Learning Latent Variable Models: Efficient Algorithms and Applications

Matteo Ruffini
PhD program in Artificial Intelligence

Universitat Politècnica de Catalunya
Department of Computer Science

Supervisor: **Ricard Gavaldà**

Abstract

Learning latent variable models is a fundamental machine learning problem, and the models belonging to this class – which include topic models, hidden Markov models, mixture models and many others – have a variety of real-world applications, like text mining, clustering and time series analysis. For many practitioners, the decade-old Expectation Maximization method (EM) is still the tool of choice, despite its known proneness to local minima and long running times. To overcome these issues, algorithms based on the spectral method of moments have been recently proposed. These techniques recover the parameters of a latent variable model by solving – typically via tensor decomposition – a system of non-linear equations relating the low-order moments of the observable data with the parameters of the model to be learned. Moment-based algorithms are in general faster than EM as they require a single pass over the data, and have provable guarantees of learning accuracy in polynomial time. Nevertheless, methods of moments have room for improvements: their ability to deal with real-world data is often limited by a lack of robustness to input perturbations. Also, almost no theory studies their behavior when some of the model assumptions are violated by the input data. Extending the theory of methods of moments to learn latent variable models and providing meaningful applications to real-world contexts is the focus of this thesis.

Assuming data to be generated by a certain latent variable model, the standard approach of methods of moments consists of two steps: first, finding the equations that relate the moments of the observable data with the model parameters and then, to solve these equations to retrieve estimators of the parameters of the model. In Part I of this thesis we will focus on both steps, providing and analyzing novel and improved model-specific moments estimators and techniques to solve the equations of the moments. In both the cases we will introduce theoretical results, providing guarantees on the behavior of the proposed methods, and we will perform experimental comparisons with existing algorithms. In Part II, we

will analyze the behavior of methods of moments when data violates some of the model assumptions performed by a user. First, we will observe that in this context most of the theoretical infrastructure underlying methods of moments is not valid anymore, and consequently we will develop a theoretical foundation to methods of moments in the misspecified setting, developing efficient methods, guaranteed to provide meaningful results even when some of the model assumptions are violated.

During all the thesis, we will apply the developed theoretical results to challenging real-world applications, focusing on two main domains: topic modeling and healthcare analytics. We will extend the existing theory of methods of moments to learn models that are traditionally used to do topic modeling – like the single-topic model and Latent Dirichlet Allocation – providing improved learning techniques and comparing them with existing methods, which we prove to outperform in terms of speed and learning accuracy. Furthermore, we will propose applications of latent variable models to the analysis of electronic healthcare records, which, similarly to text mining, are very likely to become massive datasets; we will propose a method to discover recurrent phenotypes in populations of patients and to cluster them in groups with similar clinical profiles – a task where the efficiency properties of methods of moments will constitute a competitive advantage over traditional approaches.

Acknowledgements

I have to confess that writing this thesis has been quite an amazing adventure, which contributed to make these years in Barcelona an unforgettable time. Without doubts, this has been possible thanks to the help and the contribution of many people, that I would like to acknowledge in this note.

My first and most special thank goes to my advisor Ricard, for his patience and flexibility, for the papers that we wrote together as coauthors and for introducing me to Computer Science back in 2015. I am grateful to my amazing coauthors Marta, Borja and Guillaume for sharing with me their ideas, being willing to listen to my proposals and helping me to become a better scientist with their help and advice; and to my Master thesis advisor Giacomo, with whom I shared back in 2011 my first steps in the path of scientific research.

Combining a PhD with a full-time job in the industry is not an easy task, and it would not have been possible without the support of ToolsGroup, who encouraged me with trust and flexibility during the first years of this adventure. To all the great people working there goes a special acknowledgement, and in particular to Yossi, Eugenio – for having been a role model – to Angela and David – for their interest and curiosity toward my studies and for always pointing me great ideas and new perspectives. Likewise, I want to thank all the people in Skyscanner for sharing with me the last part of this journey, and in particular my amazing team – the Albatross squad.

My lifelong friends in Melegnano deserve a special mention for being an invaluable support during such a hard-working time, and for sharing millions of adventures across Europe that made this PhD easier and more fun.

I want to acknowledge all the institutions who provided us clinical data allowing its use for our studies – the MIMIC project, the Hospital de Sant Pau and the Catalan Health Service (CatSalut) – and all the people who helped me on the analysis and interpretation of healthcare data carried on

during this thesis: Drs. Julianna Ribera, Salvador Benito, Mireia Puig, Montserrat Busting and Esther Limón. Completing this thesis would have been way harder without the contribution of the projects and the institutions that partially funded my PhD: the projects SGR2014-890 and 2017SGR-856 (MACDA) of AGAUR (Generalitat de Catalunya); the projects MDM-2014-0445 (BGSMath María de Maeztu Programme for Units of Excellence in R&D), TIN2014-57226-P (APCOM), and TIN2017-89244-R (MACDA) from MINECO (Ministerio de Economia, Industria y Competitividad).

I will be forever grateful to my family for always pushing me beyond my limits, to my mum Paola, my dad Marco and to my lovely grandparents who are a continuous source of inspiration.

Last, but foremost, I want to dedicate this thesis to my fiancée Francesca, for supporting me in this idea of starting a PhD, for her help in proofreading, editing and improving this thesis and for her infinite patience: I can not imagine how I could have completed this thesis without such an incredible partner.

Contents

Abbreviations

ALS Alternating Least Square

ASVTD Approximate Singular Value-based Tensor Decomposition

D-SIDIWO Data-centric Simultaneous Diagonalization based on Whitening and Optimization

EHR Electronic Healthcare Records

EM Expectation Maximization

LDA Latent Dirichlet Allocation

LVM Latent Variable Models

MAP Maximum A Posteriori

MCMC Markov-Chain Monte Carlo

MDL Minimum Description Length

MMD Maximum Mean Discrepancy

MoM Method of Moments

SIDIWO Simultaneous Diagonalization based on Whitening and Optimization

SVD Singular-Values Decomposition

SVTD Singular Value-based Tensor Decomposition

TPM Tensor Power Method

Introduction

Cerca una maglia rotta nella rete
che ci stringe, tu balza fuori, fuggi!
Va, per te l'ho pregato. Ora la sete
mi sarà lieve, meno acre la ruggine

E. Montale

In our conception of the world, we are naturally led to think that what we observe around us is the result and consequence of a series of hidden mechanisms that we can not perceive. As humans, we have the ambition of understanding and explaining those mechanisms, finding meaningful models that enable us to describe, explain and predict the hidden relationship between what can be observed and its latent causes.

Latent Variable Models (LVM) are a formidable tool to accomplish this need. Formally, a latent variable model is a probabilistic graphical model, designed as a set of hidden variables that influence the behavior of other observable random variables (called *features*). Both observable and latent variables are assumed to be random and to follow certain probability distribution; the influence that latent variables play on the observable features is generally described in terms of probabilistic dependence. Latent variable models can be applied as a tool in all the fundamental tasks of machine learning, like clustering – with mixture models – time series analysis – with Hidden Markov Models and their extensions – natural language processing – with topic models – and prediction.

When collecting data from reality, it is common to perform hypotheses on the intrinsic mechanisms governing their generation; these assumptions hypothesize that reality can not behave in totally arbitrary ways – a scenario where no prediction nor modeling would be possible – but follows some mechanisms, that make it, to certain extent interpretable and predictable. Latent

variable models are synthetic tools to describe these mechanisms. *Learning* a latent variable model means to find a model that properly describes the process that governs the generation of the data that we are observing. This task is commonly performed by algorithms that take as input a set of observed data, in terms of observable variables, and return an hypothesized model that properly describes them. Algorithms can come with different levels of complexity: the structure of the model may be known completely – leaving to the algorithm only the duty of estimating the parameters of a model – or partially, with the algorithm that also has to estimate part of the structure of the model.

Recent technology development has boosted the generation of massive datasets, characterized by thousands of observable features and millions of observations. Learning algorithms for latent variable models need to be able to deal with these kind of data, returning meaningful models in conveniently short running time. Providing algorithms to learn latent variable models is the objective of this thesis. In particular, our focus will be on learning techniques able to run efficiently on massive high dimensional datasets, maintaining the guarantee of returning provably good models. We will study algorithms for several latent variable models in various different scenarios, from the case when the structure of the model is assumed known, and the algorithm is required to retrieve only the parameters of the model, to the case when some information is missing, and the algorithm is required to retrieve a meaningful model with only partial information.

The standard approach to learn a parametric LVM is based on maximum likelihood principle, and consists of finding the model that maximizes the likelihood of the observed data. Given a dataset

$$\mathcal{X} = \{x^{(1)}, ..., x^{(n)}\}$$

the maximum likelihood approach first assumes data to be generated by a certain probability distribution $\mathbb{P}[X|\Theta]$ – depending on a set of unknown parameters Θ – and then it looks for the value of Θ maximizing the likelihood:

$$\Theta_{ML} = \operatorname*{argmax}_{\Theta} \mathbb{P}[\mathcal{X}|\Theta] = \operatorname*{argmax}_{\Theta} \prod_{i=1}^{n} \mathbb{P}[x^{(i)}|\Theta].$$

This approach is theoretically appealing, but it comes with several complications: the likelihood function is in general non-convex, and admits closed-form

solutions only for trivial models. During the years, several heuristics have been designed to approximatively maximize the likelihood. For many decades, the standard approach to accomplish this task has been the Expectation Maximization (EM) method (Dempster et al., 1977), an heuristic that is easy to implement, but is known to be prone to poor local optima. Furthermore, EM requires several passes across the entire dataset at each iteration, which makes it unusable with massive high-dimensional datasets (Balle et al., 2014). EM is not always applicable, as complex models do not allow an explicit calculation of the likelihood function; for these cases approximate inference approaches have been developed, which have similar limitations to those of EM. Some methods are based on Monte Carlo sampling, and are computationally demanding. Others are based on convex relaxations of the objective functions to be optimized, consequently returning suboptimal models.

To overcome these limitations, in recent years a new framework to learn latent variable models emerged: the Method of Moments (MoM). Instead of explicitly relying on the probability density function of a model, methods of moments work by finding equations that relate the low order moments of the observable data with the parameters of the latent variable model that one wants to learn, and solving these equations typically employing tensor/matrix factorization techniques. Compared to EM, moment-based algorithms provide a stronger theoretical foundation. In particular, it is known that in the setting where data is generated by a given model, whose characteristics are known to the user – a setting called the *realizable setting* – the output of a method of moments will converge to the true parameters of the model as the amount of training data increases. Furthermore, MoM algorithms only make a single pass over the training data, they are highly parallelizable and they always terminate in polynomial time. These promising characteristics make methods of moments the ideal learning framework for the challenging settings that we have in mind, as MoM allows to deal with massive datasets with guarantees of retrieving a meaningful model. The theory of methods of moments to learn latent variable models will be the core focus of this thesis.

The first part of this thesis studies methods of moments when data is assumed to be generated by a known latent variable model, whose parameters are to be estimated. This is the standard setting in which methods of moments are traditionally employed. Once a latent variable model has been designed, the learning procedure consists of two steps: first, finding the equations that

relate the moments of the observable data with the model parameters and second, to solve these equations to retrieve estimators of the parameters of the model. In this part of the thesis we will focus on both steps, providing and analyzing novel estimators for the moments and novel techniques to solve the equations that allow to retrieve, from the moments, the model parameters.

The second part of the thesis focuses on methods of moments when some of the information about the model generating the data is missing, or not properly specified by a user. In this context, we will observe that most of the theoretical infrastructure underlying methods of moments is not valid anymore – a fact that limits the applicability of methods of moments in scenarios where the model generating the data is unknown or misspecified, like it commonly happens with real-world data. The objective of the second part of this thesis will thus be to develop theoretical foundations to methods of moments in the misspecified setting, developing efficient methods, guaranteed to provide meaningful results even when some of the model assumptions are unknown or violated.

Despite the huge number of applications of latent variable models to real-world problems, the adoption of methods of moments is still limited, and practitioners still prefer to use more traditional approaches, like EM or approximate inference methods. Providing stable and accurate MoM algorithms and pioneering their usage in a real experimental framework are thus two strictly related research goals. For this reason, during all the thesis, we will apply the developed theoretical results to challenging real world applications, focusing on two main domains: text mining and healthcare analytics.

Text mining is a discipline of natural language processing that aims at finding latent topics in text corpora. Here, the observable data, generally coincides with the words appearing in a document, while the hidden variable can be, for example, the topics with which each document is dealing. Datasets in this discipline tend to be large – with several texts in it – high-dimensional – as the vocabularies contain thousands of words – sparse – especially for corpora with just short texts likes feeds, tweets or medical annotations. These characteristics, make the traditional approaches based on the likelihood principle difficult to apply efficiently, while they constitute the ideal application scenarios for methods of moments. During the thesis we will apply methods of moments to learn models that are traditionally used to do topic modeling, comparing them with more traditional approaches.

Healthcare analytics also constitute a natural application field for latent variable models: every time a patient visits an healthcare center, a record is produced, containing for example clinical findings and prescriptions. In this context, data represents the observable manifestation of a hidden status, representing the true, unobservable status of patient's diseases. Similarly to text mining, healthcare records are very likely to become massive datasets, with several dimensions associated to millions of patients, where the efficiency properties of methods of moments constitute a competitive advantage over traditional approaches. In this thesis, we will apply methods of moments to learn latent variable models able to provide a latent representation of patient statuses, allowing to discover recurrent phenotypes in populations of patients and to cluster patients in groups with similar clinical profiles.

Research Problems and Related Works

The seminal works on learning latent variable models with methods of moments date back to the years around 2010, when the machine learning community started realizing that a frequentist approach exploiting the moments could help solving problems that where known to be intractable if faced with traditional likelihood-based approaches. Generalizing an approach introduced by Pearson (1894), methods of moments first store in multidimensional tensors the low-order moments of the observable data, and then recover the parameters of the desired model via tensor decomposition techniques. The idea produced several early applications for Markov models (Chang, 1996), evolutionary trees (Mossel and Roch, 2005), mixture models (Feldman et al., 2006) and multi-view models (Chaudhuri et al., 2009), finding a more explicit formalization in the subsequent works by Hsu et al. (2012); Hsu and Kakade (2013); Anandkumar et al. (2012b) and Anandkumar et al. (2012a). Anandkumar et al. (2014) presented an exhaustive survey showing that spectral learning of most of the LVMs can be abstracted in two steps: first, given a prescribed LVM, they show how to operate with the data to obtain an empirical estimate of the moments, expressed in the form of symmetric low rank matrices and tensors; in a second step, the moments are decomposed using tensor/matrix decomposition techniques, to obtain the unknown parameters of the model. That paper also recalls and studies a method to derive such decomposition, the Tensor Power Method (TPM). This abstraction allows to split the research on methods of moments in two main non-disjoint areas.

A first area concentrates on methods to retrieve from data empirical estimates of the moments for specific LVMs. Examples are the works by Jain and Oh (2014) for mixture models with categorical observations, by Chaganty and Liang (2013) for mixtures of linear regressions, by Anandkumar et al. (2012b) for naive Bayes models, and by Hsu et al. (2012) for hidden Markov models. Important examples come from text mining, where equations to retrieve a moment representation for the single-topic model are provided by Anandkumar et al. (2012b) and Zou et al. (2013). An alternative and more complex model is Latent Dirichlet Allocation (LDA) (see Griffiths and Steyvers, 2004; Blei et al., 2003), for which Anandkumar et al. (2012a) provide empirical estimators of the moments.

A second research area focuses on methods to decompose the low-order moments to retrieve the parameters of the model. Literature on tensor decomposition is vast and it is believed to originate from the works by Hitchcock (1927, 1928). The topic received attention in the field of psychometrics (Carroll and Chang, 1970; Harshman, 1970; Tucker, 1966) and chemometrics (Appellof and Davidson, 1981) and extensive surveys are provided by Tomasi and Bro (2006); Kolda and Bader (2009) and Sidiropoulos et al. (2017). These works also present several classical techniques to obtain the so-called *canonical polyadic decomposition* (CPD) of a three dimensional tensor, which expresses it as a linear combination of rank-1 tensors. Kolda and Bader (2009) recall one of the most popular algorithms to obtain the CPD of a generic tensor: Alternating Least Square (ALS) (whose origins traces back to Carroll and Chang, 1970; Harshman, 1970). ALS is an heuristic that works by fixing all-but-one of the factors decomposing a tensor, and updating the remaining factor by solving an overdetermined linear least squares problem. ALS is easy to implement and to understand but is known to be prone to local minima, requiring several random restarts to return meaningful results (see the discussion by Kolda and Bader, 2009, pg. 18). A different line of work consists in approaching the tensor decomposition problem with simultaneous Schur decompositions (Van Der Veen and Paulraj, 1996; De Lathauwer et al., 2004; Colombo and Vlassis, 2016) or matrix optimization methods to perform simultaneous diagonalization (Kuleshov et al., 2015). In theory, one could use any of these methods to decompose the three-dimensional tensor containing the third-order moments, but the high dimensionality of that tensor makes this approach often computationally unfeasible. On the other hand, one

can rely on the specific structure of the whole set of low-order moments by using not only the third, but also the first and the second order moments to explicitly exploit their symmetry properties and obtain the desired model parameters more efficiently and with provable guarantees of global optimality. This is the approach followed by methods of moments.

The reference method here is the *Tensor Power Method* (TPM) from Anandkumar et al. (2014). TPM manipulates the low-order moments to obtain a symmetric orthogonal tensor that is decomposed using a high-dimensional extension of the well-known matrix power method, with an approach proposed by Zhang and Golub (2001) and Kolda (2001). The main issue of this algorithm is its scalability, since its running time grows as k^5 where k is the number of latent components. An alternative, which in general has better dependence on the number of latent factors, consists in using matrix-based techniques and a *simultaneous diagonalization* approach, specializing an approach that can be traced back to Sanchez and Kowalski (1990) and Leurgans et al. (1993). Examples of these methods of moments are provided by Anandkumar et al. (2012b), with an algorithm based on the eigenvectors of a linear operator; Anandkumar et al. (2012a) also present an alternative method based on the singular vectors of a Singular-Values Decomposition (SVD). These two methods are faster than TPM by far, but they are more sensitive to perturbations on the input data.

A method of moments uses a set of n samples to empirically estimate the moments and then uses tensor decomposition methods to recover the model parameters from the decompositions of the estimated, perturbed, moments. Methods of moments can thus be improved by *improving the convergence rate of the estimators* – obtaining better moments with smaller sample sizes – by *working on the decomposition algorithms to reduce their sensitivity to input perturbations* – returning better model parameters when the input moments are perturbed – and *improving their performance* – performing the decompositions in shorter running times. In this thesis, we will focus on all these aspects, working on novel moment-estimators for several latent variable models and providing improved tensor decomposition techniques that allow to recover, from an estimation of the moments, robust estimates of the parameters of a latent variable model in short running times.

As explained in the introduction, the traditional approach to learn latent

variable models is based on the maximum-likelihood principle and aims at finding the model that maximizes the likelihood of an observed dataset. However, maximizing the likelihood is a difficult task even for simple models: Arora et al. (2012) for example proved that maximum likelihood estimates for the single topic model are NP-hard, while Roch (2006) proved the same for latent trees. Consequently, several heuristics have been developed aiming at solving the maximum likelihood problem under an approximate point of view. Expectation Maximization (Dempster et al., 1977), for example is an iterative technique that, starting from a user defined initialization (typically random), iteratively updates the parameter of a model, increasing the likelihood at each iteration. EM has only local guarantees of convergence, but is easy to understand and to implement, which makes it the most used technique in this field. Furthermore, EM requires the complete calculation of the posterior distributions, which may not be feasible for models with a complex structure; in this case it is common to use approximate inference approaches based on variational inference or Monte Carlo-based approaches (see Bishop 2006 for a detailed presentation of these techniques). Approximate inference techniques are widely used in tasks like topic modeling (Blei et al., 2003; Griffiths and Steyvers, 2004) and mixture-models learning (Marin et al., 2005). Thanks to their flexibility, they allow to deal with complex models, but, unlike methods of moments, they fail to provide optimal models in small running times.

Methods of moments and likelihood-based techniques aim at the same purpose – retrieving the parameters of a model – following two different frameworks. However, these two frameworks are not unrelated, but complementary. For example, empirical studies indicate that initializing EM with the output of a MoM algorithm can improve the convergence speed of EM by several orders of magnitude, yielding a very efficient strategy to accurately learn latent variable models (Balle et al., 2014; Chaganty and Liang, 2013; Bailly, 2011). In the case of relatively simple models this approach can be backed by intricate theoretical analyses (Zhang et al., 2014), showing that initializing EM with the output of a method of moments can provide guarantees of global optimality.

Despite their theoretical appeal, the practical applications of methods of moments are still limited, and EM remains the tool of choice of most of practitioners, notwithstanding the slow running times and the well-known proneness to poor local optima. While this is partially due to the novelty of

the MoM framework, there are also deeper theoretical issues that limit the robustness of these methods in real-world applications. Methods of moments in fact, to work with guarantees, require the user to know the exact structure of the latent variable model that he wants to learn; this means knowing the number of latent variables of a model, their distribution, and their probabilistic relation with observable data. However, most of these characteristics, are by definition unknown to the user, who can only estimate them with cross-validation techniques. Kulesza et al. (2014) demonstrated that when the number of latent states imposed by the user is not enough to describe the training data, a MoM can lead to unexpected results, and provided in a subsequent work (Kulesza et al., 2015) potential ways to overcome this issue. In general, it is possible to demonstrate (see Chapter 5) that when the characteristics of the model generating the data are unknown or misspecified, most of the traditional methods of moments provide no guarantees on their results. Furthermore, for methods of moments to work properly, it is crucial to deal with moments whose expectations are in a prescribed relation with respect to the parameters of the model one wants to learn; however, efficient formulations of the moments are known only for few, simple, latent variable models, requiring heavy assumptions on the data, that are systematically violated in real world scenarios. The problem of the applicability of methods of moments in settings where the model generating the data is unknown or not properly specified is largely unstudied; in this context, the traditional approach consists of refining the results of a MoM with EM, so to obtain a parameter that, at least locally, is guaranteed to be meaningful, locally maximizing the likelihood. While this approach is effective in practice, a theory on why in the misspecified setting a MoM would return a meaningful model (and thus a good initializer for EM) is still not present. *Providing a theoretical analysis of methods of moments in the misspecified setting* is one of the objectives of this thesis.

Overview of contributions

The first part of this thesis (Chapters 1 to 4) is dedicated to the theory of methods of moments for learning latent variable models when all the characteristics of the model generating the data are known, and only the parameters of that model need to be recovered.

Chapter 1 is dedicated to recalling the basis of methods of moments. We will present how methods of moments work by presenting a simple application for learning Naive Bayes Models with Poisson-distributed features. We will show how it is possible to recover a set of equations that relate the low-order moments of the observable data with the parameters of the model and we will then present the most used techniques to solve these equations – namely the tensor decomposition methods from Anandkumar et al. (2012b, 2014), and Alternate Least Square (Kolda and Bader, 2009). We will conclude the chapter with a brief discussion on how this approach can be generalized to any latent variable model, and on the conditions required to guarantee the identifiability of a model in this setting.

A fundamental part in methods of moments is the step that decomposes the matrices/tensors containing the moments to retrieve an estimate of the parameters of the model generating the data. This step is generally accomplished using algorithms that employ matrix/tensor decomposition methods, which ideally are required to run in short running times and to guarantee good robustness to random perturbations – providing good learning accuracy when only a limited amount of data is available. A scan of the literature for these methods present some alternative tools of choice: Anandkumar et al. (2012a,b) present methods with high scalability, but poor robustness to perturbations, while Anandkumar et al. (2014) introduce a method with better robustness but significantly worse running times. At the same time, existing methods from tensor factorization theory (see for example Tomasi and Bro 2006; Kolda and Bader 2009; Sidiropoulos et al. 2017) provide heuristics with a worse computational complexity and less guarantees on the learning accuracy. These limitations of existing methods led us to search for an algorithm with high learning accuracy and strong efficiency guarantees. In Chapter 2 we provide a novel algorithm for the simultaneous solution of moment equations that allows to retrieve, from estimates of the moments, a set of estimates of the model parameters. This algorithm will rely on simple matrix operations – being consequently fast – and will guarantee a best-in-class learning accuracy. We will analyze the proposed method under the theoretical point of view, studying its computational complexity and its robustness to random perturbations of the input. Finally we will experimentally compare the proposed method with existing techniques, showing that the proposed algorithm outperforms each existing method either in terms of speed or in learning accuracy.

One of the areas where latent variable models find a natural field of application is natural language processing and topic modeling. In this field, the observable variables are the words appearing in a text, while the latent variables are the topics with which a text is dealing. A simple model is the *single-topic model*, where each text is assumed to be about only one topic and the probability of a given word appearing in a text depends on the topic of the text. An alternative and more complex model is Latent Dirichlet Allocation (LDA) (see Griffiths and Steyvers, 2004; Blei et al., 2003), where each text is assumed to deal with a multitude of topics, and each word of a text is assumed to be associated to a unique topic. Learning techniques for topic modeling are in general dominated by Bayesian approaches, with techniques based on Markov Chain Monte Carlo methods (Griffiths and Steyvers, 2004) and variational techniques (Blei et al., 2003). At the same time, methods of moments exist to learn standard models like the single-topic model (Anandkumar et al., 2012b; Zou et al., 2013) and LDA (Anandkumar et al., 2012a). In Chapter 3 we present novel methods of moments to learn these two models aiming at improving existing ones. In particular we will introduce new of estimators of the moments for the single-topic model and for LDA, studying their sample complexity and providing novel sample complexity bounds. We will show, both experimentally and with a theoretical analysis, that the proposed estimators have an improved robustness in comparison with existing ones. Furthermore we will compare the proposed approaches with the standard Bayesian approach proposed by Griffiths and Steyvers (2004) on real-world text corpora, both for single-topic model and LDA. We will show that the moment-based method presented in this chapter recovers higher-quality topics in much smaller running time.

In Chapter 4 we will present an application of latent variable models to healthcare analytics, focusing on the specific task of patient clustering. We will consider the most simple kind of electronic healthcare records – namely ICD9 records (Geraci et al., 1997) registering patient diagnostics – and we will use these data with two objectives: clustering patients in groups with homogeneous clinical profiles and retrieving meaningful phenotypes – i.e. abstract representations describing the most recurrent diseases-patterns in patient statuses. To accomplish this task we will model our data as mixture models, learn the mixture with a method of moments and use that mixture to cluster the various patients, using a model-based clustering approach (Marin et al., 2005). The choice of a model-based approach over the more traditional

distance-based methods like *k-means* (Macqueen, 1967) or *spectral clustering* (Ng et al., 2002) has two benefits: first it allows to deal with high-dimensional sparse data, as it will be the case for our dataset, and second it provides a meaningful generative model for our dataset, allowing an easy construction of phenotypes. The task of performing a moment-based learning of a mixture model will come with a peculiar challenge, because no efficient moment-estimators are known for such kind of latent variable models. As a consequence, we will propose an approximate approach where we will develop moment-estimators that are provably almost-unbiased – i.e. with a bias that can be calculated and is provably small – and feed a decomposition algorithm with these estimators. The resulting parameters will then be refined with EM, in order to remove the bias inherited from the moment estimation. We will see that this hybrid MoM-EM approach will work effectively, with EM rapidly converging to good optima. Furthermore, it will allow to present an efficient implementation of the decomposition method presented in Chapter 2, allowing it to run with a better computational complexity. We will apply this approach to the considered dataset of patients obtaining clusters and phenotypes that make sense under the clinical point of view. Furthermore, we will compare on synthetic data our approach with more traditional clustering algorithms, outperforming them in terms of clustering accuracy.

In all the chapters above we have focused on scenarios where data was generated by a latent variable model whose structure was assumed to be known to the user, and the task of a learning algorithm was reduced solely to recovering the parameters of the model. Methods of moments require the user to explicitly know the structure of the latent variable model that he wants to learn, as this information is explicitly used in two points: one is when the moment are calculated, as the formulation of the moment estimators depend on the specific model generating the data; different model will have different estimators. The second is in the decomposition algorithm, where a user needs to ask for a specific number of latent states to be returned by the algorithm. All the theory of methods of moments has been developed in the *realizable* setting where the model characteristics are known to the user, and data does not violate any model assumption. This approach allowed to build a clear and round theoretical framework for methods of moments, but also limited the theoretical reliability of these methods in real-world scenarios, where the model generating the data is always unknown, and model assumptions are sure to be violated. In the second part of this thesis (Chapters 5 and 6), we

will concentrate on analyzing methods of moments when data does not follow a model of the mathematical form assumed by the algorithm, violating some of the assumptions that the user makes.

In Chapter 5 we will focus on the scenario where a method of moments is required to learn a latent variable model from data, but the number of latent states required by a user is too small to accurately represent the training data. This is a very common scenario, in particular when data is high-dimensional and it is difficult to find a number of latent states to comprehensively describe the dataset, or this number is too high to be estimated. For example, an important application of low-dimensional learning comes from exploratory data analysis, where a mixture model with two states is required to bisect a dataset into two well-distinct classes. The desired behavior of a learning technique when run in a low-dimensional setting, is to return a small model that synthetically describes the data, providing the optimal low-dimensional model approximating the data we are observing. In Chapter 5 we will demonstrate that this is not the behavior of existing methods of moments, which instead are likely to return unexpected results when plugged with a misspecified number or latent states. As a consequence, we provide a novel decomposition algorithm for method of moment, that phrases the decomposition task as a non-convex optimization problem and generalizes the method presented in Chapter 2. We demonstrate that the proposed algorithm, when run in a low-dimensional setting, returns the optimal low-dimensional model approximating the one generating the data, according to an intuitive definition of optimality. Starting from these remarks, we apply this method to hierarchically learn latent variable models, starting with a simple, two-dimensional model, which is then refined iterating the learning step on each of the retrieved dimensions. The hierarchical nature of this method allows for a fast and accurate solution of the optimization problem raising in the decomposition task, based on low-dimensional grid search. An immediate application of this approach is to perform hierarchical clustering, where a mixture with two classes is learned, used to bisect our dataset, and then the procedure is iterated on each of the two retrieved clusters. In this chapter we will also present an application of this approach to natural language processing, providing a specialization of our method to perform hierarchical topic modeling.

All the guarantees provided by the theoretical analysis of methods of moments

come with the assumption that a user is able to calculate unbiased estimators of the moments, that asymptotically exhibit a prescribed relation with the parameters of the model. However, this assumption is possible only when data is accurately described by a latent variable model of known structure, while when the structure of the model is not known or no finite model accurately describes the data – as it is the case in real-world scenarios – nothing is known on the asymptotic behavior of the moment equations. Consequently, the theoretical infrastructure guaranteeing that methods of moments will return a good model approximating the data, loses most its strength in real-world contexts. In Chapter 6 we demonstrate that theoretical guarantees on the behavior of methods of moments can be retrieved also when no hypotheses on the model generating the data are made. In particular, we will consider a simple calculation formula for the moments, and we will demonstrate that the decomposition technique introduced in Chapter 5 provides a meaningful relation between the input data and the output model, regardless the model generating the data. This means that its output will accurately describe the data, even if the user imposes no hypotheses on the input data. Additionally, we will analyze the cost function of the considered method and its action on the input data, observing that it automatically suggests a rule to split them into meaningful clusters. From this observation, we will introduce a novel model-independent clustering algorithm that will be particularly suitable for high-dimensional binary data. We will thus apply this approach to the medical records studied in Chapter 4, obtaining meaningful hierarchical phenotypes and clinically reasonable trees of clusters.

The last chapter concludes this work and highlights the main research challenges emerging from the problems faced in this thesis. Some of them are theoretical and aim at improving our current understanding of methods of moments, studying the relations that link this family of techniques with traditional likelihood-based methods. Some others are applied, and focus on extending the applicability of methods of moments to real-world scenarios.

Publications derived from the thesis

The following papers have been written as an outcome of this thesis.

- Ruffini, M. (2018). Hierarchical Methods of Moments for Clustering High Dimensional Binary Data. *Submitted*

- Ruffini, M., Casanellas, M., and Gavaldà, R. (2018). A new method of moments for latent variable models. *Machine Learning*, 107(8-10):1431–1455[1]

- Ruffini, M., Rabusseau, G., and Balle, B. (2017b). Hierarchical methods of moments. In *Neural Information Processing Systems*, pages 1901–1911 [2]

- Ruffini, M., Gavalda, R., and Limon, E. (2017a). Clustering patients with tensor decomposition. In *Machine Learning for Healthcare Conference*, pages 126–146 [3]

[1] `https://mruffini.github.io/assets/papers/method-moments-LVM-preprint.pdf`

[2] `http://papers.nips.cc/paper/6786-hierarchical-methods-of-moments.pdf`

[3] `http://proceedings.mlr.press/v68/ruffini17a/ruffini17a.pdf`

Part I

Methods of Moments for Learning Latent Variable Models

Chapter 1

Background: Methods of Moments

In this chapter we present the basis of Method of Moments. We begin by introducing a simple family of latent variable models called naive Bayes models and show how to learn them with methods of moments, presenting an overview of the state-of-the-art approaches. We will then show how the proposed technique can be extended to several other models.

1.1 Naive Bayes Models

A naive Bayes model is a latent variable model characterized by $d+1$ random variables $(Y, X_1, \ldots, X_d)$ with the following characteristics:

- Y is a hidden (unobservable) discrete variable with a finite number of possible outcomes: $Y \in \{1, \ldots, k\}$. We define

$$\omega_j = \mathbb{P}(Y = j), \quad \omega = (\omega_1, \ldots, \omega_k) \in \Delta^{k-1},$$

  where Δ^{k-1} is the $k-$dimensional simplex:

$$\Delta^{k-1} = \{(\pi_1, ..., \pi_k) \in \mathbb{R}^k : \sum \pi_i = 1\}.$$

  The entries of the vector ω are commonly named the *mixing weights* of the model.

- The vector $X = (X_1, \ldots, X_d)$ is observable, its distribution depends on the value of the hidden variable Y and the random variables $X_1, \ldots, X_d$ are conditionally independent given Y. It is common to call the entries of X the *features* of the model, and their conditional probability density functions may follow any parametric distribution:

$$Distr(X|Y = j) \approx \mathbb{P}_j[X], \quad Distr(X_i|Y = j) \approx \mathbb{P}_{j,i}[X_i].$$

  A special role is played by the conditional expectations of the features:

$$\mu_j = \mathbb{E}[X|Y = j] \in \mathbb{R}^d, \quad \mu_{j,i} = \mathbb{E}[X_i|Y = j]$$

$$M = [\mu_1|, ..., |\mu_k] \in \mathbb{R}^{d\times k}.$$

  The vectors $\mu_1, ..., \mu_k$ are commonly called *centers* of the model.

Remark 1.1.1. The value of d typically represents the amount of observable information, while the value k represents the number of latent states that are enough to provide a synthetic description of the data generated by the model. Even if d and k can take any value, during this thesis we will be mostly interested in the case of high-dimensional models, where $d \geq k$.

The generative process determined by a naive Bayes model, first generates an unobservable outcome for Y, say j, and then generates a set of observations $X_1, \ldots, X_d$, whose distribution depends on the value of Y. The conditional independence of the features allows the following simple factorization of the probability density function:

$$\mathbb{P}[X_1, \ldots, X_d] = \sum_{j=1}^{k} \omega_j \mathbb{P}[X_1, \ldots, X_d|Y = j] = \sum_{j=1}^{k} \omega_j \prod_{i=1}^{d} \mathbb{P}_{j,i}[X_i].$$

Naive Bayes models are a family of latent variable models, and their specific distribution depends on the special characteristics of the conditional distributions of the features, namely the probability density function $\mathbb{P}_{j,i}[\,\cdot\,]$.

