algorithms, in particular local algorithms such as gradient descent and its variants. Given that this questions arises in multiple disciplines especially in computer science. As an example we can give neural networks.

Multiple popular methods in machine learning are based on gradient descent with a lot of success in practice, yet the theoretical analysis of their success is still unclear. Indeed, gradient descent are usually used to optimize high dimensional non convex landscapes with exponentially large number in the system size of local minima, yet gradient descent is able to find relevant optimums and do not get trapped in spurious ones.

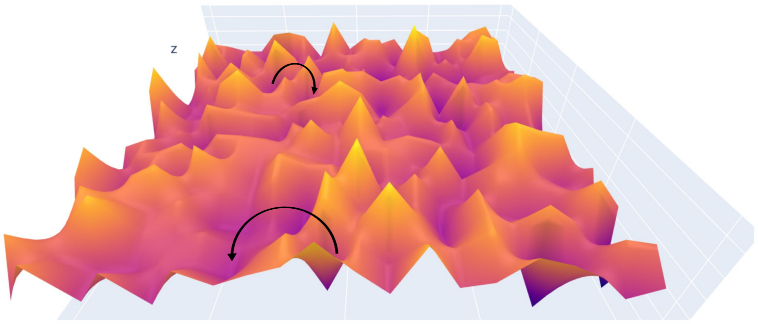


Figure 2.4: Rough landscape illustration

Note that [CHM+15] investigated a very similar landscape to Tensor PCA in order to gain insights on neural network landscapes.

In more details, the problem consists in analyzing the statistical properties of functions defined on very high dimensional spaces. Important statistical information that have been computed is the number of critical points for a given index (number of positive directions) at a given energy, in particular the number of minima and the spectral properties of their corresponding Hessian. This helps to give insights on the dynamics of local algorithms such as gradient descent within these landscapes.

For random landscapes, these questions can be approached within a statistical framework. The spherical p-spin model gives one of the simplest incarnations of a random landscape. In this model the random fluctuations give rise to a rugged landscape [RBABC19], with an exponentially-large (in the dimension N) number $N \exp(N + o(N))$ of stationary points, being their 'complexity'.

Tensor PCA, as well as a weaker version of it, the matrix-tensor PCA introduced in [MBC+20], has been considered by the scientific community as a prototypical inference problem in order to analyze the interplay between the loss landscape and performance of descent algorithms [MKUZ19, MBC+20, BAGJ+20]. They show that there is a region of parameters where the gradient-flow algorithm finds a good global minimum despite the presence of exponentially many spurious local minima and show that this is achieved by surfing on saddles that have strong negative direction towards the global minima.

The paper [RBABC19] show that p-spin glass exhibits two transitions that we denote β_c and β_{stat} as illustrated in Figure 2.5. First, for $\beta < \beta_c$, the most numerous and deepest minima only achieve asymptotically vanishing correlation with the signal. The zone with an exponential number of minima form a band. For $\beta_c < \beta < \beta_{\text{stat}}$, there is a local minimum with non-trivial correlation that detaches itself but the maximum likelihood estimator still has vanishing correlation. Finally, for $\beta_{\text{stat}} < \beta$ the maximum likelihood estimator has strictly positive correlation.

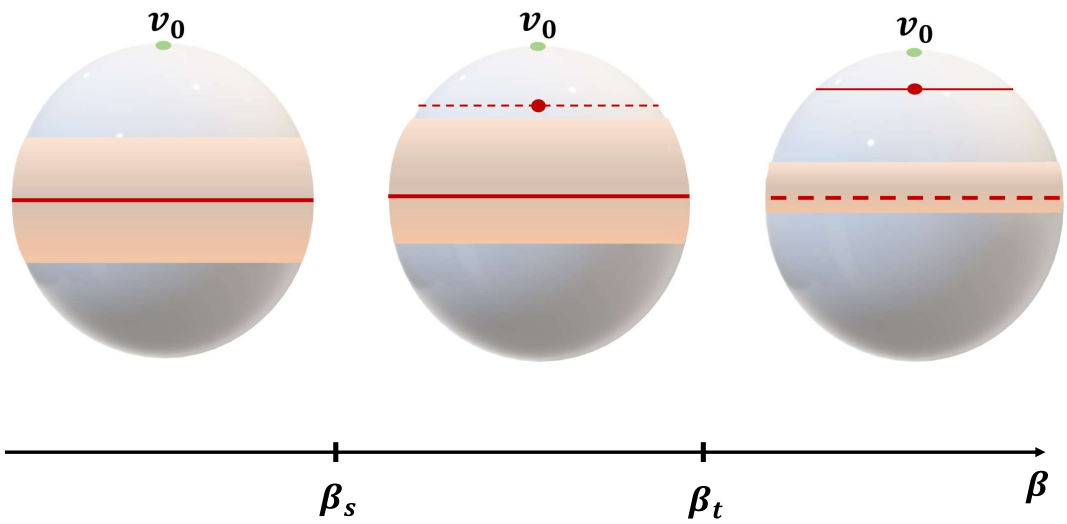


Figure 2.5: Evolution of the minima with the SNR. The light orange zone indicates the zone with an exponential number of minima. The equator is the zone with the most numerous and deepest minima. A straight line indicates the presence of the global minimum, while a dashed line is associated to only local minima.

2.3 The conjectured statistical algorithmic gap: computational hardness

2.3.1 Statistical computational gap in inference problems

The statistical-computational gap is an empirical attribute that is common to various different inference problems. It describes the fact that inference has been proven possible, from an information-theoretically point of view, above a given threshold that we denote statistical (or theoretical) threshold; yet no polynomial-time algorithms is known to succeed below a second larger threshold denoted the computational (or algorithmic) threshold. This intriguing experimental property has attracted a huge amount of interest, especially since it has been observed that several central inference problems exhibit such a gap, including sparse PCA [BR13], tensor PCA [RM14], planted clique [Jer92], random constraint satisfaction problems [ACO08], and many others.

The true nature of this statistical-computational gap is still enigmatic, as it is still not clear whether it is characteristic of an inherent computational intractability in the models that exhibit it, or if a successful algorithm with appropriate properties still needs to be uncovered.

For that purpose, multiple studies addressed this question and obtained important results. These works can roughly categorized in two distinct but complementary approaches:

- Establishing computational equivalence between different problems via average reducing. Typically, one proves that the studied problem is at least as difficult as the average case of a standard model which is presumed to be hard.
- Ruling out families of known algorithms through the study of specific classes of algorithms as well as properties of the geometric landscape associated to the model investigated. Characterizing geometric properties of the problem mainly aims at understanding the behavior of local algorithms such as the gradient-based methods.

2.3.2 Statistical computational gap in Tensor PCA

The statistical threshold of Tensor PCA

Investigating the statistical threshold is less intricate than addressing the computational threshold. Different theoretical tools have been wielded in order to study the best statistical threshold above which inference is possible if we remove the polynomial-complexity constraints.

[JLM+20] gave an exact expression for the statistical threshold β_{stat}. They showed that the maximum likelihood estimator is the most optimal of all measurable estimators in regard of the correlation with the signal vector $\boldsymbol{v}_0$ above the statistical threshold. This optimal correlation is reproduced in Figure 2.6. Similarly to the BBP transition for matrices [BBAP05], the maximum likelihood estimator shows a discontinuous transition: The optimal correlation with the signal vector is equal to zero below the statistical threshold but achieves a correlation close to 1 above the statistical threshold.

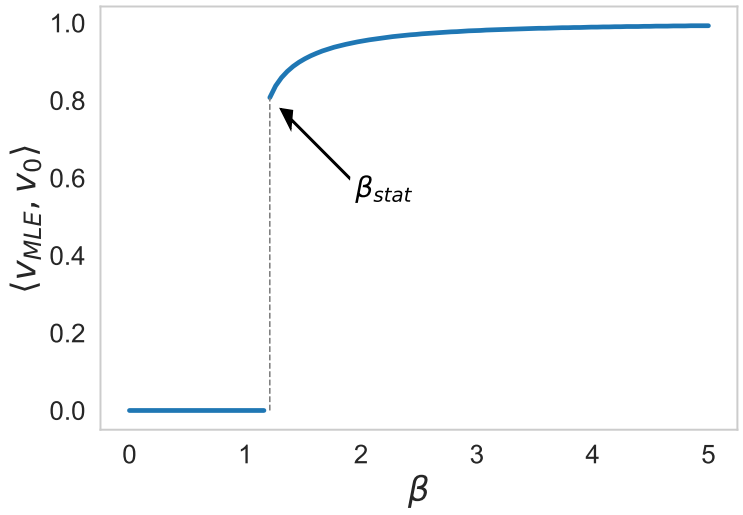


Figure 2.6: Asymptotic correlation of the Maximum Likehood Estimator with the signal as a function of the SNR β.

Among the many tools used to address the statistical threshold, we mention the example of random tensor models (that we present in more details in the next section). [Gur20] proposed a generalisation of the Wigner law to real symmetric tensors. This generalization is based on a new resolvent. He then shows analytically that the expectation of the newly introduced resolvent exhibits a sharp transition at the statistical threshold. This method presents the advantage of being understandable in terms of the eigenvectors concept in tensors.

NP Hardness of Tensor PCA

With the rise of use of tensor methods, researchers has been interested in the computability of multiple tensor problems. It has been shown in [HL13] that, in general, they are fundamentally more difficult compared to their matrix counterpart . In particular, [HL13] proved the following theorem on approximating a tensor with a single rank-one element.

Theorem 1. *[HL13] Rank-1 tensor approximation is NP-hard*

However, NP-hardness only suggests that there exist hard problem instances. This does not rule out the computability of these problems on specific cases of tensors, for instance a low-rank tensor disturbed by a Gaussian random tensor. This represents the distinction between a worst-case problem where every instance need to be solved and an average-case problem which requires only specific instances to be computable. The latter case is more difficult than and overall very different from the former.

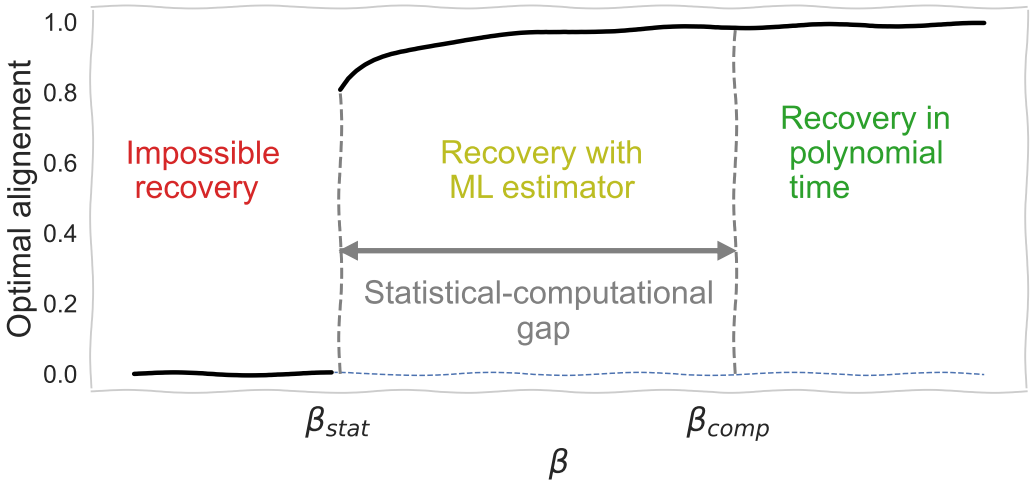


Figure 2.7: Illustration of the statistical-computational gap.

Algorithmic threshold from existent algorithms

While [RM14] proved that the optimal theoretical threshold is of order $\beta_{\text{opt}} = O(1)$, many of the suggested methods have been predicted (based on empirical results for $25 \leq n \leq 1000$) to have at best an algorithmic threshold of $O(n^{1/4})$. This led to a conjecture that it is not possible to achieve the recovery of the signal vector with polynomial time for a β below $O(n^{1/4})$. A rich theoretical literature emerged in order to understand the fundamental reason behind the apparent computational hardness of Tensor PCA. Average-case reduction has been investigated in [BB20, LZ20]. While several papers (such as [PWB+20, LML+17, RBABC19, JLM+20]) provided new results on the statistical threshold of Tensor PCA, there have been many results for thresholds of specific algorithmic models. In particular, [KWB19] proves the failure of low-degree methods for $1 \ll \beta \ll n^{1/4}$ and surveys a recent and interesting line of research that explores the conjecture that the failure of low-degree methods indicates the existence of statistical algorithmic gap in high-dimensional inference problems. [BAGJ+20] provides a possible explanation for the failure of the Langevin dynamics and gradient descent (in the infinitely small learning rate limit) which is another class of algorithms.

2.3.3 Existent approaches for computational hardness of statistical problems

This discrepancy between the statistical and computational thresholds the conjecture of a statistical-computational gap illustrated in Figure 2.7.

Average-Case Hardness of Hypergraphic Planted Clique Detection

Zhang [ZX18] showed a hardness equivalence between Hypergraph Planted Clique (HPC) detection conjecture and Tensor PCA. Luo and Zhang [LZ20] provides some evidence for

Hypergraph Planted Clique (HPC) detection conjecture and investigates the equivalence of computational hardnes between HPC and Planted Clique (PC).

Overlap Gap

The overlap gap is a geometrical property that has been shown to be present in most models known to exhibit a statistical-computational gap. It allows to rule out algorithms classes that shows an input stability which means that a small perturbation on the input leads to a small perturbation on the output. [Gam21].

Low degree polynomials

the low-degree polynomial model cover all the algorithms which outputs could be expressed as a polynomial of degree bounded by $O(log(n))$ on the input entries where n is the system size. This encompasses methods such as Approximate Message Passing [DMM09, BM11], spectral methods, power iteration with a logarithmic number of iterations, etc. This model raised a lot of interest as a framework for addressing the statistical-computational gaps. Indeed, it has been shown that low-degree polynomials are able to match the best algorithmic performance for multiple standard inference problems, for instance planted clique [BHK+19], sparse PCA [DKWB19] and Tensor PCA [HKP+17]. This led to a conjecture that low-degree polynomials could indicate the success of failure of algorithms [Hop18].

Chapter 3

Random Tensor Theory for Tensor PCA

3.1 Brief review of Random Tensor Theory

3.1.1 From eigenvalues to trace invariants

An important concept in problems involving matrices is the spectral theory. It refers to the study of eigenvalues and eigenvectors of a matrix. It is of fundamental importance in many areas. In machine learning, the matrix PCA computes the eigenvectors and eigenvalues of the covariance matrix of the features to perform a dimensional reduction while

ensuring most of the key information is maintained. In this case, the eigenvalues is a very efficient tool to describe data variability. In the case of signal processing, eigenvalue can contain information about the intensity of the signal, while the eigenvector points out to its direction. Lastly, a more theoretical example involves quantum physics where the spectrum of the matrix operator is used to calculate the energy levels and the state associated.

In all of these examples, an important property of the eigenvalues of a n-dimensional matrix $\boldsymbol{M}$ is its invariance under orthogonal transformations $\{\boldsymbol{M} \to \boldsymbol{OMO}^{-1}, \boldsymbol{O} \in \mathrm{O}(n)\}$ where $\mathrm{O}(n)$ is the n-dimensional orthogonal group (i.e. the group of real matrices that satisfies $\boldsymbol{OO}^\top = \boldsymbol{I}_n$, which should not be confused with the computational complexity $O(n)$). Since these transformations essentially just rotate the basis to define the coordinate system, they must not affect intrinsic information like data variability, signal intensity or the energy of a system. The eigenvalues are able to capture some of these inherent information, but recovering the complete general information requires computing their respective eigenvectors (for example to find the principal component, the direction of the signal or the physical state). There are more such invariants than eigenvalues. Another important set worth mentioning are the traces of the n first matrix powers $\mathrm{Tr}(\boldsymbol{A}), \mathrm{Tr}(\boldsymbol{A}^2), \ldots, \mathrm{Tr}(\boldsymbol{A}^n)$. Obtaining them uses slightly different methods than eigenvalues, but they contain the same information since each set can be inferred from the other through some basic algebraic operations.

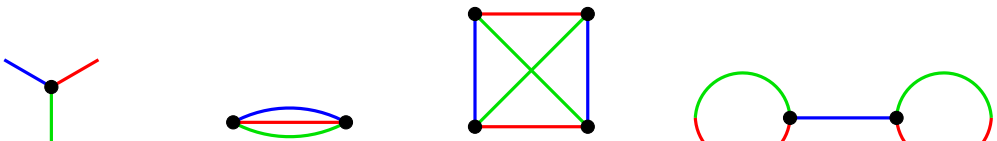
On the basis of the matrix case, we expect that for a tensor $\mathbf{T} \in \bigotimes_{i=1}^{k} \mathbb{R}^{n_i}$, tensor quantities that are invariant under orthogonal transformations ($T_{a_j^1 \ldots a_j^k} \to O^{(1)}_{a_j^1 b_j^1} \ldots O^{(k)}_{a_j^k b_j^k} T_{b_j^1 \ldots b_j^k}$ for $\boldsymbol{O}^{(i)} \in \mathrm{O}(n_i) \quad \forall i \in [k]$) should capture similar intrinsic information like the intensity of the signal, and conceivably, there should be other objects related to these quantities that are able to indicate the direction of the signal. However, the concept of eigenvalue and eigenvector is ill defined in the tensor case and not practical giving that the number of eigenvalues is exponential with the dimension n ([Qi05], [CS13]) and computing them is very complicated. In contrast, we have a very convenient generalization of the traces of the power matrices for the tensors that we call trace invariants. They have been extensively studied during the last years in the context of high energy physics and many important properties have been proven ([Gur17]).

Random Tensor Theory (RTT) provides a set of combinatorial tools dedicated to the study of trace invariant graphs [Gur17]. Trace invariants of a tensor $\mathbf{T} \in \bigotimes_{i=1}^{k} \mathbb{R}^{n_i}$ are tensor networks scalars that are invariant under the following $O(n_1) \times \cdots \times O(n_k)$ transformations:

$$\mathbf{T}_{i_1 \ldots i_k} \longrightarrow \mathbf{T}'_{i_1 \ldots i_k} = \sum_{j_1 \ldots j_k} O^{(1)}_{i_1 j_1} \ldots O^{(k)}_{i_k j_k} \mathbf{T}_{j_1 \ldots j_k}$$

RTT allows to obtain important probabilistic results on trace invariants by using simple enumerative combinatorics. In particular, it gives a simple way to compute the moments (expected value, variance, etc.) of the distribution of these scalars for random tensors. In the following, it should be understood from the context that

Einstein summation convention: It is important to keep in mind throughout the



(a) T_{ijk} (b) $T_{ijk}T_{ijk}$ (c) $T_{ijk}T_{ij'k'}$ $T_{i'jk'}T_{i'j'k}$ (d) $T_{ijj}T_{ikk}$

Figure 3.1: Example of graphs and their associated invariants.

paper that we will follow the Einstein summation convention: when an index variable appears twice in a single term and is not otherwise defined, it implies summation of that term over all the values of the index. For example: $T_{ijk}T_{ijk} \equiv \sum_{ijk} T_{ijk}T_{ijk}$. It is a common convention when addressing tensor problems that helps to make the equations more comprehensible.

3.1.2 Trace invariants and their representations as graphs

We first give a more formal definition of trace invariants. Let $\mathbf{T}$ be a tensor whose entries are $T_{i_1,\dots,i_k}$. Let's define a contraction of a pair of indices as setting them equal to each other and summing over them, as in calculating the trace of a matrix ($\boldsymbol{A}_{ij} \to \sum_{i=1}^{n} \boldsymbol{A}_{ii}$). The trace invariants of the tensor $\mathbf{T}$ correspond to the different ways to contract pairs of indices in a product of an even number of copies of $\mathbf{T}$. The degree of the trace invariants consists in the number of copies of $\mathbf{T}$ contracted. For example, $\sum_{i_1,i_2,i_3} T_{i_1i_2i_3}T_{i_1i_2i_3}$ and $\sum_{i_1,i_2,i_3} T_{i_1i_2i_2}T_{i_1i_3i_3}$ are trace invariants of degree 2.

A trace invariant of degree d of a tensor $\mathbf{T}$ of order k admits a practical graphical representation as an edge colored graph $\mathcal{G}$ obtained by following two steps: we first draw d vertices representing the d different copies of $\mathbf{T}$. The indices of each copy is represented by k half-edges with a different color in $\{1,\dots,k\}$ for each index position as shown in Figure 3.1a. Then, when two different indices are contracted in the tensor invariant, we connect their corresponding half-edges in $\mathcal{G}$. Reciprocally, to obtain the tensor invariant associated to a graph $\mathcal{G}$ with d vertices, we take d copies of $\mathbf{T}$ (one for each vertex), we associate a color for each index position $\{1,\dots,k\}$, and we contract the indices of the d copies of $\mathbf{T}$ following the coloring of the edges connecting the vertices. We denote this invariant $I_{\mathcal{G}}(\mathbf{T})$. Three important examples of trace invariants worth mentioning are: the melon diagram (Figure 3.1b), the tetrahedral (3.1c) and the tadpole (3.1d). [ABGD20] provides a thorough study about the number of trace invariants for a given degree d.

3.1.3 Combinatorial tools for statistics of trace invariants

Covering graph:

In order to be able to compute the moments of trace invariants in a simple way, we introduce the concept of covering graph used in [Gur14]: a covering graph of $\mathcal{G}$ consists in adding $d/2$ new edges of color 0 (also called propagators) relying pairwise the vertices of $\mathcal{G}$. In order to distinguish these edges, we will represent them as dashed lines. If we denote $\mathcal{E}^0(\mathcal{G})$ the edges of color 0 of a graph $\mathcal{G}$, then $\{\mathcal{G}', \mathcal{G}' \backslash \mathcal{E}^0(\mathcal{G}') = \mathcal{G}\}$ denotes the

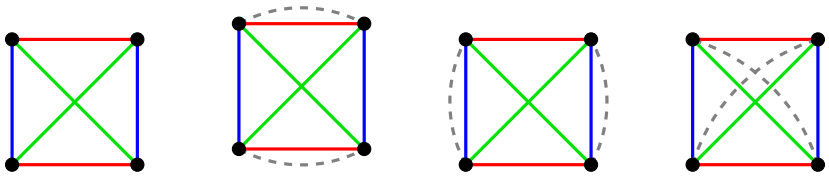


Figure 3.2: Tetrahedral and its covering graph

graphs which restrict to the graph $\mathcal{G}$ when we remove their edges of color 0. These are by the definition the covering graphs. In Figure 3.2 we represent the three covering graphs of the tetrahedral grapha s an example.

Faces of a graph:

Let $c_1, c_2 \in \{0, \dots, d\}$ be two different colors of edges. We denote $F^{c_1,c_2}(\mathcal{G})$ the number of closed cycles (that we also call faces) of 2 colors of $\mathcal{G}$. More explicitly, it consists of the number of connected sub-graphs left when we keep in $\mathcal{G}$ only the edges of colors c_1, c_2. In Figure 3.3, we represent the faces that include the color 0 of the first covering graph of the tetrahedral trace graph.

Simple expectation formula of a trace invariant:

Given a trace invariant $I_{\mathcal{G}}(\mathbf{T})$ and its associated graph $\mathcal{G}$, we can now give a simple formula provided in [Gur17] and based on the two previous concepts to compute the expectations of trace invariants of a random tensor $\mathbf{T}$ whose components are normally distributed

$$\mathbb{E}(I_{\mathcal{G}}(\mathbf{T})) = \sum_{\mathcal{G}', \mathcal{G}' \setminus \mathcal{E}^0(\mathcal{G}') = \mathcal{G}} n^{\sum_c F^{0,c}(\mathcal{G}')}. \tag{3.1}$$

This will be the formula (of enumerative combinatorial nature) that we will use to calculate the expectations of our graphs. We call the expectation and the variance of a graph, the expectation and variance of the tensor invariant associated to it. The details of its derivation are given in the appendix B.

Example of an expectation calculation: the tetrahedral graph Let's calculate the expectation of the tetrahedral graph $\mathcal{B}$:

The first step is to find all the covering graphs of the tetrahedral (drawn in Figure 3.2). Then, for each of the three covering graphs, and for each color c we count the number of faces associated to the colors $(0, c)$. We drew in Figure 3.3 the faces for the different colors for the first graph. For every covering graph, there is one color having two faces (the red for the first covering graph) and two colors having only one face. Hence the expectation of the tetrahedral graph is given by $3n^4$:

$$\mathbb{E}(I_{\mathcal{G}}(\mathbf{T})) = \sum_{\mathcal{G}', \mathcal{G}' \setminus \mathcal{E}^0(\mathcal{G}') = \mathcal{G}} n^{\sum_c F^{0,c}(\mathcal{G}')} = n^{2+1+1} + n^{1+2+1} + n^{1+1+2} = 3n^4. \tag{3.2}$$

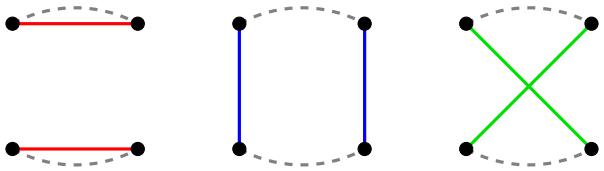


Figure 3.3: The faces of the first covering graph

Variance formula:

Let's denote the graph corresponding to the invariant $I_{\mathcal{G}}^2 = I_{\mathcal{G}}.I_{\mathcal{G}}$, that consists in two copies of $\mathcal{G}$, by $\mathcal{G} \times \mathcal{G}$. The variance formula is

$$\begin{aligned} \text{Var}(I_{\mathcal{G}}) &= \mathbb{E}(I_{\mathcal{G}}^2) - \mathbb{E}(I_{\mathcal{G}})^2 \\ &= \mathbb{E}(I_{\mathcal{G}\times\mathcal{G}}) - \mathbb{E}(I_{\mathcal{G}})^2 \end{aligned} \tag{3.3}$$

Thus, computing the variance for an invariant associated to a graph could be done by using the equation 3.1 for $\mathcal{G} \times \mathcal{G}$ but by taking into account only covering graphs which are connected (there is at least a propagator linking the two copies such as in Figure 3.4) given that $\mathbb{E}(I_{\mathcal{G}})^2$ is the contribution of the disconnected covering graphs . An example of the connected covering graph to be included is given in 3.4.

$$\text{Var}(I_{\mathcal{G}}(\mathbf{T})) = \sum_{\substack{\mathcal{G}', \mathcal{G}' \setminus \mathcal{E}^0(\mathcal{G}') = \mathcal{G}\times\mathcal{G} \\ \mathcal{G}' \text{connected}}} n^{\sum_c F^{0,c}(\mathcal{G}')}. \tag{3.4}$$

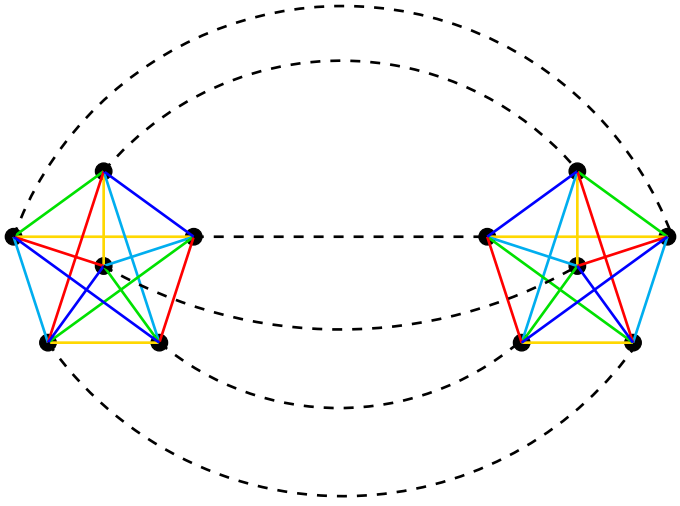


Figure 3.4: Example of a covering graph contributing to the variance

3.2 Random Tensor Theory for Tensor PCA

In this section, we will first demonstrate how to recover the signal in the Tensor PCA model using trace invariants. But first we give the general idea of the proposed framework. As we previously explained, generalizing eigenvalues and eigenvectors to the tensor case is not convenient. Thus, our approach is to associate a matrix to the tensor of interest in order to recover the signal by exploiting the well mastered spectral theory of matrices. However, there is two main important characteristics for these matrices that are required for them to be of interest: they have to be relevant, in the sense that they should reveal the information/signal hidden in the tensor even in low signal regime, and they also have to be easy to study from a probabilistic point of view in order to provide theoretical guarantees. Conveniently, RTT allows us to select matrices that meet these requirements. Indeed, we provide matrices that are able to obtain the signal in the high noise regime, and we have access to simple enumerative combinatorial tools in order to have theoretical guarantees for their performance.

In the remaining of this thesis, we will loosen the definition of an invariant as it will be no more restricted to product of random tensors but could also be an expression of the signal spike tensor $\boldsymbol{v}_0^{\otimes k}$. We will need in this case the generalization of the formula for the expectation of such invariants.

3.2.1 Matrices associated to trace invariants and new tools

Given that our the objective of Tensor PCA is to recover the signal, we should find mathematical objects that are able to provide a vector. To this effect, we introduce a new set of tools in the form of matrices. We denote by $\boldsymbol{M}_{\mathcal{G},e}$ the matrix obtained by cutting an edge e of a graph $\mathcal{G}$ in two half edges (see Figure 3.5 for an example). This cut amounts to not summing over the two indices i_1 and i_2 associated to these two half-edges and using them to index the matrix instead. We will drop the index $\mathcal{G}, e$ of the matrix when the choice of the graph and edge is clear. Advantageously, we can compute the operator norms of these matrices using the same tools described above.

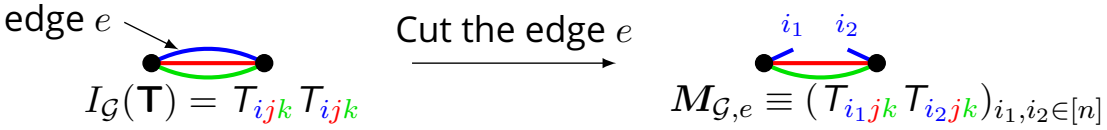


Figure 3.5: Obtaining a matrix by cutting the edge of a trace invariant graph $\mathcal{G}$.

Operator norm of a matrix.

In our proofs we will use Wedin perturbation theorem (stated in Appendix B). This theorem requires computing the operator norm of a matrix $\boldsymbol{M}_{\mathcal{G},e}$ that will be in our case associated to a graph $\mathcal{G}$ and an edge e.

Let $\boldsymbol{A} \in \mathbb{C}^{n\times n}$ be a matrix and $\lambda_1, \ldots, \lambda_n$ its (real or complex) eigenvalues. The spectral radius of $\boldsymbol{A}$ is its highest eigenvalue in absolute value. In particular, when a matrix $\boldsymbol{A}$ is

symmetric, we have the result $\rho(\boldsymbol{A}) = \|\boldsymbol{A}\|_{\text{op}}$. Thus, in order to compute the operator norm of a given matrix $\boldsymbol{M}$, we can determine the spectral radius of the symmetric matrix $\boldsymbol{A} \equiv \boldsymbol{M}^\top \boldsymbol{M}$ and take its square root since $\left\|\boldsymbol{A}^\top \boldsymbol{A}\right\|_{\text{op}} = \|\boldsymbol{A}\|_{\text{op}}^2$.

In order to compute the spectral radius of $\boldsymbol{A} \equiv \boldsymbol{M}^\top \boldsymbol{M}$, we use Gelfand's Formula (stated in Appendix B) with the Frobenius norm. Thus, the formula to keep in mind for computing the operator norm of a matrix $\boldsymbol{M}$ is:

$$\|\boldsymbol{M}\|_\infty = \lim_{r\to\infty} \operatorname{Tr}\Big((\boldsymbol{M}^\top \boldsymbol{M})^r\Big)^{1/2r} \tag{3.5}$$

If we denote λ_{max} the largest eigenvalue of $\boldsymbol{M}$ in absolute value, we have $\operatorname{Tr}\big((\boldsymbol{M}^\top \boldsymbol{M})^r\big)^{1/2r} < n.|\lambda_{\text{max}}|$. Given that λ_{max} is a polynomial of at most degree n of the components of $\boldsymbol{M}$ which are products of Gaussian variables, its expectation exists and is finite. The dominated convergence theorem for random variables thus states that

$$\mathbb{E}(\|\boldsymbol{M}\|_\infty) = \mathbb{E}(\lim_{r\to\infty} \operatorname{Tr}\Big((\boldsymbol{M}^\top \boldsymbol{M})^r\Big)^{1/2r}) = \lim_{r\to\infty} \mathbb{E}(\operatorname{Tr}\Big((\boldsymbol{M}^\top \boldsymbol{M})^r\Big)^{1/2r}) \tag{3.6}$$

Expectation formula in the case of a graph with an open edge.

We encounter this case when we want to study the statistics of a matrix built out of a trace invariant graph.

Let's consider a graph $\mathcal{G}$ of order d associated to a trace invariant $I_{\mathcal{G}}(\mathbf{T})$ where $\mathbf{T}$ is a tensor of order k. Since the trace invariant is a contraction of pairs of indices of d copies of $\mathbf{T}$, we denote the c-th index of the i-th copy of $\mathbf{T}$ by a_i^c and the set of the d indices of the i-th copy by $a_j^{\mathcal{D}}$.

By the definition of the Gaussian measure we have the equation (more details on this formula is given in the appendix B):

$$\mathbb{E}(I_{\mathcal{G}}(\mathbf{T})) = \sum_a \delta_{aa}^{\mathcal{G}} \sum_{\tau\in\mathfrak{S}(k)} \prod_{j=1}^{k} \delta_{a_j^{\mathcal{D}} a_{j\tau(j)}^{\mathcal{D}}}, \tag{3.7}$$

$\sum_a$ indicates a sum over all the indices involved in the computation of the trace invariant, illustrated by a half-edge in the graph (we have $k \times d$ indices in total: k indices for each one of the d copies of $\mathbf{T}$ of the trace invariant expression). $\delta_{aa}^{\mathcal{G}}$ indicates that contracted indices (illustrated by being two ends of the same edge) of the tensors have to be equal. The term $\delta_{a_j^{\mathcal{D}} a_{j\tau(j)}^{\mathcal{D}}}$ indicates that we set equal the c-th indices of the copy j and $\tau(j)$ of $\mathbf{T}$ for $c \in \{1, \dots, d\}$.

