Tensor Modeling of High-dimensional Distributions and Nonlinear Functions

A DISSERTATION
SUBMITTED TO THE FACULTY OF THE GRADUATE SCHOOL
OF THE UNIVERSITY OF MINNESOTA
BY

Nikolaos Kargas

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR THE DEGREE OF
DOCTOR OF PHILOSOPHY

Prof. Nicholas D. Sidiropoulos, Advisor

December, 2020

Acknowledgments

First and foremost, I would like to thank my advisor, Professor Nikolaos D. Sidiropoulos for giving me the opportunity to embark on this journey as a graduate student at the University of Minnesota. His ingenuity, breadth of knowledge, scientific honesty and dedication to research has served as an inspiration to me. He has been there in every step of this journey, supporting me through good and bad times. I am grateful for all his valuable advice which has shaped me into a better researcher.

I am thankful to Professor Georgios Giannakis, Professor George Karypis and Professor Mingyi Hong for serving on my thesis committee and providing valuable feedback on my thesis. I am also thankful to Professor Tom Luo, Profesor Yousef Saad and Professor Jarvis Haupt who have influenced me greatly through their research work and lectures. I benefited greatly through their teaching which helped me build a solid foundation on essential areas and played important role in my research afterwards. Also, special thanks to my undergraduate and MSc advisors Professor Michail G. Lagoudakis and Professor Aggelos Bletsas for encouraging and supporting me to pursue a PhD degree in the US.

My sincerest thanks go to my former and current labmates: Kejun Huang, Xiao Fu, John Tranter, Aritra Konar, Charilaos Kanatsoulis, Bo Yang, Ahmed Zamzam, Cheng Qian, Faisal Almutairi, Panagiotis Alevizos, Yunmei Shi, Mikael Sorensen, Mohamed Salah, Magda Amiridi and Paris Karakasis for the insightful discussions and the great moments we had together.

Additional thanks also goes out to all the friends that I made in Minnesota and Virginia: Agi, Andreas, Ioanna, Eva, Konstantina, Vasilis, Panagiotis, Giannis, Kostas, Maria and Thanos for making my PhD life much more enjoyable. Many thanks go to my friends in Greece: Katerina, Marina, Vasilis, Konstantinos, Eleni, Nikos, Christos, Katerina, Giorgos and Giannis who have been there for me for over a decade.

I would like to thank my family: my father Kiriakos, my mother Despoina and my sister

Anna for their unconditional love, their absolute confidence in me and continuous support.

Last but not least, I would like to thank my partner Eleni. She is always there for me and has helped me overcome the difficulties I faced during my PhD. This work could not have been done without her support.

Nikos Kargas, December 21 2020.

Dedication

In loving memory of my grandmother Argiro.

Abstract

Distribution estimation lies at the heart of statistical learning. Given the probability distribution of a set of random variables it is possible to compute marginal or conditional probabilities, moments and design optimal estimators. In this dissertation, we reveal an interesting link between multivariate statistics and tensors and propose a novel framework for joint Probability Mass Function (PMF) estimation given limited and possibly very incomplete data samples. We show that any joint PMF admits a naive Bayes model representation with a finite-alphabet latent variable. If the latent alphabet size is under a certain threshold, then the joint PMF of an arbitrary number of random variables can be identified from three-dimensional marginal distributions. We develop a practical and efficient algorithm that is shown to work well on both simulated and real data. We also extend our approach to mixture models of continuous variables. We consider the special case of mixture models whose component distributions factor into the product of the associated marginals and propose a two-stage approach which recovers the component distributions of the mixture under a smoothness condition.

The second part of the dissertation focuses on the problem of nonlinear function approximation. In practice, when labeled data are available we are often interested in methods that directly model the relationship or the conditional distribution function between the features and the target variable. It is desirable to develop methods that are expressive enough to capture a wide class of functions and at the same time are scalable and efficient. We show that the canonical polyadic decomposition model offers an appealing solution for modeling and learning a general nonlinear function. We formulate the problem as a smooth tensor decomposition problem with missing data and prove that under certain conditions correct nonlinear system identification is possible. We extend our method to multivariate functions of continuous inputs by proposing a generalization of the canonical polyadic decomposition from tensors to multivariate functions. The merits of our approach are illustrated using several synthetic and real multivariate regression tasks.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

1.1 Background and Motivation

Machine Learning (ML) and Artificial Intelligence (AI) have gained increasing attention and experienced tremendous growth over the past decade. ML techniques harness the power of user generated data to learn, discover hidden patterns, provide actionable insights and make predictions. ML and AI have had significant impact in many diverse areas including healthcare, finance, retail, transportation and online entertainment. One of the main ML applications in healthcare is the identification and diagnosis of diseases [78]. ML techniques can assist physicians and improve accuracy in early detection and diagnosis by modeling the relationship between medical features and the absence or presence of a specific decease. ML had already a significant impact in content platforms which rely heavily on user generated data to improve their services [36]. For example, based on a user's rating history as well as the history of other users ML techniques are being used to understand and predict user preferences, pertaining to news, movies or music and improve recommendations. E-commerce websites analyze complex and highly unstructured text data corresponding to user feedback to obtain deeper insights into customer opinion [30]. Finally, in the context of transportation systems, ML techniques are used for traffic congestion identification and prediction [21]. There are important questions and challenges that arise in the aforementioned applications. Real data are often high-dimensional, noisy, unstructured or incomplete which pose significant challenges on how to store, process, analyze and extract useful information. For example, healthcare data is often characterized by relatively few training instances and a high-dimensional feature space. Recomender systems data are highly incomplete because users interact only with a small portion of the available entities. Text data are high-dimensional and unstructured combining a vast amount of terms.

This dissertation studies principled ML approaches for learning from high-dimensional and potentially incomplete data. We focus on two fundamental ML problems namely *distribution estimation* and *nonlinear function approximation*. Distribution estimation lies at the heart of statistical learning and is a fundamental problem in *unsupervised ML*. Having access to the true joint probability distribution function of a set of random variables, allows us to calculate the conditional and marginal probability distributions of arbitrary subsets of variables, their corresponding expectations, sample new data points and it enables optimal estimation in terms of mean-square error and minimum probability of error. On the other hand, when labeled data are available we are often interested in methods that directly model the functional relationship between the features and the target variable, e.g., the conditional expectation of the target variable given the features (supervised ML). Nonlinear function learning is the problem of identifying the unknown nonlinear mapping between the input and output of a system given a finite number of observed input-output pairs. We focus on both theoretical and practical aspects of the two studied problems. We study conditions under which it is possible to provably recover the true data generating distribution or the nonlinear relationship between input and output data. We develop expressive models which are intuitive, interpretable and provide algorithmic solutions which are scalable and efficient.

Our main building blocks are tensors and tensor decomposition. Tensors are high dimensional generalization of matrices. Tensor data arise in numerous domains e.g., medical images [81, 117], knowledge bases [109], spatio-temporal data [56] and discrete functions [61] can all be naturally modeled using tensors. Tensor decomposition is a powerful tool for analyzing multi-way data and had major successes in applications spanning machine learning, statistics, signal processing and data mining [66, 90, 98]. Tensor decomposition have been applied in many diverse machine learning applications e.g., modeling large multi-dimensional and multi-relational data [54, 112], in recommender systems [115], in community detection and clustering to model user interactions across different networks [89], in knowledge base completion and link prediction [72, 109] and crowdsourcing [108]. Tensor decomposition has also been used in deep learning, for speeding up and compressing neural network models [73, 80, 86].

In Chapter 3 and chapter 4 we focus on the problem of distribution estimation. In chapter 5 and chapter 6, we turn our attention to the supervised case and the problem of learning a nonlinear function of input-output data. Our contributions are summarized in the next section.

1.2 Contributions

1.2.1 Chapter 3 — 'Kolmogorov Extension' for Finite-alphabet Random Vectors

The problem of joint Probability Mass Function (PMF) estimation can be stated as follows: Given a set of independently and identically generated data-points $\{\mathbf{x}_m\}_{m=1}^{M}$, find an estimate of the true underlying data distribution. Without structural assumptions joint PMF estimation is often considered mission impossible – the number of unknowns grows exponentially with the number of variables. But who gives us the structural model? Is there a generic, 'non-parametric' way to control joint PMF complexity without relying on a priori structural assumptions regarding the underlying probability model? Is it possible to discover the operational structure without biasing the analysis up front? What if we only observe random subsets of the variables, can we still reliably estimate the joint PMF of all? In this work, we reveal an interesting link between tensors and multivariate statistics. Any joint PMF can be represented by a naive Bayes model with a bounded number of latent states (rank of the joint PMF) and the rank can be interpreted as a measure of statistical dependence of the associated random variables. Exploiting tensor algebra, we prove that under certain conditions the joint PMF of an arbitrary number of random variables can be recovered from three-dimensional marginal distributions. The result is reminiscent of Kolmogorov's extension theorem – consistent specification of lower-dimensional distributions induces a unique probability measure for the entire process. We propose a novel framework for joint PMF estimation based on coupled tensor factorization and perform synthetic and real-data experiments on movie recommendation and data classification to showcase the effectiveness of the approach. The results of this chapter are reported in [57, 59].

1.2.2 Chapter 4 — Learning Mixtures of Smooth Product Distributions

In chapter 4, we focus on the problem of *joint Probability Density Function* (PDF) estimation extending our results to continuous variables. Specifically, we study the problem of learning a mixture model of non-parametric product distributions. The problem is well-studied in the parametric setting, i.e., when the component distributions are members of a parametric family – such as Gaussian distributions. In this chapter, we study multivariate mixtures of *non-parametric* product distributions. We propose a two-stage approach which recovers the component distributions of the mixture under a smoothness condition. Our approach builds upon the identifiability properties of the Canonical Polyadic Decomposition (CPD), in tandem with Fourier and Shannon-Nyquist sampling staples from signal processing. We demonstrate the effectiveness of the approach on synthetic and real datasets. The results of this chapter are reported [60].

1.2.3 Chapter 5 — Supervised Learning via Tensor Completion

In chapter 5, we focus on the problem of nonlinear function approximation. Function approximation from input and output data pairs constitutes a fundamental problem in *supervised learning*. We show that identifying a general nonlinear function $y = f(x_1, \ldots, x_N)$ from input-output examples can be formulated as a tensor completion problem and under certain conditions provably correct nonlinear system identification is possible. We model the interactions between the N input variables and the scalar output of a system by a single N-way tensor, and setup a weighted low-rank tensor completion problem with smoothness regularization which we tackle using a block coordinate descent algorithm. We extend our method to the multi-output setting and the case of partially observed data, which cannot be readily handled by neural networks. Finally, we demonstrate the effectiveness of the approach using several regression tasks including some standard benchmarks and a challenging student grade prediction task. The results of this chapter are reported in [61, 62].

1.2.4 Chapter 6 — Canonical Decomposition of Multivariate Functions

In chapter 6, we propose a generalization of the Canonical Polyadic Decomposition (CPD) from tensors to multivariate functions of continuous variables. We approximate a compactly supported multivariate function using a tensor of truncated multidimensional Fourier series coefficients and propose a hidden tensor factorization formulation for learning a low-rank CPD model of the Fourier coefficients tensor. In contrast to prior work, our method is quite general as it can model any compactly supported multivariate function that can be well-approximated by a finite multidimensional Fourier series, and under certain conditions it guarantees that the unknown function is uniquely characterized by the given input-output data. Furthermore, our model naturally allows stochastic gradient updates allowing it to scale to larger datasets. We develop two optimization algorithms and demonstrate promising results on synthetic and real multivariate regression tasks. The results of this chapter are reported in [58].

1.3 Thesis Outline

- In Chapter 2, we review necessary background on tensors and tensor decomposition. We introduce the Canonical Polyadic Decomposition (CPD) a powerful tensor model and discuss its identifiability properties.

- In chapter 3 we focus on the problem of joint Probability Mass Function (PMF) estimation from limited and partially observed data. Exploiting tensor algebra, we prove that under certain conditions the joint PMF of an arbitrary number of random variables can be recovered from three-dimensional marginal distributions. We propose a novel framework for joint PMF estimation based on coupled tensor factorization.

- In chapter 4, we study the problem of learning a mixture model of non-parametric product distributions. We propose a two-stage approach which recovers the component distributions of the mixture under a smoothness condition.

- In chapter 5, we show that identifying a general nonlinear function from input-output examples can be formulated as a tensor completion problem and under certain conditions provably correct nonlinear system identification is possible. We setup a weighted low-rank tensor completion problem with smoothness regularization which we tackle using a block coordinate descent algorithm.

- In chapter 6, we propose a generalization of the CPD from tensors to multivariate functions of continuous variables. We approximate a compactly supported multivariate function using a tensor of truncated multidimensional Fourier series coefficients and propose a hidden tensor factorization formulation for learning a low-rank CPD model of the Fourier coefficients tensor.

- In Chapter 7 we present conclusions and discussion on future research directions.

Chapter 2

Notation and Tensor Preliminaries

2.1 Notation

We use the symbols $\mathbf{x}$, $\mathbf{X}$, $\mathcal{X}$ for vectors, matrices and tensors respectively. We use the notation $\mathbf{x}[n]$, $\mathbf{X}[:, n]$, $\mathcal{X}[:, :, n]$ to refer to a particular element of a vector, a column of a matrix and a slab of a tensor. Symbols $\circ$, $\otimes$, $\circledast$, $\odot$ denote the outer, Kronecker, Hadamard and Khatri-Rao (column-wise Kronecker) product respectively. The vectorization operator is denoted as $\mathrm{vec}(\mathbf{X})$, $\mathrm{vec}(\mathcal{X})$ for a matrix and tensor respectively. Additionally, diag$(\mathbf{x})$ denotes the diagonal matrix with the elements of vector $\mathbf{x}$ on its diagonal. The set of integers $\mathcal{S} = \{1, \ldots, N\}$ is denoted as $[N]$ and $|\mathcal{S}|$ denotes the cardinality of the set $\mathcal{S}$. Uppercase letters (e.g., X) denote scalar random variables.

2.2 Canonical Polyadic Decomposition

An N-way tensor $\mathcal{X} \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_N}$ is a multi-dimensional array whose entries are indexed by N coordinates. A polyadic decomposition expresses $\mathcal{X}$ as a sum of rank-1 components

$$\mathcal{X} = \sum_{r=1}^{R} \mathbf{a}_r^1 \circ \mathbf{a}_r^2 \circ \cdots \circ \mathbf{a}_r^N, \tag{2.1}$$

where $\mathbf{a}_r^n \in \mathbb{R}^{I_n}$. If the number of rank-1 components is minimal then the decomposition is called the Canonical Polyadic Decomposition (CPD) of $\mathcal{X}$ and R is called the rank of $\mathcal{X}$ [98]. By defining factor matrices $\mathbf{A}_n = [\mathbf{a}_1^n, \cdots, \mathbf{a}_R^n] \in \mathbb{R}^{I_n \times R}$, the elements of the tensor $\mathcal{X}$ can be

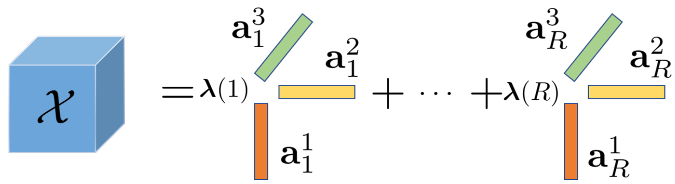


Figure 2.1: Illustration of the canonical polyadic decomposition of a three-way tensor.

expressed as

$$\mathcal{X}[i_1,\ldots,i_N]=\sum_{r=1}^{R}\prod_{n=1}^{N}\mathbf{A}_n[i_n,r]. \tag{2.2}$$

We use the notation $\mathcal{X}=[\![\mathbf{A}_1,\ldots,\mathbf{A}_N]\!]$ to denote the tensor synthesized from the CPD model using these factors. The CPD model is one of the most popular tensor models mainly due to its simplicity and its uniqueness properties. The CPD model is universal i.e., every tensor admits a CPD of finite rank and it is unique under mild conditions [98]. Sometimes one wishes to restrict the columns of $\mathbf{A}_n$'s to have unit norm (e.g., as in SVD). Therefore, the tensor can be represented as

$$\mathcal{X}[i_1,i_2,\ldots,i_N]=\sum_{r=1}^{R}\boldsymbol{\lambda}[r]\prod_{n=1}^{N}\mathbf{A}_n[i_n,r], \tag{2.3}$$

where $\|\mathbf{A}_n[:,r]\|_p=1$ for a certain $p\geq 1$, $\forall\, n,r$, and $\boldsymbol{\lambda}=[\boldsymbol{\lambda}[1],\ldots,\boldsymbol{\lambda}[R]]^T$ with $\|\boldsymbol{\lambda}\|_0=R$ is employed to 'absorb' the norms of columns. An illustration of a three-way tensor and its CPD is shown in Figure 2.1. Under such cases, we denote the N-way tensor as $\mathcal{X}=[\![\boldsymbol{\lambda},\mathbf{A}_1,\ldots,\mathbf{A}_N]\!]$. We will refer to the decomposition of $\mathcal{X}$ into nonnegative factors $\boldsymbol{\lambda}\in\mathbb{R}_+^R$, $\mathbf{A}_n\in\mathbb{R}_+^{I_n\times R}$ as nonnegative decomposition.

The following definitions will prove useful in the rest of the paper. We define the *mode-n matrix unfolding* of $\mathcal{X}$ as the matrix $\mathbf{X}^{(n)}$ of size $\prod_{\substack{k=1\\k\neq n}}^{N} I_k\times I_n$. We have that $\mathcal{X}[i_1,i_2,\ldots,i_N]=\mathbf{X}^{(n)}[j,i_n]$, where

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

In terms of the CPD factors, the mode-n matrix unfolding can be expressed as

$$\mathbf{X}^{(n)}=\left(\bigodot_{j\neq n}\mathbf{A}_j\right)\mathrm{diag}(\boldsymbol{\lambda})\mathbf{A}_n^T, \tag{2.4}$$

where $\underset{j\neq n}{\odot} \mathbf{A}_j = \mathbf{A}_N \odot \cdots \odot \mathbf{A}_{n+1} \odot \mathbf{A}_{n-1} \odot \cdots \odot \mathbf{A}_1$. We can also express a tensor in a vectorized form $\mathcal{X}[i_1, i_2, \ldots, i_N] = \mathbf{x}[j]$, where

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

In terms of the CPD factors, the vectorized form of a tensor can be expressed as

$$\mathrm{vec}(\mathcal{X}) = \left(\underset{j=1}{\overset{N}{\odot}} \mathbf{A}_j \right) \boldsymbol{\lambda}. \tag{2.5}$$

The n*-mode product* of a tensor $\mathcal{X} \in \mathbb{R}^{I_1 \times \cdots \times I_N}$ with a matrix $\mathbf{V} \in \mathbb{R}^{I_n \times J}$ is denoted by $\mathcal{X} \times_n \mathbf{V}$, and is given by

$$(\mathcal{X} \times_n \mathbf{V})[i_1, \ldots, i_{n-1}, j, i_{n+1}, \ldots, i_N] = \sum_{i_n=1}^{I_n} \mathcal{X}[i_1, \ldots, i_N] \mathbf{V}[i_n, j],$$

i.e., each n-th mode fiber is multiplied by the columns of $\mathbf{V}$

$$(\mathcal{X} \times_n \mathbf{V})^{(n)} = \mathcal{X}^{(n)} \mathbf{V}. \tag{2.6}$$

If $\mathcal{X} = [\![\boldsymbol{\lambda}, \mathbf{A}_1, \ldots, \mathbf{A}_N]\!]$, then by combining (2.4) and (2.6) we have

$$\mathcal{X} \times_n \mathbf{V} = [\![\boldsymbol{\lambda}, \mathbf{A}_1, \ldots, \mathbf{V}^T \mathbf{A}_n, \ldots, \mathbf{A}_N]\!]. \tag{2.7}$$

2.3 Uniqueness of Canonical Polyadic Decomposition

A distinctive feature of tensors is that they have *essentially* unique CPD under mild conditions, even when R is much larger than $I_1, \ldots, I_N$. Let us first formally define what we mean by essential uniqueness of rank decomposition of tensors.

Definition 2.1. *(Essential uniqueness) For a tensor $\mathcal{X}$ of rank R, we say that a decomposition $\mathcal{X} = [\![\mathbf{A}_1, \ldots, \mathbf{A}_N]\!]$, $\mathbf{A}_n \in \mathbb{R}^{I_n \times R}$ is essentially unique if the factors are unique up to a common permutation and scaling/counterscaling of the columns. This means that if there exists another decomposition $\mathcal{X} = [\![\widehat{\mathbf{A}}_1, \ldots, \widehat{\mathbf{A}}_N]\!]$, then, there exists a permutation matrix $\mathbf{\Pi}$ and diagonal matrices $\mathbf{\Lambda}_n$ such that $\widehat{\mathbf{A}}_n = \mathbf{A}_n \mathbf{\Pi} \mathbf{\Lambda}_n$ and $\prod_{n=1}^{N} \mathbf{\Lambda}_n = \mathbf{I}$.*

Note that if a tensor has an essentially unique *nonnegative* CPD, then the only ambiguity is column permutation of the column-normalized factors $\{\mathbf{A}_n\}_{n=1}^N$, which simply amounts to a permutation of the rank-one 'chicken feet' outer products (rank-one tensors) in Fig. 2.1, that is clearly unavoidable. There is no scaling ambiguity for *nonnegative* column-normalized representation $\mathcal{X} = [\![\boldsymbol{\lambda}, \mathbf{A}_1, \ldots, \mathbf{A}_N]\!]$, where there is obviously no sign ambiguity and all scaling is 'absorbed' in $\boldsymbol{\lambda}$.

Regarding the essential uniqueness of tensors, let us consider the three-way case first. The following is arguably the most well-known uniqueness condition that was revealed by Kruskal in 1977.

Theorem 2.1. *[70] Let $\mathcal{X} = [\![\boldsymbol{\lambda}, \mathbf{A}_1, \mathbf{A}_2, \mathbf{A}_3]\!]$, where $\mathbf{A}_1 \in \mathbb{R}^{I_1 \times R}$, $\mathbf{A}_2 \in \mathbb{R}^{I_2 \times R}$, $\mathbf{A}_3 \in \mathbb{R}^{I_3 \times R}$. If $k_{\mathbf{A}_1} + k_{\mathbf{A}_2} + k_{\mathbf{A}_3} \geq 2R + 2$ then rank$(\mathcal{X}) = R$ and the decomposition of $\mathcal{X}$ is essentially unique.*

Here, $k_{\mathbf{A}}$ denotes the Kruskal rank of the matrix $\mathbf{A}$ which is equal to the largest integer such that every subset of $k_{\mathbf{A}}$ columns are linearly independent. Theorem 2.1 implies the following generic result: The decomposition $\mathcal{X} = [\![\boldsymbol{\lambda}, \mathbf{A}_1, \mathbf{A}_2, \mathbf{A}_3]\!]$ is essentially unique, almost surely, if

$$\min(I_1, R) + \min(I_2, R) + \min(I_3, R) \geq 2R + 2. \tag{2.8}$$

This is because $k_{\mathbf{A}_n} = \min(I_n, R)$ with probability one if the elements of $\mathbf{A}_n$ are generated following a certain absolutely continuous distribution. More relaxed and powerful uniqueness conditions have been proven in recent years.

Theorem 2.2. *[22], [27] Let $\mathcal{X} = [\![\boldsymbol{\lambda}, \mathbf{A}_1, \mathbf{A}_2, \mathbf{A}_3]\!]$, where $\mathbf{A}_1 \in \mathbb{R}^{I_1 \times R}$, $\mathbf{A}_2 \in \mathbb{R}^{I_2 \times R}$, $\mathbf{A}_3 \in \mathbb{R}^{I_3 \times R}$, $I_1 \leq I_2 \leq I_3$, $I_1 \geq 3$ and $R \leq I_3$. Then, rank$(\mathcal{X}) = R$ and the decomposition of $\mathcal{X}$ is essentially unique, almost surely, if and only if $R \leq (I_1 - 1)(I_2 - 1)$.*

Theorem 2.3. *[22] Let $\mathcal{X} = [\![\boldsymbol{\lambda}, \mathbf{A}_1, \mathbf{A}_2, \mathbf{A}_3]\!]$, where $\mathbf{A}_1 \in \mathbb{R}^{I_1 \times R}$, $\mathbf{A}_2 \in \mathbb{R}^{I_2 \times R}$, $\mathbf{A}_3 \in \mathbb{R}^{I_3 \times R}$, $I_1 \leq I_2 \leq I_3$. Let α, β be the largest integers such that $2^\alpha \leq I_1$ and $2^\beta \leq I_2$. If $R \leq 2^{\alpha+\beta-2}$ then the decomposition of $\mathcal{X}$ is essentially unique almost surely. The condition also implies that if $R \leq \frac{(I_1+1)(I_2+1)}{16}$, then $\mathcal{X}$ has a unique decomposition almost surely.*

There are many more different uniqueness conditions for CPD. The take-home point here is that the CPD model is essentially generically unique even if R is much larger than I_1, I_2, I_3 – so long it is less than maximal possible rank. For example, in Theorem 2.3, R can be as large as $\mathcal{O}(I_1 I_2)$ (but not equal to $I_1 I_2$), and the CPD model is still unique.

Chapter 3

‘Kolmogorov Extension’ for Finite-alphabet Random Vectors

3.1 Introduction

Estimating a joint Probability Mass Function (PMF) of a set of random variables is of great interest in numerous applications in the fields of machine learning, data mining and signal processing. In many cases, we are given partial observations and/or statistics of the data, i.e., incomplete data, marginalized lower-dimensional distributions, or lower-order moments of the data, and our goal is to estimate the missing data. If the full joint PMF of all variables of interest were known, this would have been a straightforward task. A classical example is in recommender systems, where users rate only a small fraction of the total items (e.g., movies) and the objective is to make item recommendations to users according to predicted ratings. If the joint PMF of the item ratings is known, such recommendation is readily implementable based on the conditional expectation or mode of the unobserved ratings given the observed ratings. A closely related problem is top-K recommendation, where the goal is to predict the K items that a user is most likely to buy next. When the joint PMF of the items is known, it is easy to identify the K items with the highest individual or joint (‘bundle’) conditional probability given the observed user ratings. Another example is data classification. If the joint PMF of the features and the label is known, then given a test sample it is easy to infer the label according to the Maximum a *Posteriori* (MAP) principle. In fact, the joint PMF can be used to infer any of the features (or subsets of them), which is useful in imputing incomplete information in surveys or databases.

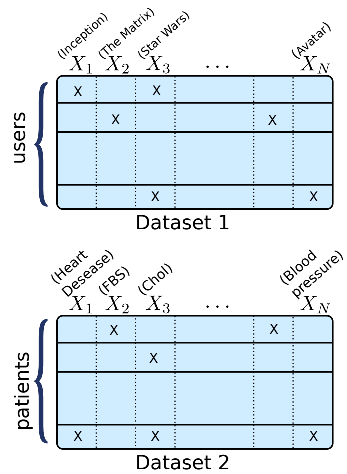


Figure 3.1: Applications of joint PMF estimation. Top: recommender systems: given partially observed ratings of a user on movies, we would like to infer the unobserved ratings. Bottom: classification problems: given medical features of people, we would like to infer if a person has heart disease.

Despite its importance in signal and data analytics, estimating the joint PMF is often considered mission impossible in general, if no structure or relationship between the variables (e.g., a tree structure or a Markovian structure) can be assumed. This is true even when the problem size is merely moderate. The reason is that the number of unknown parameters is exponential in the number of variables. Consider a simple scenario of 10 variables taking 10 distinct values each. The number of parameters we need to estimate in this case is 10^{10}. The 'naive' approach for joint PMF estimation is counting the occurences of the joint variable realizations. In practice, however, when dealing with even moderately large sets of random variables, the probability of encountering any particular realization is very low. Therefore, only a small portion of the empirical distribution will be non-zero given a reasonable amount of data samples – this makes the approach very inaccurate.

In many applications, different workarounds have been proposed to circumvent this sample complexity problem. For example, in recommender systems, instead of trying to estimate the joint PMF of the ratings (which would be the estimation-theoretic gold standard), the most popular approach is based on low-rank matrix completion [49, 68, 83]. The idea is that the users can be roughly clustered into several types, and users of the same type would rate different movies similarly. Consequently, the user-rating matrix is approximately low rank and this is used as prior information to infer the missing ratings. In classification, parsimonious function approximations are employed to model the relationship (or the conditional probability function) between the features and the label. Successful methods that fall into this category are support vector machines (linear function approximation), logistic regression (log-linear function approximation) and more recently kernels and neural networks (nonlinear function approximation) [13].

The above mentioned methods are nice and elegant, have triggered a tremendous amount of theoretical research and practical applications, and have been successful in many ways. However, these workarounds have not yet answered our question of interest: Can we ever reliably estimate the joint PMF of variables given limited data? This question is very well-motivated in practice, since knowledge of the joint PMF is indeed the gold standard: it enables optimal estimation under a variety of well-established criteria, such as mean-square error and minimum probability of error or Bayes risk. Knowing the joint PMF can facilitate a large variety of applications including recommender systems and classification in a unified and statistically optimal way, instead of resorting to often ad-hoc modeling tools.

In this chapter, we show that if the joint PMF of any three variables can be estimated, then the joint PMF of all the variables can be provably recovered under relatively mild conditions. The result is reminiscent of Kolmogorov's extension theorem – consistent specification of lower-dimensional distributions induces a unique probability measure for the entire process. The difference is that for processes of limited complexity (rank of the high-dimensional PMF) it is possible to obtain complete characterization from only three-dimensional distributions. In fact not all three-dimensional PMFs are needed; and under more stringent conditions even two-dimensional will do. The rank condition on the high-dimensional joint PMF has an interesting interpretation: loosely speaking, it means that the random variables are 'reasonably (in)dependent'. This makes sense, because estimation problems involving fully independent or fully dependent regressors and unknowns are contrived – it is the middle ground that is interesting. It is also important to note that the marginal PMFs of triples can be reliably estimated at far smaller sample complexity than the joint PMF of all variables. For example, for user-movie ratings, the marginal PMF of three given variables (movies) can be estimated by counting the co-occurrences of the given ratings (values of the variables) of the three given movies; but no user can rate *all* movies.

Contributions Our specific contributions are as follows:

- We propose a novel framework for joint PMF estimation given limited and possibly very incomplete data samples. Our method is based on a nice and delicate connection between the CPD [20, 40] and the naive Bayes model. The CPD model has been used to model and analyze tensor data (data with more than two indices) in signal processing and machine learning, and it has found many successful applications, such as speech separation [85], blind CDMA detection [100], array processing [99], spectrum sensing and unmixing in cognitive radio [33], topic modeling [6], and community detection [5]. Nevertheless, CPD has never been considered as a statistical learning tool for recovering a *general* joint PMF and our work is the first to establish the exciting connection [1].

• We present detailed identifiability analysis of the proposed approach. We first show that, any joint PMF can be represented by a naive Bayes model with a finite-alphabet latent variable – and the size of the latent alphabet (which happens to be the *rank* of the joint PMF tensor, as we will see) is bounded by a function of the alphabet sizes of the (possibly intermittently) observed variables. We further show that, if the latent alphabet size is under a certain threshold, then the joint PMF of *an arbitrary number* of random variables can be identified from three-dimensional marginal distributions. We prove this identifiability result by relating the joint PMF and marginal PMFs to the CPD model, which is known for its uniqueness even when the tensor rank is much larger than its outer dimensions.

• In addition to the novel formulation and identifiability results, we also propose an easily implementable joint PMF recovery algorithm. Our identification criterion can be considered as a coupled simplex-constrained tensor factorization problem, and we propose a very efficient alternating optimization-based algorithm to handle it. To deal with the probability simplex constraints that arise for PMF estimation, the celebrated Alternating Direction Method of Multipliers (ADMM) algorithm is employed, resulting in lightweight iterations. Judiciously designed simulations and real experiments on movie recommendation and classification tasks are used to showcase the effectiveness of the approach.

3.2 Problem Statement

Consider a set of N random variables, i.e., $\{X_n\}_{n=1}^N$. Assume that each X_n can take I_n discrete values and only the joint PMFs of variable triples, i.e., $\mathsf{Pr}(X_j = i_j, X_k = i_k, X_\ell = i_\ell)$'s, are

[1] There are works that considered using CPD to model a joint PMF for some specific problems [6]. However, these works rely on specific physical interpretation of the associated model, which is sharply different to our setup – in which we employ the CPD model to explain a general joint PMF without assuming any physical model.