1.1.1 Learning Naive Bayes Models

In the most generic scenario, each $\mathbb{P}_{j,i}$ is a specific probability density function following a certain probability distribution and depending on a certain set of parameters $\Theta_{i,j}$:

$$\mathbb{P}_{j,i}[\,\cdot\,] = \mathbb{P}_{j,i}[\,\cdot\,|\Theta_{i,j}]$$

The knowledge of the distributional properties of $\mathbb{P}_{j,i}$, together with the number of latent states k, constitute the probabilistic structure of the model, and we will abstract it with the notation $\mathbb{S}$. In general, this information will not be observable from a dataset, and constitutes the model assumption that a user does on the data that he is observing.

The union of all the parameters $\Theta_{i,j}$ characterizing each distribution $\mathbb{P}_{j,i}$, together with the mixing weights ω, constitute the set of parameters of the model, and it will be denoted with the pair (Θ, ω), where $\Theta = \{\Theta_{j,i}\}_{j,i}$. Also the parameters of the model will not be directly observable from data, but unlike the model structure, they will not be imposed by the user, as they will be the object of a learning process. Consider for example a dataset $\mathcal{X}$, and impose then the model assumption that data is generated by a naive Bayes model with a certain structure $\mathbb{S}$. The objective of a learning task will thus be to find the model parameters (Θ, ω) that allow a model with structure $\mathbb{S}$ to best describe the data. A learning algorithm for a prescribed latent variable model with structure $\mathbb{S}$ will thus be an algorithm that, when inputted with a dataset $\mathcal{X}$, it returns a meaningful approximation of the parameters:

$$\mathcal{A} : \ (\mathcal{X}, \mathbb{S}) \to (\tilde{\Theta}, \tilde{\omega}).$$

The definition of meaningfulness that one will adopt will determine how a learning algorithm will operate, and we will see an example in the proceeding of this chapter.

1.1.2 Naive Bayes Models and Clustering

The main application of naive Bayes models is clustering. Consider a dataset

$$\mathcal{X} = \{x^{(1)}, ..., x^{(n)}\},$$

and assume it to be generated by a naive Bayes model with k states. This means that we assume to have k probability distributions $\mathbb{P}_1, .., \mathbb{P}_k$ and each observation $x^{(i)}$ is assumed to be sampled by one of these distributions; the mixing weights ω determine the asymptotic frequencies of the distributions in the dataset. This suggests an intuitive approach to clustering, by assigning each sample to the distribution that has the highest likelihood of having generated it, which allows to partition the considered dataset into k classes.

Assume to have learned from data a naive Bayes model with a known structure. We thus have k distributions $\mathbb{P}_1, .., \mathbb{P}_k$ that we can use to partition our dataset. We want to assign each sample to its most likely latent state, where this probability is defined as follows:

$$\mathbb{P}[Y = j|X = x] \approx \omega_j \mathbb{P}[X = x|Y = j].$$

This implies

$$Cluster(x) = \underset{j}{\text{argmax}}\ \omega_j \mathbb{P}_j[x].$$

This approach to clustering is commonly called model-based clustering, as it is grounded on a model assumption on the data. Furthermore, this assumption exploits the model structure to provide a synthetic description of the data we are observing, being an interpretable and probabilistically robust approach.

1.1.3 Example: Poisson's Naive Bayes Models

In this section we will analyze a special case of naive Bayes model, where all the features are conditionally distributed as Poisson random variables.

Poisson distribution is an integer-valued probability distribution with the following density function

$$\mathbb{P}[x] = \frac{\mu^x}{x!} e^{-\mu}, \ \ x \in \{0, 1, 2.., \infty\}.$$

A Poisson distribution only depends on the parameter $\mu > 0$, which coincides with the expectation and the variance of the distribution:

$$\mathbb{E}[X] = \mu, \ \ Var[X] = \mu \ \ \text{if} \ X \approx Poiss(\mu).$$

A Poisson's naive Bayes model is a naive Bayes model where all the features are conditionally distributed as Poisson random variables. The distribution of the jth feature conditioned to state i will thus be

$$\mathbb{P}_{j,i}[\,\cdot\,] \approx Poiss(\mu_{j,i})$$

for a certain parameter $\mu_{j,i}$. It is immediate to observe that if $X = (X_1, ..., X_d)$ is the vector of features of the considered model, then

$$\mathbb{E}[X_i|Y = j] = \mu_{j,i}.$$

The vector $\mu_j = (\mu_{j,1}, ..., \mu_{j,d}) \in \mathbb{R}^d$ will thus coincide with the jth center of the model: $\mu_j = \mathbb{E}[X|Y = j]$. For a Poisson's naive Bayes model the model structure is thus simple:

$$\mathbb{S} = \begin{cases} \mathbb{P}_{j,i} \approx Poiss(\cdot) \\ k \text{ latent states} \end{cases}$$

and the parameters characterizing each distribution will coincide with the centers of the model:

$$\mu_{j,i} = \Theta_{j,i}.$$

Poisson's naive Bayes models are thus entirely determined by their mixing weights ω and their matrix of the centers M, which are their only parameters: $(M, \omega) = (\Theta, \omega)$.[1] From now on, we will call the pair (M, ω) the parameters of a Poisson's naive Bayes model.

Considering a dataset $\mathcal{X}$ generated by a Poisson's naive Bayes model with parameters (M, ω), we want an algorithm able to return an estimate $(\tilde{M}, \tilde{\omega})$ of its parameters.

1.2 The Method of Moments

In this section we present a framework to learn latent variable models, called method of moments. We will use as instrumental example Poisson's naive Bayes models, clarifying how the approach can be easily generalized to several well-known latent variable models.

The idea at the basis of the method of moments is an immediate consequence of the law of large numbers, which we recall here:

Theorem 1.2.1. *Let $\{x^{(1)}, ..., x^{(n)}\}$ be an iid sample generated from a distribution X such that $\mathbb{E}[X] = \mu$. Then the following relation holds:*

$$\lim_{n \to \infty} \sum_{i=1}^{n} \frac{x^{(i)}}{n} = \mu.$$

[1] We remark that this is the case for any naive Bayes model whose features conditional distributions $\mathbb{P}_{j,i}$ depend on a unique parameter $\mu_{j,i}$; for example, this is the case for Bernoulli or Exponential random variables.

The consequence of this theorem is that we can calculate for a distribution a good estimator of its expectation, by just averaging out a large-enough set of observed samples. Furthermore, the estimator is guaranteed to approximate the parameters μ with increasing accuracy while the sample size increases. If for example, the samples are generated by a Poisson distribution, and if we have enough data, the sample average will be a good estimator for the parameter μ, which is the sole parameter of the considered distribution. The sample average $\sum x^{(i)}/n$ is commonly called *first-order sample moment*, and it is an approximation of the theoretical first-order moment, that is the expectation μ of the random variable; the usage of the moments of a distribution for calculating its parameters is commonly called *method of moments.*

While the Poisson distribution example is a trivial application of the method of moments, it provides a possible learning strategy for more complex latent variable models. We will now see how to extend this approach to naive Bayes models, focusing on the special case of Poisson's naive Bayes model. Consider a dataset generated by a Poisson's naive Bayes model with parameters (M, ω) and k latent states:

$$\mathcal{X} = \{x^{(1)}, ..., x^{(n)}\},$$

where each observation is a d dimensional vector. Our goal is to provide a method to recover an estimate $(\tilde{M}, \tilde{\omega})$ of the model parameters. Trying to repeat the steps used to estimate the parameters of a Poisson distribution, we begin by calculating the first order moment of the observed sample, which will be called $\tilde{M}_1$:

$$\tilde{M}_1 = \sum_{i=1}^{n} \frac{x^{(i)}}{n} \in \mathbb{R}^d.$$

It is easy to observe that the expectation of $\tilde{M}_1$ has a nice form:

$$\mathbb{E}[\tilde{M}_1] = M_1 = \sum_{i=1}^{k} \omega_i \mu_i, \tag{1.1}$$

which suggests to solve the system of d equations that arises by substituting M_1 with its estimator in equation (1.1), to obtain asymptotically converging estimators of the model parameters. However, even in the theoretical scenario where we have infinite data and $\tilde{M}_1 = M_1$, the system $M_1 = \sum_{i=1}^{k} \omega_i \mu_i$ has, in most of the cases, several solutions: we have $k \times d + (k - 1)$ parameters and just d equations, which are in general not enough to identify the

model we want to learn. To see a simple example, consider the case where $d > k$ and assume that (M, ω) is a solution of the system (1.1); then also $(MO, O^\top \omega)$ will be a solution of the same system for any orthogonal matrix O.

A possible learning strategy may thus consist in providing additional constraints to system (1.1), by including also the second and the third order moments. These constraints will be provided by the following proposition:

Proposition 1.2.1. *Define the following estimators, called respectively the second and the third order sample moments :*

$$\tilde{M}_2 = \sum_{i=1}^{n} \frac{x^{(i)} \otimes x^{(i)}}{n} - \mathrm{diag}(\tilde{M}_1) \in \mathbb{R}^{d \times d}$$
$$\tilde{M}_3 = \sum_{i=1}^{n} \frac{x^{(i)} \otimes x^{(i)} \otimes x^{(i)}}{n} - \tilde{M}_{3,2} \in \mathbb{R}^{d \times d \times d}$$

where

$$\begin{aligned}(\tilde{M}_{3,2})_{h,l,m} =& \chi_{h=l=m}((\tilde{M}_1)_h + 3(\tilde{M}_2)_{h,h}) \\ &+ \chi_{(h\neq l=m)\vee(l\neq h=m)}(\tilde{M}_2)_{h,l} + \chi_{h=l\neq m}(\tilde{M}_2)_{h,m},\end{aligned}$$

χ is the indicator function and $\otimes$ represents the Kronecker product between vectors. Then we have, for a Poisson naive Bayes model

$$\mathbb{E}[\tilde{M}_2] = M_2 = \sum_{i=1}^{k} \omega_i \mu_i \otimes \mu_i, \tag{1.2}$$

$$\mathbb{E}[\tilde{M}_3] = M_3 = \sum_{i=1}^{k} \omega_i \mu_i \otimes \mu_i \otimes \mu_i. \tag{1.3}$$

Proof. We only provide the proof for the second order moment, that for M_3 being conceptually identical. Consider first the off-diagonal entries of our

estimator, for certain $l \neq j$:

$$\begin{aligned}
\mathbb{E}[(\tilde{M}_2)_{l,j}] =& \mathbb{E}[\sum_{i=1}^{n} \frac{(x^{(i)})_l (x^{(i)})_j}{n}] \\
=& \mathbb{E}[(X)_l (X)_j] \\
=& \sum_{h=1}^{k} \omega_h \mathbb{E}[(X)_l (X)_j | Y = h] \\
=& \sum_{h=1}^{k} \omega_h \mathbb{E}[(X)_l | Y = h] \mathbb{E}[(X)_j | Y = h] \\
=& \sum_{h=1}^{k} \omega_h \mu_{h,l} \mu_{h,j}.
\end{aligned}$$

The proof for the diagonal terms works similarly:

$$\begin{aligned}
\mathbb{E}[(\tilde{M}_2)_{l,l}] =& \mathbb{E}[(X)_l^2] - \mathbb{E}[(X)_l] \\
=& \sum_{h=1}^{k} \omega_h \mathbb{E}[(X)_l^2 | Y = h] - \sum_{h=1}^{k} \omega_h \mu_{h,l} \\
=& \sum_{h=1}^{k} \omega_h (\mu_{h,l} + \mu_{h,l}^2) - \sum_{h=1}^{k} \omega_h \mu_{h,l} \\
=& \sum_{h=1}^{k} \omega_h \mu_{h,l}^2
\end{aligned}$$

where we have used the fact that for a Poisson distribution X with expectation μ, it holds that $\mathbb{E}[X^2] = \mu + \mu^2$. □

The estimators for the second and the third order moments provided above, allow to add $d^2 + d^3$ equations to system (1.1), obtaining the following set of equations:

$$\begin{cases}
M_1 = \sum_{i=1}^{k} \omega_i \mu_i, \\
M_2 = \sum_{i=1}^{k} \omega_i \mu_i \otimes \mu_i, \\
M_2 = \sum_{i=1}^{k} \omega_i \mu_i \otimes \mu_i \otimes \mu_i.
\end{cases} \tag{1.4}$$

Unlike system (1.1), system (1.4) admits a unique set of solutions if $d \geq k$ and if M is a full-rank matrix, which is a quite generic constraint. In fact,

Comon et al. (2017) guarantee that if equation (1.3) has a unique solution and system (1.4) has at least one solution, then also the whole system (1.4) admits the same unique solution. Generic uniqueness of the real solutions of (1.3) is guaranteed by known results on tensor decomposition (Chiantini et al., 2017; Qi et al., 2016), and can be verified using Kruskal's (Kruskal, 1977) criterion (see also Kolda and Bader, 2009, Sec. 3.2).

The fact that the system (1.4) has a unique solution, suggests a possible learning strategy for Poisson's naive Bayes models:

1. Calculate estimators for the low-order moments of the model, namely $\tilde{M}_1$, $\tilde{M}_2$ and $\tilde{M}_3$, using equation (1.1) and Proposition 1.2.1.
2. Plug the estimated moments in the left terms of the system (1.4) and find a set of parameters $(\tilde{M}, \tilde{\omega})$ that approximately solve that system:

$$\begin{cases} \tilde{M}_1 \approx \sum_{i=1}^{k} \tilde{\omega}_i \tilde{\mu}_i, \\ \tilde{M}_2 \approx \sum_{i=1}^{k} \tilde{\omega}_i \tilde{\mu}_i \otimes \tilde{\mu}_i \\ \tilde{M}_2 \approx \sum_{i=1}^{k} \tilde{\omega}_i \tilde{\mu}_i \otimes \tilde{\mu}_i \otimes \tilde{\mu}_i. \end{cases} \tag{1.5}$$

Notice that the approximations symbols in system (1.5) are required, because the existence and the uniqueness of a solution is only guaranteed in the theoretical scenario where we have infinite data. The idea of this approach is that while the sample size increases, the estimators of moments get closer to their theoretical values, and, as a consequence, the parameters $(\tilde{M}, \tilde{\omega})$ will asymptotically approach their theoretical homologous (M, ω).

It is important to remark that the usage of the third order moment is necessary to guarantee the uniqueness of solution for system (1.4), for which it is not sufficient the second order moment, as the following theorem shows.

Theorem 1.2.2. *Let (M, ω) a pair of parameters solving the following system of equations:*

$$\begin{cases} M_1 = \sum_{i=1}^{k} \omega_i \mu_i, \\ M_2 = \sum_{i=1}^{k} \omega_i \mu_i \otimes \mu_i. \end{cases} \tag{1.6}$$

Then, also the pair $(\hat{M}, \hat{\omega})$, where

$$\hat{M} = M \Omega^{1/2} O \operatorname{diag}(O^\top \omega^{1/2})^{-1} \tag{1.7}$$

$$\hat{\omega} = (O^\top \omega^{1/2}) \circ (O^\top \omega^{1/2}), \tag{1.8}$$

is a solution of the same system, for any orthogonal matrix O, where $\circ$ is the point-wise (Hadamard) product between vectors.[2]

Proof. We can rewrite the system (1.6) as follows

$$\begin{cases} M_1 = M\omega \\ M_2 = M\Omega M^\top \end{cases}$$

Our thesis is proved if we demonstrate that

$$\begin{cases} \hat{M}\hat{\omega} = M\omega \\ \hat{M}\hat{\Omega}\hat{M}^\top = M\Omega M^\top \end{cases}$$

where $\hat{\Omega} = \operatorname{diag}(\hat{\omega})$. We begin with the first equality:

$$\begin{aligned} \hat{M}\hat{\omega} &= M\Omega^{1/2} O \operatorname{diag}(O^\top \omega^{1/2})^{-1} (O^\top \omega^{1/2}) \circ (O^\top \omega^{1/2}) \\ &= M\Omega^{1/2} O O^\top \omega^{1/2} \\ &= M\omega. \end{aligned}$$

The second equality works similarly. □

The only remaining challenge is thus to find a technique able to solve system (1.5). Concretely, we will need an algorithm $\mathcal{A}$ able to recover exactly (M, ω) from the unperturbed system (1.4):

$$\mathcal{A} : (M_1, M_2, M_3, k) \to (M, \omega)$$

Furthermore, when the exact moments M_1, M_2, M_3 are not available, and only their estimators $\tilde{M}_1, \tilde{M}_2, \tilde{M}_3$ are accessible to the user, $\mathcal{A}$ needs to return approximate pairs $(\tilde{M}, \tilde{\omega})$ that get closer to (M, ω) while $\tilde{M}_1, \tilde{M}_2, \tilde{M}_3$ get closer to M_1, M_2, M_3. In the next section we present the most popular approaches to accomplish this task.

1.2.1 Tensor Decomposition

In this section we concentrate on the task of solving the system

$$\begin{cases} M_1 = \sum_{i=1}^k \omega_i \mu_i, \\ M_2 = \sum_{i=1}^k \omega_i \mu_i \otimes \mu_i, \\ M_3 = \sum_{i=1}^k \omega_i \mu_i \otimes \mu_i \otimes \mu_i \end{cases}$$

[2] We have used the notation $\Omega = \operatorname{diag}(\omega)$.

where only the left terms of the equations are known. The objective the task will be to recover the values of the vector ω and of the matrix of the centers $M = [\mu_1, ..., \mu_k]$, assuming to know the value of k. While an overview of approaches for tensor decomposition is presented in the introduction of this thesis, we present here in more detail a few techniques that are widely used in the literature: *Alternating Least Square* (Carroll and Chang, 1970; Harshman, 1970), *SVD method* from Anandkumar et al. (2012a) and *tensor power method* from Anandkumar et al. (2014).

Alternating Least Square and Other General-Purpose Decomposition Methods

Theoretically, the solution to system 1.4 could be found by looking just at the third order moment, solving the following set of d^3 equations:

$$M_3 = \sum_{i=1}^{k} \omega_i \mu_i \otimes \mu_i \otimes \mu_i \tag{1.9}$$

This is a consequence of the well-known Kruskal's criterion (Kruskal, 1977), which guarantees that M_3 has a unique CP decomposition of the form described at Equation (1.9). As a consequence, one could simply try to directly decompose the third order moment, without considering the constraints imposed by M_2 and M_1 which are theoretically redundant. To this objective, a variety of tensor decomposition algorithms exist to approximatively solve this problem, with the algorithm named Alternating Least Square (ALS) (Carroll and Chang, 1970; Harshman, 1970; Kolda and Bader, 2009) being by far the one most used by practitioners.

ALS is a general-purpose decomposition algorithm, that works even for non-symmetric tensors, a more general case with respect to that studied by methods of moments. Given a three-dimensional tensor T, it aims at finding three $d \times k$ matrices

$$A = [a_1, ..., a_k], \quad B = [b_1, ..., b_k], \quad C = [c_1, ..., c_k]$$

enabling the following decomposition:

$$T = \sum_{i=1}^{k} \lambda_i a_i \otimes b_i \otimes c_i \tag{1.10}$$

for a certain vector $\lambda = (\lambda_1, ..., \lambda_k)$. After a random initialization, the decomposition is found via an iterative algorithm that first fixes two of the matrices to be learned (say A and B) and then retrieves C by solving the overdetermined system of equations (1.10), leaving C as the only unknown variable. This algorithm, which is described in detail by Kolda and Bader (2009), however may have very large running times, because M_3 may be very high dimensional. Furthermore ALS is not guaranteed to find the matrices A, B and C allowing the decomposition at Equation (1.10), even when such decomposition is possible, being prone to poor local optima.

Beside ALS, literature provides several examples of general-purpose decomposition methods aimed at finding the decomposition of M_3 described at Equation (1.9) (see for example Tomasi and Bro, 2006; Kolda and Bader, 2009; Sidiropoulos et al., 2017). A recent example, that specifically focuses on the case of a symmetric tensor with the structure of M_3 is the *Random Projections* method, proposed by Kuleshov et al. (2015). The method consists in finding the matrix M by directly performing simultaneous diagonalization – using a variation of the well-known Jacobi algorithm (Cardoso and Souloumiac, 1996) – of random linear combinations of the slices of M_3, which the authors observe admitting the following writing

$$M_{3,r} = M\Omega^{1/2}\,\mathrm{diag}(m_r)\Omega^{1/2}M^\top,$$

where $M_{3,r}$ is the rth slice of M_3 and m_r is the rth row of M. This method, under an incoherence assumption on the vectors μ_i, can robustly recover the weights ω_i and vectors μ_i from the tensor M_3, even when it is not orthogonally decomposable. However, in practice the Random Projections method is way slower than ALS (see for example the experiments in Chapter 5) and hardly usable on high-dimensional tensors, reason why we will omit it from most of the experiments of this thesis.

The SVD Method

Anandkumar et al. (2012a) provide a simple technique to solve the system of equations (1.4), based on the well-known Singular Value Decomposition (SVD) of a matrix (Golub and Reinsch, 1970). Letting $M_2 = USU^\top$ be the SVD of M_2, the authors develop two equivalent writings for the second order moment:

$$M_2 = EE^\top, \;\; M_2 = M\Omega M^\top, \tag{1.11}$$

where $E = US^{1/2} \in \mathbb{R}^{d \times k}$ will be called the *whitening* matrix and $\Omega = \text{diag}(\omega)$, and show that there exists a unique orthogonal matrix $O \in \mathbb{R}^{k \times k}$ such that $M\Omega^{1/2} = EO$. The SVD method consists thus of finding O and retrieving the desired parameters by solving the following system of equations:

$$\begin{cases} M\Omega^{1/2} = EO \\ M\omega = M_1. \end{cases}$$

To find O, they introduce the following matrix-based characterization of the slices of M_3:

$$M_{3,r} = M\Omega^{1/2}\,\text{diag}(m_r)\Omega^{1/2}M^\top = EO\,\text{diag}(m_r)O^\top E^\top,$$

where $M_{3,r}$ is the rth slice of M_3 and m_r is the rth row of M. Multiplying $M_{3,r}$ at the left and at the right with the pseudoinverse (Stewart and Sun, 1990) of matrix E, they get a new matrix

$$H_r = E^\dagger M_{3,r}(E^\dagger)^\top = O\,\text{diag}(m_r)O^\top$$

whose left singular vectors coincide with the matrix O. However, the singular vectors O may not be uniquely determined if two or more values of m_r are identical; to solve this issue, Anandkumar et al. (2012a) suggest to perform a random linear combination of the various matrices H_r, using some random vector $\eta \in \mathbb{R}^d$, and get the matrix

$$H_\eta = O\,\text{diag}((\eta\mu_1^\top, ..., \eta\mu_k^\top))O^\top$$

whose left singular vectors are almost surely unique.

SVD method requires the matrix M to be full rank, and asks for $d \geq k$. Under these conditions, it guarantees to return the exact solution of the system (1.4) (Anandkumar et al., 2012a) – a guarantee that ALS does not present. The behavior of this method has been also studied in the context where it only has access to perturbed moments, providing guarantees of learning accuracy even in a perturbed setting.

Another critical advantage over ALS lays in the fact that it works with low-dimensional whitened matrices, a fact that dramatically reduces its memory requirements and speeds up its running time, which is extremely competitive.

However, despite its theoretical soundness, SVD method is known to have a poor learning accuracy, not being able to learn reliable model parameters when the input moments are perturbed (see Anandkumar et al. 2014, and Chapter 2).

Tensor Power Method

The tensor power method (TPM) (Anandkumar et al., 2014) also considers the whitening matrix defined above, $E = US^{1/2} \in \mathbb{R}^{d \times k}$, as defined in the previous section, but instead of whitening the slices of M_3, it directly whitens the full tensor M_3 into a symmetric orthogonally decomposable tensor:

$$T = \sum_{i=1}^{k} \omega_i E^\dagger \mu_i \otimes E^\dagger \mu_i \otimes E^\dagger \mu_i \in \mathbb{R}^{k \times k \times k}$$

The authors then observe that if (v, λ) is a robust eigenvector/eigenvalue pair of T, then there exists a column μ_i of M such that $\lambda E v = \mu_i$. Consequently, the problem reduces to find the robust eigenvectors of T, a task that is performed via a three-dimensional extension of the well-known matrix power method to find the eigenvectors of the matrix.

The requirements and the guarantees of TPM are similar to those of SVD method: M should be full rank, $d \geq k$ and it guarantees to return the exact solution of the system (1.4). A study of its robustness to perturbation has been carried on by Anandkumar et al. (2014); in general, TPM is known to provide a better learning accuracy with respect to the SVD method. TPM works with whitened tensors, which makes it faster than ALS. However, its running time is asymptotically slower than that of SVD method – depending a factor $O(k^5)$ on the number of latent states k. This makes the scalability the major drawback of TPM.

1.3 A Framework for Several Models

In the previous sections we have seen, for the Poisson's naive Bayes models, how to estimate from data three entities M_1, M_2 and M_3 of the form

$$M_1 = \sum_{i=1}^{k} \omega_i \mu_i, \tag{1.12}$$

$$M_2 = \sum_{i=1}^{k} \omega_i \mu_i \otimes \mu_i, \tag{1.13}$$

$$M_3 = \sum_{i=1}^{k} \omega_i \mu_i \otimes \mu_i \otimes \mu_i, \tag{1.14}$$

and how to recover from them the parameters that characterize the model, namely the mixing weights ω and the matrix of the centers $M = [\mu_1, ..., \mu_k]$.

The same approach can be extended to any latent variable model admitting a similar parameterizations, obtaining a general two-steps approach:

1. The first step consists of the estimation of the moments, as defined in Equations 1.12,1.13 and 1.14. The implementation of this step depends on the particular structure of the considered model. Above, we have seen an example for naive Bayes models, but analogous estimators can be found in the literature for e.g. Gaussian mixture models (Hsu and Kakade, 2013; Ge et al., 2015), hidden Markov models (Balle et al., 2014) or mixtures of linear regressions (Chaganty and Liang, 2013).

2. The second step of method of moments consists in decomposing the moments 1.12,1.13 and 1.14 to retrieve the desired model parameters, using, for example, one of the decomposition algorithms described above. This step is model-independent, in the sense that does not depend on the specific characteristics of the model. However, decomposition algorithms may impose requirements on the matrix M, as it is the case with TPM and SVD method, which require M to be full rank.

In the next chapters, we will focus on several aspects of methods of moments; we will introduce improved and more robust tensor decomposition methods, we will study how the robustness of a method of moments depends on the random noise in the input data and we will present applications of several latent variable models to various domains.

Chapter 2

Singular Value-Based Tensor Decomposition

A fundamental building block in methods of moments is the step that, from an estimate of the moments, retrieves an estimate of the parameters of a latent variable model describing the data. As explained in Chapter 1, this step is accomplished by solving the equations of the moments

$$M_1 = \sum_{i=1}^{k} \omega_i \mu_i,$$
$$M_2 = \sum_{i=1}^{k} \omega_i \mu_i \otimes \mu_i,$$
$$M_3 = \sum_{i=1}^{k} \omega_i \mu_i \otimes \mu_i \otimes \mu_i,$$

where only the left terms of the equations are known, providing algorithms that take as input the moments M_1, M_2 and M_3 and return the model parameters $\omega \in \mathbb{R}^k$ and $M = [\mu_k, ..., \mu_k] \in \mathbb{R}^{d \times k}$. Furthermore, in the most realistic setting, these methods are required to deal with approximate estimates of the moments $\tilde{M}_1, \tilde{M}_2$ and $\tilde{M}_3$ rather than with their unperturbed theoretical values; in this case, they are required to be robust to perturbations, returning approximate parameters $(\tilde{M}, \tilde{\omega})$ that are provably close to their theoretical counterparts (M, ω) if $\tilde{M}_1, \tilde{M}_2$ and $\tilde{M}_3$ are close to M_1, M_2 and M_3.

In Chapter 1, we have seen that there exist several algorithms able to accomplishing this task, from the naive approaches that directly decompose the tensor M_3 (Kolda and Bader, 2009), to the most recent methods from Anandkumar et al. (2012b, 2014) that, making use of the full set of low-order moments, provide theoretically robust methods. Existing decomposition methods differ from each other on various points. The methods from Anandkumar et al. (2012b,a) are only based on matrix operations, and are thus simple to implement and understand. They are fast but sensitive to perturbations, as they are based on the intensive usage of the singular vectors of perturbed matrices, which are known to be unstable to random noise (to this purpose, see the work by Stewart 1990, Section 6). Tensor Power Method instead (Anandkumar et al., 2014) is more difficult to implement, requiring deep familiarity with tensor operations and is slower, requiring $O(k^5)$ operations, but guarantees a better stability with respect to the perturbations of the input moments.

In this chapter we provide a new decomposition algorithm that aims at overcoming the limitations of existing techniques, named SVTD, ***S**ingular **V**alue based **T**ensor **D**ecomposition*. This method is alternative to the ones presented above, and is based on the singular values of a SVD, which are known to be stable under random perturbations (unlike the singular vectors, as shown by Stewart 1990, Section 2). Our algorithm is simple to implement and understand, is deterministic and based only on standard matrix operations rather than tensor ones. Experimental results (see Section 2.3) show that SVTD outperforms existing matrix-based methods from Anandkumar et al. (2012a,b) in terms of reconstruction accuracy, is an order of magnitude faster than TPM providing a similar learning accuracy.

The outline of the chapter is the following: Section 2.1 contains the proposed decomposition algorithm, Section 2.2 studies its sensitivity to perturbations and Section 2.3 tests the presented method against existing decomposition techniques.

2.1 Singular Value based Tensor Decomposition

In this section we present an algorithm to solve the system of Equations (1.4), retrieving the parameters of a latent variable model from the equations

of the moments. The proposed method only relies on matrix decomposition techniques, and is deterministic. We will experimentally see in Section 2.3 see that it also scales better in time and memory than TPM (Anandkumar et al., 2014), and is more robust than the matrix-based approaches from Anandkumar et al. (2012b,a).

Our method relies on the observation that M_2 and the slices of M_3 admit a representation in terms of matrix products as

$$M_2 = M\Omega M^\top, \tag{2.1}$$

$$M_{3,r} = M\Omega^{1/2}\,\mathrm{diag}(m_r)\Omega^{1/2}M^\top, \tag{2.2}$$

where $\Omega = \mathrm{diag}(\omega)$, $M_{3,r} \in \mathbb{R}^{d\times d}$ is the rth slice of M_3 and m_r is the rth row of M. As a first step, it stores the whitened slices of M_3 into a three-dimensional tensor H in $\mathbb{R}^{d\times k\times k}$ and then performs d SVD on the slices of H (belonging to $\mathbb{R}^{k\times k}$) obtaining the rows of M as the singular values of that slices; for this reason we name our method Singular Value-based Tensor Decomposition (SVTD). The key steps of our algorithm (SVTD) are outlined in Algorithm (1).

Algorithm 1 SVTD

Require: M_1, M_2, M_3, and the number of latent states $k \leq d$

1: Decompose M_2 as $M_2 = U_k S_k U_k^\top$ with a SVD truncated at the kth singular vector.
2: Define the whitening matrix $E = U_k S_k^{1/2}$ and calculate its pseudoinverse $E^\dagger = (S_k)^{-1/2}U_k^\top$.
3: Select a feature r and compute $M_{3,r}$
4: Compute O as the left singular vectors of $H_r = E^\dagger M_{3,r} {E^\dagger}^\top$ and m_r as the singular values.
5: **for** $i = 1 \to d$ **do**
6: Compute $H_i = E^\dagger M_{3,i} {E^\dagger}^\top$, where $E^\dagger = (S_k)^{-1/2}U_k^\top$ is the Moore-Penrose pseudo-inverse of E
7: Obtain the ith row of M as the diagonal entries of $O^\top H_i O$
8: **end for**
9: Obtain ω solving $M_1 = M\omega$
10: Return (M, ω)

The constructive proof of the following theorem will explain why SVTD

performs a correct retrieval of the desired model parameters.

Theorem 2.1.1. *Let r be the feature selected at Step 3 of SVTD, and assume $k \leq d$. If all the elements of m_r are distinct and M_2 and M_3 have rank k, then SVTD produces the values of (M, ω) exactly.*

Proof. As a first step we perform a SVD to $M_2 = U_k S_k U_k^\top$, where $U_k \in \mathbb{R}^{d\times k}$ and $S_k \in \mathbb{R}^{k\times k}$ are the matrices of the singular vectors and values truncated at the kth greatest singular value. Then we define the whitening matrix $E = U_k S_k^{1/2} \in \mathbb{R}^{d\times k}$, and for a slice $M_{3,r}$ of M_3, we define $H_r \in \mathbb{R}^{k\times k}$ as

$$H_r = E^\dagger M_{3,r} (E^\top)^\dagger,$$

where $E^\dagger = (S_k)^{-1/2} U_k^\top$ is the Moore-Penrose pseudo-inverse of E. Now, observe that there exists a unique orthogonal $k \times k$ matrix O, such that

$$M\Omega^{1/2} = EO. \tag{2.3}$$

To see that such a matrix exists, it is enough to observe that the matrix $O = E^\dagger M \Omega^{1/2}$ is orthogonal (which can be seen from equation $M\Omega M^\top = EE^\top$) and fulfills the requested relation. To see that it is unique, assume that that O_1 and O_2 are two orthogonal matrices such that $M\Omega^{1/2} = EO_1 = EO_2$. Then, multiplying by $E^\dagger$ on the left, we obtain $O_1 = E^\dagger E O_1 = E^\dagger E O_2 = O_2$. Using Equation (2.3), one gets the following characterization of $M_{3,r}$:

$$M_{3,r} = M\Omega^{1/2} \operatorname{diag}(m_r) \Omega^{1/2} M^\top = EO \operatorname{diag}(m_r) O^\top E^\top,$$

from which it follows that

$$H_r = O \operatorname{diag}(m_r) O^\top.$$

Now, one gets the rth row of M as the *singular values* of H_r. Repeating these steps for all the $i \in \{1, ..., d\}$ will provide the full matrix M. In order to avoid ordering issues with the columns of the retrieved M, one can use the same matrix O to diagonalize all the matrices H_i, as O is uniquely determined, because the elements of m_r are distinct. So, compute O as the singular vectors of H_r, for a certain feature r, and re-use it for all the other features. The subsequent estimations of ω is straightforward, and can be obtained by solving the linear system $M_1 = M\omega$. □

Remark 2.1.1 (On the generality of the algorithm)**.** Like other decomposition algorithms for methods of moments, our method requires the number of latent states to be smaller than the features size, $k \leq d$, which is in general a realistic requirement: think for example of the naive Bayes models to perform clustering; in general one wants to retrieve few tens of clusters from hundreds of available features. Also, we remark that during the construction of the algorithm we have not made any hypotheses on the probability distribution of the data; instead, we have required the matrix M to be full rank and, in addition, to have at least one feature r with different conditional expectations on the various clusters. Furthermore, we do not need to know in advance what this feature is, as Remark 2.1.2 will explain. This last requirement is not present in the other matrix-based methods, as they rely on a randomized matrix to guarantee the uniqueness of the results (see Section 2.1.1); but the reconstruction accuracy of these methods is significantly worse than that of SVTD, as we will show experimentally. Finding a deterministic method that joins the scalability properties of simultaneous diagonalization methods without requiring this separation condition is an interesting open problem.

Remark 2.1.2 (On the selection of feature r)**.** The initial steps of the algorithm require the isolation of a feature r to compute the matrix O. While theoretically we could select any feature r such that all elements of $m_r = (\mu_{1,r}, ..., \mu_{k,r})$ are distinct, it is clear that, if matrices M_1, M_2 and M_3 are subject to perturbations, the results obtained by the algorithm might vary, depending on the selected feature r. However, known results from matrix perturbation theory (see for example Stewart and Sun 1990), indicate that the matrix O (which contains the singular vectors of H_r) is more sensitive to random perturbations when the minimum difference between the entries of $m_r = (\mu_{1,r}, ..., \mu_{k,r})$ is small. This fact suggests that a possible heuristic to choose the feature r (and hence a reliable matrix O), is to repeat steps 3 and 4 of the algorithm, isolating different features and selecting the one that maximizes the quantity $\min_{i \neq j}(|\mu_{i,r} - \mu_{j,r}|)$. With this heuristic, a user can run SVTD without any previous knowledge of the feature to extract. Its cost is that of performing d times a $k \times k$ SVD, therefore $O(d\,k^3)$, and is dominated by the costs of other parts of the algorithm as discussed next.