Cutting an edge of color c between the i-th and the j-th copies of the tensor, removes the contraction of two indices and leaves them free as variables i_1 and i_2. Then in the formula 3.7, we replace $\delta_{a_j^c, a_k^c}$ by $\delta_{a_j^c, i_1}\delta_{i_2, a_k^c}$ with i_1 and i_2 fixed. Thus, to compute the expectation we use the formula 3.1 with the following addition: if a face of color $0, c$ contains only one of the two vertices of the open edge (if the graph is connected, this only happens when we calculate the variance), its contribution is equal to one. If it contains the two vertices, it is δ_{i_1, i_2}.

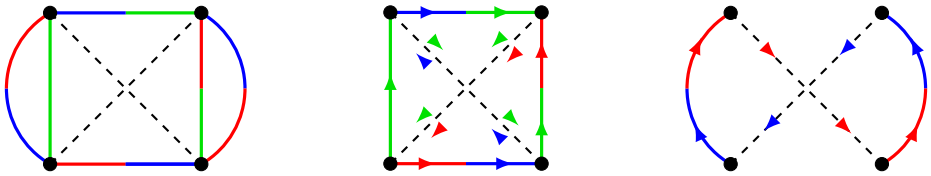


Figure 3.6: A mixed graph and its two cycles

Expectation formula in the case of a graph with edges of mixed color

For a general trace invariant, an edge can have two half edges of different colors. The propagators (edges of color 0) still identify all the indices of their two end vertices as formulated in the equation 3.7. So we adapt the steps in [Gur14] to obtain the equivalent equation 3.8 of the equation B.2 for this new situation. More precisely, the cycles that we will count in a covering graph $\mathcal{G}'$ of $\mathcal{G}$ are built by beginning from a half-edge of $\mathcal{G}$ then alternating until they go back to the starting half-edge, by:

- going from a half-edge of $\mathcal{G}$ of color $c_1 \neq 0$ to its associated half edge of color $c2$.
- going through the propagator to the half-edge of color c_2 connected to the other end of the propagator.

$$\mathbb{E}(I_{\mathcal{G}}(\mathbf{T})) = \sum_{\mathcal{G}', \mathcal{G}' \setminus \mathcal{E}^0(\mathcal{G}') = \mathcal{G}} n^{\text{mixed cycles}} \tag{3.8}$$

This is illustrated in the Figure 3.6.

Expectation formula in the case of a graph with a mix of spikes and random tensors.

We encounter this case when we want to calculate the contribution of the signal to an invariant, which lead us to replace a vertex by the tensorial product of our signal vector. Since we are interested by an expectation, and knowing that the Gaussian distribution is invariant by rotation, we may as well take the signal vector as the unit vector $(1, 0, \ldots, 0)$. Then in the formula B.2 in the appendix B, we replace $\delta_{a_j^c, a_k^c}$ by $\delta_{a_j^c, 0}$ and we restrict the propagators to the tensors vertices. First, we notice that if there is an odd number of spikes, the expectation will be equal to zero. Furthermore, there is no free sum if the index identified in a cycle ends up with a contraction with a spike. Thus, its contribution will be equal to one.

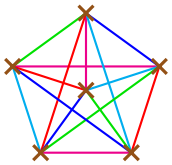
Decomposition of a graph/matrix by explicitly writing the signal and noise terms:

In Tensor PCA, we can represent the tensor from which we hope to extract the signal graphically as:

$$T_{ij_1 \ldots j_{k-1}} = \sqrt{n}\beta\, v_i v_{j_1} \ldots v_{j_{k-1}} + Z_{ij_1 \ldots j_{k-1}}$$

Let's decompose a tensor invariant $I_{\mathcal{G}}(\mathbf{T})$ of degree d using the graphical decomposition of $\mathbf{T} = \mathbf{Z} + \beta \boldsymbol{v}^{\otimes k}$. In each of the graphs illustrating the decomposition (like in Figure 3.7), we call each vertex: a spike vertex if it corresponds to the spike tensor $\boldsymbol{v}^{\otimes 3}$ and a pure noise vertex if it corresponds to the random tensor $\mathbf{Z}$. This leads us to denote the graphs illustrating the decomposition of the tensor invariant (that we will call the decomposition graphs):

- The pure noise graph: the graph where all the vertices are replaced by pure noise vertices. It gives a contribution of $I_{\mathcal{G}}(\mathbf{Z})$.
- The pure signal graph: the graph where all the vertices are spike vertices. It gives a contribution of β^d.
- The intermediate graphs: All the other graphs, which have a mix of noise vertices **and** spike vertices.



(a) Pure signal graph

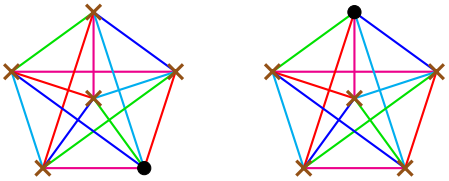


Figure 3.7: Example of a pure signal graph in the top and of intermediate graphs in the bottom

A similar decomposition can be carried out for the matrix based on a graph $\mathcal{G}$ and an edge e. Let's consider a tensor $\mathbf{T}$, a graph $\mathcal{G}$ and its associated trace invariant $I_{\mathcal{G}}(\mathbf{T})$. Let's denote $I'_{\mathcal{G}}(\mathbf{T})$ the invariant associated to the sub-graph obtained by removing from $\mathcal{G}$ the edge e and its two vertices and denote $I'^{N}_{\mathcal{G}}(\mathbf{T}) \equiv I'_{\mathcal{G}}(\mathbf{Z})$ and $I'^{I}_{\mathcal{G}}(\mathbf{T}) \equiv I'_{\mathcal{G}}(\mathbf{T}) - I'^{N}_{\mathcal{G}}(\mathbf{T})$ where . We can distinguish three kind of contributions to the matrix $\boldsymbol{M}_{\mathcal{G},e}$ that we denote $\boldsymbol{M}^{(N)}_{\mathcal{G},e}$, $\boldsymbol{M}^{(I)}_{\mathcal{G},e}$ and $\boldsymbol{M}^{(R)}_{\mathcal{G},e}$, illustrated in Figure 3.8 (where we denoted the invariant $I'_{\mathcal{G}}(\mathbf{T})$ by I' and dropped the index $\mathcal{G}, e$ for simplicity).

We also define the pure signal spike matrix as

$$\boldsymbol{M}^{(S)}_{\mathcal{G},e}(\mathbf{T}) \equiv \boldsymbol{M}_{\mathcal{G},e}(\sqrt{n}\beta \boldsymbol{v}_0^{\otimes k}) \tag{3.9}$$

Lemma 2. $\mathbb{E}(\boldsymbol{M}^{(N)}) = \frac{\mathbb{E}(I^{(N)}_{\mathcal{G}})}{n} \boldsymbol{I}_n$.

Proof. Let $\boldsymbol{M}$ be the matrix obtained by cutting the edge e of a trace invariant graph $\mathcal{G}$. The equation 3.7 could be interpreted as identifying the indices i and j that are ends of

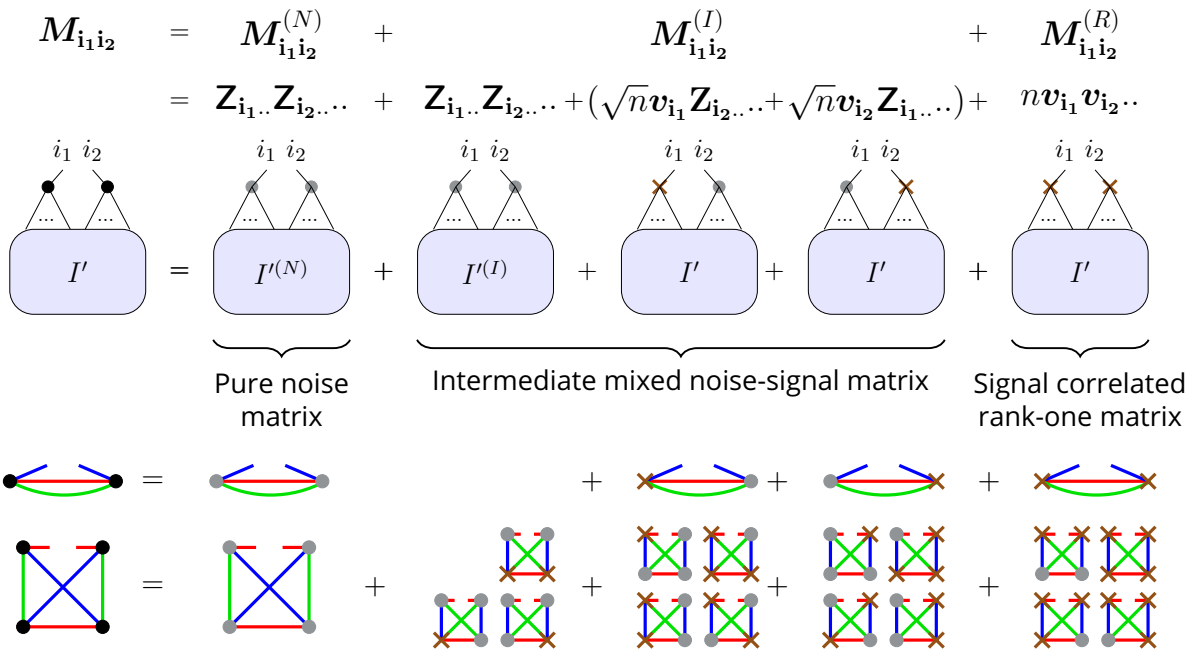


Figure 3.8: Decomposition of a matrix graph and the melon and tetrahedral examples

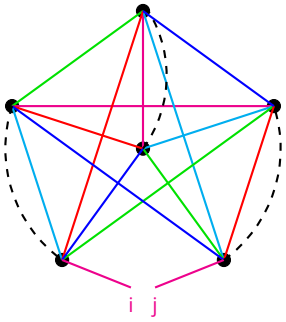
a same edge by including a factor δ_{ij}. Hence, the non diagonal components $M_{ij},\ i \neq j$ have a null expectation since the cycle involving the edge e will identify i and j with a $\delta_{ij} = 0$. By invariance, $\{\mathbb{E}(M_{ii})\}_{i\in[n]}$ are all equal. Then, by linearity of the expectation, $\mathbb{E}(M_{ii}) = \mathbb{E}(\mathrm{Tr}(\boldsymbol{M}))/n = \mathbb{E}(I_{\mathcal{G}}^{(N)})/n$. □

Using the lemma 2, we identify three possible phases depending on which matrix operator norm is much larger than the others:

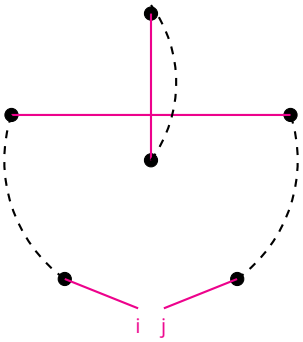
- **No detection and no recovery:** If $\left\|\boldsymbol{M}^{(N)} - \mathbb{E}(\boldsymbol{M}^{(N)})\right\|_{\text{op}} \gg \left\|\boldsymbol{M}^{(I)}\right\|_{\text{op}}, \left\|\boldsymbol{M}^{(R)}\right\|_{\text{op}}$ then no recovery and no detection is possible we can't distinguish if there is a signal. It is for example the phase for $\beta \to 0$.
- **Detection but no recovery:** If $\left\|\boldsymbol{M}^{(I)}\right\|_{\text{op}} \gg \left\|\boldsymbol{M}^{(N)} - \mathbb{E}(\boldsymbol{M}^{(N)})\right\|_{\text{op}}, \left\|\boldsymbol{M}^{(R)}\right\|_{\text{op}}$ then detection but no recovery. We can detect the presence of the signal (thanks to the highest eigenvalue) but we can't recover the signal vector since the leading eigenvector is not correlated to the signal vector.
- **Detection and recovery:** $\left\|\boldsymbol{M}^{(R)}\right\|_{\text{op}} \gg \left\|\boldsymbol{M}^{(N)} - \mathbb{E}(\boldsymbol{M}^{(N)})\right\|_{\text{op}}, \left\|\boldsymbol{M}^{(I)}\right\|_{\text{op}}$. We recover the signal vector. It is for example the phase for $\beta \to \infty$.

3.2.2 Tensor PCA framework for algorithms associated to a trace invariant

Note that our results hold for the large n limit, and the experimental results suggest that the approximation of large n limit is valid for $n \geq 25$. We claim that an event X occurs with high probability if Probability$(X) \to 1$ when $n \to \infty$.

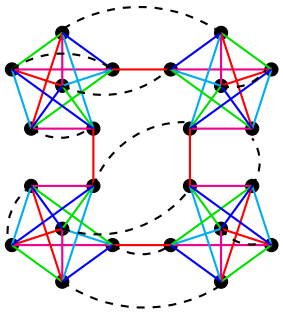


(a) Example of a covering graph for a matrix component

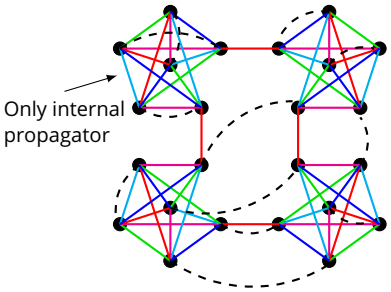


(b) The cycle containing the open edge contributing by δ_{ij}

Figure 3.9: Cycle with an open edge for the expectation of a matrix



(a) Covering graph contributing



(b) Covering graph not contributing to $\mathbb{E}(\mathrm{Tr}\big((\boldsymbol{M}-\mathbb{E}(\boldsymbol{M}))^2\big))$

Figure 3.10: Two covering graphs for the graph of $\mathrm{Tr}\big((\boldsymbol{M}^\top\boldsymbol{M})^2\big)$

The idea of our proofs is to decompose the matrix $\boldsymbol{M}_{\mathcal{G},e}$ into a sum of terms by writing $\mathbf{T}=\mathbf{Z}+\sqrt{n}\beta\boldsymbol{v}^{\otimes k}$ as illustrated in Figure 3.8. Intuitively, our proofs consist in showing that if the signal is large enough the term that only involves a product of the signal $\beta\boldsymbol{v}^{\otimes k}$ will dominate, at large n, the other terms. Thus, the largest eigenvector of $\boldsymbol{M}_{\mathcal{G},e}$ will be close to $\boldsymbol{v}_0$. Therefore, each proof of theoretical guarantees of a graph will mainly consist in bounding the operator norm of the different terms in the sum illustrated in Figure 3.8 by using enumerative combinatorics as described in the precedent section.

Definitions and lemmas

We begin by giving useful lemmas and definitions that will be used in the proofs.

Simplification of the notations: In the remainder of this section, we will often assume that the matrix is symmetric to simplify the notations. One should keep in mind that if the matrix is not symmetric, we have to consider $\boldsymbol{A}=\boldsymbol{M}^\top\boldsymbol{M}$ instead of just $\boldsymbol{M}$, which affect only the notations in the proofs.

The graph used to investigate the operator norm of a matrix:

Let $\boldsymbol{A}=\boldsymbol{M}_{\mathcal{G},e}^\top\boldsymbol{M}_{\mathcal{G},e}$ where $\boldsymbol{M}_{\mathcal{G},e}$ is the matrix associated to a graph $\mathcal{G}$ and an edge e. The trace of the power r of $\boldsymbol{A}$, $\mathrm{Tr}(\boldsymbol{A}^r)$ can be represented as the graph gluing the open edges of $2r$ sub-graphs $\mathcal{G}'$ where $\mathcal{G}'$ is the graph $\mathcal{G}$ with the edge e open, that represents the matrix $\boldsymbol{M}_{\mathcal{G},e}$ (see Figure 3.10). We denote the total graph representing $\mathrm{Trace}(\boldsymbol{A}^r)$ by $\mathcal{G}^{tot}$.

Elementary sub-graph: We call the elementary sub-graphs of $\mathcal{G}^{tot}$ representing $\mathrm{Tr}\big((\boldsymbol{M}^\top\boldsymbol{M})^r\big)$ the sub-graph corresponding to a single matrix $\boldsymbol{M}$. Thus, $\mathcal{G}^{tot}$ consists in gluing together $2r$ elementary graphs, like in Figure 3.10.

Internal propagator: We call a propagator of a covering graph an internal propagator if it connects two vertices of the same elementary sub-graph.

Lemma 3. *The covering graphs involved in the calculation of $E(\mathrm{Tr}(\boldsymbol{M}-\mathbb{E}(\boldsymbol{M}))^r)$ are the same that are involved in the calculation of $E(\mathrm{Tr}(\boldsymbol{M}^r))$ minus the covering graphs where one of the elementary sub-graphs has only internal propagators.*

Proof. We can first begin by showing that if we have two matrices $\boldsymbol{M}_1$ of degree d_1 and $\boldsymbol{M}_2$ of degree d_2 with open edges of the same color denoted l, the covering graphs involved in $\mathrm{Tr}((\boldsymbol{M}_1 - \mathbb{E}(\boldsymbol{M}_1))\boldsymbol{M}_2)$ are the covering graphs of $\mathrm{Tr}(\boldsymbol{M}_1\boldsymbol{M}_2)$ minus the ones where the propagators are internal.

Let's denote the graph associated to $\mathrm{Tr}(\boldsymbol{M}_1\boldsymbol{M}_2)$ by $\mathcal{G}$. To distinguish the $d_1 + d_2$ vertices and their indices, we assign numbers to each vertex. Namely the vertices of $\boldsymbol{M}_1$ are denoted by $1, \ldots, d_1$ with 1 and 2 associated to the two vertices adjacent to the open edge of $\boldsymbol{M}_1$, and $d_1+1, \ldots, d_1+d_2$ for $\boldsymbol{M}_2$ where the vertices of the open edge of $\boldsymbol{M}_2$ are d_1+1, $d_1 + 2$.

We denote all the indices by a, and the indices of all colors associated to a vertex m by $a_m^{\mathcal{D}}$.

A subset of the symmetric group $\sigma(d_1 + d_2)$ are the permutations which could be written as $\tau_1\tau_2$ where τ_1 is a permutation of $(1, \ldots, d_1)$ and τ_2 is a permutation of $(d_1, \ldots, d_1 + d_2)$.

We denote by the symbol $\delta_{aa}^{\mathcal{G}}$ the δ of all the indices that are contracted in the graph $\mathcal{G}$.

Let $\mathcal{G}_1$ the graph associated to $\boldsymbol{M}_1$ without the open edges and $\mathcal{G}_2$ the graph associated to $\boldsymbol{M}_2$ without the open edges.

We have

$$\mathbb{E}(\boldsymbol{M}_1)_{ij} = \delta_{a_1{}^l,i}\delta_{a_2{}^l,j} \sum_{\substack{a \\ a \neq a_1{}^l, a_2{}^l}} \delta_{aa}^{\mathcal{G}_1} \sum_{\tau \in \mathfrak{S}(d_1)} \prod_{l=1}^{d_1} \delta_{a_l{}^{\mathcal{D}} a_{\tau(l)}^{\mathcal{D}}}, \tag{3.10}$$

$$\mathbb{E}(\mathrm{Tr}(\boldsymbol{M}_1\boldsymbol{M}_2)) = \sum_a \delta_{aa}^{\mathcal{G}} \sum_{\tau \in \mathfrak{S}(d_1+d_2)} \prod_{l=1}^{d_1+d_2} \delta_{a_l{}^{\mathcal{D}} a_{\tau(l)}^{\mathcal{D}}}, \tag{3.11}$$

Since the edges of $\mathcal{G}$ are the union of the edges of $\mathcal{G}_1$ and $\mathcal{G}_2$ and two edges of color l connecting the vertices of the open edges: 1 with $d_1 + 1$ and 2 with $d_1 + 2$, we have: $\delta_{aa}^{\mathcal{G}} = \delta_{aa}^{\mathcal{G}^1}\delta_{aa}^{\mathcal{G}^2}\delta_{a_1^l, a_{d_1+1}^l}\delta_{a_2^l, a_{d_1+2}^l}$. This let us write:

$$\begin{aligned}
&\mathbb{E}(\mathrm{Tr}((\boldsymbol{M}_1 - \mathbb{E}(\boldsymbol{M}_1))\boldsymbol{M}_2)) \\
&= \sum_a \delta_{aa}^{\mathcal{G}} \Bigg[\sum_{\tau \in \mathfrak{S}(d_1+d_2)} \prod_{p=1}^{d_1+d_2} \delta_{a_p{}^{\mathcal{D}} a_{\tau(p)}^{\mathcal{D}}} \\
&- \sum_{\tau_1 \in \mathfrak{S}(d_1), \tau_2 \in \mathfrak{S}(d_2)} \prod_{p=1}^{d_1} \delta_{a_p{}^{\mathcal{D}} a_{\tau_1(p)}^{\mathcal{D}}} \prod_{q=d_1+1}^{d_1+d_2} \delta_{a_q{}^{\mathcal{D}} a_{\tau_2(q)}^{\mathcal{D}}} \Bigg],
\end{aligned} \tag{3.12}$$

Thus, the only permutations that are not cancelled are the permutations that does not leave invariant the subset $\{1, \ldots, d_1\}$, which means that $\exists i \in \{1, \ldots, d_1\}$ such that

$\tau(i) \in \{d_1+1, \dots, d_1+d_2\}$. These permutations correspond to the covering graphs that does not have only internal propagators connecting the vertices of $\boldsymbol{M}_1$. This proves our claim.

We can then generalize to the case of a product of multiple matrices by induction, which proves Lemma 3. □

Comparison of the contributions of the graphs and of the matrices:

In our framework, we will in general compare a graph made only of the noise tensor $\mathbf{Z}$ that we denote $\mathcal{G}_1$ with a graph made only of signal spike $\boldsymbol{v}_0^{\otimes k}$ that we denote $\mathcal{G}_2$. We say that the graph $\mathcal{G}_2$ has a much larger contribution than the graph $\mathcal{G}_1$ if

$$\lim_{n\to\infty} \sqrt{\mathbb{E}(I_{\mathcal{G}_1}^2)}/I_{\mathcal{G}_2} = 0 \tag{3.13}$$

with high probability.

This is based on the combination of the Chebyshev inequality and Cauchy-Schwarz inequality.

$$\alpha|\mathbb{E}(I_{\mathcal{G}_1})| + \beta\sqrt{\mathsf{Var}(I_{\mathcal{G}_1})} \leq \sqrt{\alpha\mathbb{E}(I_{\mathcal{G}_1})^2 + \beta\mathsf{Var}(I_{\mathcal{G}_1})} \leq \max(\alpha,\beta)\sqrt{\mathbb{E}(I_{\mathcal{G}_1}^2)} \quad \alpha,\beta > 0 \tag{3.14}$$

In the case of a matrix $\boldsymbol{M}$ with an open edge of color l, we compare the operator norms $\lim_{r\to\infty} \mathrm{Tr}\big((\boldsymbol{M}^\top\boldsymbol{M})^r\big)^{1/2r}$, which requires to compare the contributions of the graphs $\mathrm{Tr}\big((\boldsymbol{M}^\top\boldsymbol{M})^r\big)$. We note that

$$\mathsf{Var}(\mathrm{Tr}\Big((\boldsymbol{M}^\top\boldsymbol{M})^r\Big)) < n^4\mathbb{E}(\mathrm{Tr}\Big((\boldsymbol{M}^\top\boldsymbol{M})^2r\Big)) \tag{3.15}$$

Indeed, the graph of $\mathrm{Tr}\big((\boldsymbol{M}^\top\boldsymbol{M})^2r\big)$ could be seen as taking two copies of the graph of $\mathrm{Tr}\big((\boldsymbol{M}^\top\boldsymbol{M})^r\big)$, cutting in each copy an edge linking two elementary sub-graphs, and gluing the two copies through the cut edges. The difference of contribution of an equivalent covering graph between $\mathrm{Tr}\big((\boldsymbol{M}^\top\boldsymbol{M})^2r\big)$ and two copies of the graph of $\mathrm{Tr}\big((\boldsymbol{M}^\top\boldsymbol{M})^r\big)$ will be at most n^4, due to the four vertices in the two edges that we cut. Based on Chebyshev inequality, we only need to compute the expectation of the operator norm of $\boldsymbol{M}$ for comparison purposes.

We can now state the important algorithm that will be essential for this section. It is important to keep in mind that the following claims concern the large n limit. Empirically, the approximation of large n limit seems valid for $n > 25$.

The proposed algorithm 1 is able to recover the spike in a tensor $\mathbf{T}$ through the construction of the matrix of size $n \times n$ $\boldsymbol{M}_{\mathcal{G},e}(\mathbf{T})$ associated to a given graph $\mathcal{G}$ and edge e.

Theorem 4. *Let $\mathcal{G}$ be a graph of degree d and an edge e, $\exists\ \beta_{rec} > 0$ so that Algorithm 1 with the matrix $\boldsymbol{M}_{\mathcal{G},e}(\mathbf{T})$ gives an estimator $\boldsymbol{v}$ strongly correlated to $\boldsymbol{v}_0$ ($\langle \boldsymbol{v}, \boldsymbol{v}_0\rangle > 0.9$) for $\beta \geq \beta_{rec}$.*

Algorithm 1: Recovery algorithm associated to the graph $\mathcal{G}$ and edge e

Input: The tensor $\mathbf{T} = \sqrt{n}\beta \boldsymbol{v}_0^{\otimes k} + \mathbf{Z}$
Goal: Estimate $\boldsymbol{v}_0$.
Calculate the matrix $\boldsymbol{M}_{\mathcal{G},e}(\mathbf{T})$
Compute its top eigenvector by matrix power iteration (repeat $v_i \leftarrow M_{ij}v_j$).
Output: Obtaining an estimated vector $\boldsymbol{v}$

Proof. Let $n \in \mathbb{N}$, $\mathcal{G}$ a graph of degree d and $\boldsymbol{M}_{\mathcal{G},e}$ the matrix obtained by cutting the edge e. As we previously explained, studying the statistical properties of the operator norm comes down to studying the graph $\mathrm{Tr}\Big((\boldsymbol{M}_{\mathcal{G},e}^\top \boldsymbol{M}_{\mathcal{G},e})^r\Big)$. If we decompose the matrix $\boldsymbol{M}_{\mathcal{G},e}$ as a sum of terms by using $\mathbf{T} = \mathbf{Z} + \sqrt{n}\beta \boldsymbol{v}_0^{\otimes k}$, it is clear that when $\beta \to \infty$ for a fixed n, the matrix with the largest operator norm will be the one where all the vertices are spike vertices (it will have a factor of β^d), and which is proportional to $\boldsymbol{v}\boldsymbol{v}^\top$. Thus, applying Wedin perturbation theorem proves that $\exists \beta_{\text{rec}}$ such that $\forall \beta \geq \beta_{\text{rec}}$ the largest eigenvector of $\boldsymbol{M}_{\mathcal{G},e}$ will be close to $\boldsymbol{v}_0$, and therefore the recovery is possible. □

Example: If we take the matrix defined as $(\boldsymbol{M}_{\mathcal{G},e})_{i_1 i_2} \equiv \mathbf{T}_{i_1 jk}\mathbf{T}_{i_2 jk}$ we have $(\boldsymbol{M}_{\mathcal{G},e})_{i_1 i_2} = \mathbf{Z}_{i_1 jk}\mathbf{Z}_{i_2 jk} + \sqrt{n}\beta(\mathbf{Z}_{i_1 jk}(\boldsymbol{v}_0^{\otimes 3})_{i_2 jk} + \mathbf{Z}_{i_2 jk}(\boldsymbol{v}_0^{\otimes 3})_{i_1 jk}) + n\beta^2(\boldsymbol{v}_0)_{i_1}(\boldsymbol{v}_0)_{i_2}$. So intuitively, in the large β limit $\boldsymbol{M}_{\mathcal{G},e_{i_1 i_2}}$ will be approximately proportional to $(\boldsymbol{v}_0)_{i_1}(\boldsymbol{v}_0)_{i_2}$.

Since the algorithm 1 consists in algebraic operations on the tensors entries, it is very suitable for a parallel architecture (for example by computing independently each entry of the matrix $\boldsymbol{M}_{\mathcal{G},e}(\mathbf{T})$). The following Theorem 5 gives a lower bound to the threshold above which we can recover a spike using a single graph of finite size (independent of n). Interestingly, this threshold which appears naturally in our framework, matches the threshold below which there is no known algorithm that is able to recover the spike in polynomial time. We call the Gaussian variance of a graph $\mathcal{G}$, the variance of the invariant $I_{\mathcal{G}}(\mathbf{B})$ where B_{ijk} are Gaussian random.

Theorem 5. *Let $k \geq 3$. It is not possible to recover the signal of a constant degree using a single graph below the threshold $\beta = O(n^{(k-2)/4})$ which is the minimal Gaussian variance of any graph $\mathcal{G}$.*

Proof. We aim to prove that $\mathbb{E}((I_{\mathcal{G}})^2) \geq n^{dk/2}$ for any graph $\mathcal{G}$ with Gaussian tensor. As previously noted, the graph corresponding to the invariant $(I_{\mathcal{G}})^2$ consists in two copies of $\mathcal{G}$ that we denote $(\mathcal{G})^2$, as in Figure 3.11. A simple way to prove our claim, is to exhibit for any graph $\mathcal{G}$ a covering graph for $(\mathcal{G})^2$ that gives a contribution of $n^{dk/2}$. Such a covering graph is obtained by connecting each vertex of $\mathcal{G}$ to its equivalent in the second copy as shown in Figure 3.11. All the cycles for the colors $0, c$ will be of length 2 (consisting in an edge and its equivalent in the other copy). Their total number is equal to $dk/2$, so this covering graph gives a contribution equal to $n^{dk/2}$. Since the expectation is equal to the sum of the contributions of the covering graphs, the variance will always be larger than $n^{dk/4}$. A similar argument proves that the expectation of $\mathrm{Tr}\big((MM^\top)^r\big)^2$ will be always superior to n^{rd}.

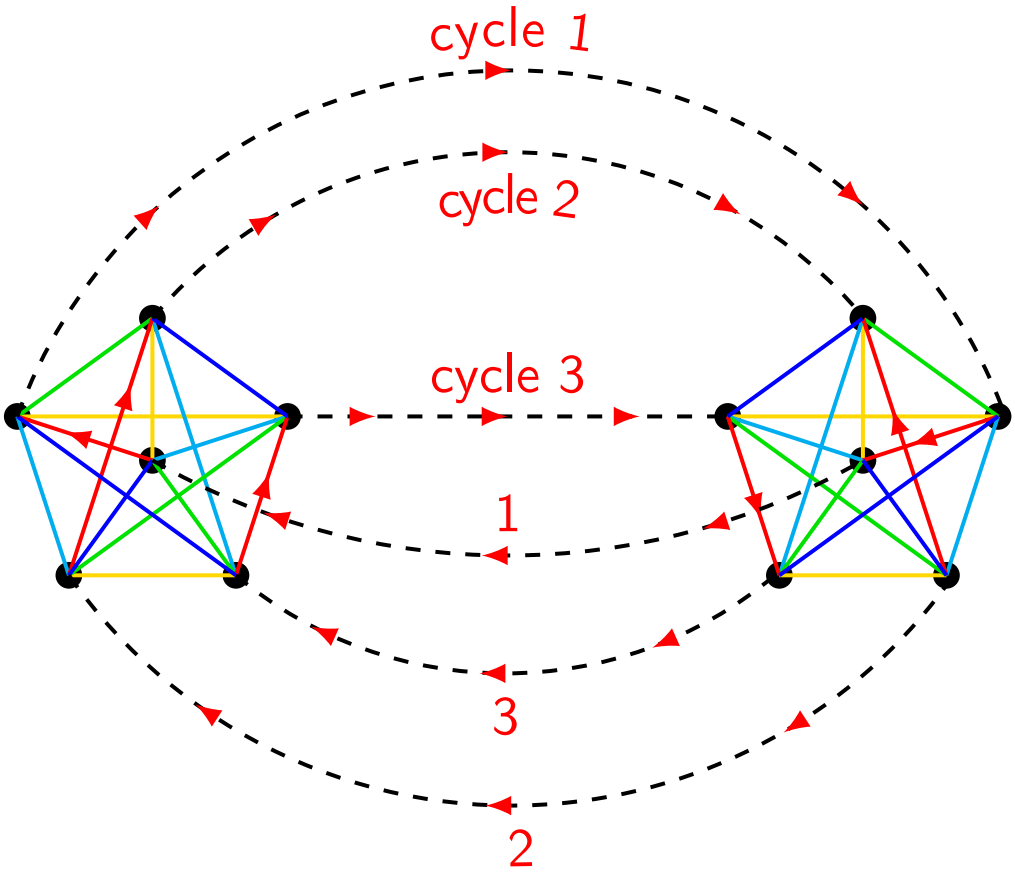


Figure 3.11: The covering graph contributing by $n^{kd/2}$ to the variance, implying that no algorithm associated to a single graph could recover a signal below the actual computational threshold.

□

The following lemma 6 has as an important consequence that if we want to investigate if a matrix associated to a graph $\mathcal{G}$ and edge $\boldsymbol{e}$ recovers the signal at $\beta = \mathcal{O}(n^{1/4})$, we can disregard the intermediate matrices and compare only $\boldsymbol{M}^{(S)}$ and $\boldsymbol{M}^{(I)}$. This will be used implicitly in the remaining of the proofs.

Lemma 6. *If for $\beta \gg n^{(k-2)/4}$, the norm operator of the pure signal matrix $\boldsymbol{M}^{(S)}$ is larger than the norm operator of the pure noise matrix $\boldsymbol{M}^{(N)}$, the operator norm of $\boldsymbol{M}^{(I)}$ is negligible comparing to the operator norm of $\boldsymbol{M}^{(S)}$ for $\beta \gg n^{(k-2)/4}$.*

Proof. The proof is based on the fact that the expectation of the square of a connected graph with mixed noise and signal spikes can't be larger than the square of its pure noise equivalent graph below $\beta = \mathcal{O}(n^{(k-2)/4})$. Thus, given that the pure signal graph is a polynomial on β with higher degree than the intermediate graphs, it means that it will have the larger contribution for $\beta \gg n^{(k-2)/4}$.

We will consider a general graph associated to an invariant but the demonstration for the matrices $\boldsymbol{M}$ follows the same steps but instead of $\mathcal{G}^2$, one has to consider $\mathrm{Tr}(\boldsymbol{M}^r)^2$.

Let $\mathcal{G}$ a connected graph whose signal $\mathcal{G}^S$ ($\mathcal{G}$ with only signal spikes vertices) and noise graphs $\mathcal{G}^N$ ($\mathcal{G}$ with only Gaussian tensor vertices) has a similar operator norm for $\beta = n^{(k-2)/4}$.

Let's assume that an intermediate graph $\mathcal{G}^{\text{int}}$ (with r vertices replaced by signal spikes) has a largest contribution than $\mathcal{G}^N$ for $\beta \leq n^{(k-2)/4}$. Let's take the covering graph of $(\mathcal{G}^{\text{int}})^2$ that is responsible for the largest contribution.