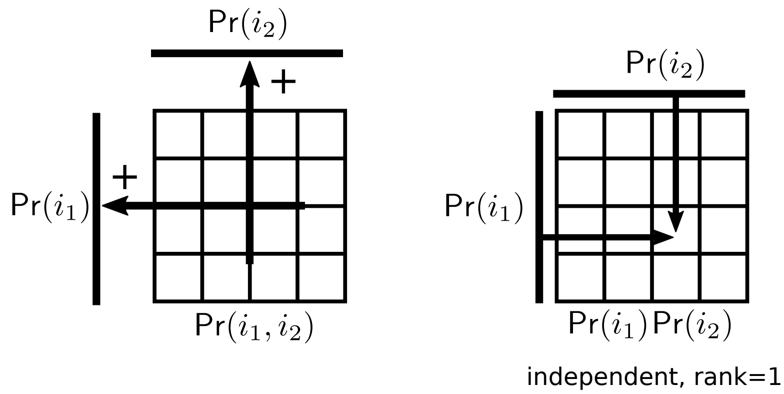


Figure 3.2: Lower-dimensional marginals as projections.

available. Can we identify the joint PMF of $\{X_n\}_{n=1}^N$, i.e., $\mathsf{Pr}(X_1 = i_1, \ldots, X_N = i_N)$, from the three-dimensional marginals? This question lies at the heart of statistical learning. To see this, consider a classification problem and let $X_1 \ldots, X_{N-1}$ represent the set of observed features, and X_N the sought label. If $\mathsf{Pr}(X_1 = i_1, \ldots, X_N = i_N)$ is known, then given a specific realization of the features, one can easily compute the posterior probability

$$\mathsf{Pr}(i_N | i_1 \ldots, i_{N-1}) = \frac{\mathsf{Pr}(i_1, \ldots, i_N)}{\sum_{i_N=1}^{I_N} \mathsf{Pr}(i_1, \ldots, i_{N-1}, i_N)},$$

and predict the label according the MAP principle (here $\mathsf{Pr}(i_1, \ldots, i_N)$ is shorthand for $\mathsf{Pr}(X_1 = i_1, \ldots, X_N = i_N)$ and likewise $\mathsf{Pr}(i_N | i_1 \ldots, i_{N-1})$ for $\mathsf{Pr}(X_N = i_N | X_1 = i_1 \ldots, X_{N-1} = i_{N-1})$. In recommender systems, given a set of observed item ratings $X_1 \ldots, X_{N-1}$ one can compute the conditional expectation of an unobserved rating given the observed ones

$$\mathbb{E}(X_N | i_1, \ldots, i_{N-1}) = \sum_{i_N=1}^{I_N} i_N \mathsf{Pr}(i_N | i_1, \ldots, i_{N-1}).$$

At this point the reader may wonder why we consider recovery from three-dimensional joint PMFs and not from one- or two-dimensional PMFs. It is well-known that recovery from one-dimensional marginal PMFs is possible when all random variables are known to be independent. In this case, the joint PMF is equal to the product of the individual one-dimensional marginals. Interestingly, recovery from one-dimensional marginals is also possible when the random variables are known to be fully dependent i.e., one is completely determined by the

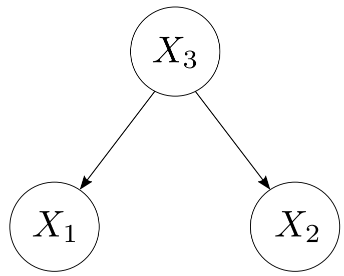


Figure 3.3: Bayesian network of three variables.

other. In this case, the joint PMF can be recovered if each one-dimensional marginal is a unique permutation of the other.

However, complete (in)dependence is unrealistic in statistical estimation and learning practice. In general it is not possible to recover a joint PMF from one-dimensional marginals. An illustration for two variables is shown in Figure 3.2: $\mathsf{Pr}(i_1, i_2)$ can be represented as a matrix, and $\mathsf{Pr}(i_1)$, $\mathsf{Pr}(i_2)$ are 'projections' of the matrix along the row and column directions using the projector $\mathbf{1}^T$ and $\mathbf{1}$, respectively: $\mathsf{Pr}(i_1) = \sum_{i_2=1}^{I_2} \mathsf{Pr}(i_1, i_2)$ and $\mathsf{Pr}(i_2) = \sum_{i_1=1}^{I_1} \mathsf{Pr}(i_1, i_2)$. In this case, if we denote $\mathbf{P}$ the matrix such that $\mathbf{P}(i_1, i_2) = \mathsf{Pr}(X_1 = i_1, X_2 = i_2)$, then $\mathrm{rank}(\mathbf{P}) = r > 1$ if X_1 and X_2 are not independent. From basic linear algebra, one can see that knowing $\mathbf{1}^T\mathbf{P}$ and $\mathbf{P1}$ is not enough for recovering $\mathbf{P}$ in general – since this is equivalent to solving a very underdetermined system of linear equations with $(I_1 + I_2) \times r$ variables but only $I_1 + I_2$ equations.

What if we know two-dimensional marginals? When the given random variables obey a probabilistic graphical model, and a genie reveals that model to us, then estimating a high-dimensional joint PMF from two-dimensional marginals may be possible. An example is shown in Figure 3.3. If we know *a priori* that random variables X_1 and X_2 are conditionally independent given X_3, one can verify that knowledge of $\mathsf{Pr}(X_1 = i_1, X_3 = i_3)$ and $\mathsf{Pr}(X_2 = i_2, X_3 = i_3)$ is sufficient to recover $\mathsf{Pr}(X_1 = i_1, X_2 = i_2, X_3 = i_3)$. However, this kind of approach hinges on knowing the probabilistic graph structure. Unfortunately, genies are hard to come by in real life, and learning the graph structure from data is itself a very challenging problem in statistical learning [67].

In our problem setup, we do not assume any *a priori* knowledge of the graph structure, and in this sense we have a 'blind' joint PMF recovery problem. Interestingly, under certain conditions, this is no hindrance.

3.3 Naive Bayes Model: A Rank-decomposition Perspective

We will show that *any* joint PMF admits a naive Bayes model *representation*, i.e., it can be generated from a latent variable model with just one hidden variable. The naive Bayes model postulates that there is a hidden discrete random variable H taking R possible values, such that given $H = h$ the discrete random variables $\{X_n\}_{n=1}^N$ are conditionally independent. It follows that the joint PMF of $\{X_n\}_{n=1}^N$ can be decomposed as

$$\mathsf{Pr}(i_1, i_2, \ldots, i_N) = \sum_{r=1}^{R} \mathsf{Pr}(r) \prod_{n=1}^{N} \mathsf{Pr}(i_n|r), \tag{3.1}$$

where $\mathsf{Pr}(r) := \mathsf{Pr}(H = r)$ is the prior distribution of the latent variable H and $\mathsf{Pr}(i_n|r) := \mathsf{Pr}(X_n = i_n|H = r)$ are the conditional distributions (Fig. 3.4). The naive Bayes model in (3.1) is also referred to as the latent class model [120] and is the simplest form of a Bayesian network [67]. It has been employed in diverse applications such as classification [84], density estimation [79] and crowdsourcing [24], just to name a few.

An interesting observation is that the naive Bayes model can be interpreted as a special nonnegative polyadic decomposition. This was alluded to in [77, 97] but not exploited for identifying the joint PMF from lower-dimensional marginals, as we do. Consider the element-wise representation in (2.3) and compare it with (3.1): each column of the factor matrices can represent a conditional PMF and the vector $\boldsymbol{\lambda}$ contains the prior probabilities of the latent variable H, i.e.,

$$\mathbf{A}_n[i_n, r] = \mathsf{Pr}(i_n|r), \quad \boldsymbol{\lambda}[r] = \mathsf{Pr}(r). \tag{3.2}$$

This is a *special* nonnegative polyadic decomposition model because it restricts $\mathbf{1}^T\boldsymbol{\lambda} = 1$. There is a subtle point however: the maximal rank R in a *canonical* polyadic decomposition model is bounded, but the number of latent states (latent alphabet size) for the naive Bayes model may exceed this bound. Even if the number of latent states is under the maximal rank bound, a naive Bayes model may be *reducible*, in the sense that there exists a naive Bayes model with fewer latent states that generates the same joint PMF. The net result is that *every* joint PMF admits a naive Bayes model *interpretation* with bounded R, and every naive Bayes model is or can be reduced to a special CPD model. We have the following result.

Proposition 3.1. *The maximum R needed to represent an* arbitrary *PMF as a naive Bayes model*

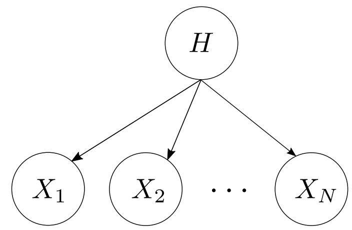


Figure 3.4: Naive Bayes model.

is bounded by the following inequality

$$R \leq \min_{k} \left(\prod_{\substack{n=1 \\ n \neq k}}^{N} I_n \right). \tag{3.3}$$

Proof: Let $\mathcal{X} \in \mathbb{R}_{+}^{I_1 \times I_2 \times I_3}$ denote a joint PMF of three random variables i.e., $\mathcal{X}[i_1, i_2, i_3] = \mathsf{Pr}(X_1 = i_1, X_2 = i_2, X_3 = i_3)$. We define the following matrices

$$\begin{aligned}
\mathbf{A}_1 &:= [\mathcal{X}[:,:,1], \cdots, \mathcal{X}[:,:,I_3]], \\
\mathbf{A}_2 &:= [\mathbf{I}_{I_2 \times I_2}, \cdots, \mathbf{I}_{I_2 \times I_2}] = \mathbf{1}_{I_3}^{T} \otimes \mathbf{I}_{I_2 \times I_2}, \\
\mathbf{A}_3 &:= \mathbf{I}_{I_3 \times I_3} \otimes \mathbf{1}_{I_2}^{T},
\end{aligned}$$

where $\mathbf{A}_1 \in \mathbb{R}_{+}^{I_1 \times I_2 I_3}, \mathbf{A}_2 \in \mathbb{R}_{+}^{I_2 \times I_2 I_3}, \mathbf{A}_3 \in \mathbb{R}_{+}^{I_3 \times I_2 I_3}$ and have used notation $\mathcal{X}[:,:,i_3]$ to denote the frontal slabs of the tensor $\mathcal{X}$. Additionally, $\mathbf{I}_{I_n \times I_n}$ denotes the identity matrix of size $I_n \times I_n$ and $\mathbf{1}_{I_n}$ is a vector of all 1's of size I_n. Then every frontal slab of the tensor $\mathcal{X}$ can be synthesized as $\mathcal{X}[:,:,i_3] = \mathbf{A}_1 \mathrm{diag}(\mathbf{A}_3[i_3,:]) \mathbf{A}_2^{T}$. Upon normalizing the columns of matrix $\mathbf{A}_1$ such that they sum to one and absorbing the scaling in $\boldsymbol{\lambda}$, i.e., $\mathbf{A}_1 = \widehat{\mathbf{A}}_1 \mathrm{diag}(\boldsymbol{\lambda})$ we can decompose the tensor as $\mathcal{X} = [\![\boldsymbol{\lambda}, \widehat{\mathbf{A}}_1, \mathbf{A}_2, \mathbf{A}_3]\!]$. The number of columns of each factor is $I_2 I_3$. Due to role symmetry, by permuting the modes of the tensor it follows that we need at most $\min(I_1 I_2, I_2 I_3, I_1 I_3)$ columns for each factor for exact decomposition.

The result is easily generalized to a four-way tensor $\mathcal{X} \in \mathbb{R}_{+}^{I_1 \times I_2 \times I_3 \times I_4}$ by noticing that each slab $\mathcal{X}[:,:,:,i_4]$ is a three-way tensor and thus can be decomposed as $[\![\boldsymbol{\lambda}_{i_4}, \widehat{\mathbf{A}}_{1,i_4}, \mathbf{A}_{2,i_4}, \mathbf{A}_{3,i_4}]\!]$

as before. We define

$$\begin{aligned} \boldsymbol{\lambda} &= [\boldsymbol{\lambda}_1^T, \cdots, \boldsymbol{\lambda}_{I_4}^T]^T, \\ \widehat{\mathbf{A}}_1 &= [\widehat{\mathbf{A}}_{1,1}, \cdots, \widehat{\mathbf{A}}_{1,I_4}], \quad \mathbf{A}_2 = [\mathbf{A}_{2,1}, \cdots, \mathbf{A}_{2,I_4}], \\ \mathbf{A}_3 &= [\mathbf{A}_{3,1}, \cdots, \mathbf{A}_{3,I_4}], \quad \mathbf{A}_4 = \mathbf{I}_{I_4} \otimes \mathbf{1}_{I_2 I_3}^T. \end{aligned}$$

The four-way tensor can therefore be decomposed as $[\![\boldsymbol{\lambda}, \widehat{\mathbf{A}}_1, \mathbf{A}_2, \mathbf{A}_3, \mathbf{A}_4]\!]$. Due to symmetry, the number of columns of each factor is at most $\min(I_1 I_2 I_3, I_2 I_3 I_4, I_1 I_3 I_4, I_1 I_2 I_4)$. By the same argument it follows that for a N-way tensor the bound on the nonnegative rank is $\min\limits_k(\prod_{\substack{n=1 \\ n\neq k}}^N I_n)$.

The proof of Proposition 3.1 employs the same type of argument used to prove the upper bound on tensor rank. The main difference is in the normalization – latent nonnegativity follows from data nonnegativity "for free" since the latent factors used for constructing the CPD are either fibers drawn from the joint PMF itself, or from identity matrices or Kronecker products thereof. While the proof is fairly straightforward for someone versed in tensor analysis, the implication of this proposition to probability theory is significant: it asserts that *every* joint PMF can be represented by a naive Bayes model with a *bounded* number of latent states $|\mathcal{H}|$. In fact, the connection between a naive Bayes model and CPD was utilized to approach some machine learning problems such as community detection and Gaussian Mixture Model (GMM) estimation in [5]. However, in those cases, the hidden variable has a specific physical meaning (e.g., $H = r$ represents the rth community in community detection) and thus connection was established using a specific data generative model. Here, we emphasize that even when there is no physically meaningful H or presumed generative model, *one can always represent an arbitrary joint PMF, possibly corresponding to a very complicated probabilistic graphical model, as a "simple" naive Bayes model with a bounded number of latent states R.* This result is very significant, also because it spells out that the latent structure of a probabilistic graphical model cannot be identified by simply assuming few hidden nodes; one has to limit the number of hidden node states as well.

We should remark that although any joint PMF admits a naive Bayes representation, this does not mean that such representation is unique. Clearly, R needs to be strictly smaller than the upper bound in (3.3) to guarantee uniqueness (cf. Theorems 2.1-2.3). Fortunately, many joint PMFs that we encounter in practice are relatively low-rank tensors, since random variables in the real world are only moderately dependent. This leads to an interesting connection between linear

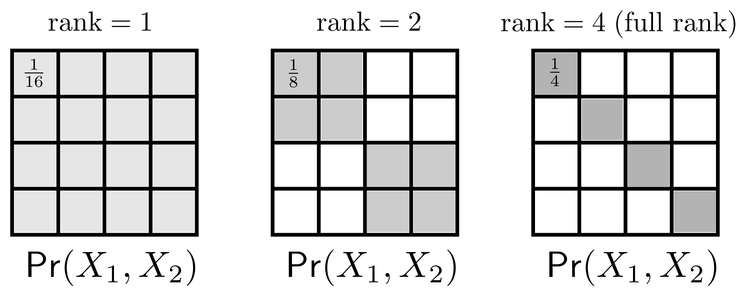


Figure 3.5: Rank and independence.

dependence/independence and statistical dependence/independence. To explain, let us consider the simplest case where $N = 2$. In this case, we have

$$\mathsf{Pr}(i_1, i_2) = \sum_{r=1}^{R} \mathsf{Pr}(r)\mathsf{Pr}(i_1|r)\mathsf{Pr}(i_2|r). \tag{3.4}$$

The two-way model corresponds to Nonnegative Matrix Factorization (NMF) and is related to Probabilistic Latent Semantic Indexing (PLSI) [42], [35]. For the two-way model, independence of the variables implies that the probability matrix is rank-1. On the other hand, when the variables are fully dependent i.e., the value of one variable exactly determines the value of the other, the probability matrix is full-rank. However, low-rank does not necessarily mean that the variables are close to being independent as shown in Figure 3.5. There, a low rank probability matrix (rank $= 2$) can also model highly dependent random variables. In practice, we expect that random variables will be neither independent nor fully dependent and we are interested in cases where the rank of the joint PMF is lower (and ideally much lower) than the upper bound given in Proposition 3.1.

As a sanity check, we conducted preliminary experiments on some real-life data. As anticipated, we verified that many joint PMFs are indeed low-rank tensors in practice. Table 3.1 shows interesting results: The joint PMF of three movies over 5 rating values was first estimated, using data from the MovieLens project. The joint PMF is then factored using a nonnegative CPD model with different rank values. One can see that with rank as low as 5, the modeling error in terms of the relative error $\|\mathcal{X} - \widehat{\mathcal{X}}\|_F / \|\mathcal{X}\|_F$ is quite small, meaning that the low-rank modeling is fairly accurate. The same applies to two more datasets drawn from the UCI repository.

Table 3.1: Relative error for different joint PMFs of 3 variables.

	Rank (R)		
	5	10	15
INCOME	2.1×10^{-2}	5.5×10^{-3}	5.1×10^{-3}
MUSHROOM	4.3×10^{-2}	2.4×10^{-2}	1.9×10^{-2}
MOVIELENS	1.8×10^{-2}	7.5×10^{-3}	4.1×10^{-3}

3.4 Joint PMF Recovery

3.4.1 General Procedures

The key observation that enables our approach is that the marginal distribution of any subset of random variables is also a nonnegative CPD model. This is a direct consequence of the law of total probability. Marginalizing with respect to the k-th random variable we have that

$$\begin{aligned}\sum_{i_k=1}^{I_k} \mathsf{Pr}(i_1, \ldots, i_N) &= \sum_{r=1}^{R} \sum_{i_k=1}^{I_k} \mathsf{Pr}(r) \prod_{n=1}^{N} \mathsf{Pr}(i_n|r) \\ &= \sum_{r=1}^{R} \mathsf{Pr}(r) \prod_{\substack{n=1 \\ n\neq k}}^{N} \mathsf{Pr}(i_n|r) \sum_{i_k=1}^{I_k} \mathsf{Pr}(i_k|r) = \sum_{r=1}^{R} \mathsf{Pr}(r) \prod_{\substack{n=1 \\ n\neq k}}^{N} \mathsf{Pr}(i_n|r),\end{aligned}$$

since $\sum_{i_n=1}^{I_n} \mathsf{Pr}(i_n|r) = 1$.

Consider the model in (3.1) and assume that the marginal distributions $\mathsf{Pr}(X_j = i_j, X_k = i_k, X_\ell = i_\ell)$, denoted $\mathsf{Pr}(i_j, i_k, i_\ell)$ for brevity, $\forall j, k, \ell \in [N],\ \ell > k > j$ are available and perfectly known. Then, there exists an exact decomposition of the form

$$\mathsf{Pr}(i_j, i_k, i_\ell) = \sum_{r=1}^{R} \mathsf{Pr}(r)\mathsf{Pr}(i_j|r)\mathsf{Pr}(i_k|r)\mathsf{Pr}(i_\ell|r). \tag{3.5}$$

The marginal distributions of triples of random variables satisfy $\mathcal{X}_{jk\ell} = [\![\boldsymbol{\lambda}, \mathbf{A}_j, \mathbf{A}_k, \mathbf{A}_\ell]\!]$, where $\{\mathbf{A}_n\}_{n=1}^{N}$ and $\boldsymbol{\lambda}$ are defined as in (3.2) and they satisfy $\mathbf{A}_\ell \geq \mathbf{0}$, $\mathbf{A}_k \geq \mathbf{0}$, $\mathbf{A}_j \geq \mathbf{0}$, $\mathbf{1}^T\mathbf{A}_\ell = \mathbf{1}^T$, $\mathbf{1}^T\mathbf{A}_k = \mathbf{1}^T$, $\mathbf{1}^T\mathbf{A}_j = \mathbf{1}^T$, $\boldsymbol{\lambda} > \mathbf{0}$, $\mathbf{1}^T\boldsymbol{\lambda} = 1$. Based on the connection between the naive Bayes model of lower-dimensional marginals and the joint PMF, we propose the following steps to recover the complete joint PMF from three-dimensional marginals.

Procedure: Joint PMF Recovery From Triples

[S1] Estimate $\mathcal{X}_{jk\ell}$ from data;

[S2] Jointly factor $\mathcal{X}_{jk\ell} = [\![\boldsymbol{\lambda}, \mathbf{A}_j, \mathbf{A}_k, \mathbf{A}_\ell]\!]$ to estimate $\boldsymbol{\lambda}, \mathbf{A}_j, \mathbf{A}_k, \mathbf{A}_\ell \ \forall\ j,k,\ell$ using a CPD model with rank R;

[S3] Synthesize the joint PMF $\mathcal{X}$ via $\mathsf{Pr}(i_1, i_2, \ldots, i_N) = \sum_{r=1}^{R} \mathsf{Pr}(r) \prod_{n=1}^{N} \mathsf{Pr}(i_n|r)$, w/ $\mathsf{Pr}(i_n|r) = \mathbf{A}_n[i_n, r]$, $\mathsf{Pr}(r) = \boldsymbol{\lambda}[r]$.

One can see from step [S2], that if the individual factorization of at least one $\mathcal{X}_{jk\ell}$ is unique, then the joint PMF is readily identifiable via [S3]. This is already very interesting. However, as we will show in Sec. 3.5, we may identify the joint PMF even when the marginal tensors do not have unique CPD. The reason is that many marginal tensors share factors and we can exploit this to come up with much stronger identifiability results.

3.4.2 Algorithm: Coupled Matrix/Tensor Factorization

Before we discuss theoretical results such as identifiability of the joint PMF using three or higher-dimensional marginals, we first propose an implementation of [S2] in the proposed procedure. For brevity, we assume we have estimates of three-dimensional marginal distributions, i.e., we are given empirical estimates $\widehat{\mathsf{Pr}}(X_j = i_j, X_k = i_k, X_\ell = i_\ell),\ \forall j,k,\ell \in [N],\ \ell > k > j$, which we put in a tensor $\mathcal{X}_{jk\ell}$ i.e., $\mathcal{X}_{jk\ell}[i_j, i_k, i_\ell] = \widehat{\mathsf{Pr}}(X_j = i_j, X_k = i_k, X_\ell = i_\ell)$.

The method can be easily generalized to any type of low-dimensional marginal distributions. Under the assumption of a low-rank CPD model, every empirical marginal distribution of three random variables can be approximated as follows

$$\widehat{\mathsf{Pr}}(i_j, i_k, i_\ell) \approx \sum_{r=1}^{R} \mathsf{Pr}(r)\mathsf{Pr}(i_j|r)\mathsf{Pr}(i_k|r)\mathsf{Pr}(i_\ell|r). \tag{3.6}$$

Therefore, in order to compute an estimate of the full joint PMF, we propose solving the following optimization problem

$$\begin{aligned} \min_{\{\mathbf{A}_n\}_{n=1}^{N}, \boldsymbol{\lambda}} \quad & \sum_{j} \sum_{k>j} \sum_{\ell>k} \frac{1}{2} \left\| \mathcal{X}_{jk\ell} - [\![\boldsymbol{\lambda}, \mathbf{A}_j, \mathbf{A}_k, \mathbf{A}_l]\!] \right\|_F^2 \\ \text{subject to} \quad & \boldsymbol{\lambda} \geq \mathbf{0},\ \mathbf{1}^T \boldsymbol{\lambda} = 1, \\ & \mathbf{A}_n \geq \mathbf{0},\ n = 1, \ldots, N, \\ & \mathbf{1}^T \mathbf{A}_n = \mathbf{1}^T,\ n = 1, \ldots, N. \end{aligned} \tag{3.7}$$

Algorithm 3.1 Coupled Tensor Factorization Approach

Input: A discrete valued dataset $\mathbf{D} \in \mathbb{R}^{M \times N}$
Output: Estimates of $\{\mathbf{A}_n\}_{n=1}^N$ and $\boldsymbol{\lambda}$
Estimate $\mathcal{X}_{j,k,\ell} \quad \forall j,k,\ell \in [N],\ \ell > k > j$ from data.
Initialize $\{\mathbf{A}_n\}_{n=1}^N$ and $\boldsymbol{\lambda}$ such that the probability simplex constraints are satisfied.
repeat
  for all $n \in [N]$ **do**
    Solve optimization problem (3.9)
  end for
  Solve optimization problem (3.10)
until convergence criterion satisfied

The optimization problem in (3.7) is an instance of coupled tensor factorization. Coupled tensor/matrix factorization is usually used as a way of combining various datasets that share dimensions and corresponding factor matrices [1, 12]. Notice that in the case where we have estimates of two-dimensional marginals, the optimization problem in (3.7) corresponds to coupled matrix factorization. The optimization problem *per se* is very challenging and deserves developing sophisticated algorithms for handling it: first, when the number of random variables (N) gets large, there is a large number of optimization variables (i.e., $\{\mathbf{A}_n\}_{n=1}^N$) to be determined in (3.7) – and each $\mathbf{A}_n$ is an $I_n \times R$ matrix where I_n (the alphabet size of the n-th random variable) can be large. In addition, the probability simplex constraints impose some extra computational burden. Nevertheless, we found that, by carefully re-arranging terms, the formulated problem can be recast in convenient form and handled in a way that is reminiscent of the classical alternating least squares algorithm with constraints.

The idea is that we cyclically update variables $\{\mathbf{A}_n\}_{n=1}^N$ and $\boldsymbol{\lambda}$ while fixing the remaining variables at their last updated values. Assume that we fix estimates $\boldsymbol{\lambda}, \mathbf{A}_n, \forall n \in [N] \setminus \{j\}$. Then, the optimization problem with respect to $\mathbf{A}_j$ becomes

$$\begin{aligned} \min_{\mathbf{A}_j} \sum_{k \neq j} \sum_{\substack{\ell \neq j \\ \ell > k}} \quad & \frac{1}{2} \left\| \mathcal{X}_{jk\ell} - [\![\boldsymbol{\lambda}, \mathbf{A}_j, \mathbf{A}_k, \mathbf{A}_\ell]\!] \right\|_F^2 \\ \text{subject to} \quad & \mathbf{A}_j \geq \mathbf{0},\ \mathbf{1}^T \mathbf{A}_j = \mathbf{1}^T. \end{aligned} \tag{3.8}$$

Note that we have dropped the terms that do not depend on $\mathbf{A}_j$. By using the *mode*-1 *matrix* *unfolding* of each tensor $\mathcal{X}_{jk\ell}$, the problem can be equivalently written as

$$\begin{aligned} \min_{\mathbf{A}_j} \sum_{k\neq j} \sum_{\substack{\ell\neq j \\ \ell>k}} \quad & \frac{1}{2} \left\| \mathbf{X}_{jk\ell}^{(1)} - (\mathbf{A}_\ell \odot \mathbf{A}_k)\mathrm{diag}(\boldsymbol{\lambda})\mathbf{A}_j^T \right\|_F^2 \\ \text{subject to} \quad & \mathbf{A}_j \geq \mathbf{0},\ \mathbf{1}^T\mathbf{A}_j = \mathbf{1}^T, \end{aligned} \tag{3.9}$$

which is a least-squares problem with respect to matrix $\mathbf{A}_j$ under probability simplex constraints on its columns. The optimization problem has the same form for each factor $\mathbf{A}_n$ due to role symmetry. In order to update $\boldsymbol{\lambda}$ we solve the following optimization problem

$$\begin{aligned} \min_{\boldsymbol{\lambda}} \sum_{j} \sum_{k>j} \sum_{l>k} \quad & \frac{1}{2} \left\| \mathrm{vec}(\mathcal{X}_{jk\ell}) - (\mathbf{A}_\ell \odot \mathbf{A}_k \odot \mathbf{A}_j)\boldsymbol{\lambda} \right\|_2^2 \\ \text{subject to} \quad & \boldsymbol{\lambda} \geq \mathbf{0},\ \mathbf{1}^T\boldsymbol{\lambda} = 1. \end{aligned} \tag{3.10}$$

Both Problems (3.9) and (3.10) are linearly constrained quadratic programs, and can be solved to optimality by many standard solvers. Here, we propose to employ the *Alternating Direction Method of Multipliers* (ADMM) to solve these two sub-problems because of its flexibility and effectiveness in handling large-scale tensor decomposition [31, 47]. Details of the ADMM algorithm for solving Problems (3.9)-(3.10) can be found in the Appendix 3.8.4. The whole procedure is listed in Algorithm 3.1. As mentioned, the algorithm is easily modified to cover the cases where higher-dimensional marginals or pairwise marginals are given, and thus these cases are omitted.

3.5 Joint PMF Identifiability Analysis

In this section, we study the conditions under which we can identify $\mathsf{Pr}(i_1, \ldots, i_N)$ from marginalized lower-dimensional distributions. For brevity, we focus on three-dimensional as lower-dimensional distributions, and even though many more results are possible, we concentrate here on the case $I_n = I\ \forall n \in [N]$ for ease of exposition and manuscript length considerations. Similar type of analysis applies when $I_1, \ldots, I_N$ are different, however the analysis should be customized to properly address particular cases. Our aim here is to convey the spirit of what is possible in terms of identifiability results, as we cannot provide an exhaustive treatment (there are combinatorially many cases, clearly).

Obviously, if $\mathcal{X}_{jk\ell}$ is individually identifiable for each combination of j, k, ℓ, then, $\mathsf{Pr}(i_j|r)$, $\mathsf{Pr}(i_k|r)$, $\mathsf{Pr}(i_\ell|r)$, and $\mathsf{Pr}(r)$ are identifiable. This means that given three-dimensional marginal distributions, $\mathsf{Pr}(i_1, \ldots, i_N)$ is generically identifiable if $R \leq \frac{3I-2}{2}$ assuming that $I_n = I\ \forall n \in [N]$. This can be readily shown by invoking Theorem 2.1, equation (2.8), and the link between the naive Bayes model and tensor factorization discussed in Sec. 3.3. Note that $R \leq \frac{3I-2}{2}$ is already not a bad condition, since in many cases we have approximately low-rank tensors in practice. However, since we have many factor-coupled $\mathcal{X}_{jk\ell}$'s, this identifiability condition can be significantly improved. We have the following theorems.

Theorem 3.1. *Assume that* $\mathsf{Pr}(i_n|r),\ \forall n \in [N]$ *are drawn from an absolutely continuous distribution, that* $I_1 = \ldots = I_N = I$*, and that the joint PMF* $\mathsf{Pr}(i_1, \ldots, i_N)$ *can be represented using a naive Bayes model of rank* R*. If* $N \leq I$ *then,* $\mathsf{Pr}(i_1, \ldots, i_N)$ *is almost surely (a.s) identifiable from the* $\mathsf{Pr}(i_j, i_k, i_\ell)$*'s if*

$$R \leq I(N-2)$$

If $N > I$ *then,* $\mathsf{Pr}(i_1, \ldots, i_N)$ *is a.s. identifiable from the* $\mathsf{Pr}(i_j, i_k, i_\ell)$*'s if*

$$R \leq \left(\lfloor \frac{\sqrt{NI-1}}{I} \rfloor I - 1 \right)^2$$

The proof is relegated to Appendix 3.8.

Theorem 3.2. *Assume that* $\mathsf{Pr}(i_n|r),\ \forall n \in [N]$ *are drawn from an absolutely continuous distribution, that* $I_1 = \ldots = I_N = I$*, and that the joint PMF* $\mathsf{Pr}(i_1, \ldots, i_N)$ *can be represented using a naive Bayes model of rank* R*. Let* α *be the largest integer such that* $2^\alpha \leq \lfloor \frac{N}{3} \rfloor I$*. Then,* $\mathsf{Pr}(i_1, \ldots, i_N)$ *is a.s. identifiable from the* $\mathsf{Pr}(i_j, i_k, i_\ell)$*'s if*

$$R \leq 4^{\alpha-1}$$

which is implied by

$$R \leq \frac{(\lfloor \frac{N}{3} \rfloor I + 1)^2}{16}.$$

The proof is relegated to Appendix 3.8. The rank bounds in Theorems 3.1-3.2 are nontrivial, albeit far from the maximal attainable rank for the cases considered. Recalling that higher-order tensors are identifiable for higher ranks, a natural question is whether knowledge of four- or

Table 3.2: Rank bounds for generic identifiability ($I = 3$).

	Number of Variables (N)				
	6	10	20	40	80
Triples	4	7	27	105	410
Quadruples	10	36	179	729	2916

higher-dimensional marginals can further enhance identifiability of the complete joint PMF. The next theorem shows that the answer is affirmative.

Theorem 3.3. *Assume that* $\mathsf{Pr}(i_n|r),\ \forall n \in [N]$ *are drawn from an absolutely continuous distribution, that* $I_1 = \ldots = I_N = I$*, and that the joint PMF* $\mathsf{Pr}(i_1, \ldots, i_N)$ *can be represented using a naive Bayes model of rank* R*. Further assume that* $\mathcal{S} = [N]$ *can be partitioned into* 4 *disjoint subsets denoted by* $\mathcal{S}_1, \ldots, \mathcal{S}_4$ *such that the four-dimensional marginals* $\mathsf{Pr}(i_j, i_k, i_\ell, i_m),\ \forall j \in \mathcal{S}_1, \forall k \in \mathcal{S}_2, \forall l \in \mathcal{S}_3, \forall m \in \mathcal{S}_4$ *are available. Then, the joint PMF* $\mathsf{Pr}(i_1, \ldots, i_N)$ *is a.s. identifiable if*

$$\begin{aligned} &R \leq I^2|\mathcal{S}_3||\mathcal{S}_4|, \\ &2R(R-1) \leq I^2|\mathcal{S}_1||\mathcal{S}_2|(I|\mathcal{S}_1|-1)(I|\mathcal{S}_2|-1). \end{aligned}$$

The proof is relegated to Appendix 3.8.

The conditions of Theorem 3.3 are satisfied for much higher rank than those of Theorems 3.1-3.2 as shown in Tables 3.2-3.3. The results related to the four-dimensional marginals are obtained following Theorem 3.3 via checking all possible partitions. The caveat is that one may need many more samples to reliably estimate the four-dimensional marginals. Nevertheless, the theorems that we present in this section offer insights regarding the choice of lower-dimensional marginals to work with – such choice depends on the size of the alphabet of each variable (I) and the number of variables (N) as well as the amount of available data samples.

Remark 3.1. The above results rely on Theorems 2.2, 2.3 and concern the identifiability of a generic choice of parameters; i.e., the parameters are assumed to be drawn randomly from a jointly continuous distribution. At this point one may wonder whether this is a realistic assumption in practice. For example, in some latent model identification problems a hidden variable has specific physical meaning and an observed variable may not depend on the state of

Table 3.3: Rank bounds for generic identifiability ($N = 6$).