Remark 2.1.3 (Complexity analysis)**.** We start analyzing the time complexity. Using randomized SVD techniques (see Halko et al., 2011), step 1 can be carried out with a total of $O(d^2k)$ steps, the SVD on $E^\dagger M_{3,r}(E^\top)^\dagger$ requires $O(k^3)$ steps while step 6 requires $O(d^2k)$ steps for matrix multiplication for

each one of the d iterations. So the overall complexity of Algorithm 1 is

$$O(d^2k + k^3 + d^3k).$$

We need to add the additional cost of the feature-selection method outlined in Remark 2.1.2, whose cost is $O(dk^3)$. Since $d \geq k$, this cost is dominated by the larger $O(d^3k)$ component.

It is important to highlight that the general implementation given in Algorithm 1 has mainly a descriptive purpose. For a specific latent variable model, optimized implementations may exists, as we will see in Chapter 4. Computations can be further accelerated by exploiting the sparsity of the data, that is, using sparse matrix arithmetic techniques. We also remark that the algorithm is trivially parallelizable: Assuming that we have m machines on which to parallelize steps 5, 6, 7 of the algorithm and the feature selection task, we can reduce the total running time to $O(d^2k + k^3 + d^3k/m)$.

Regarding memory, notice that we never use the full tensor M_3, but only its rth slice, for the selected feature r. This means that only that slice has to be computed and stored, and the memory complexity of the algorithm is $O(d^2)$.

These complexity requirements are comparable to those of the methods from Anandkumar et al. (2012a,b); however, those methods are randomized, with nontrivial variance in their output, so they may require several runs of the full algorithm in order to provide accurate results. The tensor power method from Anandkumar et al. (2014) has in general higher cost. It is an iterative technique, with a number of iterations difficult to bound a priori; the authors suggest that accuracy ϵ can be reached with $O(k^{5+\delta}(\log(k) + \log\log(1/\epsilon)))$ operations, among iterations, random restarts and actual matrix operations, higher than our $O(k^3)$. To this time we need to add the time to get the $k \times k \times k$ whitened tensor from the sample, which is not trivial for many LVMs.

2.1.1 Comparison with other decomposition methods.

As said in Chapter 1, Kruskal's criterion guarantees that M_3 has a unique CP decomposition, so, instead of SVTD, one could simply try to directly decompose the third order moment, using for example one of the general-purpose tensor decomposition methods described by Tomasi and Bro (2006); Kolda and Bader (2009); Sidiropoulos et al. (2017), to retrieve the parameters (M, ω). However, this approach is impractical: M_3 may be too large,

and explicitly decomposing it may not even be computationally feasible on common machines. SVTD instead relies on the full system of equations (1.4), and uses a whitening matrix to work with low dimensional tensors, while maintaining guarantees of provably solving the considered system. For example, the whitening step that SVTD performs at step 4 of Algorithm 1 reduces the slices of M_3 to small $k \times k$ matrices, reducing the complexity of the algorithm and allowing for fast yet optimal solutions on commodity machines, even when d has is very high.

The tensor power method (TPM) – described by Anandkumar et al. (2014) and recalled in Chapter 1 – is the most popular alternative. While very robust, the implementation of this method may be complex for someone who is not familiar with tensors. In addition, it is an iterative method, and so it requires a tuning of the hyperconvergence parameters, which might require many trial-and-error tests. These practical considerations, together with its higher time and memory complexity outlined in Remark 2.1.3, are drawbacks compared to matrix-based methods like SVTD.

Matrix-based methods from Anandkumar et al. (2012a,b), are technically more similar to the method presented here, and they are two variations of the same approach, one (Anandkumar et al., 2012b) using eigenvectors, and the other (Anandkumar et al., 2012a) using singular vectors, and have been presented as well in Chapter 1. There are at least two differences between SVTD and these approaches:

1. The first difference is the way the matrix M is retrieved. In fact, the SVD method from Anandkumar et al. (2012a) recovers M essentially from Equation (2.3), relying on the identity

   $$M\Omega^{1/2} = EO$$

   while SVTD recovers the rth row of M from the singular values of the matrix

   $$H_r = E^\dagger M_{3,r} (E^\dagger)^\top = O \operatorname{diag}(m_r) O^\top.$$

2. The second difference, is how the matrix O is retrieved; SVD method gets it as the singular vectors of

   $$H_\eta = E^\dagger (\sum_{r=1}^{d} \eta_r M_{3,r})(E^\dagger)^\top = O \operatorname{diag}((\eta\mu_1^\top, ..., \eta\mu_k^\top)) O^\top,$$

for some random vector $\eta \in \mathbb{R}^d$. The introduction of the random vector η has the purpose of guaranteeing that the elements of $(\eta\mu_1^\top, ..., \eta\mu_k^\top)$ are almost surely distinct, and so O is unique. In SVTD instead we fix a specific feature r, choosing the one with the maximum minimum variation between the feature components; this is the same of saying that we fix η to be the rth coordinate vector, giving a recipe for finding r in Remark 2.1.2, providing in this way the choice that maximizes stability.

2.2 Perturbation Analysis

In the previous section we have presented an algorithm to recover the parameters (M, ω) from the unperturbed low-order moments M_1, M_2 and M_3. However, in real world applications, we do not have access to the exact values of the moments, but to a set of estimators $\tilde{M}_1$, $\tilde{M}_2$ and $\tilde{M}_3$ that aim at approximating them, with the guarantee that each estimator $\tilde{M}_i$ gets closer to its theoretical counterpart M_i while the sample size increases. When learning a latent variable model from data, we thus plug the estimated moments $\tilde{M}_1$, $\tilde{M}_2$ and $\tilde{M}_3$ into a decomposition algorithm like SVTD; ideally, the more similar the perturbed moments are to their expectations, the better the output parameters $(\tilde{M}, \tilde{\omega})$ of the algorithm are expected approximate (M, ω). The following theorem shows that SVTD presents this property, showing how the approximation of the input moments propagates to the estimated model parameters.

Theorem 2.2.1. *Given the unperturbed versions of M_1, M_2 and M_3 and the feature we want to isolate, r, let α_r and α_{M_2} be*

$$\alpha_r = \min_{i \neq j}(|\mu_{i,r} - \mu_{j,r}|) > 0, \quad \alpha_{M_2} = \min_{i \leq k} (\sigma_i(M_2)^2 - \sigma_{i+1}(M_2)^2) > 0$$

where $\sigma_i(M_2)$ are the singular values of M_2. Assume the empirical estimates $\tilde{M}_2$ and $\tilde{M}_3$ satisfy

$$||\tilde{M}_2 - M_2||_F < \epsilon, \quad ||\tilde{M}_3 - M_3||_F < \epsilon.$$

Then, there exists a function[1] $\gamma(M, \omega)$, of the model parameters, such that, if $\epsilon < \gamma(M, \omega)$, Algorithm 1, fed with $\tilde{M}_1$, $\tilde{M}_2$ and $\tilde{M}_3$, provides an estimated

[1]For an explicit formulation of the value of $\gamma(M, \omega)$, we refer the reader to the proof of the theorem and to Remark 2.2.1.

matrix $\tilde{M}$ *whose rows* $\tilde{m}_i$*, for all* $i \in \{1, ..., d\}$*, satisfy*

$$||m_i - \tilde{m}_i||_2 \leq C_1\epsilon + \sqrt{k}\frac{\epsilon}{\alpha_r}(C_2 + \frac{C_3}{\alpha_{M_2}}) + O(\sqrt{k}C_4\epsilon^2)$$

where m_i *are the rows of* M*,* C_1*,* C_2 *and* C_3 *are polynomial functions of* $||E||_F, ||O||_F$ *and* $||M_{3,i}||_F$*, and* C_4 *is a polynomial function of* $||E||_F$*,* $||O||_F$*,* $||M_{3,i}||_F$*,* $1/\alpha_r$ *and* $1/\alpha_{M_2}$*.*

A key role is played by α_r, the minimum difference between the elements of $(\mu_{1,r}, ..., \mu_{k,r})$; when samples are large enough, the theorem guarantees that the algorithm works correctly, although "large enough" depends on α_r. When this condition is not satisfied, the learning algorithm still works, but might provide output results that are different from the theoretical generative model. We recall here Remark 2.1.2, where we wondered how to select the proper feature r; Theorem 2.2.1 confirms the intuition that the one guaranteeing the highest accuracy would be the one with the highest possible α_r.
The role played by α_{M_2} is less intuitive, as it is not directly related to the parameters of the model. Looking at the proof of the theorem, it is possible to see that α_{M_2} emerges as a consequence of the whitening step. In particular, the whitening matrix E is calculated from the singular vectors of the perturbed M_2, whose sensitivity to perturbations gets worse when α_{M_2} is small. We can conjecture that if we were able to find a different method to retrieve the whitening matrix, we could get rid of the component dependent on α_{M_2} in the perturbation theorem; this is equivalent to claim that the components depending on α_{M_2} in the perturbation bound are a consequence of the algorithm that we are using rather than of the complexity of the problem. The exploration of this conjecture and the research of algorithms to calculate the whitening matrix not depending on α_{M_2} is an open problem for future research.

2.2.1 Proof of Theorem 2.2.1

The goal of the proof is to develop a perturbation bound for each row m_i of the unknown matrix M such that, $||\tilde{m}_i - m_i||_2 \leq Bound(\epsilon)$, for a certain function $Bound(\epsilon)$. We notice, from Algorithm 1, that each $\tilde{m}_i$ is obtained as the diagonal entries of the following matrix:

$$\tilde{O}^\top \tilde{E}^\dagger \tilde{M}_{3,i} (\tilde{E}^\top)^\dagger \tilde{O},$$

where the tildes indicate that we are dealing with perturbed matrices. Therefore, we will need to find the perturbations of the matrices composing this equation, as the following relation holds:

$$||\tilde{m}_i - m_i||_2 \leq ||\tilde{O}^\top \tilde{E}^\dagger \tilde{M}_{3,i} (\tilde{E}^\top)^\dagger \tilde{O} - O^\top E^\dagger M_{3,i} (E^\top)^\dagger O||_F.$$

In short, having perturbation bounds on $\tilde{M}_{3,i}$,$\tilde{E}^\dagger$ and $\tilde{O}$ will be sufficient to reach our goal.

Perturbations on $\tilde{M}_{3,i}$

We know by hypothesis that $||\tilde{M}_{3,i} - M_{3,i}||_F = ||\Delta_{M_{3,i}}||_F < \epsilon$.

Perturbations on $\tilde{E}^\dagger$

It is a known fact (see Stewart, 1990) that, given the SVD $\tilde{M}_2 = \tilde{U}\tilde{S}\tilde{V}$, and $M_2 = USU^\top$, if $||\tilde{M}_2 - M_2||_F < \epsilon$, we have that

$$||\tilde{S} - S||_F \leq \epsilon. \tag{2.4}$$

Algorithm 1, when fed with perturbed moments retrieves a perturbed whitening matrix $\tilde{E} = \tilde{U}_k(\tilde{S}_k)^{1/2}$, while the unperturbed value of E is defined as $E = U_k(S_k)^{1/2}$, where the subscript k indicates the truncation at the kth singular value. So, to reach a perturbation bound on $\tilde{E}^\dagger$, we first need to look for a perturbation bound on $\tilde{E}$, that will be obtained bounding the error of $(\tilde{S}_k)^{1/2}$ and $\tilde{U}_k$. The first one is a consequence of equation (2.4): if $\Delta_S = (\tilde{S}_k)^{1/2} - S_k^{1/2}$, we have

$$||\Delta_S||_F < \frac{\epsilon}{2\sqrt{\sigma_k(M_2)}},$$

where $\sigma_k(M_2)$ is the kth the singular value of M_2. To find a bound on $\tilde{U}_k$ we will use Lemma 2.2.1 to get

$$||\tilde{U}_k - U_k||_F < \sqrt{8k}\frac{2\epsilon||M_2||_F + \epsilon^2}{\alpha_{M_2}},$$

where $\alpha_{M_2} = \min_{i \leq k} (\sigma_i(M_2)^2 - \sigma_{i+1}(M_2)^2)$ and $\sigma_i(M_2)$ are the singular values of M_2. We thus conclude that, if $\Delta_U = \tilde{U}_k - U_k$, $\tilde{E} = E + \Delta_U S^{1/2} + U\Delta_S + \Delta_U\Delta_S$,

and hence

$$||\tilde{E} - E||_F < f(\epsilon) = \epsilon\Bigg(||S^{1/2}||_F\sqrt{8k}\frac{2||M_2||_F + \epsilon}{\alpha_{M_2}} + \frac{||U||_F}{2\sqrt{\sigma_k(M_2)}} + \\ + \sqrt{8k}\frac{2\epsilon||M_2||_F + \epsilon^2}{\alpha_{M_2}(2\sqrt{\sigma_k(M_2)})}\Bigg).$$

We are now ready to find a perturbation bound on the pseudoinverse of $\tilde{E}$, using a known bound from Stewart and Sun (1990): if $||\tilde{E} - E||_F < f(\epsilon)$ then

$$||\tilde{E}^\dagger - E^\dagger||_F \leq f(\epsilon)\tau(E)$$

where $\tau(E) = ||E^\dagger||_F^2 + ||(E^\top E)^{-1}||_F||\mathbb{I} - EE^\dagger||_F$.

Perturbations on $\tilde{O}$

We now look for the value of $g(\epsilon) = ||O - \tilde{O}||_F$. If SVTD is fed with unperturbed moments, O comes from the decomposition of $E^\dagger M_{3,r}(E^\top)^\dagger$:

$$E^\dagger M_{3,r}(E^\top)^\dagger = O\,\mathrm{diag}(m_r)O^\top. \tag{2.5}$$

$\tilde{O}$ will be obtained from the singular vectors of the same, but perturbed, matrix: $\tilde{E}^\dagger\tilde{M}_{3,r}(\tilde{E}^\top)^\dagger$. First, we observe that

$$\begin{aligned} ||E^\dagger M_{3,r}(E^\top)^\dagger - \tilde{E}^\dagger\tilde{M}_{3,r}(\tilde{E}^\top)^\dagger||_F \leq & \\ \leq & 2f(\epsilon)\tau(E)||E^\dagger||_F||M_{3,r}||_F + \epsilon||E^\dagger||_F^2 \\ & + 2\epsilon f(\epsilon)\tau(E)||E^\dagger||_F \\ & + f(\epsilon)^2\tau(E)^2(||M_{3,r}||_F + \epsilon) \\ = & h(\epsilon). \end{aligned}$$

Using Corollary 2.2.1, we assume the hypothesis that

$$h(\epsilon) \leq \frac{\alpha_r^2}{\sqrt{2}\left(2||H_r||_F(1 + \sqrt{1 - \frac{1}{k}}) + \sqrt{\alpha_r^2 + 4||H_r||_F^2(1 + \sqrt{1 - \frac{1}{k}})^2}\right)} \tag{2.6}$$

where $H_r = E^\dagger M_{3,r}(E^\top)^\dagger$ and $\alpha_r = \min_{i\neq j}(|\mu_{i,r} - \mu_{j,r}|)$, to get that $||\tilde{O} - O||_F < 2\sqrt{2}h(\epsilon)/\alpha_r = g(\epsilon)$. We are now able to conclude our proof by analyzing

$$||m_i - \tilde{m}_i||_2 \leq ||\tilde{O^\top}\tilde{E}^\dagger\tilde{M}_{3,i}(\tilde{E^\top})^\dagger\tilde{O} - O^\top E^\dagger M_{3,i}(E^\top)^\dagger O||_F \leq$$

$$\leq P_1 g(\epsilon) + P_2\epsilon + P_3 f(\epsilon) + O(\epsilon^2 P_4),$$

where P_1, P_2 and P_3 are polynomials in $||E||_F, ||O||_F$ and $||M_{3,i}||_F$, and P_4 is a polynomial in $||E||_F, ||O||_F, ||M_{3,i}||_F, g(\epsilon), \epsilon$ and $f(\epsilon)$. The thesis follows making explicit these polynomials.

□

Remark 2.2.1. In the statement of Theorem 2.2.1, we said that there exists a number $\gamma(M, \omega)$, such that, if $\epsilon < \gamma(M, \omega)$, the perturbation bound of the thesis works. Looking at the proof of the theorem, we are able to explicitly calculate this number, just by solving the inequality (2.6). We can practically think at $\gamma(M, \omega)$ as the largest value of ϵ that satisfies this inequality.

Lemma 2.2.1. *Consider M_2, the perturbed $\tilde{M}_2$, and their SVD, $\tilde{M}_2 = \tilde{U}\tilde{S}\tilde{V}^\top$, $M_2 = USU^\top$. Let $\tilde{U}_k$ and U_k be matrices of the first k left singular vectors of $\tilde{M}_2$ and M_2. Define $\alpha_{M_2} = \min_{i \leq k}(\sigma_i(M_2)^2 - \sigma_{i+1}(M_2)^2)$. If $||\tilde{M}_2 - M_2|| < \epsilon$, then $||\tilde{U}_k - U_k||_F < \sqrt{8k}\frac{2\epsilon||M_2||_F + \epsilon^2}{\alpha_{M_2}}$.*

Proof. Consider $\tilde{M}_2\tilde{M}_2{}^\top = \tilde{U}\tilde{S}^2\tilde{U}^\top$ and $M_2 M_2^\top = US^2U^\top$. Then $||\tilde{M}_2\tilde{M}_2{}^\top - M_2 M_2{}^\top||_F \leq 2\epsilon||M_2||_F + \epsilon^2$. Take now the matrix of the first k columns of U and $\tilde{U}$, that are eigenvectors of $M_2 M_2{}^\top$ and $\tilde{M}_2\tilde{M}_2{}^\top$, obtaining $U_k = [u_1, .., u_k]$ and $\tilde{U}_k = [\tilde{u}_1, .., \tilde{u}_k]$. From Theorem 2.2.2 we have that, for any $i = 1, ..., k$, holds

$$\begin{aligned} ||u_i - \tilde{u}_i|| &\leq 2^{\frac{3}{2}} \frac{2\epsilon||M_2||_F + \epsilon^2}{\min(\sigma_{i-1}(M_2)^2 - \sigma_i(M_2)^2, \sigma_i(M_2)^2 - \sigma_{i+1}(M_2)^2)} \\ &\leq 2^{\frac{3}{2}} \frac{2\epsilon||M_2||_F + \epsilon^2}{\alpha_{M_2}} \end{aligned}$$

from which the thesis follows.

□

The following results, taken from Yu et al. (2015) and Chen et al. (2012), present perturbation bounds on the eigenvectors and on the singular vectors of symmetric matrices.

Theorem 2.2.2 (Cor. 1, pg. 4 Yu et al. 2015)**.** *Consider A and $\tilde{A}$ two symmetric matrices in $\mathbb{R}^{d\times d}$, with eigenvalues $\lambda_1 \geq ... \geq \lambda_d$ and $\tilde{\lambda}_1 \geq ... \geq \tilde{\lambda}_d$. Fix a $j \in \{1, ..., d\}$ and assume that $\min(\lambda_{j-1} - \lambda_j, \lambda_j - \lambda_{j+1}) > 0$, where we*

define $\lambda_0 = \infty$ *and* $\lambda_{d+1} = -\infty$. *If* $v \in R^d$ *(resp.* $\tilde{v}$*) is an eigenvector of* A *(resp.* $\tilde{A}$*), associated to* λ_i *(resp.* $\tilde{\lambda}_i$*), then*

$$||v - \tilde{v}|| \leq \frac{2^{\frac{3}{2}}||A - \tilde{A}||_F}{\min(\lambda_{j-1} - \lambda_j, \lambda_j - \lambda_{j+1})}$$

Theorem 2.2.3 (Thm. 3.2, Chen et al. 2012)**.** *Let* $B \in \mathbb{R}^{k \times k}$ *be a matrix, with SVD* $B = Udiag((\sigma_1, ..., \sigma_k))V^\top$ *with* $\sigma_1 > \sigma_2 > ... > \sigma_k > 0$*, and let* $\tilde{B} = B + \Delta_B$ *be a perturbed matrix, with SVD* $\tilde{B} = \tilde{U}diag((\tilde{\sigma}_1, ..., \tilde{\sigma}_k))\tilde{V}^\top$. *Define:*

$$\alpha_B = \min_{i \neq j} |\sigma_i - \sigma_j| > 0, \quad \epsilon_B = \frac{\sqrt{2}||\Delta_B||_F}{\alpha_B}, \quad \gamma_B = \frac{||B||_F}{\alpha_B}(1 + \sqrt{1 - \frac{1}{k}})$$

Then, if

$$\epsilon_B \leq \frac{1}{2\gamma_B + \sqrt{1 + 4\gamma_B^2}} \tag{2.7}$$

The following upper bound holds:

$$||U - \tilde{U}||_F \leq \frac{\sqrt{2}\epsilon_B}{\sqrt{1 - 2\gamma_B\epsilon_B + \sqrt{1 - \epsilon_B^2 - 4\gamma_B\epsilon_B}}} \tag{2.8}$$

The following corollary is essentially a rewriting of the previous theorem.

Corollary 2.2.1. *In the same setting of Theorem 2.2.3, if there exists an* $\epsilon > 0$ *such that*

$$||\Delta_B|| < \epsilon \leq \frac{\alpha_B^2}{\sqrt{2}\left(2||B||_F(1 + \sqrt{1 - \frac{1}{k}}) + \sqrt{\alpha_B^2 + 4||B||_F^2(1 + \sqrt{1 - \frac{1}{k}})^2}\right)}$$

then $||U - \tilde{U}||_F \leq 2\sqrt{2}\frac{\epsilon}{\alpha_B}$.

Proof. Note that if ϵ satisfies the hypothesis of the corollary, then (2.7) is satisfied and hence we have (2.8):

$$||U - \tilde{U}||_F \leq \frac{\sqrt{2}\epsilon_B}{\sqrt{1 - 2\gamma_B\epsilon_B + \sqrt{1 - \epsilon_B^2 - 4\gamma_B\epsilon_B}}} \leq \frac{\sqrt{2}\epsilon_B}{\sqrt{1 - 2\gamma_B\epsilon_B}}$$

Now we plug in the last equation the bound of (2.7), to get

$$
\begin{aligned}
||U - \tilde{U}||_F &\leq \frac{\sqrt{2}\epsilon_B}{\sqrt{1 - \frac{2\gamma_B}{2\gamma_B + \sqrt{1+4\gamma_B^2}}}} \\
&= \sqrt{2}\epsilon_B \sqrt{\frac{2\gamma_B + \sqrt{1+4\gamma_B^2}}{\sqrt{1+4\gamma_B^2}}} \\
&\leq \sqrt{2}\epsilon_B \sqrt{\frac{2\gamma_B}{\sqrt{1+4\gamma_B^2}} + 1} \\
&\leq 2\epsilon_B
\end{aligned}
$$

□

2.3 Experiments

In this section we test the algorithm presented in this chapter against other decomposition methods.

Experimental setting: All the experiments in this section have been run on a MacBook Pro with a 2.7 GHz Intel Core i5 processor and 8 GB of RAM memory. All the algorithms have been implemented in Python 2.7 (interpreted, not compiled), using the *numpy* (Walt et al., 2011) library for all linear algebra operations, including Singular Value Decomposition, for which we used numpy's non-randomized SVD.[2] SVTD and the methods from Anandkumar et al. (2012a,b, 2014) have been implemented by this thesis author. For ALS we have used the implementation provided by *scikit-tensor*.[3] All the implementations written by the author of this thesis, as well as the code of the implementations of the competing methods, have been publicly disclosed and are freely accessible.[4]

We test SVTD on the task of decomposing a tensor of moments obtained from

[2] `https://docs.scipy.org/doc/numpy/reference/generated/numpy.linalg.svd.html`

[3] `https://github.com/mnick/scikit-tensor`

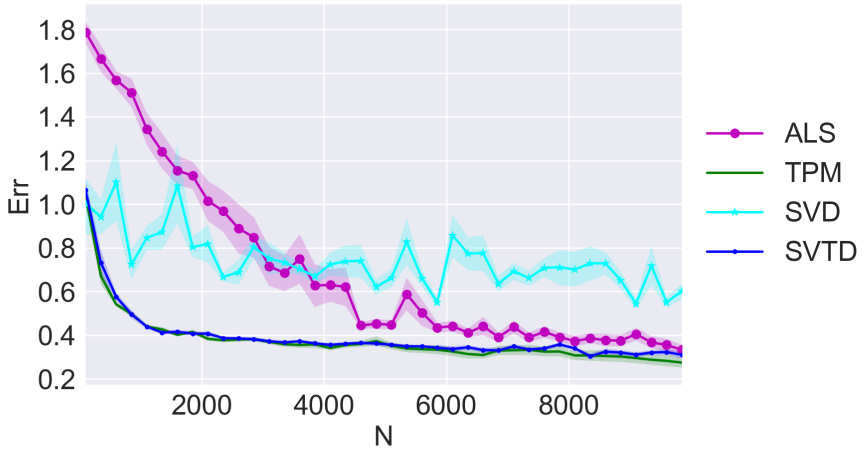
[4] `https://github.com/mruffini/SpectralMethod.git`

a random sample, comparing its performance with that of existing methods. We fix a dimension of $d = 100$ features with $k = 5$, and, for several sample sizes comprised between $n = 1000$ and $n = 10,000$, we generate a random sample $\mathcal{X}$ distributed as a Poisson's naive Bayes model (see Chapter 1) with parameters[5] (M, ω). For each synthetic sample, we proceed as follows:

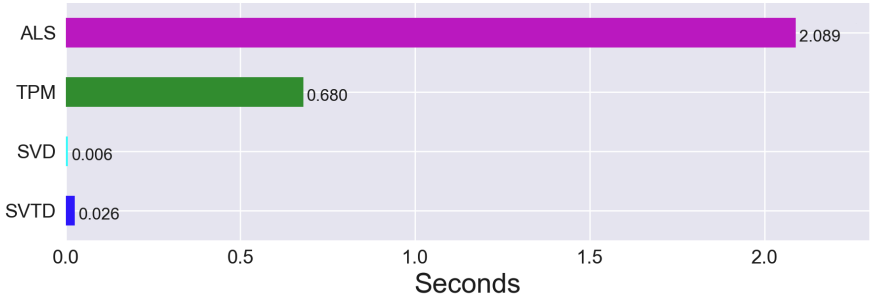
- We estimate the values of $\tilde{M}_1, \tilde{M}_2$ and $\tilde{M}_3$ using Theorem 1.2.1.
- We retrieve from the estimated $\tilde{M}_1, \tilde{M}_2$ and $\tilde{M}_3$ the pair of unknowns $(\tilde{M}, \tilde{\omega})$ using SVTD as in Algorithm 1. We also generate alternative solutions using the decomposition algorithms from Anandkumar et al. (2014) ("Tensor power method", with 25 random restarts and 20 iterations per restart), from Anandkumar et al. (2012a) ("SVD method"), and with ALS as described by Kolda and Bader (2009) — where ALS is used to directly find the CP decomposition of the $\tilde{M}_3$, with a random initialization and stopping after 250 iterations.
- Each time we generate a solution, we register the time in seconds employed by the various algorithms. For each method, we represent the average time used to recover the parameters over the various runs in Figure 2.1b.
- For each set of retrieved parameters $(\tilde{M}, \tilde{\omega})$ we calculate the learning error as: $Err = \|\tilde{M} - M\|_F$, where M is the matrix used in the simulations. For each sample size, we repeat the experiment 10 times (each time with newly generated parameters), and we plot in Figure 2.1a the results in function of n.

In Figure 2.1b, the average running times of the various methods are presented. On average, SVTD is way faster than ALS and TPM — as a consequence of the better dependence on the number of latent states — and it is slightly slower than the SVD method, due to the feature selection process outlined in Remark 2.1.2. With respect to accuracy, Figure 2.1a shows that SVTD and TPM are quite comparable among themselves, being the top performers in terms of learning accuracy.

[5]For each sample size, the parameters (M, ω) have been randomly generated by sampling a uniform distribution and normalizing the columns of M to have sum 5, and the vector ω to have sum 1.



(a)



(b)

Figure 2.1: In Figure 2.1b the average running times are represented. Figure 2.1a contains the analysis of the learning accuracy; the $x-$axis represents the size of the synthetic dataset while the $y-$axis is Err, the reconstruction error for the various tested methods. The shaded areas represent the variance of the output of the experiments over 10 runs with same sample size.

We conclude that SVTD provides learning accuracy similar to TPM while having a running time in the order of the more inaccurate SVD-based method. Also, it is deterministic and requires little hyperparameter tuning. ALS also has worse performance in terms of learning accuracy and is far slower than the other methods tested.

2.4 Conclusions

In this Chapter we have introduced SVTD, a new algorithm to solve the moment Equations (1.12), (1.13) and (1.14) and retrieve from them the parameters of a latent variable model. Experimentally, SVTD guarantees a leaning accuracy comparable or better to that of TPM while having significantly shorter running times – running times that are comparable to those of the less robust SVD method. The robustness of SVTD to random perturbations of input data has been theoretically assessed in Theorem 2.2.1, providing a bound that depends on the minimum difference between the elements of the vector $(\mu_{1,r}, ..., \mu_{k,r})$ – an expected result, being the condition of such difference to be greater than zero a constraint for the proper functioning of SVTD. Improving SVTD to remove such constraint and studying the feasibility of tighter perturbations bounds are open points for future research.

Chapter 3

Methods of Moments for Topic Models

The automatic mining of topics in large text corpora is one of the most natural fields of applications of latent variable models. Consider for example a text document, that is dealing with one or more topics. Clearly, these topics will influence the words that are likely to appear in the text; if for example we are reading a paper about *mathematics*, it will be very unlikely to find words like *onion* or *tomato*. Vice-versa it is unlikely to find words like *limit* or *partial derivative* in a cooking book. In general, we can assume that a text is dealing with a limited number of topics, which will determine the probability of each word of appearing the text. It is easy to map this framework into a latent variable model perspective, where the topics are the latent variables, and the observable features are the words, which we can represent as discrete random variables whose distribution is determined by the value of the latent topic.

This approach is the one followed by most of the topic models that can be found in the literature. The simplest model is probably the single-topic model (Hofmann, 2017), where a text is assumed to deal with a single topic that determines the probability of the various words of the vocabulary to appear in the text. This model makes the additional assumption that the words are exchangeable, in the sense that each word is independent from the other words appearing in the text and generated by a discrete distribution only dependent on the value of the latent topic. Despite its simplicity, the single-topic model is a powerful tool to describe texts – especially if short, and likely to deal with a unique topic – and also allows to perform text clus-

tering, grouping together texts dealing with the same latent topic. However, the single-topic hypothesis is commonly considered an over-simplification and more powerful models allowing texts to deal with multiple topics have emerged. In this direction, one of the most popular models is the Latent Dirichlet Allocation (LDA) (Blei et al., 2003). With LDA, each word of a text is sampled by a unique topic, but different words may be sampled from different topics allowing in this way the text to deal with multiple subjects. After its introduction, LDA evolved in several directions, allowing for example the topics to evolve with the time (Blei and Lafferty, 2006), or allowing forms of correlations between the words appearing in a text (Blei and Lafferty, 2005).

Topic models are probabilistic graphical models with a pretty simple structure. Nevertheless, their learning has always been considered a hard task; Arora et al. (2012) for example proved that maximizing the likelihood of the single-topic model is an NP-hard problem. The most popular approach in general consists in using approximate inference approaches aiming at approximately maximizing the likelihood function of a topic model, recovering in this way a reasonably good (but not provably optimal) model. Variational methods for example (Blei et al., 2003) are used for learning LDA; here a convex relaxation of the likelihood is provided and maximized with a variation of Expectation Maximization. More recently, approaches based on Gibbs sampling and monte Carlo methods emerged (Griffiths and Steyvers, 2004) and established as standard learning techniques. However, these methods suffer of bad computational performances, and tend to scale poorly when the model dimensions or the sample size are big.

In this scenario, methods of moments emerged as a powerful alternative. In fact, the scalability properties of these methods together with their provable guarantees of learning accuracy are characteristics that approximate inference approaches do not have. A first approach was presented by Anandkumar et al. (2012b) who introduced a simple technique to learn the single-topic model. Some of the same authors presented then extensions to LDA (Anandkumar et al., 2012a) and to the correlated topic model (Arabshahi and Anandkumar, 2017). All these approaches follow the learning strategy outlined in Chapter 1 – retrieving unbiased moment estimators that are then decomposed to retrieve the parameters of the model – and represent a promising alternative to the likelihood-based approaches outlined above. However, despite the theoretical soundness of these works, a little attention is paid to make them robustly

applicable in real-world contexts. For example, the estimators provided by Anandkumar et al. (2012a) and Anandkumar et al. (2012b) are not usable in practice, because they are extremely sensitive to the noise. More robust estimators have been introduced by Zou et al. (2013), however without theoretically assessing their convergence rate, or experimentally testing their performance on real data.

In this chapter, we will consider two classic topic models, namely the single-topic model and LDA, we will provide improved estimators for their moments – in Section 3.2 for the single-topic model and in Section 3.3 for LDA – and present a theorem that relates their sample accuracy to the sample size and to the lengths of the documents. In Section 3.4 we will experimentally show that the proposed estimators outperform those from existing literature in terms of convergence speed. Section 3.4.2 will be dedicated to use the provided estimators, together with SVTD, to learn from real-world text corpora the topic models studied in this chapter and to compare their learning performance with those of traditional methods based on Markov-Chain Monte Carlo (MCMC); we will show that the proposed approach has significantly better performance in terms of both speed and quality of the learned model.

3.1 Learning Topic Models with Methods of Moments

In Chapter 1, we have seen an example of application of methods of moments to learn latent variable models. We have seen that for any model admitting a parametrization in terms of centers and weights (M, ω), the following two-steps approach allows to learn asymptotically converging estimators of the model parameters:

1. Recover a set of estimators $\tilde{M}_1$, $\tilde{M}_2$ and $\tilde{M}_3$ whose expectation exhibit

the following relations with the model parameters:

$$\mathbb{E}[\tilde{M}_1] = M_1 = \sum_{i=1}^{k} \omega_i \mu_i, \tag{3.1}$$

$$\mathbb{E}[\tilde{M}_2] = M_2 = \sum_{i=1}^{k} \omega_i \mu_i \otimes \mu_i, \tag{3.2}$$

$$\mathbb{E}[\tilde{M}_3] = M_3 = \sum_{i=1}^{k} \omega_i \mu_i \otimes \mu_i \otimes \mu_i. \tag{3.3}$$

2. Plug the retrieved estimates into a decomposition algorithm – like for example SVTD from Chapter 2 – to retrieve from those estimates an estimated pair of model parameters $(\tilde{M}, \tilde{\omega})$.

In this chapter we will focus on learning topic models with methods of moments, concentrating on the single-topic model and on LDA, and following the two-steps approach described above. As the second step, namely the decomposition part, has been extensively treated in Chapter 2, we will focus here on the first part, the moment estimation procedure. Once we will have unbiased moment estimators, we will be able to retrieve asymptotically converging estimates of the model parameters, decomposing the moments for example with SVTD.

3.2 Moments Estimators for the Single-Topic Model

In this section we recall the single-topic model for which we introduce a new estimator of the moments; we then provide a sample accuracy bound for this estimator and we compare it with existing methods.

We consider a corpus of n text documents and a set of k topics; each document is deemed to belong to only one topic. The vocabulary appearing in the corpus consists of d words, from which it is immediate to label all the words of the vocabulary with a number between 1 and d. The generative process works as follows:

- First, a (hidden) topic $Y \in \{1, ..., k\}$ is drawn, according to a given probability distribution; we define, for any $j \in \{1, ..., k\}$ the probability of drawing the topic j as follows:

$$\omega_j = \mathbb{P}(Y = j), \text{ and } \omega = (\omega_1, \ldots, \omega_k)^\top.$$

- Once the topic has been chosen, all the words of the documents are generated according to a multinomial distribution; for each $i \in \{1, ..., d\}$, $\mu_{j,i}$ will be the probability of generating the word i under topic j:

$$\mathbb{P}(\text{Drawing word } i | Y = j) = \mu_{j,i}, .$$

  Also we will denote by $\mu_j = (\mu_{j,1}, ..., \mu_{j,d})$ the vectors containing the probabilities of the various words to appear under topic j. It is a common practice to identify a topic with the probability distribution of the words under that topic, i.e. with each of the vectors $\mu_1, \ldots, \mu_k$. We will denote with M the matrix whose columns coincide with the topics: $M = [\mu_1|, ..., |\mu_k]$.