We build a covering graph for the pure noise graph $(\mathcal{G}^N)^2$ from this covering graph that has a superior or equal contribution than the initial covering graph for $(\mathcal{G}^{\text{int}})^2$. To do so, we disconnect the propagators that are connected to signal spike vertices from these signal vertices and glue them together (in any way we want), while we connect by a new propagator each signal spike vertex to its equivalent in the mirror graph. This covering graph will give a superior or equal contribution to the expectation of the square of the intermediate graph since:

- The cycles that didn't involve a spike stay give the same contributions
- The cycles that involved a spike were giving a contribution equal to one in addition to the $(\sqrt{n}\beta)^p$ direct contributions of the p signal spikes vertices. But in the new covering graph of the pure noise graph, the contribution of these cycles will be $n^{3p/4}$. Indeed each cycle has at most four signal spike vertices, as it connects a signal spike to its equivalent in the mirror graph.

□

Comparison of matrices that appears in our decomposition

One needs to only compare the pure noise matrix and the pure signal matrix to determine if the algorithm associated to a connected graph succeeds above the computational threshold $n^{\frac{k-2}{4}}$.

Using these tools and this algorithm, we are now able to investigate the performance of our framework in various theoretical settings. In the first two paragraphs, we study the algorithms associated to two trace invariants of degree 2. They consist of the melonic diagram and the tadpole diagram. Interestingly, it turns out that they are equivalent to the two state-of-the-art algorithms for Tensor PCA the tensor unfolding and homotopy-based method as illustrated in Figure 3.12. Thus, we decide to goes further in terms of graph's degree and investigate the algorithms associated to the perfect one-factorization graphs (consisting in the tetrahedral when $k = 3$). In the last subsection, we will prove that our methods allows us to derive a new algorithmic threshold for more general cases.

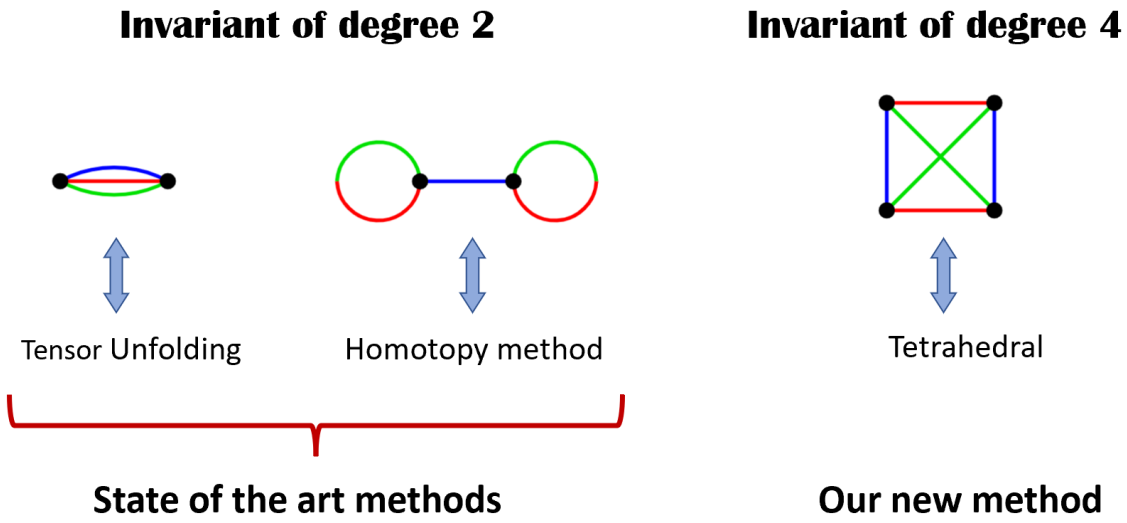


Figure 3.12: Equivalence between algorithms associated to graphs and state of the art methods.

A framework that provides algorithms with different performance/complexity trade-offs

This framework offers multiple algorithms with different performance-complexity trade-off. The state-of-the-art algorithms correspond to degree two invariants. Higher degree graphs offer better empirical results but with higher computational complexity.

3.2.3 Some derived algorithms from the new framework

In this subsection we will investigate the melon graph and the tadpole graph. These two invariants of degree two actually correspond to existent algorithms that represents the state of the art, respectively the Tensor Unfolding and homotopy-based algorithms. Thus, we take a step forward and investigate a higher degree graph. the perfect one-factorization, that is the tetrahedral in the case $k = 3$ is a good choice for a next step to investigate how algorithms associated to higher degrees graphs perform.

Melonic graph

Let's consider the invariant $T_{i_1 \dots i_n} T_{i_1 \dots i_n}$ (illustrated by the graph in Figure 3.1b when $k = 3$). Its recovery algorithm (with the matrix obtained by cutting any of the edges) is similar to the tensor unfolding method presented in [MR15].

Theorem 7. *The algorithm 1 with a matrix $M_{\mathcal{G},e}$ where $\mathcal{G}$ is the melon graph with $\beta_{rec} = O(n^{(k-2)/4})$ in linear time and $O(n^2)$ memory requirement.*

Proof. Let's prove by induction the following

$$\mathbb{E}(\mathrm{Tr}\Big(((\boldsymbol{M}-\mathbb{E}(\boldsymbol{M}))(\boldsymbol{M}-\mathbb{E}(\boldsymbol{M}))^\top)^r\Big)) \leq n^{kr+1} \tag{3.16}$$

where $\boldsymbol{M}$ is the matrix associated to the melon by opening one of its edge (all the edges are equivalent in this situation given that the dimensions are equal).

- For $r = 1$ it is straightforward by simply enumerating the cycles and applying the formula 3.1.
- Let $r > 1$ and let's assume the statement is true for all $r' < r$. Let's consider one of its elementary sub-graphs (defined in the beginning of the section). Using lemma 3, we can divide the edges of the sub-graph in two sets:
  - The $k-1$ edges that are in faces of at least two edges.
  - One external edge divided in half edges.

  If the $k-1$ edges are in faces of exactly two edges, it means that the propagators at both the left and right vertices are connected to the two vertices of another elementary sub-graph.

  This will lead to two possibilities:

  - The two elementary sub-graphs are next to each other. The expectation of $\mathrm{Tr}\big((\boldsymbol{M}\boldsymbol{M}^\top)^r\big)$ is equal to the expectation of the graph after removing these two elementary sub-graphs and connecting the two left open edges, to which we add the contribution of n^k from the two elementary sub-graphs. Using the induction hypothesis, the graph with the two sub-graphs removed has a contribution of maximum $n^{k(r-1)+1}$. Combining them we have a total contribution of $n^{k(r-1)+1+k} = n^{kr+1}$ which proves the theorem.
  - The two sub-graphs are not next to each other: Then the open edges have a maximal contribution of $1/2$. Cutting this elementary sub-graph and using the induction hypothesis proves the theorem.

  Thus, using Gelfand's Formula, the norm operator is equal to $n^{k/2}$. The signal matrix has an operator norm equal to $(\sqrt{n}\beta)^2$, which gives a recovery threshold for the melonic algorithm of $\beta = n^{(k-2)/4}$ using Wedin perturbation.

For the symmetric case, since $T^{sym}_{ijk} = \sum_{\tau\in\mathfrak{S}_k} T_{\tau(i)\tau(j)\tau(k)}$ where $\mathfrak{S}_k$ is the symmetric group introduced previously. We just need to expand the melonic matrix $T^{sym}_{i_1jk}T^{sym}_{i_2jk}$ in function of the asymmetric tensor $\mathbf{T}$. We obtain a sum of nine matrices (corresponding to graphs with mixed colors). We can show that the operator norm of each of these matrices is at most equal to $n^{3/2}$. □

Tadpole graph

Figure 3.1d has a special characteristic: we can obtain two disconnected parts by cutting only one line. Therefore, the matrix obtained by cutting that edge is of rank one (in the form of $\boldsymbol{v}\boldsymbol{v}^\top$). Thus, the vector $\boldsymbol{v}$ has a weak correlation with the signal $\boldsymbol{v}_0$, which allow

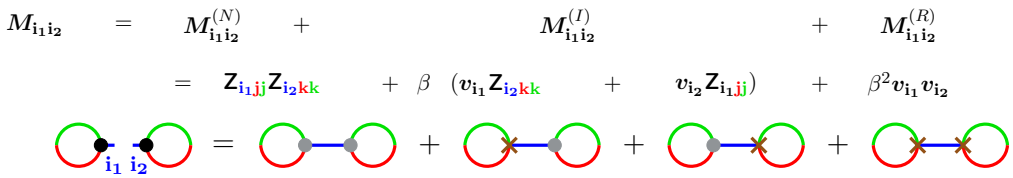


Figure 3.13: Decomposition of a tadpole matrix

the tensor power iteration ($v_i \leftarrow T_{ijk} v_j v_k$) to empirically recover it (formal proofs require to consider some more sophisticated variants of power iteration like in [ADGM17b] and [BCRT20]). This algorithm is a variant of the already existent homotopy algorithm.

Theorem 8. *The tadpole graph allows to recover the signal vector for $k \geq 3$ and $\beta = O(n^{(k-2)/4})$ by using local algorithms to enhance the signal contribution of the vector T_{ijj}.*

Proof. Using the matrix decomposition illustrated in 3.13, we can see that we have a sum of four matrices of rank one. Using the usual formula to calculate the expectation of the operator norm of the matrices we obtain:

- $\left\|\boldsymbol{M}^{(N)}\right\| = n^2$
- $\left\|\boldsymbol{M}^{(I)}\right\| = n$
- $\left\|\boldsymbol{M}^{(R)}\right\| = n^{3/2}$

we can see that the first one dominates by $n^{1/4}$ the last one (consisting in the signal vector) so we are able to use power iteration or one of its variants (like in [ADGM17b] and [BCRT20]) to recover it.

□

Tensor invariant based on the perfect one-factorization graphs

The theoretical physics community that had developed the theory of trace invariants for tensor have made a particular focus on a family of graphs called the perfect one-factorization graphs [FRV19b] (more details in the appendix). This focus is motivated by their nice combinatorial properties due to their symmetries. It was then natural for us to explore the algorithmic potential of these graphs in our tensor decomposition context. Our first candidate was the simplest next graph which is of degree strictly superior to two named tetrahedral graph. Our investigation through the tetrahedral shows that for $k = 3$, the algorithms based on the tetrahedral graph shows a very interesting improvement of empirical results and thus highlights the richness of the proposed framework. Moreover, the properties of this family of graphs also simplify the proofs for recovery theorems. Therefore, the standard methods involved in the demonstrations of these theorems are instructive for the study of more general graphs.

Theorem 9. *The algorithm associated to a perfect one-factorization graph is able to recover the signal vector for $\beta = O(n^{(k-2)/4})$.*

Proof. Let G_i be the elementary sub-graphs of $\mathcal{G} \equiv \mathrm{Tr}((\boldsymbol{M})^r)$ and let's define $(s_i)_{i\in\{0,\dots,r\}}$ such that $s_i/2$ is equal to the number of propagators having their two vertices in the sub-graph $\mathcal{G}_i$ (the internal propagators). Thus, we have $s/2 = \sum_i s_i/2$ internal propagators in total. We can divide the edges of the sub-graph $\mathcal{G}_i$ in three classes. Each edge could be part of a face of either one edge, two edges, three edges, etc.. Let's count how many we can have of each type in an optimal configuration (maximizing the number of faces) for each class.

- Edges only connected to internal propagators, there is at most $\sum_i s_i(s_i-1)/2$ of such edges.
  - We can have a maximum of $s_i/2$ faces of one edge since that is the number of internal edges.
  - Then, from the $s_i(s_i-2)/2$ remaining ones, if we want to put them in faces of two edges, the two edges needs too to be in the same sub-graph. We can create a maximum of $s_i/2(s_i/2-1)/2 = s_i(s_i-2)/8$ different couples of propagators. Each couple can't create more than one face since otherwise it will contradict the maximally single trace hypothesis of the graph. So there is a maximum of $s_i(s_i-2)/4$ edges that can be put in two edges faces.
  - The remaining $s_i(s_i-2)/4$ will be in faces of at least three edges.
- Edges connected to one internal and one external propagator, there is at most $\sum_i s_i(k+1-s_i)$ of such edges. They have to be in faces of at least three edges.
- Edges only connected to external propagators, there is at most $\sum_i ((k+1-s_i)(k-s_i)/2)$ such edges. They can only be put in faces of at least two edges.
- The open edge (defined as the two half-edges we use to glue the elementary sub-graphs to the rest of the graph) can be put in a face of maximum one edge.

So we have a maximal number of faces given by:

$$\begin{aligned}&\sum_{i=1}^{r}\Big(\frac{s_i}{2}+\frac{s_i(s_i-2)}{8}+\frac{s_i(s_i-2)}{12}\\&+\frac{s_i(k+1-s_i)}{3}+\frac{(k+1-s_i)(k-s_i)-2}{4}+1\Big)\\&=\sum_{i=1}^{r}\left(\frac{k(k+1)}{4}-s_i\frac{4k-3s_i-4}{24}+\frac{1}{2}\right).\end{aligned} \tag{3.17}$$

Let's focus on the case where $s_i = 0$. If all the points of the elementary sub-graph are connected to points of only one other elementary sub-graph, then the two vertices of the open edge connect to the same elementary sub-graph. In that case the open edge has a contribution of at most $1/2$. In the other case, then there is at least k edges which contributes only by at most $1/3$ making them lose $1/6$. Either way the contribution of the sub-graph can't go further then $k(k+1)/4$. Taking into account this observation, we can write the maximal number of faces as:

$$\sum_{i=1}^{r}\left(\frac{k(k+1)}{4}-s_i\frac{4k-3s_i-4}{24}+\frac{1-\delta_{s_i,0}}{2}\right). \tag{3.18}$$

We can easily check that this quantity is always smaller than $\frac{rk(k+1)}{4}$ for $k>5$. The case $k=3$ (the tetrahedral) and $k=4$ could be easily done by manually counting the number of cycles. Thus

$$\lim_{r\to\infty}\mathrm{Tr}\Big((\boldsymbol{M}^{(N)})^r\Big)^{1/r}\sim n^{\frac{k(k+1)}{4}} \tag{3.19}$$

On the other hand, the operator norm of the pure signal matrix $\boldsymbol{M}^{(S)}$ is equal to $(\sqrt{n}\beta)^{(k+1)}$ thanks to the $k+1$ vertices. $\boldsymbol{M}^{(S)}$ dominates $\boldsymbol{M}^{(N)}$ for $\beta\gg\frac{k-2}{4}$. This completes the proof. □

3.2.4 New theoretical threshold for an asymetrical tensor with different dimensions $n_1\neq n_2\neq\ldots\neq n_k$

We consider the more general case where the tensor $\mathbf{T}$ has axes of different dimensions n_i ($\mathbf{T}\in\bigotimes_{i=1}^{k}\mathbb{R}^{n_i}$). We can assume without any loss of generality that $n_1\geq n_2\geq\cdots\geq n_k$.

$$\mathbf{T}=\beta\boldsymbol{v}_1\otimes\cdots\otimes\boldsymbol{v}_k+\mathbf{Z}\quad\text{where}\quad\boldsymbol{v}_i\in\mathbb{R}^{n_i},\quad n_i\in\mathbb{N}. \tag{3.20}$$

Our framework naturally handles this case and allows us to derive a new algorithmic threshold. It is, to the best of our knowledge, the first generalization of the threshold $\beta=n^{(k-2)/4}$ derived in [RM14] when $n_i=n\ \forall i\in[k]$ and appears easily using our framework.

Theorem 10. *Using the melon graph, the threshold for $\boldsymbol{v}_1$ is given by $\max\left((\prod_{i=1}^{k}n_i)^{1/4},n_1^{1/2}\right)$ while the thresholds for $\boldsymbol{v}_j,\ j\geq 2$ are equal to $(\prod_{i=1}^{k}n_i)^{1/4}$.*

This result coincides with the previously known threshold when $n_i=n$. We also note that the threshold $\max\left((\prod_{i=1}^{k}n_i)^{1/4},n_1^{1/2}\right)$ for $\boldsymbol{v}_1$ is not surprising. Indeed, if $n_2=\cdots=n_k=1$, the tensor can be seen as a vector of dimension n_1. And since the expectation of the norm of a random Gaussian vector is $n_1^{1/2}$, $\beta>n_1^{1/2}$ is required to be able to detect the signal vector.

Proof. Let's consider the melonic graph with the edge associated to the index of the space of dimension n_l open. It is associated to the matrix:

$$M_{pq}=T_{i_1\ldots i_{l-1}pi_{l+1}\ldots i_k}T_{i_1\ldots i_{l-1}qi_{l+1}\ldots i_k} \tag{3.21}$$

For simplicity the graphs will be drawn for $k=3$.

We will treat two separate cases:

- Case $n_l\leq\prod_{i\neq l}n_i$:

  Let's prove by induction the statement: S_r: $\mathbb{E}[\mathrm{Tr}((\boldsymbol{M}-\mathbb{E}(\boldsymbol{M}))^r)]\sim n_l\prod_{i=1}^{k}n_i^{r/2}$.

- Initialisation with $r = 2$: For $\boldsymbol{M} - \mathbb{E}(\boldsymbol{M})$ we have the following graph and its covering graphs:

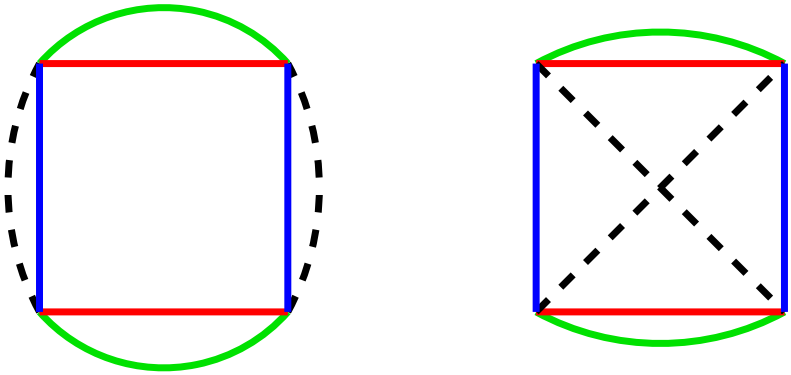


Figure 3.14: The two covering graphs of $\mathrm{Tr}\big((\boldsymbol{M} - \mathbb{E}(\boldsymbol{M}))^2\big)$

which gives

$$\mathbb{E}(\mathrm{Tr}((\boldsymbol{M} - \mathbb{E}(\boldsymbol{M}))^r) = n_l \prod_{i=1}^{k} n_i + \prod_{i=1}^{k} n_i \leq 2n_l \prod_{i=1}^{k} n_i^{r/2}$$

- Assume S_r is true for all $r' < r$, let's prove for r that no covering graph has a contribution strictly superior to $n_l \prod_{i=1}^{k} n_i^{r/2}$ and then, that we can provide a covering graph that contribute by exactly $n_l \prod_{i=1}^{k} n_i^{r/2}$ when r is even.
  * In the case where there is no cycle of color l of length 1 (which means a propagator that shares the same ends that an edge of color l), then all edges have a contribution of $1/2$ at maximum to the number of faces (no internal propagators are allowed in the covering graph of $\mathbb{E}(\mathrm{Tr}((\boldsymbol{M} - \mathbb{E}(\boldsymbol{M}))^r)$ given lemma 3). So the contribution of that covering graph will be at most equal to $\prod_{i=1}^{k} n_i^{r/2}$.
  * In case there is such a propagator, which means that there is a propagator coinciding with an edge of color l between two elementary sub-graphs. Let r Then either the two other ends of the two elementary sub-graphs are connected by a propagator like in 3.15 or not like in 3.16.
    · In the first case, the total number of cycles will be at most equal to one cycle of each color, plus the number of cycles of a graph with $r - 2$ elementary sub-graphs, which is $n_l \prod_{i=1}^{k} n_i^{r/2}$. This could be seen as counting the cycles in the two connected elementary sub-graphs and then removing them from the graph $\mathrm{Tr}((\boldsymbol{M} - \mathbb{E}(\boldsymbol{M}))^r)$ and replacing them by an edge of color l to count the remaining cycles as illustrated in 3.15.
    · In the second case, we can remove the cycle of one edge of color l and mix the two elementary sub-graphs adjacent to it into one, as it will not

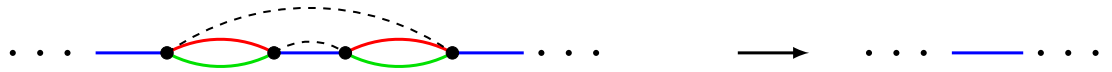


Figure 3.15: Elementary sub-graph completely connected to the elementary sub-graph next to it

Figure 3.16: Elementary sub-graph not completely connected to the elementary sub-graph next to it

change the number of remaining cycles. We thus have a contribution of n_l from the cycle associated to the the removed edge, and a contribution of $n_l \prod_{i=1}^{k} n_i^{(r-1)/2}$ from the remaining graph using then the induction hypothesis. Given that we are in the case of $n_l \leq \prod_{i \neq l} n_i$, the total contribution $n_l \times n_l \prod_{i=1}^{k} n_i^{(r-1)/2} \leq n_l \prod_{i=1}^{k} n_i^{r/2}$ which proves the statement.

- The covering graph with propagators disposed as in Figure 3.15 will have a contribution of exactly $n_l^r \prod_{i \neq l} n_i$ if r is even.

- Case $n_l \geq \prod_{i \neq l} n_i$:

  Let's prove by induction the statement: S_r: $\mathbb{E}[\mathrm{Tr}((\boldsymbol{M} - \mathbb{E}(\boldsymbol{M}))^r)] \sim n_l^r \prod_{i \neq l} n_i$.

  - Initialisation with $r = 2$: For $\boldsymbol{M} - \mathbb{E}(\boldsymbol{M})$ we have the covering graphs in Figure 3.14 which checks S_2

  - Assume S_r is true for all $r' < r$, let's prove for r that no covering graph has a contribution strictly superior to $n_l^r \prod_{i \neq l} n_i$ and then that we can provide a covering graph that contribute by exactly $n_l^r \prod_{i \neq l} n_i$.

    * In the case where there is no cycle of color l of length 1 (which means a propagator that shares the same ends that an edge of color l), then all edges have a contribution of $1/2$ at maximum. So the contribution of that covering graph will be at most equal to $\prod_{i=1}^{k} n_i^{r/2}$.

    * In case there is such a propagator, which means that there is a propagator coinciding with an edge of color l between two elementary sub-graphs. Then either the two other ends of the two elementary sub-graphs are connected by a propagator like in Figure 3.15 or not.

      · In the first case, the total number of cycles will be at most equal to one cycle of each color, plus the number of cycles of a graph with $r - 2$ elementary sub-graphs, so the contribution of the covering graph will be at most $n_l^{r-2} \prod_{i \neq l} n_i \times \prod_{i=1}^{k} n_i$, which is smaller than $n_l^r \prod_{i \neq l} n_i$.

- In the second case (the case of Figure 3.16), by induction the number of cycles will be at most the number of cycles in the remaining $r-1$ elementary sub-graphs plus a cycle of color blue. Thus, it is at most equal to $n_l^r \prod_{i\neq l} n_i$.

* The covering graph consisting in propagators coinciding with blue edges has a contribution exactly equal to $n_l^r \prod_{i\neq l} n_i$.

Thus, in the case $n_l \leq \prod_{i\neq l} n_i$, the pure signal matrix will have a larger operator norm if $\beta > (\prod_i n_i)^{\frac{1}{4}}$ and in the second case $\beta > n_l^{\frac{1}{2}}$ □

Theoretical guarantees for more general situations

This framework allows us to derive a new theoretical threshold $\max\left((\prod_{i=1}^k n_i)^{1/4}, n_i^{1/2}\right)$ for tensors with different dimensions, which are very common in real life applications.

3.3 Generalization to Tensor decomposition

3.3.1 Adaptation to low-rank CP decomposition

We consider a symmetric tensor with multiple orthogonal spikes.

$$\mathbf{T} = \sum_{i=1}^{p} \beta_i \boldsymbol{v}_i^{\otimes k} + \mathbf{Z} \quad \text{where} \quad \langle \boldsymbol{v}_i, \boldsymbol{v}_j \rangle = 0 \quad \forall i \neq j. \tag{3.22}$$

Theorem 11. *If we have a number of spikes p that is constant in respect to n, we can recover the p spikes by an alternating the use of melonic diagram, power iteration and deflation.*

We first use the tetrahedral diagram to obtain a vector correlated with the signal vectors, then we follow by power iteration to obtain a normalized vector $\boldsymbol{v}$ highly correlated to one of the spikes and then deflation which consists in replacing the tensor $\mathbf{T}$ by $\mathbf{T} - \alpha \boldsymbol{v}^{\otimes 3}$ where $\alpha = \mathbf{T}.\boldsymbol{v}^{\otimes 3}$. The experimental results suggest that it works not only for small values of p but also for a number of spikes up to n.

3.3.2 Adaptation to Tucker decomposition

We consider the decomposition of a tensor into a set of matrices (loadings) with orthogonal columns and a core tensor. Thus, we adapted in a simple way our framework to this other decomposition scheme. This highlights how generic and important this new framework is. We compared the principal methods of Tucker decomposition with a straightforward adaptation of these methods where we use the matrix associated to the tetrahedral instead of the melonic (tensor unfolding matrices) that are initially used in these methods (as loadings in the initialization as well as in the power iteration).

Algorithm 2: Recovery algorithm for for CP decomposition

Input: The tensor $\mathbf{T} = \sum_{l=1}^{p} \sqrt{n}\beta_l \boldsymbol{v}_l^{\otimes 3} + \mathbf{Z}$
Goal: Estimate $\{\boldsymbol{v}_l\}_{l=1}^{p}$ and $\{\beta_l\}_{l=1}^{p}$.
$\mathcal{E} = \emptyset$
$\mathbf{T}^0 = \mathbf{T}$
Compute the matrix $\boldsymbol{M}_{\mathcal{G},e}(\mathbf{T})$
Compute its k top eigenvectors and eigenvalues.
Output: Estimated vectors $\{\hat{\boldsymbol{v}}_l\}_{l=1}^{p} = \text{argmax}_{\mathcal{E}' \subset \mathcal{E}, |\mathcal{E}'|=p} \sum_{\boldsymbol{v} \in \mathcal{E}'} \mathbf{T}(\boldsymbol{v}, \boldsymbol{v}, \boldsymbol{v})$ and $\hat{\beta}_l \equiv \alpha_l / \sqrt{n}$

The case r=1 It is straightforward to see that when $r_1 = r_2 = r_3 = 1$, HOOI is exactly equivalent to Tensor Unfolding with power iteration. From a theoretical point of view, not only our algorithm also achieve the optimal estimation error rate but we also unveiled a new phase transition appearing in the asymmetric case when the tensor dimensions are of the form $n1 > n2 * n3$. This was not studied in [ZX18] as they considered only the cases where $\exists C_0$ an universal constant such that $n_1, n_2, n_3 \leq C_0 \min(n_1, n_2, n_3)$.

The case r>1 We adapted the framework to the Tucker decomposition case to compare to HOOI. Indeed one straightforward application of our new results is to replace the matricization $\mathcal{M}_k$ which correspond to the melonic graph (since SVD(**M**)= Eigenvalues(**MM**$^\top$) with a new matricization corresponding to the tetrahedral graph. $\Rightarrow$ We perform SVD on the tetrahedral matrices.

Algorithm 3: Recovery algorithm for for Tucker decomposition

Input: The tensor $\mathbf{T} = \sum_{l=1}^{p} \sqrt{n}\beta_l \boldsymbol{v}_l^{\otimes 3} + \mathbf{Z}$, m_{init}, m_{iter},ε,$\Lambda(\sim n)$
Goal: Estimate $\{\boldsymbol{v}_l\}_{l=1}^{p}$ and $\{\beta_l\}_{l=1}^{p}$.
for j=0 to 3 **do**
    Compute the matrix $\boldsymbol{M}_{\mathcal{G},e}(\mathbf{T})$ associated to an edge corresponding to the index i
end for
Use the previously computed matrices as an initialisation matrix for Tucker decomposition.
Output: Estimated vectors mode matrices and core tensor.

3.4 Numerical experiments

In this section we will investigate the empirical results of the previously mentioned applications in order to see if they match with our theoretical results. We restrict to the dimension $k = 3$ for simplicity. Simulations were run in Python on a Dell computer running Ubuntu 18.04.5 LTS with eight Intel Core i7-4800MQ processors at 2.70 GHz and 16GB of RAM.

3.4.1 Tensor PCA

Settings and comparison of methods for Tensor PCA

Each experiment is an independent instance consisting in the following steps:

- We generate randomly the n components of the signal vector $\boldsymbol{v}_0$ and then normalize it.
- We generate randomly the n^3 components of the random tensor $\mathbf{Z}$. If we are in the symmetric case, we symmetrize it with the same normalization than [RM14]):

$$Z_{ijk} = (Z_{ijk} + Z_{ikj} + Z_{jik} + Z_{jki} + Z_{kij} + Z_{kji})/\sqrt{12}$$

- We compute the tensor $\mathbf{T} = \mathbf{Z} + \sqrt{n}\beta \boldsymbol{v}_0^{\otimes k}$
- We compute our matrix associated to the invariant $\boldsymbol{M}$ constructed from $\mathbf{T}$ using the numpy method tensordot in Python.
- We compute its leading eigenvector $\boldsymbol{v}$
- We plot the correlation between the vector resulted from the algorithm $\boldsymbol{v}$ and the signal vector we aim to recover $\boldsymbol{v}_0$ (namely the scalar product $< \boldsymbol{v}, \boldsymbol{v}_0 >$).

We focus on the symmetric case and, as in [RM14], for every algorithm we use two variants: the simple algorithm outputting $\boldsymbol{v}$ and an algorithm where we apply 100 power iterations on $\boldsymbol{v}$: $v_i \leftarrow T_{ijk}v_jv_k$, distinguishable by a prefix "p-". In Figures 3.18, 3.19 and 3.20, we run 200 experiments for each value of β and plot the 95% confidence interval of the correlation of the vector recovered with the signal vector. We compare our method (tetrahedral) to other algorithmic methods: the melonic (tensor unfolding) [RM14] and the homotopy [ADGM17b]. To the best of our knowledge, they give the state of art respectively for the symmetric and asymmetric tensor [BCRT20]. Other methods exist but are either too computationally expensive (sum of squares) or are variants of these algorithms.

We see in Figure 3.17 that in the asymmetric case that the tetrahedral algorithm performs the best.

Spike with different dimensions:

We aim to recover the three vectors $\boldsymbol{v}_1, \boldsymbol{v}_2$ and $\boldsymbol{v}_3$ from a tensor $\mathbf{Z} + \beta \boldsymbol{v}_1 \otimes \boldsymbol{v}_2 \otimes \boldsymbol{v}_3$. We repeat 100 times each instance consisting in choosing randomly $\boldsymbol{v}_1, \boldsymbol{v}_2$ and $\boldsymbol{v}_3$ and the Gaussian random tensor $\mathbf{Z}$ and we plot the correlation of the signal vectors with the vectors recovered using the tetrahedral.

In detail, each experiment is an independent instance consisting in the following steps:

- We generate randomly the n_1, n_2 and n_3 components of respectively the signal vectors $\boldsymbol{v}_1$, $\boldsymbol{v}_2$ and $\boldsymbol{v}_3$ and then normalize them.
- We generate randomly the $n_1 \times n_2 \times n_3$ components of the random tensor $\mathbf{Z}$.
- We compute the tensor $\mathbf{T} = \mathbf{Z} + \beta \boldsymbol{v}_1 \otimes \boldsymbol{v}_2 \otimes \boldsymbol{v}_3$

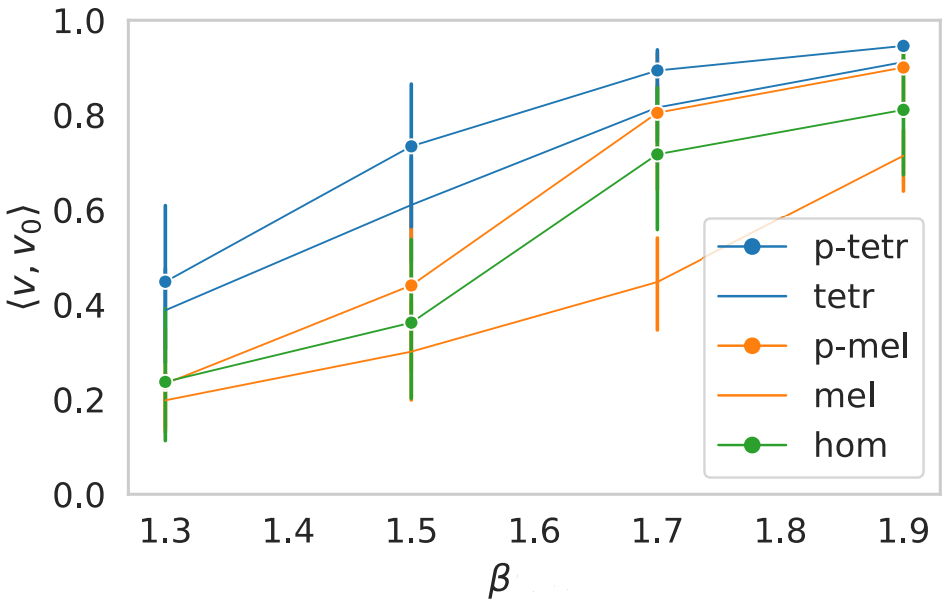


Figure 3.17: Comparison of different methods for asymmetric tensor for $n = 100$

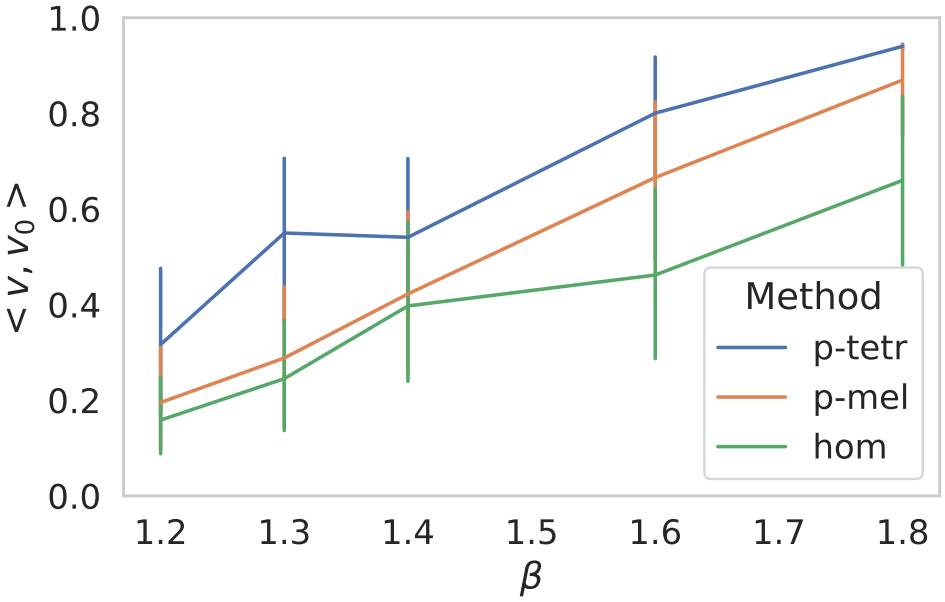


Figure 3.18: Comparison of different methods for symmetric recovery for $n = 100$.