	Alphabet size (I)				
	6	10	20	40	80
Triples	24	40	105	410	1620
Quadruples	45	131	544	2220	8966

the hidden variable for one or more of its values. Consider a Hidden Markov Model (HMM) where we denote the observed variable at time t as X_t and the hidden state is S_t. The conditional distribution $\widetilde{\mathbf{A}}_t[i, s] := \mathsf{Pr}(X_t = i|S_t = s)$ may be the same for two different values s_1 and s_2 of the hidden state S_t. In such a case, the Kruskal rank of matrix $\widetilde{\mathbf{A}}_t$ would be equal to 1, thereby rendering the deterministic identifiability condition (Theorem 2.1) useless. Do note, however, that in our setting the latent variable H does not necessarily have a physical interpretation; the CPD is just a convenient 'universal' parametrization of the joint PMF. Therefore the conditional distribution of an observed variable may be the same for two values of the hidden state, but it may still depend on the value of the 'virtual' global latent variable H, and hence recovery of the the joint PMF using lower-dimensional marginals could still be possible.

3.6 Numerical Results

In this section, we employ judiciously designed synthetic data simulations to showcase the effectiveness of the proposed joint PMF recovery methods. We also apply the approach to real-data problems such as classification and recommender systems to demonstrate its usefulness in real machine learning tasks.

3.6.1 Synthetic-Data Simulations

We first evaluate the proposed approach using synthetic data. We consider a case where $N = 5$ random variables are present, and each variable can take $I_n = 10$ discrete values. We assume that the joint PMF of the 5 random variables can be represented by a naive Bayes model whose latent variable H can take R values, where R is set to be $\{5, 10, 15\}$. We generate matrices $\mathbf{A}_n \in \mathbb{R}_+^{I_n \times R}$, that model the conditional probabilities i.e., $\mathbf{A}_n[i_n, r] = \mathsf{Pr}(i_n|r)$. A vector $\boldsymbol{\lambda} \in \mathbb{R}_+^R$ is also generated for the latent random variable H such that $\boldsymbol{\lambda}[r] = \mathsf{Pr}(r)$. The elements of each $\mathbf{A}_n$ and the vector $\boldsymbol{\lambda}$ are drawn independently from a uniform distribution between zero and one, and each column is normalized to sum to 1. The ground-truth joint PMF is then constructed following the naive Bayes model, i.e., $\mathsf{Pr}(i_1, \ldots, i_5) = \mathcal{X}[i_1, \ldots, i_5] = \sum_{r=1}^{R} \boldsymbol{\lambda}[r] \prod_{n=1}^{5} \mathbf{A}_n[i_n, r]$. We assume that the observable data are two-, three- and four-dimensional marginals of the joint PMF. Under such settings, we can verify if the proposed procedure and algorithm can effectively recover the joint PMF, if there is no modeling error and the joint PMF does have low rank. We run 20 Monte Carlo simulations and compute the mean relative error of the factors as well as the mean relative error of the recovered tensor which are defined as follows

$$\text{MRE}_{\text{fact}} = \mathbb{E}\left(\frac{1}{N}\sum_{n=1}^{N}\frac{\|\mathbf{A}_n - \widehat{\mathbf{A}}_n\boldsymbol{\Pi}\|_F}{\|\mathbf{A}_n\|_F}\right),$$

$$\text{MRE}_{\text{ten}} = \mathbb{E}\left(\frac{\|\mathcal{X} - \widehat{\mathcal{X}}\|_F}{\|\mathcal{X}\|_F}\right),$$

where $\boldsymbol{\Pi}$ is a permutation matrix to fix the permutation ambiguity, and $\widehat{\mathcal{X}}$, $\widehat{\mathbf{A}}_n$ are the estimated joint PMF and the corresponding conditional PMFs.

Table 3.4 shows the mean relative errors for estimating the conditional PMFs and joint PMF using the different types of input under different choices of rank. Consistent with our analysis, one can see that using marginal distributions of triples or quadruples of random variables (i.e., three- and four-dimensional marginals) we are able to recover the joint PMF of 5 random variables. Here recovery with high accuracy has been demonstrated; exact recovery is also possible in certain cases, see [98]. However, using pairs (i.e., two-dimensional marginals) is not as promising. Recall that our identifiability result is built upon the identifiability of third- and higher-order CPD models. These nice identifiability results in general do not hold for matrices – which explains the sharp performance difference between using the pairs and higher-dimensional marginals. Pairs can work, however, when the conditional probability matrices are sufficiently sparse, and under more stringent constraints on the rank R.

The above simulation serves as sanity check – if the available data and the model perfectly match with each other and we have noiseless marginal distributions, the proposed approach can indeed recover the joint PMF. In practice, we usually do not have exact estimates of the lower-dimensional marginal distributions. Next, we provide a set of more realistic simulations where we estimate the marginal PMFs using sample averages from the observed data.

Table 3.4: Mean relative factor and tensor error when lower-dimensional marginals are perfectly known.

Rank		MRE_{fact}	MRE_{ten}
	Pairs	0.277	0.148
$R = 5$	Triples	1.18×10^{-7}	4.58×10^{-8}
	Quadruples	3.39×10^{-8}	1.19×10^{-8}
	Pairs	0.440	0.187
$R = 10$	Triples	3.58×10^{-7}	8.70×10^{-8}
	Quadruples	1.26×10^{-7}	2.58×10^{-8}
	Pairs	0.466	0.184
$R = 15$	Triples	6.77×10^{-7}	1.52×10^{-7}
	Quadruples	1.78×10^{-7}	3.57×10^{-8}

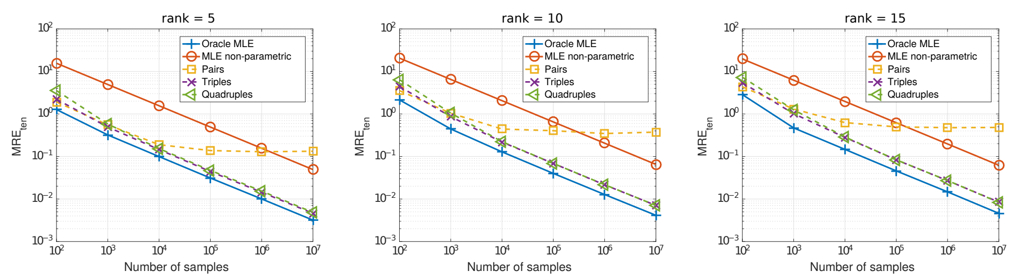


Figure 3.6: Mean relative error of the estimated joint PMF under different number of available samples (fully observed data).

Fully-observed data

We follow the same way of generating the ground-truth joint PMF as in the previous simulation. Then, drawing from the joint PMF, we generate a synthetic dataset of M five-dimensional data points. The data is generated as follows: for each data point, we first draw a sample h_m according to $\boldsymbol{\lambda}$; i.e., a realization of the hidden variable H. Then the data point (vector) $\mathbf{s}_m = [\mathbf{s}_m[1], \ldots, \mathbf{s}_m[N]]^T$ is generated by drawing its elements independently from $\{\mathbf{A}_n\}[:, h_m]_{n=1}^N$, i.e., $\mathbf{s}_m[n]$ is drawn from $\{\mathbf{A}_n\}[:, h_m]$. This is equivalent to synthesizing the five-way joint PMF tensor and drawing an outcome from it (cf. the naive Bayes interpretation).

We use the generated 5-dimensional data points to estimate lower-dimensional marginals and run our ADMM algorithm to recover the full joint PMF. We repeat for a total of 10 Monte Carlo simulations. Figure 3.6 shows the tensor mean relative error of the estimated joint PMF under different dataset sizes M. We also include the performance of two additional methods for

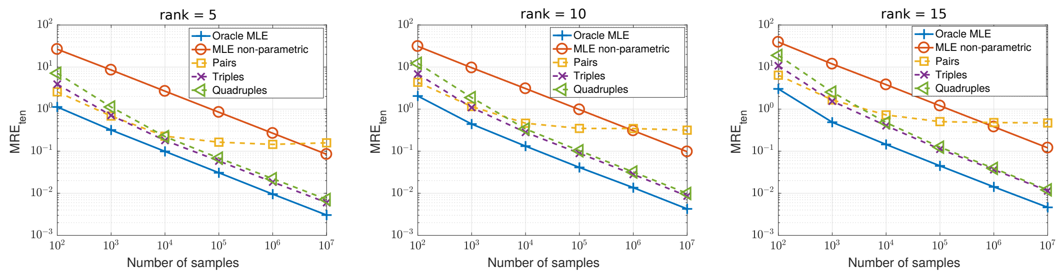


Figure 3.7: Mean relative error of the estimated joint PMF under different number of available samples.

estimating the joint PMF. Given the full data together with 'oracle' observations of the hidden variable H, we perform Maximum Likelihood Estimation (MLE) of the naive Bayes parameters, denoted as oracle MLE, which is done simply by frequency counting

$$\mathsf{Pr}(r)_{ML} = \frac{\text{count}(r)}{M}, \ \mathsf{Pr}(i_n|r)_{ML} = \frac{\text{count}(i_n, r)}{\text{count}(r)},$$

where $\text{count}(i_n, r)$ is the number of times that $X_n = i_n$ and $H = r$ appear together in the dataset and $\text{count}(\text{r})$ the number of times H takes the value r. We also include the MLE of a non-parametric approach in which we use the empirical 5-dimensional distribution as our estimate i.e.,

$$\mathsf{Pr}(i_1, \ldots, i_N)_{ML} = \frac{\text{count}(i_1, \ldots, i_N)}{M}.$$

One can see that the estimation performance of our method is similar under different rank values and approaches that of oracle MLE using three- and four-dimensional marginals. In addition, as expected, the recovery accuracy steadily improves as the size of the available dataset increases. On the other hand, when using two-dimensional marginals the performance improves until it reaches a plateau at approximately $M = 10^5$. The ability to recover the joint PMF using pairs of random variables is obviously limited by identifiability of matrix factorization.

Missing data

We repeat the above experiment when some of the dataset entries are missing. We randomly hide 20% of the data and compute estimates of two- three- and four-dimensional marginals using only the available data. We run the ADMM-based algorithm and repeat for 10 Monte Carlo simulations (Fig 3.7). The estimation performance of our method is again similar under different rank values and approaches that of oracle MLE. As expected, we observe a slight decrease in performance which is due to the less accurate estimation of the lower-dimensional marginals. In this case, the MLE non-parametric method takes into account only the fully observed samples.

Note that when empirical estimates of the lower-dimensional distributions are used and the number of samples is limited, three-dimensional distributions may give lower relative error compared to the four-dimensional ones. This shows that in some cases using lower-dimensional distributions can be more beneficial than higher-order ones in terms of parameter estimation accuracy. Actually, this is not very surprising since it can be shown that empirical lower-dimensional marginals are always more accurate than higher-dimensional ones when estimated given the same data [45].

3.6.2 Real-Data Experiments

In real applications, the ground-truth joint PMF and the conditional PMFs are not known. Nevertheless, we can evaluate the method on a variety of standard machine learning tasks to observe its effectiveness. In this subsection, we test the proposed approach on two different tasks, namely, data classification and recommender systems. Note that both tasks can be easily accomplished if the joint PMF of pertinent variables (e.g., features and labels in classification) is known and thus are suitable for evaluating our method. Note that the rank of the joint PMF tensor, or, $R = |\mathcal{H}|$, cannot be known as in the simulations. Fortunately, this is a single discrete variable that can be easily tuned, e.g., via observing validation errors as in machine learning.

Classification Task

We evaluate the performance of our approach on 7 different datasets from the UCI machine learning repository [76]. Five of the selected datasets correspond to binary classification and two to multi-class classification. For each dataset, we represent the training samples using its discrete features so that the PMF-based approach can be applied. We split each dataset such that 70% of the data samples is used for training, 10% used for validation and 20% for testing.

For each dataset, we let X_N be the label and $X_1, \ldots, X_{N-1}$ be the selected features. We estimate lower-dimensional marginal distributions of pairs, triples and quadruples of variables using the samples in the training set. Then, we use the marginals to estimate $\mathsf{Pr}(i_n|r)$, and $\mathsf{Pr}(r)$.

Table 3.5: Dataset information.

Dataset	N	R
INCOME	8	$[1, 20]$
CREDIT	9	$[1, 20]$
HEART	9	$[1, 10]$
MUSHROOM	22	$[1, 20]$
VOTES	17	$[1, 10]$
CAR	7	$[1, 15]$
NURSERY	9	$[1, 15]$

After applying the proposed approach for estimating the joint PMF, we predict for each data point of the test set the corresponding label using the MAP rule. The MAP estimator of the label $l(\mathbf{s}_m)$ of the m-th observation $\mathbf{s}_m = [\mathbf{s}_m[1], \ldots, \mathbf{s}_m[N-1]]^T$ in the test set can be written as

$$\widehat{l}_{\text{map}}(\mathbf{s}_m) = \underset{i_N \in \{1, \ldots, I_N\}}{\arg\max} \; \mathsf{Pr}(i_N | \mathbf{s}_m[1], \ldots, \mathbf{s}_m[N-1]),$$

where I_N is the number of classes. Equivalently, using the Bayes rule the above can be found by

$$\widehat{l}_{\text{map}}(\mathbf{s}_m) = \underset{i_N \in \{1, \ldots, I_N\}}{\arg\max} \sum_{r=1}^{R} \mathsf{Pr}(r) \mathsf{Pr}(i_N | r) \prod_{n=1}^{N-1} \mathsf{Pr}(\mathbf{s}_m[n] | r).$$

For each dataset, we run 10 Monte Carlo simulations with randomly partitioned training, validation and test sets and observe the average result. As mentioned, we do not know *a priori* what is an appropriate rank for our model. Therefore, for each dataset, we fit models of different rank values and choose the one which minimizes the misclassification error of the validation set as in standard machine learning practice.

Table 3.6: Misclassification error on different UCI datasets.

	Binary				
Method	INCOME	CREDIT	HEART	MUSHROOM	VOTES
CP (Pairs)	0.177±0.004	0.134±0.019	0.151±0.023	0.010±0.007	0.046±0.024
CP (Triples)	0.175±0.003	0.129±0.018	0.147±0.031	0.006±0.002	0.043±0.024
CP (Quadruples)	**0.171**±0.003	**0.123**±0.018	**0.138**±0.029	0.002±0.001	0.042±0.020
SVM (Linear)	0.179±0.004	0.146±0.027	0.170±0.053	**0**±0	**0.038**±0.025
SVM (RBF)	0.174±0.004	0.136±0.018	0.187±0.055	**0**±0	0.079±0.024
Naive Bayes	0.209±0.005	0.140±0.018	0.166±0.026	0.044±0.005	0.096±0.022

Table 3.7: Misclassification error on different UCI datasets.

	Multiclass	
Method	CAR	NURSERY
CP (Pairs)	0.128±0.021	0.101±0.009
CP (Triples)	0.089±0.016	0.069±0.011
CP (Quadruples)	0.074±0.015	0.061±0.007
SVM (Linear)	0.065±0.006	0.063±0.004
SVM (RBF)	**0.026**±0.008	**0.006**±0.001
Naive Bayes	0.151±0.016	0.097±0.007

We use 3 different classical classifiers from the MATLAB Statistics and Machine Learning Toolbox as baselines; linear SVM, kernel SVM with radial basis function and a naive Bayes classifier. For SVM classifiers, we use both the original data encoding as well as the one-hot encoding which usually is more suitable for discrete data and report the best result among the two. Note that the baseline naive Bayes approach is very different from ours: the baseline method assumes that the features are independent given the label, while we assume that the label and the features are independent given an unknown latent variable. The former is a very strong assumption that is rarely satisfied by real data, but our assumption holds for an arbitrary set of random variables provided R is large enough, as we showed in Proposition 3.1.

Table 3.6 shows the classification errors obtained on the datasets. One can see that our approach outperforms the naive Bayes classifier which assumes that the features are independent given the label. Several observations are in order. First, using higher-dimensional marginals, the proposed approach gives better classification results compared to using lower-dimensional ones. This is consistent with our analysis in Sec. 3.5 – higher-dimensional marginals lead to stronger overall identifiability of the joint PMF. One can see that for all the datasets under test, using four-dimensional marginal distributions gives the best classification accuracy compared to the three- and two-dimensional ones. Second, for the five binary classification experiments, the proposed method works better than (on three datasets) or comparable to the baselines. This is quite surprising since our method does not directly optimize a classification criterion as SVM does. The result suggests that the proposed method indeed captures the essence of the joint distribution and the recovered joint PMF can be utilized to make inference in practice. Third, for the multiclass datasets, the proposed method yields accuracy that is less than the SVM methods. This also makes sense: when X_N has a value set whose cardinality grows from 2 to 5, the joint PMFs of X_N and X_i, X_j for $i, j < N$ require more samples to estimate accurately.

This also shows an interesting sample complexity-accuracy trade-off of the proposed method. Nevertheless, the method still works comparably well with the linear SVM, which supports the usefulness of the joint PMF estimation method.

Recommender Systems

We also evaluate the method for the task of recommender systems using the MovieLens dataset [39]. MovieLens is a dataset that contains ratings on 5-star scale, with half-star increments, by a number of users. In order to test our algorithm we select three different subsets of the full dataset and round the ratings to the next integer. Initially, three different categories (action, animation and romance) are selected. From each category we extract a user-by-rating submatrix by keeping the 20 most rated movies and form the 3 datasets for our experiments. Note that the constructed three datasets have many missing values, since not all users watched and rated all movies. The task of recommender systems is to recommend unwatched movies to users based on prediction of the user's rating given the available data.

Table 3.8: RMSE and MAE of different algorithms on MovieLens dataset.

	MovieLens Dataset 1		**MovieLens Dataset 2**		**MovieLens Dataset 3**	
Method	RMSE	MAE	RMSE	MAE	RMSE	MAE
CP (Pairs)	0.802±0.003	0.608±0.003	0.795±0.002	0.611±0.002	0.897±0.003	0.702±0.002
CP (Triples)	0.783±0.002	0.591±0.002	**0.785 ±0.002**	**0.599 ±0.002**	0.887±0.002	0.691±0.002
CP (Quad.)	**0.778±0.002**	**0.588±0.002**	0.786±0.002	0.600±0.002	**0.884 ±0.002**	**0.689 ±0.002**
Global Avg	0.945±0.001	0.693±0.001	0.906±0.002	0.653±0.002	0.996±0.002	0.798±0.001
User Avg	0.879±0.002	0.679±0.001	0.830±0.003	0.625±0.002	1.010±0.002	0.768±0.002
Movie Avg	0.886±0.002	0.705±0.001	0.889±0.002	0.673±0.002	0.942±0.002	0.754±0.001
BMF	0.797±0.002	0.623±0.002	0.792±0.002	0.604±0.002	0.904±0.003	0.701±0.003

We aim at estimating the joint PMF of the movie ratings. In this case, each random variable X_n represents a movie, and it takes values from $\{1, \ldots, 5\}$, i.e., the ratings. Consequently, the joint PMF is a twenty-way tensor which has 5^{20} elements. The 3 partially observed datasets are used in order to estimate lower-dimensional marginal distributions of pairs, triples and quadruples of the variables (movies). We use the estimated PMF to compute the expected value of users' ratings that we do not observe given the ones we observe. More specifically, let $\mathbf{s}_m = [\mathbf{s}_m[1], \ldots, \mathbf{s}_m[N]]$ be the ratings of the m-th user and $\mathbf{s}_m[N] = 0$ i.e., the user has a missing rating. The conditional expectation of the movie's rating is given by

$$\widehat{s}_N = \sum_{i_N=1}^{I_N} i_N \mathsf{Pr}(i_N | \mathbf{s}_m[1], \ldots, \mathbf{s}_m[N-1]).$$

As a baseline algorithm, we use the Biased Matrix Factorization (BMF) method [68], which is a commonly used method in recommender systems. The BMF method is essentially low-rank matrix completion with modifications. Additionally, we present results obtained by global average of the ratings, the user average, and the item average as baselines for predicting the missing entries. For each dataset we randomly hide 20% ratings that we use as a test set, 10% ratings that we use as a validation set and the remaining dataset is used as a training set. We run 10 Monte Carlo simulations using our approach and the BMF algorithm. We select the parameters of both methods based on the RMSE of the validation set.

Table 3.8 shows the performance of the two algorithms in terms of the RMSE and Mean Absolute Error (MAE). One can see that, for the three datasets under test, the proposed method and the BMF method output clearly lower RMSEs and MAEs relative to the naive methods using averaging. In addition, the proposed method slightly outperforms BMF on all of the three datasets. Note that BMF is considered a state-of-art method for movie recommendation, and it incorporates application-specific custom features, such as user bias and movie bias to achieve good performance. The proposed method, on the other hand, only uses basic probability to handle the same task – it is completely application-blind. This suggests that the joint PMF modeling and the proposed algorithm are both quite effective. Last, we also observe accuracy improvement when we increase the dimension of the marginal distributions used in the approach. Again, this performance may come from the identifiability gain as we analyzed in Theorem 3.3.

3.7 Chapter Summary

We have taken a fresh look at one of the most fundamental problems in statistical learning – joint PMF estimation. Due to the curse of dimensionality, naive count-based estimation is mission impossible in most cases. One popular approach has historically been to assume a plausible structural model, such as a Markov chain, tree, or other probabilistic graphical model, and do inference using this model. We showed that a very different 'non-parametric' tensor-based approach is possible, and it features several key benefits. Foremost among them is guaranteed identifiability of the high-dimensional joint PMF from low-dimensional marginals, which can be reliably estimated using counting from much fewer examples, even if there are (many) samples missing from each example. This ability to infer a unique higher-dimensional joint PMF by specifying lower-dimensional ones is reminiscent of Kolmogorov extension, which is intuitively very pleasing.

We have also proven two more results that appear fundamental and close to the heart of probability and learning theory: i) *every* joint PMF can be *interpreted* as a naive Bayes model; and ii) probabilistic graphical models, which are very popular in statistical and computer science, are never identifiable if one simply limits the number of hidden nodes; one needs to bound the number of hidden states as well. Our non-parametric approach can reveal the true hidden structure, instead of assuming it; and this alleviates the risk of up-front bias in the analysis.

On the practical side, our approach is appealing since lower-dimensional marginals can be more reliably estimated from a limited amount of partially observable data. We have also provided a practical and easily implementable algorithm that is based on factor-coupled tensor factorization to handle the recovery problem. Simulations and judicious experiments with real data have shown that the performance of the proposed approach is consistent with our analysis, and approaches or exceeds that of state-of-art application-specific solutions that have come out after years of intensive research, which is satisfying.

3.8 Appendix

3.8.1 Proof of Theorem 3.1

Each three-dimensional marginal satisfies $\mathcal{X}_{jk\ell} = [\![\boldsymbol{\lambda}, \mathbf{A}_j, \mathbf{A}_k, \mathbf{A}_\ell]\!]$, where $\mathbf{A}_\ell \geq \mathbf{0}$, $\mathbf{A}_k \geq \mathbf{0}$, $\mathbf{A}_j \geq \mathbf{0}$, $\mathbf{1}^T\mathbf{A}_\ell = \mathbf{1}^T$, $\mathbf{1}^T\mathbf{A}_k = \mathbf{1}^T$, $\mathbf{1}^T\mathbf{A}_j = \mathbf{1}^T$, $\boldsymbol{\lambda} > \mathbf{0}$, $\mathbf{1}^T\boldsymbol{\lambda} = 1$. Consider a partition of the set $\mathcal{S} = [N]$ into three disjoint sets $\mathcal{S}_1, \mathcal{S}_2, \mathcal{S}_3$ and define the following factors

$$\begin{aligned}
\widehat{\mathbf{A}}_1 &= [\mathbf{A}_{u_1}^T, \cdots, \mathbf{A}_{u_{|\mathcal{S}_1|}}^T]^T \\
\widehat{\mathbf{A}}_2 &= [\mathbf{A}_{v_1}^T, \cdots, \mathbf{A}_{v_{|\mathcal{S}_2|}}^T]^T \\
\widehat{\mathbf{A}}_3 &= [\mathbf{A}_{w_1}^T, \cdots, \mathbf{A}_{w_{|\mathcal{S}_3|}}^T]^T
\end{aligned} \tag{3.11}$$

with $u_t \in \mathcal{S}_1$, $v_t \in \mathcal{S}_2$, $w_t \in \mathcal{S}_3$. Then, we can construct a single virtual nonnegative CPD model

$$\widehat{\mathbf{X}}^{(1)} = (\widehat{\mathbf{A}}_3 \odot \widehat{\mathbf{A}}_2)\mathrm{diag}(\boldsymbol{\lambda})\widehat{\mathbf{A}}_1^T, \tag{3.12}$$

where $\widehat{\mathbf{A}}_1 \in \mathbb{R}_+^{I|\mathcal{S}_1|\times R}$, $\widehat{\mathbf{A}}_2 \in \mathbb{R}_+^{I|\mathcal{S}_2|\times R}$, $\widehat{\mathbf{A}}_3 \in \mathbb{R}_+^{I|\mathcal{S}_3|\times R}$ and $\widehat{\mathcal{X}} \in \mathbb{R}_+^{I|\mathcal{S}_1|\times I|\mathcal{S}_2|\times I|\mathcal{S}_3|}$. We have used a subset of the available information to synthesize a virtual single nonnegative CPD model of size $I_1 \times I_2 \times I_3$, with $I_k := I|\mathcal{S}_k|$. Therefore, we can apply identifiability results of three-way tensors. We observe that the sizes of the different modes of the virtual tensor depend on the way we partition the variables. We distinguish between two cases and apply Theorem 2.2.

1. $(N \leq I)$: We partition the variables into three sets such that $I_1 = I, I_2 = I$ and $I_3 = (N-2)I$. Clearly we have that $I_3 > I_2, I_1$ and $I_3 < (I_1-1)(I_2-1)$. According to Lemma 2.2 tensor $\widehat{\mathcal{X}}$ admits unique decomposition for $R \leq \min\left(I_3, (I_1-1)(I_2-1)\right) = (N-2)I$.

2. $(N > I)$: In this case we can partition the variables into three sets such that $I_3 = (I_1 - 1)(I_2 - 1)$. We can always have that $I_1 = I_2$. The equality is satisfied when $|\mathcal{S}_1| = |\mathcal{S}_2| = \frac{\sqrt{(NI-1)}}{I}$. According to Lemma 2.2 tensor $\widehat{\mathcal{X}}$ admits unique decomposition for $R \leq \min(I_3, (I_1-1)(I_2-1)) \leq (\lfloor\frac{\sqrt{(NI-1)}}{I}\rfloor I - 1)^2$.

3.8.2 Proof of Theorem 3.2

As above, but this time choosing $|\mathcal{S}_1| = |\mathcal{S}_2| = \lfloor\frac{N}{3}\rfloor$, $|S3| = N - 2|S1|$ and invoking Theorem 2.3.

3.8.3 Proof of Theorem 3.3

Consider a partitioning the variables into four disjoint sets similar to the three-way case. We obtain a single nonnegative CPD of the following form

$$\widehat{\mathbf{X}}^{(1)} = (\widehat{\mathbf{A}}_4 \odot \widehat{\mathbf{A}}_3 \odot \widehat{\mathbf{A}}_2)\mathrm{diag}(\boldsymbol{\lambda})\widehat{\mathbf{A}}_1^T,$$

which is a fourth-order tensor $\widehat{\mathcal{X}} \in \mathbb{R}_+^{I|\mathcal{S}_1|\times I|\mathcal{S}_2|\times I|\mathcal{S}_3|\times I|\mathcal{S}_4|}$. A fourth-order tensor can be viewed as a third-order tensor of size $I|\mathcal{S}_1| \times I|\mathcal{S}_2| \times I^2|\mathcal{S}_4||\mathcal{S}_3|$ with a specially structured factor matrix. We can write the mode-1 unfolding of the tensor $\widehat{\mathcal{X}}$ as

$$\widehat{\mathbf{X}}^{(1)} = (\bar{\mathbf{A}}_3 \odot \widehat{\mathbf{A}}_2)\mathrm{diag}(\boldsymbol{\lambda})\widehat{\mathbf{A}}_1^T, \tag{3.13}$$

where $\bar{\mathbf{A}}_3 = \widehat{\mathbf{A}}_4 \odot \widehat{\mathbf{A}}_3$. Lemmas 2.2-2.3 cannot be applied in this case because of the Khatri-Rao structure of one factor. We will use the following result

Lemma 3.1. *[52] Let $\mathcal{X} = [\![\mathbf{A}_1, \mathbf{A}_2, \mathbf{A}_3]\!]$, where $\mathbf{A}_1 \in \mathbb{R}^{I_1 \times R}$, $\mathbf{A}_2 \in \mathbb{R}^{I_2 \times R}$, $\mathbf{A}_3 \in \mathbb{R}^{I_3 \times R}$. If rank$(\mathbf{A}_3) = R$ and rank$\big(\mathbf{M}_2(\mathbf{A}_1) \odot \mathbf{M}_2(\mathbf{A}_2)\big) = \binom{R}{2}$, then rank$(\mathcal{X})$ = R and the decomposition is essentially unique.*

$\mathbf{M}_2(\mathbf{A})$ denotes the $\binom{I}{2} \times \binom{R}{2}$ compound matrix containing all 2×2 minors of $\mathbf{A}$. The generic version of Lemma 3.1 states that if $R \leq I_3$ and $2R(R-1) \leq I_1(I_1-1)I_2(I_2-1)$, then the decomposition of $\mathcal{X}$ is essentially unique, a.s. [105]. We know that a Khatri-Rao product of two matrices is full rank almost surely [51, Corollary 1]. For the three-way tensor in (3.13) we have $I_1 = I|\mathcal{S}_1|$, $I_2 = I|\mathcal{S}_2|$ and $I_3 = I^2|\mathcal{S}_3||\mathcal{S}_4|$. Applying the generic version of Lemma 3.1 we obtain the desired result.

3.8.4 Algorithm

We reformulate optimization problem (3.9) by introducing an auxiliary variable $\widehat{\mathbf{A}}_j$. The problem can be equivalently written as

$$\begin{aligned} &\min_{\mathbf{A}_j, \widehat{\mathbf{A}}_j} \; f(\widehat{\mathbf{A}}_j) + r(\mathbf{A}_j) \\ &\text{subject to } \mathbf{A}_j = \widehat{\mathbf{A}}_j^T \end{aligned} \tag{3.14}$$

where

$$f(\widehat{\mathbf{A}}_j) = \sum_{k \neq j} \sum_{\substack{\ell \neq j \\ \ell > k}} \frac{1}{2} \left\| \mathbf{X}_{jk\ell}^{(1)} - (\mathbf{A}_\ell \odot \mathbf{A}_k)\text{diag}(\boldsymbol{\lambda})\widehat{\mathbf{A}}_j \right\|_F^2,$$

$r(\mathbf{A})$ is the indicator function for the probability simplex constraints $\mathcal{C} = \{\mathbf{A} \mid \mathbf{A} \geq 0, \mathbf{1}^T\mathbf{A} = \mathbf{1}^T\}$

$$r(\mathbf{A}) = \begin{cases} 0, & \mathbf{A} \in \mathcal{C} \\ \infty, & \mathbf{A} \notin \mathcal{C} \end{cases} .$$

Optimization problem (3.14) can be readily solved by applying the ADMM algorithm. We solve for $\mathbf{A}_j$ by performing the following updates

$$\begin{aligned}
\widehat{\mathbf{A}}_j^{(\tau+1)} &= (\mathbf{G}_j + \rho\mathbf{I})^{-1}\left(\mathbf{V}_j + \rho\left(\mathbf{A}_j^{(\tau)} + \mathbf{U}_j^{(\tau)}\right)^T\right), \\
\mathbf{A}_j^{(\tau+1)} &= \mathcal{P}_{\mathcal{C}}\left(\mathbf{A}_j^{(\tau)} - \widehat{\mathbf{A}}_j^{(\tau+1)T} + \mathbf{U}_j^{(\tau)}\right), \\
\mathbf{U}_j^{(\tau+1)} &= \mathbf{U}_j^{(\tau)} + \mathbf{A}_j^{(\tau+1)} - \widehat{\mathbf{A}}_j^{(\tau+1)T},
\end{aligned}$$

where

$$\begin{aligned}
\mathbf{G}_j &= (\boldsymbol{\lambda}\boldsymbol{\lambda}^T) \circledast \sum_{k\neq j}\sum_{\substack{\ell\neq j\\ \ell>k}} \mathbf{Q}_{\ell k}^T\mathbf{Q}_{\ell k}, \\
\mathbf{V}_j &= \mathrm{diag}(\boldsymbol{\lambda}) \sum_{k\neq j}\sum_{\substack{\ell\neq j\\ \ell>k}} \mathbf{Q}_{\ell k}^T\mathbf{X}_{jk\ell}^{(1)}, \\
\mathbf{Q}_{\ell k} &= \mathbf{A}_\ell \odot \mathbf{A}_k.
\end{aligned}$$

$\mathcal{P}_{\mathcal{C}}$ is the projection operator onto the convex set $\mathcal{C}$ and τ denotes the iteration index. Various methods exist for projecting onto the probability simplex, e.g., see [114]. Note that in order to efficiently compute matrix $\mathbf{G}_j$ we use a property of the Khatri-Rao product; $\mathbf{Q}_{\ell k}^T\mathbf{Q}_{\ell k} = (\mathbf{A}_\ell^T\mathbf{A}_\ell) \circledast (\mathbf{A}_k^T\mathbf{A}_k)$. Efficient algorithms also exist for the computation of matrix $\mathbf{V}_j$ which is a sum of Matricized Tensor Times Khatri-Rao Product (MTTKRP) terms [8, 102]. Similarly, we can derive updates for $\boldsymbol{\lambda}$. At each ADMM iteration we perform the following updates

$$\begin{aligned}
\widehat{\boldsymbol{\lambda}}^{(\tau+1)} &= (\mathbf{G} + \rho\mathbf{I})^{-1}\left(\mathbf{V} + \rho\left(\boldsymbol{\lambda}^{(\tau)} + \mathbf{u}^{(\tau)}\right)\right), \\
\boldsymbol{\lambda}^{(\tau+1)} &= \mathcal{P}_{\mathcal{C}}\left(\boldsymbol{\lambda}^{(\tau)} - \widehat{\boldsymbol{\lambda}}^{(\tau+1)} + \mathbf{u}^{(\tau)}\right), \\
\mathbf{u}^{(\tau+1)} &= \mathbf{u}^{(\tau)} + \boldsymbol{\lambda}^{(\tau+1)} - \widehat{\boldsymbol{\lambda}}^{(\tau+1)}.
\end{aligned}$$

where

$$\begin{aligned}
\mathbf{G} &= \sum_j\sum_{k>j}\sum_{\ell>k} \mathbf{Q}_{\ell kj}^T\mathbf{Q}_{\ell kj}, \\
\mathbf{V} &= \sum_j\sum_{k>j}\sum_{\ell>k} \mathbf{Q}_{\ell kj}^T\mathrm{vec}(\mathcal{X}_{jk\ell}), \\
\mathbf{Q}_{\ell kj} &= \mathbf{A}_\ell \odot \mathbf{A}_k \odot \mathbf{A}_j.
\end{aligned}$$

Chapter 4

Learning Mixtures of Smooth Product Distributions

4.1 Introduction

Learning mixture models is a fundamental problem in statistics and machine learning having numerous applications such as density estimation and clustering. In this chapter, we consider the special case of mixture models whose component distributions factor into the product of the associated marginals. An example is a mixture of axis-aligned Gaussian distributions, an important class of Gaussian Mixture Models (GMMs). Consider a scenario where different diagnostic tests are applied to patients, and test results are assumed to be independent conditioned on the binary disease status of the patient which is the latent variable. The joint Probability Density Function (PDF) of the tests can be expressed as a weighted sum of two components, and each component factors into the product of univariate marginals. Fitting a mixture model to an unlabeled dataset allows us to cluster the patients into two groups by determining the value of the latent variable using the Maximum a Posteriori (MAP) principle.