A practical encoding of the words in a document consists of identifying the tth word of a text with T words with a $d-$tuple $X_t \in \mathbb{R}^d$, defined as:

$$(X_t)_h = \begin{cases} 1 & \text{if the word is } h, \\ 0 & \text{otherwise.} \end{cases}$$

In fact, if X_t is the tth word of a document of T words, we can define a vector X whose coordinate h represents the number of times the word h has appeared in the document: $X = \sum_{t=1}^{T} X_t$. It is common to call X a *bag of words* representation of a document. We can see that, if the topic is j, each coordinate of X is distributed as a binomial distribution with parameter T and $\mu_{j,i}$:

$$Distr((X)_i | Y = j) = B(T, \mu_{j,i}).$$

Remark 3.2.1. The single-topic model is a powerful tool for text mining; assume to have a corpus of texts $\mathcal{X}$ and to have a single-topic model with parameters (M, ω) that accurately describes it. Then, it is possible to infer, for each text x of the corpus, the latent topic that best describes that text, using the following formula:

$$\mathbb{P}(Y = j | X = x) = \frac{\mathbb{P}(X = x | Y = j)\, \omega_j}{\sum_{i=1}^{k} \mathbb{P}(X = x | Y = i)\, \omega_i}. \tag{3.4}$$

This formula also enables us to cluster the texts of a corpus, grouping together those that deal with the same topic.

It is immediate to observe that the single-topic model admits a parametrization similar to the one provided in Chapter 1 for naive Bayes models, in terms of a vector of weights ω and a matrix of centers (here, the topics) M. We now consider a corpus of n documents

$$\mathcal{X} = \{x^{(1)}, ..., x^{(n)}\}$$

where each vector $x^{(i)}$ is a bag of words representation of the document i, whose length is t_i:

$$x^{(i)} = \sum_{t=1}^{t_i} x_t^{(i)}.$$

Any learning algorithm for the single-topic model will aim at recovering from a dataset $\mathcal{X}$, an estimate of the parameter (M, ω) of a single topic model that is approximately generating the data. We know from Chapter 1, that, if we assume the matrix of the topics M to be full rank, it will be enough to recover a set of estimators $\tilde{M}_1$, $\tilde{M}_2$ and $\tilde{M}_3$ whose expectation will respect the relations described at Equations 3.1,3.2 and 3.3. The following theorem provides these estimators.

Theorem 3.2.1. *Fix a value $n \in \mathbb{N}$, and let $x^{(1)}, ..., x^{(x)}$ be n sample documents generated according to a single-topic model with parameters (M, ω). Define the vector $\tilde{M}_1 \in \mathbb{R}^d$, and the symmetric tensors $\tilde{M}_2 \in \mathbb{R}^{d \times d}$ and $\tilde{M}_3 \in \mathbb{R}^{d \times d \times d}$ whose entries are as follows, for $h \leq l \leq m$:*

$$\begin{aligned}
(\tilde{M}_1)_h =& \frac{\sum_{i=1}^n (x^{(i)})_h}{\sum_{i=1}^n t_i}, \\
(\tilde{M}_2)_{h,l} =& \frac{\sum_{i=1}^n (x^{(i)})_h (x^{(i)} - \chi_{h=l})_l}{\sum_{i=1}^n (t_i - 1) t_i}, \\
(\tilde{M}_3)_{h,l,m} =& \chi_{h<l<m} \frac{\sum_{i=1}^n (x^{(i)})_h (x^{(i)})_l (x^{(i)})_m}{\sum_{i=1}^n (t_i - 2)(t_i - 1) t_i} \\
&+ \chi_{(h=l<m)\vee(h<l=m)} \frac{\sum_{i=1}^n (x^{(i)})_h (x^{(i)} - 1)_l (x^{(i)})_m}{\sum_{i=1}^n (t_i - 2)(t_i - 1) t_i} \\
&+ \chi_{h=l=m} \frac{\sum_{i=1}^n (x^{(i)})_h (x^{(i)} - 1)_l (x^{(i)} - 2)_m}{\sum_{i=1}^n (t_i - 2)(t_i - 1) t_i},
\end{aligned}$$

where χ is the indicator function and t_i is the number of words appearing in document i. Then, the estimators $\tilde{M}_1$, $\tilde{M}_2$ and $\tilde{M}_3$ respect the relations described at Equations 3.1,3.2 and 3.3.

Proof. We will prove the statements only for

$$\mathbb{E}\left(\frac{\sum_{i=1}^{n}(x^{(i)})_h(x^{(i)})_l}{\sum_{i=1}^{n}(t_i-1)t_i}\right)=\sum_{j=1}^{k}\omega_j\mu_{j,j}\mu_{j,l}.$$

Similar arguments hold for the other equations. It is easy to see, by conditional independence, that

$$E((x^{(i)})_h(x^{(i)})_l)=\sum_{j=1}^{k}\omega_j E((x^{(i)})_h(x^{(i)})_l|Y=j)$$

but the conditioned $(x^{(i)})_h$ and $(x^{(i)})_l$ are components of a multinomial distribution and so

$$\sum_{j=1}^{k}\omega_j E((x^{(i)})_h(x^{(i)})_l|Y=j)=\sum_{j=1}^{k}\omega_j(t_i^2-t_i)\mu_{j,h}\mu_{j,l},$$

which implies the thesis.

□

Notice that, because M_2 and M_3 are symmetric, the previous theorem defines all the entries of these tensors. Given a sample, we are able to calculate the three estimators $\tilde{M}_1$, $\tilde{M}_2$ and $\tilde{M}_3$. It is immediate to see that $\lim_{n\to\infty}\tilde{M}_i=M_i$ for $i=1,2,3$.

The following theorem provides a sample accuracy bound for the estimators of Theorem 3.2.1.

Theorem 3.2.2. *Let $\tilde{M}_2$ and $\tilde{M}_3$ be the empirical estimates of M_2 and M_3 obtained using Theorem 3.2.1, and let $(t_1,...,t_n)$ be a vector containing the lengths of the various documents of the corpus. Define*

$$T_1=\sum_{i=1}^{n}t_i,\quad T_2=\sum_{i=1}^{n}(t_i-1)t_i,\quad T_3=\sum_{i=1}^{n}(t_i-2)(t_i-1)t_i,$$

$$W_2^{(n)}=\frac{\sum_{i=1}^{n}(t_i(t_i-1))^2}{T_2^2},\quad W_3^{(n)}=\frac{\sum_{i=1}^{n}(t_i(t_i-1)(t_i-2))^2}{T_3^2}.$$

Then, for any pair $\epsilon > 0$ *and* $0 \leq \delta < 1$*, such that*

$$\sqrt{W_2^{(n)}(1 - ||M_2||_F^2)} + \sqrt{2W_2^{(n)} \log(\frac{1}{\delta})} < \epsilon,$$

it holds that

$$\mathbb{P}(||M_2 - \tilde{M}_2||_F < \epsilon) > 1 - \delta.$$

Also, for any pair $\epsilon > 0$ *and* $0 \leq \delta < 1$*, such that*

$$\sqrt{W_3^{(n)}(1 - ||M_3||_F^2)} + \sqrt{2W_3^{(n)} \log(\frac{1}{\delta})} < \epsilon,$$

it holds that

$$P(||M_3 - \tilde{M}_3||_F < \epsilon) > 1 - \delta.$$

Remark 3.2.2. We briefly comment on the meaning of the theorem by focusing on M_2 (similar arguments hold for M_3) and analyzing the case where all the documents have the same length t ($t_i = t$ for all i). Then the bound simplifies to:

$$\sqrt{\frac{1}{n}(1 - ||M_2||_F^2)} + \sqrt{\frac{1}{n} \log(\frac{2}{\delta})}.$$

It is interesting to notice that the worst-case accuracy of the bound is $\epsilon = O(1/\sqrt{n})$. An interpretation for this fact is that, even if the documents in the corpus are very long (large t), it is impossible to accurately learn the model with just a few samples, as in particular we may not even see all the topics.

Remark 3.2.3 (Alternative estimates of the moments)**.** The simplest estimation of the low-order moments for the single-topic model is provided by Anandkumar et al. (2012b), who, for each document $i \in \{1, ..., n\}$ consider three randomly selected words, $x_1^{(i)}, x_2^{(i)}, x_3^{(i)}$, and then show that

$$\frac{\sum_{i=1}^n (x_1^{(i)})_h}{n} \xrightarrow[n\to\infty]{} (M_1)_h, \quad \frac{\sum_{i=1}^n (x_1^{(i)})_h (x_2^{(i)})_l}{n} \xrightarrow[n\to\infty]{} (M_2)_{h,l},$$

$$\frac{\sum_{i=1}^n (x_1^{(i)})_h (x_2^{(i)})_l (x_3^{(i)})_m}{n} \xrightarrow[n\to\infty]{} (M_3)_{h,l,m}.$$

This method only uses three words per document, and has mainly illustrative purposes, as noticed by the authors. A method more similar to the one

proposed in Theorem 3.2.1 is described by Zou et al. (2013). Both the method by Zou et al. (2013) and that in Theorem 3.2.1 average the estimators with the document lengths, taking into consideration all the words in each document. However, in the method by Zou et al. (2013), the averaging is done for each document and then they are averaged together with the same weight; for example, Zou et al. (2013) calculate the off-diagonal entries of $\tilde{M}_2$ as follows:

$$(\tilde{M}_2)_{h,l} = \frac{1}{n}\sum_{i=1}^{n}\frac{(x^{(i)})_h(x^{(i)})_l}{(t_i-1)t_i}.$$

This calculation is a simple average of many estimators giving the same weight to all documents, namely $1/n$. Instead, in Theorem 3.2.1, we propose the following different formula:

$$(\tilde{M}_2)_{h,l} = \frac{\sum_{i=1}^{n}(x^{(i)})_h(x^{(i)})_l}{\sum_i (t_i-1)t_i} = \sum_{i=1}^{n}\frac{(x^{(i)})_h(x^{(i)})_l}{(t_i-1)t_i}\frac{(t_i-1)t_i}{\sum_j (t_j-1)t_j}.$$

We can see that here we perform a weighted average, where the weight of the sample i is $\frac{(t_i-1)t_i}{\sum_j (t_j-1)t_j}$, giving in practice more weight to longer documents, which are supposed to be the most reliable; we will experimentally see in Section 3.4 that the proposed approach is less sensitive to noise, leading to improved results. If all the documents have the same length, the two estimates will produce the same number.

3.2.1 Proof of Theorem 3.2.2

We want to express the elements of the matrix $\tilde{M}_2 - M_2$ in an appropriate way and then express a bound using McDiarmid's inequality (McDiarmid 1998 and Lemma 3.2.1). We consider the set of all the documents in a corpus:

$$\tilde{\mathcal{X}} = (x^{(1)}, ..., x^{(n)}).$$

It is easy to see that $\tilde{M}_2$ can be expressed as a function of $\tilde{\mathcal{X}}$:

$$\tilde{M}_2(\mathcal{X}) = \sum_{i=1}^{n} w_i \tilde{M_2}^{(i)} \tag{3.5}$$

where $w_i = \frac{t_i(t_i-1)}{T_2}$, t_i is the length of the ith document, $T_2 = \sum_{i=1}^{n} t_i(t_i-1)$, and $\tilde{M_2}^{(i)}$ are independent matrices defined as follows:

$$(\tilde{M_2}^{(i)})_{(u,v)} = \frac{(x^{(i)})_u(x^{(i)} - \chi_{u=v})_v}{t_i(t_i-1)}. \tag{3.6}$$

Notice that, for any i, $\mathbb{E}(\tilde{M_2}^{(i)}) = M_2$. We now define the following function:

$$\Phi(\tilde{\mathcal{X}}) = ||\tilde{M_2}(\tilde{\mathcal{X}}) - M_2||_F$$

and observe that, given two samples differing only by one element

$$\tilde{\mathcal{X}} = (x^{(1)}, ..., x^{(l)}, ..., x_i^{(n)})$$

$$\tilde{\mathcal{X}}' = (x^{(1)}, ..., x^{(l)'}, ..., x_i^{(n)})$$

we obtain the following inequality:

$$|\Phi(\tilde{\mathcal{X}}) - \Phi(\tilde{\mathcal{X}}')| \leq ||\tilde{M_2}(\tilde{\mathcal{X}}) - \tilde{M_2}(\tilde{\mathcal{X}}')||_F =$$

$$= ||w_l(\tilde{M_2}^{(l)} - \tilde{M_2}^{(l)'})||_F$$

where $\tilde{M_2}^{(l)'}$ is calculated plugging $x^{(l)'}$ in formula (3.6). Now, observing that $||\tilde{M_2}^{(l)}||_F \leq 1$ and $||\tilde{M_2}^{(l)'}||_F \leq 1$, we obtain:

$$||w_l(\tilde{M_2}^{(l)} - \tilde{M_2}^{(l)'})||_F \leq 2w_l$$

The inequality above enables us to apply McDiarmid's inequality, stating that

$$\mathbb{P}(||\tilde{M_2} - M_2||_F > \mathbb{E}(||\tilde{M_2} - M_2||_F) + \epsilon) \leq e^{-\frac{\epsilon^2}{2\sum_{i=1}^n w_i^2}} = e^{-\frac{\epsilon^2}{2W_2^{(n)}}} = e^{-t^2},$$

where we used the notation $W_2^{(n)} = \sum_{i=1}^n w_i^2$, and defined

$$t = \frac{\epsilon}{\sqrt{2W_2^{(n)}}}.$$

We now provide a bound for $\mathbb{E}(||\tilde{M_2} - M_2||_F)$. Using Jensen's inequality we have

$$\mathbb{E}(||\tilde{M_2} - M_2||_F) \leq \sqrt{\mathbb{E}(||\tilde{M_2} - M_2||_F^2)} =$$

$$= \sqrt{\mathbb{E}(||\tilde{M_2}||_F^2) - ||M_2||_F^2} = \sqrt{\sum_{u,v} \mathbb{E}((\sum_{i=1}^n w_i(\tilde{M_2}^{(i)})_{(u,v)})^2) - ||M_2||_F^2}$$

where we have used Equation (3.5). This last term is equal to

$$\sqrt{\sum_{u,v} \mathbb{E}(\sum_{i=1}^n w_i^2(\tilde{M_2}^{(i)})_{(u,v)}^2) + \sum_{u,v} \mathbb{E}(\sum_{i \neq j} w_j w_i(\tilde{M_2}^{(i)})_{(u,v)}(\tilde{M_2}^{(j)})_{(u,v)}) - ||M_2||_F^2}$$

and using the fact that $\mathbb{E}(\tilde{M_2}^{(i)}_{(u,v)}\tilde{M_2}^{(j)}_{(u,v)}) = (M_2)^2_{(u,v)}$, this equals

$$\sqrt{\sum_{i=1}^{n} w_i^2 \mathbb{E}(||\tilde{M_2}^{(i)}||_F^2) + \sum_{i \neq j} w_j w_i ||M_2||_F^2 - ||M_2||_F^2}\,.$$

Now using that $||\tilde{M_2}^{(i)}||_F \leq 1$, we can bound this from above by

$$\sqrt{\sum_{i=1}^{n} w_i^2 + ||M_2||_F^2(\sum_{i \neq j} w_j w_i - 1)} = \sqrt{\sum_{i=1}^{n} w_i^2(1 - ||M_2||_F^2)}.$$

where in the last equality we used the fact that $\sum_{i \neq j} w_j w_i = 1 - \sum_{i=1}^{n} w_i^2$. So, using the notation $W_2^{(n)} = \sum_{i=1}^{n} w_i^2$, we have $\mathbb{E}(||\tilde{M_2} - M_2||_F) \leq \sqrt{W_2^{(n)}(1 - ||M_2||_F^2)}$, from which we obtain

$$\mathbb{P}(||\tilde{M_2} - M_2||_F > \sqrt{W_2^{(n)}(1 - ||M_2||_F^2)} + t\sqrt{2W_2^{(n)}}) \leq e^{-t^2}.$$

In conclusion, we can state that if $e^{-t^2} = \delta$ we get, for any $\delta \in (0, 1]$

$$\mathbb{P}(||\tilde{M_2} - M_2||_F > \epsilon) \leq \delta$$

where

$$\epsilon = \sqrt{W_2^{(n)}(1 - ||M_2||_F^2)} + \sqrt{2W_2^{(n)} \log(\frac{1}{\delta})}$$

A similar argument works for M_3.

□

Lemma 3.2.1 (McDiarmid's inequality, McDiarmid 1998.)**.** *Let $X_1, ..., X_m$ be independent random variables all taking values in the set $\mathcal{C}$. Furthermore, let $f : \mathcal{C}^m \to \mathbb{R}$ be a function of $X_1, ..., X_m$ satisfying for all i and for all $x_1, ..., x_m, x_i' \in \mathcal{C}$,*

$$|f(x_1, ..., x_i, ..., x_m) - f(x_1, ..., x_i', ..., x_m)| \leq c_i.$$

Then for all $\epsilon > 0$,

$$\mathbb{P}(f - E[f] \geq \epsilon) \leq exp(-\frac{2\epsilon^2}{\sum_{i=1}^{m} c_i^2}).$$

3.3 Extension to Latent Dirichlet Allocation

In this section we extend the results of the previous section to a more complex latent variable model, the latent Dirichlet allocation or, for short, LDA.

The obvious criticism of the single-topic model is that each document can deal with a unique topic, a hypothesis that is commonly considered unrealistic. To overcome this issue, more complex models have been introduced, one of them being LDA (Griffiths and Steyvers, 2004; Blei et al., 2003). In its simplest form, LDA assumes that each document deals with a multitude of topics, in proportions that are governed by the outcome of a Dirichlet distribution. More precisely, considering our text corpus with n documents with a vocabulary of d words, the generative process for each text is the following:

- First a vector of topic proportions is drawn from a Dirichlet distribution with parameter $\alpha \in \mathbb{R}_+^k$, $Dir(\alpha)$; we recall that Dirichlet distribution is distributed over the simplex

$$\Delta^{k-1} = \{v \in \mathbb{R}^k : \forall i, v_i \in [0,1],\ and\ \sum v_i = 1\},$$

  and has the following density function, for $h \in \Delta^{k-1}$:

$$\mathbb{P}(h) = \frac{\Gamma(\alpha_0) \prod_{i=1}^k h_i^{\alpha_i - 1}}{\prod_{i=1}^k \Gamma(\alpha_i)}.$$

  Where $\alpha_0 = \sum \alpha_i$. From a practical point of view, this step consists of drawing a vector of parameters $h \in \Delta^{k-1}$ such that h_i represents the proportion of the topic i in the document.

- Once the topic proportions (also named *mixture of topics*) have been chosen, each word of the document is generated according to the following procedure: first a (hidden) topic of the word, say $Y \in \{1, ..., k\}$, is drawn, according to the probabilities defined by h (so we will have probability h_j of drawing topic j) and then we will generate the word itself according to a multinomial distribution; for each $i \in \{1, ..., d\}$, $\mu_{j,i}$ will be the probability of generating the word i under the topic j. Again, we will denote with $\mu_j = (\mu_{j,1}, ..., \mu_{j,d})$ the topics of the model, and we will store them as columns of a matrix $M = [\mu_1|, ..., |\mu_k]$.

Keeping the notation used in the previous section, we will write $x_j^{(i)} \in \mathbb{R}^d$ for the coordinate vector indicating the word at position j in document i, $x^{(i)} = \sum x_j^{(i)}$ for the word-count vector of document i, and $t_i = \sum_{j=1}^{n} (x^{(i)})_j$ for the number of words in that document. In the case of LDA, the unknown model parameters are the pair (M, α).

As in the case of the single-topic model, we want to manipulate the observable moments in order to obtain a set of symmetric low-rank tensors. The following theorem is an immediate modification of the one presented by Anandkumar et al. (2012a, Lemma 3.2), and relates the observable moments of the known variables with the unknowns (M, α), providing the required representation. The only modification consists of the fact that we have used the estimates of Theorem 3.2.1 instead of the standard ones of Remark 3.2.3.

Theorem 3.3.1. *Let $\tilde{M}_1, \tilde{M}_2$ and $\tilde{M}_3$ be the empirical estimates defined in Theorem 3.2.1. Define*

$$\tilde{M}_2^{\alpha} = \tilde{M}_2 - \frac{\alpha_0}{\alpha_0 + 1} \tilde{M}_1 \otimes \tilde{M}_1,$$

$$\tilde{M}_3^{\alpha} = \tilde{M}_3 - \frac{\alpha_0}{\alpha_0 + 2} (M_{1,2}) + \frac{2\alpha_0^2}{(\alpha_0 + 2)(\alpha_0 + 1)} \tilde{M}_1 \otimes \tilde{M}_1 \otimes \tilde{M}_1,$$

where $M_{1,2} \in \mathbb{R}^{d \times d \times d}$ is a three-dimensional tensor such that

$$(M_{1,2})_{h,l,m} = ((\tilde{M}_2)_{h,l} (\tilde{M}_1)_m + (\tilde{M}_2)_{l,m} (\tilde{M}_1)_h + (\tilde{M}_2)_{m,h} (\tilde{M}_1)_l).$$

Then

$$\mathbb{E}[\tilde{M}_2^{\alpha}] = M_2^{\alpha} = \sum_{i=1}^{k} \frac{\alpha_i}{(\alpha_0 + 1)\alpha_0} \mu_i \otimes \mu_i,$$

$$\mathbb{E}[\tilde{M}_3^{\alpha}] = M_3^{\alpha} = \sum_{i=1}^{k} \frac{2\alpha_i}{(\alpha_0 + 2)(\alpha_0 + 1)\alpha_0} \mu_i \otimes \mu_i \otimes \mu_i.$$

This technique allows to express the observable moments in the form of a symmetric tensor. Both M_2^{α} and M_3^{α} have symmetric-rank less than or equal to k, and so we can use any tensor decomposition algorithm to retrieve from them the unknown model parameters (M, α).

Remark 3.3.1 (Advantages of our approach)**.** The approach outlined above has various advantages over traditional methods based on MCMC algorithms,

like the one presented by Griffiths and Steyvers (2004). While MCMC methods require several passes over the data, our method only uses one pass to compute the moments described in Theorem 3.3.1. This guarantees higher efficiency, which is confirmed by our experiments in Section 3.4.2. Furthermore, if the tensor decomposition method applied to the moments is robust, and provably recovers (M, α) from the moments, then the proposed approach will be guaranteed to recover the parameters of the model generating the data, guarantees that do not seem to exist for MCMC approaches.

Remark 3.3.2 (Inference)**.** Similarly to the single-topic model, one of the main usages of LDA is to infer the mixture of hidden topics of each document in a corpus. Unfortunately, an exact formula to perform this inference is not known, but a number of approximate approaches exist, like Gibbs sampling (Griffiths and Steyvers, 2004; Newman et al., 2009) and Expectation Propagation (Blei et al., 2003). In our case, if we assume to know the values of model parameters (M, α), we can apply a modified Gibbs sampler to infer the topic mixture for a given text; consider a text, whose words are $x_1, ..., x_t$; then, in LDA, each word x_i is generated by a unique topic Y_{x_i}. Using the equations for Gibbs sampling from Griffiths and Steyvers (2004), if Y_{x_i} is the hidden topic of word x_i and y_{-x_i} is the set of topic assignment for all the words in the document excluded x_i, it can be shown that

$$\mathbb{P}(Y_{x_i} = j | y_{-x_i}, x_i) \approx \mu_{j,x_i} \frac{n_{-i,j} + \alpha_i}{t - 1 + \alpha_0}, \tag{3.7}$$

where $n_{-i,j}$ is the number of words assigned to topic j excluding x_i, t is the total number of words in the document and μ_{j,x_i} is the probability of drawing the word x_i under topic j. So, given a document, first we have to assign to each word a hidden topic, and then update this assignment word by word in a iterative way, using a Monte-Carlo assignment governed by equation (3.7). Each iteration updates the number of words assigned to a given topic; after a suitable number of iterations, we can estimate the topic mixture for a given document as the vector $h \in \mathbb{R}^k$ such that $h_j = \frac{n_j + \alpha_i}{t + \alpha_0}$, where n_j is the number of words assigned to topic j.

3.4 Experiments

In this section we will study experimentally, the learning performance of methods of moments for the single-topic model and for LDA. We will begin

by experimentally studying the convergence rate of the estimators described in section 3.2, comparing it with that of existing moments estimators for the single topic model. Section 3.4.2 instead will focus on the end-to-end learning of topic models from real-world text corpora. In particular, we will use the moments estimators provided in this chapter together with SVTD from Chapter 2 to learn from real-world corpora single-topic models and LDA. The quality of the learned models will be assessed against that of standard MCMC approach (Griffiths and Steyvers, 2004).

Experimental setting: We use here the same experimental setting described in Chapter 2. Additionally, the implementation of LDA based on MCMC that we use in Section 3.4.2 is based on the open-source *LDA* python package.[1].

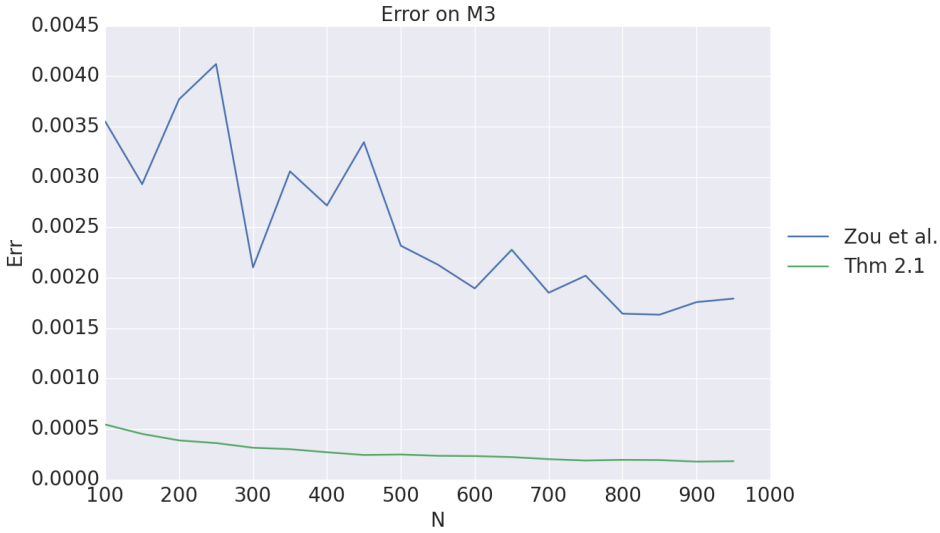
3.4.1 Recovering M_2 and M_3

In Section 3.2 we described new estimators to recover the matrix M_2 and tensor M_3 from a sample, comparing it with the methods presented in the state of the art literature from Zou et al. (2013), outlined in Remark 3.2.3. In this section we compare, using synthetically generated data, how well the two different methods recover M_2 and M_3 as a function of the sample size. To perform this experiment, we generated a set of 1000 synthetic corpora according to the single-topic model described in Section 3.2, with different sizes (the number of texts for each corpus); the smallest corpus contained 100 texts, the largest 10000; each text contained a random number of words, from a minimum of 3 to a maximum of 100. For each corpus, the values of the unknowns (M, ω) have been randomly generated by sampling a uniform distribution and normalizing both the columns of M and the vector ω to have sum 1. From them, we have been able to obtain the theoretical values of M_2 and M_3 using equations (1.2) and (1.3) and to compare those values with the one empirically estimated from data using the equations in Theorem 3.2.1 for the presented method and the method from Zou et al. (2013) for the competing one. Results appear in Figure 3.1, where we show how the estimated M_2 and M_3, namely $\tilde{M}_2$ and $\tilde{M}_3$, approach the theoretical values; in particular, the chart presents the errors

$$Err_2 = ||\tilde{M}_2 - M_2||_F \quad \text{and} \quad Err_3 = ||\tilde{M}_3 - M_3||_F$$

[1]Link to the code: `https://github.com/lda-project/lda`

(a)



(b)

Figure 3.1: The $x-$axis of the figures represents the sizes of the synthetic text corpora, while the $y-$axis is Err_2 for the top chart and Err_3 for the bottom chart. Green lines represent the errors obtained with the method presented in Theorem 3.2.1, while blue lines represent the errors obtained with the method by Zou et al. (2013) on the same corpora.

.

as a function of the sample size n used to find $\tilde{M}_2$ and $\tilde{M}_3$. We can see that the method in Theorem 3.2.1 outperforms the state-of-the-art technique; this is because it gives more weight to longer documents, where the signal is stronger, and less to shorter ones, where the signal is noisier.

3.4.2 Real Data

In the previous section, we have seen that the moment estimators provided in this chapter present superior convergence properties in comparison with those from existing literature. In this section we use these estimators in conjunction with SVTD from Chapter 2 to learn topic models on real world text corpora. In particular, we will consider two corpora: Dante's "*Divina Commedia*" and the *State of the Union* addresses from 1945 to 2005. In both cases, we will learn a single-topic model – using the moment estimates from Section 3.2 and SVTD as a decomposition engine – and an LDA – using the moment estimates from Section 3.3 and again SVTD for the decomposition – and compare quantitatively and qualitatively their behavior. Additionally, we will compare these approaches with a standard method to learn LDA based on Markov chain Monte Carlo (MCMC) approach, described by Griffiths and Steyvers (2004).

Dante's Divina Commedia

Dante's "Divina Commedia" (Alighieri, 1979) is an Italian epic poem written in the first half of the 14th century.[2] It narrates the imaginary trip of the main character, Dante himself, in the afterlife, guided by Virgilio, the famous Latin poet, and Beatrice, a Florentine woman that inspired most of Dante's works. The storyline represents an allegorical description of death soul's journey towards God according to medieval world view. It begins with Dante's travel trough the "Inferno" (Hell), where damned souls are deemed to eternal punishment according to their sins; the journey then moves to "Purgatorio", a seven-level mountain, where, at each level, a capital sin (sins less serious than those punished in Hell) is allegorically described; here souls are discounting their punishment, before finally moving to "Paradiso", Heaven, which is visited by Dante in the last third of the book. The book is divided into 100 chapters: 34 for Hell, 33 for Purgatory and 33 for Heaven.

[2]The full text can be found here: `http://www.gutenberg.org/files/1012/1012-0.txt`

We analyze the performance of methods of moments on this dataset, using both the single-topic model and LDA (setting $\alpha_0 = 0.2$), employing SVTD as decomposition engine and fixing the vocabulary size to $d = 1820$ words, which corresponds to the 70% most frequent ones.[3] Also, we experimentally compare our results with those of a classic approach to learn LDA described by Griffiths and Steyvers (2004) based on MCMC — setting as a stopping criterion 2000 iterations of the sampler (which is the default value provided by the Python package we used). As a first step, we study how the quality of the results depends on the number of topics k requested to the algorithm. To do this, we use the various competing methods to learn a set of model parameters (M, ω), for all the values of k between 2 and 32. For each returned pair of parameters, we evaluate the running time of the algorithm and the average topic coherence across the various topics, displaying the results in Figure 3.2 ("ST SVTD" and "LDA SVTD" represents the results of the single-topic model and LDA learned with the proposed method of moments and "LDA MC" represents the results of LDA learned with MCMC). The *coherence* (Mimno et al., 2011) of a topic in a corpus is a quantitative indicator of the quality of a topic, and evaluates how much pairs of words that are highly probable in a topic tend to co-occur in the texts of the corpus; a good model is expected to generate topics with high coherence scores. Formally, the coherence of a topic μ in a corpus is defined as

$$Coherence(\mu) = \sum_{j=2}^{L} \sum_{i=1}^{j-1} \log \frac{D(w_i, w_j) + 1}{D(w_i)}$$

where $(w_1, ..., w_L)$ is the list of the $L = 20$ most frequent words in topic μ, $D(w_i)$ (resp. $D(w_i, w_j)$) is the count of documents containing the word w_i (resp. w_i and w_j). In Figure 3.2, we can see that the proposed method of moments outperforms MCMC in terms of speed and coherence. Furthermore, while the performance of moment-based method remains good as model grows, the coherence of MCMC degrades strongly.[4] For method of moments, the single-topic model and LDA have similar performance in terms of coherence

[3] We tested the same experiment with different vocabulary sizes — like 80% and 90% of the most frequent words — obtaining nearly identical topics.

[4] We tried to improve the quality of the topics obtained with MCMC by allowing for further iterations, but the increased running time yielded no improvements in the coherence results.

(left chart), while the running times of LDA are larger (right chart). From the coherence chart, we can see that $k \in \{2, 3, 4\}$ are all good numbers of topics for the moment-based approaches; so, we keep $k = 3$ and perform two additional analysis, focusing only on the models learned with SVTD and method of moments. First, we plot in Figure 3.3 the topic assignment for the various texts of the corpus, with the single-topic model (top figure) using Equation (3.4) for the assignment, and for LDA (bottom figure), using the approach described in Remark 3.3.2. The $x-$axis represents the chapter of the book: the first 34 are Hell, 35 to 67 are Purgatory, and the last 33 are Heaven, while the areas represent how much each chapter belongs to each of the topics. It is interesting to observe that, both with the LDA and single-topic models, each of the three topics is clearly dominant in one of the three areas of the corpus (Hell, Purgatory and Heaven): topic 1, is clearly dominant in Purgatory's chapters, topic 2 in Hell, while topic 3 dominates Heaven. To see that this makes sense, we print in Table 3.1 the most relevant words for each topic, for the two models, where the *relevance* (Sievert and Shirley, 2014) is an indicator of how much a word characterizes a topic. Formally, the relevance of a word w with respect to a topic μ and is defined as

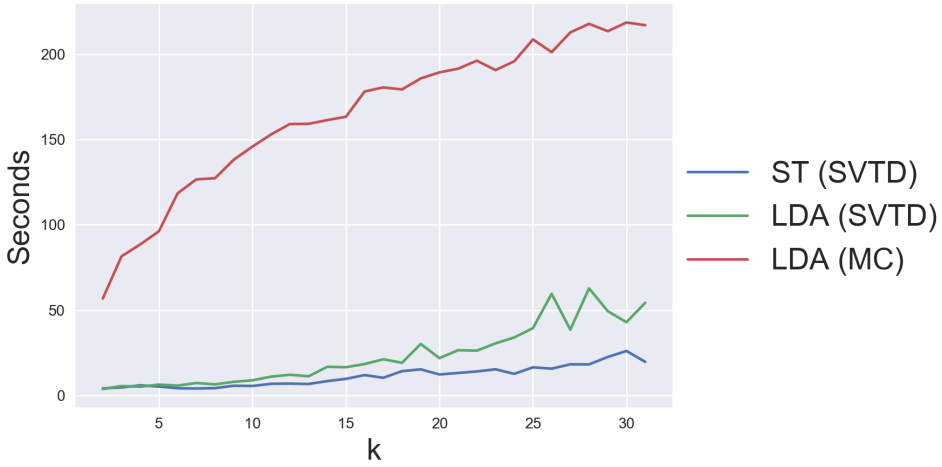
$$r(w, \mu) = \lambda \log \mathbb{P}(w|\mu) + (1 - \lambda) \log \frac{\mathbb{P}(w|\mu)}{\mathbb{P}(w)} \ , \tag{3.8}$$

where $\mathbb{P}(w|\mu)$ (resp. $\mathbb{P}(w)$) is the probability of sampling w with topic μ (resp. in the overall corpus) and the weight parameter was set to $\lambda = 0.7$. Again, the topics are similar between the LDA and single-topic models, and are coherent with the topic assignment results. For example, the most relevant words for topic 3 are *Cristo, luce, lume*, meaning Christ and light. The similarity of the topics between the LDA and single-topic models is expected from the analysis of the coherence, where the LDA and single-topic models gave very similar scores. An interpretation of this is the fact that Divina Commedia is close to following a single-topic model, with each chapter dealing with a single topic.