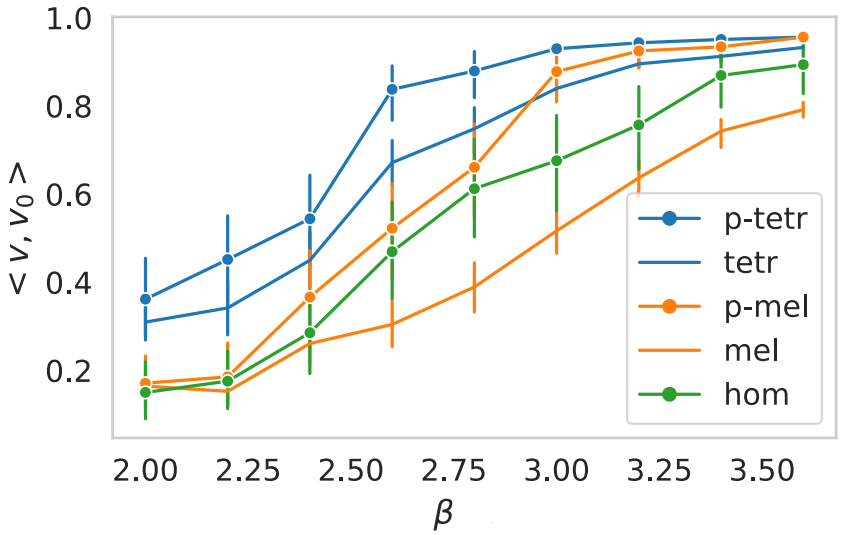


Figure 3.19: Comparison of different methods for symmetric recovery for $n = 150$.

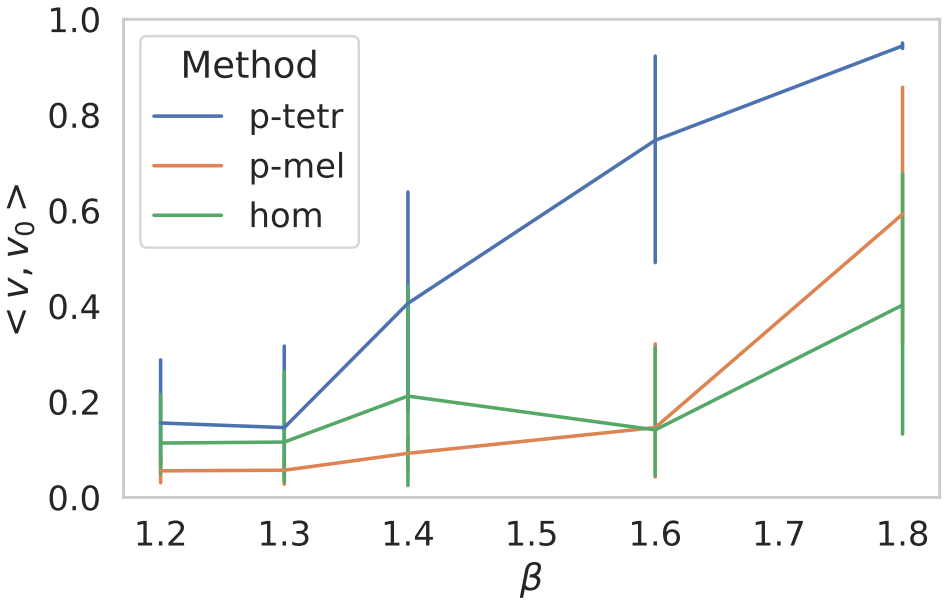


Figure 3.20: Comparison of different methods for symmetric recovery for $n = 200$.

- We compute the three matrices to recover each of the vectors by opening an edge of the color corresponding to the position of the vector using the numpy method tensordot in Python.

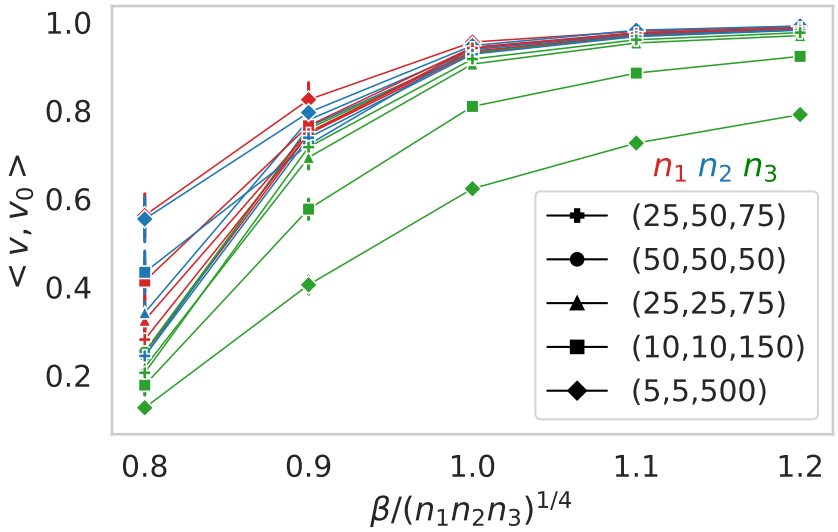


Figure 3.21: Recovery of a spike with different dimensions.

- We compute their leading eigenvectors.
- We plot the correlation between the vectors resulted from the algorithm and their corresponding signal vectors we aim to recover $\boldsymbol{v}_1$, $\boldsymbol{v}_2$ and $\boldsymbol{v}_3$ (namely scalar product $< \boldsymbol{v}, \boldsymbol{v}_i >$).

We see in the Figure 3.21 that the threshold $(n_1 n_2 n_3)^{1/4}$ for the three vectors matches perfectly with the experiences when $n_3 < n_1.n_2$. We also see that when $n_3 > n_1.n_2$ the recovery of v_3 (in green and with the diamond and square markers) have a different asymptotic behavior than v_1 and v_2 (it becomes $n_3^{1/2}$ since $n_3^{1/2} \geq (n_1 n_2 n_3)^{1/4}$), corresponding to what our theoretical study predicted.

We see in the Figure 3.22 that when $n_3 > n_1 n_2$, the asymptotic behavior of the recovery threshold of v_3 matches with our theoretical result $n_3^{1/2}$

The number of iterations of power iteration for each method In the symmetric case, we plot in Figure 3.23 the number of iterations needed for the power iteration method to converge with an initialization given by the result of one of the methods we compare: tetrahedral, unfolding, homotopy. In addition to these methods, we plot in red (that we call perf) the correlation of the power iteration starting from the signal vector $\boldsymbol{v}_0$.

Empirical results shows the fast convergence using the power iteration method following the tetrahedral method.

3.4.2 Memory and time requirements of the methods

General requirements:

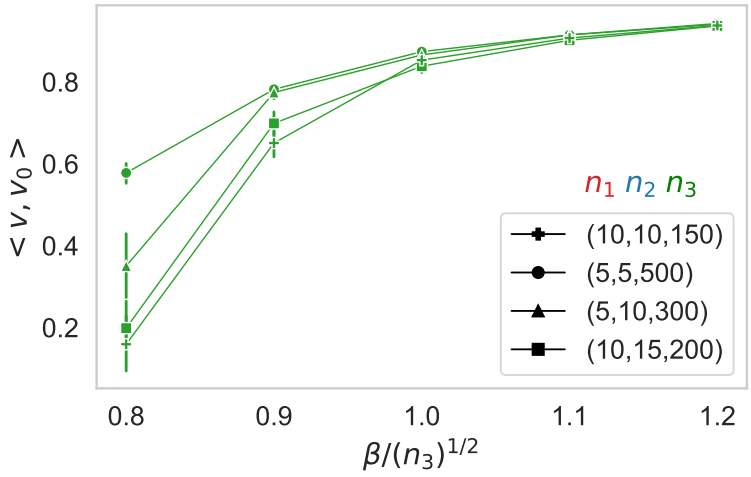


Figure 3.22: Recovery of $\boldsymbol{v}_3$ where $n_3 > n_1 n_2$

Since our method consists on computing algebraic operations on tensors entries to store in matrix $n \times n$, the naive recovery method for a tensor of order k and for a tensor invariant of degree d would require $O(n^2)$ space requirements and $O(n^{kd})$ time complexity.

However, similarly to the matrix case where naive implementation requires n^3 but different methods propose better asymptotic bounds on the time required to multiply matrices, it is possible to simplify the computations by using decomposition on blocks ([MKV+13]). There has been recently encouraging research investigating the optimization of tensor contractions ([KSRH+18], [Mat18],[HMvdG18]) in the same way it was done for matrix calculations, especially with the rise of the use of tensorial objects. The ability to be parallelized (each component of the matrix $\boldsymbol{M}_{\mathcal{G},e}$ can be computed independently) is also an important feature of our methods for practical applications.

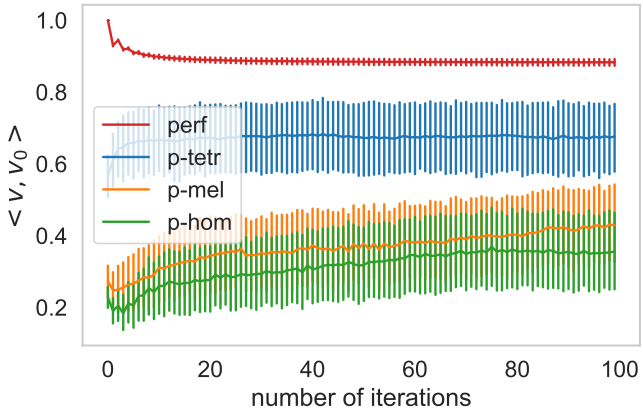
Tetrahedral method with $\tilde{O}(n^4)$ time and $\tilde{O}(n^3)$ space requirement:

The unfolding algorithm (melonic) was proven to have a nearly linear time ($\tilde{O}(n^3)$, since the input tensor is of size n^3, $\tilde{O}$ notation neglects logarithmic factors).

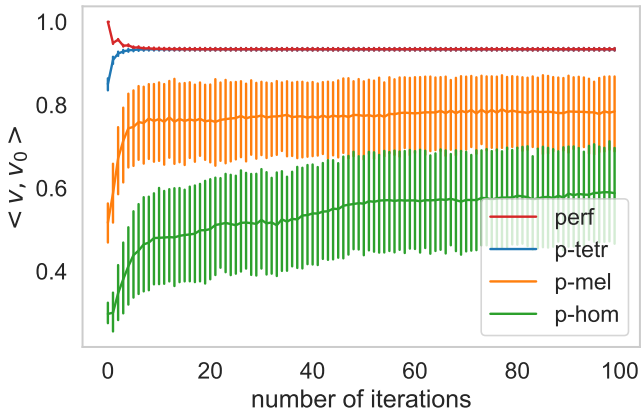
We use a similar approach to provide a recovery algorithm for the tetrahedral with $\tilde{O}(n^4)$ time and $\tilde{O}(n^3)$ space requirement.

Indeed, using the decomposition previously described, we can prove that there is an gap between the highest eigenvalue $\lambda_1 \sim \beta^4$ and the second largest one λ_2. Thus, we can use matrix power method in order to recover the leading eigenvector $\boldsymbol{v} \leftarrow \boldsymbol{M}^\top \boldsymbol{v}$ with $\log(n)$ complexity (the convergence of the matrix power method is geometric with ratio $|\lambda_1/\lambda_2|$).

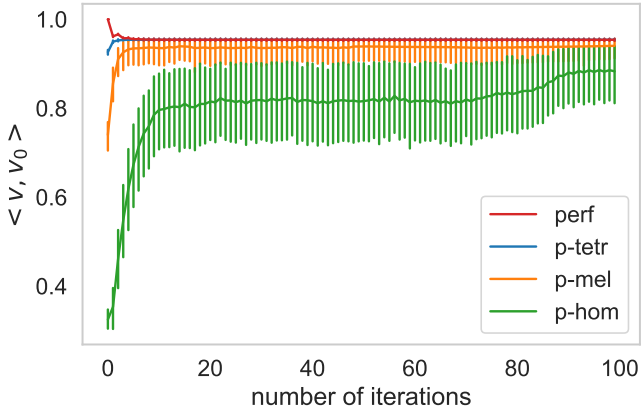
This lets us calculate the leading eigenvector of the matrix associated to the tetrahedral: $M^t_{i_1 i_2} \equiv T_{i_1 jk} T_{i_2 j'k'} T_{ij'k} T_{ijk'}$.



(a) $\beta = 25$



(b) $\beta = 30$



(c) $\beta = 35$

Figure 3.23: Correlation with the signal vector in function of the number of iterations of the power iteration for $n = 100$. In red an initialization with $\boldsymbol{v}_0$ for the comparison. In blue, orange and green an initialization with the output of respectively the tetrahedral, the unfolding and the homotopy algorithms.

We can compute find the leading eigenvector in a time of $\tilde{O}(n^4)$ and space $\tilde{O}(n^3)$. Indeed each iteration of the matrix power iteration method requires two intermediate matrices we denote $\boldsymbol{A}, \boldsymbol{B}$ and a intermediate $3-$dimensional tensor we denote $\mathbf{C}$, in every step there is at most n^4 operations. we follow these steps:

- We take an initial vector $\boldsymbol{v}$
- We compute the matrix $\boldsymbol{A}$ defined as $A_{j'k'} \equiv T_{i_2 j' k'} v_{i_2}$
- We compute the tensor $\mathbf{C}$ defined as $C_{k'ik} \equiv A_{j'k'} T_{ij'k}$
- We compute the matrix $\boldsymbol{B}$ defined as as $B_{kj} \equiv C_{k'ik} T_{ijk'}$
- Return the vector $\boldsymbol{v}'$ defined as $v'_{i_1} \equiv B_{kj} T_{i_1 jk}$

Comparison between different methods

We use our results and the results reported in [ADGM17b] to fill the Table 3.1. $\tilde{O}$ and $\tilde{\Omega}$ notation ignores logarithmic factors.

Thus, our framework could be seen as providing multiple algorithms from which we could choose from depending in how much we prioritize the performance over the speed.

Table 3.1: Algorithmic threshold, time and space requirements for each method

Method	Time	Space	Threshold
Melonic recovery/ Unfolding	$O(n^3)$	$O(n^2)$	$\Omega(n^{3/4})$
Tetrahedral recovery	$O(n^4)$	$O(n^3)$	$\Omega(n^{3/4})$
Sum of squares	$> \Omega(n^6)$	$> \Omega(n^6)$	$\tilde{\Omega}(n^{3/4})$
Homotopy	$O(n^3)$	$O(n)$	$\Omega(n^{3/4})$
Information-theoretic	Exp	$O(n)$	$\tilde{\Omega}(n^{1/2})$

3.4.3 CP and Tucker decomposition on synthetic and real data

In Figure 3.24 and 3.25, we compare the proposed tensor decomposition algorithms that we derived from our framework with the state of the art for both CP and Tucker decompositions. The comparisons are done for $n = 100$ and for different β over 20 independent runs.

The figure 3.24 concerns the CP decomposition and suggests that the proposed Tetrahedral (Tetra) method is more robust to noise than the commonly used algorithm of the TensorLy package (based on ALS), as well as the power iteration method. For Tucker decomposition in Figure 3.25, the results show that the proposed method (Tetra HOOI) provides better results than HOOI and HOSVD in the symmetric case with $r_1 = r_2 = r_3 = 20$ in the large noise regime. Furthermore, we carried out experiments on structured real data, the Yale Face Database B [LHK05]. In more details, we considered a set of stacked face images that form this database as an initial tensor to which we added Gaussian noise. First, we compare HOOI and the proposed Tetra-HOOI algorithms for a fixed value of the

rank of the matrices involved in this type of methods $(r_1, r_2, r_3) = (10, 10, 10)$ and for different values of the noise intensity (λ). To evaluate the denoising performance of those methods, we compute the average and standard deviation (over 5 runs) of the Frobenius norm $\frac{\|\mathbf{X}-\mathbf{V}\|}{\|\mathbf{V}\|}$ where $\mathbf{V}$ is the input tensor and $\mathbf{X}$ is the output of the algorithm. The results which are reported in the table 3.4 show that Tetra HOOI again outperforms HOOI even on real data. Note that, we obtained similar results for a fixed ($\lambda = 1000$) and for different values of (r_1, r_2, r_3). Note that the Frobenius norm of the difference between two completely uncorrelated normalized vectors is equal to 2.

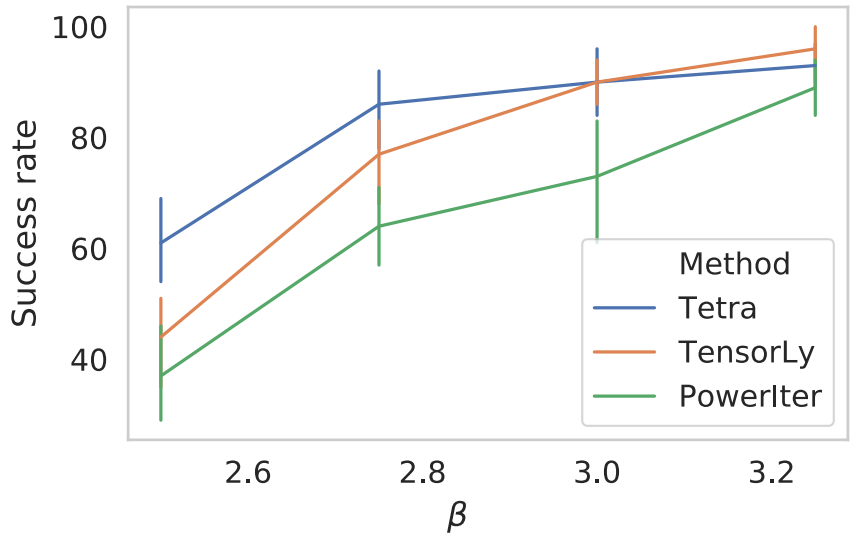


Figure 3.24: Comparison CP decomposition methods for $n = 100$ and $n_{\text{spikes}} = 20$.

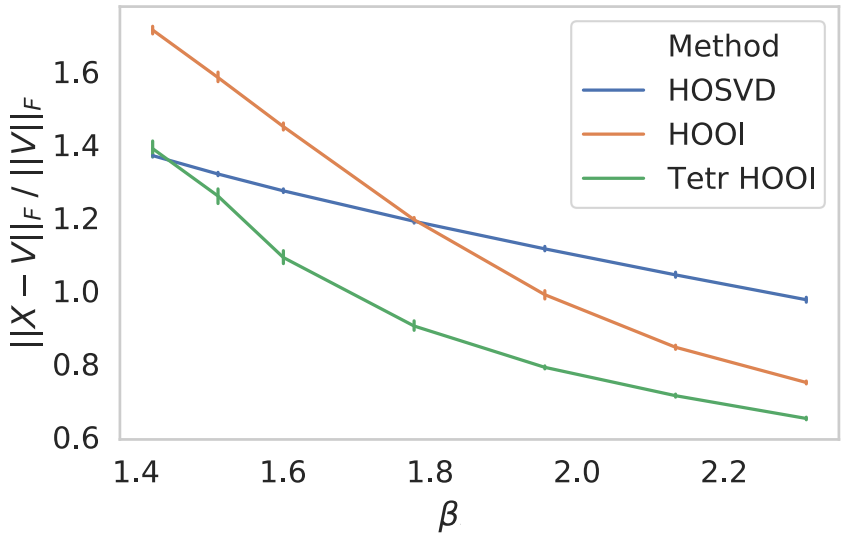


Figure 3.25: Comparison Tucker decomposition methods for $n = 100$ and $r_1 = r_2 = r_3 = 20$

The settings

CP decomposition on synthetic data

We choose to take the β_i equal since we expect it to be the hardest case (it is more difficult to distinguish the spikes). Each experiment is an independent instance consisting in the following steps:

- We generate randomly the p orthogonal vectors $\boldsymbol{v}_i$ and then normalize them.
  - Generate the components of $\boldsymbol{v}_i$ randomly.
  - $\forall j \in [i]$ we compute $\boldsymbol{v}_i \leftarrow \boldsymbol{v}_i - \langle \boldsymbol{v}_i, \boldsymbol{v}_j \rangle \boldsymbol{v}_j$ in order to have orthogonal spikes.
  - Normalize $\boldsymbol{v}_i$
- We generate randomly the n^3 components of the random tensor $\mathbf{Z}$.
- We compute the tensor $\mathbf{T} = \mathbf{Z} + \sum_i \beta_i \boldsymbol{v}_i^{\otimes 3}$
- We compute the matrix associated to the tetrahedral with the numpy method tensordot in Python.
- We compute its eigenvectors $\boldsymbol{x}_i$.
- We iterate the following steps beginning from $i = 0$ to $i = p$:
  - We use the power iteration method on $\boldsymbol{x}_i$ with 100 iterations.
  - We update the tensor $\mathbf{T} \leftarrow \mathbf{T} - \alpha_i \boldsymbol{x}_i^{\otimes 3}$ where $\alpha_i = T_{jkl}(\boldsymbol{x}_i)_j(\boldsymbol{x}_i)_k(\boldsymbol{x}_i)_l$

- Since the vectors $\boldsymbol{x}_i$ are not necessarily ordered, we want to make a one for one correspondence with the vectors $\boldsymbol{v}_i$. So for $\boldsymbol{v}_1$, we search for the $\boldsymbol{x}_j$ that maximizes $\langle \boldsymbol{x}_j, \boldsymbol{v}_i \rangle$. We call rename it $\boldsymbol{w}_i$. Then for the other $\boldsymbol{v}_i$ we search for the $\boldsymbol{x}_j$ that maximizes $\langle \boldsymbol{x}_j, \boldsymbol{v}_i \rangle$ in the set $X = \{\boldsymbol{x}_j,\ j \in [p]\} \setminus \{\boldsymbol{w}_k,\ k \in [i-1]\}$. In that way, for each $\boldsymbol{v}_i$ there is a one to one correspondence to a vector $\boldsymbol{x}_j$ that we rename $\boldsymbol{w}_i$.
- We plot the correlation between the vectors resulted from the algorithm w_i and the signal vectors we aim to recover $\boldsymbol{v}_i$ (namely scalar product $\langle \boldsymbol{w}_i, \boldsymbol{v}_i \rangle$) for each i.

We see that empirically, our method is able to recover any number of spikes inferior to n, and with a similar threshold than with a single spike.

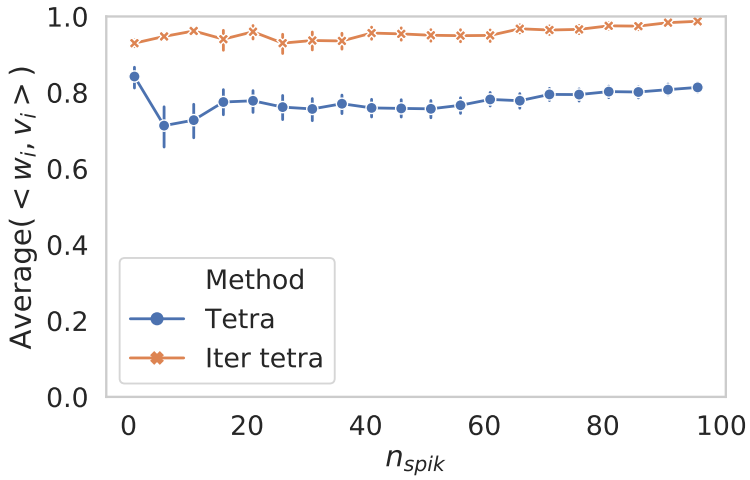


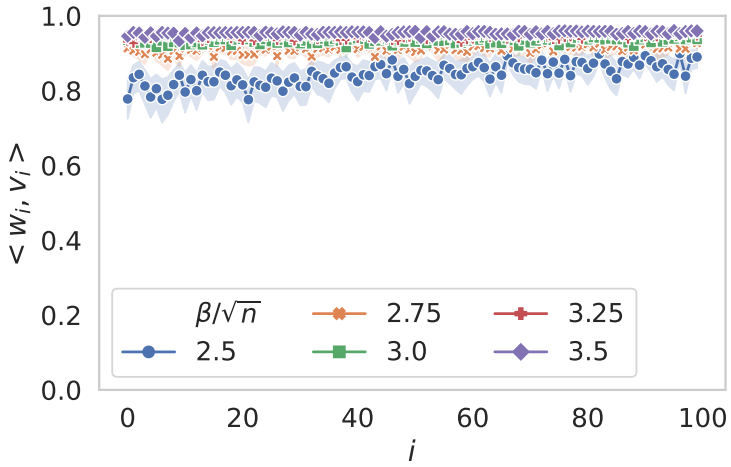
Figure 3.26: Average of the correlation of the recovered vector with their corresponding signal vector in function of the number of spikes for $n = 100$ and $\beta = 30$

We see in Figure 3.27 that when the spacing between the β_i gets bigger, it is easier to retrieve the signal vectors, since it will be easier to distinguish them. Thus, the case where all β_i are equal seems to be the hardest case.

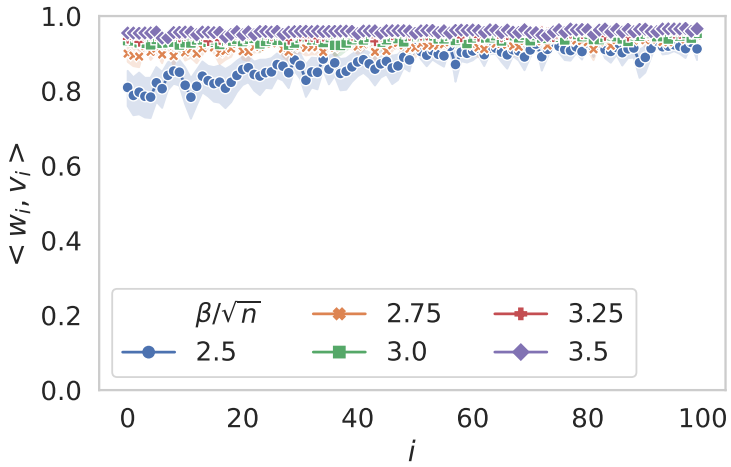
CP decomposition on real data

We apply our CP decomposition on a real application that consists in the Hyperspectral images (HSI). As explained in [Nas13], "Typically, a hyperspectral spectrometer provides hundreds of narrow contiguous bands over a wide range of the electromagnetic spectrum. Hyperspectral sensors measure the reflective (or emissive) properties of objects in the visible and short-wave infrared (IR) regions (or the mid-wave and long-wave IR regions) of the spectrum. Processing of these data allows algorithms to detect and identify targets of interest in a hyperspectral scene by exploiting the spectral signatures of the materials".

Denoising is an important preprocessing step to further analyze a hyperspectral image



(a) Recovery many spikes with $\beta_i = \beta + 0.02i$



(b) Recovery many spikes with $\beta_i = \beta + 0.05i$

Figure 3.27: Recovery of many spikes with non equal β_i

(HSI). The common denoising procedures are either based on 2-dimensional (2D) methods (mainly 2D filters) or tensor decomposition methods.

In [LBF12], the authors compared CP decomposition method based on the Alternating Least Square (ALS) algorithm with existent methods (two-dimensional filter and Tucker3 that is based on a Tucker decomposition) to denoise HSI. They performed their experiments on a real world data: the Airborne Visible/Infrared Imaging Spectrometer (AVIRIS) HSI, an airborne hyperspectral system flown by NASA/Jet Propulsion Laboratory (JPL). To analyze quantitatively the denoising results, they compare the signal-to-noise ratio (SNR) of the denoised image that is defined as:

$$\text{SNR}_{\text{out}} = 10 \log_{10} \frac{||\hat{\mathcal{Y}}||^2}{||\mathcal{Y} - \hat{\mathcal{Y}}||^2} \tag{3.23}$$

where $\mathcal{Y}$ is the original image and $\hat{\mathcal{Y}}$ the estimated image after denoising.

Their numerical results show that the CP decomposition model using the ALS algorithm performs better than other considered methods as a denoising procedure.

In order to judge the performance of our algorithm, we perform the same experiment with the same estimator and compare it with the ALS algorithm using the Python TensorLy package [KPAP16]. The hyperspectral image we use is the open source data given in [MGT+18] that we normalize. It consists of a tensor of size $\mathbf{R}^{425\times 861\times 475}$ where 425 is the number of spectral bands and 865×475 is the spatial resolution.

CP decomposition model decomposes a tensor as a sum of rank-one tensors (that we call spikes). Thus, we compare the SNR for different number of spikes in Table 3.2

n_{spikes}	1	3	5	10
TensorLy	41.88	42.75	43.58	46.81
Melonic	41.88	43.20	43.99	46.2
Tetra	41.88	**43.22**	**44.33**	**47.28**

Table 3.2: We compare the tetrahedral algorithm with the melonic algorithm and the ALS algorithm from TensorLy.

Tucker decomposition on real data To evaluate the denoising performance of the different Tucker decomposition methods, we compute the average and standard deviation (over 5 runs) of the Frobenius norm. The results which are reported in the table 3.3 for fixed rank and table 3.4 for β fixed show that Tetra HOOI again outperforms HOOI even on real data.

(r_1, r_2, r_3)	$(5, 7, 10)$	$(10, 20, 30)$	$(15, 15, 15)$	$(20, 20, 20)$
HOOI	0.673 ± 0.010	1.143 ± 0.003	0.977 ± 0.005	1.183 ± 0.001
Tetra-HOOI	0.621 ± 0.009	0.713 ± 0.020	0.666 ± 0.017	0.965 ± 0.009

Table 3.3: We compare HOOI and the proposed Tetra-HOOI algorithms for a fixed value of the noise intensity ($\lambda = 1000$). We compute the average and standard deviation (over 5 runs) of the Frobenius norm for different values of the rank of the matrices involved in this type of methods (r_1, r_2, r_3).

β	900	1100	1300	1500
HOOI	0.696 ± 0.008	0.845 ± 0.006	0.993 ± 0.008	1.125 ± 0.007
Tetra-HOOI	0.610 ± 0.017	0.648 ± 0.010	0.707 ± 0.019	0.754 ± 0.013

Table 3.4: We compare HOOI and the proposed Tetra-HOOI algorithms for a fixed value of the rank of the matrices involved in this type of methods $(p_1, p_2, p_3) = (10, 10, 10)$. We compute the average and standard deviation (over 5 runs) of the Frobenius norm for different values of the noise intensity (β).

Chapter 4

A new algorithm : Selective Multiple Power Iteration (SMPI)

4.1 Power iteration based algorithms

Power iteration is a simple method that has been extensively used in multiple tensor problems [AGH+12, AGHK13]. [RM14] investigated the empirical performance of power iteration with a random initialization in the range of $n \in [50, 800]$ and observed an empirical threshold of $n^{1/2}$. The conclusion to which they arrive when using a naive approach

is the following.

$$\boldsymbol{v}^1 = \frac{\beta\tau_0^{k-1}v_0 + n^{-1/2}g}{\sqrt{\beta^2\tau_0^{2(k-1)} + 1}} + o(1) \tag{4.1}$$

where $g \sim \mathrm{N}(0, \mathbf{I}_N)$ and $o(1)$ is a vector whose norm converges to zero in probability as $n \to \infty$. However this is not rigorous and may overlook some subtle mechanism.

Through an improved noise analysis, [WA16] showed that for a symmetrical tensor, power iteration is indeed able to recover the signal for a SNR β above $n^{1/2}$ with a constant number of initialization and a number of iterations logarithmic on n. Their experiments in the range of $n \in [25, \ldots, 250]$ suggested that this threshold is tight. A recent paper [HHYC20] investigates the simple power iteration for a non-symmetric tensor for Tensor PCA, they prove that the algorithmic threshold is strictly equal to $n^{1/2}$ as an asymetrical power iteration with a random initialization outputs a random vector below this threshold. The results of their experiments for $n \in [200, \ldots, 800]$ match their theoretical results.

The algorithm of [BCRT20] consists in choosing one initial point x_0 that they consider as a center of mass and that will be updated at each step. Then, they sample R points that are orthogonal to x_0. After evaluating the gradient on each of the R points, they average them and use the obtained gradient in order to update the position of the centre of mass x_0. Moreover they have a stopping condition given by $\|x(t) - x(t-1)\|_2 < \varepsilon$. Also, they require a "rate η small enough so that the discrete updates in the algorithm can be considered a good approximation of a continuous time algorithm." The idea of [BCRT20] is that "One can then substantially reduce the noise by evaluating an empirical average of the gradient obtained as a sum over many random independent positions in the space of parameters to be optimized". Averaging these gradients obtained from independent points will lower the norm of g_N compared to v_0 as they state: "In fact, the average over the replicas leads to a relative amplification of the informative contribution produced by the signal with respect to the noise. " However their algorithm has a theoretical limit as "the smoothing of the landscape using different replicas becomes ineffective when $R > R_{opt} \sim N(k-1)/2$ " which correspond to the threshold limit of $n^{1/4}$. Under that threshold g_S will be too small and could not be boosted enough so that it becomes stronger than g_N.

In this section, we aim to draw attention to a surprising observation that contrasts with previous work: if we impose five essential features for an algorithm based on power iteration or gradient descent (use a symmetrized power iteration, impose a polynomial number of initializations and iterations, etc.), we observe that a novel powerful mechanism for the convergence towards the signal takes place, leading to a fundamentally different performance. In fact, for $n \in [50, 1000]$, SMPI is the first algorithm to exhibit an empirical threshold corresponding to $O(1)$ and whose results matches the theoretically-optimal correlation at large n.