Most of the existing literature in this area has focused on the fully-parametric setting, where the mixture components are members of a parametric family, such as Gaussian distributions. The most popular algorithm for learning a parametric mixture model is Expectation Maximization (EM) [25]. Recently, methods based on tensor decomposition and particularly the Canonical Polyadic Decomposition (CPD) have gained popularity as an alternative to EM for learning various latent variable models [6].

In this chapter we propose a two-stage approach based on tensor decomposition for recovering the conditional densities of mixtures of smooth product distributions. We show that when the unknown conditional densities are approximately band-limited it is possible to uniquely identify and recover them from partially observed data. The key idea is to jointly factorize histogram estimates of lower-dimensional PDFs that can be easily and reliably estimated from observed samples. The conditional densities can then be recovered using an interpolation procedure. We formulate the problem as a coupled tensor factorization and propose an alternating-optimization algorithm. We demonstrate the effectiveness of the approach on both synthetic and real data.

4.2 Background

4.2.1 Learning Problem

Let $\mathcal{X} = \{X_n\}_{n=1}^N$ denote a set of N random variables. We say that a PDF $f_\mathcal{X}$ is a mixture of R component distributions if it can be expressed as a weighted sum of R multivariate distributions

$$f_\mathcal{X}(x_1, \ldots, x_N) = \sum_{r=1}^{R} w_r f_{\mathcal{X}|H}(x_1, \ldots, x_N | r), \tag{4.1}$$

where $f_{\mathcal{X}|H}$ are conditional PDFs and $\{w_r\}_{r=1}^R$ are non-negative numbers such that $\sum_{r=1}^R w_r = 1$, called mixing weights. When each conditional PDF factors into the product of its marginal densities we have that

$$f_\mathcal{X}(x_1, \ldots, x_N) = \sum_{r=1}^{R} w_r \prod_{n=1}^{N} f_{X_n|H}(x_n | r), \tag{4.2}$$

which can be seen as a continuous extension of the CPD model. A sample from the mixture model is generated by first drawing a component r according to w and then independently drawing samples for every variable $\{X_n\}_{n=1}^N$ from the conditional PDFs $f_{X_n|H}(\cdot|r)$. The problem of learning the mixture is that of finding the conditional PDFs as well as the mixing weights given observed samples.

4.2.2 Related Work

Mixture models have numerous applications in statistics and machine learning including clustering and density estimation to name a few [82]. A common assumption made in multivariate mixture models is a parametric form of the conditional PDFs. For example, when the conditional PDFs are assumed to be Gaussian, the goal is to recover the mean vectors and covariance matrices defining each multivariate Gaussian component and the mixing weights. Other common choices include categorical, exponential, Laplace or Poisson distributions. The most popular algorithm for learning the parameters of the mixture is the EM algorithm [25] which maximizes the likelihood function with respect to the parameters. EM-based methods have been also considered for learning mixture models of non-parametric distributions[1] by parameterizing the unknown conditional PDFs using kernel density estimators [10, 75], which lack however theoretical guarantees.

Tensor decomposition methods can be used as an alternative to EM for learning various latent variable models [6]. High-order moments of several probabilistic models can be expressed using low-rank CPDs. Decomposing these tensors reveals the true parameters of the probabilistic models. In the absence of noise and model mismatch, algebraic algorithms can be applied to compute the CPD under certain conditions, see [98] and references therein, and [44] for the application to GMMs. Tensor decomposition approaches have been proposed for learning mixture models but are mostly restricted to Gaussian or categorical distributions [38, 44, 50]. In practice, mainly due to sampling noise the result of these algorithms may not be satisfactory and EM can be used for refinement [96, 121]. In the case of non-parametric mixtures of product distributions, identifiability of the components has been established in [2]. The authors have shown that it is possible to identify the conditional PDFs given the true joint PDF, if the conditional PDFs of each X_n across different mixture components are linearly independent i.e., the continuous factor "matrices" have linearly independent columns. However, the exact true joint PDF is never given – only samples drawn from it are available in practice, and elements may be missing from any given sample. Furthermore, [2] did not provide an estimation procedure, which limits the practical appeal of an interesting theoretical contribution.

We focus on mixtures of product distributions of continuous variables and do not specify a parametric form of the conditional density functions. We show that it is possible to recover

[1]The term non-parametric is used to describe the case in which no assumptions are made about the form of the conditional densities.

mixtures of *smooth* product distributions from observed samples. The key idea is to first transform the problem to that of learning a mixture of categorical distributions by decomposing lower-dimensional and (possibly coarsely) discretized joint PDFs. Given that the conditional PDFs are (approximately) band-limited (smooth), they can be recovered from the discretized PDFs under certain conditions.

4.3 Approach

Our approach consists of two stages. We express the problem as a tensor factorization problem and show that if $N \geq 3$, we can recover points of the unknown conditional CDFs. Under a smoothness condition, these points can be used to recover the true conditional PDFs using an interpolation procedure.

4.3.1 Problem Formulation

We assume that we are given M N-dimensional samples $\{\mathbf{x}_m\}_{m=1}^{M}$ that have been generated from a mixture of product distributions as in Equation (4.2). We discretize each random variable X_n by partitioning its support into I uniform intervals $\{\Delta_n^i = \left(d_n^{i-1}, d_n^i\right)\}_{1 \leq i \leq I}$. Specifically, we consider a discretization of the PDF and define the probability tensor (histogram) $\mathcal{X}[i_1, \ldots, i_N] \triangleq \mathsf{Pr}\left(X_1 \in \Delta_n^{i_1}, \ldots, X_N \in \Delta_n^{i_N}\right)$ given by

$$
\begin{aligned}
\mathcal{X}[i_1, \ldots, i_N] &= \sum_{r=1}^{R} w_r \prod_{n=1}^{N} \int_{\Delta_n^{i_n}} f_{X_n|H}(x_n|r) dx_n \\
&= \sum_{r=1}^{R} w_r \prod_{n=1}^{N} \mathsf{Pr}\left(X_n \in \Delta_n^{i_n} \middle| H = r\right).
\end{aligned}
\tag{4.3}
$$

Let $\mathbf{A}_n[i_n, r] \triangleq \mathsf{Pr}\left(X_n \in \Delta_n^{i_n} \middle| H = r\right)$, $\boldsymbol{\lambda}[r] \triangleq w_r$. Note that $\mathcal{X}$ is an N-way tensor and admits a CPD with non-negative factor matrices $\{\mathbf{A}_n\}_{n=1}^{N}$ and rank R, i.e., $\mathcal{X} = [\![\boldsymbol{\lambda}, \mathbf{A}_1, \ldots, \mathbf{A}_N]\!]$. From equation (4.3) it is clear that the discretized conditional PDFs are identifiable and can be recovered by decomposing the true joint discretized probability tensor, if $N \geq 3$ and R is small enough, by virtue of the uniqueness properties of CPD [98].

In practice we do not observe $\mathcal{X}$ but we have to deal with perturbed versions. Based on the observed samples, we can compute an approximation of the probability tensor $\mathcal{X}$ by counting how many samples fall into each bin and normalizing the tensor by dividing with the total number of samples.

The size of the probability tensor grows exponentially with the number of variables and therefore the estimate will be highly inaccurate even when the number of discretization intervals is small. More importantly, datasets often contain missing data and therefore its impossible to construct such tensor. On the other hand, it may be possible to estimate low-dimensional discretized joint PDFs of subsets of the random variables which correspond to low-order tensors. For example, in the clustering example given in the introduction some patients may be tested on a subset of the available tests. Finally, the model of Equation (4.3) is just an approximation of our original model, as our ultimate goal is to recover the true conditional PDFs. To address the aforementioned challenges we have to answer the following two questions

1. Is it possible to learn the mixing weights and discretized conditional PDFs from missing/limited data?

2. Is it possible to recover non-parametric conditional PDFs from their discretized counterparts?

Regarding the first question, we have shown in Chapter 3 that a joint Probability Mass Function (PMF) of a set of random variables can be recovered from lower-dimensional joint PMFs if the joint PMF has low enough rank. This result allows us to recover the discretized conditional PDFs from low-dimensional histograms but cannot be extended to the continuous setting in general because of the loss of information induced from the discretization step. Next, we discuss and provide conditions under which we can overcome these issues.

4.3.2 Identifiability using Lower-dimensional Statistics

In this section we provide insights regarding the first issue. It turns out that realizations of subsets of only three random variables are sufficient to recover $\mathsf{Pr}\left(X_n \in \Delta_n^{i_n} \middle| H = r\right)$ and $\{w_r\}_{r=1}^R$. Under the mixture model (4.2), a histogram of any subset of three random variables X_j, X_k, X_ℓ denoted as $\mathcal{X}_{jk\ell}$, with $\mathcal{X}_{jk\ell}[i_j, i_k, i_\ell] = \mathsf{Pr}(X_j \in \Delta_j^{i_j}, X_k \in \Delta_k^{i_k}, X_\ell \in \Delta_\ell^{i_\ell})$ can be written as $\mathcal{X}_{jk\ell}[i_j, i_k, i_\ell] = \sum_{r=1}^R \boldsymbol{\lambda}[r]\mathbf{A}_j[i_j, r]\mathbf{A}_k[i_k, r]\mathbf{A}_\ell[i_\ell, r]$, which is a third-order tensor of rank R. A fundamental result on the uniqueness of tensor decomposition of third-order tensors was given by Kruskal and is discussed in Chapter 2 (Theorem 2.1). The result states that if $\mathcal{X}$ admits a decomposition $\mathcal{X} = [\![\boldsymbol{\lambda}, \mathbf{A}_1, \mathbf{A}_2, \mathbf{A}_3]\!]$, with $k_{\mathbf{A}_1} + k_{\mathbf{A}_2} + k_{\mathbf{A}_3} \geq 2R + 2$ then rank$(\mathcal{X}) = R$ and the decomposition of $\mathcal{X}$ is unique.

When the rank is small and the decomposition is exact, the parameters of the CPD model can be computed exactly via Generalized Eigenvalue Decomposition (GEVD) and related algebraic algorithms [26, 74, 98].

Theorem 4.1. *[74] Let $\mathcal{X}$ be a tensor that admits a decomposition $\mathcal{X} = [\![\boldsymbol{\lambda}, \mathbf{A}_1, \mathbf{A}_2, \mathbf{A}_3]\!]$, $\mathbf{A}_1 \in \mathbb{R}^{I_1 \times R}$, $\mathbf{A}_2 \in \mathbb{R}^{I_2 \times R}$, $\mathbf{A}_3 \in \mathbb{R}^{I_3 \times R}$, $\boldsymbol{\lambda} \in \mathbb{R}^R$ and suppose that $\mathbf{A}_1$, $\mathbf{A}_2$ are full column rank and $k_{\mathbf{A}_3} \geq 2$. Then rank$(\mathcal{X}) = R$, the decomposition of $\mathcal{X}$ is unique and can be found algebraically.*

More relaxed uniqueness conditions from the field of algebraic geometry have been proven in recent years (see e.g., Theorem 2.3). Theorem 2.3 is a generic uniqueness result i.e, all non-identifiable parameters form a set of Lebesgue measure zero. To see how theorems 2.1, 2.3 can be applied in our setup, consider the joint decomposition of the probability tensors $\mathcal{X}_{jk\ell}$. Let $\mathcal{S}_1$, $\mathcal{S}_2$, and $\mathcal{S}_3$ denote disjoint ordered subsets of $[N]$, with cardinality $c_1 = |\mathcal{S}_1|$, $c_2 = |\mathcal{S}_2|$, and $c_3 = |\mathcal{S}_3|$, respectively. Let $\mathcal{Y}$ be the $c_1 \times c_2 \times c_3$ block tensor whose (j, k, ℓ)-th block is the tensor $\mathcal{X}_{jk\ell}$, $j \in \mathcal{S}_1$, $k \in \mathcal{S}_2$, $\ell \in \mathcal{S}_3$. It is clear that the tensor $\mathcal{Y}$ admits a CPD $\mathcal{Y} = [\![\boldsymbol{\lambda}, \widehat{\mathbf{A}}_1, \widehat{\mathbf{A}}_2, \widehat{\mathbf{A}}_3]\!]$ where $\widehat{\mathbf{A}}_1 = [\mathbf{A}^T_{\mathcal{S}_1(1)}, \cdots, \mathbf{A}^T_{\mathcal{S}_1(c_1)}]^T$, $\widehat{\mathbf{A}}_2 = [\mathbf{A}^T_{\mathcal{S}_2(1)}, \cdots, \mathbf{A}^T_{\mathcal{S}_2(c_2)}]^T$, $\widehat{\mathbf{A}}_3 = [\mathbf{A}^T_{\mathcal{S}_3(1)}, \cdots, \mathbf{A}^T_{\mathcal{S}_3(c_3)}]^T$. By considering the joint decomposition of lower-dimensional discretized PDFs, we have constructed a single virtual non-negative CPD model and therefore uniqueness properties hold. For example, by setting $\mathcal{S}_1 = \{1, \ldots, \lfloor \frac{N-1}{2} \rfloor - 1\}$, $\mathcal{S}_2 = \{\lfloor \frac{N-1}{2} \rfloor, \ldots, N-1\}$, $\mathcal{S}_3 = \{N\}$ we have that

$$\mathbf{Y}^{(1)} = \left(\begin{bmatrix} \mathbf{A}_{\lfloor \frac{N-1}{2} \rfloor} \\ \vdots \\ \mathbf{A}_{N-1} \end{bmatrix} \odot \begin{bmatrix} \mathbf{A}_1 \\ \vdots \\ \mathbf{A}_{\lfloor \frac{N-1}{2} \rfloor - 1} \end{bmatrix} \right) \mathrm{diag}(\boldsymbol{\lambda}) \mathbf{A}_N^T.$$

According to Theorem 4.1, the CPD can be computed exactly if $R \leq (\lfloor \frac{N-1}{2} \rfloor - 1)I$. Similarly, it is easy to verify that by setting $c_1 = c_2 = \lfloor \frac{N}{3} \rfloor I$, i.e., $\alpha = \lfloor \log_2(\lfloor \frac{N}{3} \rfloor I) \rfloor$, the CPD of $\mathcal{Y}$ is generically unique for $R \leq 2^{2(\alpha-1)}$ according to Theorem 2.3. The later inequality is implied by $R \leq \frac{(\lfloor \frac{N}{3} \rfloor I + 1)^2}{16}$ which shows that the bound is quadratic in N and I.

Remark 1: The previous discussion suggests that finer discretization can lead to improved identifiability results. The number of hidden components may be arbitrarily large and we may still be able to identify the discretized conditional PDFs by increasing the dimensions of the sub-tensors i.e., the discretization intervals of the random variables. The caveat is that one will need many more samples to reliably estimate these histograms. Ideally, one would like to have the minimum number of intervals that can guarantee identifiability of the conditional PDFs.

Remark 2: The factor matrices can be recovered by decomposing the lower-order probability tensors of dimension $N \geq 3$. It is important to note that histograms of subsets of two variables correspond to Non-negative Matrix Factorization (NMF) which is not identifiable unless additional conditions such as sparsity are assumed for the latent factors [32]. Therefore, second-order distributions are not sufficient for recovering dense latent factor matrices.

4.3.3 Recovery of the Conditional PDFs

In the previous section we have shown that given lower-dimensional discretized PDFs, we can uniquely identify and recover discretized versions of the conditional PDFs via joint tensor decomposition. Recovering the true conditional PDFs from the discretized counterparts can be viewed as a signal reconstruction problem. We know that this is not possible unless the signals have some smoothness properties. We will use the following result.

Proposition 4.1. *A PDF that is (approximately) band-limited with cutoff frequency ω_c can be recovered from uniform samples of the associated CDF taken $\frac{\pi}{\omega_c}$ apart.*

Proof: Assume that the PDF f_X is band-limited with cutoff frequency ω_c i.e., its Fourier transform $\mathcal{F}(\omega) = 0, \forall\ |\omega| \geq \omega_c$. Let F_X denote the CDF of f_X, $F_X(x) = \int_{-\infty}^{x} f_X(t)dt$. We can express the integration as a convolution of the PDF with a unit step function, i.e., $F_X(x) = \int_{-\infty}^{\infty} f_X(t)u(x-t)d\tau$. The Fourier transform of a convolution is the point-wise product of Fourier transforms. Therefore, we can express the Fourier transform $\mathcal{G}(\omega)$ of the CDF as

$$\mathcal{G}(\omega) = \pi\delta(\omega)\mathcal{F}(0) + \frac{\mathcal{F}(\omega)}{j\omega}, \tag{4.4}$$

where $\delta(\cdot)$ is the Dirac delta. From Equation (4.4), it is clear that the CDF obeys the same band-limit as the PDF . From Shannon's sampling theorem we have that

$$F_X(x) = \sum_{n=-\infty}^{\infty} F_X(nT)\,\mathrm{sinc}\left(\frac{x-nT}{T}\right), \tag{4.5}$$

where $T = \frac{\pi}{\omega_c}$. The PDF can then be determined by differentiation, which amounts to linear interpolation of the CDF samples using the derivative of the sinc kernel. Note that for exact reconstruction of f_X an infinite number of data points are needed. In signal processing practice we always deal with finite support signals which are only approximately band-limited; the point is that the bandlimited assumption is accurate enough to afford high-quality signal reconstruction. In our present context, a good example is the Gaussian distribution: even though it is of infinite extent, it is not strictly bandlimited (as its Fourier transform is another Gaussian); but it is approximately bandlimited, and that is good enough for our purposes, as we will see shortly.

In section 4.3.2, we saw how lower-dimensional histograms can be used to obtain estimates of the discretized conditional PDFs. Now, consider the conditional PDF of the n-th variable given the r-th component. The corresponding column of factor matrix $\mathbf{A}_n$ is

$$\mathbf{A}_n[:, r] = [F_{X_n|H}(d_n^1|r) - F_{X_n|H}(d_n^0|r), \ldots, 1 - F_{X_n|H}(d_n^{I-1}|r)]^T.$$

Since, $F_{X_n|H}(d_n^0|r) = 0$, we can compute $F_{X_n|H}(d_n^i|r), \forall i \in [I-1], n \in [N]$. We also know that $F_{X_n|H}(x_n|r) = 1, \forall x_n \geq d_n^I$. Therefore, we can recover the conditional CDFs using the interpolation formula

$$F_{X|H}(x_n|r) = \sum_{k=-L}^{L} F_{X_n|H}(kT|r) \operatorname{sinc}\left(\frac{x - kT}{T}\right), \tag{4.6}$$

where $T = d_n^i - d_n^{i-1}$ and L a large integer. The conditional PDF $f_{X_n|H}$ can then be recovered via differentiation.

4.3.4 Toy example

An example to illustrate the idea is shown in Figure 4.1. Assume that the PDF of a random variable is a mixture of two Gaussian distributions with means $\mu_1 = -6$, $\mu_2 = 10$ and standard deviations $\sigma_1 = \sigma_2 = 5$. It is clear from Figure 4.1 that $\mathcal{F}(\omega) \approx 0$ for $|\omega| \geq \omega_c = 0.8$ and therefore the PDF is approximately band-limited. The CDF has the same band-limit, thus, it can be recovered from points being $T = \frac{\pi}{\omega_c} \approx 4$ apart. In this example we have used only 10 discretization intervals as they suffice to capture 99% of the data. We use the finite sum formula of Equation (4.6) to recover the CDF and then we recover the PDF by differentiating the CDF. The recovered PDF essentially coincides with the true PDF given a few exact estimates of the CDF as shown in Figure 4.1.

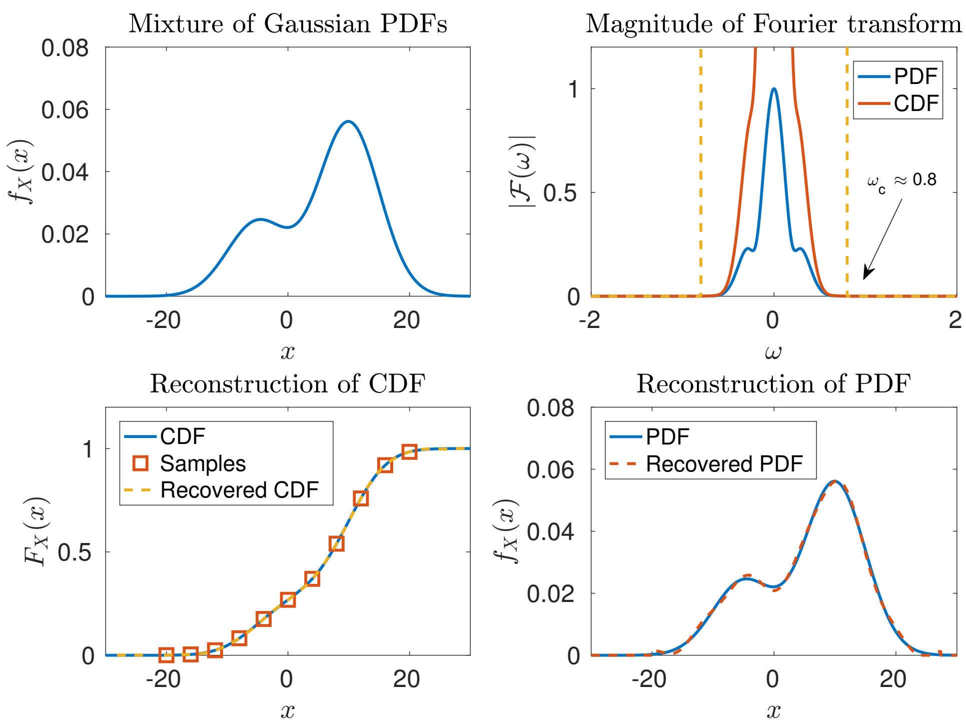


Figure 4.1: Illustration of the key idea on a univariate Gaussian mixture. The CDF can be recovered from its samples if $T_s \leq \frac{\pi}{0.8}$.

Figure 4.2 shows the approximation error for different methods when we do not have exact points of the CDF but estimate them from randomly drawn samples. We know that a histogram converges to the true PDF as the number of samples grows and the bin width goes to 0 at appropriate rate. However, when the conditional PDF is smooth, the interpolation procedure using a few discretization intervals leads to a lower approximation error compared to plain histogram estimates as illustrated in the figure.

4.4 Algorithm

In this section we develop an algorithm for recovering the latent factors of the CPD model given the histogram estimates of lower-dimensional PDFs (Alg. 4.1). We define the following

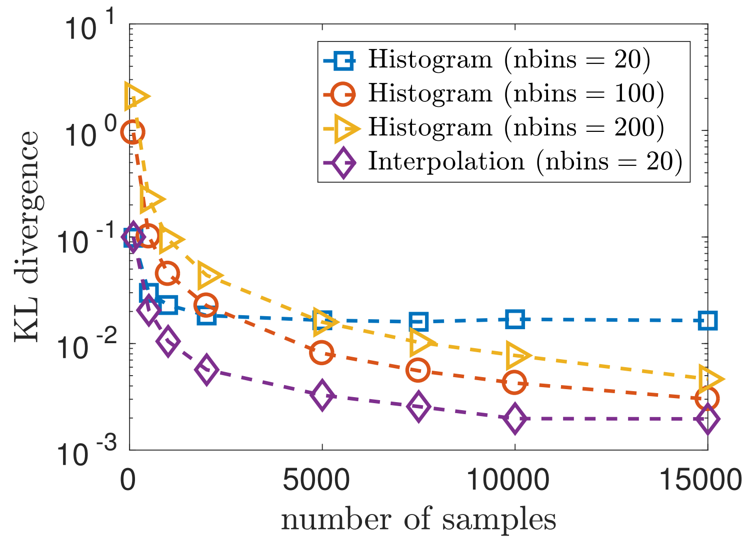


Figure 4.2: KL divergence between the true mixture of Gaussians and different approximations.

optimization problem

$$\begin{aligned}
\min_{\{\mathbf{A}_n\}_{n=1}^N, \boldsymbol{\lambda}} \quad & \sum_{j=1}^{N} \sum_{k>j}^{N} \sum_{\ell>k}^{N} \mathrm{D}\left(\widehat{\mathcal{X}}_{jk\ell}, [\![\boldsymbol{\lambda}, \mathbf{A}_j, \mathbf{A}_k, \mathbf{A}_\ell]\!]\right) \\
\text{s.t.} \quad & \boldsymbol{\lambda} \geq \mathbf{0}, \mathbf{1}^T \boldsymbol{\lambda} = 1 \\
& \mathbf{A}_n \geq \mathbf{0}, \ n = 1 \ldots N \\
& \mathbf{1}^T \mathbf{A}_n = \mathbf{1}^T, \ n = 1 \ldots N
\end{aligned} \tag{4.7}$$

where $\mathrm{D}(\cdot, \cdot)$ is a suitable metric. The Frobenious norm and Kullback-Leibler (KL) divergence are considered in this work. For probability tensors $\mathcal{X}, \mathcal{Y}$ we define

$$\mathrm{D}_{\mathrm{KL}}(\mathcal{X}, \mathcal{Y}) \triangleq \sum_{i_1, i_2, i_3} \mathcal{X}[i_1, i_2, i_3] \log \frac{\mathcal{X}[i_1, i_2, i_3]}{\mathcal{Y}[i_1, i_2, i_3]}$$

$$\mathrm{D}_{\mathrm{FRO}}(\mathcal{X}, \mathcal{Y}) \triangleq \sum_{i_1, i_2, i_3} \left(\mathcal{X}[i_1, i_2, i_3] - \mathcal{Y}[i_1, i_2, i_3]\right)^2.$$

Optimization problem (4.7) is non-convex and NP-hard in its general form. Nevertheless, sensible approximation algorithms can be derived, based on well-appreciated nonconvex optimization

Algorithm 4.1 Proposed Algorithm

Input: A dataset $\mathbf{D} \in \mathbb{R}^{M \times N}$
Estimate $\mathcal{X}_{jk\ell}\ \forall j, k, \ell \in [N],\ \ell > k > j$ from data.
Initialize $\{\mathbf{A}_n\}_{n=1}^{N}$ and $\boldsymbol{\lambda}$.
repeat
  for all $n \in [N]$ **do**
    Solve opt. problem (4.8) via mirror descent.
  end for
  Solve opt. problem (4.10) via mirror descent.
until convergence criterion is satisfied
for all $n \in [N]$ **do**
  Recover $f_{X_n|H}$ by differentiation using Eq. (4.6)
end for

tools. The idea is to cyclically update the variables while keeping all but one fixed. By fixing all other variables and optimizing with respect to $\mathbf{A}_j$ we have

$$\min_{\mathbf{A}_j \in \mathcal{C}} \sum_{k \neq j} \sum_{\substack{\ell \neq j \\ \ell > k}} \mathrm{D}\left(\mathbf{X}_{jk\ell}^{(1)}, (\mathbf{A}_\ell \odot \mathbf{A}_k)\mathrm{diag}(\boldsymbol{\lambda})\mathbf{A}_j^T\right), \tag{4.8}$$

where $\mathcal{C} = \{\mathbf{A} \mid \mathbf{A} \geq 0, \mathbf{1}^T\mathbf{A} = \mathbf{1}^T\}$. Problem (4.8) is convex and can be solved efficiently using Exponentiated Gradient (EG) [65] – which is a special case of mirror descent [9]. At each iteration τ of mirror descent we update $\mathbf{A}_j^\tau$ by solving

$$\mathbf{A}_j^\tau = \arg\min_{\mathbf{A}_j \in \mathcal{C}} \langle \nabla f\big(\mathbf{A}_j^{\tau-1}\big), \mathbf{A}_j \rangle + \frac{1}{\eta_\tau} B_\Phi\big(\mathbf{A}_j, \mathbf{A}_j^{\tau-1}\big)$$

where $B_\Phi(\mathbf{A}, \widehat{\mathbf{A}}) = \Phi(\mathbf{A}) - \Phi(\widehat{\mathbf{A}}) - \langle \mathbf{A} - \widehat{\mathbf{A}}, \nabla\Phi(\widehat{\mathbf{A}})\rangle$ is a Bregman divergence. Setting Φ to be the negative entropy $\Phi(\mathbf{A}) = \sum_{i,j} \mathbf{A}(i,j) \log \mathbf{A}(i,j)$, the update becomes

$$\mathbf{A}_j^\tau = \mathbf{A}_j^{\tau-1} \circledast \exp\left(-\eta_\tau \nabla f\left(\mathbf{A}_j^{\tau-1}\right)\right), \tag{4.9}$$

where $\circledast$ is the Hadamard product, followed by column normalization $\mathbf{A}_j^\tau[:, r] = \frac{\mathbf{A}_j[:,r]}{1^T\mathbf{A}[:,r]}$.

The optimization problem with respect to $\boldsymbol{\lambda}$ is the following

$$\min_{\boldsymbol{\lambda} \in \mathcal{C}} \sum_{j,k,\ell} \mathrm{D}\left(\mathrm{vec}(\mathcal{X}_{jk\ell}), (\mathbf{A}_\ell \odot \mathbf{A}_k \odot \mathbf{A}_j)\boldsymbol{\lambda}\right). \tag{4.10}$$

The update rules for $\boldsymbol{\lambda}$ are similar

$$\boldsymbol{\lambda}^{\tau} = \boldsymbol{\lambda}^{\tau-1} \circledast \exp\left(-\eta_{\tau} \nabla f\left(\boldsymbol{\lambda}^{\tau-1}\right)\right). \tag{4.11}$$

The step η_{τ} can be computed by the Armijo rule [11].

4.5 Experiments

4.5.1 Synthetic Data

In this section, we employ synthetic data simulations to showcase the effectiveness of the proposed algorithm. Experiments are conducted on synthetic datasets $\{\mathbf{x}_m\}_{m=1}^{M}$ of varying sample sizes, generated from R component distributions. We set the number of variables to $N = 10$, and vary the number of components $R \in \{5, 10\}$. We run the algorithms using 5 different random initializations and for each algorithm keep the model that yields the lowest cost. We evaluate the performance of the algorithms by calculating the KL divergence between the true and learned model, which is approximated using Monte Carlo integration. Specifically, we generate $\{\mathbf{x}_{m'}\}_{m'=1}^{M'}$ test points, $M' = 1000$ drawn from the mixture and approximate the KL divergence between the true and learned model by

$$\mathrm{D}_{\mathrm{KL}}\left(f_{\mathcal{X}}, \widehat{f}_{\mathcal{X}}\right) \approx \frac{1}{M'} \sum_{m'=1}^{M'} \log f_{\mathcal{X}}(\mathbf{x}_{m'}) / \widehat{f}_{\mathcal{X}}(\mathbf{x}_{m'}).$$

We also compute the clustering accuracy on the test points as follows. Each data point $\mathbf{x}_{m'}$ is first assigned to the component yielding the highest posterior probability $\widehat{c}_m = \arg\max_c f_{H|\mathcal{X}}(c|\mathbf{x}_m)$. Due to the label permutation ambiguity, the obtained components are aligned with the true components using the Hungarian algorithm [71]. The clustering accuracy is then calculated as the ratio of wrongly labeled data points over the total number of data points.For each scenario, we repeat 10 Monte Carlo simulations and report the average results. We explore the following settings for the conditional PDFs: (1) Gaussian (2) GMM with two components (3) Gamma and (4) Laplace. The mixing weights are drawn from a Dirichlet distribution $\omega \sim \mathrm{Dir}(\alpha_1, \ldots, \alpha_r)$ with $\alpha_r = 10\ \forall r$. We emphasize that our approach does not use any knowledge of the parametric form of the conditional PDFs; it only assumes smoothness.