State of the Union Addresses

Every year, the president of United States of America presents a speech to a joint session of the United States Congress, where he outlines his governative agenda, the national priorities and legislative projects. We considered the set of $n = 65$ state of the union addresses presented between 1945 and 2005, and we perform the same analysis performed for the Divina Commedia corpus.

(a)



(b)

Figure 3.2: Figure 3.2a contains the topic coherence of a single-topic model and an LDA learned with method of moments (resp. ST SVTD and LDA SVTD) and an LDA learned with MCMC (LDA MC), in function of the number of topics k; Figure 3.2b contains an analysis of the running times. Both the figures refer to the *Divina Commedia* corpus.

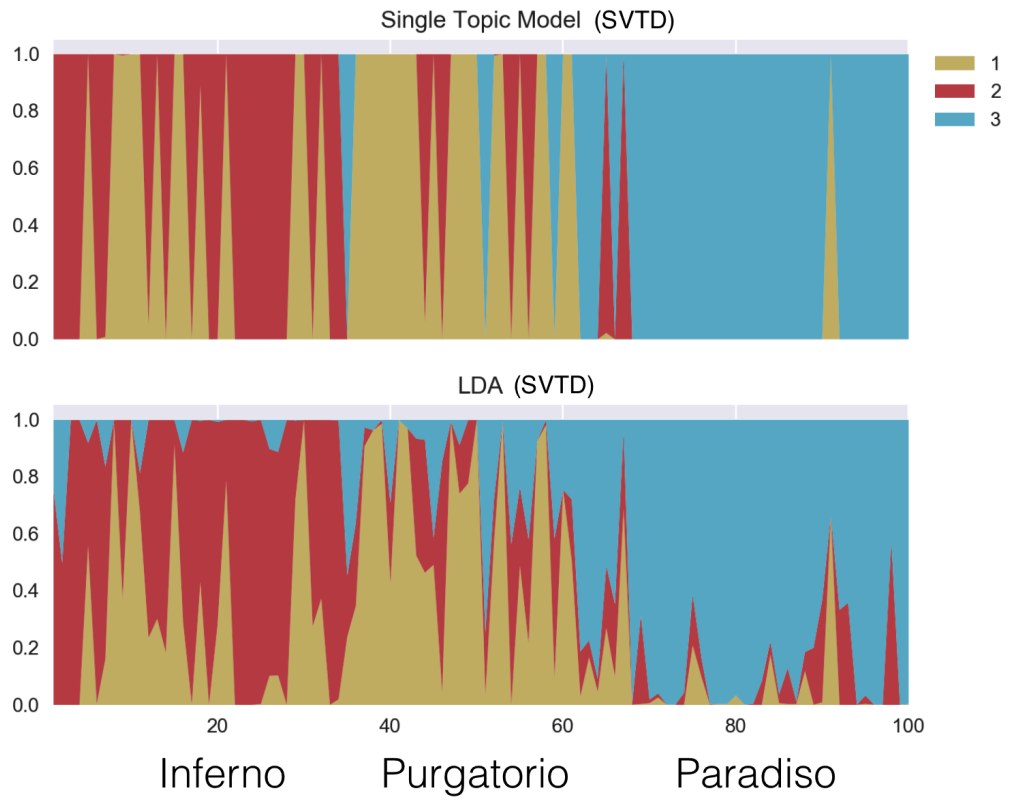


Figure 3.3: The topic assignment for the single-topic model and LDA learned with method of moments for the Divina Commedia.

single-topic model (SVTD)
Topic 1: disse, saprai, morta, torre, maestro *(Transl.) said, will know, dead, tower, master*
Topic 2: vidi, serpente, greve, maestro, sovra *(Transl.) I saw, snake, heavy, master, above*
Topic 3: luce, Cristo, creata, lume, vicine *(Transl.) light, Christ, created, light, near*

LDA (SVTD)
Topic 1: saprai, morta, torre, saggio, disse *(Transl.) will know, dead, tower, wise, said*
Topic 2: vidi, cotai, greve, serpente, maestro *(Transl.) I saw, so many, heavy, snake, master*
Topic 3: luce, creata, vicine, Cristo, lume *(Transl.) light, created, near, Christ, light*

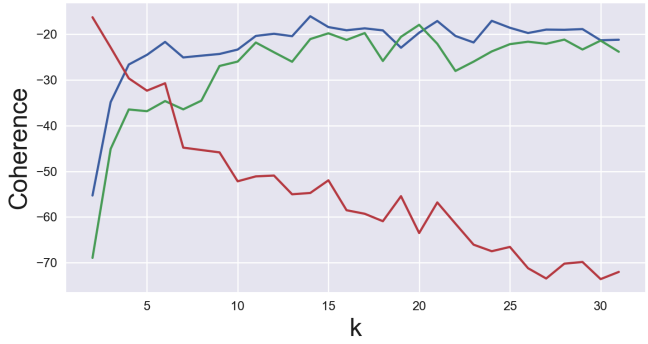
Table 3.1: The most relevant words for each topic for the *Divina Commedia*, learned by method of moments with SVTD setting $k = 3$, both for the single-topic model and for LDA.

Again, we keep the 70% most frequent words, obtaining a total vocabulary of $k = 1184$ words, and analyze topic coherence (Figure 3.4a) and running times (in Figure 3.4b) for method of moments with SVTD — using LDA ("LDA SVTD") and single-topic model ("ST SVTD") — and MCMC — only for LDA ("LDA MC"). Again, the performance of MCMC degrades as the model increases, while moment-based methods, both when learning LDA and single-topic models, keeps providing meaningful results, always using smaller running times. From now on we will only focus on the methods leaned with the moment-based approach.

Overall, the single-topic model presents a better coherence and smaller running times than LDA. We set $k = 18$, which is also good for both the models and present in Figure 3.4c the topic assignment for the two models, where the areas represent how much each document deals with each topic. Also, Table 3.2 presents the most relevant words for each topic according to the LDA and single-topic models. The topics make sense and look similar across the two models. However, unlike the Divina Commedia case, they are not identical. For example, topic 8 for the single-topic model looks like a mixture of topic 7 and 8 in LDA. Looking at the topic assignments in Figure 3.4c, we can see that speeches from the same president share similar topic assignments: for example, topic 15 is dominant for G.W. Bush, and mainly deals with terrorism and Iraq. Cold-war presidents have a strong predominance of topics involving the Soviet Union, space missions and the Vietnam war. The two models are often coherent, with some notable exceptions, like president Kennedy: LDA assigns him to a mixture of topics that properly describe the challenges of cold war (LDA topics: 12, 13, 17, 18), while the single-topic model provides a simpler characterization with speeches assigned to topics 18 (again a cold war topic) and 14 (cold war, with a focus on Europe and foreign politics).

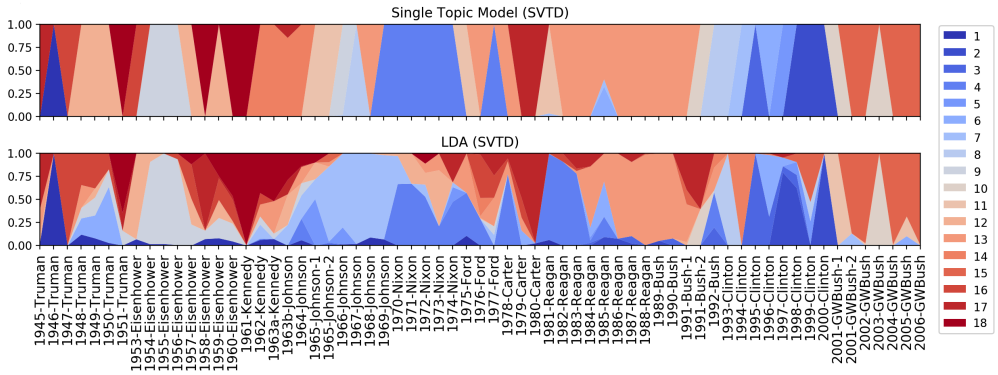
3.5 Conclusions

In this chapter, we have presented an application of methods of moments to learn two well-known topic models: the single-topic model and the LDA, analyzing the theoretical properties of the proposed approach in Sections 3.2 and 3.3. Furthermore, we have applied the proposed approach in conjunction with SVTD – presented in Chapter 2 – to learn the topics of two real-world text corpora, comparing its performance with that of the standard MCMC-



(a)

(b)



(c)

Figure 3.4: Figure 3.4a: the topic coherence of a single-topic model and an LDA learned with method of moments, and an LDA learned with MCMC, in function of the number of topics k; Figure 3.4b: an analysis of the running times. Figure 3.4c: the topic assignment for the two models learned by SVTD setting $k = 18$. All the figures refer to the *state of the union* corpus.

single-topic model (SVTD)	
Topic 1:	dollars, 1947, 1945, estimated, reconversion
Topic 2:	21st, century, affordable, children, Medicare
Topic 3:	class, work, cold, people, worked, working
Topic 4:	years, people, energy, elected, congress, peace
Topic 5:	21st, challenge, children, century, parents
Topic 6:	challenge, children, working, challenges, work
Topic 7:	produced, care, health, kids, people
Topic 8:	Vietnam, plan, recommend, deficit, numbers
Topic 9:	highway, Vietnam, recommend, program, federal
Topic 10:	Hussein, Saddam, intelligence, aids, weapons
Topic 11:	applause, program, government, federal, people
Topic 12:	benefits, democratic, economic, great, life
Topic 13:	federal, government, programs, Hussein, intelligence
Topic 14:	alliance, Atlantic, people, free, Europe
Topic 15:	applause, terrorists, Iraq, Iraqi, terror
Topic 16:	strikes, bargaining, collective, labor, management
Topic 17:	space, soviet, disarmament, military, defense
Topic 18:	disarmament, space, defense, soviets, military

LDA (SVTD)	
Topic 1:	dollars, 1947, 1945, estimated, reconversion
Topic 2:	21st, college, affordable, children, child
Topic 3:	class, cold, worked, cuts, talk
Topic 4:	regulations, plan, government, reducing, inflation
Topic 5:	Vietnam, south, 21st, tonight, principle
Topic 6:	challenge, children, working, work, challenges
Topic 7:	Vietnam, south, tonight, north, conflict
Topic 8:	companies, plan, deficit, invest, care
Topic 9:	highway, maintenance, postal, planning, program
Topic 10:	Hussein, Saddam, seniors, aids, intelligence
Topic 11:	applause, Medicare, Hussein, seniors, Saddam
Topic 12:	controls, demands, labor, study, initiative
Topic 13:	percent, family, people, America, tonight
Topic 14:	produced, care, kids, health, renew
Topic 15:	applause, Iraq, terrorists, terrorist, seniors
Topic 16:	collective, strikes, bargaining, management, labor
Topic 17:	soviet, soviets, military, peace, disarmament
Topic 18:	space, disarmament, civil, defense, Latin

Table 3.2: The most relevant words for each topic for the *state of the union* corpus, learned by SVTD setting $k = 18$, both for the single-topic model and for LDA.

based approach. In all the experiments, the proposed approach outperforms the MCMC-based one in terms of both coherence of the learned topics and running times.

Chapter 4

Methods of Moments for Clustering and an Application to Healthcare Analytics

4.1 Introduction

Healthcare analytics is a natural field of application for machine learning. On a daily basis, hospitals and healthcare institutions produce millions of data points providing clinical information on the status of their patients. Machine learning can help in finding patterns in these data, helping doctors in making data-driven decisions.

Latent variable models find a natural field of applications in this context. For example, imagine to assign to each patient a latent variable, indicating his true (unobservable) medical status, and that all the data that a hospital collects about the patient are random variables whose distribution depends on the latent condition of the patient. This strategy provides a generic modeling framework to use latent variable models to study and synthesize the statuses of patients visiting a hospital, enabling also to predict the future evolution of patient latent statuses, given their observable data. In this chapter we will use this modeling framework with the objective of clustering patients into groups with homogeneous clinical profiles.

4.1.1 Clustering Patients

Clustering patients is a strategic activity for modern healthcare systems. First, similar patients share the need for similar cares, so the system can design specific guidelines to treat and prescribe diagnostic patterns rather than single diagnostics. Also, clear and tested clusters based on comorbidities help clinicians to decide treatments on specific patients. Third, characterizing patient patterns helps the system in planning resources and in performing fair comparisons between institutions.

In this chapter we aim at providing a technique for clustering patients into homogeneous groups. We want this technique to rely on generic, widely available data – so to be easily usable in several different hospitals – to be able to run in nearly real time and to be simple to use – so that its implementations could be seamlessly used by doctors or hospital professionals. In general, these professionals have little or no training in data science; their organizations often have a statistics department, but asking the department for analysis usually takes weeks, which hinders intuition and innovation, and leads in most cases to abandoning the analysis because of more pressing day-to-day matters. Our goal is to propose an algorithm that would be largely autonomous, requiring no ad-hoc tuning before every analysis required by the user, and efficient so that intuitions can be explored almost in real time, even for large populations.

Despite the increasing availability of Electronic Healthcare Records (EHR), this data is in general heterogeneous, differing in content and structure between various healthcare systems; this hinders the development of algorithms that can be re-used on several hospitals as well as the reproducibility of research in different contexts. At the same time, hospitals can benefit from the existence of standard, widely used procedures, as this reduces the development costs and provides trusted, out-of-the-box methods ready to be used on their EHR. A patient clustering method that aims to be used on several healthcare systems should thus be based on data available in most of the hospitals. The most universal EHR are probably patient ICD records, in their older and more recent versions, respectively ICD9 (Geraci et al., 1997) and ICD10 (World Health Organization, 2004). ICD-based EHR contain information about patients diagnostics registered by the hospital. A clustering algorithm able to leverage on such data will be general enough to be universally applied

in various healthcare systems.

Clustering is a fundamental task of machine learning, and literature is rich of examples of applications to populations of patients. A line of work gathers papers that apply standard clustering methods like k-means (Macqueen, 1967) in a variety of scenarios (see for example Rixen et al., 1996; Kshetri, 2011; Pérez et al., 2016); while literature here is copious, these works tend to be very specific on the clinical problem the authors intend to solve and require domain knowledge to properly engineer, select and process the data and the algorithm, a perspective that is far from the standardized clustering algorithm we have in mind. Furthermore, general-purpose clustering methods often depend on a concept of *distance* between the features, a concept that may lose part of its meaning in high dimensional settings (Aggarwal et al., 2001; Kriegel et al., 2009), especially when data is sparse and/or categorical – a very frequent case in healthcare analytics. Additionally, the "right" notion of distance may be different for every application of the algorithm, i.e., may change if the manager is trying to cluster patients with diabetes, young or older population, or users of semi-critical units, or any other possible profile. For instance, Kshetri (2011) use k-means to model patient states in intensive care units, defining feature specific distances to face the issue of mixed data types. Designing a distance function may involve selecting or weighting the relevant diagnostics, clinical findings, discretizing or combining attributes, etc., and this may be difficult and error-prone in a everyday usage. The same problem applies to other distance-based methods that we are aware of. Linkage-based methods such as DBSCAN (which also require a distance function) are designed for finding compact "shapes" in data rather than interpretable results or homogeneous groups.

An alternative line of work on patient clustering leverages on tensor decomposition techniques to find recurrent phenotypes and use them to cluster patients (see Ho et al., 2014b,a; Wang et al., 2015; Perros et al., 2017); using several standard sources of data (like ICD codes and procedures) these techniques first create a multidimensional tensor, that, for each patient, counts the cooccurrences of the data along various observable modes (for example, entry (i, j, h) to be 1 if patient i has disease j and has been prescribed procedure h). A low-rank tensor decomposition will then return synthetic representations of recurrent phenotypes and allow to cluster patients according to their similarity to one of the phenotypes. While widely applicable, these techniques tend to suffer of long computational times, requiring the decompo-

sition of very high dimensional tensors, making difficult any real time analysis.

A different class of clustering techniques relies on the distributional properties of the observed variables, using mixture models (see Marin et al., 2005; Sun et al., 2007; Melnykov et al., 2010). Mixture models describe the observable data as outcomes of joint probability distributions and present many advantages over the cited techniques: they do not need an a-priori defined distance, their nature of probabilistic graphical models allows natural and objective interpretations and their flexibility allows to work with both single and multiple sources of data. Additionally, mixture models enable to describe the data one is using under the latent variable models perspective, where the latent variable indicates the true, unobservable medical status of the patient, while the observable features – like for example the diseases of a patient – are random variables whose outcomes depend on the latent states of the patient. This is the approach we take here, focusing on a special case of mixture models, namely naive Bayes models (see Chapter 1) where all the observable features are conditionally independent binary variables (this model is typically called *mixture of independent Bernoulli variables*). Performing clustering with these models is easy, and can be done with a 1-row formula, but it requires to know the parameters of the considered mixture model. As a consequence, a learning procedure is typically required to recover, from a a dataset that has to be clustered, the mixture model that will be used to perform the clustering task.

The traditional approach to learn mixture of independent Bernoulli variables, consists in using Expectation Maximization (EM) (Dempster et al., 1977), a technique that however tends to scale too badly for the kind of interactive high-dimensional discovery that we envision. At the same time, methods of moments (see Chapter 1) provide guarantees of speed, stability and results quality that make them suitable for our purposes. This is the approach we take here. However, applying methods of moments to learn mixture of independent Bernoulli variables is not a trivial task, as direct approaches for calculating unbiased estimators of the moments (like those outlined in Chapters 1 or 3) are not known; while several indirect approaches have been proposed – Jain and Oh (2014) for example rely on optimization techniques, while Anandkumar et al. (2012b) propose a multi-view factorizations of the features – none of them, when used on real data provides the efficient and robust performance we require for our goals (see Section 4.3 for

a more detailed description of existing techniques).

In this chapter, we propose a novel learning technique for mixtures of independent Bernoulli variables and use it to perform patient clustering. Our approach, instead of using a method of moments to retrieve asymptotically convergent estimates of the unknown parameters, uses this decomposition procedure to generate approximate parameters that are then used to feed EM as starting point. In the practical applications, using this technique, EM converges fast, reaching in few steps surprisingly good optima; also the high scalability allows the usage of this method on high-dimensional datasets. Additionally, we apply our method to three real-world datasets to perform patient clustering. The first two datasets have been obtained by a collaboration with the Servei Català de la Salut – the major healthcare provider in Catalonia (Spain) – and contain data from hospital admissions of two selected profiles: patients with Congestive Heart Failure, and "Tertiary" patients (patients with serious diseases that can only be treated by top specialists or with highly specialized equipment). The third dataset is the publicly available MIMIC III (Johnson et al., 2016), which contains clinical care data collected at Beth Israel Deaconess Medical Center (Boston, US) from 2001 to 2012. In all the cases, the data we consider contains essentially the list of ICD9 codes of patients' diagnostics. Data is turned into a binary matrix where columns represent diagnostics, so highly sparse, as every patient has most often less than a dozen diagnostics out of several thousand possible ones. The proposed method finds well-characterized clusters in the data, which clinicians find meaningful, novel to some extent, and potentially useful for refining clinical guidelines.

In Section 4.2, we provide details on mixture of independent Bernoulli variables, together with some indications on how to use them to perform clustering. Section 4.3 overviews learning techniques for these models and presents a novel approach for this purpose. Section 4.5 experimentally compares the proposed technique with other methods, both from the clustering and methods of moments literature. Finally in Section 4.6, we apply our method to the real-world datasets described above to perform patient clustering.

4.2 Clustering with Mixtures of Independent Bernoulli Variables

In this section we will present the latent variable model we are going to use in the rest of the chapter, together with some details on how it can be used to perform clustering.

We will focus on a special class of naive Bayes models (see Chapter 1), where the observable features are binary variables. In this setting, our model will be characterized by the following generative process:

- First, a latent state $Y \in \{1, ..., k\}$ is drawn from a discrete random variable:
$$\mathbb{P}(Y = j) = \omega_j, \;\; \omega = (\omega_1, ..., \omega_k) \in \Delta^{k-1}.$$
- Then, a random vector with d binary features $X = (X_1, ..., X_d)$ is drawn. The features are conditionally independent, and their distribution only depends on the value of the latent state Y:
$$Distr(X_i|Y = j) = Binary(\mu_{j,i}) \;\; \mathbb{P}(X_i = 1|Y = j) = \mu_{j,i} \in [0, 1].$$
Similarly with what done with Poisson naive Bayes models, we will define *centers* of the mixture, the vectors $\mu_j = (\mu_{j,1}, ..., \mu_{j,d})$, and we will store them in the columns of a matrix $M = [\mu_1, ..., \mu_k] \in \mathbb{R}^{d \times k}$.

A consequence of the binary nature of the features and of their conditional independence, is the fact that the model is entirely characterized by its mixing weights ω and by its matrix of centers M. Learning a mixture of independent Bernoulli variables from a set of observations $\mathcal{X} = \{x^{(1)}, ..., x^{(n)}\}$ means to recover an estimated pair of model parameters $(\tilde{M}, \tilde{\omega})$ of a mixture of independent Bernoulli variables that approximatively describe the data.

Mixture of independent Bernoulli are a powerful tool to perform clustering. In fact, given a random sample x generated by such a model, we can derive a conditional distribution for its latent state Y:

$$\mathbb{P}(Y = j|X = x) \propto \omega_j \prod_{i=1}^{d} \mu_{j,i}^{x_i} (1 - \mu_{j,i})^{1-x_i} \tag{4.1}$$

and perform clustering by assigning it to the class corresponding to its most likely latent state:

$$Cluster(X) = \underset{j=1,...,k}{\text{argmax}}(\mathbb{P}(Y = j|X)). \tag{4.2}$$

This clustering procedure, has the additional advantage of providing a clear and interpretable description of the clusters. In fact, the vectors μ_j contain the probabilities of observing non-zero values for the various features; taken a vector X belonging to the cluster $Y = j$, its ith entry will be likely to be 1 if the corresponding entry in the center μ_j, namely $\mu_{j,i}$, will be high. The centers μ_j will thus be synthetic descriptors of the clusters, highlighting the expected behavior of a sample belonging to that class.

Given a dataset of independent binary vectors $\mathcal{X} = \{x^{(1)}, ..., x^{(n)}\}$, we can model it as generated by a mixture of independent Bernoulli variables, and use Equation (4.2) to cluster its samples into k groups. However, Equation (4.2) requires to know the parameters (M, ω) of the mixture we want to use to perform clustering, which, to be retrieved, require a learning algorithm. The next section is dedicated to this task.

4.3 Learning Mixtures of Independent Bernoulli Variables

In this section we present a learning algorithm for mixtures of independent Bernoulli variables. We will begin with a quick overview of existing methods, to propose then a novel technique.

4.3.1 Existing Methods

The standard approach to learn mixtures of independent Bernoulli variables consists in employing **E**xpectation **M**aximization (EM) (Dempster et al., 1977). EM, starting from an initial estimate $(\tilde{M}, \tilde{\omega})$, iteratively updates those parameters, increasing at each step the likelihood of the model, until convergence is reached. The working mechanism of EM is very simple: *first*, Equation (4.1) is used to calculate the probability that each observed sample belongs to each cluster. *Then* each center μ_j is re-calculated as the weighted average of all the points in the dataset, where the weight for each point

is the probability that this point would belong to cluster j. The issue of EM is that it may converge to poor local optima, if initialized poorly, and that at each iteration it requires a full pass across all the samples in the dataset. Random initializations are thus unlikely to provide meaningful results in reasonable amounts of time (See Marin et al., 2005, for details on EM).

Given the fact that mixtures of independent Bernoulli variables admit a parametrization in terms of a pair (M, ω), exactly like the models studied in Chapters 1 and 3, methods of moments represent a viable and promising alternative for their learning. To follow this approach, we require a family of estimators $\tilde{M}_1$, $\tilde{M}_2$ and $\tilde{M}_3$ providing the following relation with the parameters of the model.

$$\mathbb{E}[\tilde{M}_1] = M_1 = \sum_{i=1}^{k} \omega_i \mu_i, \tag{4.3}$$

$$\mathbb{E}[\tilde{M}_2] = M_2 = \sum_{i=1}^{k} \omega_i \mu_i \otimes \mu_i, \tag{4.4}$$

$$\mathbb{E}[\tilde{M}_3] = M_3 = \sum_{i=1}^{k} \omega_i \mu_i \otimes \mu_i \otimes \mu_i. \tag{4.5}$$

As we have seen in previous chapters, the typical approach to retrieve these estimators consists of relying on the low order moments of the observed data, providing additional translations to obtain unbiased estimators (like we did for example in Theorems 1.2.1 or 3.2.1). However, equations allowing such a direct approach for mixtures of independent Bernoulli variables are not known, and finding moment-based estimators for all the entries of M_2 and M_3 ended up being a challenging task.

To see this, take a dataset $\mathcal{X} = \{x^{(1)}, ..., x^{(n)}\}$, and define the raw moments

as follows:

$$\hat{M}_1 = \sum_{i=1}^{n} \frac{x^{(i)}}{n}, \tag{4.6}$$

$$\hat{M}_2 = \sum_{i=1}^{n} \frac{x^{(i)} \otimes x^{(i)}}{n}, \tag{4.7}$$

$$\hat{M}_3 = \sum_{i=1}^{n} \frac{x^{(i)} \otimes x^{(i)} \otimes x^{(i)}}{n} \tag{4.8}$$

It is trivial to see that the raw-moments are good estimators for M_1 and for the off-diagonal entries of M_2 and M_3, because we have that, for each triple of mutually distinct i, j and h,

$$\mathbb{E}[(\hat{M}_1)_i] = (M_1)_i, \tag{4.9}$$

$$\mathbb{E}[(\hat{M}_2)_{i,j}] = (M_2)_{i,j}, \tag{4.10}$$

$$\mathbb{E}[(\hat{M}_3)_{i,j,h}] = (M_3)_{i,j,h}. \tag{4.11}$$

However, the relations above fail if at least two of i, j and h coincide; that means that the raw moments, do not suffice to estimate the off diagonal entries of M_2 and M_3; unfortunately, it is enough to spend some time playing with the estimates above to realize that no trivial variations of the raw moments allow to retrieve unbiased estimators for the diagonal entries of M_2 and M_3.

This challenging scenario led to the introduction of several indirect techniques to retrieve unbiased estimators of the moments at Equations (4.3), (4.4) and (4.5). Jain and Oh (2014) for example follow an optimization-based approach: first they estimate the raw moments, then they discard their diagonal entries, which are treated as missing entries. A tensor/matrix completion technique based on alternating least square is then used to fill these missing entries. This approach, however, only works when the number of features is significantly higher than the number of clusters k; furthermore, it requires the storage of the full three dimensional tensor M_3 in memory, reducing the scalability of the algorithm (depending on a factor d^3 for both memory and time requirements, where d is the number of features). Another method has been proposed by Anandkumar et al. (2012b) – and subsequently refined (Anandkumar et al., 2014) – and it is based on the so-called multi-view

approach. Here, the features are first split into three *views*, X_a, X_b and X_c, like in the following example:

$$X = (\underbrace{X_1, \ldots, X_{d_a}}_{X_a}, \underbrace{X_{d_a+1}, \ldots, X_{d_a+d_b}}_{X_b}, \underbrace{X_{d_a+d_b+1}, \ldots, X_{d_a+d_b+d_c}}_{X_c}).$$

After that, a number of linear transformations are performed on the cross-moments between the views – including matrix inversions, compositions and singular value decompositions – to retrieve consistent estimators of the moments. This approach, which requires to have at least $d \geq 3k$ features, is known to be extremely unstable under random perturbations. Furthermore, the definition of the views – i.e. the decision of which features to put in each view – is a non-trivial task, that – when only limited data is available – meaningfully impacts the output parameters. These limitations make the multi-view approach to clustering hardly applicable in the medical setting, where the stability of a learned model represents a key requirement.

4.3.2 A New, Approximate Approach

All the existing techniques for estimating mixtures of independent Bernoulli variables present some drawback, whether in terms of efficiency, or in terms of poor robustness to perturbations. In this section we will present a new approach that aims at overcoming the limitations listed above.

Our approach relies on the observation that the raw moments provide unbiased estimators for the desired moments (4.3), (4.4) and (4.5) for most of their entries – namely all the off-diagonal entries – and only the diagonal entries present a certain kind of bias. In the following theorem we show that this bias of the raw moments is small.

Theorem 4.3.1. *Let $\hat{M}_1$,$\hat{M}_2$ and $\hat{M}_3$ be the raw moments defined at Equations* (4.6),(4.7) *and* (4.8). *Define the values of the conditional second and third order moments for an observable variable:*

$$\mu_{j,i}^{(2)} = \mathbb{E}[X_i^2 | Y = j], \quad \mu_{j,i}^{(3)} = \mathbb{E}[X_i^3 | Y = j].$$

If $\mu_{j,i} \in [0,1]$ $\forall i,j$, we have for all $i \neq j$:

$$|(M_2)_{i,i} - (\tilde{M}_2)_{i,i}| \leq \sum_{j=1}^{k} \omega_j \mu_{j,i}^{(2)} - (M_1)_i^2,$$

$$|(M_3)_{i,i,l} - (\tilde{M}_3)_{i,i,l}| \leq \sum_{j=1}^{k} \omega_j \mu_{j,i}^{(2)} \mu_{j,l} - \frac{(M_2)_{i,l}^2}{(M_1)_l},$$

$$|(M_3)_{i,i,i} - (\tilde{M}_3)_{i,i,i}| \leq \sum_{j=1}^{k} \omega_j \mu_{j,i}^{(3)} - (M_1)_i^3.$$

Proof. First, recall that because each $\mu_{j,i} \in [0,1]$, we have that

$$\mu_{j,i}^{(2)} \geq \mu_{j,i}^2, \quad \mu_{j,i}^{(3)} \geq \mu_{j,i}^3.$$

We now prove each claimed equation one by one.

1. $|(M_2)_{i,i} - (\tilde{M}_2)_{i,i}| \leq \sum_{j=1}^{k} \omega_j \mu_{j,i}^{(2)} - (M_1)_i^2$.
   The thesis is a consequence of the following chain of inequalities:

$$(\tilde{M}_2)_{i,i} = \mathbb{E}[x_i^2] = \sum_{j=i}^{k} \omega_j \mu_{j,i}^{(2)} \geq \sum_{j=i}^{k} \omega_j \mu_{j,i}^2 =$$

$$= (M_2)_{i,i} \geq (\sum_{j=i}^{k} \omega_j \mu_{j,i})^2 = (M_1)_i^2,$$

   where we have applied the Jensen inequality.

2. $|(M_3)_{i,i,l} - (\tilde{M}_3)_{i,i,l}| \leq \sum_{j=1}^{k} \omega_j \mu_{j,i}^{(2)} \mu_{j,l} - \frac{(M_2)_{i,l}^2}{(M_1)_l}$.
   We have

$$(\tilde{M}_3)_{i,i,l} = \sum_{j=1}^{k} \omega_j \mu_{j,i}^{(2)} \mu_{j,l} \geq \sum_{j=1}^{k} \omega_j \mu_{j,i}^2 \mu_{j,l} =$$

$$= (M_3)_{i,i,l} \geq \frac{(M_2)_{i,l}^2}{(M_1)_l},$$

   from which again we get the desired inequality.

3. $|(M_3)_{i,i,i} - (\tilde{M}_3)_{i,i,i}| \leq \sum_{j=1}^{k} \omega_i \mu_{j,i}^{(3)} - (M_1)_i^3$.
   The proof of this point is identical to that of point 2.

□

The theorem above says that if we are learning a mixture of independent Bernoulli variables, the expected value of the bias of the raw moments is small. Starting from this observation, we propose the following approximate approach:

1. First, use the raw moments as input to SVTD, to retrieve a pair $(\hat{M}, \hat{\omega})$ of model parameters – which, will be biased as a consequence of the bias in the input moments.

2. Then, we plug these estimated parameters as initializers for EM, with the objective of improving them, obtaining a model that locally maximizes the likelihood.

This approach is an heuristic, but presents several advantages over existing techniques: first, it allows an efficient and optimized implementation of SVTD – that we will call Approximate Singular Value-based Tensor Decomposition (ASVTD): this implementation, instead of taking as input the moments, takes as input a dataset, and embeds the calculations of the moments with the inner steps of SVTD, obtaining, as we will see in Section 4.3.2, improved computational performance. The second advantage will be that, despite the lack of theoretical guarantees, this approach will lead to high-quality model parameters, with EM converging fast to surprisingly good local optima, as we will experimentally see in Section 4.5.

Algorithmic Details: Implementing ASVTD

Our approach consists of using the raw moments as input to SVTD, obtaining a pair of approximate estimates $(\hat{M}, \hat{\omega})$ that are then updated with EM, until when convergence is reached. The usage of the raw moments allows to modify the structure of SVTD in order to reduce its complexity, by embedding the moment calculation in the inner steps of SVTD (we refer the reader to Chapter 2 and Algorithm 1 for details on SVTD). This improved implementation will lead exactly the same results as SVTD inputted with the raw moments, but with an improved efficiency.

Pure SVTD works with all the explicit slices of the $d \times d \times d$ tensor M_3, which may be computationally demanding, especially when the number of features

is high. Using the raw moments, enable us to avoid the explicit usage of the slices of M_3, working with matrices with a lower dimension, improving the complexity of the algorithm. Given a dataset

$$\mathcal{X} = \{x^{(1)}, ..., x^{(n)}\}$$

we define $\bar{x} \in \mathbb{R}^{n \times d}$ the matrix whose ith row is the vector $x^{(i)}$. This matrix, together with the number of latent states k will be the input of ASVTD.

Algorithm 2 Approximate - SVTD

Require: The sample $\bar{x} \in \mathbb{R}^{n \times d}$, the number of latent states $k \leq d$

1: Compute $\hat{M}_2 = \frac{\bar{x}^\top \bar{x}}{n} \in \mathbb{R}^{d \times d}$
2: Decompose $\hat{M}_2$ as $\hat{M}_2 = U_k S_k U_k^\top$ with a SVD.
3: Define the whitening matrix $E = U_k S_k^{1/2}$ and calculate its pseudoinverse $E^\dagger = (S_k)^{-1/2} U_k^\top$.
4: Project $\bar{x}$ on $\mathbb{R}^{n \times k}$: $\bar{z} = \bar{x}(E^\dagger)^\top$
5: Initialize $\alpha_{min} = -\infty$ and $O = \mathbb{I}_k$ as the $k \times k$ identity matrix.
6: **for** $i = 1 \to d$ **do**
7: Compute $H_i := \frac{\bar{z}^\top \operatorname{diag}((x_i^{(1)}, ..., x_i^{(n)}))\bar{z}}{N}$
8: Compute the singular values of H_i, $(s_1, ..., s_k)$ and the left singular vectors O_i.
9: **if** $\min_{i \neq j}(|s_i - s_j|) > \alpha_{min}$ **then**
10: Set: $\alpha_{min} = \min_{i \neq j}(|s_i - s_j|)$, and $O = O_i$
11: **end if**
12: **end for**
13: **for** $i = 1 \to d$ **do**
14: Obtain the ith row of $\hat{M}$ as the diagonal entries of $O^\top H_i O$
15: **end for**
16: Obtain $\hat{\omega}$ solving $(\sum_i x_j^{(i)}/n)_{j=1..d} = (\hat{M}\hat{\omega})_{j=1..d}$
17: Return $(\hat{M}, \hat{\omega})$

Comparing SVTD (Algorithm 1) and its approximate implementation (Algorithm 2), we notice that the main differences are in the loop from row 6 to 12 of Algorithm 2. In fact, standard SVTD computes the matrices H_i, whitening the ith slice of M_3:

$$H_i = E^\dagger M_{3,i} (E^\dagger)^\top;$$

this operation is pretty expensive, as it requires $O(d^2k)$ operations, and is repeated d times, for a total cost of $O(d^3k)$. Instead, in ASVTD, we never compute explicitly $\tilde{M}_3$, but we compute directly H_i, exploiting the fact that, in this case

$$H_i = \frac{E^\dagger \bar{x}^\top \operatorname{diag}((x_i^{(1)}, ..., x_i^{(n)}))\bar{x}(E^\dagger)^\top}{n}.$$

So, at row 4, we whiten the matrix of the observed samples, and we use it during all the algorithm; the calculation of each H_i now costs only $O(k^2n)$, which in many cases is a substantial improvement (especially when the number of the features is high).