Link between Power Iteration and Gradient descent

Maximising $\mathbf{T}(\boldsymbol{v}, \boldsymbol{v}, \boldsymbol{v})$ is equivalent to finding the minimum of the cost function defined as $H(\boldsymbol{v}) \equiv -\mathbf{T}(\boldsymbol{v}, \boldsymbol{v}, \boldsymbol{v})$ [BCRT20]. Given that we restrict ourselves to the unit sphere $S^1 = \{\boldsymbol{v} \in \mathbb{R}^n \mid \boldsymbol{v}/\|\boldsymbol{v}\| = 1\}$, the gradient is equal to $\boldsymbol{g} \equiv \nabla H(\boldsymbol{v}) - (\nabla H(\boldsymbol{v}).\boldsymbol{v}).\boldsymbol{v}$ which

will be for our model $\boldsymbol{g} = -\mathbf{T}(:, \boldsymbol{v}, \boldsymbol{v}) + \mathbf{T}(\boldsymbol{v}, \boldsymbol{v}, \boldsymbol{v}).\boldsymbol{v}$. Therefore, power iteration could be written as:

$$\begin{aligned} \boldsymbol{v} \leftarrow \frac{\mathbf{T}(:, \boldsymbol{v}, \boldsymbol{v})}{\|\mathbf{T}(:, \boldsymbol{v}, \boldsymbol{v})\|} &= \frac{-\boldsymbol{g} + \mathbf{T}(\boldsymbol{v}, \boldsymbol{v}, \boldsymbol{v}).\boldsymbol{v}}{\|\mathbf{T}(:, \boldsymbol{v}, \boldsymbol{v})\|} \\ &= \frac{\mathbf{T}(\boldsymbol{v}, \boldsymbol{v}, \boldsymbol{v})}{\|\mathbf{T}(:, \boldsymbol{v}, \boldsymbol{v})\|}.(\boldsymbol{v} - \frac{\boldsymbol{g}}{\mathbf{T}(\boldsymbol{v}, \boldsymbol{v}, \boldsymbol{v})}) \end{aligned} \tag{4.2}$$

We can see that the power iteration could be seen as a gradient descent with an adaptive step size equal to $1/(\mathbf{T}(\boldsymbol{v}, \boldsymbol{v}, \boldsymbol{v}))$. This step size has the convenient particularity that it is large for a random $\boldsymbol{v}$ but becomes small for vectors $\boldsymbol{v}$ such that $\mathbf{T}(\boldsymbol{v}, \boldsymbol{v}, \boldsymbol{v})$ is large, for example when we are close to a minimum of H.

4.2 General Principle of SMPI

Table 4.1: The five essential features of SMPI compared to previous works investigating Power Iteration

Paper	Symmetry	Discreet step size	Poly. nb of initialisat.	Poly. nb of iterations	No stopping condition
Wang et al., 2017 [WDDFS17]	Yes	Yes	No	No	No
Huang et al., 2020 [HHYC20]	No	Yes	Yes	Yes	Yes
Ben Arous et al., 2020 [BAGJ+20]	Yes	No	Yes	Yes	Yes
Dudeja et al., 2022 [DH22]	Yes	Yes	Yes	No	Yes
SMPI, 2021	Yes	Yes	Yes	Yes	Yes

The proposed SMPI (Algorithm 1) consists in applying, in parallel, the power iteration method with m_{iter} iterations to m_{init} different random initialization. Then, SMPI uses the maximum likelihood estimator to select the output vector in this subset by choosing the vector that maximizes $\mathbf{T}(\boldsymbol{v}, \boldsymbol{v}, \boldsymbol{v})$.

4.2.1 The essential features of SMPI

We stress here an important and fundamental difference between our algorithm with previous algorithms based on power iteration. In order for this method to succeed in the low SNR setting, we need these five features that, for the best of our knowledge, we are the first to impose:

1. Using power iteration or a gradient descent with a large enough step size.
2. In the case of power iteration and non-symmetric tensor, using the symmetrized version (or equivalently, symmetrize the tensor).

Illustration of the principle of Selective Multiple Power Iteration

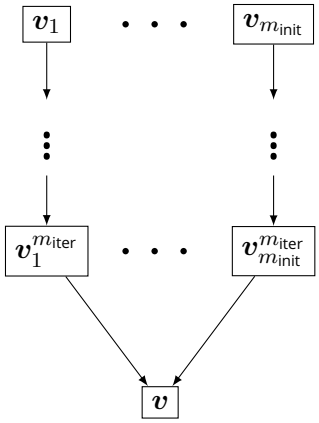
Input

Tensor $\mathbf{T} = \mathbf{Z} + \beta \boldsymbol{v}_0^{\otimes k}$

First step: Generate m_{init} random vectors.

Second step: Iterate m_{iter} times : $\boldsymbol{v}_i^{j+1} = \dfrac{\mathbf{T}\boldsymbol{v}_i^j \boldsymbol{v}_i^j}{\left\| \mathbf{T}\boldsymbol{v}_i^j \boldsymbol{v}_i^j \right\|}$

Third step: Choose the vector $\boldsymbol{v} = \arg\max_{1 \leq i \leq m_{\text{init}}} \left(\mathbf{T}.\boldsymbol{v}_i^{m_{\text{iter}}}.\boldsymbol{v}_i^{m_{\text{iter}}}.\boldsymbol{v}_i^{m_{\text{iter}}} \right)$:



Output

Figure 4.1: Illustrative figure for the SMPI algorithm

Algorithm 4: Selective Multiple Power Iteration

Input: The tensor $\mathbf{T} = \mathbf{Z} + \beta \boldsymbol{v}_0^{\otimes k}$, $m_{\text{init}} > 10n$, $m_{\text{iter}} > 10n$,Λ
Goal: Estimate $\boldsymbol{v}_0$.
for i=0 to m_{init} **do**
  Generate a random vector $\boldsymbol{v}_{i,0}$
  for j=0 to m_{iter} **do**
    $\boldsymbol{v}_{i,j+1} = \dfrac{\mathbf{T}(:, \boldsymbol{v}_{i,j}, \boldsymbol{v}_{i,j})}{\|\mathbf{T}(:, \boldsymbol{v}_{i,j}, \boldsymbol{v}_{i,j})\|}$
    if $j > \Lambda$ and $|\langle \boldsymbol{v}_{i,j-\Lambda}, \boldsymbol{v}_{i,j} \rangle| \geq 1 - \varepsilon$ **then**
      $\boldsymbol{v}_{i,m_{\text{iter}}} = \boldsymbol{v}_{i,j}$
      break
    end if
  end for
end for
Select the vector $\boldsymbol{v} = \arg\max_{1 \leq i \leq m_{\text{init}}} \mathbf{T}(\boldsymbol{v}_{i,m_{\text{iter}}}, \boldsymbol{v}_{i,m_{\text{iter}}}, \boldsymbol{v}_{i,m_{\text{iter}}})$
Output: the estimated vector $\boldsymbol{v}$

3. Prohibiting a stopping criteria on two consecutive iterations (such as $1 - |\langle \boldsymbol{v}_{i-1}, \boldsymbol{v}_i \rangle| < \varepsilon$ for $\varepsilon > 0$) and instead, use a criteria based on non-consecutive iterations distant by Λ : $1 - \langle \boldsymbol{v}_{i-\Lambda}, \boldsymbol{v}_i \rangle < \varepsilon$ and $\Lambda = O(n)$.

4. Using at least a polynomial number of iterations.

5. Using at least a polynomial number of initialization.

4.2.2 Generalization to Tensor decomposition

SMPI is simple, parallelizable and easy to generalize. In fact, we proposed two variants of SMPI to deal with the recovery of a spike with different dimensions and with the multiple spikes recovery (related to CP tensor decomposition problem). These proposed algorithms outperform existent methods in this context which shows a huge potential impact of SMPI for practical applications.

Algorithm for spike with different dimensions

In order to make these tools versatile, less restrictive and usable for a majority of applications, we have to consider the case where the dimensions are different: $n_1 \neq n_2 \neq n_3$.. For example in a video, there is no reason to impose that the time dimension is equal to the two spatial dimensions. Thus the problem is to infer a spike in the form of $\boldsymbol{v}_1 \otimes \boldsymbol{v}_2 \otimes \boldsymbol{v}_3$ from the following tensor

$$\mathbf{T} = \beta \left(\frac{n_1 + n_2 + n_3}{3} \right)^{1/2} \boldsymbol{v}_1 \otimes \boldsymbol{v}_2 \otimes \boldsymbol{v}_3 + \mathbf{Z}, \tag{4.3}$$

where $\mathbf{Z} \in \mathbb{R}^{n_1 \otimes n_2 \otimes n_3}$ is a tensor with random gaussian entries. Algorithm 5 is an adaptation of SMPI to tackle this model.

Algorithm 5: Recovery algorithm for a spike with different dimensions

1: **Input:** The tensor $\mathbf{T} = \beta \left(\frac{n_1+n_2+n_3}{3} \right)^{1/2} \boldsymbol{v}_a \otimes \boldsymbol{v}_b \otimes \boldsymbol{v}_c + \mathbf{Z}$, m_{init}, m_{iter}
2: **Goal:** Estimate $\boldsymbol{v}_a, \boldsymbol{v}_b, \boldsymbol{v}_c$.
3: Generate m_{init} random vectors $\{\boldsymbol{v}_i\}_{1 \leq i \leq n_1}$ and initialize $\boldsymbol{v}_{a_i}^0 = \boldsymbol{v}_{b_i}^0 = \boldsymbol{v}_{c_i}^0 = \boldsymbol{v}_i$;
4: Perform m_{iter} times power iteration:

$$\begin{aligned}
\boldsymbol{v}_{a_i}^{j+1} &\leftarrow \mathbf{T}(:, \boldsymbol{v}_{b_i}^{j}, \boldsymbol{v}_{c_i}^{j}) / \left\| \mathbf{T}(:, \boldsymbol{v}_{b_i}^{j}, \boldsymbol{v}_{c_i}^{j}) \right\| \\
\boldsymbol{v}_{b_i}^{j+1} &\leftarrow \mathbf{T}(\boldsymbol{v}_{a_i}^{j+1}, :, \boldsymbol{v}_{c_i}^{j}) / \left\| \mathbf{T}(\boldsymbol{v}_{a_i}^{j+1}, :, \boldsymbol{v}_{c_i}^{j}) \right\| \\
\boldsymbol{v}_{c_i}^{j+1} &\leftarrow \mathbf{T}(\boldsymbol{v}_{a_i}^{j+1}, \boldsymbol{v}_{b_i}^{j+1}, :) / \left\| \mathbf{T}(\boldsymbol{v}_{a_i}^{j+1}, \boldsymbol{v}_{b_i}^{j+1}, :) \right\|
\end{aligned} \tag{4.4}$$

5: Select the vectors $(\boldsymbol{v}_{af}, \boldsymbol{v}_{bf}, \boldsymbol{v}_{cf}) = \arg\max_{1 \leq i \leq m_{\text{init}}} \mathbf{T}(\boldsymbol{v}_{a_i}^{m_{\text{iter}}}, \boldsymbol{v}_{b_i}^{m_{\text{iter}}}, \boldsymbol{v}_{c_i}^{m_{\text{iter}}})$:
6: **Output:** Obtaining estimated vectors $(\boldsymbol{v}_{af}, \boldsymbol{v}_{bf}, \boldsymbol{v}_{cf})$

Low-rank CP decomposition algorithm

We consider a generalization of the tensor PCA where we consider the problem of estimating multiple signal vectors. In this case, we can write the symmetric tensor with multiple spikes as:

$$\mathbf{T} = \sum_{l=1}^{p} \sqrt{n} \beta_l \boldsymbol{v}_l^{\otimes 3} + \mathbf{Z} \tag{4.5}$$

The algorithm 6 is a simple variant of SMPI. Although it shares some similarities with existing methods based on power iteration such as [WA16], it is fundamentally different in its operating mode. Indeed, it is to the best of our knowledge, the first algorithm that is able to take advantage of the noise-based convergence mechanism given that it shares the same essential features with SMPI. This difference is illustrated by the substantial improvement over existing algorithms on our numerical experiments that we present in the next section.

Algorithm 6: Recovery algorithm for for CP decomposition

Input: The tensor $\mathbf{T} = \sum_{l=1}^{p} \sqrt{n}\beta_l \boldsymbol{v}_l^{\otimes 3} + \mathbf{Z}$, m_{init}, m_{iter},ε,$\Lambda(\sim n)$
Goal: Estimate $\{\boldsymbol{v}_l\}_{l=1}^{p}$ and $\{\beta_l\}_{l=1}^{p}$.
$\mathcal{E} = \emptyset$
$\mathbf{T}^0 = \mathbf{T}$
for i=0 to m_{init} **do**
  Generate a random vector $\boldsymbol{v}_{i,0}$
  for j=0 to m_{iter} **do**
    $\boldsymbol{v}_{i,j+1} = \dfrac{\mathbf{T}^i(:, \boldsymbol{v}_{i,j}, \boldsymbol{v}_{i,j})}{\left\|\mathbf{T}^i(:, \boldsymbol{v}_{i,j}, \boldsymbol{v}_{i,j})\right\|}$
  end for
  if $\|\boldsymbol{v}_{i,m_{\text{iter}}-\Lambda}, \boldsymbol{v}_{i,m_{\text{iter}}}\| \geq 1 - \varepsilon$ **then**
    $\alpha_i = \langle \mathbf{T}^i, \boldsymbol{v}_{i,m_{\text{iter}}}^{\otimes k} \rangle$
    $\mathbf{T}^{i+1} = \mathbf{T}^i - \alpha_i \boldsymbol{v}_{i,m_{\text{iter}}}^{\otimes k}$
    $\mathcal{E} = \mathcal{E} + \{\boldsymbol{v}_{i,m_{\text{iter}}}\}$
  else
    $\mathbf{T}^{i+1} = \mathbf{T}^i$
  end if
end for
Output: Estimated vectors $\{\hat{\boldsymbol{v}}_l\}_{l=1}^{p} = \text{argmax}_{\mathcal{E}' \subset \mathcal{E}, |\mathcal{E}'|=p} \sum_{\boldsymbol{v} \in \mathcal{E}'} \mathbf{T}(\boldsymbol{v}, \boldsymbol{v}, \boldsymbol{v})$ and $\hat{\beta}_l \equiv \alpha_l / \sqrt{n}$

4.3 Empirical insights

In Figure 4.2, we compare the results of our algorithm with the state of art for $\beta \in \{1.2, 1.3, 1.4, 1.8, 2.2\}$ and for $n \in \{100, 200, 400\}$. We averaged over 50 different realizations of $\mathbf{T} = \mathbf{Z} + \sqrt{n}\beta \boldsymbol{v}^{\otimes 3}$ where $\mathbf{Z}$ is a tensor with random Gaussian components. We plot the correlation of the vector $\boldsymbol{v}$ output by each algorithm with the signal vector $\boldsymbol{v}_0$ as well as the 95% confidence interval bars. The algorithms considered are SMPI (that we perform after symmetrizing the tensor $\mathbf{T}$), the Homotopy-based algorithm (Hom) [ADGM17b], the Unfolding algorithm [RM14] (which are considered as the two main successful algorithms for Tensor PCA), as well as the CP tensor decomposition algorithm of the Python package TensorLy [KPAP19] used with a rank equal to one. Similar results are obtained for $n = 1000$ (only for SMPI and Homotopy) and on the non symmetric case and are provided in the Appendix.

The range of n used is the range commonly considered in empirical investigations of algorithms on Tensor PCA. Indeed, for $n = 1000$ the tensor has 10^9 non-zero entries which becomes extremely costly in memory and computational power. Furthermore, to the best of our knowledge, all algorithms investigated exhibited negligible finite size effects in this range of n (more details in subsection 4.5.3). We observe in Figure 4.2, that even for small instances of n, like $n = 100$, our algorithm performs significantly better than the state of art. The gap between the results of our algorithm and the state of art drastically increases with n.

Complexity of SMPI

The complexity of SMPI is equal to $m_{\text{init}} \times m_{\text{iter}} \times n^2$. In practice, $m_{\text{init}} = m_{\text{iter}} = 10n$ already gives us excellent results for a SNR in the range $1 < \beta < n^{1/4}$, the complexity is thus $\sim 10n^4$.

4.3.1 Theoretical insights on the SMPI algorithm

The new surprising empirical observation

Probability (Success from a random initialization after n power iterations) $\sim \frac{1}{\sqrt{n}}$ (4.6)

The mechanism: the role of the noise for signal recovery

In the power iteration, let's denote the part associated to the noise $\boldsymbol{g}_N$ and the one associated to the signal $\boldsymbol{g}_S$.

$$\mathbf{T}(:, \boldsymbol{v}, \boldsymbol{v}) = \mathbf{Z}(:, \boldsymbol{v}, \boldsymbol{v}) + \beta \langle \boldsymbol{v}, \boldsymbol{v}_0 \rangle^2 \boldsymbol{v}_0 \tag{4.7}$$

$$\equiv \boldsymbol{g}_N + \boldsymbol{g}_S \tag{4.8}$$

In Figure 4.3, we illustrate by an example, a pattern observed on all successful convergence in low SNR. For a given random initialization $\boldsymbol{v}^1$, we plot in blue (in both the top and bottom subfigures) the correlation between the signal $\boldsymbol{v}_0$ and $\boldsymbol{v}^i$ (obtained after i iterations on $\boldsymbol{v}^1$). In the top subfigure, we plot in orange the correlation between $\boldsymbol{v}_0$ and $\boldsymbol{g}_N / \|\boldsymbol{g}_N\|$, the normalized gradient associated to the noise tensor. In the bottom we plot in orange the ratio between the contribution of the noise gradient to $\boldsymbol{v}_0$: $\boldsymbol{g}_{N,0} \equiv \langle \boldsymbol{g}_N, \boldsymbol{v}_0 \rangle \boldsymbol{v}_0$ and the signal gradient $\boldsymbol{g}_S$. We observe an unexpected result: the gradient $\mathbf{Z}(:, \boldsymbol{v}^i, \boldsymbol{v}^i)$ is non-trivially correlated to $\boldsymbol{v}_0$ and thus partially converges to $\boldsymbol{v}_0$. Moreover, the spikes in the right figure at the beginning of the convergence suggest that $\mathbf{Z}(:, \boldsymbol{v}^i, \boldsymbol{v}^i)$ triggers the convergence towards $\boldsymbol{v}_0$ as it gives the largest contribution to the component of $\boldsymbol{v}^i$ correlated to $\boldsymbol{v}_0$ at the start of the convergence. This suggests that the gradient associated to the noise actually triggers the convergence towards the signal.

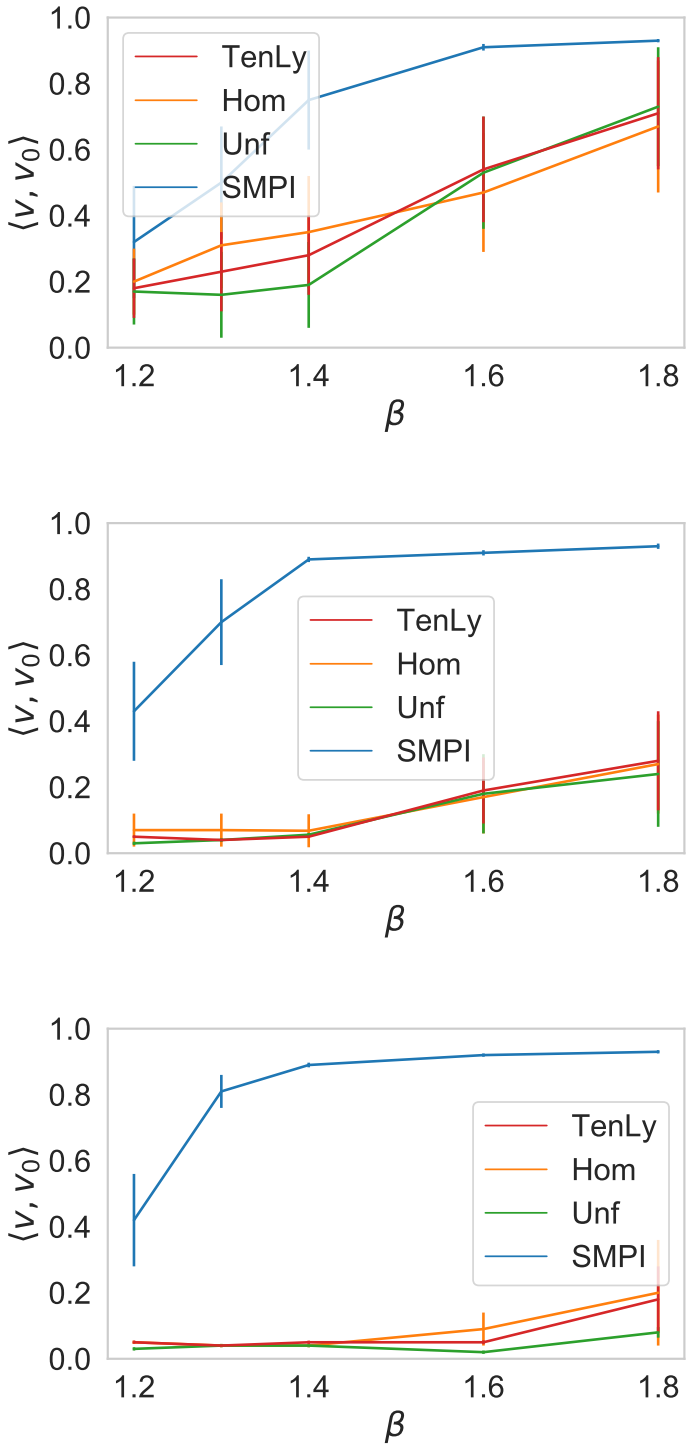


Figure 4.2: Comparison of the results of SMPI with TensorLy (TenLy) and the State-of-the-art represented here by the Unfolding (Unf) and Homotopy-based (Hom) methods for four values of the dimension of each axe of the tensor ($n = 100, 200, 400$). The results consist of the correlation between the output of each algorithm and the signal vector.

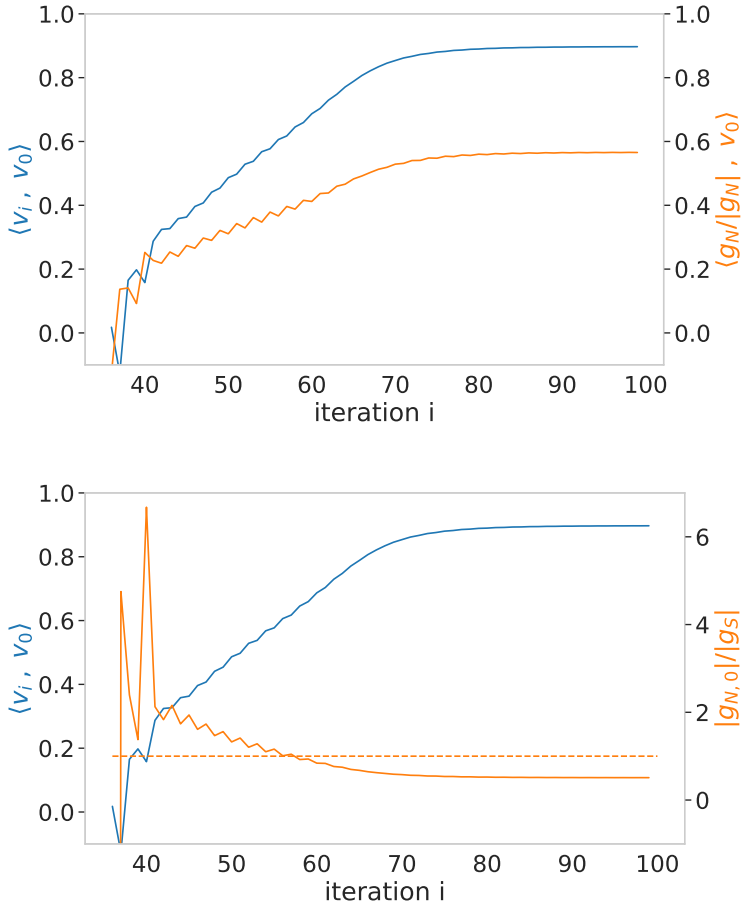


Figure 4.3: In blue, the correlation $\langle \boldsymbol{v}_i, \boldsymbol{v}_0 \rangle$ at each iteration i. In orange in the left, the correlation $\langle \frac{\boldsymbol{g}_N}{\|\boldsymbol{g}_N\|}, \boldsymbol{v}_0 \rangle$. In orange in the right, the ratio $\frac{\langle \boldsymbol{g}_N, \boldsymbol{v}_0 \rangle}{\langle \boldsymbol{g}_S, \boldsymbol{v}_0 \rangle}$

Towards the quantification of this mechanism

Let's denote $\boldsymbol{v}^*$ the output of the MLE (i.e. $\boldsymbol{v}^* = \arg\max_{\|\boldsymbol{v}\|=1}(\mathbf{T}(\boldsymbol{v},\boldsymbol{v},\boldsymbol{v})$) and $\hat{\boldsymbol{v}}$ the output of SMPI. We can compare the experimental value of the plateau of $\langle \frac{\boldsymbol{g}_N}{\|\boldsymbol{g}_N\|}, \boldsymbol{v}_0 \rangle = \langle \frac{\mathbf{Z}(:,\hat{\boldsymbol{v}},\hat{\boldsymbol{v}})}{||\mathbf{Z}(:,\hat{\boldsymbol{v}},\hat{\boldsymbol{v}})||}, \boldsymbol{v}_0 \rangle$ with its theoretical value obtained for $n \to \infty$ using analytic formula for $\|\mathbf{T}(:,\boldsymbol{v}^*,\boldsymbol{v}^*)\|$ and $\langle \boldsymbol{v}^*, \boldsymbol{v}_0 \rangle$ provided in [JLM+20] and the following formula for $\langle \frac{\mathbf{Z}(:,\boldsymbol{v},\boldsymbol{v})}{||\mathbf{Z}(:,\boldsymbol{v},\boldsymbol{v})||}, v_0 \rangle$ (a proof could be found in appendix 4.3.1):

$$\frac{\langle v, v_0 \rangle (\|\mathbf{T}(:,\boldsymbol{v},\boldsymbol{v})\| - \beta \langle \boldsymbol{v}, \boldsymbol{v}_0 \rangle)}{\sqrt{\|\mathbf{T}(:,\boldsymbol{v},\boldsymbol{v})\|^2 + \beta^2 \langle \boldsymbol{v}, \boldsymbol{v}_0 \rangle^4 - 2\beta \langle \boldsymbol{v}, \boldsymbol{v}_0 \rangle^3 \|\mathbf{T}(:,\boldsymbol{v},\boldsymbol{v})\|}} \tag{4.9}$$

For $\beta = 1.44\sqrt{n}$, the theoretical value is $\langle \frac{\mathbf{Z}(:,\boldsymbol{v}^*,\boldsymbol{v}^*)}{\|\mathbf{Z}(:,\boldsymbol{v}^*,\boldsymbol{v}^*)\|}, v_0 \rangle_{\text{th}} = 0.496$. The table 4.2

give the average and the standard deviation obtained experimentally for different n. We see that the theoretical value 0.496 is well inside the error bar and that the standard deviation gets smaller as n grows. This shows an excellent adequacy between the empirical results of the plateau related to $\hat{\boldsymbol{v}}$ and the theoretical expectation related to $\boldsymbol{v}^*$. For that matter, it suggests that the exploration of the landscape of Tensor PCA by SMPI already reached the large n regime in the experimental values of n considered.

Table 4.2: Experimental plateau for $\beta = 1.44\sqrt{n}$

n	50	100	150	200	400
$\langle \frac{\mathbf{Z}(:,\hat{\boldsymbol{v}},\hat{\boldsymbol{v}})}{\|\mathbf{Z}(:,\hat{\boldsymbol{v}},\hat{\boldsymbol{v}})\|}, \boldsymbol{v}_0 \rangle$	0.469 ± 0.148	0.518 ± 0.074	0.507 ± 0.056	0.487 ± 0.038	0.511 ± 0.025

Importance of the main features of SMPI

Importance of the symmetrized power iteration

For a non-symmetrical tensor, [HHYC20] proved that simple power iteration exhibits an algorithmic threshold strictly equal to $n^{1/2}$. For a SNR below this threshold, the output of the power iteration behaves like a random vector at each iteration. In Figure 4.4 we plot $\mathbf{T}(\boldsymbol{v}_i, \boldsymbol{v}_i, \boldsymbol{v}_i)$ for a simple power iteration (in the left) and for a symmetrized power iteration (in the right) for a non-symmetrical tensor and for small SNR ($\beta = 1.2\beta_{\text{th}}$ where β_{th} is the theoretical optimal threshold [JLM$^+$20]). We observe that while the result of simple power iteration matches the theoretical results obtained in [HHYC20], the symmetrized power iteration exhibits a fundamentally different and unexpected behavior that, to the best of our knowledge, has not been fully investigated so far. The fundamentally different behavior is very likely to its correspondence to a gradient descent that we explained in the beginning of this section.

The role of a large step size and the stopping condition

The landscape of the cost function $H(\boldsymbol{v}) \equiv -\mathbf{T}(\boldsymbol{v}, \boldsymbol{v}, \boldsymbol{v})$ is characterized by an exponentially large number of critical points [RBABC19]. To understand how a large step size (or equivalently the use of symmetrized power iteration) is essential for the gradient descent to escape many of the spurious minima that it may get trapped in, let's denote $\boldsymbol{m}_i$ a minimum of $\boldsymbol{v} \to \mathbf{T}(\boldsymbol{v}, \boldsymbol{v}, \boldsymbol{v})$ and $\{\boldsymbol{v}_i\}_{1 \leq i \leq n}$ the eigenvalues of the matrix $\mathbf{T}(:, :, \boldsymbol{m}_i)$. Let's assume that we are in the vicinity of $\boldsymbol{m}_i$ and initialize with $\boldsymbol{y}_0 = \dfrac{\boldsymbol{m}_i + \epsilon \boldsymbol{v}_1}{1+\epsilon^2}$ and let's note $\boldsymbol{y}_1 = \dfrac{\mathbf{T}(:, \boldsymbol{y}_0, \boldsymbol{y}_0)}{\|\mathbf{T}(:, \boldsymbol{y}_0, \boldsymbol{y}_0)\|}$. Thus

$$\begin{aligned} \mathbf{T}(:, \boldsymbol{y}_0, \boldsymbol{y}_0) =& \mathbf{T}(:, \boldsymbol{m}_i, \boldsymbol{m}_i) + 2\varepsilon \mathbf{T}(:, \boldsymbol{m}_i, \boldsymbol{v}_1) + O(\epsilon^2) \\ =& \mathbf{T}(\boldsymbol{m}_i, \boldsymbol{m}_i, \boldsymbol{m}_i)\boldsymbol{m}_i + \varepsilon\, 2\lambda_1 \boldsymbol{v}_1 + O(\epsilon^2) \end{aligned}$$

Hence, if $2|\lambda_1| > \mathbf{T}(\boldsymbol{m}_i, \boldsymbol{m}_i, \boldsymbol{m}_i) : \langle \boldsymbol{y}_1, \boldsymbol{m}_i \rangle < \langle \boldsymbol{y}_0, \boldsymbol{m}_i \rangle$ and $\langle \boldsymbol{y}_1, \boldsymbol{v}_1 \rangle > \langle \boldsymbol{y}_0, \boldsymbol{v}_1 \rangle$. Which means that if any of the eigenvalues of the Hessian matrix at $\boldsymbol{m}_i$ is smaller than $-2\mathbf{T}(\boldsymbol{m}_i, \boldsymbol{m}_i, \boldsymbol{m}_i)$, then the minimum is unstable under power iteration and the algorithm will diverge away from it.

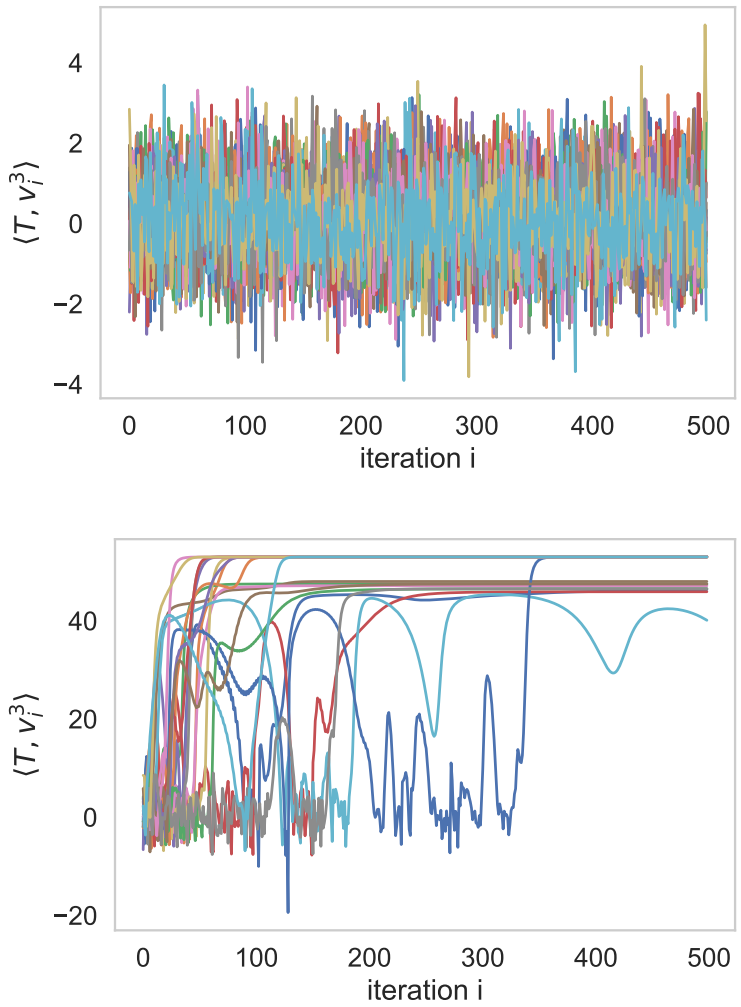


Figure 4.4: For different initialization, we plot $\mathbf{T}(\boldsymbol{v}_i, \boldsymbol{v}_i, \boldsymbol{v}_i)$ at each iteration i : in the top using a simple power iteration and in the bottom using a symmetrized power iteration ($n = 200$ and $\beta = 1.2\beta_{\text{th}}$)

This simple first order approximation analysis is enough to capture the behavior of SMPI when escaping local minima. Indeed, we can see in the Figure 4.5, a numerical example that illustrates this mechanism. $\boldsymbol{m}_j$ denotes the closest local minimum whose basin of attraction is slowing the gradient descent. $\boldsymbol{w}^j_{\text{min}}$ denotes the eigenvector associated to the smallest eigenvalue to the Hessian matrix at $\boldsymbol{m}_j$. $\boldsymbol{v}_i$ is the vector obtained after i iterations on a random initialization. In Figure 4.a, SMPI gets temporarily stuck in a local minima $\boldsymbol{m}_j$ (a stopping condition based on two consecutive iterations will likely trigger early stopping in this local minimum), we observe that at the same time $\langle \boldsymbol{v}_{i+1} - \boldsymbol{v}_i, \boldsymbol{w}^j_{\text{min}} \rangle$ grows and $\langle \frac{\boldsymbol{v}_{i+1} - \boldsymbol{v}_i}{\|\boldsymbol{v}_{i+1} - \boldsymbol{v}_i\|}, \boldsymbol{w}^j_{\text{min}} \rangle$ becomes close to 1. This illustrates an oscillation around the minimum $\boldsymbol{m}_j$ along the axis corresponding to the minimal eigenvector $\boldsymbol{w}^j_{\text{min}}$ of $\mathbf{T}(:, \boldsymbol{m}_i, :)$: w^j_{min} before diverging away from it as pictured in Figure 4.c. This exact same pattern happens in most of the initializations that succesfully converge towards the signal. In the appendix, we give numerical results on the averaged number of escaped minima.