Gaussian Conditional Densities: In the first experiment we assume that each conditional

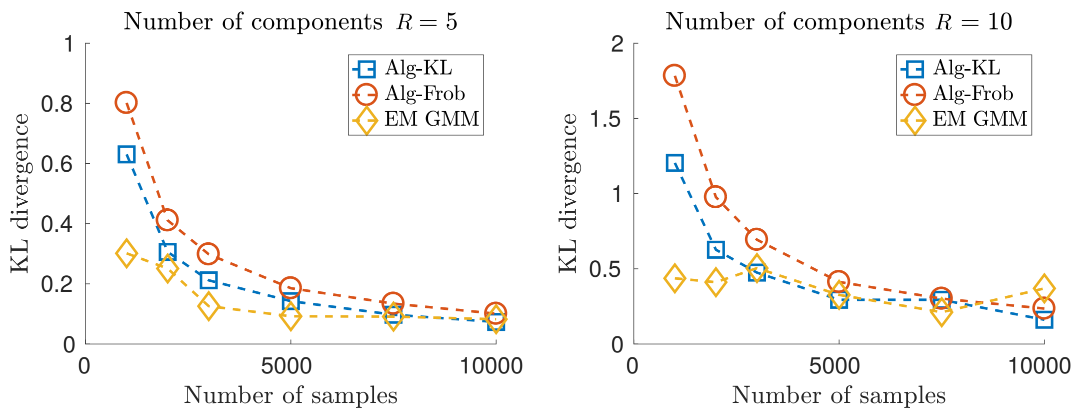


Figure 4.3: KL divergence (Gaussian).

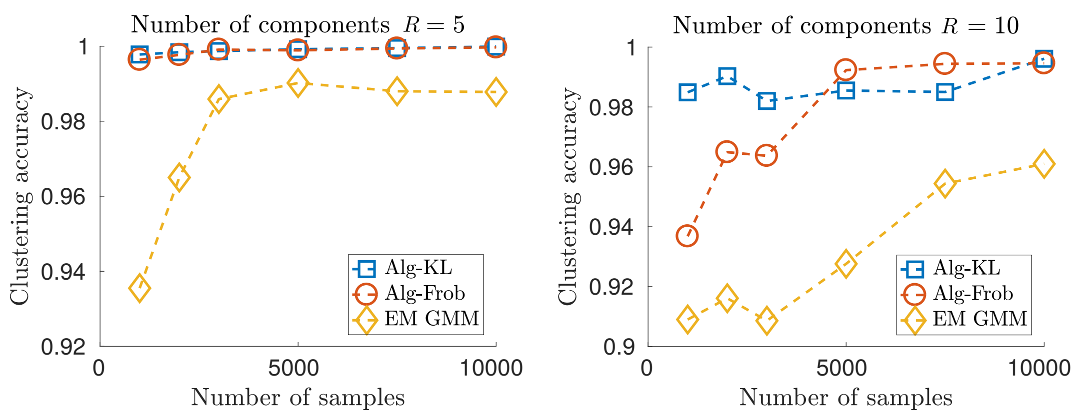


Figure 4.4: Clustering accuracy (Gaussian).

PDF is a Gaussian. For cluster r and random variable X_n we set $f_{X_n|H}(x_n|r) = \mathcal{N}(\mu_{nr}, \sigma_{nr}^2)$. Mean and variance are drawn from uniform distributions, $\mu_{nr} \sim \mathcal{U}(-5, 5)$, $\sigma_{nr}^2 \sim \mathcal{U}(1, 2)$. We compare the performance of our algorithms to that of EM (EM GMM). Figure 4.3 shows the KL divergence between the true and the learned model for various dataset sizes and different number of components. We see that the performance of our methods converges to that of EM despite the fact that we do not assume a particular model for the conditional densities. Interestingly, our approach performs better in terms of clustering accuracy as shown in Figure 4.4. We can see that although the joint distribution learned by EM is closer to the true in terms of the KL divergence, EM may fail to identify the true parameters of every component.

GMM Conditional Densities: In the second experiment we assume that each conditional

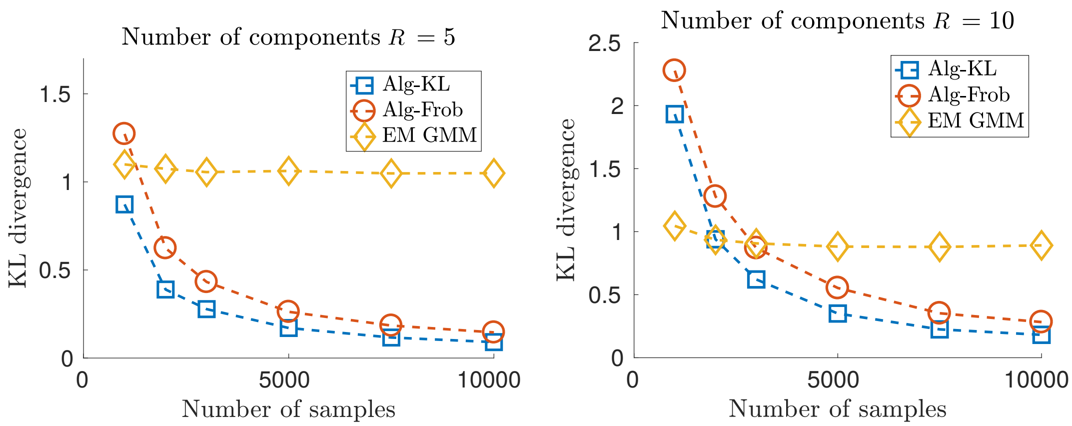


Figure 4.5: KL divergence (GMM).

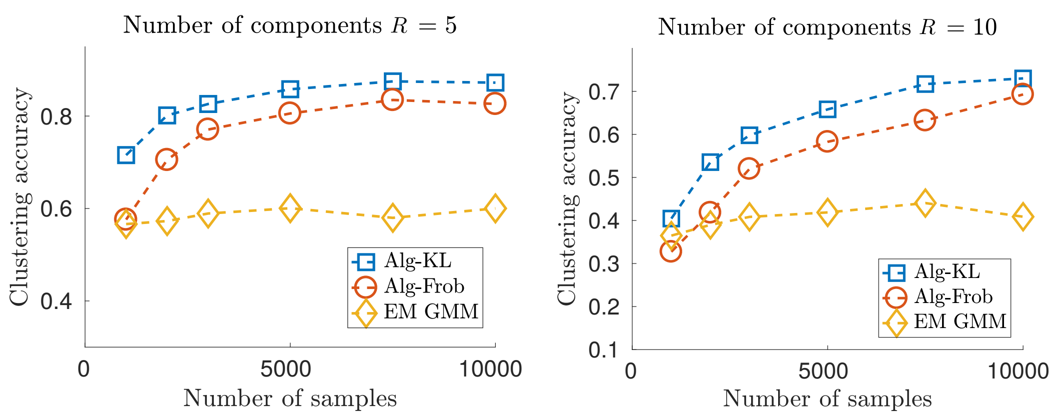


Figure 4.6: Clustering Accuracy (GMM).

PDF is itself a mixture model of two univariate Gaussian distributions. More specifically, we set $f_{X_n|H}(x_n|r) = \frac{1}{2}\mathcal{N}\left(\mu_{nr}^{(1)}, \sigma_{nr}^{(1)2}\right) + \frac{1}{2}\mathcal{N}\left(\mu_{nr}^{(2)}, \sigma_{nr}^{(2)2}\right)$. Means and variances are drawn from uniform distributions $\mu_{nr}^{(1)} \sim \mathcal{U}(0,7)$, $\sigma_{nr}^{(1)2} \sim \mathcal{U}(1,4)$, $\mu_{nr}^{(2)} \sim \mathcal{U}(-7,0)$, $\sigma_{nr}^{(2)2} \sim \mathcal{U}(1,4)$. Our method is able to learn the mixture model in contrast to the EM GMM which exhibits poor performance, due to the model mismatch, as shown in Figures 4.5, 4.6.

Gamma Conditional Densities: Another example of a smooth distribution is the shifted Gamma distribution. We set $f_{X_n|H}(x_n|r) = \frac{1}{\beta^\alpha \Gamma(\alpha)}(x - \mu_{nr})^{\alpha-1}\exp(-\frac{x-\mu_{nr}}{\beta})$ with $\alpha = 5$, $\mu_{nr} \sim \mathcal{U}(-5,0)$, $\beta_{nr} \sim \mathcal{U}(0.1, 0.5)$. As the number of samples grows our method exhibits better performance, significantly outperforming EM GMM as shown in Figures 4.7, 4.8.

Laplace Conditional Densities: In the last simulated experiment we assume that each

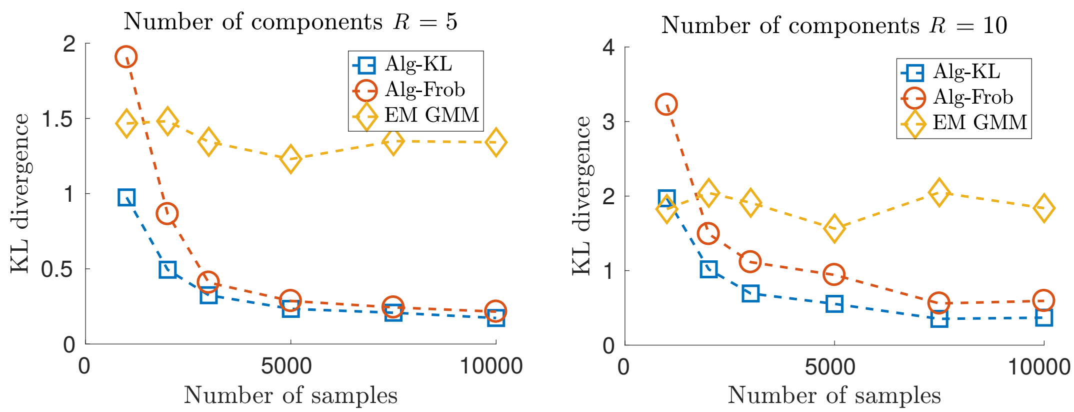


Figure 4.7: KL divergence (Gamma).

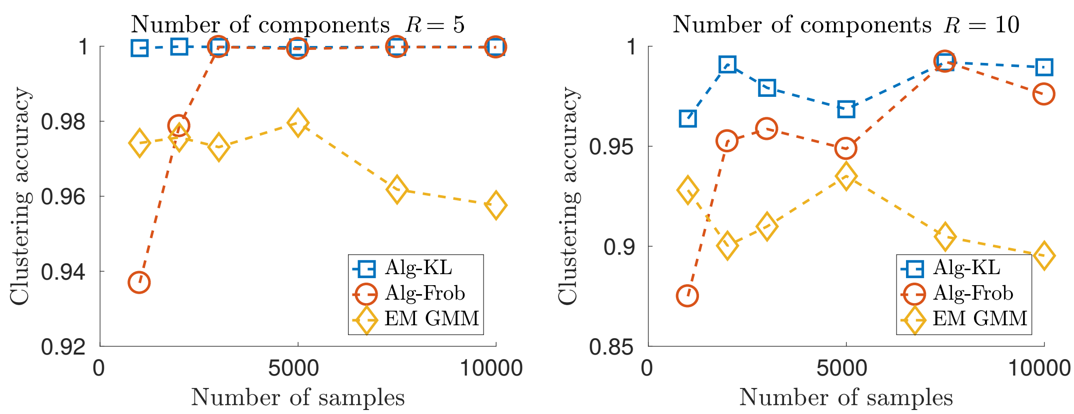


Figure 4.8: Clustering accuracy (Gamma).

conditional PDF is a Laplace distribution with mean μ_{nr} and standard deviation σ_{nr} i.e., $f_{X_n|H}(x_n|r) = \frac{1}{\sqrt{2}\sigma_{nr}} \exp\left(\frac{\sqrt{2}|x_n-\mu_{nr}|}{\sigma_{nr}}\right)$. A Laplace distribution in contrast to the previous cases is not smooth (at its mean). Parameters are drawn from uniform distributions, $\mu_{nf} \sim \mathcal{U}(-5,5)$, $\sigma_{nf}^2 \sim \mathcal{U}(5,10)$. We compare the performance of our methods to that of the EM GMM and an EM algorithm for a Laplace mixture model (EM LMM). The proposed method approaches the performance of EM LMM and exhibits better performance in terms of KL and clustering accuracy compared to the EM GMM for higher number of data samples, as shown in Figures 4.9, 4.10.

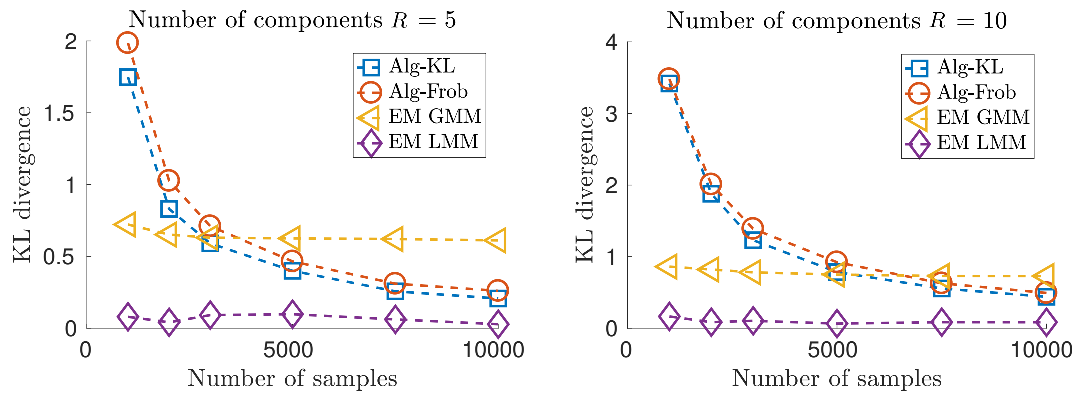


Figure 4.9: KL divergence (Laplace).

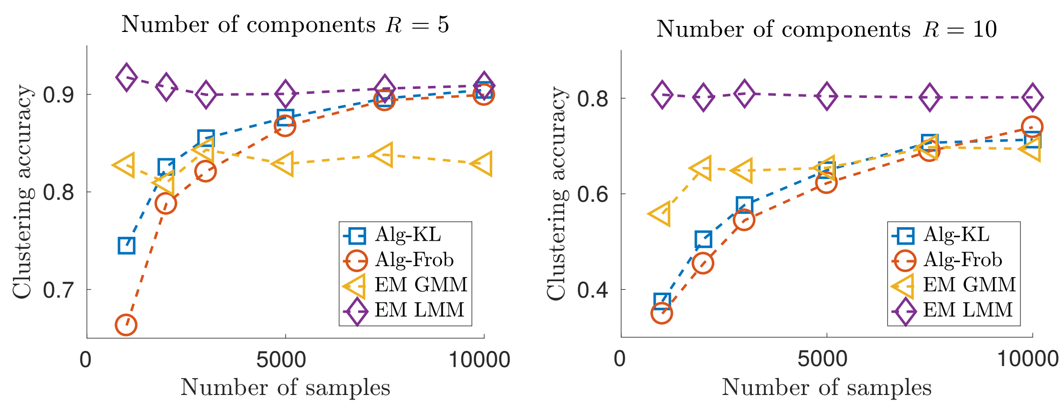


Figure 4.10: Clustering accuracy (Laplace).

4.5.2 Real Data

Finally, we conduct several real-data experiments to test the ability of the algorithms to cluster data. We selected 7 datasets with continuous variables suitable for classification or regression tasks from the UCI repository. For each labeled dataset we hide the label and treat it as the latent component. For datasets that contained a continuous variable as a response, we discretized the response into R uniform intervals and treated it as the latent component. For each dataset we repeated 10 Monte Carlo simulations by randomly splitting the dataset into three sets; 70% was used as a training set, 10% as a validation set and 20% as a test set. The validation set was used to select the number of discretization intervals which was either 5 or 10. We compare our methods against the EM GMM with diagonal covariance, EM GMM with full-covariance and the K-means

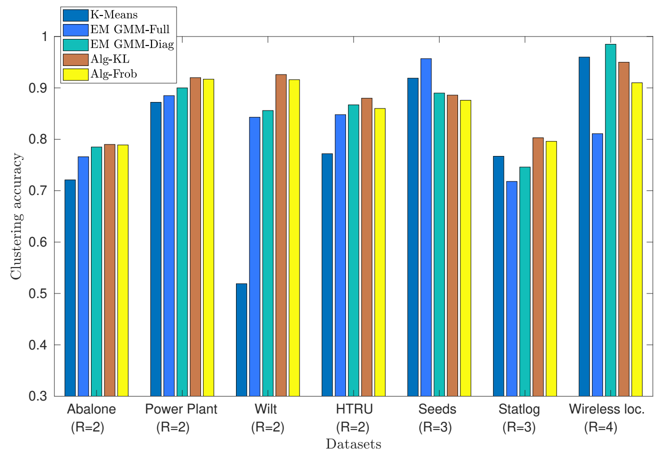


Figure 4.11: Clustering accuracy on real datasets.

algorithm in terms of clustering accuracy. Note that although the conditional independence assumption may not actually hold in practice, almost all the algorithms give satisfactory results in the tested datasets. The proposed algorithms perform well, outperforming the baselines in 5 out of 7 datasets while performing reasonably well in the remaining.

4.6 Chapter Summary

We have proposed a two-stage approach based on tensor decomposition and signal processing tools for recovering the conditional densities of mixtures of smooth product distributions. Our method does not assume a parametric form for the unknown conditional PDFs. We have formulated the problem as a coupled tensor factorization and proposed an alternating-optimization algorithm. Experiments on synthetic data have shown that when the underlying conditional PDFs are indeed smooth our method can recover them with high accuracy. Results on real data have shown satisfactory performance on data clustering tasks.

Chapter 5

Supervised Learning via Tensor Completion

5.1 Introduction

The problem of identifying a nonlinear function $y = f(x_1, \ldots, x_N)$ from input-output examples is of paramount importance in machine learning, dynamical system identification and control, communications, and many other disciplines. In machine learning in particular, most of the supervised learning tasks are nonlinear system identification problems. For example, binary/multiclass classification, where the goal is to predict a discrete variable denoting the class label of each realization, and regression/prediction, where the goal is to predict real or complex valued variables. Algorithmic advancements, availability of vast amounts of data and increasing computational power have led to the development of state-of-the-art prediction models with unprecedented success in various domains such as image classification, speech recognition, and language processing. Kernel methods, random forests, neural networks and deep learning are powerful classes of machine learning models that can learn highly nonlinear functions and have been successfully applied in many supervised machine learning tasks [41]. Each of the aforementioned methods can be well suited for a particular problem, but may perform badly for another. In general it is seldom known in advance which method will perform best for any given problem.

In this chapter we present a simple and elegant alternative for nonlinear system identification based on low-rank tensor decomposition. We show that the CPD model offers an appealing

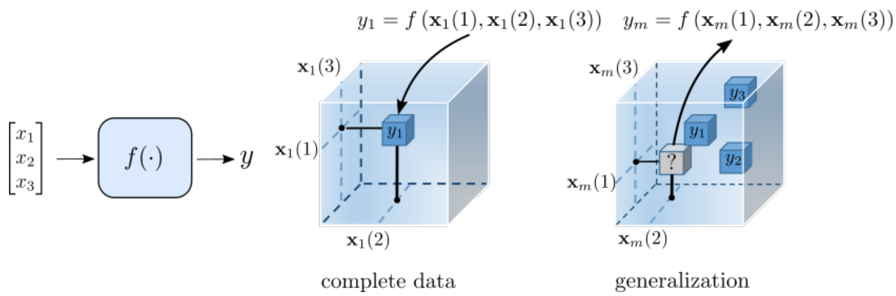


Figure 5.1: Nonlinear system identification as tensor completion.

solution for modeling and learning a *general* nonlinear system using a single high-order tensor. Tensors have been used to model *low-order multivariate polynomial* systems: a multivariate polynomial of order d is represented by a tensor of order d – e.g., a second-order polynomial is represented by a quadratic form involving a single matrix [95]. However, such an approach requires prior knowledge of polynomial order, and assuming that one deals with a polynomial of a given degree can be highly restrictive in practice. Instead, what we advocate here is a simple and general approach: a nonlinear system having N discrete inputs and a single output can be naturally represented as an N-way tensor where the tuple of input variables $[\mathbf{x}_m[1], \ldots, \mathbf{x}_m[N]]$ can be viewed as a cell multi-index and the cell content is the response of the system y_m. Given a new data point, the corresponding tensor cell is queried and the output is used as the predictor. Note that only a small fraction of the tensor entries are observed during training, and we are ultimately interested in answering queries for unobserved data points (Figure 5.1). This motivates the use of low-rank tensor models as a tool for capturing interactions between the predictors and imputing the missing data.

Both experimental [107] and theoretical studies [50, 69, 104] have shown that exact tensor completion from limited samples is possible under certain conditions. The implication of our simple but profound modeling idea is very compelling, since:

- The CPD can model any nonlinearity (even of ∞ order) for high-enough rank – because every tensor admits a CPD of bounded rank. Even for low ranks, it can model highly nonlinear operators such as products or sums of the signs of the input variables.

- Provably correct nonlinear system identification is possible from limited samples. If the associated tensor describing the nonlinear operator is low rank, then it can be fully identified.

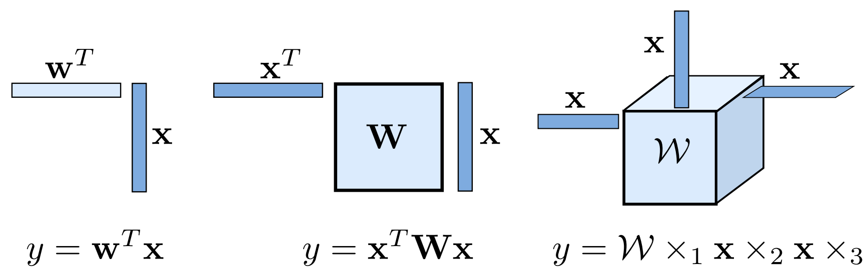


Figure 5.2: Traditional approaches.

- In practice, tensors corresponding to real-world systems may not be low-rank; nevertheless even if a system is not exactly low rank, our approach will identify the principal components of the unknown nonlinear mapping, in a sense that will be clarified in the sequel.

Even though tensor recovery can be guaranteed under a low-rank assumption, tensor decomposition can often benefit from additional knowledge regarding the application by incorporating constraints such as non-negativity, sparsity or smoothness [98]. In our present context, smoothness is a desirable property for applications where we expect that small perturbations in the input will most probably cause small changes in the output of the system. Therefore, we propose augmenting the CPD tensor completion problem with smoothness regularization on the ordinal latent factors.

Contributions: We model a general nonlinear system using a single high-order tensor admitting a CPD model. Specifically, we formulate the problem as a smooth tensor decomposition problem with missing data. Although our method is naturally suited to handle discrete features, it can also be used for continuous valued features [60] and be enhanced using ensemble techniques. Additionally, leveraging the structure of the CPD model, we propose a simple yet effective approach to handle randomly missing input variables. Finally, we discuss how the approach can be extended to vector valued function prediction. The proposed approach requires little parameter tuning, and can model complex nonlinear functions. We propose an easy to implement Block Coordinate Descent (BCD) algorithm and demonstrate the performance in UCI machine learning datasets against competitive baselines as well as a challenging grade prediction task, using real student grade data.

5.2 Related Work

Tensors have been mostly used to model low-order multivariate polynomial systems. A multivariate polynomial of degree (order) d can be represented by a tensor of order d. For example, a second-order polynomial is represented by a quadratic form involving a single matrix i.e., $f(\mathbf{x}) = \mathbf{x}^T\mathbf{W}\mathbf{x}$ while a third order polynomial is represented using a 3-way tensor i.e., $f(\mathbf{x}) = \mathcal{W} \times_1 \mathbf{x} \times_2 \mathbf{x} \times \mathbf{x}_3$ (Figure 5.2). The number of parameters grows exponentially with the order of the approximation making this approach computationally demanding. One way to reduce the number of parameters is to assume that the coefficient tensor is low-rank.

Polynomial Networks (PN) and Factorization Machines (FM) utilize mainly third-order CPD models in order to parameterize the polynomial coefficients with applications in recommender systems and link prediction [14, 15, 95]. Such approaches require prior knowledge of polynomial order, and assuming that one deals with a polynomial of a given degree can be restrictive. Additionally, even when dealing with the simplest possible approximation model which is rank-1, the number of parameters grows linearly with d meaning that this approach cannot model high-degree polynomial functions. Similarly, another tensor model, known as the Tucker model, has also been used for parametrization of polynomial functions in a chemogenomics data prediction task [92]. Finally, a tensor train model [88] has also been used in multivariate polynomial regression [87]. Unlike CPD, the model parameters of Tucker and tensor train models are not identifiable.

Output-only ('blind') identification of *linear* systems has also been considered from a tensor point of view. Specifically, identification of Finite Impulse Response (FIR) systems *using only output examples* has been shown in [17, 111].

Our work is radically different from existing approaches. We model a general nonlinear system using a single tensor of order equal to the number of inputs and propose using a high-order tensor completion approach for system identification. One of the earliest applications of tensor decomposition with smooth latent factors has been fluorescence data analysis [19, 31]. Recently, it is has been mostly proposed in the area of image processing. Specifically, CPD and Tucker models with smoothness constraints or regularization have been used for the recovery of incomplete 3- and 4-dimensional image data [48, 118]. To the best of our knowledge, tensor completion (with or without smooth latent factors) has not been considered yet as a tool for general nonlinear system identification.

5.3 Proposed Approach

5.3.1 Canonical System Identification (CSID)

We are given a training dataset of M input-output pairs $D = \{(\mathbf{x}_1, y_1), (\mathbf{x}_2, y_2), \ldots, (\mathbf{x}_M, y_M)\}$. Let us assume that all predictors are discrete and take values from a common alphabet $\mathcal{I} = \{1, \ldots, I\}$. The scalar output y_m is a nonlinear function of the input $\mathbf{x}_m$ distorted by some unknown noise

$$y_m = f\left(\mathbf{x}_m[1], \ldots, \mathbf{x}_m[N]\right) + \epsilon_m.$$

The nonlinear function $f : \{1, \ldots, I\}^N \to \mathbb{R}$ can be modeled as an N-way tensor $\mathcal{X}$ where each input vector $[\mathbf{x}_m[1], \ldots, \mathbf{x}_m[N]]$ can be viewed as a cell multi-index and the cell content is the estimated response of the system $\widehat{y}_m$. We are interested in building a model that minimizes the Mean Square Error (MSE) between the model predictions and the actual response.

However, it is evident that it is impossible to infer the response of unobserved data without any assumptions on $\mathcal{X}$. To alleviate this problem we aim for the principal components of the nonlinear operator by minimizing the tensor rank. Assuming a low-rank CPD model, the problem of finding the rank-R approximation which best fits our data can be formulated as

$$\begin{aligned}\min_{\mathcal{X}, \{\mathbf{A}_n\}_{n=1}^N} \; & \frac{1}{M} \sum_{m=1}^{M} \left(y_m - \mathcal{X}(\mathbf{x}_m[1], \ldots, \mathbf{x}_m[N])\right)^2 \\ & \qquad + \sum_{n=1}^{N} \rho \|\mathbf{A}_n\|_F^2 \\ \text{s.t.} \quad & \mathcal{X} = \sum_{r=1}^{R} \mathbf{A}_1[:, r] \odot \cdots \odot \mathbf{A}_N[:, r],\end{aligned} \tag{5.1}$$

where ρ is a regularization parameter. It is convenient to express the problem in the following equivalent form

$$\begin{aligned}\min_{\mathcal{X}, \{\mathbf{A}_n\}_{n=1}^N} \; & \frac{1}{M} \left\| \sqrt{\mathcal{W}} \circledast (\mathcal{Y} - \mathcal{X}) \right\|_F^2 + \sum_{n=1}^{N} \rho \|\mathbf{A}_n\|_F^2 \\ \text{subject to} \quad & \mathcal{X} = \sum_{r=1}^{R} \mathbf{A}_1[:, r] \odot \cdots \odot \mathbf{A}_N[:, r],\end{aligned} \tag{5.2}$$

where $\mathcal{W}$ is a tensor containing the number of times a particular data point $\mathbf{x} = [i_1, \dots, i_n]^T$ appears in the dataset and $\mathcal{Y}$ is a tensor containing the mean response of the corresponding data points. The equivalence between Problems (5.1), (5.2) is straightforward

$$\begin{aligned}
&\min \sum_{m=1}^{M} \Big(y_m - \mathcal{X}[x_m([1], \dots, x_m[N]]\Big)^2 \Leftrightarrow \\
&\min \sum_{i_1,\dots,i_N} \sum_{m' \in S_{i_1,\dots,i_N}} \big(y_m - \mathcal{X}[i_1, \dots, i_N]\big)^2 \Leftrightarrow \\
&\min \sum_{i_1,\dots,i_N} \sum_{m' \in S_{i_1,\dots,i_N}} \big(y_m - \mathcal{Y}[i_1, \dots, i_N] + \mathcal{Y}[i_1, \dots, i_N] - \mathcal{X}[i_1, \dots, i_N]\big)^2 \Leftrightarrow \\
&\min \sum_{i_1,\dots,i_N} \mathcal{W}[i_1, \dots, i_N]\big(\mathcal{Y}[i_1, \dots, i_N] - \mathcal{X}[i_1, \dots, i_N]\big)^2.
\end{aligned}$$

The set $S_{i_1,\dots,i_N}$ contains the indices the data point $[i_1, \dots, i_N]^T$ appears in the dataset. Oftentimes, datasets contain both categorical and ordinal predictors, the later, being either discrete or continuous. In the presence of ordinal predictors a desirable property of a regression model is having smooth prediction surfaces i.e., small variations in the input will cause small changes in the output. As an example consider the task of estimating students' grades in future courses based on their grades in past courses, an important topic in educational data mining as it can facilitate the creation of personalized degree paths which will potentially lead to timely graduation [94]. The predictors correspond to the grades in N past courses that a student has received and the the predicted response is the student's grade in a future course. We are interested in building a model that maps an N-dimensional discrete feature vector to the output response. Adding a smoothness constraint or regularization will guarantee that the model will produce similar outputs for two students that differ slightly in their past grades as they are likely to perform similarly in the future. Therefore, we propose augmenting the CPD tensor completion problem with smoothness regularization:

$$\begin{aligned}
\min_{\mathcal{X}, \{\mathbf{A}_n\}_{n=1}^N} \quad & \frac{1}{M} \left\| \sqrt{\mathcal{W}} \circledast (\mathcal{Y} - \mathcal{X}) \right\|_F^2 + \sum_{n=1}^{N} \rho \|\mathbf{A}_n\|_F^2 \\
& + \sum_{n=1}^{N} \mu_n \|\mathbf{T}_n \mathbf{A}_n\|_F^2 \qquad (5.3) \\
\text{s.t.} \quad & \mathcal{X} = \sum_{r=1}^{R} \mathbf{A}_1[:, r] \odot \cdots \odot \mathbf{A}_N[:, r],
\end{aligned}$$

where the matrix $\mathbf{T}_n$ is a smoothness promoting matrix typically defined as $\mathbf{T}_n \in \mathbb{R}^{(I_n - 1) \times I_n}$

with $\mathbf{T}_n[i,i] = 1$ and $\mathbf{T}_n[i,i+1] = -1$ or $\mathbf{T}_n \in \mathbb{R}^{(I_n-2)\times I_n}$ with $\mathbf{T}_n(i,i) = -1$, $\mathbf{T}_n(i,i+1) = 2$ and $\mathbf{T}_n(i,i+2) = -1$. We set $\mu_n = 0$ for categorical predictors and $\mu_n > 0$ otherwise. Penalizing the difference of consecutive row elements of a factor $\mathbf{A}_n$ guarantees that varying the n-th dimension and keeping the remaining fixed will have a small impact on the predicted response. Another appealing feature of the proposed smoothness regularization is that it can potentially measure feature importance. Note that the effect a variable will have in the prediction is minimized if each column of the corresponding factor is a constant number. Irrelevant features are more likely to have factors that vary slightly. On the contrary, factors associated with predictive features will have more variations and induce a larger penalty cost.

Remark: CPD can model any nonlinear operator for high-enough rank, but even for low ranks, it can model highly nonlinear operators such as

$$f_1(x_1,\ldots,x_N) = \prod_{n=1}^{N} \operatorname{sign}(x_n),$$
$$f_2(x_1,\ldots,x_N) = \sum_{n=1}^{N} \operatorname{sign}(x_n).$$

Comparing these equations with Equation (2.2) we can verify that the former corresponds to a rank-1 CPD model, while the later to rank N.

5.3.2 Tensor Completion: Identifiability

In this section we briefly review existing probabilistic and deterministic theoretical results on tensor recovery from a few samples. This is important because, using our approach of casting system identification as tensor completion, the results below directly yield new results on nonlinear multivariate system identification - even for systems of unbounded nonlinearity degree. Recovering a tensor from samples depends mainly on how the samples $(X_1,\ldots,X_N)$ are generated – randomly or systematically, and if randomly from what distribution – as well as the operational $\mathbf{A}_n$. Practical experience suggests that the generic sample complexity for randomly drawn point samples is proportional to the degrees of freedom $O(RNI)$ in the model. This has been proven for randomly drawn *linear* (generalized, aggregated) samples, but not yet for point samples [16].