Notice that there is a second difference between the implementation of ASVTD and that of SVTD, that is the fact that ASVTD has explicitly implemented the feature selection process described at remark 2.1.2. In fact, SVTD requires, at step 3, to isolate a feature, for which to calculate the matrix O that will be used to diagonalize the various matrices H_r, and retrieve consequently the rows of the matrix M. This procedure is explicitly implemented in ASVTD in the loop from row 6 to 12, where the feature that maximizes the difference between the two nearest singular values is selected.

We now analyze the complexity requirements of ASVTD. It is easy to see that the steps from 1 to 4 have a time complexity of $O(d^2n)$ for step 1, $O(d^2k)$ for step 2, using randomized techniques (Halko et al., 2011) and $O(ndk)$ for step 4. In the loop from step 6 to 12 we have the calculation of H_i, costing $O(k^2n)$, and a $k \times k$ SVD, costing $O(k^3)$, giving a total time complexity of $O(d^2n + dk^2n + dk^3)$ in the realistic cases where $d, k < n$. Also the memory requirements are mild, as the only storage is for the matrices H_i, costing $O(dk^2)$, $\hat{M}_2$, costing $O(d^2)$ and $\bar{x}$, for $O(dn)$. The total memory is thus $O(dk^2 + d^2 + dn)$.

4.4 Putting it All Together

We are now ready to merge all the elements presented till now, and to show how to use the presented learning technique to cluster a dataset of independent binary vectors. Starting from a dataset $\mathcal{X}$, our proposed approach consists of the following flow: *First*, estimate the raw moments $\hat{M}_1$, $\hat{M}_2$, $\hat{M}_3$, as in Equations (4.6), (4.7) and (4.8) and retrieve the model parameters $(\hat{M}, \hat{\omega})$

with SVTD (or, directly use ASVTD for an optimized implementation). *Then*, apply EM to improve the quality of the estimated parameters. *Last*, use the clustering formula described at Equation (4.2) to group the observed samples into k clusters.

Remark 4.4.1 (Tuning the clustering results)**.** The runs of EM performed after the parameter retrieval guarantee that the retrieved mixture model is (locally) optimal in terms of likelihood. Additional fine-tuning is possible re-running the algorithms in different configurations, for example modifying the number of clusters, adding/excluding certain features, or performing additional clustering within one of the clusters. The interpretation technique provided in the Section 4.6 allows a user to visually inspect results and add expert considerations on the discovered patterns. Other types of manual fine-tuning (like manually assign a patient to a different cluster) may be possible, at the cost of adding human bias to the results of the model.

4.5 Experiments

In the previous section we have described an approach to cluster a dataset of high-dimensional binary vectors into k groups. In this section we want to experimentally compare the clustering accuracy of the proposed method with that of state-of-the-art methods.

We proceed as follows: *First* we generate a synthetic dataset, distributed as a mixture of independent Bernoulli, registering for each sample, the cluster to which it belongs (i.e. the latent variable generating the observation); the synthetic parameters (M, ω) have been generated as exponentially distributed random numbers, normalized to be between 0 and 1. Then, we cluster the samples in an unsupervised way, and we compare the obtained clusters with the theoretical ground-truth, registered when generating the data. We then repeat this procedure for various model and sample sizes. In order to analyze the clustering accuracy, we use the Adjusted Rand Index from Hubert and Arabie (1985) between the theoretical ground-truth partition and the one obtained with the clustering algorithm. We recall that the adjusted rand index is an indicator that compares whether two partitionings of a dataset are or not similar; an adjusted rand index of 1 indicates that two clusterings are identical, while a random labeling should have a value close to 0. Our comparison will be performed between several different algorithms:

1. The approach described in this chapter: in particular, we will use Algorithm 2 to recover a set of approximate parameters that will be then refined with EM; last, Equation (4.2) will be used to perform clustering. EM will be run until convergence, stopping the algorithm when the norm of the variation on the estimated ω between an iteration and the next will be less than 0.01.

2. With formula (4.2), where the parameters are retrieved using the three-views method of moments from Anandkumar et al. (2012b) (3V-MoM in the charts). Here, we obtained the views by randomly splitting the features into three groups. For this case, we test both, the scenario where EM is used after the parameter retrieval (to get improved parameters) and scenario where not.

3. As above, with formula (4.2), where the parameters are retrieved with the three-views method of moments, but using the variation from Anandkumar et al. (2014) (3V-TPM in the charts) and obtaining the views following the procedure of the previous point. Also for this case, we test both, the scenario where EM is used after the parameter retrieval and scenario where not.

4. K-Means.

5. Spectral Clustering from von Luxburg (2007), using a linear kernel ("SC-lin" in the figures).

6. PCA clustering, where k-means is used to cluster the sample where the dimensionality has been reduced using a PCA.

The results are displayed in figure 4.1. The top three figures represent the experiment where EM is not run after the multi-view clusterings from Anandkumar et al. (2012b, 2014), while the three figures below represent the experiment where EM is used. In the leftmost figures we fixed the number of features $d = 99$ and the number of latent states $k = 12$, varying the sample size from $n = 100$ to $n = 10000$. In the central figures, the sample size $n = 10000$ and the number of features $d = 99$ are fixed, while the number of latent states varies from $k = 2$ to $k = 33$. In the rightmost figure, the sample size $n = 10000$ and the number of latent states $k = 4$ are fixed, while the number of features varies from $d = 12$ to $d = 99$. It is interesting to notice that all the reference methods (K-means, Spectral Clustering and

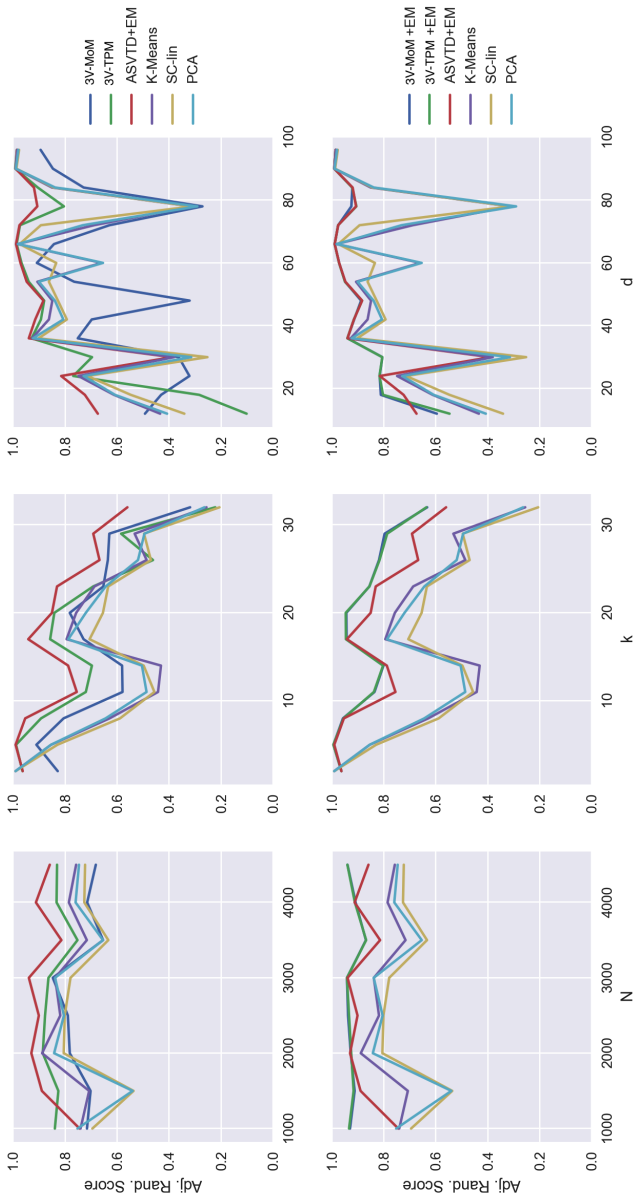
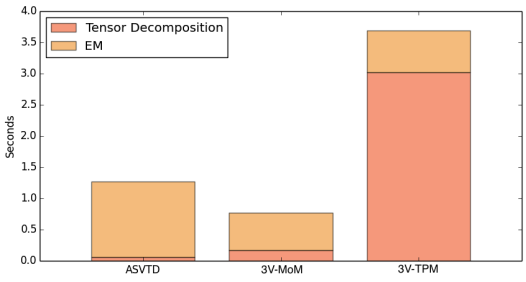


Figure 4.1: Adjusted Rand Index of the compared methods, when EM is not used after the multi-view methods (top three figures) and when is used (bottom three figures).

PCA) are outperformed by methods of moments; also, it seems that the intuition of using a mixed method of moments + EM approach provides good results, as ASVTD + EM, despite the lack of theoretical guarantees, performs better than the pure (i.e. without EM) multi-view methods of moments from Anandkumar et al. (2012b, 2014). If we plug some EM iterations after the multi-view approaches (three figures below), we can see an improvement in their learning performance, behaving similarly to ASVTD + EM. However, while ASVTD does not need any preprocessing of the data, multi-view methods of moments do require to split the features into conditionally independent views, a task that may require several trial and errors when dealing with unbalanced real-world datasets.



(a)

Method	**Time**
ASVTD + EM	1.3 sec.
3V-TPM + EM	3.68 sec.
3V-MoM + EM	0.77 sec.
KMeans	0.76 sec.
SC - lin	7.48 sec.
PCA	0.38 sec.

(b)

Figure 4.2: The average clustering time spent for each method.

We now briefly discuss the running times. For each run of the experiment described in the previous paragraph we registered the time employed by each of the clustering methods. We then average all these records, in order to get, for each method, the average time needed to perform the clustering. Results are displayed in Table 4.2b.

It is immediate to see that Spectral Clustering method is much slower than the competing techniques. This is a consequence of dealing with an $n \times n$ affinity matrix, that requires a number of operations quadratic in the sample size. PCA and K-means are pretty fast, but we have seen that they perform poorly on binary data. Looking at methods of moments, 3V-TPM + EM seems to be the slowest method, consequence of a worst scalability on the

number of latent components, while ASVTD + EM and 3V-MoM + EM have similar performance. Figure 4.2a displays for each method of moments how much of this running time is spent in the decomposition part – the orange portion – and how much in EM. As expected, ASVTD is significantly faster than other decomposition methods, due to the better scalability. However, as it does not provide a guaranteed estimation of the model parameters, EM takes a bit more time to converge. If we look at the other algorithms we can see a significantly longer time in decomposing tensors, and a shorter time in doing EM.

Experimental setting: All the experiments in this section have been run on a MacBook Pro with a 2.7 GHz Intel Core i5 processor and 8 GB of RAM memory. All the algorithms have been implemented in Python 2.7 (interpreted, not compiled), using the *numpy* (Walt et al., 2011) library for all linear algebra operations, including Singular Value Decomposition.[1] The methods from Anandkumar et al. (2012b, 2014) have been implemented by the author of this thesis. For K-Means, spectral clustering and PCA, we have used the implementation provided by Scikit-Learn python library (Pedregosa et al., 2011).[2]. Implementations for ASVTD have been publicly disclosed.[3]

4.6 Patient Clustering

In this section we present the results of applying the proposed approach to cluster patients form real-world EHRs.

4.6.1 The Datasets

We consider three datasets. The first two have been provided by Servei Català de la Salut, the major provider of public healthcare in Catalonia, Spain.[4] The datasets are subsets of a bigger database containing all hospitalizations in Catalonia for the year 2016. The first dataset contains all the patients affected by Heart Failure, to be precise patients having a primary

[1] `https://docs.scipy.org/doc/numpy/reference/generated/numpy.linalg.svd.html`

[2] `http://scikit-learn.org`

[3] `https://github.com/mruffini/NaiveBayesClustering`

[4] Due to privacy constraints, these datasets are not publicly available.

or secondary diagnostic 428 in the ICD-9 code. The second dataset contains "tertiary" patients, those with a serious disease that is supposed to be treated in one of the 5-6 large, reference hospitals in the area, because either the expertise or the required resources are only found there; criteria for tertiarism include all oncology surgery, all heart surgery, all neurosurgery, major traumatism, acute ictus, myocardial infarct and sepsis, transplants, and a few rarer conditions. The third dataset we consider is MIMIC III (Johnson et al., 2016), a freely available dataset containing clinical care data collected at Beth Israel Deaconess Medical Center from 2001 to 2012, focusing again on the diagnostic ICD9 records. The datasets have the same format: each row represents a visit of a patient to a hospital, and the columns contain the codes of the diagnostics that the doctor annotated in the patient history during the stay; their order is considered irrelevant. Diagnostics are coded in the international ICD-9 code (Geraci et al., 1997). The codes are hierarchical, of the form ddd.dd, with the .dd part being optional: the first 3 digits are a general diagnostic and extra digits, if present, make the diagnostic more precise. We kept only the first three digits of the ICD-9 codes, which are those with more consensus among doctors, obtaining a total vocabulary of 696 codes. In this way it is straightforward to map a dataset into a $n \times 696$ matrix with binary entries, where n is the number of records of each dataset, using an approach analogous to the bag-of-words: first associate each registered disease to a unique number between 1 and 696, and then populate a matrix $\bar{x}$ placing a 1 at position (i, j) if the record i presents diseases j, otherwise a zero. After this processing, we obtained three datasets: for the Tertiarism dataset, $\bar{x}_t$ with $n = 16311$ samples, for the Heart Failure dataset $\bar{x}_c$ with $n = 23154$ samples and, for the MIMIC dataset $\bar{x}_m$ with $n = 56360$ samples.[5]

4.6.2 Modeling Strategy

With the approach described above we will obtain, for each dataset, a binary matrix, whose rows will correspond to the patients, and columns will correspond to the diseases. We will model the rows of each dataset as sampled from a mixture of independent Bernoulli variables, with k latent states.[6]

[5]For MIMIC we also removed the patients whose age was greater than 100 years, as we found these records less reliable – for example finding date of birth dating back to mid XIX century, a problem not present in the other datasets.

[6]In all the experiments of this section, the value of k has been manually defined, as a consequence of expert's considerations.

Each latent state will represent a possible unobservable medical status, while the centers $\mu_1, ..., \mu_k$ will represent the probabilities of developing a certain disease when one is in a given clinical status. Each patient finds himself in a specific medical status, which can not be observed; instead, we can only observe some features – the diseases – which can give us indications on the latent status of the patient. With this modeling strategy, *clustering* means to group together the patients with the same latent status. The centers will represent synthetic descriptors of the various statuses, describing how likely each diseases is in each status.[7]

4.6.3 Analysis of the Results

In all the cases we run ASVTD, followed by EM, to then perform clustering using Equation (4.2); the stopping criteria for EM was the moment in which convergence was reached; in particular, we stopped the algorithm when the norm of the variation on the estimated ω between an iteration and the next was less than 0.01. In order to analyze the results, we plot two charts for each dataset: a heat-map and a disease-frequency chart. The heat-map, is in figure 4.3a. In that figure, we took the 40 most common diseases and plot the matrix $\bar{x}_c$ only for the columns associated to those diseases. We ordered the rows of $\bar{x}_c$ according to the cluster to which they belong, and draw them sequentially: each row of Figure 4.3a is a record, the black points indicate whether a disease is present. The background color of each row indicates the cluster (so: white background is the first cluster, clear gray the second, and so on). The purpose of this figure is to give a first visual inspection of the patterns present inside the clusters; heat-maps also give an idea of the dimension of the clusters with respect to the total dataset. Disease-frequency charts are instead used to give a meaning to those patterns, an example is at Figure 4.3b: the top-20 diseases in terms of frequency are displayed, each one in a different row. Then, for each one of the clusters, we show the relative frequencies of the considered diseases; for example, in Figure 4.3b, "Dyslipidemia" is present in the 50% of the patients belonging to the first cluster. To have an additional indication of the content of the clusters, we represent in Tables 4.1, 4.2 and 4.3 the most *relevant* diagnostics, where the *relevance* (Sievert and Shirley, 2014) of a diagnostic i with respect to a cluster

[7] In this framework, the centers are commonly called *phenotypes*.

j, given a weight parameter λ, is defined as

$$r(i,j) = \lambda \log(\mu_{j,i}) + (1-\lambda) \log(\mu_{j,i}/(\sum_{h=1}^{k} \mu_{h,i}\omega_h)) \tag{4.12}$$

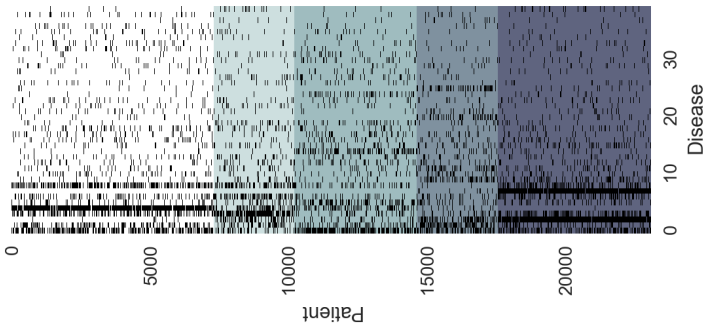
where $\mu_{j,i} = \mathbb{P}(X_i = 1|Y = j)$. The relevance of a diagnostic with respect to a cluster j is an indicator that has a high value if the frequency of the considered diagnostic inside the cluster is much higher then its frequency on the full dataset; diagnostics with a high relevance are those that characterize a given cluster with respect to the others. An analysis of the content of a cluster can be performed merging the information contained in the disease-frequency charts with those provided by the tables of the highly relevant diseases: disease-frequency charts will show us the diseases that are frequent in a given cluster, regardless what happens in the other clusters; conversely, tables of the highly relevant diseases will highlight those diagnostics that characterize a cluster with respect to the others, providing in this way a complete view on the patterns that can be found in the data.

Heart failure dataset

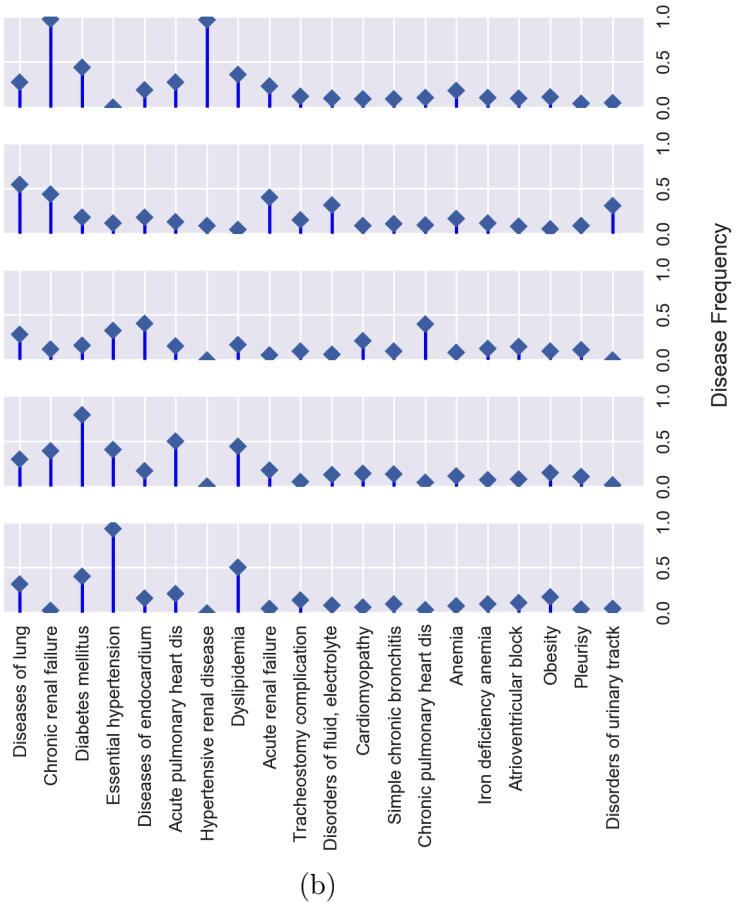
The clusters obtained in the heart failure dataset are plotted in Figure 4.3b, with the associated table (4.1) of the highly relevant diseases. Clusters can be described with high coherence: Cluster 1 could be called the "purely metabolic" cluster: high prevalence of hypertension, and also dyslipidemia and diabetes mellitus. Cluster 2 contains the above complicated with kidney problems; Patients in Cluster 3 present valvular problems and pulmonary hypertension, well-known to be related. Cluster 4 is a mixture of kidney and pulmonary problems (lung diseases are pretty frequent here), with little metabolic implications. Cluster 5 contains the purely kidney sufferers, with diabetes mellitus (nephropaty being a common complication of diabetes) and dyslipidemia. Clinicians confirm that these clusters make sense once seen, and may be useful for guiding treatment. For example, medication that might be indicated for the purely cardiac clusters might be not advisable for the clusters with renal problems if it is suspected to be nephrotoxic.

Tertiarism dataset

The clusters for the tertiarism dataset are plotted in Figure 4.4b, with the associated Table 4.2 of the high relevant diseases. Clinically, Cluster 1 might

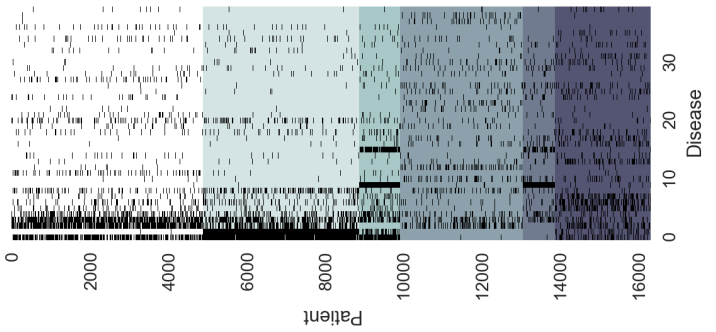


(a)

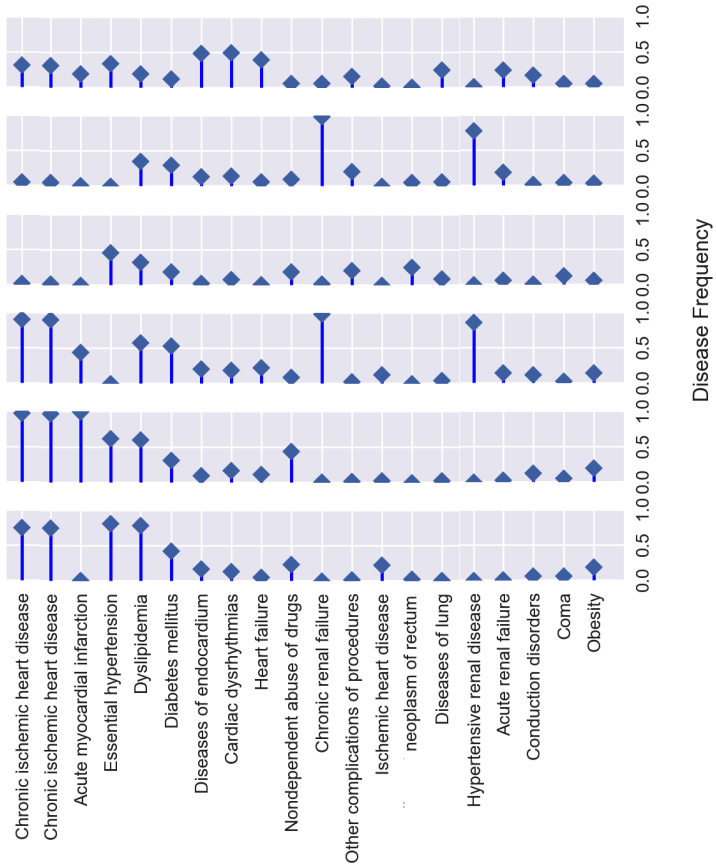


(b)

Figure 4.3: 4.3b Disease-frequency chart for the Heart Failure dataset. 4.3a The corresponding heatmap. The "Heart Failure" disease is not represented as it appears in all the records of the dataset.



(a)



(b)

Figure 4.4: 4.4b Disease-frequency chart for the Tertiarism dataset. 4.4a The corresponding heatmap.

Cluster	Most Relevant Diagnostics	Cluster Size
1	Essential hypertension; Dyslipidemia; Cardiac arrhythmias; Diabetes mellitus; Obesity;	7290
2	Diabetes mellitus; Atherosclerosis; Acute pulmonary heart disease; Retinal disorders; Hypertensive heart and renal disease;	2915
3	Chronic pulmonary heart disease; Diseases of endocardial structures; Diseases of endocardium; Hypertensive heart disease; Cardiac arrhythmias;	4480
4	Bacterial infection; Disorders of urethra - urinary tractk; Acute renal failure; Hypertensive heart - renal disease; Disorders of fluid, electrolyte, acid-base balance; Diseases of lung;	2936
5	Hypertensive renal disease; Chronic and/or acute renal failure; Diabetes mellitus; Anemia;	5533

Table 4.1: The most relevant diseases for each cluster for the Heart failure dataset.

correspond to patients in need of coronary intervention (heart surgery or interventional cardiology), again with strong presence of metabolic anomalies. Cluster 2 is similar, but including myocardial infarction. Cluster 3 includes nephrology patients, who are also taken to tertiary hospitals when they need transplant or are in very advanced phase, and who tend to develop arteriopathy in the long term; interestingly, diabetes mellitus shows up strongly in this cluster, as it is the leading cause of entrance to dialysis programs. Cluster 4 has a surprising behavior: on a first hand, no clear signal comes out from the disease-frequency chart, besides the ever-present hypertension, dyslipidemia and a small signal of colo-rectum cancer. However, if we look at the table of high relevant diseases, we can see that the most relevant diagnostics for this cluster are neoplasms. Oncological problems are large constituents of tertiarism; they are characterized by a large set of codes (almost 100 for neoplasms, by organ of origin mostly), reason why it is not surprising the fact that we do not find them in the disease-frequency plot, as there are many distinct diagnostics indicating oncological diseases; however, the fact that the most relevant diagnostics for this cluster are neoplasms allows us to

Cluster	Most Relevant Diagnostics	Cluster Size
1	Ischemic heart disease; Essential hypertension; Angina pectoris; Dyslipidemia; Chronic ischemic heart disease;	4892
2	Acute myocardial infarction; Chronic ischemic heart disease; Nondependent abuse of drugs; Essential hypertension; Dyslipidemia;	3982
3	Hypertensive renal disease; Chronic renal failure; Chronic ischemic heart disease; Diabetes mellitus; Acute myocardial infarction;	1043
4	Malignant neopl. of rectum, rectosigmoid junction, and anus; Secondary malignant neopl. of respiratory and digestive systems; Malignant neopl. of trachea, bronchus, and lung; Malignant neopl. of stomach; Diseases of white blood cells;	3133
5	Chronic renal failure; Hypertensive renal disease; Disorders involving the immune mechanism; Anemia; Disorders of urethra and urinary tractk;	819
6	Diseases of endocardial structures; Chronic pulmonary heart disease; Heart failure; Diseases of endocardium; Diseases of lung;	2442

Table 4.2: The most representative diseases for each cluster for the Tertiarism dataset.

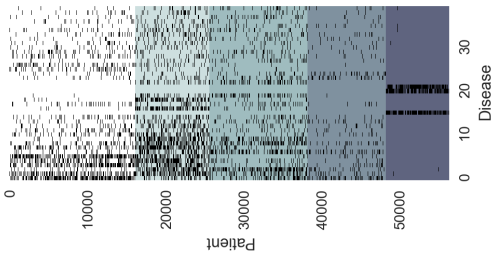
consider this family of diagnostics as characterizing this cluster. Cluster 5 groups kidney patients without arteriopathy, so similar to the first and second clusters but without previous infarction. Cluster 6 includes patients with cardiac valvular disease; normally they need heart surgery or interventional cardiology, but their associated complications are quite different from those in Clusters 1 and 2.

MIMIC dataset

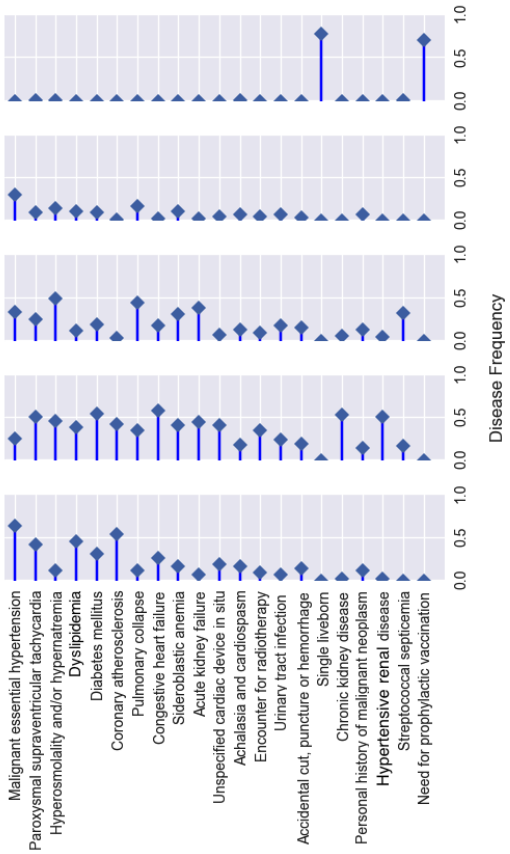
Cluster	Most Relevant Diagnostics	Cluster Size
1	Coronary atherosclerosis; Essential hypertension; Dyslipidemia; Mitral valve disorders; Acute myocardial infarction;	16120
2	Chronic kidney disease; Hypertensive renal disease; Congestive heart failure; Encounter for radiotherapy; Diabetes mellitus;	9504
3	Septicemia; Alcoholic fatty liver; Abscess of liver; Other anaphylactic reaction; Hyperosmolality and/or hypernatremia;	12559
4	Motor vehicle traffic accident; Accidental fall; Closed fracture; Closed fracture of cervical vertebra;6 Open wound;	10065
5	Single liveborn; Need for prophylactic vaccination; Observation for suspected infectious condition; Extreme immaturity; Perinatal jaundice;	8112

Table 4.3: The most representative diseases for each cluster for the MIMIC dataset.

The clusters obtained with the MIMIC dataset are plotted in Figures 4.5a and 4.5b, while Table 4.3 contains the most relevant diseases for the each cluster. Again, the clusters are neat, and clearly characterized under the clinical point of view. Cluster 1 is characterized by patients suffering of Dyslipidemia and hearth-related issues, Cluster 2 is characterized by kidney-related problems, while Cluster 3 seems to contain infections and liver-related issues. Cluster 4 contains patients who suffered traumatic events (like accidents and fractures) while Cluster 5 is essentially characterized by new-born patients.



(a)



(b)

Figure 4.5: 4.5b Disease-frequency chart for the MIMIC dataset. 4.5a The corresponding heatmap.

4.7 Conclusions

We have presented an efficient method for clustering high dimensional binary data, based on applying methods of moments to learn mixtures of independent Bernoulli variables. Under the clinical perspective, our method is able to find interesting and potentially useful patterns even in the simplest format of EHR, namely, diagnostic codes only. A strong point noted by clinicians on this clustering approach is that it provides a fresh look, without prejudice, based on an aseptic algorithm. A natural follow-up consists in including additional medical information to the features, like demographics, lab results, vital signs or medications. Also, we believe that including higher-level diagnostic groups among the features may improve the clarity of some patterns; these are more clearly defined in the more recent coding scheme ICD10 than in ICD9.

Part II

Hierarchical Methods of Moments

Chapter 5

A Method of Moments Robust to Model Misspecification

5.1 Introduction

In the previous chapters, we have presented several results showing how methods of moments can be used to learn from data latent variable models. In Chapters 3 and 4 we have presented practical applications of these methods, showing that they represent a promising framework, that can be successfully applied on complex, noisy real-world data, with performance that is often superior to those of existing, classic techniques. Without doubt, efficiency is one of the competitive advantages of methods of moments: requiring a single pass through the data – to calculate the moments – and working with small, low-dimensional tensors – thanks to the whitening step – they allow to deal with high dimensional massive datasets, returning appealing results in relatively short time.

At the same time, if the structure of the model generating the data is known, methods of moments often have the guarantee to recover a model that, asymptotically with the sample size, converges to the one generating the data. Consider for example a corpus of texts generated by a single topic model, then methods of moments – using the procedure described at Chapter 3 – enable us to learn a model that asymptotically approaches the one generating the texts we are observing. These guarantees have an intrinsic theoretical interest – opening a new learning perspective on several models, for which

previously no accurate learning techniques were known – but it is important to understand how they translate to the reality, and if they ensure that a good model will be returned by a method of moments, even with real world data. Real-world data in fact, may be generated by no finitely specified model, with an unknown structure, and will likely present outliers; in few words, real world data will always present model misspecifications. Any learning technique for latent variable models that aims at being competitive on real-world data, should be able to return the optimal model describing the data, according to some definition of optimality. This for example is what happens with the likelihood-based techniques, that aim at returning the model that (locally) maximizes the likelihood of the observed data. How do methods of moments behave in this setting? Do the guarantees that hold in theory, where the structure of the model generating the data is known, hold also when model misspecifications are present?

This kind of misspecification can happen for example when the number of latent states plugged into a learning algorithm is not enough to accurately describe the data we are observing. For example, we are given a corpus of texts dealing with k topics and we run a leaning algorithm to learn $l < k$ topics. In this setting, we expect to retrieve a model with l topics that optimally describes the data (according to some definition of *optimality*). Surprisingly, currently no theory describes the behavior of methods of moments in this setting, and consequently no guarantees of any kind are provided. In contrast, the model obtained by likelihood-based methods like EM is in this case reasonable and desirable: when asked for a small number of latent variables EM yields a model which is easy to interpret and can be useful for data visualization and exploration. An important application of low-dimensional learning can be found in mixture models, where latent class assignments provided by a simple model can be used to split the training data into disjoint datasets to which EM is applied recursively to produce a hierarchical clustering (Steinbach et al., 2000; Savaresi and Boley, 2001). The tree produced by such clusterings procedure provides a useful aid in data exploration and visualization.

In this chapter, we analyze the behavior of methods of moments when the number of requested latent states is not enough to accurately describe the data; we will show that most of the existing decomposition methods have no guarantees to return in this setting a meaningful model (Section 5.2). We

therefore introduce a novel decomposition method, that does presents this kind of guarantees, producing meaningful results even in misspecified settings (Section 5.3). Then, we propose to use this model as a building block of a hierarchical method of moments, a technique that will enable us to learn hierarchical representations of latent variable models, that will be applicable, for example, to perform hierarchical clustering (Section 5.4).

Our method will follow a different approach with respect to previous attempts to design methods of moments for misspecified models. Instead of looking for convex relaxations of existing methods of moments (Balle et al., 2012; Balle and Mohri, 2012; Quattoni et al., 2014) or analyzing the behavior of a method of moments with a misspecified number of latent states (Kulesza et al., 2014, 2015), we will generalize well-known simultaneous diagonalization approaches to tensor decomposition by phrasing the problem as a non-convex optimization problem. Despite its non-convexity, the hierarchical nature of our method allows for a fast accurate solution based on low-dimensional grid search. We test our method on synthetic and real-world datasets on the topic modeling task, showcasing the advantages of our approach and obtaining meaningful results.

5.2 Existing Decomposition Algorithms and the Misspecified Setting

In this section we recall the behavior of methods of moments, showing that existing decomposition algorithms have no guarantees to return meaningful models when the number of requested latent states is not enough to accurately describe the data.