Theoretical expression for the plateau in section 4.1

Let's consider $\boldsymbol{v}$ the final minima obtained by the maximum likehood estimator, given that it is a minima of a symmetric tensor we have the equality

$$\frac{\mathbf{T}(:, \boldsymbol{v}, \boldsymbol{v})}{\|\mathbf{T}(:, \boldsymbol{v}, \boldsymbol{v})\|} = \boldsymbol{v} \tag{4.10}$$

Computing the scalar product with $\mathbf{v_0}$ of each side and using that $\mathbf{T} = \mathbf{Z} + \beta \boldsymbol{v}_0^{\otimes 3}$ we have

$$\langle \mathbf{Z}(:, \boldsymbol{v}, \boldsymbol{v}), \boldsymbol{v}_0 \rangle + \beta \langle \boldsymbol{v}, \boldsymbol{v}_0 \rangle^2 = \langle \boldsymbol{v}, \boldsymbol{v}_0 \rangle ||\mathbf{T}(:, \boldsymbol{v}, \boldsymbol{v})|| \tag{4.11}$$

so

$$\langle \mathbf{Z}(:, \boldsymbol{v}, \boldsymbol{v}), \boldsymbol{v}_0 \rangle = \langle \boldsymbol{v}, \boldsymbol{v}_0 \rangle (||\mathbf{T}(:, \boldsymbol{v}, \boldsymbol{v})|| - \beta \langle \boldsymbol{v}, \boldsymbol{v}_0 \rangle) \tag{4.12}$$

On the other side, we also have

$$\|\mathbf{T}(:, \boldsymbol{v}, \boldsymbol{v})\| = \left\| \mathbf{Z}(:, \boldsymbol{v}, \boldsymbol{v}) + \beta \langle \boldsymbol{v}, \boldsymbol{v}_0 \rangle^2 \boldsymbol{v}_0 \right\| \tag{4.13}$$

so

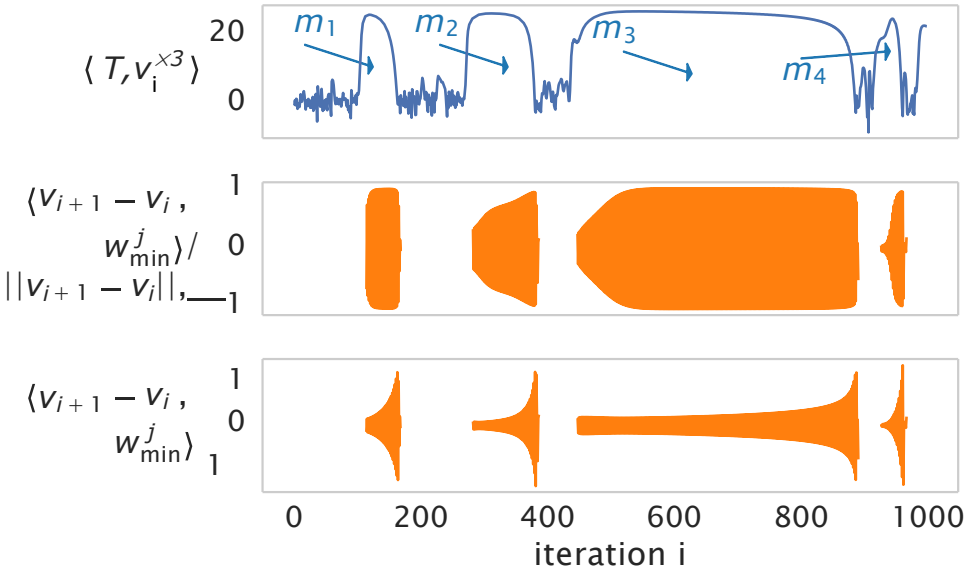
$$\|\mathbf{T}(:, \boldsymbol{v}, \boldsymbol{v})\|^2 = \|\mathbf{Z}(:, \boldsymbol{v}, \boldsymbol{v})\|^2 + \beta^2 \langle \boldsymbol{v}, \boldsymbol{v}_0 \rangle^4 + 2\beta \langle \boldsymbol{v}, \boldsymbol{v}_0 \rangle^2 \langle \mathbf{Z}(:, \boldsymbol{v}, \boldsymbol{v}), \boldsymbol{v}_0 \rangle \tag{4.14}$$

Thus

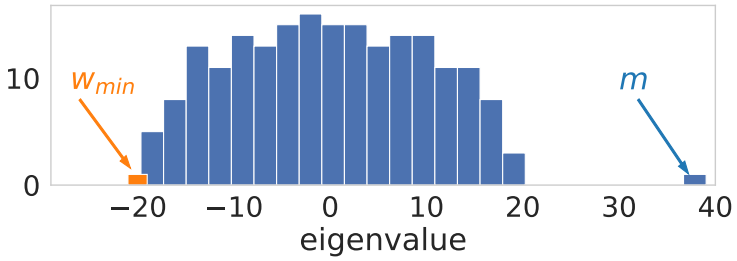
$$\frac{\langle \mathbf{Z}(:, \boldsymbol{v}, \boldsymbol{v}), \boldsymbol{v}_0 \rangle}{||\mathbf{Z}(:, \mathbf{v}, \mathbf{v})||} = \frac{\langle \boldsymbol{v}, \boldsymbol{v}_0 \rangle (||\mathbf{T}(:, \boldsymbol{v}, \boldsymbol{v})|| - \beta \langle \boldsymbol{v}, \boldsymbol{v}_0 \rangle)}{\sqrt{\|\mathbf{T}(:, \boldsymbol{v}, \boldsymbol{v})\|^2 - \beta^2 \langle \boldsymbol{v}, \boldsymbol{v}_0 \rangle^4 - 2\beta \langle \boldsymbol{v}, \boldsymbol{v}_0 \rangle^2 \langle \mathbf{Z}(:, \boldsymbol{v}, \boldsymbol{v}), \boldsymbol{v}_0 \rangle}} \tag{4.15}$$

replacing the obtaining expression of $\langle \mathbf{Z}(:, \boldsymbol{v}, \boldsymbol{v}), \boldsymbol{v}_0 \rangle$ in the right side of the equation

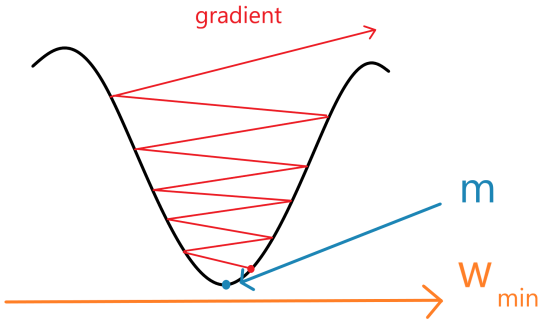
$$\frac{\langle \mathbf{Z}(:, \boldsymbol{v}, \boldsymbol{v}), \boldsymbol{v}_0 \rangle}{||\mathbf{Z}(:, \boldsymbol{v}, \boldsymbol{v})||} = \frac{\langle \boldsymbol{v}, \boldsymbol{v}_0 \rangle (||\mathbf{T}(:, \boldsymbol{v}, \boldsymbol{v})|| - \beta \langle \boldsymbol{v}, \boldsymbol{v}_0 \rangle)}{\sqrt{\|\mathbf{T}(:, \boldsymbol{v}, \boldsymbol{v})\|^2 + \beta^2 \langle \boldsymbol{v}, \boldsymbol{v}_0 \rangle^4 - 2\beta \langle \boldsymbol{v}, \boldsymbol{v}_0 \rangle^3 \|\mathbf{T}(:, \boldsymbol{v}, \boldsymbol{v})\|}}. \tag{4.16}$$



(a)



(b)



(c)

Figure 4.5: In (a) we observe that the algorithm gets stuck temporiraly in the basin of attraction of a local minimum $\boldsymbol{m}_j$. $\boldsymbol{v}_{i+1} - \boldsymbol{v}_i$ becomes correlated to w^j_{min} (the smallest eigenvector of $\mathbf{T}(:, \boldsymbol{m}_i, :)$: w^j_{min} as illustrated in (b)) and its norm grows until the algorithm diverges away from $\boldsymbol{m}_j$. This simple mechanism is illustrated in (c)

The role of a polynomial iterations

It is commonly assumed that Tensor power iteration functions by increasing the correlation with the signal at each step similarly to the matrix case. [RM14] performed a heuristic analysis with a zero order approximation that suggests that an initialization that verifies $\beta\langle \boldsymbol{v}, \boldsymbol{v}_0\rangle > 1$ is required for the method to succeed by increasing the correlation. Yet, the results in Figure 4 left, where we plot in red the initial correlation of successful vectors, strongly suggests that the success is not correlated with its initial correlation. Moreover, Figure 4 right where we plot the correlation with the signal in function of the iteration shows that SMPI has a first long phase of fluctuation of exploring the landscape. Indeed, the operating mode of power iteration seems to be different when we consider moderately long times ($O(n)$), which indicates that this new mechanism is fundamentally different from what happens in the matrix case, and that a polynomial number of iterations is required. Indeed a logarithmic convergence without exploration of the landscape would require a large initial correlation with the signal vector. Below the threshold $O(n^{1/2})$, this has an exponentially small probability to happen given that the distribution of the correlation of a random vector $\boldsymbol{v}$ with the signal follows a normal law. A recent paper [DH22] shows that power iteration fails with a number lower than polynomial using communication complexity.

The role of a polynomial initialisation

In Table 4.3, we reported in green the average of the number of initializations required to reach a success rate of $r = 99\%$ with $m_{\text{iter}} = 10 * n$ over 10 independent runs for each n with SMPI. This is calculated by computing the percentage of successful initializations p and then using the formula that gives the probability that at at least one of the initializations succeeds: $1 - (1-p)^{m_{99\%}} = 0.99$. In red we reported the number of initializations required for the naive power initialization with logarithmic steps to succeed, which is approximately exponential. The drastic discrepancy between the two quantities suggest that SMPI has a polynomial complexity (and not an exponential complexity) even for $n > 1000$.

n	50	100	200	400
$\exp(n) \simeq$	10^{21}	10^{43}	10^{86}	10^{173}
m_{init} - 99%	10	45	71	228

Table 4.3: In green the average number of initializations required for a recovery rate success of 99% where we see that it is linear in n. In red the approximation of $\exp(n)$ which is the number of required initializations if the complexity were exponential.

4.3.2 Insight on the success

Tensor power iteration with a random initialization is supposed to perform poorly with a computational threshold scaling as $n^{1/2}$ [RM14] in contrast with other algorithms (such as unfolding, sum of squares, homotopy, etc.) whose algorithms thresholds scale as $n^{1/4}$. In order to understand the reason behind this failure, a first line of research focused on

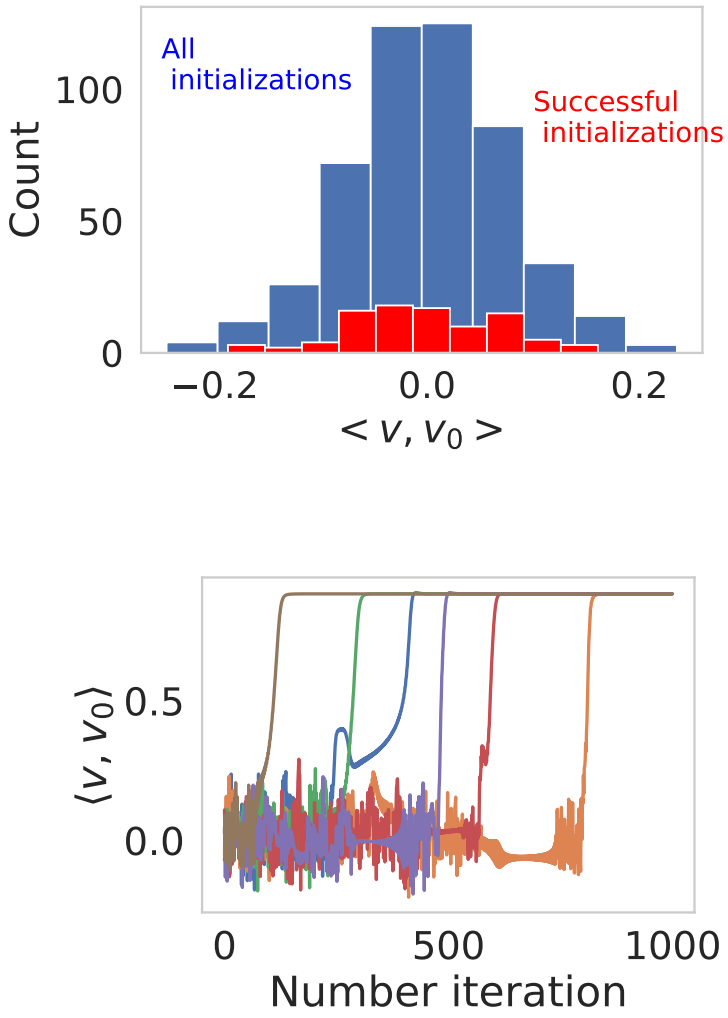


Figure 4.6: Top : In blue the distribution of the correlation between the signal and all the initializations, and in red the correlations of the initializations that succeeded. Bottom: Each color represents the trajectory of the algorithm for an initalization ($n = 200$, $\beta = 1.2$).

the complexity of the landscape (such as [BAMMN19] and [RBABC19]) showing the existence of an exponential number of spurious minima where the algorithm could get stuck. However, a more recent paper [BAGJ+20] provided a proof for the failure of Langevin dynamics as well as gradient descent with infinitely small learning rate suggesting that the failure of local algorithms are "*actually due to the weakness of the signal in the region of maximal entropy for the uninformative prior*".

It is thus very interesting to take advantage of the numerical analysis we performed on SMPI to understand how it would be able to bypass this these possible explanations to the failure of local iterative algorithms. (i) In the previous subsection, we observed that for the majority of the successful convergence towards the signal, the algorithm runs through many spurious minima but is still able to escape them thanks to its large step size. This provides a possible explanation in how SMPI is able to navigate through such a rough landscape in order to attain the signal vector. (ii) Numerical simulations for a low enough SNR and a large enough n (e.g. $n \geq 100$) shows that for every successful convergence towards the signal, it is the gradient associated to the noise $\mathbf{Z}(:, \boldsymbol{v}, \boldsymbol{v})$ that not only triggers the convergence but also carries it. This mechanism that we exhibited is consistent with [BAGJ+20] and the fact that the signal is indeed too small for its associated gradient to converge towards the signal by itself. However, our results bring a novel important element (which has never been considered before to the best of our knowledge) which is that the noise gradient is also able to play an crucial role in the convergence. Thus, thanks to this phenomenon, the smallness of the signal does not necessarily imply the failure of the algorithm.

4.4 Numerical simulations details

Simulations for the comparison between SMPI and the unfolding method for $n = 1000$ were run on a cloud provider (AWS). All the other simulations were run in Python on a Dell computer running Ubuntu 18.04.5 LTS with eight Intel Core i7-4800MQ processors at 2.70 GHz and 16GB of RAM.

Spike with different dimensions

In Figure 4.7, we investigate the case of recovering a spike $\boldsymbol{v}_1 \otimes \boldsymbol{v}_2 \otimes \boldsymbol{v}_3$ with different dimensions $\mathbf{T} = \beta \left(\frac{n_1+n_2+n_3}{3}\right)^{1/2} \boldsymbol{v}_1 \otimes \boldsymbol{v}_2 \otimes \boldsymbol{v}_3 + \mathbf{Z}$. For different sets of three dimensions $\{(50, 75, 100), (75, 75, 75), (50, 50, 150), (50, 100, 100)\}$ we plot the correlation between the outputs of the algorithm and the signal vectors (in red $\langle \hat{\boldsymbol{v}_1}, \boldsymbol{v}_1 \rangle$, in blue $\langle \hat{\boldsymbol{v}_2}, \boldsymbol{v}_2 \rangle$ and in green $\langle \hat{\boldsymbol{v}_3}, \boldsymbol{v}_3 \rangle$) averaged over 50 different realizations. We see that the empirical threshold matches $(n_1 + n_2 + n_3)^{1/2}$ which corresponds to the optimal theoretical threshold for a tensor with dimension $n_1 \otimes n_2 \otimes n_3$.

Multiple spikes case

We investigate in Figure 4.8 the performance of the variant of SMPI for CP decomposition. We take β equal for all the spikes since it is considered as the most difficult case for spectral algorithms.

We repeat 30 times the following instance:

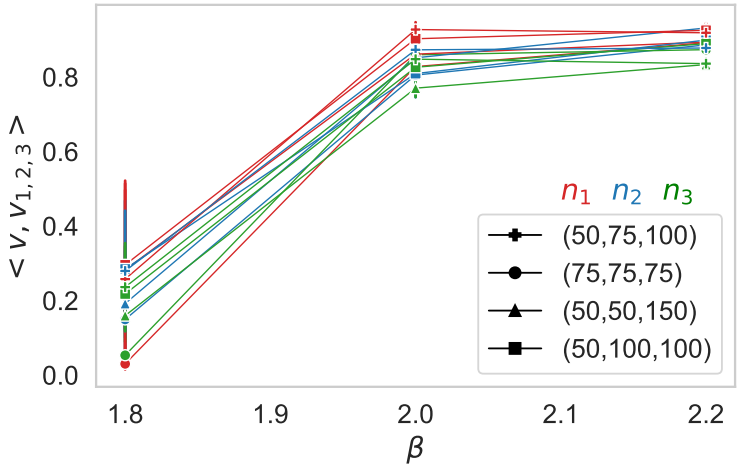


Figure 4.7: For different sets of three dimensions (n_1, n_2, n_3), we generate different realizations of $\mathbf{T} = \beta \left(\frac{n_1+n_2+n_3}{3}\right)^{1/2} \boldsymbol{v}_1 \otimes \boldsymbol{v}_2 \otimes \boldsymbol{v}_3 + \mathbf{Z}$ where $\mathbf{Z}$ is a gaussian random tensor. We plot the correlation between the output of Algorithm 5 and the signal vector (in red for $\langle \hat{\boldsymbol{v}_1}, \boldsymbol{v}_1 \rangle$, in blue for $\langle \hat{\boldsymbol{v}_2}, \boldsymbol{v}_2 \rangle$ and in green for $\langle \hat{\boldsymbol{v}_3}, \boldsymbol{v}_3 \rangle$) in function of β.

- Generate the tensor $\mathbf{Z}$ with iid Gaussian components
- Generate n_{spikes} independent unitary random vectors. Each of these vectors is obtained by generating a vector with iid Gaussian components and then normalizing it.
- Compute $\mathbf{T} = \mathbf{Z} + \beta \sum_{l=1}^{m} \boldsymbol{v}_l$
- Plot the percentage of successfully recovered vectors by algorithm 6, naive power iteration and the CP decomposition algorithm provided by the package TensorLy [KPAP19].

In Figure 4.8, we compare the percentage of succesfully recovered vectors with a naive power iteration and the CP decomposition algorithm provided by the package TensorLy [KPAP19]. We see that our algorithm outperforms existing methods.

We investigate in Figure 4.9 the case of a number of spikes larger than the dimension $n_{\text{spikes}} > n$. We see that even if the number of spikes is larger than the dimension, the algorithm still outperforms other methods.

4.4.1 The averaged number of escaped spurious minima for a successful initialization in function of n

In Table 4.4, we counted, for a successful initialization, the averaged number of spurious minima where the algorithm gets temporarily stuck before escaping thanks to its large step size. For different values of n, we repeated the experience for 20 independent instances and averaged the number of escaped minima before converging to the signal.

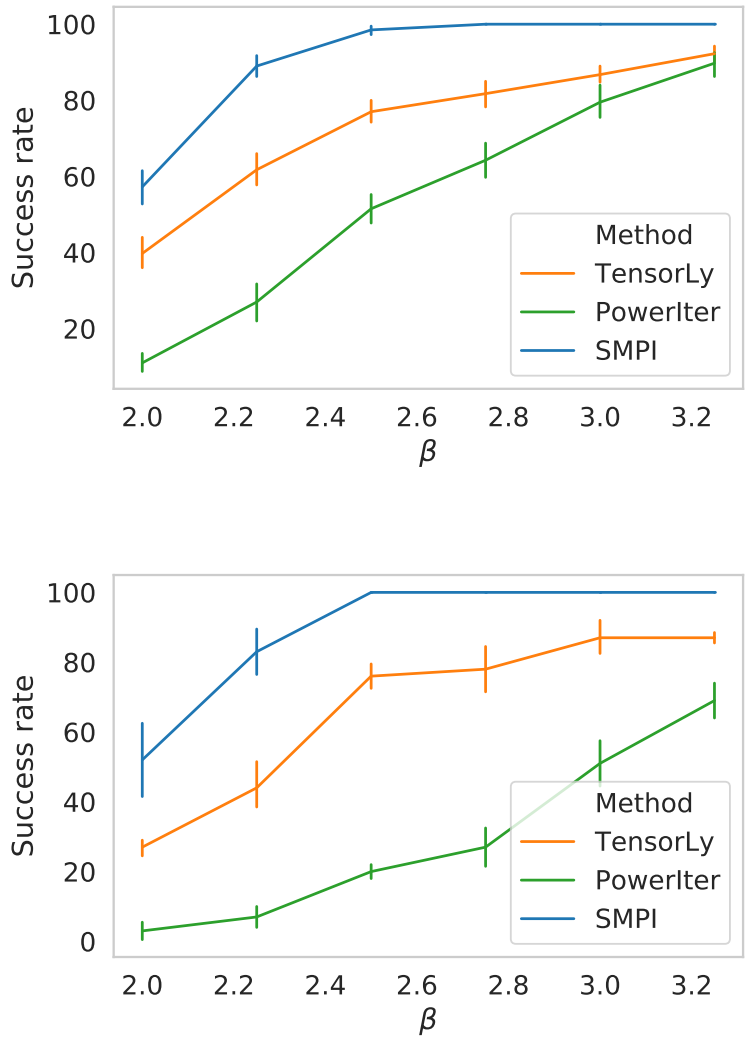


Figure 4.8: For a number of spikes equal to 20, we plot the percentage of recovered spikes for different β for $n = 100$ in the top and $n = 150$ in the bottom. We see that SMPI (blue) outperforms the naive power iteration algorithm (green) and the TensorLy algorithm (orange)

Table 4.4: The averaged number of escaped spurious minima for a successful initialization in function of n

n	50	100	150	200
number of escaped minima	0.63	1.11	1.23	2.16

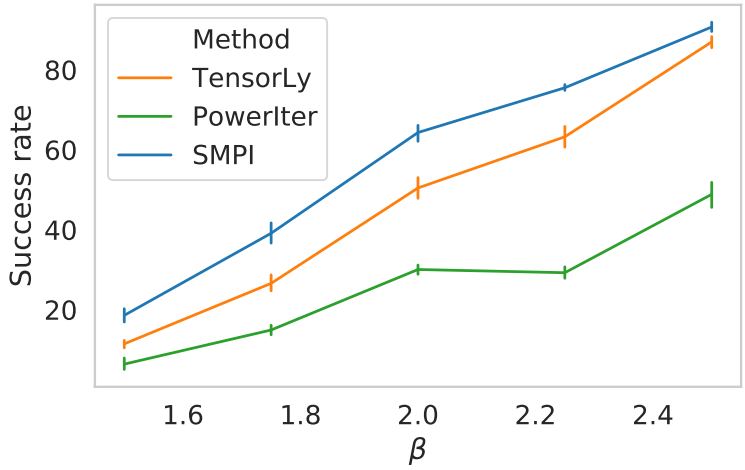


Figure 4.9: For a number of spikes equal to 150, we plot the percentage of recovered spikes in function of β averaged over 50 different tensors $\mathbf{T}$ with $n = 100$

4.4.2 Practical applications: Hyperspectral images (HSI).

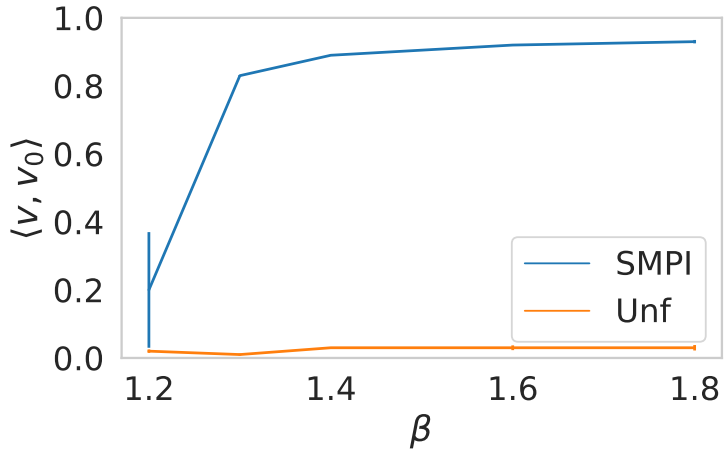
As explained in [Nas13], "Typically, a hyperspectral spectrometer provides hundreds of narrow contiguous bands over a wide range of the electromagnetic spectrum. Hyperspectral sensors measure the reflective (or emissive) properties of objects in the visible and short-wave infrared (IR) regions (or the mid-wave and long-wave IR regions) of the spectrum. Processing of these data allows algorithms to detect and identify targets of interest in a hyperspectral scene by exploiting the spectral signatures of the materials".

Denoising is an important preprocessing step to further analyze a hyperspectral image (HSI). The common denoising procedures are either based on 2-dimensional (2D) methods (mainly 2D filters) or tensor decomposition methods. In particular, there is in general two mainly used models for tensor decomposition: Tucker decomposition and CP decomposition (also called PARAFAC) that have different advantages and shortcomings. A detailed survey of these methods could be found in [KB09] for example.

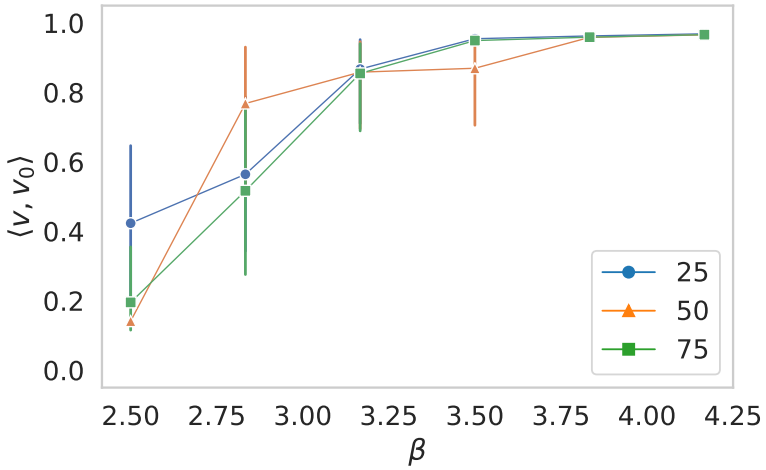
In [LBF12], the authors compared CP decomposition method based on the Alternating Least Square (ALS) algorithm with existent methods (two-dimensional filter and Tucker3 that is based on a Tucker decomposition) to denoise HSI. They performed their experiments on a real world data: the Airborne Visible/Infrared Imaging Spectrometer (AVIRIS) HSI, an airborne hyperspectral system flown by NASA/Jet Propulsion Laboratory (JPL). To analyze quantitatively the denoising results, they compare the signal-to-noise ratio (SNR) of the denoised image that is defined as:

$$\text{SNR}_{\text{out}} = 10 \log_{10} \frac{||\hat{\mathcal{Y}}||^2}{||\mathcal{Y} - \hat{\mathcal{Y}}||^2} \tag{4.17}$$

where $\mathcal{Y}$ is the original image and $\hat{\mathcal{Y}}$ the estimated image after denoising.



(a) Comparison between the unfolding method and SMPI for $n = 1000$ for different β



(b) Correlation of the SMPI output $\boldsymbol{v}$ with the signal $\boldsymbol{v}_0$ for different $n \in \{25, 50, 75\}$ for a tensor of order 4: $\mathbf{T} \in \mathbb{R}^{n^{\otimes 4}}$

Figure 4.10: The case $n = 1000$ in the left and the performance of SMPI in the case of $k = 4$ in the right

Their numerical results show that the CP decomposition model using the ALS algorithm performs better than other considered methods as a denoising procedure.

In order to judge the performance of our algorithm, we perform the same experiment with the same estimator and compare it with the ALS algorithm using the Python TensorLy package [KPAP16]. The hyperspectral image we use is the open source data given in [MGT+18] that we normalize. It consists of a tensor of size $\mathbf{R}^{425\times 861\times 475}$ where 425 is the number of spectral bands and 865×475 is the spatial resolution.

CP decomposition model decomposes a tensor as a sum of rank-one tensors (that we call spikes). Thus, we first compare the SNR for different number of spikes in Table 4.5. Then we compare the time taken for each algorithm (in seconds) in Table 4.6.

Table 4.5: Comparison between ALS based on TensorLy and SMPI

n_{spikes}	150	200	400	800
ALS (TensorLy)	43.58	54.24	60.58	66.53
SMPI	**44.24**	**54.53**	**60.93**	**66.91**

Table 4.6: Time for each method

n_{spikes}	150	200	400	800
ALS (TensorLy)	**4**	**377**	1566	2325
SMPI	33	387	**995**	**1823**

We see that the proposed SMPI method gives better denoising results independently of the number of spikes. And we see that the ALS algorithm (using the TensorLy package) is faster for small number of spikes but becomes slower than SMPI for a larger number of spikes. Given that the optimal number of spikes is in general > 100 (in [LBF12] the optimal was for 169 spikes), this suggests that SMPI gives better results in a shorter amount in time than ALS.

More generally, it is important to note that there is many more practical applications where CP decomposition and where SMPI could be an excellent candidate for improving existent performance. Here is a non-exhaustive list of such applications:

In telecommunication, CP decomposition is used for Tensor-based modulation [DLG20]. It used for "Massive random access, whereby a large number of transmitters communicate with a single receiver, constitutes a key design challenge for future generations of wireless systems."

An important other application is the convolutional neural networks compression: for example a CP decomposition method based on power iteration has been suggested in [AL17] that showed a significant reduction in memory and computation cost. Given that our method could be seen as a refinement of the simple power iteration method, we believe that it could be very interesting to try SMPI on these models.

In [SB20], where different types of higher order data in manufacturing processes are described, and their potential usage is addressed using methods like CP tensor decomposition.

4.5 Potential impact and open questions

4.5.1 Insights on the gradient-based exploration of high-dimensional non-convex landscapes

Many recent papers [MBC+19, MKUZ19, MBC+20] utilized the landscape of Tensor PCA and its variant Matrix Tensor PCA in order to investigate the behavior of gradient descent in non convex high dimensional landscapes even in regimes where it should be hard. To the best of our knowledge, the mechanism we exhibit has not been considered before. This may be due to the fact that a large step size is required especially that our results suggest that this may be fundamentally important . Thus, this algorithm could bring a novel perspective in how the random landscape itself can play an important role in the convergence towards the signal. Furthermore, it pushes us to be careful when generalizing results obtained using gradient flow to results on gradient descent with large step size.

4.5.2 Insights on the statistical-computational gap conjecture

Comparison with the predicted maximal theoretical results

[JLM+20] gives an analytical expression for the asymptotic theoretical optimal correlation with the signal vector for $n \to \infty$. We plot in Figure 4.11 the asymptotical curve in dashed line next to the results of SMPI for $n = 50, 75, 100, 150, 300, 400$. We obtain a remarkable result: the empirical performance of SMPI converges towards the optimal performance for $n \to \infty$. Indeed the transition gets sharper when n grows and its behavior gets closer to a discontinuous transition. Thus, these results suggest that not only the SMPI algorithm outperforms the state of the art but it may be matching the optimal information theory performance.

Empirical scaling of the threshold

More specifically, to quantify this scaling, we will assume that the threshold could be written as $\beta = cn^{\alpha}$ and our aim is to recover α empirically. Given n_1 and n_2, two values of n, we have:

$$\log(\beta_{n_1}) = \log(c) + \alpha \log(n_1) \tag{4.18}$$

$$\log(\beta_{n_2}) = \log(c) + \alpha \log(n_2) \tag{4.19}$$

Thus, the empirical α can be computed from the empirical thresholds as follow:

$$\alpha_{\text{emp}} = \frac{\log\left(\beta_{n_2}^{\text{emp}}\right) - \log\left(\beta_{n_1}^{\text{emp}}\right)}{\log(n_2) - \log(n_1)} \tag{4.20}$$

In the Table 4.7, we fix $n_1 = 100$ and vary n_2 in the set $\{150, 200, 400, 800\}$. We perform 50 times the experiment for each n_2 and define β^{emp} as the smaller β such that $\langle \boldsymbol{v}, \boldsymbol{v}_0 \rangle > 0.95 \langle \boldsymbol{v}, \boldsymbol{v}_0 \rangle_{\text{th}}$ where $\langle \boldsymbol{v}, \boldsymbol{v}_0 \rangle_{\text{th}}$ is the theoretical optimal correlation in the large n

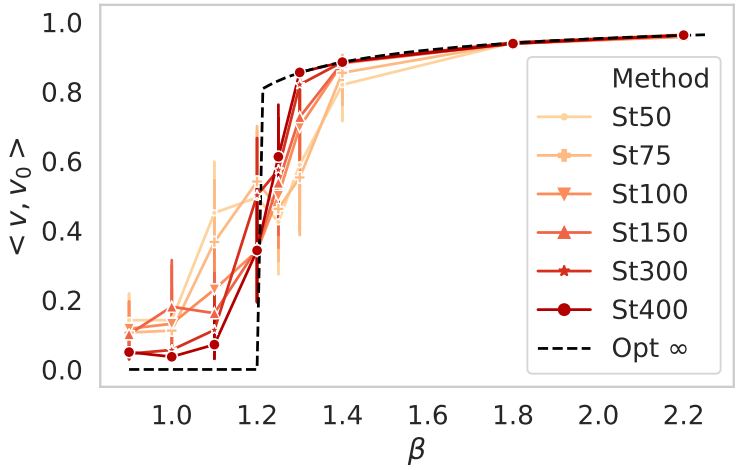


Figure 4.11: Asymptotic behavior of SMPI method (denoted St) illustrated by different results on various values of n (from 50 to 400). The dashed line (Opt ∞) corresponds to the optimal theoretical result for $n = \infty$.

limit given in [JLM+20]. The four methods have approximately constant α, which corroborate our assumption that the threshold behaves like $\beta = cn^{\alpha}$ in this range of n. Simple PI, Homotopy and Unfolding exhibits empirical thresholds similar to their theoretically conjectured ones (respectively $\frac{1}{2}, \frac{1}{4}, \frac{1}{4}$), while SMPI has a threshold approximately equal to zero (slightly negative as its performance improves with n).