An adaptive sampling method with an estimation algorithm has been proposed by [69] that provably recovers an N-th order rank-R tensor using $O(IR^{N-0,5}\mu^{N-1}N\log R)$ samples where μ is a coherence bound on the factor matrices. Later, [50] proposed a method that can recover an N-th order rank-R tensor with orthogonal factor matrices and random sampling using $O(I^{N/2}\mu^6 R^5 \log^4 I)$ samples. A necessary condition for these methods is that the rank needs to be less than the maximum outer dimension although a CPD model can be unique even if rank exceeds this bound. Probabilistic results on tensor completion have also been proven for incomplete tensors that have low mode-n ranks under certain incoherence conditions, relying on minimization of the sum of nuclear norms of the tensor unfoldings [34]. For $R < I$, the result in [119] can be used to show that for uniform random point samples, the sample complexity for our low-rank model is $O(\sqrt{R}I^{N/2}\log I)$.

Deterministic conditions based on a specific sampling strategy, namely fiber sampling, have been given by [104]. Necessary and sufficient conditions are provided which are dependent on the sampling pattern, assuming that the rank is low enough. The authors propose an eigenvalue decomposition algorithm and demonstrate exact recovery of low-rank incomplete tensors even when, less than one percent of the tensor entries are available. The authors have also extended the results in the case of not fully observed fibers. Finally, several *regular* sampling strategies are investigated and generic identifiability conditions are provided by [53].

5.3.3 Algorithm

The work-horse of tensor decomposition is the so-called Alternating Least Squares (ALS) algorithm. ALS is a special type of BCD which offers two distinct advantages: monotonic decrease of the cost function, and no need for parameter tuning. In this section, we propose an ALS approach to tackle Problem 5.3.

Tensors $\mathcal{W}, \mathcal{Y}$ despite being high-dimensional, are in general very sparse and optimized sparse tensor formats can offer huge memory and computational savings [103]. The idea of ALS is that we cyclically update variables $\{\mathbf{A}_n\}_{n=1}^N$ while fixing the remaining variables at their last updated values. Assume that we fix estimates $\mathbf{A}_n, \forall n \in [N] \setminus \{k\}$ we need to solve the following optimization problem

$$\begin{aligned}\min_{\mathbf{A}_k} & \big\| \widehat{\mathcal{W}}^{(k)} \circledast \mathcal{Y}^{(k)} - \widehat{\mathcal{W}}^{(k)} \circledast (\mathbf{Q}_k \mathbf{A}_k^T) \big\|_F^2 \\ & + \rho \|\mathbf{A}_k\|_F^2 + \mu_k \|\mathbf{T}_k \mathbf{A}_k\|_F^2,\end{aligned} \tag{5.4}$$

where $\mathbf{Q}_k = (\odot_{n\neq k}\mathbf{A}_n)$, $\widehat{\mathcal{W}} = \sqrt{\mathcal{W}}$ with the square root computed element-wise. Equivalently, we have

$$\begin{aligned}\min_{\mathbf{A}_k} \sum_{i_k=1}^{I_k} \left\| \operatorname{diag}(\widehat{\mathbf{w}}_{i_k}^k) \left(\mathbf{y}_{i_k}^k - \mathbf{Q}_k \mathbf{a}_i^k \right) \right\|_2^2 \\ + \rho \|\mathbf{A}_n\|_F^2 + \mu_k \|\mathbf{T}_k \mathbf{A}_n\|_F^2,\end{aligned} \tag{5.5}$$

where $\widehat{\mathbf{w}}_{i_k}^k = \widehat{\mathbf{W}}^{(k)}[:, i_k]$, $\mathbf{y}_{i_k}^k = \mathbf{Y}^{(k)}[:, i_k]$ and $\mathbf{a}_i^k = \mathbf{A}_k[i_k, :]^T$. Note that we do not need to instantiate $\mathbf{Q}_k$ because only the non-zero elements of the sparse vector $\widehat{\mathbf{w}}_{i_k}^k$ contribute to the cost function. The non-zero elements of $\widehat{\mathbf{w}}_{i_k}^k$ correspond to the observed data points for which the k-th variable takes the value i_k and therefore we need to compute the corresponding rows of the Khatri-Rao product. Problem 5.5 can be optimally solved by finding the solution to a set of linear equations obtained after setting the gradient to zero e.g., using the conjugate Gradient descent algorithm [11]. Simpler updates can be obtained by fixing all variables except for a single row of the factor $\mathbf{A}_k$. Let us fix every parameter except for the i_k-th row of $\mathbf{A}_k$

$$\begin{aligned}\min_{\mathbf{a}_i^k} \frac{1}{M} \|\operatorname{diag}(\widehat{\mathbf{w}}_{i_k}^k)(\mathbf{y}_i^k - \mathbf{Q}_k \mathbf{a}_i^k\|_2^2 + \rho \|\mathbf{a}_i^k\|_2^2 \\ + \mu_k \left\| \mathbf{a}_{i-1}^k - \mathbf{a}_i^k \right\|_2^2 + \mu_k \left\| \mathbf{a}_{i+1}^k - \mathbf{a}_i^k \right\|_2^2,\end{aligned} \tag{5.6}$$

The solution for $\mathbf{a}_i^k$ is given by

$$\begin{aligned}\mathbf{a}_i^k = (\mathbf{Q}_k^T \operatorname{diag}(\mathbf{w}_i)^2 \mathbf{Q}_k + (\rho + 2\mu_k)\mathbf{I})^{-1} \\ (\mathbf{Q}_k^T \operatorname{diag}(\mathbf{w}_i)^2 \mathbf{y}_i^k - \mu_k(\mathbf{a}_{i-1}^k + \mathbf{a}_{i+1}^k))\end{aligned} \tag{5.7}$$

which results in very lightweight row-wise updates. BCD algorithms usually offer faster convergence in terms of the cost function compared to stochastic algorithms for small or moderate size problems. For large-scale problems on the other hand, Stochastic Gradient Descent (SGD) can attain moderate solution accuracy faster than BCD. The merits of both alternating optimization and stochastic optimization can be combined by considering block-stochastic updates [116]. In this work, we propose an easy to implement ALS algorithm as our main goal is to present a fresh perspective on the nonlinear identification problem through low-rank tensor completion. Further algorithmic developments are underway, but beyond the scope of this first submission. Next, we show how the proposed approach can be extended to handle partially observed and multi-output regression tasks.

5.3.4 Missing Data

It is quite common in general to have observations with missing values for one or more predictors. For example, in the grade prediction task described in the introduction, the predictions for a student rely on the student's performance achieved in previously taken courses. Consider a student-grade matrix $\mathbf{D} \in \mathbb{R}^{M \times N}$ where our goal is to predict the N-th course. The matrix will be in general sparse since each student enrolls in only few of the available courses, and the selected courses vary from student to student.

Common approaches for handling missing data include (1) removal of observations with any missing values, (2) imputing the missing values before training e.g., by replacing them with the mean, median, or the mode, and (3) directly handling the imputation by the algorithm. Let $\mathcal{O} = \{o_1, \ldots, o_T\}$ and $\mathcal{M} = \{m_1, \ldots, m_L\}$ denote the indices of the observed and missing entries of a single observation respectively. Instead of ignoring observations with missing entries we aim at computing the expectation of the nonlinear function conditioned on the observed variables i.e., we set

$$\begin{aligned} f(\mathbf{x}_{\mathcal{O}}) &= \mathbb{E}_{\mathbf{x}_{\mathcal{M}}|\mathbf{x}_{\mathcal{O}}}[f(\mathbf{x}_{\mathcal{O}}, \mathbf{x}_{\mathcal{M}})] \\ &= \sum_{\mathbf{x}_{\mathcal{M}}} \mathsf{Pr}(\mathbf{x}_{\mathcal{M}}|\mathbf{x}_{\mathcal{O}}) f(\mathbf{x}_{\mathcal{O}}, \mathbf{x}_{\mathcal{M}}). \end{aligned} \tag{5.8}$$

Estimating the conditional probability $\mathsf{Pr}(\mathbf{x}_{\mathcal{M}}|\mathbf{x}_{\mathcal{O}})$ is not possible since the number of parameters grows exponentially with the number of missing entries. Given the low-rank structure of the nonlinear function we propose modeling the Probability Mass Function (PMF) using a nonnegative CPD model which is a universal model for PMF estimation [57]. For the sake of simplicity, we adopt a simple rank-one joint PMF model estimated via the empirical first-order marginals [46]. Without loss of generality assume that the first T predictors are known and the remaining missing, then, the expectation can be computed very efficiently

$$\begin{aligned} f(\mathbf{x}_{\mathcal{O}}) &= \mathbb{E}_{\mathbf{x}_{\mathcal{M}}|\mathbf{x}_{\mathcal{O}}}[f(\mathbf{x}_{\mathcal{O}}, \mathbf{x}_{\mathcal{M}})] \\ &= \mathcal{X}[i_1, \ldots, i_T, :, \ldots, :] \times_{T+1} \mathbf{p}_{T+1} \cdots \times_{T+L} \mathbf{p}_N \\ &= \sum_{r=1}^{R} \prod_{n=1}^{T} \mathbf{A}_n[i_n, r] \prod_{n=T+1}^{N} \mathbf{p}_n^T \mathbf{A}_n[:, r]. \end{aligned} \tag{5.9}$$

In this case, we minimize the squared error between the target value and the conditional expectation of the function. The modification can be easily incorporated in the ALS algorithm. Rich dependencies between the variables can also be captured using a higher-order PMF model, but we defer this discussion to follow-up work due to space limitations.

5.3.5 Multi-Output Regression

The proposed framework is quite flexible and can easily be extended to vector-valued functions $f : \{1, \ldots, I\}^N \to \mathbb{R}^K$. When there is no correlation between the output variables of a system, one can build K independent models, one for each output, and then use those models to independently predict each one of the K outputs. However, it is likely that the output values related to the same input are themselves correlated and often a better way is to build a single model capable of predicting simultaneously all K outputs. We can treat each different model as an N-way tensor and stack them together to build an $(N+1)$-way tensor. The new tensor model can be described by $N+1$ factors associated with the N predictors and an additional mode of dimension K, $\mathcal{X} = [\![\mathbf{A}_1, \ldots, \mathbf{A}_N, \mathbf{V}]\!]$. The vector-valued prediction for $[i_1, \ldots, i_N]^T$ is given by $\mathcal{X}[i_1, \ldots, i_N, j] = \sum_{r=1}^{R} \mathbf{V}[j, r] \prod_{n=1}^{N} \mathbf{A}_n[i_n, r]$. In matrix form we have

$$\mathcal{X}[i_1, \ldots, i_N, :] = \left(\mathbf{A}_1[i_1, :] \circledast \cdots \circledast \mathbf{A}_N[i_N, :]\right) \mathbf{V}^T$$

No modification is needed for the ALS updates. Depending on the application one may or may not need to apply smoothness regularization on $\mathbf{V}$.

5.4 Experiments

We evaluate the proposed approach in single output regression tasks using several datasets obtained from the UCI machine learning repository [76]. Our proposed approach is implemented in MATLAB using the Tensor Toolbox [7] for tensor operations. We then assess the ability of our model to handle missing predictors by hiding 30% of the data as well as its ability to predict vector valued responses. For each experiment we split the dataset into two sets, 80% used for training and 20% for testing, and run 10 Monte-Carlo simulations. Finally, we evaluate the performance of our approach in a challenging student grade prediction task using a real student grade dataset. For each method we tune the hyper-parameters using 5-fold cross-validation. We compare the performance of the different algorithms in terms of the Root Mean Square Error (RMSE).

Table 5.1: Comparison of RMSE performance of different models on UCI datasets without missing data.

Dataset	RR	SVR (RBF)	SVR (polynomial)	DT	MLP (5 Layer)	CSID
Energy Eff. (1)	2.91±0.17	2.68±0.17	4.09±0.49	0.56±0.03	**0.48±0.06 [50]**	**0.39±0.05**
Energy Eff. (2)	3.09±0.19	3.03±0.21	4.14±0.44	1.86±0.19	**0.97±0.14 [50]**	**0.57±0.09**
C. Comp. Strength	10.47±0.42	9.72±0.38	11.30±0.36	6.57±0.82	**4.92±0.63 [50]**	**4.67±0.50**
SkillCraft Master Table	1.68±1.61	**0.99±0.03**	1.22±0.05	1.03±0.04	1.00±0.03 [10]	**0.91±0.02**
Abalone	2.25±0.10	**2.19±0.08**	3.90±3.43	2.35±0.08	**2.09±0.09 [10]**	2.23±0.09
Wine Quality	0.76±0.02	**0.69±0.02**	1.01±0.39	0.75±0.03	0.72±0.02 [10]	**0.70±0.02**
Parkinsons Tel. (1)	7.51±0.11	6.66±0.14	7.89±0.88	**2.40±0.26**	3.60±0.18 [100]	**1.33±0.10**
Parkinsons Tel. (2)	9.75±0.15	9.14±0.17	10.04±0.43	**2.60±0.38**	5.01±0.19 [100]	**1.79±0.17**
C. Cycle Power Plant	5.51±0.09	4.13±0.09	8.00±0.19	**3.98±0.13**	4.06±0.11 [50]	**3.76±0.15**
Bike Sharing (1)	36.45±0.46	32.67±0.81	34.93±0.97	18.89±0.36	**14.81±0.44 [100]**	**15.17±0.44**
Bike Sharing (2)	122.65±2.87	113.18±1.73	117.25±2.01	42.06±2.06	**38.69±1.24 [100]**	**36.93±1.19**
Phys. Prop.	5.19±0.03	4.91±1.26	6.49±1.15	4.40±0.04	**4.20±0.05 [100]**	**4.21±0.04**

Table 5.2: Comparison of RMSE performance of different models on UCI datasets with 30% missing data.

Dataset	RR	SVR (RBF)	SVR (polynomial)	DT	MLP (5 Layer)	CSID
Energy Eff. (1)	3.01±0.15	3.38±0.27	6.88±0.63	2.57±0.49	**2.49±0.48 [10]**	**2.17±0.25**
Energy Eff. (2)	3.26±0.16	3.57±0.30	6.65±0.48	**2.64±0.28**	3.02±0.36 [10]	**2.48±0.22**
C. Comp. Strength	10.33±0.61	11.39±0.48	13.16±1.17	**9.90±1.05**	10.01±0.54 [10]	**9.69±0.79**
SkillCraft Master Table	1.79±1.63	**1.05±0.03**	1.61±0.33	1.08±0.03	1.10±0.04 [10]	**1.05±0.01**
Abalone	**2.27±0.07**	2.31±0.08	3.12±0.79	2.42±0.07	**2.28±0.07 [10]**	2.40±0.13
Wine Quality	0.76±0.02	**0.73±0.02**	0.93±0.21	0.78±0.02	**0.76±0.03 [10]**	0.78±0.02
Parkinsons Tel. (1)	7.52±0.11	6.91±0.13	8.12±0.11	**3.10±0.22**	5.90±0.28 [10]	**4.98±0.12**
Parkinsons Tel. (2)	9.76±0.18	9.38±0.21	10.68±0.23	**3.59±0.81**	7.67±0.18 [10]	**6.58±0.18**
C. Cycle Power Plant	5.51±0.09	6.16±0.15	10.45±0.31	**5.29±0.36**	5.33±0.07 [50]	**5.04±0.12**
Bike Sharing (1)	37.40±0.52	35.50±0.31	36.85±0.38	25.41±1.5	**21.51±0.83± [50]**	**23.89±0.19**
Bike Sharing (2)	123.81±1.26	127.06±1.55	130.20±1.13	**71.93±1.18**	**64.03±1.66 [50]**	75.65±1.51
Phys. Prop.	5.18±0.02	7.53±0.67	7.87±0.83	5.08±0.03	**4.99±0.09 [100]**	**4.70±0.03**

Table 5.3: Comparison of RMSE performance of different models on multi-output regression.

Dataset	RR	MLP (1 Layer)	MLP (3 Layer)	MLP (5 Layer)	DT	CSID
En. Eff. (2)	2.70±0.19	2.82±0.08 [50]	2.73±0.11[100]	2.67±0.11[10]	**2.19±0.19**	**2.01±0.14**
Park. Tel. (2)	12.19±0.09	7.59±0.21[250]	6.54±0.06[250]	6.18±0.42[250]	**3.37±0.39**	**2.85±0.22**
B. Shar. (2)	127.75±3.32	64.12±6.49[250]	43.60±1.95[100]	**42.25±1.22[100]**	46.21±1.20	**45.29±1.47**

5.4.1 UCI Datasets

We used four different machine learning algorithms as baselines, Ridge Regresion (RR), Support Vector Regression (SVR), Decision Tree (DT) and Multilayer Perceptrons (MLPs) using the implementation of scikit-learn [91]. For RR, SVR and MLP we standardize each ordinal feature such that it has zero mean and unit variance. Categorical features are transformed using one-hot encoding. For DT no preprocessing step is required. For our method, we fix the alphabet size to be $I = 25$ and use Lloyd-Max scalar quantizer for discretization of continuous predictors.

For the MLPs, we set the number of hidden layers to 1,3 or 5 and varied the number of nodes per layer $10, 50$, 100 and 250. We observed that in most cases the MLP with 5 hidden layers performed better than the 1 or 3 layer MLP and that further increasing the number of layers did not improve the performance.

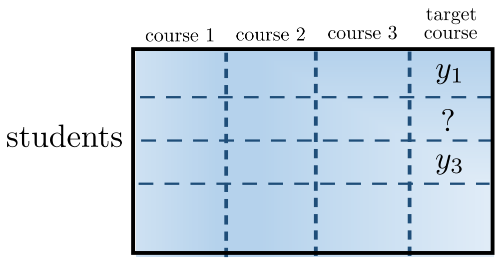
Table 5.1 shows the RMSE performance of the different methods when there are no missing predictors on the datasets. The number inside the square brackets denotes the number of nodes for each layer of MLP. We highlight the two best performing methods for each dataset. Our approach performs similarly or better than best baseline in most of the datasets. Note that both decision trees and our approach rely on discretization of continuous predictors however, adding the smooth regularization plays a significant role in boosting the RMSE performance for our method.

Next, we evaluate our approach on partially observed datasets. We randomly hide 30% of the full dataset and repeat 10 Monte-Carlo simulations. Before fitting the data to the baseline algorithms we replace each missing entry of an ordinal predictor with the mean and for each categorical predictor we use the most frequent value (mode). For our algorithm we use a rank-1 approximation of the joint PMF tensor estimated from the training data. Table 5.2 shows the performance of the different algorithms in this setting. Again, our approach similarly or better than best baseline.

Finally, we test our approach in predicting multi-output responses against RR, DT tree and MLPs. Table 5.3 contains the results for three datasets. Similarly to the single output setting our approach performs the same or slightly better compared to the baseline methods.

5.4.2 Grade Prediction Datasets

Finally we evaluate our method in a student grade prediction task on a real dataset obtained from the CS department of a university. The predictors correspond to the course grades the students have received. Specifically, we used the 20 most frequent courses to build 20 independent single output regression tasks each one of them having 34 predictors. Grades take 11 discrete values (A-F) and due to the natural ordering between the different values smoothness regularization was applied on all factors. We used the Grade Point Average (GPA) and Biased Matrix Factorization as our baselines. Low-rank matrix completion is considered a state-of-art method in student grade prediction [3, 94]. Note that in the matrix case each course is represented by a column while in the proposed tensor approach, each course is represented by a tensor mode (Figure 5.3).



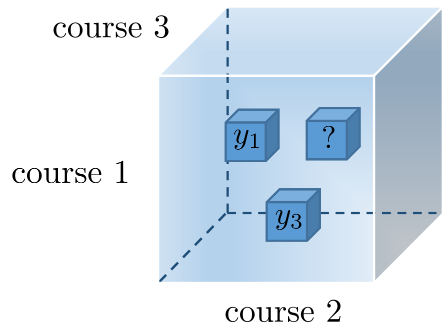


Figure 5.3: Low-rank matrix completion (left) canonical system identification (right).

Table 5.4: Comparison of RMSE performance on student grade data.

Dataset	GPA	BMF	CSID
CSCI-1	0.52±0.02	**0.48±0.03**	**0.48±0.03**
CSCI-2	0.56±0.02	**0.55±0.02**	**0.55±0.03**
CSCI-3	**0.48±0.04**	**0.48±0.04**	**0.48±0.05**
CSCI-4	0.53±0.03	0.52±0.04	**0.51±0.03**
CSCI-5	0.43±0.02	0.43±0.02	**0.42±0.02**
CSCI-6	0.63±0.03	0.58±0.03	**0.57±0.03**
CSCI-7	0.57±0.02	0.58±0.01	**0.56±0.02**
CSCI-8	0.52±0.02	0.49±0.03	**0.47±0.02**
CSCI-9	0.61±0.03	0.60±0.05	**0.57±0.03**
CSCI-10	0.58±0.04	**0.56±0.04**	**0.56±0.04**
CSCI-11	0.68±0.06	**0.66±0.04**	0.67±0.03
CSCI-12	0.58±0.04	0.51±0.04	**0.48±0.01**
CSCI-13	0.67±0.03	0.55±0.05	**0.54±0.03**
CSCI-14	0.70±0.06	**0.62±0.03**	0.65±0.07
CSCI-15	0.56±0.03	**0.56±0.06**	0.57±0.03
CSCI-16	0.52±0.03	0.51±0.03	**0.50±0.02**
CSCI-17	0.60±0.02	**0.58±0.05**	0.59±0.05
CSCI-18	0.57±0.03	0.56±0.05	**0.55±0.04**
CSCI-19	0.68±0.04	0.70±0.04	**0.61±0.04**
CSCI-20	0.61±0.06	**0.58±0.02**	0.63±0.04

Table 5.4 shows the results for the different algorithms. Our approach outperforms BMF in 11 tasks, performs the same in 4 and worse in 5.

5.5 Chapter Summary

In this paper, we considered the problem of nonlinear system identification. We formulated the problem as a smooth tensor completion problem with missing data and developed a lightweight BCD algorithm to tackle it. We have proposed a simple approach to handle randomly missing data and extended our model to vector valued function approximation. Experiments on several real data regression tasks showcased the effectiveness of the proposed approach.

Chapter 6

Canonical Decomposition of Multivariate Functions

Currently, the most popular methods for approximating nonlinear functions rely on neural networks. Despite their huge success, it is not well understood yet why they work so well, are difficult to interpret, and do not offer insights regarding the structure of the function being approximated. The universal approximation theorem [43] states that a feedforward neural network with a linear output layer and at least one hidden layer can approximate any continuous function on a compact set, provided that the number of hidden units is large enough. However, it is not guaranteed that the neural network will actually learn the true function; not only because training is a hard non-convex problem and the optimization algorithm may fail, but more importantly because the neural network may learn a different function which is also able to reproduce the training examples (i.e., one that agrees with the sought one over the training set). Two different networks that have fitted the training data equally well, may differ substantially on samples that have not been observed [37]. This is because the sought function may not be identifiable from the given input-output data. This motivates the study of models that possess identifiability guarantees, and learn unique mappings between the input-output data pairs. In this work, we take a step towards this direction and propose a novel method for learning a nonlinear function based on the Canonical Polyadic Decomposition (CPD), also known as Parallel Factor Analysis (PARAFAC) [40] of N-way tensors.

Tensor decomposition has been successfully applied in many fields such as machine learning, data mining, signal processing and statistics [63, 90, 98]. The CPD has been successful in modeling large multi-dimensional and multi-relational data [54, 112], understanding large knowledge bases [72, 109] and building recommender system models [55]. Tensor decomposition has also been used in deep learning, for speeding up and compressing neural network models [73, 86]. CPD is a powerful tool, that can help us build identifiable and parsimonious models which can approximate complex nonlinear functions [61]. Learning parsimonious models leads to significant reduction in the number of parameters, and identifiability of the parameters can offer important insights regarding the inner structure of a function.

In this paper, we tackle the problem of learning a nonlinear function, using a low-rank CPD model of a finite multidimensional Fourier series expansion of the unknown multivariate function. If a function is compactly supported and sufficiently smooth, then it can be well approximated using a finite sum of orthogonal complex exponentials. Finite orthogonal series expansion has been mainly used in univariate or bivariate regression tasks where an expansion based on cosine, wavelet or polynomial functions is used to predict a single output variable [29]. However, when dealing with multivariate functions with $N \gg 2$, the approach quickly becomes infeasible from the computation and memory point of view, since the number of series coefficients scales as $O\left(K^N\right)$, where K is the number of coefficients associated with each variable and N is the number of variables. To remedy this problem, we propose to decompose the N-way Fourier series tensor using CPD. This yields a *generalized canonical polyadic decomposition* (GCPD) representation of compactly supported multivariate functions. We present two methods for fitting the resulting model and evaluate our approach on several regression tasks. Our approach has the following important properties:

- **Expressiveness**: Our model is quite general, as it can model any compactly supported multivariate function that can be well approximated by a finite multidimensional Fourier series. The basis functions imply smoothness of the function so that the model is able to generalize, while the tensor rank and the number of Fourier coefficients are used to control the expressiveness of the model. Specifically, the rank and number of coefficients let us control the approximation error, which is bounded for a broad class of functions.

- **Efficiency**: The complexity of predicting the output response given an input vector scales linearly with the dimension of the input, which makes our approach suitable for large-scale problems.

- **Identifiability**: Under certain conditions, our approach guarantees that the unknown function is uniquely characterized by the given input-output data i.e., there is a single set of parameters that describes the nonlinear mapping between the input-output pairs. Identifiability of the parameters is necessary for model interpretability.

6.1 Related Work

Polynomial Networks (PN) and Factorization Machines (FM) are related approaches which rely on low-rank parameterization of the coefficient tensor of multivariate polynomials [15, 95]. The core model of the two approaches is the Polynomial Regression (PR) model. Polynomial regression is usually limited to low-order interactions since the computational complexity scales as $O\left(N^d\right)$, where d is the degree of the polynomial, rendering it impractical even for small number of variables. Both PN and FM use low-rank CPD models to break the curse of dimensionality. Their main difference is that FM uses a subset of all possible feature interactions, which makes the optimization problem multi-convex, and allows for a simple coordinate descent algorithm to be applied. FM has been extended to higher degree polynomials ($d > 3$) and has been very successful in sparse high-dimensional data regression tasks [14]. Other tensor decomposition models, such as the Tensor Train (TT) [88] and the Tucker model [110], have also been used for parameterizing polynomial functions [87, 92]. In [28] a tensor-based method was proposed that computes a decoupled representation of a vector-valued polynomial function. The proposed method builds a tensor that contains as frontal slabs the Jacobian matrix of the function, evaluated at a set of points and extracts a simpler function representation based on univariate polynomials. Its identifiability properties have been studied in [23].

Tensor decomposition has also been used for learning more general functions. For example, a TT decomposition approach has been proposed for learning a multidimensional Fourier series expansion of a function [113]. The authors have proposed a simple block coordinate descent algorithm with linear complexity on the input dimension. TT decomposition depends on the ordering of the input variables and has been shown to possess structure similar to that of a recurrent neural network [64], which is not desirable for non-sequential data. In addition, unlike the CPD, the model parameters of the TT decomposition are not identifiable, i.e., the parameterization is not unique and there exists a different set of parameters that can synthesize the same tensor.

Recently, a single high-order CPD model was used for modeling nonlinear functions with finite-alphabet inputs [61]. The problem was formulated as a tensor completion problem with smooth latent factors. The proposed approach cannot be directly applied to continuous inputs without a discretization step, which results in performance degradation when the discretization is coarse, or high complexity and poor generalization when the discretization is very fine. Generalizations of the CPD to multivariate functions of continuous inputs have been proposed in [101, 106]. The authors *assume* a low-rank model of the multivariate function where each component belongs to a reproducing kernel Hilbert space (RKHS). However, it is not clear what class of functions this model spans, and whether this is a valid assumption for a general multivariate function.

In this chapter, we propose a novel approach based on a generalized canonical polyadic decomposition suitable for modeling compactly supported multivariate functions with continuous inputs. Our model is very general since it only requires that the multivariate function has compact support and continuous derivatives. A drawback of the models in [101,106] is that the complexity depends on the number of training samples, which can be very high. For example, in the ALS algorithm developed in [106], the authors propose solving a kernel regression problem for each component at each ALS iteration, which is computationally very expensive. On the other hand, our approach is simple and computationally efficient – a parsimonious model that naturally lends itself to stochastic gradient updates, as we will see.

Last but not least, our use of multidimensional Fourier series to convert a continuous multivariate function approximation problem to finite multidimensional tensor decomposition is pleasing, being near and dear to our signal processing fundamentals. Note that unlike conventional uses of multivariate Fourier series, which are typically limited to 2-D or 3-D signals, our approach entails N-D Fourier series with N that can easily run in the order of $30 - 100$, and much higher in modern applications. Our judicious use of CPD modeling enables us to tackle problems in this regime with relative ease.

6.2 Background

In this section, we review necessary background on Fourier series that will prove useful for developing our method.

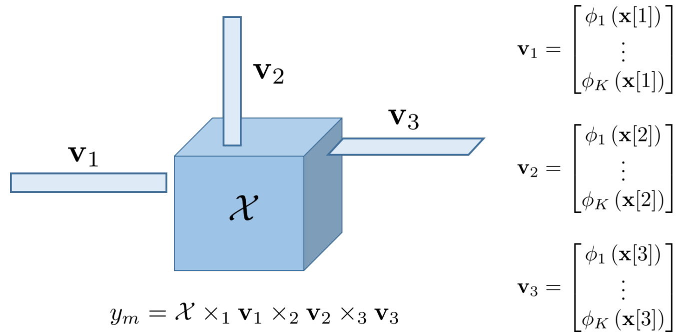


Figure 6.1: Fourier series representation of a function of three variables. Tensor $\mathcal{X} \in \mathbb{R}^{K \times K \times K}$ contains the Fourier coefficients and vectors $\mathbf{v}_1, \mathbf{v}_2, \mathbf{v}_3$ the values of the (cosine) basis functions for the input $\mathbf{x} \in \mathbb{R}^3$. The output is computed using the n-mode product.

6.2.1 Fourier Series

Fourier series can be used to represent a large class of periodic functions. A function $f : \mathbb{R} \to \mathbb{R}$ is periodic with period T, if $f(x + T) = f(x)\ \forall x \in \mathbb{R}$. One class of periodic functions that can be represented using a Fourier series, is square integrable functions $f \in L_2$ i.e., functions that have finite energy over a single period

$$\|f\|_{L_2}^2 := \int_T |f(x)|^2 dx < \infty. \tag{6.1}$$

Theorem 6.1. *(e.g., see [93][p. 10]) Every periodic function $f \in L_2$ has a unique representation of the form*

$$f(x) = \sum_{k \in \mathbb{Z}} \alpha_k e^{jk\omega_0 x},\ \alpha_k = \frac{1}{T} \int_T f(x) e^{-jk\omega_0 x} dx,$$

where $\omega_0 := \frac{2\pi}{T}$ and $(S_K f) := \sum_{k=-K}^{K} \alpha_k e^{jk\omega_0 x}$ converges in L_2 to f, i.e.,

$$\lim_{K \to \infty} \|S_K f - f\|_{L_2} = 0.$$

Fourier series can also be used for modeling non-periodic functions with compact support. Consider a univariate function $f(\cdot)$ supported on the interval $[0, 1]$. Note that restricting the domain of the function on the interval $[0, 1]$ can be done without loss of generality for any compactly supported function, by appropriate shift and scaling of the input. The function $f(\cdot)$ can be extended to $[-1,1]$ as an even or odd function, by defining $f(x) = f(-x)$ or $f(x) = -f(-x)$ for $x < 0$. The new function can then be made periodic with period $T = 2$, by demanding that $f(x) = f(x+2) = f(x-2)$. This is called the *even (odd) periodic extension* of $f(\cdot)$ and it can be shown that it is expressed by a Fourier cosine or sine series respectively. Using the even periodic extension we have

$$f(x) = \alpha_0 + \sum_{k=1}^{\infty} \alpha_k \sqrt{2} \cos(k\pi x), \tag{6.2}$$

using orthogonal basis functions $\phi_0(x) = 1$ and $\phi_k(x) = \sqrt{2}\cos(k\pi x)$, $k > 0$.

Similar to the univariate case, a multivariate function $f : [0,1]^N \to \mathbb{R}$, can be represented using a tensor-product basis as

$$f(\mathbf{x}) = \sum_{k_1=0}^{\infty} \cdots \sum_{k_N=0}^{\infty} \alpha_{\mathbf{k}} \prod_{n=1}^{N} \phi_{k_n}\left(\mathbf{x}[n]\right), \tag{6.3}$$

where $\mathbf{k} = (k_1, \ldots, k_N) \in \mathbb{N}^N$ is a multi-index . The idea of a series estimator is to approximate $f(\cdot)$ by a truncated series with cutoffs $K_1, \ldots, K_N$ i.e.,

$$f(\mathbf{x}) = \sum_{k_1=0}^{K_1-1} \cdots \sum_{k_N=0}^{K_N-1} \alpha_{\mathbf{k}} \prod_{n=1}^{N} \phi_{k_n}\left(\mathbf{x}[n]\right). \tag{6.4}$$

Note that Equation (6.4) defines an inner-product between two tensors. Let us define tensor $\mathcal{X}$ holding the Fourier coefficients with $\mathcal{X}[k_1, \ldots, k_N] = \alpha_{\mathbf{k}}$ and a rank-1 tensor $\mathcal{V} = [\![\mathbf{v}_1, \ldots, \mathbf{v}_N]\!]$, where $\mathbf{v}_n[k_n] = \phi_{k_n}(\mathbf{x}[n])$. Then, the inner between these two tensors is given by

$$\begin{aligned} \langle \mathcal{X}, \mathcal{V} \rangle &= (\mathbf{v}_N \otimes \cdots \otimes \mathbf{v}_1)^T \mathrm{vec}(\mathcal{X}) \\ &= \mathcal{X} \times_1 \mathbf{v}_1 \cdots \times_N \mathbf{v}_N. \end{aligned} \tag{6.5}$$

A visualization is shown in Figure 6.1 for a function of 3 variables.