Starting from a dataset $\mathcal{X}$, sampled from a certain latent variable model, a method of moments first recovers a set of (typically unbiased) estimators,

exhibiting a prescribed relation with the parameters of the models:

$$\mathbb{E}[\tilde{M}_1(\mathcal{X})] = M_1 = \sum_{i=1}^{k} \omega_i \mu_i, \tag{5.1}$$

$$\mathbb{E}[\tilde{M}_2(\mathcal{X})] = M_2 = \sum_{i=1}^{k} \omega_i \mu_i \otimes \mu_i, \tag{5.2}$$

$$\mathbb{E}[\tilde{M}_3(\mathcal{X})] = M_3 = \sum_{i=1}^{k} \omega_i \mu_i \otimes \mu_i \otimes \mu_i. \tag{5.3}$$

Then, it decomposes these moments in order to find parameters $(\tilde{M}, \tilde{\omega})$ that approximatively solve the equations above. In Chapter 1, we have seen several methods to solve these equations. All of them have been analyzed in the case where the algorithm only has access to noisy estimates of the moments. However, such analyses assume that the data was generated by a model from the hypothesis class, that the matrix M has rank k, and that this rank is known to the algorithm. In practice the dimension k of the latent variable can be cross-validated, but in many cases this is not enough: data may come from a model outside the class, or from a model with a very large true k. Besides, the moment estimates might be too noisy to provide reliabe estimates for large number of latent variables. It is thus frequent to use these algorithms to estimate $l < k$ latent variables. However, we will now see that existing algorithms are not robust in this setting, as they have not been designed to work in this regime, and there is no theoretical explanation of what their outputs will be.

When the value of k is known, the SVD method from Anandkumar et al. (2012b,a) observes that, given the whitening matrix[1] E, there exists a unique orthogonal matrix $O \in \mathbb{R}^{k \times k}$ such that $M\Omega^{1/2} = EO$, and that the same matrix O is also the common diagonalizer of all the whitened slices of M_3:

$$H_r = E^\dagger M_{3,r} E^{\dagger\top} = O \,\mathrm{diag}(m_r) O^\top. \tag{5.4}$$

Consequently, it calculates the singular vectors of a random linear combination of the matrices H_r to find O, and recovers M by solving the system of equations $M\Omega^{1/2} = EO$ – see Chapter 1 for more details. However, when a

[1]Recall that given the SVD $M_2 = USU^\top$, the whitening matrix is $E = US^{1/2} \in \mathbb{R}^{d \times k}$

value $l < k$ is used, the method will perform the whitening using the matrix $E_l^\dagger$ obtained from the low-rank SVD truncated at rank l: $M_2 \approx U_l S_l U_l^\top = E_l E_l^\top$. Then, it will compute the matrices $H_{l,r} = E_l^\dagger M_{3,r} E_l^{\dagger\top}$ for $r \in [d]$ that may not be jointly diagonalizable, and there is no theoretical justification of what what the result of this algorithm will be.

The hypothesis that the whitened slices of M_3 are jointly diagonalizable is also used by SVTD, that we discussed in Chapter 2. So, similarly to what is said above, SVTD will also lack of guarantees when plugged with a misspecified number of latent states.

The tensor power method (TPM) (Anandkumar et al., 2014) starts with a whitening step to transform M_3 into a symmetric orthogonally decomposable tensor

$$T = \sum_{i=1}^{k} \omega_i E^\dagger \mu_i \otimes E^\dagger \mu_i \otimes E^\dagger \mu_i \in \mathbb{R}^{k \times k \times k}. \tag{5.5}$$

The weights ω_i and vectors μ_i are then recovered from T using a tensor power method and inverting the whitening step. However, when a value $l < k$ is used, TPM will use E_l to whiten the tensor M_3 to a tensor $T_l \in \mathbb{R}^{l \times l \times l}$, which may T_l may not admit a symmetric orthogonal decomposition. Consequently, it is not clear what TPM will return in this case and there are no guarantees it will even converge. To see this, we present here a simple counterexample by constructing a tensor $M_3 \in \mathbb{R}^{3\times3\times3}$ whose $2 \times 2 \times 2$ whitening does not admit a symmetric orthogonal decomposition. Consider the following parameters

$$\mu_1 = \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix}, \quad \mu_2 = \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix}, \quad \mu_3 = \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix}, \; \omega = \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix}$$

from which one can recover a matrix M_2 and a tensor M_3 from equations (5.2) and (5.3), both of rank $k = 3$. Using the top 2 singular vectors and values of M_2, M_3 would be whitened to a $2 \times 2 \times 2$ tensor T with the following entries:

$$\begin{aligned}
(T)_{1,1,1} &= 2\left(\tfrac{1+\sqrt{3}}{2\sqrt{9+5\sqrt{3}}}\right)^3 + \left(\tfrac{2+\sqrt{3}}{\sqrt{9+5\sqrt{3}}}\right)^3, \\
(T)_{1,2,2} &= (T)_{2,1,2} = (T)_{2,2,1} = \tfrac{1+\sqrt{3}}{2\sqrt{9+5\sqrt{3}}}, \\
(T)_{1,2,1} &= (T)_{1,1,2} = (T)_{2,2,2} = (T)_{2,1,1} = 0.
\end{aligned}$$

To see that T is not orthogonally decomposable, one can check that in this case Eq. (5.6) from Lemma 5.2.1 is equivalent to $(T)_{1,1,1} = (T)_{2,2,1}$, which here does not hold. Hence T is not orthogonally decomposable.

Lemma 5.2.1 (Robeva (2016), Example 1.2.3)**.** *A $2 \times 2 \times 2$ symmetric tensor T is orthogonally decomposable if and only if its entries satisfy the following equation:*

$$(T)_{1,1,1}(T)_{2,2,1} + (T)_{2,1,1}(T)_{2,2,2} = (T)^2_{2,1,1} + (T)^2_{2,2,1}. \tag{5.6}$$

Similarly to what said above, also the decomposition methods that aim at directly finding the CP-decomposition of M_3 do not provide guarantees to return meaningful model parameters M when $l < k$.

ALS (Kolda and Bader, 2009) will return a matrix that approximately decomposes M_3, but no theory guarantees that the returned model parameters will properly describe the data.

The Random Projections method (see Chapter 1 and Kuleshov et al. 2015 for a more detailed explanation) finds M by jointly diagonalizing various random linear combinations of slices of M_3 without any whitening step. When the value of k is known to the algorithm, it provides – under certain incoherence assumptions – guarantees of provably decomposing M_3. Instead, when asked for $l < k$ latent states, it will return a matrix that nearly diagonalizes the slices of M_3, but again, no analysis is given for this setting – and no trivial explanations support the intuition that this matrix should represent a good model describing the data.

5.3 Simultaneous Diagonalization Based on Whitening and Optimization

This section presents a new decomposition algorithm to solve equations (5.1), (5.2) and (5.3) that we call Simultaneous Diagonalization based on Whitening and Optimization (SIDIWO). When asked to produce $l = k$ components in the noiseless setting, SIDIWO will return the same output as any of the methods discussed in Section 5.2. However, in contrast with those methods, SIDIWO will provide useful results with a clear interpretation even in a misspecified setting ($l < k$).

5.3.1 SIDIWO in the Realizable Setting

To derive our SIDIWO algorithm we first observe that in the noiseless setting and when $l = k$, the pair (M, ω) returned by all methods described in Section 5.2 is the solution of the optimization problem given in the following lemma.

Lemma 5.3.1. *Let $M_{3,r}$ be the r-th slice across the second mode of the tensor M_3 from (5.3) with parameters (M, ω). Suppose $rank(M) = k$ and let $\Omega = \mathrm{diag}(\omega)$. Then the matrix $(M\Omega^{1/2})^\dagger$ is the unique optimum (up to column rescaling) of the optimization problem*

$$\min_{D \in \mathcal{D}_k} \sum_{i \neq j} \left(\sum_{r=1}^{d} (D M_{3,r} D^\top)_{i,j}^2 \right)^{1/2}, \tag{5.7}$$

where $\mathcal{D}_k = \{D : D = (EO_k)^\dagger$ for some O_k s.t. $O_k O_k^\top = \mathbb{I}_k\}$ and E is the whitening matrix recalled in Section 5.2.

Proof. Consider the SVD of M_2:

$$M_2 = USU^\top$$

where $U \in \mathbb{R}^{d \times k}$ and $S \in \mathbb{R}^{k \times k}$ are obtained from the first k singular vectors and values. Define now the matrix $E = US^{1/2}$; then there exists a unique $k \times k$ orthogonal matrix O such that $M\Omega^{1/2} = EO$. This implies that the slices of M_3 can be rewritten as follows:

$$M_{3,r} = M\,\Omega^{1/2}\,\mathrm{diag}(m_r)\,\Omega^{1/2}\,M^\top = EO\,\mathrm{diag}(m_r)\,(EO)^\top.$$

Take now a generic matrix $D \in \mathcal{D}_k$; it can be written as

$$D = O_k^\top E^\dagger$$

for a certain orthonormal matrix O_k. So we have

$$D M_{3,r} D^\top = O_k^\top E^\dagger EO diag(m_r)\,(EO)^\top (O_k^\top E^\dagger)^\top = O_k^\top O diag(m_r)\, O O_k$$

This matrix is diagonal if and only if $O = O_k$, so the problem

$$\min_{D \in \mathcal{D}_k} \sum_{i \neq j} (\sum_{r=1}^{d} (D M_{3,r} D^\top)_{i,j}^2)^{1/2} \tag{5.8}$$

is optimized by $D = (M\Omega^{1/2})^\dagger = (EO)^\dagger$, which is the unique (up to a columns rescaling) feasible optimum.

□

Remark 5.3.1 (The role of the constraints (I))**.** Consider the cost function of Problem (5.7): in an unconstrained setting, there may be several matrices minimizing that cost. A trivial example is the zero matrix. A less trivial example is when the rows of D belong to the orthogonal complement of the column space of the matrix M. The constraint $D = (EO_k)^\dagger$ for some orthonormal matrix O_k first excludes the zero matrix from the set of feasible solutions, and second guarantees that all feasible solutions lay in the space generated by the columns of M.

Algorithm 3 SIDIWO: Simultaneous Diagonalization based on Whitening and Optimization

Require: M_1, M_2, M_3, the number of latent states l

1: Compute a SVD of M_2 truncated at the lth singular vector: $M_2 \approx U_l S_l U_l^\top$.

2: Define the matrix $E_l = U_l S_l^{1/2} \in \mathbb{R}^{d \times l}$.

3: Find the matrix $D \in \mathcal{D}_l$ optimizing Problem (5.7).

4: Find $(\hat{M}, \hat{\omega})$ solving $\begin{cases} \hat{M}\hat{\Omega}^{1/2} = D^\dagger \\ \hat{M}\hat{\omega}^\top = M_1 \end{cases}$

5: **return** $(\hat{M}, \hat{\omega})$

The problem (5.7) opens a new perspective on using simultaneous diagonalization to learn the parameters of a latent variable model. In fact, one could recover the pair (M, ω) from the relation $M\Omega^{1/2} = D^\dagger$ by first finding the optimal D and then individually retrieving M and ω by solving a linear system using the vector M_1. This approach, outlined in Algorithm 3, is an alternative to the ones presented in the literature up to now (even though in the noiseless, realizable setting, it will provide the same results). Similarly to existing methods, this approach requires to know the number of latent states. We will however see in the next section that Algorithm 3 provides meaningful results even when a misspecified number of latent states $l < k$ is used.

5.3.2 The Misspecified Setting

Algorithm 3 requires as inputs the low order moments M_1, M_2, M_3 along with the desired number of latent states l to recover. If $l = k$, it will return the exact model parameters (M, ω); we will now see that it will also provide meaningful results when $l < k$. In this setting, Algorithm 3 returns a pair $(\hat{M}, \hat{\omega}) \in \mathbb{R}^{d \times l} \times \mathbb{R}^l$ such that the matrix $D = (\hat{M}\hat{\Omega}^{1/2})^\dagger$ is optimal for the optimization problem

$$\min_{D \in \mathcal{D}_l} \sum_{i \neq j} \left(\sum_{r=1}^{d} (D M_{3,r} D^\top)_{i,j}^2 \right)^{1/2}. \tag{5.9}$$

Analyzing the space of feasible solutions (Theorem 5.3.1) and the optimization function (Theorem 5.3.2), we will obtain theoretical guarantees on what SIDIWO returns when $l < k$, showing that the trivial solutions are not feasible, and that, in the space of feasible solutions, SIDIWO's optima will approximate the true model parameters according to an intuitive geometric interpretation.

The role of the constraints (II)

The first step consists in analyzing the space of feasible solutions $\mathcal{D}_l$ when $l < k$. The observations outlined in Remark 5.3.1 still hold in this setting: the zero solution and the matrices laying in the orthonormal complement of M are not feasible. Furthermore, the following theorem shows that other undesirable solutions will be avoided.

Theorem 5.3.1. *Let $D \in \mathcal{D}_l$ with rows $d_1, ..., d_l$, and let $\mathbb{I}_{r,s}$ denote the $r \times s$ identity matrix. The following facts hold under the hypotheses of Lemma 5.3.1:*

1. *For any row d_i, there exists at least one column of M such that $\langle d_i, \mu_j \rangle \neq 0$.*
2. *The columns of any $\hat{M}$ satisfying $\hat{M}\hat{\Omega}^{1/2} = D^\dagger$ are a linear combination of those of M, laying in the best-fit l-dimensional subspace of the space spanned by the columns of M.*
3. *Let π be any permutation of $\{1, ..., d\}$, and let M_π and Ω_π be obtained by permuting the columns of M and Ω according to π. If $\langle \mu_i, \mu_j \rangle \neq 0$ for any i, j, then $((M_\pi \Omega_\pi^{1/2})\mathbb{I}_{k,l})^\dagger \notin \mathcal{D}_l$, and similarly $\mathbb{I}_{l,k}(M_\pi \Omega_\pi^{1/2})^\dagger \notin \mathcal{D}_l$.*

Proof. Let us recall the notation we are going to use. Consider the matrix M_2 and its SVD:

$$M_2 = USU^\top.$$

For any $l \leq k$, define $E_l = U_l S_l^{1/2} \in \mathbb{R}^{d \times l}$, where U_l and S_l are U and S truncated at the lth singular vector (recall: $U \in \mathbb{R}^{d \times k}$ and $S \in \mathbb{R}^{k \times k}$). We know that there exists an orthonormal matrix O such that

$$M\Omega^{1/2} = EO.$$

Let us prove the various points of the theorem.

1. Consider any matrix $D \in \mathcal{D}_l$. Then we will have, for an orthonormal O_l,

   $$D = (E_l O_l)^\dagger = O_l^\top S_l^{-1/2} U_l^\top.$$

   To prove the statement, it is enough to show that the matrix $C = DM\Omega^{1/2}$ has rank l. To see this, explicitly represent C:

   $$C = DM\Omega^{1/2} = O_l^\top S_l^{-1/2} U_l^\top EO = O_l^\top S_l^{-1/2} U_l^\top US^{1/2}O = O_l^\top \mathbb{I}_{l,k} O$$

   the fact that O and O_l are orthogonal proves the claim.

2. Consider again any matrix $D \in \mathcal{D}_l$, then

   $$D^\dagger = \hat{M}\hat{\Omega}^{1/2} = (E_l O_l) = U_l S_l^{1/2} O_l.$$

   The columns of U_l are the left singular vectors of $M\Omega^{1/2}$, that span the best fit l-dimensional subspace of the space generated by the columns of M.

3. To prove this we will proceed by contradiction. Assume that $(M\Omega^{1/2}\mathbb{I}_{k,l})^\dagger \in \mathcal{D}_l$; this means that there exists an orthonormal matrix O_l such that

   $$M\Omega^{1/2}\mathbb{I}_{k,l} = E_l O_l = E\mathbb{I}_{k,l} O_l.$$

   But $M\Omega^{1/2} = EO$, so

   $$EO\mathbb{I}_{k,l} = E\mathbb{I}_{k,l} O_l.$$

   This would imply that

   $$\mathbb{I}_{l,k} O \mathbb{I}_{k,l} = O_l$$

and so, for some $P \in \mathbb{R}^{k-l \times k-l}$

$$O = \left[\begin{array}{c|c} O_l & 0 \\ \hline 0 & P \end{array}\right].$$

Observe now that the matrix $Z = \Omega^{1/2} M^\top M \Omega^{1/2}$ has all the entries that are different from zero, by the hypothesis that $\langle \mu_i, \mu_j \rangle \neq 0$ for any i, j. However, we have that

$$\begin{aligned} Z = \Omega^{1/2} M^\top M \Omega^{1/2} &= O^\top S O \\ &= \left[\begin{array}{c|c} O_l^\top & 0 \\ \hline 0 & P^\top \end{array}\right] \left[\begin{array}{c|c} S_l & 0 \\ \hline 0 & S_{l,k} \end{array}\right] \left[\begin{array}{c|c} O_l & 0 \\ \hline 0 & P \end{array}\right] \\ &= \left[\begin{array}{c|c} O_l^\top S_l O_l & 0 \\ \hline 0 & P^\top S_{l,k} P \end{array}\right] \end{aligned}$$

where $S_{l,k}$ is the diagonal matrix with the last $k - l$ singular values. So Z has some zero entry. This contradiction proves the claim.

The proof of the fact that $\mathbb{I}_{l,k}(M_\pi \Omega_\pi^{1/2})^\dagger \notin \mathcal{D}_l$ is identical.

□

The second point of Theorem 5.3.1 states that the feasible solutions will lay in the best l-dimensional subspace approximating the one spanned by the columns of M. This has two interesting consequences: if the columns of M are not orthogonal, point 3 guarantees that $\hat{M}$ cannot simply be a sub-block of the original M, but rather a non-trivial linear combination of its columns laying in the best l-dimensional subspace approximating its column space.

Consider for example a single-topic model from Chapter 3 with k topics; in this case, when asked to recover $l < k$ topics, Algorithm 3 will not return a subset of the original k topics, but a matrix $\hat{M}$ whose columns gather the original topics via a non trivial linear combination: the original topics will all be represented in the columns of $\hat{M}$ with different weights.

When the columns of M are orthogonal, this space coincides with the space of the l columns of M associated with the l largest ω_i; in this setting, the matrix $(M_\pi \Omega_\pi^{1/2}) \mathbb{I}_{k,l}$ (for some permutation π) is a feasible solution and minimizes Problem (5.9). Thus, Algorithm 3 will recover the top l topics.

Interpreting the optima.

Let $\hat{M}$ be such that $D = (\hat{M}\hat{\Omega}^{1/2})^{\dagger} \in \mathcal{D}_l$ is a minimizer of Problem (5.9). In order to better understand the relation between $\hat{M}$ and the original matrix M, we will show that the cost function of Problem (5.9) can be written in an equivalent form, that unveils a geometric interpretation.

Theorem 5.3.2. *Let $d_1, ..., d_l$ denote the rows of $D \in \mathcal{D}_l$ and introduce the following optimization problem*

$$\min_{D \in \mathcal{D}_l} \sum_{i \neq j} \sup_{v \in \mathcal{V}_M} \sum_{h=1}^{k} \langle d_i, \mu_h \rangle \langle d_j, \mu_h \rangle \omega_h v_h \tag{5.10}$$

where $\mathcal{V}_M = \{v \in \mathbb{R}^k : v = \alpha^{\top} M, \text{where } \|\alpha\|_2 \leq 1\}$. Then this problem is equivalent to (5.9)*.*

Proof. Recall the considered problem:

$$\min_{D \in \mathcal{D}_l} \sum_{i \neq j} \sup_{v \in \mathcal{V}_M} \sum_{h=1}^{k} \langle d_i, \mu_h \rangle \langle d_j, \mu_h \rangle \omega_h v_h \tag{5.11}$$

where

$$\mathcal{V}_M = \{v \in \mathbb{R}^k : v = \alpha^{\top} M, \text{where } \|\alpha\|_2 \leq 1\}$$

Consider any $v \in \mathcal{V}_M$, then it admits the following representation, for some α with $\|\alpha\|_2 \leq 1$:

$$v = [\langle \alpha, \mu_1 \rangle, ..., \langle \alpha, \mu_k \rangle]^{\top}.$$

This allows the following chain of equalities on the cost function:

$$\begin{aligned}
\sum_{i\neq j} \sup_{v\in\mathcal{V}_M} \sum_{h=1}^{k} \langle d_i, \mu_h\rangle\langle d_j, \mu_h\rangle\omega_h v_h &= \sum_{i\neq j} \sup_{\alpha:\|\alpha\|_2\leq 1} \sum_{h=1}^{k} \langle \alpha, \mu_h\rangle\langle d_i, \mu_h\rangle\langle d_j, \mu_h\rangle\omega_h \\
&= \sum_{i\neq j} \sup_{\alpha:\|\alpha\|_2\leq 1} \sum_{r=1}^{d} \alpha_r \sum_{h=1}^{k} \mu_{h,r}\langle d_i, \mu_h\rangle\langle d_j, \mu_h\rangle\omega_h \\
&= \sum_{i\neq j} \sup_{\alpha:\|\alpha\|_2\leq 1} \sum_{r=1}^{d} \alpha_r (DM_{3,r}D^\top)_{i,j} \\
&= \sum_{i\neq j} \sup_{\alpha:\|\alpha\|_2\leq 1} \langle \alpha, t_{i,j}\rangle \\
&= \sum_{i\neq j} \|t_{i,j}\|_2
\end{aligned}$$

Where the vector $t_{i,j}$ is defined as

$$t_{i,j} = ((DM_{3,1}D^\top)_{i,j}, ..., (DM_{3,d}D^\top)_{i,j})$$

and the last equality has been obtained from the fact that, for any vector $w \in \mathbb{R}^k$, we have

$$\|w\| = \sup_{\alpha:\|\alpha\|_2\leq 1} \langle \alpha, w\rangle$$

This last equation proves our statement; in fact,

$$\sum_{i\neq j} \|t_{i,j}\|_2 = \sum_{i\neq j} (\sum_{r=1}^{d} (DM_{3,r}D^\top)_{i,j}^2)^{1/2}.$$

□

We now provide an interpretation of this theorem. First, observe that the cost function in Equation (5.10) prefers matrices D such that the vectors $u_i = [\langle d_i, \mu_1\sqrt{\omega_1}\rangle, ..., \langle d_i, \mu_k\sqrt{\omega_k}\rangle]$, $i \in [l]$, have disjoint support. This is a consequence of the $\sup_{v\in\mathcal{V}_M}$, and requires that, for each j, the entries $\langle d_i, \mu_j\sqrt{\omega_j}\rangle$ are close to zero for at least all but one of the various d_i. Consequently, each center will be almost orthogonal to all but one row of the optimal D; however the number of centers is greater than the number of rows of D, so the same row d_i may be nonorthogonal to various centers.

For illustration, consider again the single-topic model: a solution D to Problem (5.10) would have rows that should be as orthogonal as possible to some topics and as aligned as possible to the others; in other words, for a given topic j, the optimization problem is trying to set $\langle d_i, \mu_j \sqrt{\omega_j} \rangle = 0$ for all but one of the various d_i. Consequently, each column of the output $\hat{M}$ of Algorithm 3 should be in essence aligned with some of the topics and orthogonal to the others.

It is worth mentioning that the constraint set $\mathcal{D}_l$ forbids the trivial solutions such as the zero matrix, the pseudo-inverse of any subset of l columns of $M\Omega^{1/2}$, and any subset of l rows of $(M\Omega^{1/2})^\dagger$ (which all have an objective value of 0).

We remark that Theorem 5.3.2 does not require the matrix M to be full rank k: we only need it to have at least rank greater or equal to l, in order to guarantee that the constraint set $\mathcal{D}_l$ is well defined.

5.3.3 An optimal solution when $l = 2$.

While Problem (5.7) can be solved in general using an extension of the Jacobi technique (Cardoso and Souloumiac, 1996; Bunse-Gerstner et al., 1993), we provide a simple and efficient method for the case $l = 2$. This method will then be used to perform hierarchical topic modeling in Section 5.4. When $l = 2$, Equation (5.9) can be solved optimally with few simple steps; in fact, the following theorem shows that solving (5.9) is equivalent to minimizing a continuous function on the compact one-dimensional set $I = [-1, 1]$, which can easily be done by griding I. Using this in Step 3 of Algorithm 3, one can efficiently compute an arbitrarily good approximation of the optimal matrix $D \in \mathcal{D}_2$.

Theorem 5.3.3. *Consider the continuous function* $F(x) = c_1 x^4 + c_2 x^3 \sqrt{1-x^2} + c_3 x \sqrt{1-x^2} + c_4 x^2 + c_5$, *where the coefficients* $c_1, ..., c_5$ *are functions of the entries of* M_2 *and* M_3. *Let* a *be the minimizer of* F *on* $[-1, 1]$, *and consider the matrix*

$$O_a = \begin{bmatrix} \sqrt{1-a^2} & a \\ -a & \sqrt{1-a^2} \end{bmatrix} .$$

Then, the matrix $D = (E_2 O_a)^\dagger$ *is a minimizer of Problem* (5.9) *when* $l = 2$.

Proof. First, observe that the set of 2×2 orthonormal matrices can be parametrized as

$$O_a = \begin{bmatrix} \sqrt{1-a^2} & a \\ -a & \sqrt{1-a^2} \end{bmatrix}, \text{ for } a \in [-1, 1]. \tag{5.12}$$

The set $\mathcal{D}_2$ can thus be rewritten in function of a, as

$$\mathcal{D}_2 = \{D : D = (E_2 O_a)^\dagger \text{ for } a \in [-1, 1]\}.$$

and Problem (5.7) can be rewritten as

$$\min_{a \in [-1,1]} \sum_{r=1}^{d} 2(O_a^\top H_{2,r} O_a)_{1,2}^2$$

where

$$H_{2,r} = E_2^\dagger M_{3,r} E_2^{\dagger\top}, \quad for\ r = \{1, ..., d\} \tag{5.13}$$

and where we used the fact that $O_a^\top H_{2,r} O_a$ is symmetric. We can then write

$$(O_a^\top H_{2,r} O_a)_{1,2}^2 = c_1^{(r)} a^4 + c_2^{(r)} a^3 \sqrt{1-a^2} + c_3^{(r)} a \sqrt{1-a^2} + c_4^{(r)} a^2 + c_5^{(r)}$$

where the coefficients can be written as

$$c_1^{(r)} = 4h^2 - f^2, \quad c_2^{(r)} = -4fh, \quad c_3^{(r)} = 2fh, \quad c_4^{(r)} = f^2 - 4h^2, \quad c_5^{(r)} = h^2$$

with $h = (H_{2,r})_{1,2}$ and $f = (H_{2,r})_{1,1} - (H_{2,r})_{2,2}$. Letting $c_j = \sum_{r=1}^{d} c_j^{(r)}$ for $j \in \{1, ..., 5\}$ it follows that optimizing Problem (5.7) is equivalent to minimizing the following smooth real function

$$F(a) = c_1 a^4 + c_2 a^3 \sqrt{1-a^2} + c_3 a \sqrt{1-a^2} + c_4 a^2 + c_5.$$

□

5.4 Case Study: Hierarchical Topic Modeling

In this section, we show how SIDIWO can be used to efficiently recover hierarchical representations of latent variable models. Given a latent variable model with k states, our method allows to recover a pair $(\hat{M}, \hat{\omega})$ from estimate of the moments M_1, M_2 and M_3, where the l columns of $\hat{M}$ offer a synthetic

representation of the k original centers. We will refer to these l vectors as *pseudo-centers*: each pseudo-center is representative of a group of the original centers. Consider the case $l = 2$. A dataset $\mathcal{X}$ of n samples can be split into two smaller subsets according to their similarity to the two pseudo-centers. Formally, this assignment is done using Maximum A Posteriori (MAP) to find the pseudo-center giving maximum conditional likelihood to each sample – as described in Remark 3.3.2 for the example of the single-topic model. The splitting procedure can be iterated recursively to obtain a divisive binary tree, leading to a hierarchical clustering algorithm. While this hierarchical clustering method can be applied to any latent variable model that can be learned with the tensor method of moments (e.g. Latent Dirichlet Allocation), we present it here for the single-topic model for the sake of simplicity.

We consider a corpus $\mathcal{X}$ of n texts encoded as in Section 3.2 and we split $\mathcal{X}$ into two smaller corpora according to their similarity to the two pseudo-centers in two steps and use MAP assignment to assign each text x to a pseudo-center. This process is summarized in Algorithm 4. Once the corpus

Algorithm 4 Splitting a corpus into two parts

Require: A corpus of texts $\mathcal{X} = \{x^{(1)}, ..., x^{(n)}\}$.

1: Estimate $\tilde{M}_1$, $\tilde{M}_2$ and $\tilde{M}_3$.
2: Recover $l = 2$ pseudo-centers with Algorithm 3 .
3: **for** $i \in [n]$ **do**
4: Assign the text $x^{(i)}$ to the cluster $Cluster(i) = \text{argmax}_j \, \mathbb{P}(X = x^{(i)} | Y = j, \hat{\omega}, \hat{M})$, where $\mathbb{P}(X|Y = j, \hat{\omega}, \hat{M})$ is the multinomial distr. associated to the j-th pseudo-center (Equation (3.4)).
5: **end for**
6: **return** The cluster assignments $Cluster$.

$\mathcal{X}$ has been split into two subsets $\mathcal{X}_1$ and $\mathcal{X}_2$, each of these subsets may still contain the full set of topics but the topic distribution will differ in the two: topics similar to the first pseudo-center will be predominant in the first subset, the others in the second. By recursively iterating this process, we obtain a binary tree where topic distributions in the nodes with higher depth are expected to be more concentrated on fewer topics.

In the next sections, we assess the validity of this approach on both synthetic

Method	Adj. Rand Idx Mean	St. dev.	Run. Time
ALS	0.70	0.17	1.9 sec.
TPM	0.93	0.06	1.2 sec.
SVD	0.52	0.13	**0.1 sec**.
Rand. Proj.	0.72	0.06	16 min.
SVTD	0.85	0.1	0.2 sec
SIDIWO	**0.98**	**0.01**	0.4 sec.

Table 5.1: Table 5.1 reports the average and standard deviation over 10 runs of the clustering accuracy for the various methods, along with average running times.

and real-world data.

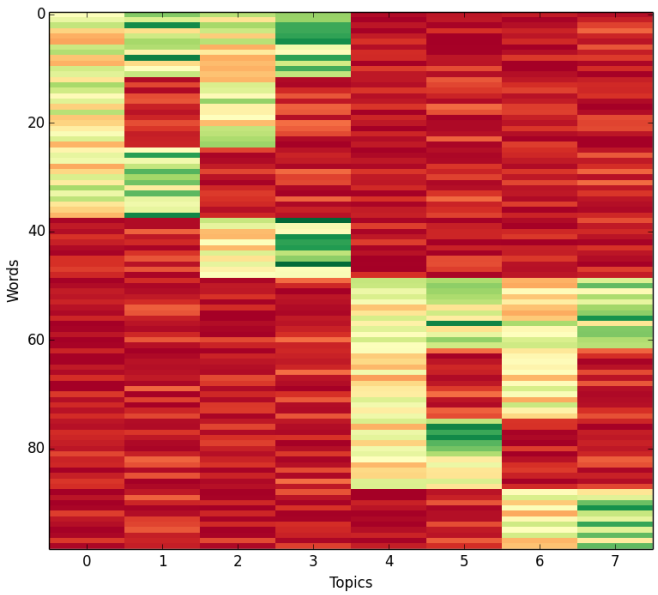
Experimental setting: The experiments in this section have been performed in Python 2.7, using *numpy* (Walt et al., 2011) library for linear algebra operations, with the exception of the implementation of the method from Kuleshov et al. (2015), for which we used the author's Matlab implementation:[2]. All the experiments were run on a MacBook Pro with an Intel Core i5 processor. The implementation of the described algorithms have been publicly disclosed.[3]

5.4.1 Experiment on Synthetic Data

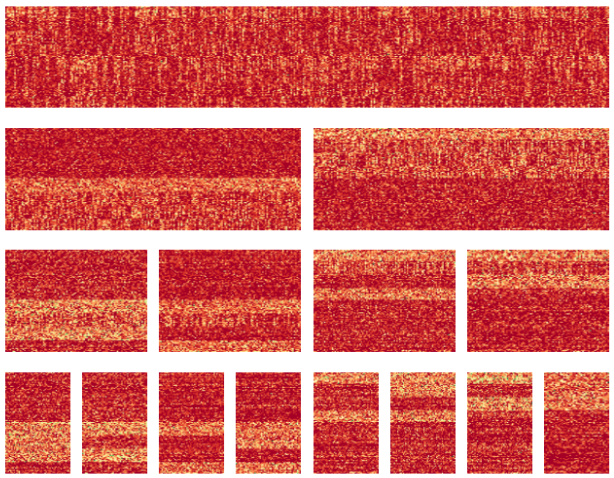
In order to test the ability of SIDIWO to recover latent structures in data, we generate a dataset distributed as a single topic model (with a vocabulary of 100 words) whose 8 topics have an intrinsic hierarchical structure depicted in Figure 5.1a. In this figure, topics are on the x-axis, words on the y-axis, and green (resp. red) points represents high (resp low) probability. We see for example that the first 4 topics are concentrated over the 1st half of the vocabulary, and that topics 1 and 2 have high probability on the

[2] `https://github.com/kuleshov/tensor-factorization`

[3] `https://github.com/mruffini/Hierarchical-Methods-of-Moments`.



(a)



(b)

Figure 5.1: Figure 5.1a provides a visualization of the topics used to generate the sample. Figure 5.1b represents the hierarchy recovered with the proposed method.

1st and 3rd fourth of the words while for the other two it is on the 1st and 4th.

We generate 400 samples according to this model and we iteratively run Algorithm 4 to create a hierarchical binary tree with 8 leaves. We expect leaves to contain samples from a unique topic and internal nodes to gather similar topics. Results are displayed in Figure 5.1b where each chart represents a node of the tree (child nodes lay below their parent) and contains the heatmap of the samples clustered in that node (x-axis corresponds to samples and y-axis to words, red points are infrequent words and clear points frequent ones). The results are as expected: each leaf contains samples from one of the topics and internal nodes group similar topics together.

We compare the clustering accuracy of SIDIWO with other methods using the Adjusted Rand Index (Hubert and Arabie, 1985) of the partition of the data obtained at the leaves w.r.t the one obtained using the true topics; comparisons are with the flat clustering on $k = 8$ topics with TPM, the method from Anandkumar et al. (2012b) (SVD), the one from Kuleshov et al. (2015) (Rand. Proj.), ALS from Kolda and Bader (2009) – where ALS is applied to decompose a whitened $8 \times 8 \times 8$ tensor T, calculated as in Equation (5.5) – and SVTD, from Chapter 2. We repeat the experiment 10 times with different random samples and we report the average results in Table 5.1; SIDIWO always recovers the original topic almost perfectly, unlike competing methods. One intuition for this improvement is that each split in the divisive clustering helps remove noise in the moments.

5.4.2 Experiment on NIPS Conference Papers 1987-2015

We consider the full set of NIPS papers accepted between 1987 and 2015, containing $n = 11463$ papers (Perrone et al., 2017). We assume that the papers are distributed according to a single topic model, we keep the $d = 3000$ most frequent words as vocabulary and we iteratively run Algorithm 4 to create a binary tree of depth 4. The resulting tree is shown in Figure 5.2 where each node contains the most *relevant* words of the cluster (see Equation 3.8, Chapter 3). The leaf clustering and the whole hierarchy are easy to interpret. Looking at the leaves, we can easily hypothesize the dominant topics for the 8 clusters. From left to right we have: [image processing, probabilistic models], [neuroscience, neural networks], [kernel methods, algorithms], [online optimization, reinforcement learning]. Also,

each node of the lower levels gathers meaningful keywords, confirming the ability of the method to hierarchically find meaningful topics. The running time for this experiment was 59 seconds.

5.4.3 Experiment on Wikipedia Mathematics Pages

We consider a subset of the full Wikipedia corpus, containing all articles ($n = 809$ texts) from the following math-related categories: linear algebra, ring theory, stochastic processes and optimization. We remove a set of 895 stop-words, keep a vocabulary of $d = 3000$ words and run SIDIWO to perform hierarchical topic modeling (using the same methodology as in the previous section). The resulting hierarchical clustering is shown in Figure 5.3 where we see that each leaf is characterized by one of the dominant topics: [ring theory, linear algebra], [stochastic processes, optimization] (from left to right). It is interesting to observe that the first level of the clustering has separated pure mathematical topics from applied ones. The running time for this experiment was 6 seconds.