Table 4.7: Experimental scaling for a non-symm. tensor for simple power iteration, unfolding, homotopy and SMPI

n_2	150	200	400	800
Simple PI	0.541	0.528	0.531	0.513
Homotopy	0.235	0.245	0.248	0.246
Unfolding	0.23	0.248	0.26	0.2516
SMPI	-0.063	-0.052	-0.053	-0.036

4.5.3 Discussion on a potential finite size effects

It is logical and important to first consider the possibility that the experimental results that we obtained could be due to finite size effects. Therefore, we stress that the main aim and motivation of this work is not closing the gap but rather to provide novel theoretical and experimental insights that will help us understand better this conjectured gap either to prove it rigorously or to rule it out.

First, it is important to note that, for the best of our knowledge, all existent algorithms exhibit negligible finite size effects for $n > 100$. Indeed, it is claimed that the empirical behavior of algorithms matches the theoretical behavior for $n \geq 50$ for Unfolding and

Tensor Power Iteration [RM14], $n \geq 30$ for Averaged Gradient Descent, $n \geq 100$ for Robust Tensor Power iteration [WA16] and $n \geq 50$ for Higher Order Orthogonal Iteration of Tensors (HOOI) [ZX18]. In our case, SMPI exhibits a constant threshold for $100 \leq n \leq 1000$. Secondly, it could be interesting to discuss our results in light to a recent paper [BKW20] on the importance of considering finite size effects for low degree polynomial methods in a matrix model. We discuss the fundamental difference between our results and their observations:

- In their paper, the final asymptotic true result and the conjectured result by [MR15] differ only by a constant factor ($\sqrt{2}$) (and is due to a slow convergence speed of $O(n^{-1/2})$ to the asymptotic value according to [BKW20]). In contrast, in our case, constant factors are not relevant. Furthermore, the difference between the threshold of our empirical results and the naive power iteration is fundamentally different and differ by a factor of $n^{1/2}$. Therefore, it seems unlikely that a slow convergence speed similar to the one in [BKW20] will cause such a fundamental and massive change in the asymptotic behavior. More specifically, the algorithmic feasibility of SMPI can be quantified by the number of initializations necessary to recover the signal with a linear number of iterations (if it is polynomial or exponential). As we show in our Table 4.7 of our main paper, the increase of required initialization has an empirical linear scaling up to $n = 800$. So it may be extremely unlikely that it will suddenly jump to an exponential number.
- It is important to point out that the dynamics of the empirical value is also fundamentally different. Indeed, in the case of [BKW20], the empirical performance stagnates at the perfect theoretical bound 1 until $n = 10^4$, where it begins decreasing and converges to the true asymptotic value 1.8 as n increases. This decrease of performance as n increases is very typical of finite size effects. In contrast, the dynamics of the evolution of the performance of SMPI is fundamentally different: in Figure 3.a of our paper, we observe that as n grows, the average correlation actually improves and converges towards the theoretical result!
- Moreover, note that, their problematic concerns the matrix case. However, we argue that we can't compare directly n for a tensor $\mathbf{T}$ and a matrix $\mathbf{M}$. Indeed, there is more random variables in a tensor with $n_1 = 10^3$ than in a matrix with $n_2 = 10^4$ given that $n_1^3 > n_2^2$. In addition, updating the vector with a power iteration for $\mathbf{M}.\mathbf{v}$ (where $\mathbf{M}$ has n_2 elements) consists in a sum of 10^4 (pondered) random gaussian variables, while $\mathbf{T}.\mathbf{vv}$ (where $\mathbf{T}$ has n_1 elements) consists in a sum of 10^6 (pondered) random variables. The same goes for the operator norm of the matrix $\max_{v \in S_{n_2}} \mathbf{Mvv}$ that sums n_2^2 variables and the tensor $\max_{v \in S_{n_1}} \mathbf{Tvvv}$ that sums n_1^3 variables (note that the Maximum likehood estimator returns the vector that maximises the operator norm). Thus, Central limit theorem suggests that a tensor with $n_1 = 10^3$ should have less fluctuations and finite size effects than a matrix with $n_2 = 10^4$. Finally, and as explained in our answer to first point of the reviewer 3, $n = 10^5$ for a tensor will require a storage capacity of 8PetaBytes that is 100 times larger than the best high-memory offers of cloud computing.

Chapter 5

Conclusion and perspectives

5.1 Conclusion

The contribution of this thesis could be divided in two main projects. In the first one, we introduced a novel framework for the tensor decomposition based on trace invariants. Within this framework, we provide different algorithms to recover a signal vector with theoretical guarantees. These algorithms use tensor contractions that has a high potential of parallelization and computing optimization. We illustrate the practical pertinence of our framework by presenting some examples of algorithms as well as generalizations of these algorithms for Canonical polyadic (CP) decomposition and Tucker decomposition methods. Our experimental results show that the tetrahedral graph performs better than the the state of the art for Tensor PCA, and that its tensor decomposition generalizations show a better robustness to noise comparing to existent algorithms. Interestingly, our framework is also able to extend the theoretical and practical study of tensor PCA to new and less restrictive situations like data where the dimensions of the axes are different. Important directions of future research is to explore the potential of more general graphs, as well as investigate the new proposed theoretical threshold for different dimensions in the context of tensor decomposition.

In the second project, we introduced a novel algorithm named Selective Multiple Power Iteration (SMPI) for the Tensor PCA problem. Various numerical simulations for $k = 3$ in the conventionally considered range $n \leq 1000$ show that the experimental performance of SMPI improves drastically upon existent algorithms and becomes comparable to the theoretical optimal recovery. We also provide in the supplementary material multiple variants of this algorithm to tackle low-rank CP tensor decomposition. These proposed algorithms also outperforms existent methods even on real data which shows a huge potential impact for practical applications. Thus, for future work, it seems very interesting to go further in terms of theoretical investigations of these new insights offered by SMPI and also their consequences for related problems: the study of the behavior of gradient descent methods for the optimization in high-dimensional non-convex landscapes that are present in various machine learning problems and also the study of the conjectured statistical-algorithmic gap.

5.2 Perspectives

Applications

One important perspective is to look more profoundly at the potential applications and more extensive examples and models where these new tools could be useful. Indeed, real life applications concern size scales similar to our experiments where we obtained very encouraging preliminary results and a substantial empirical improvement over the state of the art was observed. Indeed, Tensor decomposition have been used to a large and various number of fields. This could cover for example the compression of neural networks, or the denoising of hyperspectral images. There have been multiple works [AL17, WY+20, WGY21] on the compression of neural network that helps the interpretability and the speed of neural network by reducing the size of each tensor layer.

There is multiple methods that investigate algorithms based on deflation [dSCdA15a]. That is an algorithm that perform CP decomposition of a high rank from an algorithm that performs a rank-one approximation. The performance of these methods may differ whether the rank is low or large.

For some applications, we can study adaptation of our frameworks and methods to other type of data such as sparse tensors [NWZ20, Cd21].

Spin glass

Another direction of research concerns optimization of Spherical p-spin glass, that is approximating the deepest minimum value by searching whether a near optimal configuration can be computed in polynomial time rather than exponential [EAMS21]. Preliminary results and empirical observation seems to suggest that we are able to recover a polynomial number of vectors among which the signal vector is found before the statistical threshold as long as we are above the critical threshold. However, we find deeper minima so for Tensor PCA, Maximum Likehood Estimator won't be able to indicate which vector is the signal. But for optimization purposes, it could be very interesting to investigate such intriguing results.

Combination of graphs

One could ask oneself what is the best class of graphs. A further idea is to investigate if a sum of graphs provide better results. Our first experiments shows a high promise. This is based on the simple fact that by summing two different variables, the variance of their sum is smaller than the sum of their variance. Variance of the noise matrix gets smaller, but the pure signal matrix adds as it is. Empirical results on the sum of two matrices confirm this intuition. This is illustrated in the Figure 5.1

Random Tensor Theory for SMPI

Oleg Evnin [Evn20] began a study of power iterations based on random tensor theory and trace invariants. As we explained in Chapter 2, it amounts to the study of the associated graph illustrated in Figure 5.2. Based on the hypothesis in 4.6, the objective is to prove

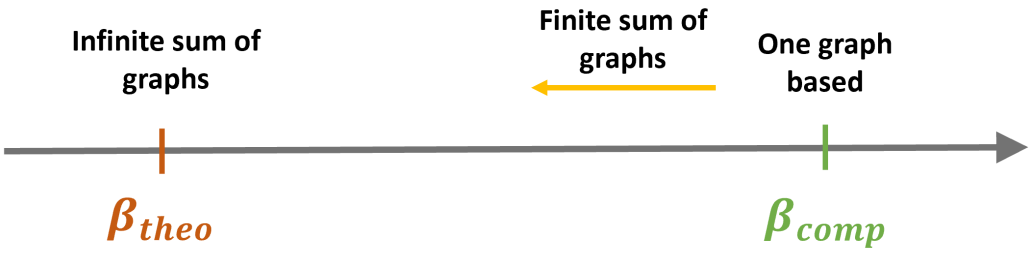


Figure 5.1: A finite sum of graphs might improve the performance

that the expectation of this graph is non zero at the first order of $1/N$. Given our empirical results, it is interesting to further pursue this investigation by considering a polynomial number of iterations. This could be done by adapting the tools used by Evnin and examining the rate of growth of a random initial vector under successive applications of a nonlinear map defined by the random tensor.

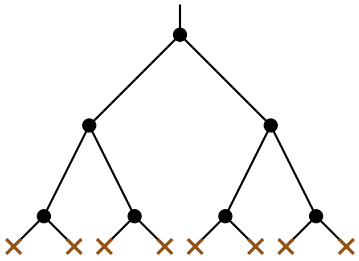


Figure 5.2: The graph associated to the power iteration method with 3 iterations for an initialization v. The cross represents the vector v and the black dot the tensor $\mathbf{T}$

Discreet step size investigation for SMPI

[BAGJ21, BAGJ22] proposed a novel approach to study statistics associated to the stochastic gradient descent (SGD) depending on the initialization and the discreet step size. They found out a new threshold for the step size above which the behavior of SGD changes fundamentally. Adapting this approach for SMPI could be extremely useful for the theoretical understanding of its performance and the importance of discreet variable step size in SMPI. On the other hand, obtaining theoretical understanding of SMPI through this approach may bring important insights on the performance of SGD in the deep learning framework.

Appendix A

Synthèse de la thèse en Français :

L'intelligence artificielle (IA) et l'apprentissage automatique (ML) ont démontré leur potentiel pour révolutionner les industries, les services publics et la société, atteignant ou même dépassant les niveaux humains de performance en termes de précision pour plusieurs applications, tels que la reconnaissance d'images et de la parole [MKS+15] et la traduction linguistique [YHPC18].

S'appuyant sur son énorme potentiel, l'IA gagne rapidement en influence dans la vie quotidienne des gens et dans des domaines professionnels tels que la santé, l'éducation, la recherche scientifique, les communications, les transports, la sécurité et l'art. Cependant, alors même que les systèmes d'IA commencent à être largement déployés dans l'économie, de multiples problèmes associés à l'IA s'amplifient.

Une difficulté majeure fréquemment évoquée est l'interprétabilité des méthodes. Cela s'est aggravé avec la diffusion des technologies basées sur le ML dans des domaines critiques pour la sécurité tels que les soins de santé, la finance, le droit, la défense et la gouvernance, qui exigent la responsabilité des décisions et de la manière dont les données sont utilisées. En effet, de tels domaines nécessitent la confiance des utilisateurs dans une décision qui est obtenue en ayant une méthode qui est facilement interprétable, liée à l'utilisateur, connecte la décision avec des informations contextuelles, des lois connues et des expériences antérieures et reflète le mécanisme de pensée de l'utilisateur pour atteindre la décision.

D'autre part, de plus en plus de travaux se concentrent sur la construction de modèles légers de ML tels que les petits réseaux de neurones (pour les appareils Internet des objets (IoT), la formation en temps réel, etc.) qui nécessitent moins de puissance de calcul, sont plus pratiques, intéressants, utiles et généralement plus interprétables.

Pour résoudre ces problèmes, diverses recherches ont été menées pour améliorer la prise de décision impartiale et impartiale, améliorer la capacité de généralisation des modèles à des domaines de données plus larges et développer des explications pour les modèles ML. Ces objectifs sont fortement dépendants les uns des autres et l'interprétabilité est un aspect fondamental qui améliore les deux autres.

Deux principaux axes de recherche sont poursuivis pour améliorer l'interprétabilité :

1. Développer et améliorer des approches intrinsèquement explicables, également connues sous le nom de modèles en boîte blanche, tels que les arbres de décision et les modèles de régression linéaire.
2. Fournir des explications post-hoc pour les modèles déjà formés, dits "boîte noire".

Une démarche commune à ces deux axes de recherche repose sur le développement de nouveaux outils tensoriels.

Les tenseurs sont un type de structure de données qui généralise les vecteurs et les matrices à des dimensions supérieures ou égales à trois. Ils sont devenus omniprésents dans l'apprentissage automatique moderne en raison de leurs capacités à conserver et à capturer une structure multidimensionnelle essentielle pour de multiples applications.

Les réseaux de neurones profonds joignent généralement des tenseurs d'ordre supérieur via des architectures telles que des couches convolutives, comme illustré sur la figure 1.2. En fait, la capacité des réseaux de neurones convolutifs profonds à préserver la structure locale de l'entrée est considérée comme une propriété cruciale pour les grandes performances obtenues [KPAP16].

L'analyse en composantes principales tensorielle (Tensor PCA) :

Le tenseur PCA a été introduit dans les travaux pionniers de [RM14] et consiste à récupérer un pic de signal $\boldsymbol{v}_0^{\otimes k}$ qui a été corrompu par un tenseur de bruit $\mathbf{Z}$: $\mathbf{T} = \mathbf{Z} + \beta \boldsymbol{v}_0^{\otimes k}$ où $\boldsymbol{v}_0$ est un vecteur unitaire et β le rapport signal sur bruit (SNR). La motivation de Tensor PCA est triple :

- Le tenseur PCA pourrait être considéré comme un simple cas de décomposition du tenseur. Cependant, il a une motivation différente qui est l'étude théorique des limitations de calcul dans le régime de très faible SNR alors que la littérature sur la décomposition du tenseur aborde principalement des applications pratiques, souvent dans un grand SNR. Pourtant, les algorithmes développés pour Tensor PCA pourraient être généralisés pour traiter la décomposition de Tensor comme dans [WA16].
- En plus de cela, Tensor PCA est également souvent utilisé comme problème d'inférence prototypique pour l'étude théorique de la difficulté de calcul de l'optimisation dans des paysages non convexes de grande dimension, en particulier en utilisant l'algorithme très répandu de descente de gradient et ses variantes ([BAGJ$^+$20, MKUZ19, MBC$^+$19, MBC$^+$20]). En effet, ces algorithmes sont utilisés avec un grand succès empirique dans de nombreux domaines du ML comme le Deep Learning, mais malheureusement ils sont généralement dépourvus de garanties théoriques. Comprendre la dynamique des méthodes de descente de gradient dans des paysages spécifiques tels que Tensor PCA pourrait apporter de nouvelles informations.
- L'une des principales caractéristiques de Tensor PCA est son écart algorithmique statistique conjecturé : alors que la théorie de l'information montre qu'il est théoriquement possible de récupérer le signal pour $\beta \sim O(1)$, tous les algorithmes existants

ont été montrés ou conjecturés avoir un seuil algorithmique pour $k \geq 3$ d'au moins $\beta \sim O(n^{(k-2)/4})$. Ainsi Tensor PCA est considéré comme un cas d'étude intéressant d'un tel écart qui apparaît dans divers autres problèmes (voir les références dans [BAGJ+20] et [LZ20]).

Première approche : Théorie des tenseurs aléatoires (RTT) pour la Tensor PCA

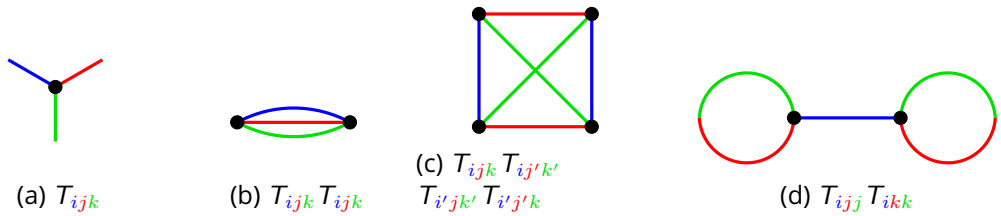
Nous avons d'abord proposé dans la première section de cette thèse un cadre théorique permettant récupérer le signal dans le modèle Tensor PCA en utilisant des invariants de trace. Mais d'abord, donnons l'idée générale du cadre proposé. Il est usuel dans le cas de la PCA matricielle d'utiliser la théorie spectrale associée aux matrices (valeurs propres et vecteurs propres) pour sa simplicité et ses excellentes performances. Cependant, la généralisation directe des valeurs propres et des vecteurs propres au cas du tenseur n'est pas pratique car leur nombre devient exponentiel et leur définition est ambigüe. Notre approche consiste donc à construire une matrice à partir du tenseur étudié, ce qui nous permettra d'exploiter la théorie spectrale bien maîtrisée des matrices afin de récupérer le signal. Cependant, il y a deux principales caractéristiques importantes que ces matrices doivent posséder pour qu'elles soient utiles : elles doivent être pertinentes, dans le sens où elles doivent révéler l'information/signal caché dans le tenseur même en régime de signal faible, et elles doivent également être faciles à étudier d'un point de vue probabiliste afin d'apporter des garanties théoriques. De manière pratique, notre cadre théorique basé sur la Théorie des tenseurs aléatoires (RTT) nous permet de sélectionner des matrices qui répondent à ces exigences. En effet, nous fournissons des matrices capables d'obtenir le signal dans le régime de bruit élevé, et nous avons accès à des outils combinatoires énumératifs simples afin d'avoir des garanties théoriques sur leurs performances.

La théorie des tenseurs aléatoires (RTT)

La théorie des tenseurs aléatoires (RTT) fournit un ensemble d'outils combinatoires dédiés à l'étude des graphes invariants de trace [Gur17]. Les invariants de trace d'un tenseur $\mathbf{T} \in \bigotimes_{i=1}^{k} \mathbb{R}^{n_i}$ sont des scalaires de réseaux de tenseurs qui sont invariants par les transformations $O(n_1) \times \cdots \times O(n_k)$:

$$\mathbf{T}_{i_1 \dots i_k} \longrightarrow \mathbf{T}'_{i_1 \dots i_k} = \sum_{j_1 \dots j_k} O^{(1)}_{i_1 j_1} \dots O^{(k)}_{i_k j_k} \mathbf{T}_{j_1 \dots j_k}$$

RTT permet d'obtenir des résultats probabilistes importants sur les invariants de trace en utilisant une combinatoire énumérative simple. En particulier, il donne un moyen simple de calculer les moments (espérance, variance, etc.) de la distribution de ces scalaires pour des tenseurs aléatoires. Ces invariants peuvent être schématisés par des graphes comme illustré dans la figure A.1.



(a) T_{ijk} (b) $T_{ijk}T_{ijk}$ (c) $T_{ijk}T_{ij'k'}$ $T_{i'jk'}T_{i'j'k}$ (d) $T_{ijj}T_{ikk}$

Figure A.1: Illustrations d'exemples d'invariants.

Matrices associées pour les traces d'invariants et nouveaux outils

Étant donné que notre objectif de Tensor PCA est de récupérer le signal, nous devrions trouver des objets mathématiques capables de fournir un vecteur. A cet effet, nous introduisons un nouvel ensemble d'outils sous forme de matrices. On note $\boldsymbol{M}_{\mathcal{G},e}$ la matrice obtenue en coupant une arête e d'un graphe $\mathcal{G}$ en deux demi-arêtes (voir Figure A.2 pour un exemple). Cette coupure revient à ne pas sommer sur les deux indices i_1 et i_2 associés à ces deux demi-arêtes et à les utiliser pour indexer la matrice à la place. Nous supprimerons l'indice $\mathcal{G}, e$ de la matrice lorsque le choix du graphe et de l'arête sera clair. Avantageusement, nous pouvons calculer les normes d'opérateur de ces matrices en utilisant les mêmes outils de RTT.

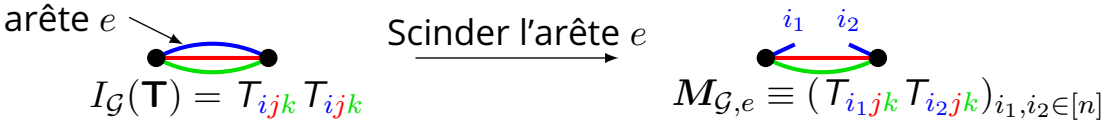


Figure A.2: Obtention d'une matrice à partir d'un invariant de trace shcématisé par un graphe $\mathcal{G}$.

En utilisant ces outils et cet algorithme, nous sommes maintenant en mesure d'étudier les performances de notre cadre dans divers contextes théoriques. Nous commençons par étudier les algorithmes associés à deux invariants de trace de degré 2. Ils sont constitués du diagramme melonique et du diagramme têtard. Fait intéressant, il s'avère qu'ils sont équivalents aux deux algorithmes de l'état de l'art pour Tensor PCA, le dépliement du tenseur et la méthode basée sur l'homotopie, comme illustré sur la figure 3.12. Ensuite, nous décidons d'aller plus loin en termes de degré de graphe et d'étudier les algorithmes associés aux graphes parfaits à une factorisation (constitués par le tétraèdre quand $k = 3$).

Les deux dernières sous-sections de cette section illustrent la versatilité de ce cadre théorique. Nous étudions le cas où les dimensions n_i du tenseur $\mathbf{T}$ ($\mathbf{T} \in \bigotimes_{i=1}^{k} \mathbb{R}^{n_i}$) ne sont pas nécessairement égales. Ce qui est important pour les applications pratiques où les dimensions sont naturellement asymétriques. Dans la dernière sous-section, nous prouvons que nos méthodes nous permettent de dériver un nouveau seuil algorithmique pour des cas plus généraux. Ce qui est à notre connaissance le premier à généraliser le seuil $n^{k/4}$ pour un tenseur $\mathbf{T} \in (\mathbb{R}^n)^{\otimes k}$. Ce cadre nous permet de dériver un nouveau seuil théorique $\max\left((\prod_{i=1}^{k} n_i)^{1/4}, n_i^{1/2}\right)$ pour des tenseurs de dimensions, qui sont très

courantes dans les applications réelles.

Seconde approche : Nouvel algorithme nommé SMPI

Dans ce chapitre, nous avons introduit un nouvel algorithme nommé Selective Multiple Power Iteration (SMPI) pour l'important problème Tensor PCA.

L'algorithme SMPI proposé (Algorithme 1) consiste à appliquer, en parallèle, la méthode d'itération de puissance avec m_{iter} itérations à m_{init} initialisations aléatoires différentes. Ensuite, SMPI utilise l'estimateur du maximum de vraisemblance pour sélectionner le vecteur de sortie dans ce sous-ensemble en choisissant le vecteur qui maximise $\mathbf{T}(\boldsymbol{v}, \boldsymbol{v}, \boldsymbol{v})$.

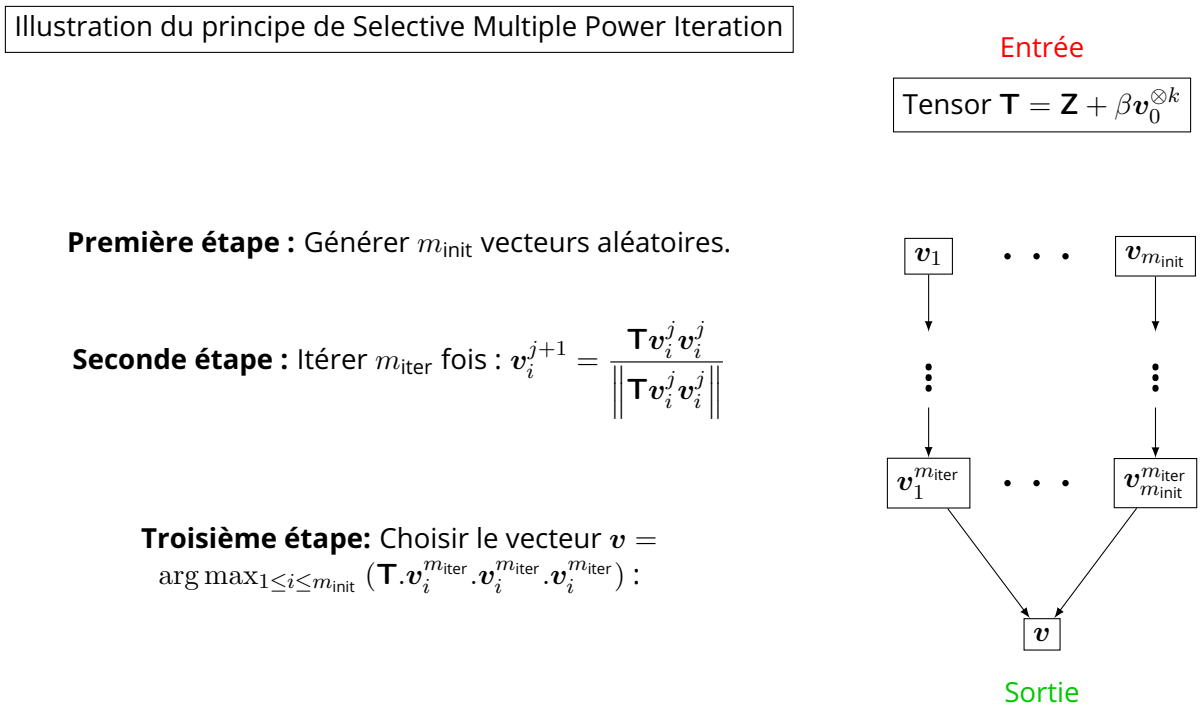


Figure A.3: Figure illustrative pour l'algorithme SMPI

Les propriétés essentielles de SMPI

Nous soulignons ici une différence importante et fondamentale entre notre algorithme et les algorithmes précédents basés sur l'itération de puissance. Pour que cette méthode réussisse dans le réglage SNR faible, nous avons besoin de ces cinq fonctionnalités que, à notre connaissance, nous sommes les premiers à imposer (voir Tableau A.1) :

1. Utilisation d'une itération de puissance ou d'une descente de gradient avec une taille de pas suffisamment grande.
2. Dans le cas d'une itération de puissance et d'un tenseur non symétrique, utiliser la version symétrisée (ou de manière équivalente, symétriser le tenseur).

3. Interdire un critère d'arrêt sur deux itérations consécutives (comme $1-|\langle \boldsymbol{v}_{i-1}, \boldsymbol{v}_i \rangle| < \varepsilon$ pour $\varepsilon > 0$) et à la place, utiliser un critères basés sur des itérations non consécutives distantes de $\Lambda : 1 - \langle \boldsymbol{v}_{i-\Lambda}, \boldsymbol{v}_i \rangle < \varepsilon$ et $\Lambda = O(n)$.

4. Utilisant au moins un nombre polynomial d'itérations.

5. Utilisant au moins un nombre polynomial d'initialisations.

Table A.1: Les cinq caractéristiques essentielles de SMPI par rapport aux travaux précédents portant sur l'itération de puissance

Papier	**Symétrie**	**Pas discret**	**Nbr. poly. d'initialisat.**	**Nbr. poly. d'iterations**	**Pas de stop condition**
Wang et al., 2017 [WDDFS17]	Oui	Oui	Non	Non	Non
Huang et al., 2020 [HHYC20]	Non	Oui	Oui	Oui	Oui
Ben Arous et al., 2020 [BAGJ$^+$20]	Oui	Non	Oui	Oui	Oui
Dudeja et al., 2022 [DH22]	Oui	Oui	Oui	Non	Oui
SMPI, 2021	Oui	Oui	Oui	Oui	Oui

Résultats

Diverses simulations numériques pour $k = 3$ dans la plage conventionnellement considérée $n \leq 1000$ montrent que les performances expérimentales de SMPI s'améliorent considérablement par rapport aux algorithmes existants et deviennent comparables à la récupération optimale théorique. Nous fournissons également de multiples variantes de cet algorithme pour aborder la décomposition du tenseur Canonical polyadic (CP) de bas rang. Ces algorithmes proposés surpassent également les méthodes existantes, même sur des données réelles, ce qui montre un très grand impact potentiel pour les applications pratiques.

De plus, nous présentons de nouveaux résultats théoriques sur le comportement des méthodes SMPI et de descente de gradient pour l'optimisation dans des paysages non convexes de grande dimension qui sont présents dans divers problèmes d'apprentissage automatique. Notons que, même si SMPI présente des résultats empiriques remarquables, nous ne donnons pas de garanties théoriques sur les performances de cet algorithme dans ce travail. Pour les applications critiques, cela peut être un inconvénient par rapport à certains algorithmes existants malgré leurs performances plus faibles (par exemple, les algorithmes du cadre théorique précedemment introduit ou Sum-of-Squares).

Perspectives

Applications

Une perspective importante est d'examiner plus en profondeur les applications potentielles et des exemples et modèles plus généraux où ces nouveaux outils pourraient être utiles. En effet, les applications réelles concernent des échelles de taille similaires à nos expériences où nous avons obtenu des résultats préliminaires très encourageants et une amélioration empirique substantielle par rapport à l'état de l'art a été observée. En effet, la décomposition du Tenseur a été utilisée pour un nombre important et varié de domaines. Cela pourrait couvrir par exemple la compression de réseaux de neurones, ou le débruitage d'images hyperspectrales. Il y a eu plusieurs travaux [AL17, WY+20, WGY21] sur la compression de réseaux de neurones qui aide l'interprétabilité et la vitesse du réseau de neurones en réduisant la taille de chaque couche de tenseur.

Il existe plusieurs méthodes qui étudient les algorithmes basés sur la déflation [dSCdA15a]. C'est un algorithme qui effectue une décomposition CP d'un rang élevé à partir d'un algorithme qui effectue une approximation de rang un. Les performances de ces méthodes peuvent différer selon que le rang est faible ou élevé.

Pour certaines applications, nous pouvons étudier l'adaptation de nos cadres théoriques et méthodes à d'autres types de données comme les tenseurs creux [NWZ20, Cd21].

Spin glass

Une autre direction de recherche concerne l'optimisation du verre sphérique p-spin, c'est-à-dire l'approximation de la valeur minimale la plus profonde en cherchant si une configuration quasi optimale peut être calculée en temps polynomial plutôt qu'exponentielle [EAMS21]. Les résultats préliminaires et l'observation empirique semblent suggérer que nous sommes capables de récupérer un nombre polynomial de vecteurs parmi lesquels le vecteur signal se trouve avant le seuil statistique tant que nous sommes au-dessus du seuil critique. Cependant, nous trouvons des minima plus profonds, donc pour Tensor PCA, l'estimateur de similarité maximale ne pourra pas indiquer quel vecteur est le signal. Mais à des fins d'optimisation, il pourrait être très intéressant d'étudier des résultats aussi intrigants.

Combinaison de graphes

On pourrait se demander quelle est la meilleure classe de graphes. Une autre idée consiste à rechercher si une somme de graphiques fournit de meilleurs résultats. Nos premières expériences sont très prometteuses. Ceci est basé sur le simple fait qu'en sommant deux variables différentes, la variance de leur somme est inférieure à la somme de leur variance. La variance de la matrice de bruit diminue, mais la matrice de signal pur s'ajoute telle quelle. Des résultats empiriques sur la somme de deux matrices confirment cette intuition. Ceci est illustré dans la figure 5.1

Théorie des tenseurs aléatoires pour SMPI

Oleg Evnin [Evn20] a commencé une étude des itérations de puissance basée sur la théorie des tenseurs aléatoires et les invariants de trace. Comme nous l'avons expliqué au chapitre 2, cela revient à l'étude du graphe associé illustré sur la figure A.4. En se basant sur l'hypothèse de 4.6, l'objectif est de prouver que l'espérance de ce graphe est non nulle au premier ordre de $1/N$. Compte tenu de nos résultats empiriques, il est intéressant de poursuivre cette investigation en considérant un nombre polynomial d'itérations. Cela pourrait être fait en adaptant les outils utilisés par Evnin et en examinant le taux de croissance d'un vecteur initial aléatoire sous des applications successives d'une carte non linéaire définie par le tenseur aléatoire.

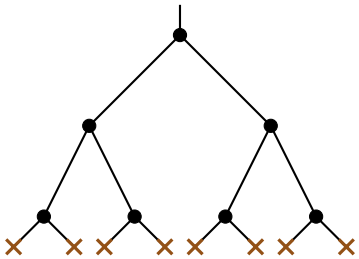


Figure A.4: Le graphe associé à la méthode d'itération puissance à 3 itérations pour une initialisation $\boldsymbol{v}$. La croix représente le vecteur $\boldsymbol{v}$ et le point noir le tenseur $\mathbf{T}$

Etude sur la taille de pas discrets pour SMPI

[BAGJ22] a proposé une nouvelle approche pour étudier les statistiques associées à la descente de gradient stochastique (SGD) en fonction de l'initialisation et de la taille du pas discret. Ils ont découvert un nouveau seuil pour la taille du pas au-dessus duquel le comportement de SGD change fondamentalement. L'adaptation de cette approche pour SMPI pourrait être extrêmement utile pour la compréhension théorique de ses performances et de l'importance de la taille de pas variable discrète dans SMPI. D'autre part, l'obtention d'une compréhension théorique de SMPI grâce à cette approche peut apporter des informations importantes sur les performances de SGD dans le cadre de l'apprentissage en profondeur.