Theorem 6.1 can be extended to multi-dimensional Fourier series as well. However, convergence of the Fourier series does not automatically imply that a truncated Fourier series will be a good approximation for $f(\cdot)$. The approximation error between the function and the truncated

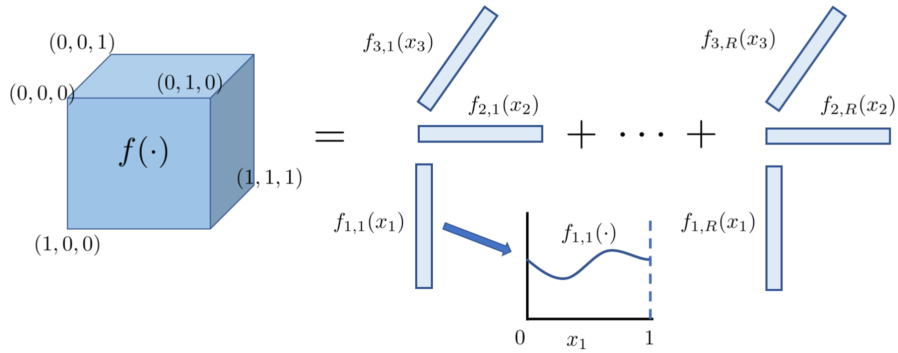


Figure 6.2: Generalization of the CPD to compactly supported multivariate functions. A compactly supported function $f : [0, 1]^3 \to \mathbb{R}$ is represented using a sum of R rank-1 components. Each rank-1 component is given by the product of three univariate functions.

series depends on how fast the Fourier coefficients tend to zero. The smoother the function is, the faster its Fourier coefficients and the error tend to zero as it is illustrated by the following theorem.

Theorem 6.2. *[93][p. 164] Let $p \in \mathbb{N}$. If the partial derivatives $\frac{\partial^{\beta_1}}{\partial x_1} \cdots \frac{\partial^{\beta_N}}{\partial x_N} f(\mathbf{x})$ of $f(\cdot)$ exist and are absolutely integrable for all $\beta_1, \ldots, \beta_N$ with $\sum_{n=1}^{N} \beta_n \leq p$ then*

$$\lim_{\|\mathbf{k}\|_2 \to \infty} (1 + \|\mathbf{k}\|_2^p) \alpha_{\boldsymbol{k}} = 0. \tag{6.6}$$

In other words, $\alpha_{\mathbf{k}}$ decays faster than the sequence $\frac{1}{1+\|\mathbf{k}\|_2^p}$. Finite orthogonal series expansion is mainly used in univariate or bivariate regression tasks where one has to estimate the Fourier coefficients from input-output data pairs.

6.3 Proposed Approach

In this section, we give a detailed description of our approach. Specifically, we present our model in Section 6.3.1, and present two different ways for training the model in Sections 6.3.2, 6.3.3.

6.3.1 Generalized Canonical Polyadic Decomposition

Our goal is to learn a high-dimensional compactly supported function $f : [0,1]^N \to \mathbb{R}$ from M input-output pairs $\{\mathbf{x}_m, y_m\}_{m=1}^M$

$$y_m = f(\mathbf{x}_m) + w_m, \quad m \in [M],$$

where w_m represents noise. The output y_m may correspond to a continuous value we want to estimate (regression) or a discrete class label (classification). Initially, we assume that the data points $\{\mathbf{x}_m\}_{m=1}^M$ are uniformly and independently sampled from the unit hypercube $[0,1]^N$ and $w_m \sim \mathcal{N}(0, \sigma^2)$.

We propose using a truncated cosine series expansion with cutoffs $K_1 = \cdots = K_N = K$. Orthogonal series approximation is mainly used when N is small. When dealing with functions of many variables the approach becomes impractical even for small K, since the number of parameters grows exponentially with N. Even if we are able to compute the series coefficients, the variance of the estimates can be high when M is small. More complicated estimation procedures are required in this case involving for example, hard or soft thresholding of the coefficients [29]. One way of dealing with these problems is to further restrict the class of functions, using for example, additive models of univariate functions [29, 41].

Instead, we propose a novel approach based on low-rank tensor decomposition. Specifically, we propose fitting a low-rank CPD on the N-way cosine series coefficient tensor. Let us assume a rank-R CPD model for the coefficient tensor $\mathcal{X}$

$$\mathcal{X}[k_1, \ldots, k_N] = \sum_{r=1}^{R} \prod_{n=1}^{N} \mathbf{a}_n^r[k_n], \tag{6.7}$$

where $\mathbf{a}_n^r \in \mathbb{R}^K$. Substituting Equation (6.7) in Equation (6.4), we have

$$\begin{aligned} f(\mathbf{x}) &= \sum_{k_1=0}^{K-1} \cdots \sum_{k_N=0}^{K-1} \sum_{r=1}^{R} \prod_{n=1}^{N} \mathbf{a}_n^r[k_n] \mathbf{v}_n[k_n] \\ &= \sum_{r=1}^{R} \sum_{k_2,\ldots,k_N=1}^{K-1} \prod_{n=2}^{N} \mathbf{a}_n^r[k_n] \mathbf{v}_n[k_n] \underbrace{\sum_{k_1=0}^{K-1} \mathbf{a}_1^r[k_1] \mathbf{v}_1[k_1]}_{f_{n,r}(\mathbf{x}[1])}, \end{aligned} \tag{6.8}$$

where $\mathbf{v}_n[k_n] = \phi_{k_n}(\mathbf{x}[n])$. We note that Equation (6.8) can also be understood using the mode-n multiplication property in Equation (2.7). By carrying out the computations in this manner, the expression for $f(\cdot)$ is greatly simplified to

$$f(\mathbf{x}) = \sum_{r=1}^{R} f_{1,r}(\mathbf{x}[1]) \cdots f_{N,r}(\mathbf{x}[N]). \tag{6.9}$$

Each univariate function in Equation (6.9) can be expressed as

$$f_{n,r}(\mathbf{x}[n]) = \sum_{k_n=0}^{K-1} \mathbf{a}_n^r[k_n]\mathbf{v}_n[k_n] = \mathbf{v}_n^T \mathbf{A}[:, r], \tag{6.10}$$

i.e., each component $f_{n,r}(\cdot)$ is a expressed using a truncated Fourier cosine series. Comparing Equations (2.2) and (6.9) we can see that our model is a generalization of the CPD to compactly supported multivariate functions. Any compactly supported multivariate function can be modeled using a countable tensor of Fourier coefficients. Continuity allows us to truncate this tensor to a finite one by choosing the number of Fourier coefficients. The CPD model is universal, i.e, any finite coefficient tensor $\mathcal{X}$ admits a CPD with finite rank, which implies that any compactly supported multivariate function that can be expressed (approximated) as a finite multivariate Fourier series can be modeled (approximated, respectively) as in Equation (6.9). The proposed generalized canonical polyadic decomposition is visualized for a function of three variables in Figure 6.2.

Employing this model, we circumvent the curse of dimensionality, as the number of parameters drops from $O(K^N)$ to $O(KNR)$. The output of the model given the input vector $\mathbf{x}$ is given via (6.9), (6.10) as

$$\widehat{y} = (\mathbf{v}_1^T \mathbf{A}_1 \circledast \cdots \circledast \mathbf{v}_N^T \mathbf{A}_N)\mathbf{1} = (\circledast_{n=1}^{N} \mathbf{v}_n^T \mathbf{A}_n)\mathbf{1}. \tag{6.11}$$

The complexity of computing the response of a new data point is $O(NKR)$, which makes our model suitable for high-dimensional data. The number of coefficients K controls the desired smoothness of the function and the rank controls the expressiveness. Next, we propose two different methods for learning the model.

Algorithm 6.1 FSA-LR & WFSA-LR

Input: $\mathbf{X}$, $\mathbf{y}$, R, K
// Initialization //
if $R \leq K$ **then**
  $\{\mathbf{A}_n\}_{n=1}^{N} \leftarrow$ CP-GEVD$(\mathcal{X})$
end if
repeat
  // Training the model //
  for $n = 1$ **to** N **do**
    Solve for $\mathbf{A}_n$ via (6.14)
  end for
until termination criterion is satisfied
// Refinement step //
repeat
  for $n = 1$ **to** N **do**
    for $k = 1$ **to** K **do**
      Solve for $\mathbf{A}_n[k, :]$ via (6.16)
    end for
  end for
until termination criterion is satisfied

6.3.2 Direct Optimization

Let us denote the training set by $\mathbf{X} = [\mathbf{x}_1, \ldots, \mathbf{x}_M]^T \in \mathbb{R}^{M \times N}$ and $\mathbf{y} = [y_1, \ldots, y_M]^T \in \mathbb{R}^M$. Our aim is to compute a low-rank representation of the Fourier series coefficients. The Fourier coefficients $\alpha_{\mathbf{k}}$ are given by

$$\alpha_{\mathbf{k}} = \int_0^1 \cdots \int_0^1 f(\mathbf{x}) \phi_{\mathbf{k}}(\mathbf{x}) d\mathbf{x},$$

where $\phi_{\mathbf{k}}(\mathbf{x}) = \prod_{n=1}^{N} \phi_{k_n}(\mathbf{x}[n])$. Assuming that $\{\mathbf{x}_m\}_{m=1}^{M}$ are uniformly sampled from the unit hypercube, a natural estimator for each coefficient $\alpha_{\mathbf{k}}$ is

$$\mathcal{X}[k_1, \ldots, k_N] = \widehat{\alpha}_{\mathbf{k}} = \frac{1}{M} \sum_{m=1}^{M} y_m \phi_{\mathbf{k}}(\mathbf{x}_m). \tag{6.12}$$

Equation (6.12) is an unbiased estimator for the coefficients i.e.,

$$\mathbb{E}[\widehat{\alpha}_{\mathbf{k}}] = \mathbb{E}\left[(f(\mathbf{x}) + w)\phi_{\mathbf{k}}(\mathbf{x})\right] = \alpha_{\mathbf{k}}.$$

Having obtained an estimate for each coefficient, a straightforward approach is to approximate $\mathcal{X}$ using a low-rank CPD model by minimizing the squared error

$$\min_{\{\mathbf{A}_n\}_{n=1}^N} \|\mathcal{X} - [\![\mathbf{A}_1, \ldots, \mathbf{A}_N]\!]_R\|_F^2 . \tag{6.13}$$

We refer to this approach as Fourier series Approximation with Low-Rank constraint (FSA-LR). Optimization problem (6.13) involves a multilinear form, and the most popular approach for tackling this kind of problem is via Alternating Least Squares (ALS) [98]. The idea is to cyclically update the variables while keeping all but one fixed. The updates for each factor matrix $\mathbf{A}_n$ are given by

$$\mathbf{A}_n \leftarrow \arg\min_{\mathbf{A}} \|\mathcal{X}^{(n)} - (\odot_{i\neq n}\mathbf{A}_i)\mathbf{A}^T\|_F^2. \tag{6.14}$$

When the tensor rank is small, CPD can be computed exactly. For example, if the factor matrices are full column rank, the CPD can be computed using a Generalized Eigenvalue Decomposition [40, 74]. In practice, $\mathcal{X}$ will never be low-rank due to noise. Nevertheless, in many practical applications, the associated tensors can be well-approximated using low-rank and the output of GEVD provides a good initial estimate.

When M is small, we expect the variance of the coefficient estimates to be large. Therefore we use a refinement step, by weighting each coefficient inversely proportional to its variance. An estimate of the variance can be obtained using the training set

$$\widehat{\sigma}^2_{k_1,\ldots,k_N} = \frac{1}{M}\sum_{m=1}^{M}\left(\mathcal{X}[k_1, \ldots, k_N] - y_m \phi_{\mathbf{k}}(\mathbf{x}_m)\right)^2 .$$

Let us define the tensor $\mathcal{W}$ as $\mathcal{W}[k_1, \ldots, k_N] = \widehat{\sigma}^{-1}_{k_1,\ldots,k_N}$. We refine the estimate obtained from (6.13) by solving

$$\min_{\{\mathbf{A}_n\}_{n=1}^N} \|\mathcal{W} \circledast (\mathcal{X} - [\![\mathbf{A}_1, \ldots, \mathbf{A}_N]\!]_R)\|_F^2 . \tag{6.15}$$

This is a weighted tensor factorizataion problem. Similarly as before, we can tackle the weighted tensor decomposition problem with an ALS algorithm. The optimization problem can be equivalently written as

$$\min_{\{\mathbf{A}_n\}_{n=1}^N} \left\|\widehat{\mathbf{X}}^{(n)} - \mathbf{W}^{(n)} \circledast \left((\odot_{i\neq n}\mathbf{A}_j)\, \mathbf{A}_n^T\right)\right\|_F^2 .$$

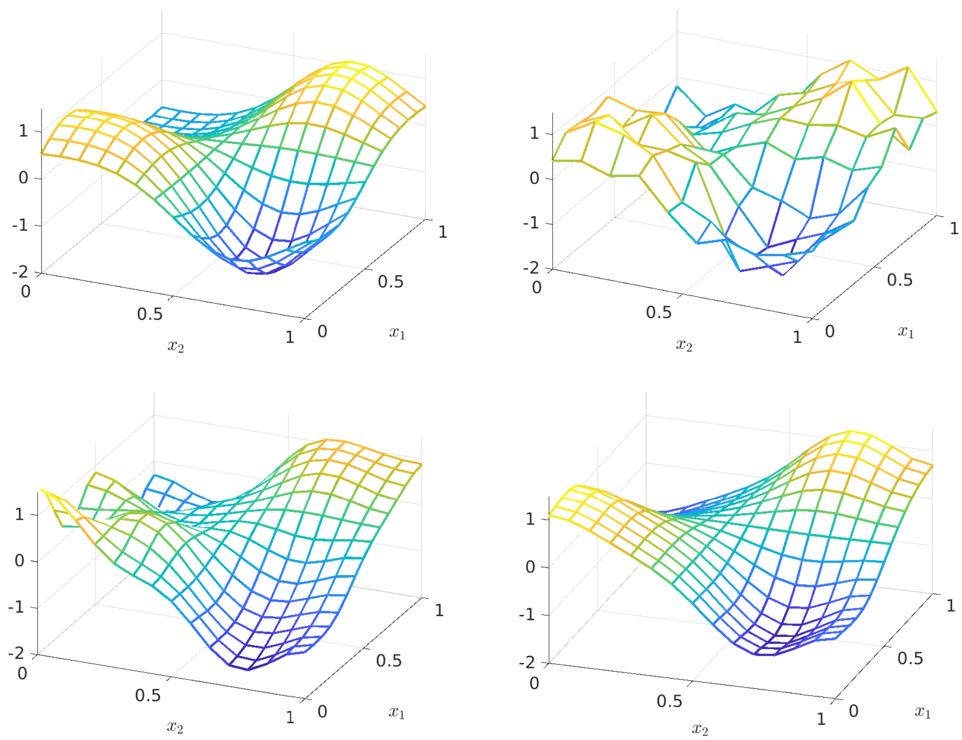


Figure 6.3: Toy example on learning a rank-1 bivariate function.

A simple update rule can be obtained by fixing all except for a single row of $\mathbf{A}_n$. The update for each row is given by

$$\mathbf{A}_n[k_n, :] \leftarrow \underset{\mathbf{a}_{n,k_n}}{\arg\min} \|\mathbf{z}_{n,k_n} - \mathbf{Q}_{n,k_n}\mathbf{a}_{n,k_n}\|_F^2 , \tag{6.16}$$

where

$$\mathbf{z}_{n,k_n} = \widehat{\mathbf{X}}^{(n)}[:, k_n], \mathbf{Q}_{n,k_n} = \mathrm{diag}(\mathbf{W}^{(n)}[:, k_n]) \left(\odot_{i \neq n}\mathbf{A}_j\right).$$

We refer to this approach as Weighted Fourier Series Approximation with Low-Rank constraint (WFSA-LR). The full procedure is shown in Algorithm 6.1.

Toy Example: An example to illustrate the effect of the weighted low-rank decomposition is shown in Figure 6.3. We generate $M = 100$ data points $\{\mathbf{x}_m, y_m\}_{m=1}^M$ according to

$$y_m = f_1(\mathbf{x}_m[1])f_2(\mathbf{x}_m[2]) + w_m,$$

which is a product of univariate functions and corresponds to a rank-1 model. The functions $f_1(\cdot), f_2(\cdot)$ are defined on the interval $[0, 1]$ as a mixture of Gaussians

$$f_n(x) = s_1^n \mathcal{N}\left(x; \mu_{1,n}, \sigma_{1,n}^2\right) + s_2^n \mathcal{N}\left(x; \mu_{2,n}, \sigma_{2,n}^2\right),$$

where $s_{i,r}^n \in \{-1, 1\}$.

The true function is shown in Figure 6.3.2. Figure 6.3.2 shows the output of the function for 100 uniformly sampled data points, distorted by white Gaussian noise. We use these points as our training set and compare Fourier Series Approximation (FSA) learned without assuming low-rank against FSA-LR and WFSA-LR with rank $R = 1$. Figure 6.3.2 shows the funtion learned without assuming a low-rank solution. WFSA-LR filters the noise and produces a smoother estimate as shown in Figure 6.3.2 (where 'plain' FSA-LR is not shown to save space). The improvement was also numerically verified by computing the Mean Squared Error (MSE) at 300 unseen points. The MSE of the different methods is 0.067 (FSA), 0.039 (FSA-LR), and 0.027 (WFSA-LR).

The downside of FSA-LR and WFSA-LR is that every Fourier coefficient needs to be estimated (but not necessarily stored, as it can be estimated on-the-fly during computations). Additionally, in order to have unbiased estimates of the coefficients, we require that the samples are uniformly drawn from the unit hypercube, something that we cannot guarantee in applications. In the following, we bypass both these restrictions.

6.3.3 Hidden Tensor Factorization

An alternative approach is to fit the coefficients directly on the training data by minimizing the error between the true and predicted response. Let us define matrices $\mathbf{V}_n \in \mathbb{R}^{K \times M}$ as $\mathbf{V}_n[k, m] = \phi_k(\mathbf{x}^m[n])$. Recall that each output response can be viewed as an inner product between two tensors as in Equation (6.5). We propose solving the following optimization problem

$$\min_{\{\mathbf{A}_n\}_{n=1}^N} \frac{1}{M} \left\|\mathbf{y} - (\odot_{n=1}^N \mathbf{V}_n)^T \left(\odot_{i=1}^N \mathbf{A}_i\right) \mathbf{1}\right\|_2^2 + \sum_{n=1}^N \rho \|\mathbf{A}_n\|_F^2,$$

where we have used the vectorized representation of the tensor $\mathcal{X}$ and ρ is a regularization parameter. We observed that adding this regularization term enhances the generalization ability of our method in pracice. The problem is conceptually similar to a linear least squares problem, but with a tensor rank-constrained solution [16]. Moreover, the measurement matrix has a very special structure, as each row corresponds to a rank-1 tensor. We refer to this approach as Fourier Series Approximation via Hidden Tensor Factorization (FSA-HTF), because we do not directly observe the full tensor. Instead, we aim at recovering the CPD model from rank-1 measurements of the 'hidden' tensor. Because of the special structure of the measurement matrix, we can avoid instantiating the tensor and the Khatri-Rao product. In scalar form we have

$$\min_{\{\mathbf{A}_n\}_{n=1}^N} \frac{1}{M} \sum_{m=1}^M \left(y_m - \left(\circledast_{n=1}^N (\mathbf{V}_n[:,m]^T \mathbf{A}_n)\right) \mathbf{1}\right)^2 + \sum_{n=1}^N \rho \|\mathbf{A}_n\|_F^2.$$

Similarly as before, we can develop a simple ALS algorithm. Fixing all variables except for $\mathbf{A}_n$ we have

$$\min_{\mathbf{A}_n} \frac{1}{M} \sum_{m=1}^M \left(y_m - \mathbf{V}_n[:,m]^T \mathbf{A}_n \mathbf{Q}_n[:,m]\right)^2 + \rho \left\|\mathbf{A}_n\right\|_F^2, \tag{6.17}$$

where $\mathbf{Q}_n \in \mathbb{R}^{R \times M}$ with

$$\mathbf{Q}_n = \left(\circledast_{i \neq n} (\mathbf{A}_i^T \mathbf{V}_i)\right). \tag{6.18}$$

Optimization problem (6.17) is a least squares problem. Setting the gradient equal to zero we have the following set of linear equations

$$\begin{aligned} \frac{1}{M}\sum_{m=1}^{M}\left(\mathbf{V}_n[:,m]^T\mathbf{A}_n\mathbf{Q}_n[:,m]\right)\mathbf{V}_n[:,m]\mathbf{Q}_n^T[:,m] \\ +\rho\mathbf{A}_n = \frac{1}{M}\sum_{m=1}^{M} y_m\mathbf{V}[:,m]\mathbf{Q}^T[:,m]. \end{aligned} \tag{6.19}$$

We solve the system of linear equations using conjugate gradient descent [11]. Conjugate gradient is an iterative algorithm with a complexity of $O(KRM)$ per each iteration.

When the number of samples M is large, we propose using Stochastic Gradient Descent (SGD). At each step we update all factors simultaneously by first sampling a batch $\mathcal{F} \subset \{1,\ldots,M\}$ of size $|\mathcal{F}|$ and taking a gradient step i.e,

$$\mathbf{A}_n \leftarrow \mathbf{A}_n - \alpha\mathbf{G}_n, \tag{6.20}$$

where α is the stepsize and $\mathbf{G}_n$ is the gradient with respect to the sampled points

$$\begin{aligned} \mathbf{G}_n = & \frac{1}{|\mathcal{F}|}\sum_{m\in\mathcal{F}}\left(\mathbf{V}_n[:,m]^T\mathbf{A}_n\mathbf{Q}_n[:,m]\right)\mathbf{V}_n[:,m]\mathbf{Q}_n^T[:,m] \\ & +\rho\mathbf{A}_n - \frac{1}{|\mathcal{F}|}\sum_{m=1}^{M} y_m\mathbf{V}[:,m]\mathbf{Q}^T[:,m]. \end{aligned}$$

Each step of SGD has a complexity of $O(NKR|\mathcal{F}|)$. For the termination of both algorithms we compute the MSE on a validation set and stop if a limit on the number of iterations has been reached, or the validation MSE has not improved in the last T iterations. The full procedures are shown in Algorithm 6.2 and Algorithm 6.3 respectively.

Initialization: It was empirically observed that random initialization was not effective in real data experiments for FSA-HTF. The algorithm exhibits better performance when starting from a smoother and simpler model. Therefore, when training FSA-HTF with ALS, we generate factor matrices $\mathbf{A}_n \in \mathbb{R}^{K_0\times F}$, where K_0 is a value smaller than K. If the current MSE on the validation stops improving, we augment factors $\mathbf{A}_n$ by adding an extra row. The algorithm is terminated when no improvement is observed on the validation set and $K_0 = K$. When training FSA-HTF via SGD, factors $\mathbf{A}_n$ are initially set to zero except for the first K_0 rows. For all

Algorithm 6.2 FSA-HTF (ALS)

Input: $\mathbf{X}$, $\mathbf{y}$, $\mathbf{X}_{\text{val}}$, $\mathbf{y}_{\text{val}}$ R, K_0, K, $\max_{\text{iter}}$
Initialize $\{\mathbf{A}_n \in \mathbb{R}^{K_0 \times F}\}_{n=1}^{N}$
repeat
  for $n = 1$ **to** N **do**
    Compute $\mathbf{Q}_n$ via (6.18)
    Solve for $\mathbf{A}_n$ via (6.19)
  end for
  Compute MSE_{val} using $\mathbf{X}_{\text{val}}$, $\mathbf{y}_{\text{val}}$
  if $K_0 < K$ and MSE_{val} has not improved **then**
    for $n = 1$ **to** N **do**
      $\mathbf{A}_n \leftarrow \begin{bmatrix} \mathbf{A}_n \\ \mathbf{0}^T \end{bmatrix}$
    end for
    $K_0 = K_0 + 1$
  end if
until $\max_{\text{iter}}$ is reached or MSE_{val} stops improving

experiments we set $K_0 = 2$.

6.3.4 Uniqueness of the proposed model

In this section, we briefly discuss conditions under which the proposed model is identifiable i.e., there exist unique factor matrices $\mathbf{A}_n$ that synthesize the tensor $\mathcal{X}$. This is important because it tells us when it is possible to recover the underlying nonlinear function, and hence learn the true mapping between the input and output pairs.

If the function $f(\cdot)$ can be written as a finite multidimensional Fourier series, then under certain rank conditions on the Fourier series coefficient tensor, the decomposition is unique. Specifically, given a decomposition of the Fourier coefficient tensor $\mathcal{X} = [\![\mathbf{A}_1, \ldots, \mathbf{A}_N]\!]_R$, if

$$\sum_{n=1}^{N} k_{\mathbf{A}_n} \geq 2R + N - 1,$$

then the factor matrices are unique up to common permutation and scaling/counterscaling of their columns [98]. Here, $k_{\mathbf{A}}$ denotes the *Kruskal rank* of the matrix $\mathbf{A}$, which is equal to the largest integer such that every subset of $k_{\mathbf{A}}$ columns are linearly independent. For generic $\mathbf{A}_n$ and maximal u such that $K \geq 2^u$, it holds that the decomposition of $\mathcal{X}$ is almost surely unique if

Algorithm 6.3 FSA-HTF (SGD)

Input: $\mathbf{X}$, $\mathbf{y}$, $\mathbf{X}_{\text{val}}$, $\mathbf{y}_{\text{val}}$ R, K, $|\mathcal{F}|$
Initialize $\{\mathbf{A}_n\}_{n=1}^{N}$
repeat
  Sample $|\mathcal{F}|$ data points
  for $n = 1$ **to** N **do**
    Update $\mathbf{A}_n$ via (6.20)
  end for
  Compute MSE_{val} using $\mathbf{X}_{\text{val}}$, $\mathbf{y}_{\text{val}}$
until $\max_{\text{iter}}$ is reached or MSE_{val} stops improving

$R \leq 2^{(N-1)(u-1)}$ [22]. The upshot is that under certain rank conditions on the Fourier coefficient tensor, it is possible to recover a unique representation of the multivariate function *from a finite set of input-output measurements*.

Remark: We should mention that the above identifiability results are derived for tensors under an exact low-rank decomposition. In practice, the Fourier series coefficient tensor will not have an exact low-rank decomposition mainly due to noise and/or limited number of samples. However, practical experience is that many real-life tensors can be well-approximated using low ranks and the uniqueness results often carry even in the approximate case – as we will also demonstrate in the experimental results section. Even when the low-rank assumption does not strictly hold, our approach is a very reasonable “principal component” approximation and can identify a controllable approximation of the true function.

6.4 Experimental Results

In this section, we evaluate the proposed approach on synthetic and real datasets.

6.4.1 Synthetic Data: Low-rank GCPD Model

As a first sanity check, we consider a function which can modeled using a low-rank generalized canonical polyadic decomposition model similar to Equation (6.9).

$$y_m = \sum_{r=1}^{R} \prod_{n=1}^{N} f_{n,r}(\mathbf{x}_m[n]) + w_m. \tag{6.21}$$

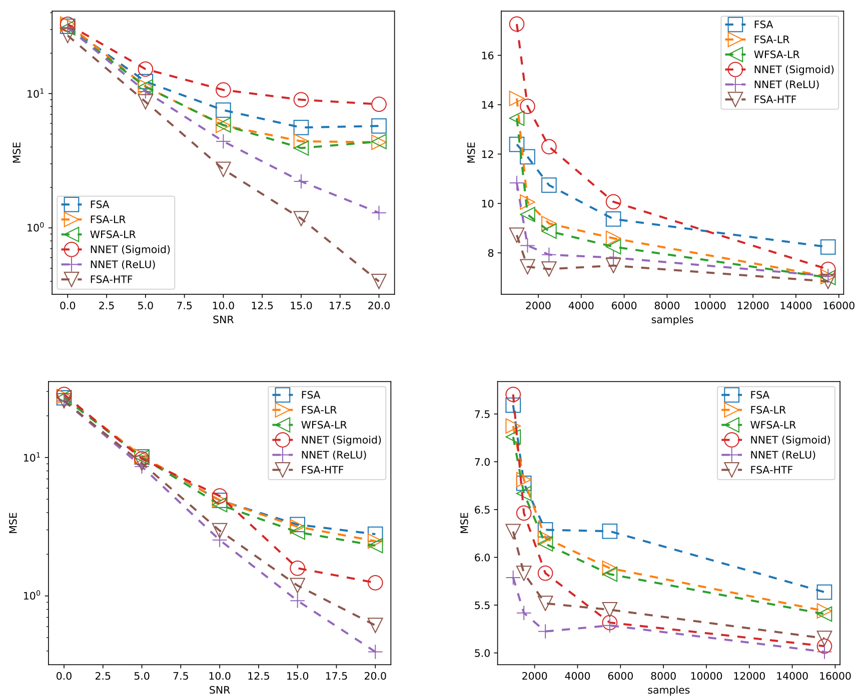


Figure 6.4: MSE on datasets generated using a mixture of products of univariate functions (top row) and a 1-Layer neural network using ReLU (bottom row).

Each function $f_{n,r}(\cdot)$ is defined on the interval $[0, 1]$ as follows

$$f_{n,r}(x) = \sum_{i=1}^{2} s_{i,r}^{n} \mathcal{N}\left(x; \mu_{i,r}^{n}, {\sigma_{i,r}^{n}}^{2}\right),$$

where $s_{i,r}^{n}$ takes the value $+1$ or -1 with equal probability, $\mu_{1,r}^{n} \sim \mathcal{U}(0.1, 0.4)$, $\mu_{2,r}^{n} \sim \mathcal{U}(0.6, 0.9)$ and $\sigma_{i,r}^{n} \sim \mathcal{U}(0.2, 0.5)$. We set $N = 5$, $R = 3$ and generate 1000 data points that are uniformly sampled from the $[0, 1]$ interval. We generate the corresponding outputs using Equation (6.21). We use 85% of the data as the training set and the remaining as the test set. We compare 6 different approaches; FSA (without low-rank assumption), FSA-LR, WFSA-LR, FSA-HTF, a feedforward neural network with sigmoid activation function and a feedforward neural network with ReLU activation function. The parameters of the different models, i.e., number of coefficients, rank, number of layers and number of nodes for each layer are tuned using 5-fold cross-validation. We repeat the above procedure 10 times and report the average results.

We first compare the performance of the methods while varying the Signal-to-Noise Ratio (SNR). Figure 6.4 (top left) shows the MSE on the test set as a function of the SNR. As expected, the performance of WFSA-LR is slightly better than FSA-LR. Both perform better than plain FSA but after a certain point they reach a plateau. FSA-HTF performs the best, as it generalizes better and is able to learn the true function. We observe that the choice of the nonlinear function is crucial as it highly affects the performance of the neural network. Specifically, the neural network with the sigmoid activation function performs the worst and fails to generalize to the test set. On the other hand, the neural network with the ReLU activation function performs reasonably well and its performance increases as we decrease the noise variance.

Next, we compare the methods while keeping the SNR fixed and increasing the number of training samples. We set SNR $= 5$ and fix the number of test samples to be 500. We vary the number of training samples from 500 to 15000. Figure 6.4 (top right) shows the MSE on the test set as a function of the total number of samples. Both FSA-HTF and the neural network with the ReLu activation function are able to learn the function using approximately 1000 samples. The performance of all methods improves as we increase the number of samples and we observe that using 15000 samples, all approaches except for FSA, perform similarly.

6.4.2 Synthetic Data: 1-Layer Neural Network

Here we consider a function which is not necessarily low-rank. We generate 1000 data points according to a neural network with 1 hidden layer with 20 nodes and ReLU activation function. We set the input dimension $N = 5$. Specifically, the input-output data pairs are generated according to

$$y = \mathbf{w}_2^T \mathrm{ReLU}(\mathbf{W}_1 \mathbf{x}), \tag{6.22}$$

where $\mathbf{W}_1 \in \mathbb{R}^{20\times 5}, \mathbf{w}_2 \in \mathbb{R}^{20}$. We use 85% of the data as the training set and the remaining as the test set. The parameters of the different models, i.e., number of coefficients, rank, number of layers and number of nodes for each layer are tuned using 5-fold cross-validation. For the two neural networks, we make sure that the true parameters are also examined during the cross-validation. We perform 10 Monte Carlo simulations and report the average results.

We compare the performance of the methods while varying the SNR. Figure 6.4 (bottom left) shows the MSE on the test set as a function of the SNR. As before, we observe that FSA,

Table 6.1: Datasets information.

DATASET	N	M
QSAR AQUATIC TOXICITY (QSAR)	9	546
ENERGY EFFICIENCY (EE)	8	788
AIRFOIL SELF-NOISE (ASN)	6	1503
SKILLCRAFT MASTER TABLE (SMT)	18	3337
ABALONE (AB)	8	4177
CYCLE POWER PLANT (CPP)	4	9568
SUPERCONDUCTIVTY (SC)	81	21263
PHYSICOCHEMICAL PROPERTIES (PP)	9	45730

FSA-LR and WFSA-LR quickly reach a plateau. The neural network which uses the correct activation function and agrees with the model that generated the data, performs the best. Note that misspecification of the activation function hurts the performance. The neural network which uses a sigmoid activation function performs much worse. FSA-HTF performs reasonably well and its performance increases as we decrease the noise variance.

Next, we compare the methods when keeping the SNR fixed $\text{SNR} = 5$ and increasing the number of training samples (Fig. 6.4 (bottom right)). The MSE drops as the number of samples increases for all methods. FSA-HTF and the two neural network approaches perform similarly when we increase the number of samples to 15000.