Appendix B

Appendix Chapter Random Tensor

B.1 Gaussian expectation of trace invariants

Let's consider a graph $\mathcal{G}$ of order d associated to a trace invariant $I_{\mathcal{G}}(\mathbf{T})$ where $\mathbf{T}$ is a tensor of order k. Since the trace invariant is a contraction of pairs of indices of d copies of $\mathbf{T}$, we denote the c-th index of the i-th copy of $\mathbf{T}$ by ${a_i}^c$ and the set of the d indices of the i-th copy by ${a_j}^{\mathcal{D}}$. By the definition of the Gaussian measure we have the equation:

$$\mathbb{E}(\mathrm{Tr}_{\mathcal{G}}(\mathbf{T})) = \sum_a \delta^{\mathcal{G}}_{aa} \sum_{\tau \in \mathfrak{S}(k)} \prod_{j=1}^{k} \delta_{{a_j}^{\mathcal{D}} {a_j}_{\tau(j)}^{\mathcal{D}}}, \tag{B.1}$$

where $\mathfrak{S}(k)$ is the symmetric group (the group of all permutations of a set made of n elements, where we define a permutation of a set by an arrangement of its elements in a linear order. For example the permutations of $\{1,2,3\}$ are $(1,2,3),(1,3,2),(2,1,3),$ $(2,3,1),(3,1,2),(3,2,1)$.).

$\sum_a$ indicates a sum over all the indices involved in the computation of the trace invariant, illustrated by a half-edge in the graph (we have $k \times d$ indices in total: k indices for each one of the d copies of $\mathbf{T}$ of the trace invariant expression). $\delta^{\mathcal{G}}_{aa}$ indicates that contracted indices (illustrated by being two ends of the same edge) of the tensors have to be equal. The term $\delta_{{a_j}^{\mathcal{D}} {a_j}_{\tau(j)}^{\mathcal{D}}}$ indicates that we set equal the c-th indices of the copy j and $\tau(j)$ of $\mathbf{T}$ for $c \in \{1,\dots,d\}$

In order to have a clearer representation of the expression in equation B.1, in particular the sum over the symmetric group, we introduce the concept of covering graph used in [Gur17]: a covering graph of $\mathcal{G}$ consists in adding $d/2$ new edges of color 0 (also called propagators) relying pairwise the vertices of $\mathcal{G}$ (Figure 3.2 for an example). We use the covering graphs to illustrate the permutations $\tau \in \mathfrak{S}(k)$. Then, an edge of color 0 between two vertices i and j will identify all the indices of the i-th and j-th copies of $\mathbf{T}$. Thus, it will be equivalent to the term $\delta_{{a_j}^{\mathcal{D}} {a_j}_{\tau(j)}^{\mathcal{D}}}$. Using the covering graph, we will be able to rewrite equation B.1.

For example, in the case where the edges could have only one color, we have the formula:

$$\mathbb{E}(\mathrm{Tr}_{\mathcal{G}}(\mathbf{T})) = \sum_{\mathcal{G}',\mathcal{G}'\backslash\mathcal{E}^0(\mathcal{G}')=\mathcal{G}} \sum_{aa} \left(\prod_{c=1}^{3} \prod_{e^c=(v_j,v_k)\in\mathcal{E}^c(\mathcal{G})} \delta_{a_j{}^c,a_k{}^c} \right) \left(\prod_{e^0=(v_p,v_q)\in\mathcal{E}^0(\mathcal{G})} \delta_{a_p{}^1,a_q{}^1}\delta_{a_p{}^2,a_q{}^2}\delta_{a_p{}^3,a_q{}^3} \right), \tag{B.2}$$

Where we denote $\mathcal{E}^c(\mathcal{G})$ the set of edges of color c of $\mathcal{G}$. Thus $\{\mathcal{G}', \mathcal{G}'\backslash\mathcal{E}^0(\mathcal{G}') = \mathcal{G}\}$ denotes the graphs which restrict to the graph $\mathcal{G}$ when we remove their edges of color 0, which are by the definition the covering graphs.

Let $c_1, c_2 \in \{0, \dots, d\}$ be two different colors of edges. We denote $F^{c_1,c_2}(\mathcal{G})$ the number of closed cycles (that we also call face) of 2 colors of $\mathcal{G}$. More explicitly, it consists of the number of connected subgraphs left when we keep in $\mathcal{G}$ only the edges of colors c_1, c_2. We denote $F(\mathcal{G}) \equiv \sum_{c_1\neq c_2\in\{1,\dots,d\}} F^{c_1,c_2}(\mathcal{G})$.

Then, with a little work (see [Gur14] for a proof), we obtain :

$$\mathbb{E}(\mathrm{Tr}_{\mathcal{G}}(\mathbf{T})) = \sum_{\mathcal{G}',\mathcal{G}'\backslash\mathcal{E}^0(\mathcal{G}')=\mathcal{G}} n^{F(\mathcal{G}')-F(\mathcal{G})} = \sum_{\mathcal{G}',\mathcal{G}'\backslash\mathcal{E}^0(\mathcal{G}')=\mathcal{G}} n^{\sum_c F^{0,c}(\mathcal{G}')}. \tag{B.3}$$

This will be the expression we will use to calculate the expectations of our graphs if each edges has only one color.

B.2 Useful theorems.

The following theorems will be useful for the demonstrations of the proposed theorems.

- **Wedin perturbation** ([Wed72]): Let $\boldsymbol{M} = \beta \boldsymbol{v}_0\boldsymbol{v}_0^\top + \boldsymbol{N}$ a square matrix with $\|\boldsymbol{v}_0\| = 1$. Let $\boldsymbol{v}$ denote the eigenvector with the largest eigenvalue of $\boldsymbol{M}$. If $\beta > 2\|\boldsymbol{N}\|_{\mathsf{op}}$ then:

$$1 - \langle \boldsymbol{v}, \boldsymbol{v}_0\rangle \leq \frac{8\|\boldsymbol{N}\|_{\mathsf{op}}}{\beta^2}. \tag{B.4}$$

- **Chebyshev inequality** ([Ros14]): Let X be a random variable with finite expected value μ and finite non-zero variance σ^2. Then for any real number $C > 0$,

$$\mathsf{Pr}(|X - \mu| \geq C\sigma) \leq \frac{1}{C^2}. \tag{B.5}$$

  where Pr denotes the probability.

- **Gelfand's Formula** ([Gel41]): For any matrix norm $\|.\|$, we have:

$$\rho(\boldsymbol{A}) = \lim_{s\to\infty} \|\boldsymbol{A}^s\|^{\frac{1}{s}}. \tag{B.6}$$

where $\rho(\boldsymbol{A})$ is the spectral radius of $\boldsymbol{A}$: the largest absolute value of its eigenvalues.

In particular, if $\boldsymbol{A}$ is a real matrix, we can choose the Frobenius norm:

$$\|\boldsymbol{A}\|_{\mathsf{F}} = \sqrt{\mathrm{trace}(\boldsymbol{A}^{\top}\boldsymbol{A})}. \tag{B.7}$$

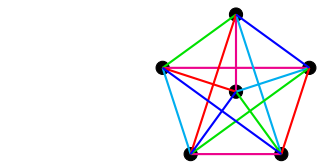
B.3 The perfect one-factorization graph.

In the mathematical field of graph theory, a graph $\mathcal{G}$ is said to be one-factorizable ([Wag92]) if:

- It is k-regular, which means that each vertex is incident to exactly k edges.
- Its edges can be colored in k colors such that no two edges incident to the same vertex have the same color.

A perfect one-factorization of $\mathcal{G}$ happens if, for every two colors c_1 and c_2 of the k colors, restricting $\mathcal{G}$ to the edges of color c_1 and c_2 and removing all the edges colored differently, leads to a single connected cycle. In other words, the number of the faces associated to the colors c_1 and c_2 is always equal to one: $F^{c_1,c_2}(\mathcal{G}) = 1\ \forall c_1, c_2$. We illustrate this property in Figure B.1 where we see that restricting to the red and green colors, or the red and blue colors for a complete graph for $k = 5$, leads to a single connected cycle.

It is conjectured ([Wag92]) that they exist for all n but, until now, they were proven only for prime numbers and up to 56. We use this particular set of graphs because of its combinatorial simplicity. The perfect one-factorization graphs have already been studied in the context of random tensors in [FRV19a].



(a) Complete graph for $k = 5$

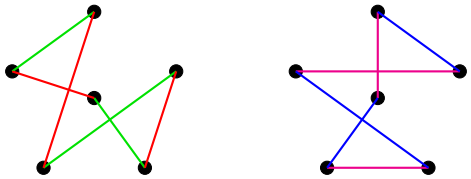


Figure B.1: Complete graph for $k = 5$ and some two colors restriction

Bibliography

[AAJM+20] Mohamed Alloghani, Dhiya Al-Jumeily, Jamila Mustafina, Abir Hussain, and Ahmed J Aljaaf. A systematic review on supervised and unsupervised machine learning algorithms for data science. *Supervised and unsupervised learning for data science*, pages 3–21, 2020.

[ABGD20] Remi C. Avohou, Joseph Ben Geloun, and Nicolas Dub. On the counting of $O(N)$ tensor invariants. *Adv. Theor. Math. Phys.*, 24(4):821–878, 2020.

[ACO08] Dimitris Achlioptas and Amin Coja-Oghlan. Algorithmic barriers from phase transitions. In *2008 49th Annual IEEE Symposium on Foundations of Computer Science*, pages 793–802. IEEE, 2008.

[ADGM17a] Anima Anandkumar, Yuan Deng, Rong Ge, and Hossein Mobahi. Homotopy analysis for tensor pca. In Satyen Kale and Ohad Shamir, editors, *Proceedings of the 2017 Conference on Learning Theory*, volume 65 of *Proceedings of Machine Learning Research*, pages 79–104. PMLR, 07–10 Jul 2017.

[ADGM17b] Anima Anandkumar, Yuan Deng, Rong Ge, and Hossein Mobahi. Homotopy analysis for tensor pca. In *Conference on Learning Theory*, pages 79–104. PMLR, 2017.

[AGH+12] Anima Anandkumar, Rong Ge, Daniel Hsu, Sham M Kakade, and Matus Telgarsky. Tensor decompositions for learning latent variable models. *arXiv preprint arXiv:1210.7559*, 2012.

[AGH+14] Animashree Anandkumar, Rong Ge, Daniel Hsu, Sham M Kakade, and Matus Telgarsky. Tensor decompositions for learning latent variable models. *Journal of Machine Learning Research*, 15:2773–2832, 2014.

[AGHK13] Animashree Anandkumar, Rong Ge, Daniel Hsu, and Sham Kakade. A tensor spectral approach to learning mixed membership community models. In *Conference on Learning Theory*, pages 867–881. PMLR, 2013.

[AGJ15] Animashree Anandkumar, Rong Ge, and Majid Janzamin. Learning overcomplete latent variable models through tensor methods. In *Conference on Learning Theory*, pages 36–112. PMLR, 2015.

[AHK12] Animashree Anandkumar, Daniel Hsu, and Sham M Kakade. A method of moments for mixture models and hidden markov models. In *Conference on Learning Theory*, pages 33–1. JMLR Workshop and Conference Proceedings, 2012.

[AL17] Marcella Astrid and Seung-Ik Lee. Cp-decomposition with tensor power method for convolutional neural networks compression. In *2017 IEEE International Conference on Big Data and Smart Computing (BigComp)*, pages 115–118. IEEE, 2017.

[AV+07] Hervé Abdi, Dominique Valentin, et al. Multiple factor analysis (mfa). *Encyclopedia of measurement and statistics*, pages 657–663, 2007.

[BA03] Gérard Ben-Arous. Aging and spin-glass dynamics. *arXiv preprint math/0304364*, 2003.

[BAGJ+20] Gerard Ben Arous, Reza Gheissari, Aukosh Jagannath, et al. Algorithmic thresholds for tensor pca. *Annals of Probability*, 48(4):2052–2087, 2020.

[BAGJ21] Gerard Ben Arous, Reza Gheissari, and Aukosh Jagannath. Online stochastic gradient descent on non-convex losses from high-dimensional inference. *J. Mach. Learn. Res.*, 22:106–1, 2021.

[BAGJ22] Gerard Ben Arous, Reza Gheissari, and Aukosh Jagannath. High-dimensional limit theorems for sgd: Effective dynamics and critical scaling. *arXiv preprint arXiv:2206.04030*, 2022.

[BAMMN19] Gerard Ben Arous, Song Mei, Andrea Montanari, and Mihai Nica. The landscape of the spiked tensor model. *Communications on Pure and Applied Mathematics*, 72(11):2282–2330, 2019.

[BB20] Matthew Brennan and Guy Bresler. Reducibility and statistical-computational gaps from secret leakage. In *Conference on Learning Theory*, pages 648–847. PMLR, 2020.

[BBAP05] Jinho Baik, Gérard Ben Arous, and Sandrine Péché. Phase transition of the largest eigenvalue for nonnull complex sample covariance matrices. *The Annals of Probability*, 33(5):1643–1697, 2005.

[BCRT20] Giulio Biroli, Chiara Cammarota, and Federico Ricci-Tersenghi. How to iron out rough landscapes and get optimal performances: averaged gradient descent and its application to tensor pca. *Journal of Physics A: Mathematical and Theoretical*, 53(17):174003, 2020.

[BHK+19] Boaz Barak, Samuel Hopkins, Jonathan Kelner, Pravesh K Kothari, Ankur Moitra, and Aaron Potechin. A nearly tight sum-of-squares lower bound for the planted clique problem. *SIAM Journal on Computing*, 48(2):687–735, 2019.

[BKW20] Afonso S Bandeira, Dmitriy Kunisky, and Alexander S Wein. Average-case integrality gap for non-negative principal component analysis. *arXiv preprint arXiv:2012.02243*, 2020.

[BLHG10] Franziska Bell, Daniel S Lambrecht, and Martin Head-Gordon. Higher order singular value decomposition in quantum chemistry. *Molecular Physics*, 108(19-20):2759–2773, 2010.

[BM11] Mohsen Bayati and Andrea Montanari. The dynamics of message passing on dense graphs, with applications to compressed sensing. *IEEE Transactions on Information Theory*, 57(2):764–785, 2011.

[BN06] Christopher M Bishop and Nasser M Nasrabadi. *Pattern recognition and machine learning*, volume 4. Springer, 2006.

[BR13] Quentin Berthet and Philippe Rigollet. Computational lower bounds for sparse pca. *arXiv preprint arXiv:1304.0828*, 2013.

[BS05] Christian F Beckmann and Stephen M Smith. Tensorial extensions of independent component analysis for multisubject fmri analysis. *Neuroimage*, 25(1):294–311, 2005.

[BSBE+16] David Bruns-Smith, Muthu M Baskaran, James Ezick, Tom Henretty, and Richard Lethin. Cyber security through multidimensional data decompositions. In *2016 Cybersecurity Symposium (CYBERSEC)*, pages 59–67. IEEE, 2016.

[CAVP21] Hongyang Chen, Fauzia Ahmad, Sergiy Vorobyov, and Fatih Porikli. Tensor decompositions in wireless communications and mimo radar. *IEEE Journal of Selected Topics in Signal Processing*, 15(3):438–453, 2021.

[CCDS97] Pierre Cazes, Ahlame Chouakria, Edwin Diday, and Yves Schektman. Extension de l'analyse en composantes principales à des données de type intervalle. *Revue de Statistique appliquée*, 45(3):5–24, 1997.

[Cd21] Davin Choo and Tommaso d'Orsi. The complexity of sparse tensor pca. *Advances in Neural Information Processing Systems*, 34:7993–8005, 2021.

[CHM+15] Anna Choromanska, Mikael Henaff, Michael Mathieu, Gérard Ben Arous, and Yann LeCun. The loss surfaces of multilayer networks. In *Artificial intelligence and statistics*, pages 192–204. PMLR, 2015.

[CKR94] LF Cugliandolo, J Kurchan, and F Ritort. Evidence of aging in spin-glass mean-field models. *Physical Review B*, 49(9):6331, 1994.

[CLDA09] Pierre Comon, Xavier Luciani, and André LF De Almeida. Tensor decompositions, alternating least squares and other tales. *Journal of Chemometrics: A Journal of the Chemometrics Society*, 23(7-8):393–405, 2009.

[CLMW11] Emmanuel J Candès, Xiaodong Li, Yi Ma, and John Wright. Robust principal component analysis? *Journal of the ACM (JACM)*, 58(3):1–37, 2011.

[CS13] Dustin Cartwright and Bernd Sturmfels. The number of eigenvalues of a tensor. *Linear algebra and its applications*, 438(2):942–952, 2013.

[DH21] Rishabh Dudeja and Daniel Hsu. Statistical query lower bounds for tensor pca. *Journal of Machine Learning Research*, 22(83):1–51, 2021.

[DH22] Rishabh Dudeja and Daniel Hsu. Statistical-computational trade-offs in tensor pca and related problems via communication complexity. *arXiv preprint arXiv:2204.07526*, 2022.

[DJPM18] Xiaowu Deng, Peng Jiang, Xiaoning Peng, and Chunqiao Mi. An intelligent outlier detection method with one class support tucker machine and genetic algorithm toward big sensor data in internet of things. *IEEE Transactions on Industrial Electronics*, 66(6):4672–4683, 2018.

[DKWB19] Yunzi Ding, Dmitriy Kunisky, Alexander S Wein, and Afonso S Bandeira. Subexponential-time algorithms for sparse pca. *arXiv preprint arXiv:1907.11635*, 2019.

[DLDMV00a] Lieven De Lathauwer, Bart De Moor, and Joos Vandewalle. A multilinear singular value decomposition. *SIAM journal on Matrix Analysis and Applications*, 21(4):1253–1278, 2000.

[DLDMV00b] Lieven De Lathauwer, Bart De Moor, and Joos Vandewalle. On the best rank-1 and rank-(r 1, r 2,..., rn) approximation of higher-order tensors. *SIAM journal on Matrix Analysis and Applications*, 21(4):1324–1342, 2000.

[DLG20] Alexis Decurninge, Ingmar Land, and Maxime Guillaud. Tensor-based modulation for unsourced massive random access. *IEEE Wireless Communications Letters*, 2020.

[DMM09] David L Donoho, Arian Maleki, and Andrea Montanari. Message-passing algorithms for compressed sensing. *Proceedings of the National Academy of Sciences*, 106(45):18914–18919, 2009.

[dSCdA15a] Alex P da Silva, Pierre Comon, and André LF de Almeida. An iterative deflation algorithm for exact cp tensor decomposition. In *2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 3961–3965. IEEE, 2015.

[dSCdA15b] Alex Pereira da Silva, Pierre Comon, and Andre Lima Ferrer de Almeida. Rank-1 tensor approximation methods and application to deflation. *arXiv preprint arXiv:1508.05273*, 2015.

[EA75] Samuel Frederick Edwards and Phil W Anderson. Theory of spin glasses. *Journal of Physics F: Metal Physics*, 5(5):965, 1975.

[EAMS21] Ahmed El Alaoui, Andrea Montanari, and Mark Sellke. Optimization of mean-field spin glasses. *The Annals of Probability*, 49(6):2922–2960, 2021.

[Evn20] Oleg Evnin. Melonic dominance and the largest eigenvalue of a large random tensor. *arXiv preprint arXiv:2003.11220*, 2020.

[FH08] Yun Fu and Thomas S Huang. Image classification using correlation tensor analysis. *IEEE Transactions on Image Processing*, 17(2):226–234, 2008.

[FJY+19] Xin Feng, Youni Jiang, Xuejiao Yang, Ming Du, and Xin Li. Computer vision algorithms and hardware implementations: A survey. *Integration*, 69:309–320, 2019.

[FRV19a] Frank Ferrari, Vincent Rivasseau, and Guillaume Valette. A New Large N Expansion for General Matrix–Tensor Models. *Commun. Math. Phys.*, 370(2):403–448, 2019.

[FRV19b] Frank Ferrari, Vincent Rivasseau, and Guillaume Valette. A new large n expansion for general matrix–tensor models. *Communications in Mathematical Physics*, 370(2):403–448, 2019.

[Gam21] David Gamarnik. The overlap gap property: A topological barrier to optimizing over random structures. *Proceedings of the National Academy of Sciences*, 118(41):e2108492118, 2021.

[GB06] Michael Greenacre and Jorg Blasius. *Multiple correspondence analysis and related methods*. Chapman and Hall/CRC, 2006.

[GBC16] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep learning*. MIT press, 2016.

[Gel41] Izrail Gelfand. Normierte ringe. *Matematicheskii Sbornik*, 9(1):3–24, 1941.

[GME11] Ben Gold, Nelson Morgan, and Dan Ellis. *Speech and audio signal processing: processing and perception of speech and music*. John Wiley & Sons, 2011.

[Gur14] Razvan Gurau. Universality for Random Tensors. *Ann. Inst. H. Poincare Probab. Statist.*, 50(4):1474–1525, 2014.

[Gur17] R. Gurau. *Random Tensors*. Oxford University Press, 2017.

[Gur20] Razvan Gurau. On the generalization of the wigner semicircle law to real symmetric tensors. *arXiv preprint arXiv:2004.02660*, 2020.

[Has20] Matthew B. Hastings. Classical and Quantum Algorithms for Tensor Principal Component Analysis. *Quantum*, 4:237, February 2020.

[HHYC20] Jiaoyang Huang, Daniel Z Huang, Qing Yang, and Guang Cheng. Power iteration for tensor pca. *arXiv preprint arXiv:2012.13669*, 2020.

[HKP+17] Samuel B Hopkins, Pravesh K Kothari, Aaron Potechin, Prasad Raghavendra, Tselil Schramm, and David Steurer. The power of sum-of-squares for detecting hidden structures. In *2017 IEEE 58th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 720–731. IEEE, 2017.

[HKZ12] Daniel Hsu, Sham M Kakade, and Tong Zhang. A spectral algorithm for learning hidden markov models. *Journal of Computer and System Sciences*, 78(5):1460–1480, 2012.

[HL13] Christopher J Hillar and Lek-Heng Lim. Most tensor problems are np-hard. *Journal of the ACM (JACM)*, 60(6):1–39, 2013.

[HMvdG18] Jianyu Huang, Devin A Matthews, and Robert A van de Geijn. Strassen's algorithm for tensor contraction. *SIAM Journal on Scientific Computing*, 40(3):C305–C326, 2018.

[Hop18] Samuel Hopkins. *Statistical inference and the sum of squares method*. PhD thesis, Cornell University, 2018.

[HSSS16] Samuel B Hopkins, Tselil Schramm, Jonathan Shi, and David Steurer. Fast spectral algorithms from sum-of-squares proofs: tensor decomposition and planted sparse vectors. In *Proceedings of the forty-eighth annual ACM symposium on Theory of Computing*, pages 178–191, 2016.

[HTG17] Felix Hummel, Theodoros Tsatsoulis, and Andreas Grüneis. Low rank factorization of the coulomb integrals for periodic coupled cluster theory. *The Journal of chemical physics*, 146(12):124105, 2017.

[Jer92] Mark Jerrum. Large cliques elude the metropolis process. *Random Structures & Algorithms*, 3(4):347–359, 1992.

[JLM+20] Aukosh Jagannath, Patrick Lopatto, Leo Miolane, et al. Statistical thresholds for tensor pca. *Annals of Applied Probability*, 30(4):1910–1933, 2020.

[KB09] Tamara G Kolda and Brett W Bader. Tensor decompositions and applications. *SIAM review*, 51(3):455–500, 2009.

[KPAP16] Jean Kossaifi, Yannis Panagakis, Anima Anandkumar, and Maja Pantic. Tensorly: Tensor learning in python. *arXiv preprint arXiv:1610.09555*, 2016.

[KPAP19] Jean Kossaifi, Yannis Panagakis, Anima Anandkumar, and Maja Pantic. Tensorly: Tensor learning in python. *Journal of Machine Learning Research*, 20(26):1–6, 2019.

[KSRH+18] Jinsung Kim, Aravind Sukumaran-Rajam, Changwan Hong, Ajay Panyala, Rohit Kumar Srivastava, Sriram Krishnamoorthy, and Ponnuswamy Sadayappan. Optimizing tensor contractions in ccsd (t) for efficient execution on gpus. In *Proceedings of the 2018 International Conference on Supercomputing*, pages 96–106, 2018.

[KWB19] Dmitriy Kunisky, Alexander S Wein, and Afonso S Bandeira. Notes on computational hardness of hypothesis testing: Predictions using the low-degree likelihood ratio. *arXiv preprint arXiv:1907.11636*, 2019.

[LAJ15] Dana Lahat, Tülay Adali, and Christian Jutten. Multimodal data fusion: an overview of methods, challenges, and prospects. *Proceedings of the IEEE*, 103(9):1449–1477, 2015.

[LB13] Tao Lin and Salah Bourennane. Survey of hyperspectral image denoising methods based on tensor decompositions. *EURASIP journal on Advances in Signal Processing*, 2013(1):1–11, 2013.

[LBF12] Xuefeng Liu, Salah Bourennane, and Caroline Fossati. Denoising of hyperspectral images using the parafac model and statistical performance analysis. *IEEE Transactions on Geoscience and Remote Sensing*, 50(10):3717–3724, 2012.

[LHK05] Kuang-Chih Lee, Jeffrey Ho, and David J Kriegman. Acquiring linear subspaces for face recognition under variable lighting. *IEEE Transactions on pattern analysis and machine intelligence*, 27(5):684–698, 2005.

[LML+17] Thibault Lesieur, Léo Miolane, Marc Lelarge, Florent Krzakala, and Lenka Zdeborová. Statistical and computational phase transitions in spiked tensor estimation. In *2017 IEEE International Symposium on Information Theory (ISIT)*, pages 511–515. IEEE, 2017.

[LOST21] Vincent Lahoche, Mohamed Ouerfelli, Dine Ousmane Samary, and Mohamed Tamaazousti. Field theoretical approach for signal detection in nearly continuous positive spectra ii: Tensorial data. *Entropy*, 23(7):795, 2021.

[LZ20] Yuetian Luo and Anru R Zhang. Open problem: Average-case hardness of hypergraphic planted clique detection. In *Conference on Learning Theory*, pages 3852–3856. PMLR, 2020.

[Mah20] Batta Mahesh. Machine learning algorithms-a review. *International Journal of Science and Research (IJSR).[Internet]*, 9:381–386, 2020.

[Mat18] Devin A Matthews. High-performance tensor contraction without transposition. *SIAM Journal on Scientific Computing*, 40(1):C1–C24, 2018.

[MB05] Damien Muti and Salah Bourennane. Multidimensional filtering based on a tensor approach. *Signal Processing*, 85(12):2338–2353, 2005.

[MBC+19] Stefano Sarao Mannelli, Giulio Biroli, Chiara Cammarota, Florent Krzakala, and Lenka Zdeborová. Who is afraid of big bad minima? analysis of gradient-flow in a spiked matrix-tensor model. *arXiv preprint arXiv:1907.08226*, 2019.

[MBC+20] Stefano Sarao Mannelli, Giulio Biroli, Chiara Cammarota, Florent Krzakala, Pierfrancesco Urbani, and Lenka Zdeborová. Marvels and pitfalls of the langevin algorithm in noisy high-dimensional inference. *Physical Review X*, 10(1):011057, 2020.

[MDDN18] Konstantinos Makantasis, Anastasios D Doulamis, Nikolaos D Doulamis, and Antonis Nikitakis. Tensor-based classification models for hyperspectral data analysis. *IEEE Transactions on Geoscience and Remote Sensing*, 56(12):6884–6898, 2018.

[MGT+18] CE Miller, RO Green, DR Thompson, AK Thorpe, M Eastwood, IB Mccubbin, W Olson-Duvall, M Bernas, CM Sarture, S Nolte, et al. Above: Hyperspectral imagery from aviris-ng, alaskan and canadian arctic, 2017–2018. *ORNL DAAC*, 2018.

[MKS+15] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A Rusu, Joel Veness, Marc G Bellemare, Alex Graves, Martin Riedmiller, Andreas K Fidjeland, Georg Ostrovski, et al. Human-level control through deep reinforcement learning. *nature*, 518(7540):529–533, 2015.

[MKUZ19] Stefano Sarao Mannelli, Florent Krzakala, Pierfrancesco Urbani, and Lenka Zdeborova. Passed & spurious: Descent algorithms and local minima in spiked matrix-tensor models. In *international conference on machine learning*, pages 4333–4342. PMLR, 2019.

[MKV+13] Wenjing Ma, Sriram Krishnamoorthy, Oreste Villa, Karol Kowalski, and Gagan Agrawal. Optimizing tensor contraction expressions for hybrid cpu-gpu execution. *Cluster computing*, 16(1):131–155, 2013.

[MR15] Andrea Montanari and Emile Richard. Non-negative principal component analysis: Message passing algorithms and sharp asymptotics. *IEEE Transactions on Information Theory*, 62(3):1458–1484, 2015.

[Nas13] Nasser M Nasrabadi. Hyperspectral target detection: An overview of current and future challenges. *IEEE Signal Processing Magazine*, 31(1):34–44, 2013.

[NLK+20] Kristina Naskovska, Stephan Lau, Alexey A Korobkov, Jens Haueisen, and Martin Haardt. Coupled cp decomposition of simultaneous meg-eeg signals for differentiating oscillators during photic driving. *Frontiers in neuroscience*, 14:261, 2020.

[NWZ20] Jonathan Niles-Weed and Ilias Zadik. The all-or-nothing phenomenon in sparse tensor pca. *Advances in Neural Information Processing Systems*, 33:17674–17684, 2020.

[OTR22] Mohamed Ouerfelli, Mohamed Tamaazousti, and Vincent Rivasseau. Random tensor theory for tensor decomposition. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 2022.

[Pea01] Karl Pearson. Liii. on lines and planes of closest fit to systems of points in space. *The London, Edinburgh, and Dublin philosophical magazine and journal of science*, 2(11):559–572, 1901.

[PFS16] Evangelos E Papalexakis, Christos Faloutsos, and Nicholas D Sidiropoulos. Tensors for data mining and data fusion: Models, applications, and scalable algorithms. *ACM Transactions on Intelligent Systems and Technology (TIST)*, 8(2):1–44, 2016.

[PKYA21] Taylor L Patti, Jean Kossaifi, Susanne F Yelin, and Anima Anandkumar. Tensorly-quantum: Quantum machine learning with tensor methods. *arXiv preprint arXiv:2112.10239*, 2021.

[Pom97] Jean-Charles Pomerol. Artificial intelligence and human decision making. *European Journal of Operational Research*, 99(1):3–25, 1997.

[PTC15] Anh-Huy Phan, Petr Tichavskỳ, and Andrzej Cichocki. Tensor deflation for candecomp/parafac—part i: Alternating subspace update algorithm. *IEEE Transactions on signal Processing*, 63(22):5924–5938, 2015.

[PWB+20] Amelia Perry, Alexander S Wein, Afonso S Bandeira, et al. Statistical limits of spiked tensor models. In *Annales de l'Institut Henri Poincaré, Probabilités et Statistiques*, volume 56, pages 230–264. Institut Henri Poincaré, 2020.

[Qi05] Liqun Qi. Eigenvalues of a real supersymmetric tensor. *Journal of Symbolic Computation*, 40(6):1302 – 1324, 2005.

[RBABC19] Valentina Ros, Gerard Ben Arous, Giulio Biroli, and Chiara Cammarota. Complex energy landscapes in spiked-tensor and simple glassy models: Ruggedness, arrangements of local minima, and phase transitions. *Physical Review X*, 9(1):011003, 2019.

[RD91] James O Ramsay and CJ1125714 Dalzell. Some tools for functional data analysis. *Journal of the Royal Statistical Society: Series B (Methodological)*, 53(3):539–561, 1991.

[RM14] Emile Richard and Andrea Montanari. A statistical model for tensor pca. In *Advances in Neural Information Processing Systems*, pages 2897–2905, 2014.

[Ros14] Sheldon Ross. *A first course in probability*. Pearson, 2014.

[SB20] Weike Sun and Richard D Braatz. Opportunities in tensorial data analytics for chemical and biological manufacturing processes. *Computers & Chemical Engineering*, page 107099, 2020.

[ŠBP21] Boštjan Šumak, Saša Brdnik, and Maja Pušnik. Sensors and artificial intelligence methods and algorithms for human–computer intelligent interaction: A systematic mapping study. *Sensors*, 22(1):20, 2021.

[SCZL18] Qiquan Shi, Yiu-Ming Cheung, Qibin Zhao, and Haiping Lu. Feature extraction for incomplete data via low-rank tensor decomposition with feature regularization. *IEEE transactions on neural networks and learning systems*, 30(6):1803–1817, 2018.

[Tur09] Alan M Turing. Computing machinery and intelligence. In *Parsing the turing test*, pages 23–65. Springer, 2009.

[VT02] M Alex O Vasilescu and Demetri Terzopoulos. Multilinear analysis of image ensembles: Tensorfaces. In *European conference on computer vision*, pages 447–460. Springer, 2002.

[VT04] M Alex O Vasilescu and Demetri Terzopoulos. Tensortextures: multilinear image-based rendering. *ACM Transactions on Graphics (TOG)*, 23(3):336–342, 2004.

[WA16] Yining Wang and Animashree Anandkumar. Online and differentially-private tensor decomposition. In *Proceedings of the 30th International Conference on Neural Information Processing Systems*, NIPS'16, page 3539–3547, Red Hook, NY, USA, 2016. Curran Associates Inc.

[Wag92] David G. Wagner. On the perfect one—factorization conjecture. *Discrete Mathematics*, 104(2):211 – 215, 1992.

[WDDFS17] Miaoyan Wang, Khanh Dao Duc, Jonathan Fischer, and Yun S. Song. Operator norm inequalities between tensor unfoldings on the partition lattice. *Linear Algebra and its Applications*, 520:44–66, May 2017.

[WEAM19] Alexander S Wein, Ahmed El Alaoui, and Cristopher Moore. The kikuchi hierarchy and tensor pca. In *2019 IEEE 60th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 1446–1468. IEEE, 2019.

[Wed72] Per-Åke Wedin. Perturbation bounds in connection with singular value decomposition. *BIT Numerical Mathematics*, 12(1):99–111, 1972.

[WGY21] Yinan Wang, Weihong "Grace" Guo, and Xiaowei Yue. Tensor decomposition to compress convolutional layers in deep learning. *IISE Transactions*, pages 1–33, 2021.

[WY+20] Yinan Wang, Xiaowei Yue, et al. Cpac-conv: Cp-decomposition to approximately compress convolutional layers in deep learning. *arXiv preprint arXiv:2005.13746*, 2020.

[YAD19] Farzane Yahyanejad, Réka Albert, and Bhaskar DasGupta. A survey of some tensor analysis techniques for biological systems. *Quantitative Biology*, 7(4):266–277, 2019.

[YHPC18] Tom Young, Devamanyu Hazarika, Soujanya Poria, and Erik Cambria. Recent trends in deep learning based natural language processing. *ieee Computational intelligenCe magazine*, 13(3):55–75, 2018.

[ZX18] Anru Zhang and Dong Xia. Tensor svd: Statistical and computational limits. *IEEE Transactions on Information Theory*, 64(11):7311–7338, 2018.