The above experiments serve as a sanity check that our method can indeed learn a function which admits a low-rank model, but can also learn functions that are not known to be low-rank – such as a neural network with random weights. Moreover, we see how important is the choice of the activation function and how it can affect the performance of a neural network.

6.4.3 Real Data

Next, we evaluate the proposed approach on several regression tasks using datasets obtained from the UCI machine learning repository [76]. The name and dimensions of each dataset are shown in Table 6.1. For real data experiments we evaluate only FSA-HTF since real data are not uniformly and independently sampled, hence violating the conditions assumed by FSA-LR. We compare our approach against various standard baselines: Linear Regression (LR), Polynomial Regression (PR), Support Vector Regression (SVR) with a Gaussian kernel, a Neural Network with ReLU activation function (NNET) and a Decision Tree (DT). We also compare our method against two other tensor methods, namely AMP [106] which assumes a low-rank generalized canonical polyadic decomposition model where each component belongs to a RKHS; and our own earlier CSID method [61] which leverages a low-rank CPD model but requires the input data to be discrete (or discretized).

Table 6.2: MSE performance of standard baseline models.

DATASET	LR	PR	SVR (RBF)	NNET	DT
QSAR	1.41 ± 0.31	1.41 ± 0.44	$\mathbf{1.19 \pm 0.30}$	1.42 ± 0.55	1.64 ± 0.37
EE [1]	8.15 ± 1.13	0.76 ± 0.08	4.69 ± 0.68	0.31 ± 0.08	0.30 ± 0.03
EE [2]	10.39 ± 1.84	2.87 ± 0.36	7.36 ± 1.66	2.03 ± 0.71	3.21 ± 0.26
ASN	23.27 ± 1.74	15.95 ± 1.21	11.09 ± 1.60	6.61 ± 1.24	6.45 ± 0.35
SMT	1.03 ± 0.06	1.02 ± 0.09	1.01 ± 0.05	1.03 ± 0.06	1.06 ± 0.04
AB	4.91 ± 0.30	4.54 ± 0.12	4.44 ± 0.21	$\mathbf{4.33 \pm 0.22}$	5.29 ± 0.22
CPP	19.57 ± 0.50	17.18 ± 0.41	15.60 ± 0.51	16.08 ± 0.50	$\mathbf{14.85 \pm 0.97}$
SC	315.93 ± 6.76	171.52 ± 5.41	213.60 ± 6.71	$\mathbf{125.12 \pm 8.13}$	130.56 ± 8.35
PP	27.10 ± 0.27	25.05 ± 1.06	21.24 ± 0.42	17.55 ± 0.38	19.24 ± 0.23

Table 6.3: MSE performance of CPD-based models.

DATASET	AMP [106]	CSID [61]	FSA-HTF (THIS PAPER)
QSAR	1.65 ± 0.50	1.48 ± 0.17	1.38 ± 0.53
EE [1]	0.31 ± 0.06	0.18 ± 0.02	$\mathbf{0.16 \pm 0.10}$
EE [2]	0.46 ± 0.10	0.34 ± 0.06	$\mathbf{0.24 \pm 0.09}$
ASN	8.73 ± 1.29	$\mathbf{3.05 \pm 0.53}$	4.01 ± 0.32
SMT	0.99 ± 0.05	0.98 ± 0.06	$\mathbf{0.94 \pm 0.06}$
AB	4.75 ± 0.3	4.95 ± 0.21	4.52 ± 0.30
CPP	N/A	15.25 ± 0.45	15.00 ± 0.66
SC	N/A	N/A	127.76 ± 11.86
PP	N/A	18.21 ± 0.45	$\mathbf{16.77 \pm 0.35}$

For each dataset we split the data into two sets; 85% used for training and 15% used for testing. The parameters for each method are chosen using 5-fold cross-validation and we report the average MSE of 5 Monte-Carlo simulations. When training the PR model, we test 2nd and 3rd order polynomials for all datasets except for SC due to the high computational cost. For the neural network model, we vary the number of layers from 1 to 5 and the number of nodes for each layer from 10 to 500. For FSA-HTF we vary the number of coefficients per mode from 5 to 50 and the rank from 5 to 30. We used alternating least squares for all datasets except for SC and PP for which we used SGD due to the larger number of samples. All the baselines except for CSID were implemented using the sklearn library [91].

The performance of the standard baseline models is shown in Table 6.2 and the performance of the tensor based models in Table 6.3. Among the standard baselines, the NNET approach works the best followed by the DT. The linear and polynomial models do not perform as well as they are usually too restrictive in practice, especially for larger datasets. Overall, we observe that the performance of our proposed method is very satisfactory, outperforming the baselines in 4 datasets and performing comparable to the winning method in the remaining ones. Our method outperforms the neural network in the smaller datasets (QSAR, EE, ASB and SMT) but also in two of the larger datasets (CPP, PP).

Compared to the tensor methods, FSA-HTF performs better in all except one dataset. AMP is a more complex method which assumes that each component $f_{n,r}(\cdot)$ belongs to a RKHS. It requires solving a kernel regression for each n and r at each step with a complexity of $O(M^3)$ making it not suitable even for datasets of moderate size. We did not run AMP for the larger datasets (CPP, SC, PP) since the computational complexity was very high and the algorithm did not terminate in a reasonable amount of time (> 1 day of training including cross validation). CSID is more suitable for datasets with discrete input as it requires a discretization step otherwise which leads to performance degradation. As shown in table 6.3, it perfoms worse than FSA-HTF in all but one datasets which may be due to the function not being sufficiently smooth.

6.5 Chapter Summary

In this chapter, we proposed a generalized CPD representation of functions which follows from compactness of support and continuous differentiability. Our method approximates a compactly supported multivariate function using a tensor of truncated multidimensional Fourier series coefficients and under a low-rank assumption in the Fourier domain, provides correct identification of the unknown function. We proposed two learning algorithms: one that explicitly constructs and decomposes the Fourier tensor (WFSA-LR), and another that learns the CPD model indirectly, without instantiating the Fourier tensor (FSA-HTF). In the synthetic data experiments, we observed that when the true function is indeed low-rank, WFSA-LR requires more data but eventually both methods converge to the same solution. For real data, FSA-HTF is more suitable since the conditions assumed by WFSA-LR are violated. Our algorithm FSA-HTF demonstrated promising results on real multivariate regression tasks with tensors of high-order using very low ranks.

Chapter 7

Thesis Summary and Future Directions

In this dissertation we studied the problems of distribution estimation and nonlinear function approximation. We developed tensor-based approaches which are principled, intuitive and come with identifiability guarantees. We addressed important and challenging problems that arise in real world applications and provided thorough experimental examination.

7.1 Summary

In chapter 3 we revealed an interesting link between tensors and multivariate statistics. We proved that it is possible to identify higher-dimensional joint probabilities from lower-dimensional ones, provided that the higher-order probability tensor has low-enough rank. We presented detailed identifiability analysis and provided a computational identification algorithm that is shown to work well on both simulated and real data.

In chapter 4 we studied the problem of non-parametric mixture model estimation. We built on the results of chapter 3 and developed a two-stage approach which recovers the component distributions of the mixture under a smoothness condition. We conducted both synthetic and real data experiments and showed that when the underlying conditional PDFs are smooth our method can recover them with high accuracy. Using real data, we showed that our method can work well in data clustering tasks.

In chapter 5, we showed that identifying a general nonlinear function from input-output examples can be formulated as a tensor completion problem. We provided conditions under which correct nonlinear system identification is possible. We extended our method to the multi-output setting and the case of partially observed data. Finally, we demonstrated the effectiveness of the approach using several regression tasks including some standard benchmarks and a challenging student grade prediction task.

In chapter 6, we extended the results of Chapter 5 to multivariate functions of continuous variables. We proposed a generalization of the canonical polyadic decomposition from tensors to multivariate functions and showed that under certain conditions our model guarantees identification of the unknown function from input-output data pairs. We developed two optimization algorithms and demonstrated promising results on synthetic and real multivariate regression tasks.

7.2 Future Directions

- Ensemble tensor models for supervised learning: In Chapter 5 we formulated the function approximation problem as a tensor completion problem. We have shown that it is possible to enhance the performance of the proposed algorithm termed Canonical System Identification (CSID) by utilizing ensemble learning techniques such as bagging and boosting [62]. A drawback of these methods is that they rely on discretization of the input features which is determined a priori and it is common for all input features. Our plan is to reformulate the random forest method [18] and propose adaptive algorithms for determining the discretization. We also intend to explore different tensor models such as the Tensor Train (TT) model.

- Non-parametric Multivariate Density Estimation: In chapters 3 and 4 we studied how to learn a joint distribution given i.i.d samples. In chapter 3 we assumed that the random variables of interest are discrete. In Chapter 4 we extended our method to continuous variables for a special probabilistic model – a mixture of product distributions. In this direction we intend to explore a wider class of probability distributions and develop more scalable algorithms. Ongoing work in this direction have been reported in [4].

References

[1] E. Acar, T. G. Kolda, and D. M. Dunlavy, "All-at-once optimization for coupled matrix and tensor factorizations," in *Proc. KDD Workshop Min. Learn. Graphs*, 2011.

[2] E. S. Allman, C. Matias, J. A. Rhodes *et al.*, "Identifiability of parameters in latent structure models with many observed variables," *The Annals of Statistics*, vol. 37, no. 6A, pp. 3099–3132, 2009.

[3] F. M. Almutairi, N. D. Sidiropoulos, and G. Karypis, "Context-aware recommendation-based learning analytics using tensor and coupled matrix factorization," *IEEE Journal of Selected Topics in Signal Processing*, vol. 11, no. 5, pp. 729–741, Aug 2017.

[4] M. Amiridi, N. Kargas, and N. D. Sidiropoulos, "Nonparametric multivariate density estimation: A low-rank characteristic function approach," *arXiv preprint arXiv:2008.12315*, 2020.

[5] A. Anandkumar, R. Ge, D. Hsu, and S. M. Kakade, "A tensor approach to learning mixed membership community models," *J. Mach. Learn. Res.*, vol. 15, no. 1, pp. 2239–2312, Jan. 2014.

[6] A. Anandkumar, R. Ge, D. Hsu, S. M. Kakade, and M. Telgarsky, "Tensor decompositions for learning latent variable models," *J. Mach. Learn. Res.*, vol. 15, no. 1, pp. 2773–2832, Aug. 2014.

[7] B. W. Bader and T. G. Kolda, "Efficient MATLAB computations with sparse and factored tensors," *SIAM Journal on Scientific Computing*, vol. 30, no. 1, pp. 205–231, December 2007.

[8] ——, “Efficient MATLAB computations with sparse and factored tensors,” *SIAM J. Sci. Comp.*, vol. 30, no. 1, pp. 205–231, 2008.

[9] A. Beck and M. Teboulle, “Mirror descent and nonlinear projected subgradient methods for convex optimization,” *Operations Research Letters*, vol. 31, no. 3, pp. 167–175, 2003.

[10] T. Benaglia, D. Chauveau, and D. R. Hunter, “An EM-like algorithm for semi-and non-parametric estimation in multivariate mixtures,” *Journal of Computational and Graphical Statistics*, vol. 18, no. 2, pp. 505–526, 2009.

[11] D. P. Bertsekas, *Nonlinear programming*. Athena scientific Belmont, 1999.

[12] A. Beutel, P. P. Talukdar, A. Kumar, C. Faloutsos, E. E. Papalexakis, and E. P. Xing, “FlexiFact: Scalable flexible factorization of coupled tensors on hadoop,” in *Proc. SIAM Int. Conf. Data Mining*, 2014, pp. 109–117.

[13] C. M. Bishop, *Pattern recognition and machine learning*. Springer, 2006.

[14] M. Blondel, A. Fujino, N. Ueda, and M. Ishihata, “Higher-order factorization machines,” in *Advances in Neural Information Processing Systems*, 2016, pp. 3351–3359.

[15] M. Blondel, M. Ishihata, A. Fujino, and N. Ueda, “Polynomial networks and factorization machines: New insights and efficient training algorithms,” in *Proceedings of the 33rd International Conference on Machine Learning*, 2016, pp. 850–858.

[16] M. Bousse, N. Vervliet, I. Domanov, O. Debals, and L. De Lathauwer, “Linear systems with a canonical polyadic decomposition constrained solution: Algorithms and applications,” *Numerical Linear Algebra with Applications*, vol. 25, no. 6, p. e2190, 2018.

[17] M. Boussé, O. Debals, and L. De Lathauwer, “Tensor-based large-scale blind system identification using segmentation,” *IEEE Transactions on Signal Processing*, vol. 65, no. 21, pp. 5770–5784, 2017.

[18] L. Breiman, “Random forests,” *Machine learning*, vol. 45, no. 1, pp. 5–32, 2001.

[19] R. Bro, “Multi-way analysis in the food industry,” Ph.D. dissertation, 1998.

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

[21] Y.-y. Chen, Y. Lv, Z. Li, and F.-Y. Wang, "Long short-term memory model for traffic congestion prediction with online open data," in *2016 IEEE 19th International Conference on Intelligent Transportation Systems (ITSC)*, 2016, pp. 132–137.

[22] L. Chiantini and G. Ottaviani, "On generic identifiability of 3-tensors of small rank," *SIAM J. Matrix Anal. Appl.*, vol. 33, no. 3, pp. 1018–1037, 2012.

[23] P. Comon, Y. Qi, and K. Usevich, "Identifiability of an x-rank decomposition of polynomial maps," *SIAM Journal on Applied Algebra and Geometry*, vol. 1, no. 1, pp. 388–414, 2017.

[24] A. P. Dawid and A. M. Skene, "Maximum likelihood estimation of observer error-rates using the EM algorithm," *Applied Statistics*, vol. 28, no. 1, pp. 20–28, 1979.

[25] A. P. Dempster, N. M. Laird, and D. B. Rubin, "Maximum likelihood from incomplete data via the EM algorithm," *Journal of the royal statistical society. Series B (methodological)*, pp. 1–38, 1977.

[26] I. Domanov and L. D. Lathauwer, "Canonical polyadic decomposition of third-order tensors: reduction to generalized eigenvalue decomposition," *SIAM Journal on Matrix Analysis and Applications*, vol. 35, no. 2, pp. 636–660, 2014.

[27] ——, "Generic uniqueness conditions for the canonical polyadic decomposition and INDSCAL," *SIAM J. Matrix Anal. Appl.*, vol. 36, no. 4, pp. 1567–1589, 2015.

[28] P. Dreesen, M. Ishteva, and J. Schoukens, "Decoupling multivariate polynomials using first-order information and tensor decompositions," *SIAM Journal on Matrix Analysis and Applications*, vol. 36, no. 2, pp. 864–879, 2015.

[29] S. Efromovich, *Nonparametric curve estimation: methods, theory, and applications*. Springer Science & Business Media, 2008.

[30] X. Fang and J. Zhan, "Sentiment analysis using product review data," *Journal of Big Data*, vol. 2, no. 1, p. 5, 2015.

[31] X. Fu, K. Huang, W.-K. Ma, N. D. Sidiropoulos, and R. Bro, "Joint tensor factorization and outlying slab suppression with applications," *IEEE Trans. Signal Process.*, vol. 63, no. 23, pp. 6315–6328, 2015.

[32] X. Fu, K. Huang, and N. D. Sidiropoulos, "On identifiability of nonnegative matrix factorization," *IEEE Signal Processing Letters*, vol. 25, no. 3, pp. 328–332, Mar. 2018.

[33] X. Fu, N. D. Sidiropoulos, J. H. Tranter, and W.-K. Ma, "A factor analysis framework for power spectra separation and multiple emitter localization," *IEEE Trans. Signal Process.*, vol. 63, no. 24, pp. 6581–6594, Aug. 2015.

[34] S. Gandy, B. Recht, and I. Yamada, "Tensor completion and low-n-rank tensor recovery via convex optimization," *Inverse Problems*, vol. 27, no. 2, p. 025010, 2011.

[35] E. Gaussier and C. Goutte, "Relation between PLSA and NMF and implications," in *Proc. 28th Annu. Int. ACM SIGIR Conf. Res. Develop. Inf. Retrieval*, 2005, pp. 601–602.

[36] C. A. Gomez-Uribe and N. Hunt, "The netflix recommender system: Algorithms, business value, and innovation," *ACM Transactions on Management Information Systems (TMIS)*, vol. 6, no. 4, pp. 1–19, 2015.

[37] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. MIT Press, 2016.

[38] O. Gottesman, W. Pan, and F. Doshi-Velez, "Weighted tensor decomposition for learning latent variables with partial data," in *Proceedings of the Twenty-First International Conference on Artificial Intelligence and Statistics*, vol. 84, Apr. 2018, pp. 1664–1672.

[39] F. M. Harper and J. A. Konstan, "The MovieLens datasets: History and context," *ACM Trans. Inter. Intel. Systems*, vol. 5, no. 4, pp. 1–19, Dec. 2016.

[40] R. A. Harshman, "Foundations of the PARAFAC procedure: Models and conditions for an "explanatory" multimodal factor analysis," *UCLA Working Papers Phonetics*, vol. 16, pp. 1–84, 1970.

[41] T. Hastie, R. Tibshirani, and J. Friedman, *The Elements of Statistical Learning*, ser. Springer Series in Statistics, 2009.

[42] T. Hofmann, “Probabilistic latent semantic indexing,” in *Proc. 22nd Annu. Int. ACM SIGIR Conf. Res. Develop. Inf. Retrieval*, 1999, pp. 50–57.

[43] K. Hornik, M. Stinchcombe, and H. White, “Multilayer feedforward networks are universal approximators,” *Neural networks*, vol. 2, no. 5, pp. 359–366, 1989.

[44] D. Hsu and S. M. Kakade, “Learning mixtures of spherical gaussians: moment methods and spectral decompositions,” in *Proceedings of the 4th conference on Innovations in Theoretical Computer Science*, 2013, pp. 11–20.

[45] K. Huang, X. Fu, and N. D. Sidiropoulos, “Learning hidden markov models from pairwise co-occurrences with applications to topic modeling,” *ArXiv preprint arXiv:1802.06894*, Feb. 2018.

[46] K. Huang and N. D. Sidiropoulos, “Kullback-Leibler principal component for tensors is not NP-hard,” in *Asilomar Conference on Signals, Systems, and Computers*, Oct 2017, pp. 693–697.

[47] K. Huang, N. D. Sidiropoulos, and A. P. Liavas, “A flexible and efficient algorithmic framework for constrained matrix and tensor factorization,” *IEEE Trans. Signal Process.*, vol. 64, no. 19, pp. 5052–5065, 2016.

[48] M. Imaizumi and K. Hayashi, “Tensor decomposition with smoothness,” in *International Conference on Machine Learning*, vol. 70, 06–11 Aug 2017, pp. 1597–1606.

[49] P. Jain, P. Netrapalli, and S. Sanghavi, “Low-rank matrix completion using alternating minimization,” in *Proc. 45th Annu. ACM Symp. Theory Comput.*, 2013, pp. 665–674.

[50] P. Jain and S. Oh, “Learning mixtures of discrete product distributions using spectral decompositions,” in *Conference on Learning Theory*, 2014, pp. 824–856.

[51] T. Jiang, N. D. Sidiropoulos, and J. M. F. ten Berge, “Almost-sure identifiability of multidimensional harmonic retrieval,” *IEEE Trans. Signal Process.*, vol. 49, no. 9, pp. 1849–1859, Sep. 2001.

[52] T. Jiang and N. D. Sidiropoulos, “Kruskal’s permutation lemma and the identification of CANDECOMP/PARAFAC and bilinear models with constant modulus constraints,” *IEEE Trans. Signal Process.*, vol. 52, no. 9, pp. 2625–2636, Sep. 2004.

[53] C. I. Kanatsoulis, X. Fu, N. D. Sidiropoulos, and M. Akçakaya, "Tensor completion from regular sub-Nyquist samples," *IEEE Transactions on Signal Processing*, vol. 68, pp. 1–16, 2019.

[54] U. Kang, E. Papalexakis, A. Harpale, and C. Faloutsos, "GigaTensor: Scaling tensor analysis up by 100 times - algorithms and discoveries," in *Proceedings of the 18th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2012, pp. 316–324.

[55] A. Karatzoglou, X. Amatriain, L. Baltrunas, and N. Oliver, "Multiverse recommendation: n-dimensional tensor factorization for context-aware collaborative filtering," in *Proceedings of the 4th ACM Conference on Recommender Systems*, 2010, pp. 79–86.

[56] N. Kargas, C. Qian, N. D. Sidiropoulos, C. Xiao, L. Glass, and J. Sun, "STELAR: Spatio-temporal tensor factorization with latent epidemiological regularization," in *35th AAAI Conference on Artificial Intelligence (AAAI)*, 2021.

[57] N. Kargas, N. D. Sidiropoulos, and X. Fu, "Tensors, learning, and "Kolmogorov extension" for finite-alphabet random vectors," *IEEE Transactions on Signal Processing*, vol. 66, no. 18, pp. 4854–4868, 2018.

[58] N. Kargas and N. D. Sidiropoulos, "Supervised learning and canonical decomposition of multivariate functions," *Under review.*

[59] ——, "Completing a joint PMF from projections: A low-rank coupled tensor factorization approach," in *Information Theory and Applications Workshop (ITA)*, 2017, pp. 1–6.

[60] ——, "Learning mixtures of smooth product distributions: Identifiability and algorithm," in *22nd International Conference on Artificial Intelligence and Statistics (AISTATS)*, 2019, pp. 388–396.

[61] ——, "Nonlinear system identification via tensor completion," in *34th AAAI Conference on Artificial Intelligence (AAAI)*, vol. 34, no. 04, 2020, pp. 4420–4427.

[62] ——, "Supervised learning via ensemble tensor completion," in *Asilomar Conference on Signals, Systems, and Computers*, 2020.

[63] N. Kargas, N. D. Sidiropoulos, and X. Fu, "Tensors, learning, and "Kolmogorov extension" for finite-alphabet random vectors," *IEEE Transactions on Signal Processing*, vol. 66, no. 18, pp. 4854–4868, 2018.

[64] V. Khrulkov, A. Novikov, and I. V. Oseledets, "Expressive power of recurrent neural networks," in *Proceedings of the 6th International Conference on Learning Representations*, 2018.

[65] J. Kivinen and M. K. Warmuth, "Exponentiated gradient versus gradient descent for linear predictors," *Information and Computation*, vol. 132, no. 1, pp. 1–63, 1997.

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

[67] D. Koller and N. Friedman, *Probabilistic graphical models: principles and techniques*. MIT Press, 2009.

[68] Y. Koren, R. Bell, and C. Volinsky, "Matrix factorization techniques for recommender systems," *Computer*, vol. 42, no. 8, pp. 30–37, Aug. 2009.

[69] A. Krishnamurthy and A. Singh, "Low-rank matrix and tensor completion via adaptive sampling," in *Advances in Neural Information Processing Systems*, 2013, pp. 836–844.

[70] J. B. Kruskal, "Three-way arrays: rank and uniqueness of trilinear decompositions, with application to arithmetic complexity and statistics," *Linear Algebra Appl.*, vol. 18, no. 2, pp. 95–138, 1977.

[71] H. W. Kuhn, "The Hungarian method for the assignment problem," *Naval research logistics quarterly*, vol. 2, no. 1-2, pp. 83–97, 1955.

[72] T. Lacroix, N. Usunier, and G. Obozinski, "Canonical tensor decomposition for knowledge base completion," in *International Conference on Machine Learning*, 2018, pp. 2863–2872.

[73] V. Lebedev, Y. Ganin, M. Rakhuba, I. Oseledets, and V. Lempitsky, "Speeding-up convolutional neural networks using fine-tuned CP-decomposition," in *Proceedings of the 3rd International Conference on Learning Representations*, 2015.

[74] S. Leurgans, R. Ross, and R. Abel, “A decomposition for three-way arrays,” *SIAM Journal on Matrix Analysis and Applications*, vol. 14, no. 4, pp. 1064–1083, 1993.

[75] M. Levine, D. R. Hunter, and D. Chauveau, “Maximum smoothed likelihood for multivariate mixtures,” *Biometrika*, pp. 403–416, 2011.

[76] M. Lichman, “UCI machine learning repository,” 2013. [Online]. Available: http://archive.ics.uci.edu/ml

[77] L.-H. Lim and P. Comon, “Nonnegative approximations of nonnegative tensors,” *J. Chemometr.*, vol. 23, no. 7-8, pp. 432–441, Jul. 2009.

[78] S. Liu, S. Liu, W. Cai, S. Pujol, R. Kikinis, and D. Feng, “Early diagnosis of alzheimer's disease with deep learning,” in *2014 IEEE 11th international symposium on biomedical imaging (ISBI)*, 2014, pp. 1015–1018.

[79] D. Lowd and P. Domingos, “Naive Bayes models for probability estimation,” in *Proc. Int. Conf. Mach. Learn.*, 2005, pp. 529–536.

[80] X. Ma, P. Zhang, S. Zhang, N. Duan, Y. Hou, M. Zhou, and D. Song, “A tensorized transformer for language modeling,” in *Advances in Neural Information Processing Systems*, vol. 32, 2019, pp. 2232–2242.

[81] M. Mardani, G. Mateos, and G. B. Giannakis, “Subspace learning and imputation for streaming big data matrices and tensors,” *IEEE Transactions on Signal Processing*, vol. 63, no. 10, pp. 2663–2677, 2015.

[82] G. J. McLachlan and D. Peel, *Finite mixture models*. Wiley Series in Probability and Statistics, 2000.

[83] A. Mnih and R. R. Salakhutdinov, “Probabilistic matrix factorization,” in *Proc. Adv. Neural Inf. Process. Syst.*, 2008, pp. 1257–1264.

[84] A. Y. Ng and M. I. Jordan, “On discriminative vs. generative classifiers: A comparison of logistic regression and naive Bayes,” in *Proc. Adv. Neural Inf. Process. Syst.*, 2002, pp. 841–848.

[85] D. Nion, K. N. Mokios, N. D. Sidiropoulos, and A. Potamianos, "Batch and adaptive PARAFAC-based blind separation of convolutive speech mixtures," *IEEE Trans. Audio, Speech, Lang. Process.*, vol. 18, no. 6, pp. 1193–1207, Aug. 2010.

[86] A. Novikov, D. Podoprikhin, A. Osokin, and D. P. Vetrov, "Tensorizing neural networks," in *Advances in Neural Information Processing Systems*, 2015, pp. 442–450.

[87] A. Novikov, M. Trofimov, and I. Oseledets, "Exponential machines," in *International Conference on Learning Representations Workshop.*, 2016.

[88] I. V. Oseledets, "Tensor-train decomposition," *SIAM Journal on Scientific Computing*, vol. 33, no. 5, pp. 2295–2317, 2011.

[89] E. E. Papalexakis, L. Akoglu, and D. Ience, "Do more views of a graph help? community detection and clustering in multi-graphs," in *International Conference on Information FUSION*, 2013, pp. 899–905.

[90] E. E. Papalexakis, C. Faloutsos, and N. D. Sidiropoulos, "Tensors for data mining and data fusion: Models, applications, and scalable algorithms," *ACM Transactions on Intelligent Systems and Technology (TIST)*, vol. 8, no. 2, pp. 1–44, 2016.

[91] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay, "Scikit-learn: Machine learning in Python," *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.

[92] I. Perros, F. Wang, P. Zhang, P. Walker, R. Vuduc, J. Pathak, and J. Sun, "Polyadic regression and its application to chemogenomics," in *SIAM International Conference on Data Mining*, 2017, pp. 72–80.

[93] G. Plonka, D. Potts, G. Steidl, and M. Tasche, *Numerical Fourier Analysis*. Springer, 2018.

[94] A. Polyzou and G. Karypis, "Grade prediction with models specific to students and courses," *International Journal of Data Science and Analytics*, vol. 2, no. 3, pp. 159–171, Dec 2016.

[95] S. Rendle, "Factorization machines," in *IEEE International Conference on Data Mining*, Dec 2010, pp. 995–1000.

[96] M. Ruffini, R. Gavalda, and E. Limon, "Clustering patients with tensor decomposition," in *Machine Learning for Healthcare Conference*, 2017, pp. 126–146.

[97] A. Shashua and T. Hazan, "Non-negative tensor factorization with applications to statistics and computer vision," in *Proc. Int. Conf. Mach. Learn.*, 2005, pp. 792–799.

[98] N. D. Sidiropoulos, L. De Lathauwer, X. Fu, K. Huang, E. E. Papalexakis, and C. Faloutsos, "Tensor decomposition for signal processing and machine learning," *IEEE Trans. Signal Process.*, vol. 65, no. 13, pp. 3551–3582, Jul. 2017.

[99] N. D. Sidiropoulos, R. Bro, and G. B. Giannakis, "Parallel factor analysis in sensor array processing," *IEEE Trans. Signal Process.*, vol. 48, no. 8, pp. 2377–2388, Aug. 2000.

[100] N. D. Sidiropoulos, G. B. Giannakis, and R. Bro, "Blind PARAFAC receivers for DS-CDMA systems," *IEEE Trans. Signal Process.*, vol. 48, no. 3, pp. 810–823, Mar. 2000.

[101] M. Signoretto, L. De Lathauwer, and J. A. Suykens, "Learning tensors in reproducing kernel hilbert spaces with multilinear spectral penalties," *arXiv preprint*, 2013.

[102] S. Smith, N. Ravindran, N. D. Sidiropoulos, and G. Karypis, "SPLATT: Efficient and parallel sparse tensor-matrix multiplication," in *IEEE Inter. Par. Dist. Process. Symposium*, May 2015, pp. 61–70.

[103] S. Smith and G. Karypis, "Tensor-matrix products with a compressed sparse tensor," in *Workshop on Irregular Applications: Architectures and Algorithms*, 2015, p. 5.

[104] M. Sorensen and L. De Lathauwer, "Fiber sampling approach to canonical polyadic decomposition and application to tensor completion," *SIAM Journal on Matrix Analysis and Applications*, vol. 40, no. 3, pp. 888–917, 2019.

[105] A. Stegeman, J. M. T. Berge, and L. D. Lathauwer, "Sufficient conditions for uniqueness in CANDECOMP/PARAFAC and INDSCAL with random component matrices," *Psychometrika*, vol. 71, no. 2, pp. 219–229, 2006.

[106] T. Suzuki, H. Kanagawa, H. Kobayashi, N. Shimizu, and Y. Tagami, "Minimax optimal alternating minimization for kernel nonparametric tensor learning," in *Advances in Neural Information Processing Systems*, 2016, pp. 3783–3791.

[107] G. Tomasi and R. Bro, "PARAFAC and missing values," *Chemometrics and Intelligent Laboratory Systems*, vol. 75, no. 2, pp. 163–180, 2005.

[108] P. A. Traganitis, A. Pages-Zamora, and G. B. Giannakis, "Blind multiclass ensemble classification," *IEEE Transactions on Signal Processing*, vol. 66, no. 18, pp. 4737–4752, 2018.

[109] T. Trouillon, J. Welbl, S. Riedel, É. Gaussier, and G. Bouchard, "Complex embeddings for simple link prediction," in *Proceedings of the 33rd International Conference on Machine Learning*, 2016, pp. 2071–2080.

[110] L. R. Tucker, "Some mathematical notes on three-mode factor analysis," *Psychometrika*, vol. 31, no. 3, pp. 279–311, 1966.

[111] F. Van Eeghem, O. Debals, N. Vervliet, and L. De Lathauwer, "Coupled and incomplete tensors in blind system identification," *IEEE Transactions on Signal Processing*, vol. 66, no. 23, pp. 6137–6147, Dec 2018.

[112] N. Vervliet, O. Debals, L. Sorber, and L. De Lathauwer, "Breaking the curse of dimensionality using decompositions of incomplete tensors: Tensor-based scientific computing in big data analysis," *IEEE Signal Processing Magazine*, vol. 31, no. 5, pp. 71–79, 2014.

[113] S. Wahls, V. Koivunen, H. V. Poor, and M. Verhaegen, "Learning multidimensional fourier series with tensor trains," in *Proceedings of the IEEE Global Conference on Signal and Information Processing*, 2014, pp. 394–398.

[114] W. Wang and M. A. Carreira-Perpinán, "Projection onto the probability simplex: An efficient algorithm with a simple proof, and an application," *ArXiv preprint arXiv:1309.1541*, 2013.

[115] L. Xiong, X. Chen, T.-K. Huang, J. Schneider, and J. G. Carbonell, "Temporal collaborative filtering with bayesian probabilistic tensor factorization," in *SIAM International Conference on Data Mining*, 2010, pp. 211–222.

[116] Y. Xu and W. Yin, "Block stochastic gradient iteration for convex and nonconvex optimization," *SIAM Journal on Optimization*, vol. 25, no. 3, pp. 1686–1716, 2015.

[117] B. Yaman, S. Weingärtner, N. Kargas, N. D. Sidiropoulos, and M. Akçakaya, "Low-rank tensor models for improved multidimensional mri: Application to dynamic cardiac t_1 mapping," *IEEE transactions on computational imaging*, vol. 6, pp. 194–207, 2019.

[118] T. Yokota, Q. Zhao, and A. Cichocki, "Smooth PARAFAC decomposition for tensor completion," *IEEE Transactions on Signal Processing*, vol. 64, no. 20, pp. 5423–5436, Oct 2016.

[119] M. Yuan and C.-H. Zhang, "On tensor completion via nuclear norm minimization," *Foundations of Computational Mathematics*, vol. 16, no. 4, pp. 1031–1068, Aug 2016.

[120] N. L. Zhang, "Hierarchical latent class models for cluster analysis," *J. Mach. Learn. Res.*, vol. 5, no. 6, pp. 697–723, Jun. 2004.

[121] Y. Zhang, X. Chen, D. Zhou, and M. I. Jordan, "Spectral methods meet EM: A provably optimal algorithm for crowdsourcing," in *Advances in neural information processing systems*, 2014, pp. 1260–1268.