5.5 Conclusions

In this chapter we proposed SIDIWO, a novel decomposition algorithm that generalizes recent methods of moments relying on tensor decomposition. While previous algorithms lack robustness to model misspecification, SIDIWO provides meaningful results even in misspecified settings. Moreover, SIDIWO can be used to perform hierarchical method of moments estimation for latent variable models. In particular, we showed through hierarchical topic modeling experiments on synthetic and real data that SIDIWO provides meaningful results while being very computationally efficient. A natural future work is to investigate the capability of the proposed hierarchical method to learn overcomplete latent variable models, where the number of latent states k is higher than the number of observable features d. Another area of interest consists in comparing the learning performance of SIDIWO with those of other existing methods of moments in the realizable setting.

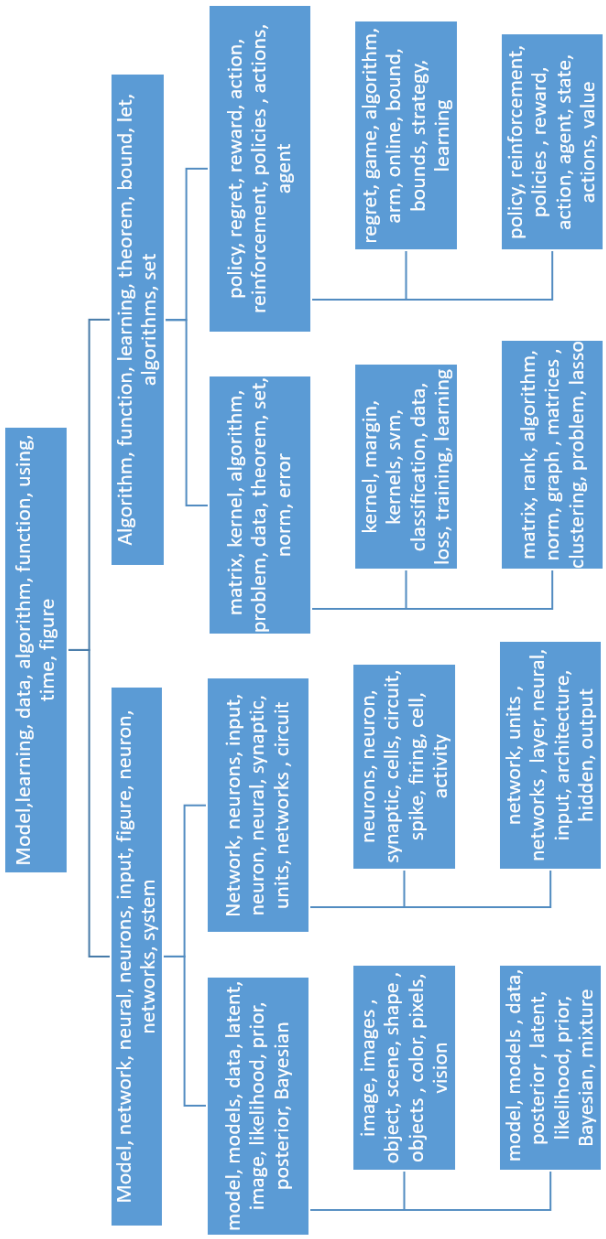


Figure 5.2: Experiment on the NIPS dataset.

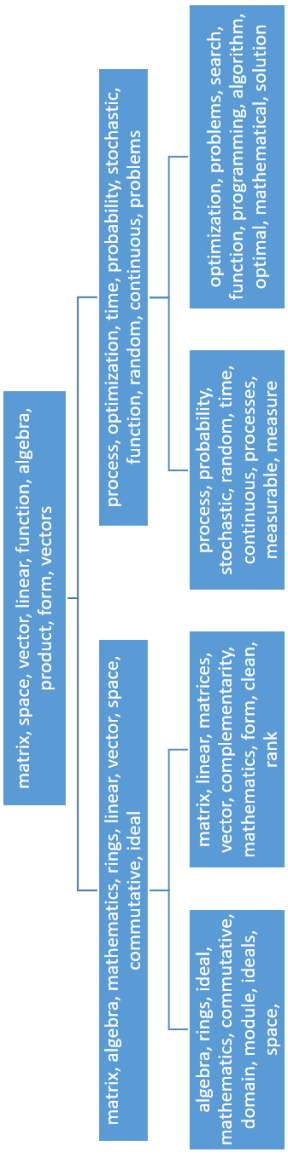


Figure 5.3: Experiment on the Wikipedia Mathematics Pages dataset.

Chapter 6

Hierarchical Methods of Moments for Clustering High-Dimensional Binary Data

6.1 Introduction

In the previous chapters of this thesis, we have seen how methods of moments can be used to learn from data latent variable models. In all of them, we have performed the assumption that data is generated by a certain model, whose structure is known either entirely – as in Chapters 1, 3 and 4 – or partially – like in Chapter 5 – to the user. In most of these settings, methods of moments present guarantees of learning accuracy, ensuring that the learned model asymptotically approaches the one generating the data. Unfortunately, these guarantees fail to ensure that a meaningful model describing the data is always returned by a method of moments.

In fact, consider the standard approach of method of moments, aiming at learning a model from a certain dataset $\mathcal{X}$: first, the user assumes that the data is generated by a certain model with k latent states. Then, (unbiased) estimators for the moments $\tilde{M}_1$, $\tilde{M}_2$ and $\tilde{M}_3$ are retrieved and decomposed to recover a set of model parameters, namely a pair $(\tilde{M}, \tilde{\omega})$, with $\tilde{M} = [\tilde{\mu}_1, ..., \tilde{\mu}_k]$. If the model assumptions are properly specified, and data is generated by a model with parameters (M, ω) and k states, then guarantees exist ensuring that the estimated pair $(\tilde{M}, \tilde{\omega})$ will converge to the true pair (M, ω) together

with the sample size. However, this does not mean that a good model will always be returned, as the sample size may not be big enough to guarantee that an estimated center $\tilde{\mu}_i$ is a good estimator for a true center μ_i. Furthermore, despite sample accuracy bounds exist (see for example Chapter 3), it is often difficult to explicitly assess if the sample size is large enough to ensure good learning accuracy. Conversely, guarantees that an algorithm would always return something meaningful with respect to the data, even if the sample size is small would be desirable. For example, the methods based on maximum likelihood principle aim at returning the optimal model to describe the data we are observing, according to a precise definition of optimality – namely the one described by the likelihood function. Moment-based methods instead do not provide such kind of interpretation: an understanding of the behavior of the output of methods of moments with respect to the data that are provided as input is still missing.

If data comes from a model whose structure is unknown to the user, or does not come from any finite model – like in the case of real world data – the situation becomes even more tricky: the limitations described above are amplified by model misspecifications, and additional complications raise. In fact, for methods of moments to work, it is crucial to deal with moments whose expectations are in a prescribed relation with respect to the parameters of the model one wants to learn:

$$\mathbb{E}[\tilde{M}_1] = M_1 = \sum_{i=1}^{k} \omega_i \mu_i, \tag{6.1}$$

$$\mathbb{E}[\tilde{M}_2] = M_2 = \sum_{i=1}^{k} \omega_i \mu_i \otimes \mu_i, \tag{6.2}$$

$$\mathbb{E}[\tilde{M}_3] = M_3 = \sum_{i=1}^{k} \omega_i \mu_i \otimes \mu_i \otimes \mu_i. \tag{6.3}$$

However, in real-world scenarios, data will not be exactly generated by any model, the model can be unknown or misspecified, or the calculation of the moments may not exhibit the desired expectations provided above. In all these settings, there are no guarantees nor obvious reasons to expect the vectors $\tilde{\mu}_i$ to be good centers of a model that approximately describes the data, even when the sample size is big. This limitation is fostered by the fact that asymptotically convergent estimates of the moments are known only for

latent variable models with a very simple structure, which imposes hypotheses on the data that are systematically violated.

The paragraphs above explain why the guarantees that are typically provided by methods of moments tend to lose part of their value when it comes to ensuring that a meaningful model is learned, especially when dealing with real-world data. At the same time, guarantees ensuring that the output of a method of moments meaningfully describes the data, may be provided by introducing a data-oriented interpretation of these methods, linking geometrically the output of a method with the data that are provided as input. Having a clear intuition on what the output of a method will be with respect to the data, is in fact a key step to ensure that this output makes sense even in the challenging setting where few data is available or model misspecifications are present.

In this chapter we present a variation of methods of moments that aims at overcoming the limitations listed above, while maintaining the efficiency properties typical of moment-based algorithms. Our approach leverages on transforming methods of moments from model-learning algorithms - i.e. methods to learn a specific latent variable model from data - to task-specific algorithms - i.e. methods aimed at directly performing a specific task - focusing in this chapter on one task: hierarchical clustering. To do this, in Section 6.2 we rewrite the method of moments as an optimization problem – leveraging on the cost function introduced by SIDIWO in Chapter 5 – and we show that if the moments are calculated in a simple, standardized way independent of the specific latent variable model, an intuitive relation between the optima and the input data can be found. Furthermore, this relation automatically suggests a clustering rule to split data into clusters, which, as we will see later, is particularly suitable for high-dimensional binary data (Section 6.3). This relation is geometric, and can be used to split the data into clusters only leveraging on geometric considerations, introducing in this way a model-free, geometry-based bisecting algorithm, that will be used as a building block of a divisive hierarchical clustering algorithm, that we will call Data-centric Simultaneous Diagonalization based on Whitening and Optimization (D-SIDIWO).

This geometry-based approach substantially differs from the one traditionally followed by methods of moments and model-based clustering. A traditional approach would have first learned a model with a method of moments and

then used that model to assign the samples to the cluster corresponding to the latent state that maximizes the conditional likelihood, via a Maximum A Posteriori (MAP) assignment. This is the approach we followed in Chapter 4, where we used a naive Bayes model and in Chapter 5, where we used SIDIWO to recursively learn a mixture model with two latent states which would then used to bisect a dataset.

Beside the novel perspective on methods of moments, D-SIDIWO has two benefits. First, a model-specific formulation of the moments – which limits the applicability of methods of moments to models with a simple structure – is not required. Second, it is grounded on a data-oriented interpretation of the results, where the entities that are used to cluster the data, have a clear and meaningful relation with the input data, even if the model that is generating the data is unknown, or the sample size is relatively small. While removing the model-dependency on the moment-based algorithm, our approach does not exclude to retrieve a latent variable model - in particular a mixture model - from the results of clustering.

In Section 6.4 we will apply this method to high-dimensional healthcare records, the same used in Chapter 4. D-SIDIWO in fact looks particularly suitable for this kind of datasets, as its hierarchical approach has two major advantages. First, users (doctors or data scientists) do not have to select the number of clusters, a non-trivial task, especially when this number is high and the population is heterogeneous. Second, it allows an easy and natural way to navigate trough a high number of clusters that are organized in meaningful hierarchical trees. We will see that the proposed approach provides a superior stability while providing best-in-class quality of the clusters, outperforming both existing methods of moments and traditional clustering algorithms. Also, we analyze the results of our technique under the qualitative point of view, obtaining meaningful clinical results.

6.2 A data-centric Interpretation of SIDIWO

SIDIWO – see Chapter 5 – is a decomposition algorithm to solve Equations (6.1),(6.2) and (6.3) and learn the parameters of a latent variable model. Starting from the values of the input moments M_1, M_2, M_3 and the number of latent states of the model k, SIDIWO recovers the parameters (M, ω) of

the desired model by solving the following constrained optimization problem:

$$\min_{D \in \mathcal{D}_k} \sum_{i \neq j} \left(\sum_{r=1}^{d} (D M_{3,r} D^\top)_{i,j}^2 \right)^{1/2}, \tag{6.4}$$

$$\mathcal{D}_k = \{D : D = (EO_k)^\dagger \text{ for some } O_k \text{ s.t. } O_k O_k^\top = \mathbb{I}_k\},$$

where $\mathbb{I}_k$ is the $k \times k$ identity matrix, and recovers the parameters of the model from the relation $M\Omega^{1/2} = EO$.

In this section, we present an analysis of SIDIWO under a data-centric point of view. To do this, we analyze the optimization problem (6.4) solved by SIDIWO and show that, if the moments are calculated in simple, standardized way, the optima present a meaningful relation with the input data. This relation is then used in Section 6.3 to introduce a method to split data into clusters. For ease of explanation, we focus on the special case where $k = 2$. While extensions to the case $k \geq 2$ are immediate, the studied example is an instrumental building block of the top-down bisecting clustering algorithm we are going to propose.

Given a dataset $\{x^{(i)}\}_{i=1}^n$ of independent samples, we call *raw moments* the following quantities:

$$\hat{M}_2 = \sum_{i=1}^{n} \frac{x^{(i)} \otimes x^{(i)}}{n} \in \mathbb{R}^{d \times d}, \tag{6.5}$$

$$\hat{M}_3 = \sum_{i=1}^{n} \frac{x^{(i)} \otimes x^{(i)} \otimes x^{(i)}}{n} \in \mathbb{R}^{d \times d \times d}. \tag{6.6}$$

Plugging these moments into SIDIWO using $k = 2$, the cost function in Equation (6.4) will become

$$\min_{D \in \mathcal{D}_2} \left(\sum_{r=1}^{d} (D \hat{M}_{3,r} D^\top)_{1,2}^2 \right)^{1/2}, \tag{6.7}$$

$$\mathcal{D}_2 = \{D : D = (E_2 O_2)^\dagger \text{ for some } O_2 \text{ s.t. } O_2 O_2^\top = \mathbb{I}_2\},$$

where $E_2 \in \mathbb{R}^{d \times 2}$ is the whitening matrix deduced from the truncated SVD of $\hat{M}_2$. Given a feasible solution $D \in \mathbb{R}^{2 \times d}$ it is possible to express it terms of

its rows: $D = [d_1|d_2]^\top$. The following theorem, provides a relation between d_1, d_2 and the input data, which is analyzed and interpreted in Section 6.3.

Theorem 6.2.1. *Let d_1 and d_2 denote the rows of a feasible $D \in \mathcal{D}_2$. Then, the optimization problem at Equation* (6.7) *is equivalent to the following one:*

$$\min_{D\in\mathcal{D}_2} \sup_{v\in\mathcal{V}_n} \sum_{h=1}^{n} v_h \langle d_1, x^{(h)}\rangle \langle d_2, x^{(h)}\rangle$$

where $\mathcal{V}_M = \{v \in \mathbb{R}^n : v_h = \langle \alpha, x^{(h)}\rangle$, where $\|\alpha\|_2 \leq 1\}$. Also, for any $D = [d_1|d_2]^\top \in \mathcal{D}_2$ the following relation holds:

$$\begin{cases} \sum_i^n \frac{\langle d_j, x^{(i)}\rangle^2}{n} = 1, & j = 1, 2 \\ \sum_i^n \frac{\langle d_1, x^{(i)}\rangle \langle d_2, x^{(i)}\rangle}{n} = 0. \end{cases}$$

Proof of Theorem 6.2.1

The statement of Theorem 6.2.1 is a consequence of the following two lemmas.

Lemma 6.2.1. *Let $D = [d_1|d_2]^\top \in \mathcal{D}_2$ be a feasible solution. Then the following relation holds:*

$$\begin{cases} \sum_i^n \frac{\langle d_1, x^{(i)}\rangle^2}{n} = 1 \\ \sum_i^n \frac{\langle d_2, x^{(i)}\rangle^2}{n} = 1 \\ \sum_i^n \frac{\langle d_1, x^{(i)}\rangle \langle d_2, x^{(i)}\rangle}{n} = 0 \end{cases}$$

Proof. We first recall the definition of $\mathcal{D}_2$:

$$\mathcal{D}_2 = \{D : D = (E_2 O_2)^\dagger \text{ for some } O_2 \text{ s.t. } O_2 O_2^\top = \mathbb{I}_2\},$$

It is enough to prove that for any feasible matrix $D \in \mathcal{D}_2$ the following relation holds:

$$D\hat{M}_2 D^\top = \mathbb{I}_2 \tag{6.8}$$

where $\mathbb{I}_2$ is the 2×2 identity matrix. To prove this, we consider the SVD of $\hat{M}_2$,

$$\hat{M}_2 = USU^\top, \tag{6.9}$$

where $U \in \mathbb{R}^{d\times l}$ is the matrix whose columns are the singular vectors and $S \in \mathbb{R}^{l\times l}$ is a diagonal matrix with the singular values in the diagonal entries.

l is such that $l \leq d$ and is the rank of $\hat{M}_2$. It is obvious from Equation (6.9) that

$$E^\dagger \hat{M}_2 (E^\dagger)^\top = \mathbb{I}_l$$

where $E = US^{1/2}$ and $E^\dagger$ is its Moore-Penrose pseudoinverse:

$$E^\dagger = (E^\top E)^{-1} E^\top.$$

The matrix E_2 that we use in the definition of $\mathcal{D}_2$ can thus be obtained by keeping only the top two columns of E:

$$E_2 = US^{1/2}\mathbb{I}_{l,2} = E\mathbb{I}_{l,2}$$

where $\mathbb{I}_{l,2}$ is the $l \times 2$ identity matrix. Now, consider any $D = (E_2 O_2)^\dagger \in \mathcal{D}_2$:

$$\begin{aligned}
D =& (E_2 O_2)^\dagger \\
=& (O_2^\top E_2^\top E_2 O_2)^{-1} O_2^\top E_2^\top \\
=& O_2^\top \mathbb{I}_{2,l} S^{-1} \mathbb{I}_{l,2} O_2 O_2^\top E_2^\top \\
=& O_2^\top \mathbb{I}_{2,l} S^{-1} \mathbb{I}_{l,2} E_2^\top \\
=& O_2^\top \mathbb{I}_{2,l} S^{-1} E^\top \\
=& O_2^\top \mathbb{I}_{2,l} S^{-1} (S^{1/2}) U^\top \\
=& O_2^\top \mathbb{I}_{2,l} (S^{-1/2}) U^\top \\
=& O_2^\top \mathbb{I}_{2,l} E^\dagger
\end{aligned}$$

From this, we can observe that

$$D\hat{M}_2 D^\top = O_2^\top \mathbb{I}_{2,l} E^\dagger \hat{M}_2 (E^\dagger)^\top \mathbb{I}_{l,2} O_2 = \mathbb{I}_2.$$

which proves the thesis. □

Lemma 6.2.2. *Consider Algorithm 3, applied to the moments defined in Equations* (6.5) *and* (6.6)*. Then, the cost function*

$$\min_{D \in \mathcal{D}_2} \left(\sum_{r=1}^{d} (D\hat{M}_{3,r} D^\top)_{1,2}^2 \right)^{1/2} \tag{6.10}$$

is equivalent to

$$\min_{D \in \mathcal{D}_2} \sup_{v \in \mathcal{V}_n} \sum_{h=1}^{n} v_h \langle d_1, x^{(h)} \rangle \langle d_2, x^{(h)} \rangle$$

where

$$\mathcal{V}_M = \{ v \in \mathbb{R}^n : v_h = \langle \alpha, x^{(h)} \rangle, \textit{where } \|\alpha\|_2 \leq 1 \}.$$

Proof. Define the vector t as follows:

$$t = ((D\hat{M}_{3,1}D^\top)_{1,2}, ..., (D\hat{M}_{3,d}D^\top)_{1,2})$$

and observe that problem (6.10) is trying to minimize the euclidean norm of t. From the fact that, for any vector $w \in \mathbb{R}^d$, we have

$$\|w\| = \sup_{\alpha:\|\alpha\|_2\leq 1} \langle \alpha, w\rangle$$

we obtain

$$\begin{aligned}
n\|t\|_2 &= n \sup_{\alpha:\|\alpha\|_2\leq 1} \langle \alpha, t\rangle \\
&= n \sup_{\alpha:\|\alpha\|_2\leq 1} \sum_{r=1}^{d} \alpha_r (D\hat{M}_{3,r}D^\top)_{1,2} \\
&= \sup_{\alpha:\|\alpha\|_2\leq 1} \sum_{r=1}^{d} \alpha_r \sum_{h=1}^{n} (x^{(h)})_r \langle d_1, x^{(h)}\rangle \langle d_2, x^{(h)}\rangle \\
&= \sup_{\alpha:\|\alpha\|_2\leq 1} \sum_{h=1}^{n} \langle \alpha, x^{(h)}\rangle \langle d_1, x^{(h)}\rangle \langle d_2, x^{(h)}\rangle \\
&= \sup_{v\in\mathcal{V}_n} \sum_{h=1}^{n} v_h \langle d_1, x^{(h)}\rangle \langle d_2, x^{(h)}\rangle
\end{aligned}$$

where

$$\mathcal{V}_M = \{v \in \mathbb{R}^n : v_h = \langle \alpha, x^{(h)}\rangle, \text{where } \|\alpha\|_2 \leq 1\}$$

and $(x^{(h)})_r$ is the rth entry of $x^{(h)}$. From this we obtain the thesis.

□

6.3 A Hierarchical Clustering Algorithm

Theorem 6.2.1 shows an interesting interpretation of the optimization problem solved by SIDIWO. Introduce two vectors w_1 and w_2 defined as follows:

$$w_i = [\langle d_i, x^{(1)}\rangle, ..., \langle d_i, x^{(n)}\rangle], \quad i = 1, 2$$

then, the cost function of SIDIWO is pushing them to have disjoint support, preferring solutions where each sample is nearly orthogonal to d_1 or to d_2. At

the same time, the constraint is pushing each of the vectors d_i to be aligned to the points to which it is not nearly-orthogonal. Each vector d_i can be seen as a linear discriminator, that is optimized in order to be as orthogonal as possible to a certain set of points of the dataset, with the constraint of being the most aligned as possible to some of them. We can thus group the points into two clusters: those that are represented by d_1 and those that are represented by d_2. d_2 will be optimized to be orthogonal to the first cluster, d_1 to the second. This automatically suggests a clustering rule for our points:

$$Cluster(x^{(i)}) = \underset{1,2}{\text{argmax}}(|\langle d_1, x^{(i)}\rangle|, |\langle d_2, x^{(i)}\rangle|) \tag{6.11}$$

Using these observations we are able to modify SIDIWO and deduce from it a bisecting algorithm. The adapted algorithm will take as input a dataset $\{x^{(i)}\}_{i=1}^n$, calculate the raw moments and from them deduce the discriminator vectors d_1 and d_2 to be used to perform clustering. We call the proposed method **D**ata-centric SIDIWO, D-SIDIWO and we describe it in details in Algorithm 5.

Algorithm 5 D-SIDIWO

Require: A dataset $\{x^{(i)}\}_{i=1}^n$

1: Get the raw moments $\hat{M}_2$ and $\hat{M}_3$ as (6.5) and (6.6).
2: Compute a SVD of $\hat{M}_2$ truncated at the 2nd singular vector: $\hat{M}_2 \approx U_2 S_2 U_2^\top$.
3: Define the whitening matrix $E_2 = U_2 S_2^{1/2} \in \mathbb{R}^{d\times 2}$.
4: Find the matrix $D = [d_1|d_2]^\top \in \mathcal{D}_2$ optimizing (6.7)
5: **for** $i \in [n]$ **do**
6: $\quad Cluster(x^{(i)}) = \text{argmax}(|\langle d_1, x^{(i)}\rangle|, |\langle d_2, x^{(i)}\rangle|)$
7: **end for**
8: **return** The cluster assignment

Equation (6.11) provides a splitting procedure that makes particularly sense for all data types where the angle between two vectors is a good indicator of their difference; this includes points on the simplex, on the sphere or on the hypercube, which is the case for binary data. In this last case for example, it makes sense to consider two points similar if they share several non-zero entries; similar points will thus provide similar values for the inner product with d_i and will finish in the same cluster.

The solution of problem (6.7) can be found easily, following the approach described in Theorem 5.3.3 of Chapter 5. In fact, it is sufficient to observe that $D \in \mathcal{D}_2$ if and only if $D = (E_2 O)^\dagger$, where O is an orthogonal matrix, admitting thus the following parametrization:

$$O = O(a) = \begin{bmatrix} \sqrt{1-a^2} & a \\ -a & \sqrt{1-a^2} \end{bmatrix}, for \;\; a \in [-1, 1].$$

The variable a is the only decision variable in problem (6.7) that can be thus optimized by griding on $[-1, 1]$.

Algorithm 5 is a method to split a dataset into two parts. Recursively iterating this procedure we obtain a top-down hierarchical clustering algorithm that can be used to find hierarchical trees of arbitrary depth. Starting from the full dataset, each iteration of D-SIDIWO generates two new leaves, on which D-SIDIWO is recursively called. A user can decide when to stop the splitting procedure: a simple heuristic may consist in halting the method when a certain maximum depth of the tree is reached.

Comparison with other methods of moments

Despite the core engine of D-SIDIWO is the same of the SIDIWO-based hierarchical method of moments described in Chapter 5, the approach that we are taking here to perform clustering is radically different. In the approach of Chapter 5 at each iteration, we would have used SIDIWO to learn a mixture model with two states - using the traditional approach to method of moments described in Chapter 1 - which is then used to bisect the dataset via a MAP assignment. Like all model-based methods of moments, also this approach requires to work with model-specific estimates of the moments, and tends to suffer of the limitations described in the introduction of this chapter.

D-SIDIWO instead uses a generic, standardized definition to estimate the moments, and directly recovers a set of discriminator vectors with a clear relation with respect to the input data. Then, these vectors are used to group the samples of a dataset into two clusters. In this way no model-specific estimators are required, making the approach easier to implement, and, as we will see in the experiment section, more stable.

Traditional methods of moments typically benefit of the application of few iterations of EM (Dempster et al., 1977) - a step that is often necessary to get reliable results (see for example what done in Chapter 4). This approach can be performed also with D-SIDIWO, recovering, after each binary split, a mixture model with two components, whose centers are the average of the points in each cluster and the mixing weights are the clusters proportions. This mixture can then be refined with EM. We will see in the next section that, while the performance of D-SIDIWO is good enough to not require such further refinement, the subsequent application of EM increases the quality of the retrieved clusters, with a slight increase in the running times. Without ever incurring in clusters with poor quality, a user can threshold between short running times and even-better clusters, using or not EM after D-SIDIWO.

6.4 Experiments

In this section we experimentally evaluate the performance of D-SIDIWO. Instead of using synthetic data, we directly use real-world high-dimensional binary data – in particular, the electronic healthcare records introduced in Chapter 4 – focusing again on the task of patients clustering. We will use these datasets to compare qualitatively and quantitatively the performance of D-SIDIWO with those of existing approaches to clustering.[1]

6.4.1 MIMIC Dataset

Like in Chapter 4, we consider MIMIC III dataset (Johnson et al., 2016), focusing on the diagnostic ICD9 records, performing an analogous pre-processing of the data. We recall that ICD9 is a hierarchical coding of the form ddd.dd, with the .dd part being optional: the first 3 digits represent a general diagnostic while extra digits make the diagnostic more precise. We kept only the first three digits of the ICD-9 codes, which are those with more consensus among doctors, obtaining a vocabulary of 696 codes. Furthermore, we further refine the collected data by keeping the 90% most frequent diseases (corresponding to the 222 top diseases) and discarding the records with less than 3 diagnostics. After this processing, MIMIC dataset will contain 51181 patients. The dataset is then mapped into a matrix with binary entries, using a bag-of-words approach: the entry (i, j) of the output matrix is 1 if the

[1]The experimental setting is identical to that of the previous chapters.

patient i presents the disease j, otherwise a zero. Similarly to what we did in Chapter 4, we aim at grouping together patients with similar clinical profiles by clustering the rows of the retrieved matrix.

Evaluation method

We compare D-SIDIWO on the clustering task against other techniques from both MoMs and clustering literature.

First, we will perform a comparison on the *predictive power* of the retrieved clusters. We will run D-SIDIWO (running it until a given maximum depth, considering the clustering induced on the leaves) to cluster the samples of a dataset into k groups, and do the same with the competing methods. Then, we will create a dataset containing, as only information, the cluster to which each sample is assigned, and perform a train/test split. We will then use a logistic regression to predict if a patient length of stay is going to be higher than a week or not (an information that is available in MIMIC dataset). The accuracy of the prediction will be measured on the test set, as a metric indicating the predictive power of the retrieved clusters. During this experiment, we register the running time of each method during each iteration.

The second comparison will be on the *stability* of the clustering methods. Consider a clustering algorithm, and run it over a dataset $\mathcal{S}_1 = \mathcal{S} \cup \mathcal{S}_a$, obtaining a clustering that we will call $\mathcal{C}_1$. Now, assume to perturb the original dataset, by removing $\mathcal{S}_a$, and including a new set $\mathcal{S}_b$ of different samples, independently generated: $\mathcal{S}_2 = \mathcal{S} \cup \mathcal{S}_b$. Then re-run the clustering algorithm obtaining a second clustering $\mathcal{C}_2$. If a clustering algorithm is stable under such kind of perturbations, the partitions induced by $\mathcal{C}_1$ and $\mathcal{C}_2$ on $\mathcal{S}$ are expected to be similar. We run this test on our dataset, by randomly taking $|\mathcal{S}| = 40944$ and $|\mathcal{S}_a| = |\mathcal{S}_b| = 5118$ (respectively, the 80% and 10% of the total sample size). For each of the tested methods, we generate two clusterings, one on $\mathcal{S}_1$ and one on $\mathcal{S}_2$, and we compare them on $\mathcal{S} = \mathcal{S}_1 \cap \mathcal{S}_2$, expecting them to be similar. We use the adjusted Rand index (Hubert and Arabie, 1985) to compare the two partitions: an index close to 1 will indicate consistent clusterings, and thus stability of the method. All the experiments will be run 5 times and the standard deviation of the results will be reported.

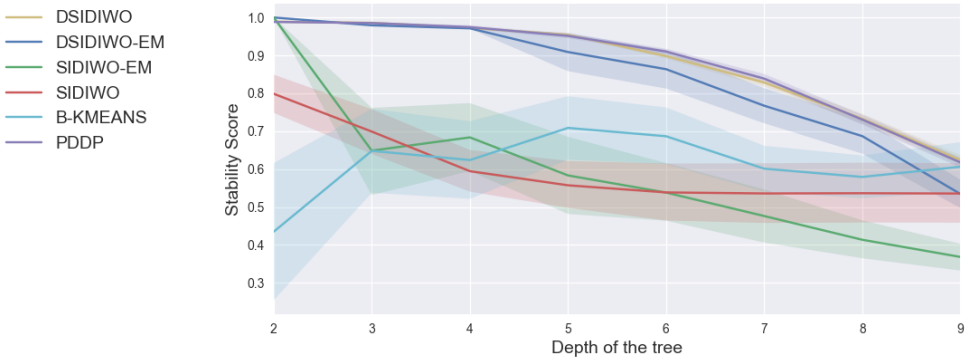
Method	Stability Mean	Stability St.dev.	Predictive Accuracy	Run Time
3V-TPM	0.1	0.06	0.57	3.6 sec.
3V-MoM	0.12	0.02	0.61	**1.6** sec.
D-SIDIWO	**0.95**	**0.01**	**0.62**	18 sec.
3V-TPM+EM	0.58	0.12	0.58	236 sec.
3V-MoM+EM	0.55	0.29	0.62	248 sec.
ASVTD+EM	0.73	0.1	0.60	238 sec.
D-SIDIWO+EM	**0.90**	**0.10**	**0.63**	**88** sec.

Table 6.1: The clustering performance of flat methods of moments compared with those of D-SIDIWO on MIMIC dataset.

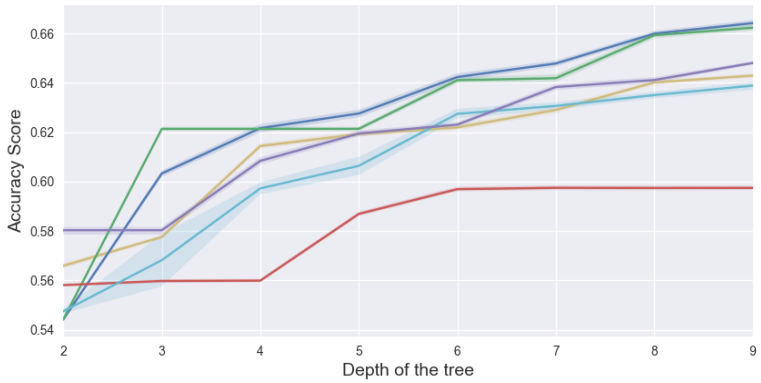
Comparison with other methods

First we compare D-SIDIWO with flat methods of moments, focusing on ASVTD (see Chapter 4), and on the multi-view approaches from Anandkumar et al. (2012b) (named 3V-MoM) and Anandkumar et al. (2014) (3V-TPM). In all the cases we model the data as generated by a mixture of independent Bernoulli variables (see Section 4.2). We select 16 clusters, run a method of moments to retrieve model parameters, and use them to perform clustering, via a MAP assignment. For D-SIDIWO, we run the method until when we reach a maximum depth of 5, corresponding to 16 leaves/clusters. For all the methods, we study the effect of adding some iterations of EM, setting as a stopping criterion the norm of the variation of the matrix M between two iterations to be less than 0.0001. We print in Table 6.1 the results of the experiments. D-SIDIWO outperforms existing methods of moments in terms of predictive power and stability, providing competing results even without EM. Without EM, pure methods of moments look faster than D-SIDIWO; however, their running times explode if aided with EM, while D-SIDIWO plus EM runs in few tens of seconds.

We now perform a comparison with other hierarchical methods. We compare D-SIDIWO with SIDIWO, using the hierarchical method of moments from Chapter 5 (with and without using EM after each binary split) where



(a)



(b)

(c)

Figure 6.1: Analysis of the stability 6.1a and of the predictive power 6.1b for the MIMIC dataset. Shaded areas represent the standard deviation of the results over 5 experiments. Figure 6.1c contains the average running time in seconds of the various methods.

again data is modeled as generated by a mixture of independent Bernoulli variables, and the moments are retrieved via a multi-view approach. Also, we compare the results with those of standard divisive clustering algorithms: bisecting K-means, which iteratively uses K-Means to recursively bisect a dataset (Savaresi and Boley, 2001) and Principal Direction Divisive Partitioning (PDDP) from Boley (1998). We run the various methods for several maximum depths, each of which will correspond to a number of clusters - a tree with maximum depth i will have 2^{i-1} clusters. Stability analysis is presented in figure 6.1a, while accuracy scores, for the analysis of predictive power, are plotted in figure 6.1b. $x-$axis always represents the maximum depth of the tree. Looking at predictive power, the top performers are D-SIDIWO and SIDIWO aided with EM; but while pure D-SIDWIO has performance similar to those of existing divisive clustering methods, the performance of pure SIDIWO look poorer. D-SIDIWO is the top performer in terms of stability, while the version aided with EM provides a similar, but slightly lower, score. Running times can be compared in Figure 6.1c: we can see that D-SIDIWO is among the fastest methods.

Analysis of the clusters

In this section we assess the quality of the clusters obtained by the proposed algorithm on the MIMIC dataset. To do this, we plot in Figure 6.2 the output of the clustering procedure, stopping, for ease of visualization, the algorithm at the maximum depth of 4 (getting in this way 8 clusters). For each leaf, we annotate the most relevant diseases (using the relevance score as in Chapter 4 at Equation 4.12). In this way we can infer the content of the clusters from their most characteristic diseases. The clusters are clearly characterized and grouped in homogeneous branches of the tree. The bottom-left part of the tree contains new-born patients; then, proceeding clockwise, we can find clusters regarding, hearth, kidney, liver, infection and injury.

6.4.2 Tertiarism Dataset

The second dataset is the Tertiarism dataset we used in Chapter 4, provided by the Servei Català de la Salut. It contains ICD9 records for all the hospitalizations in year 2016 of "tertiary" patients, that are those with a serious disease that is supposed to be treated in the larger hospitals in the area.

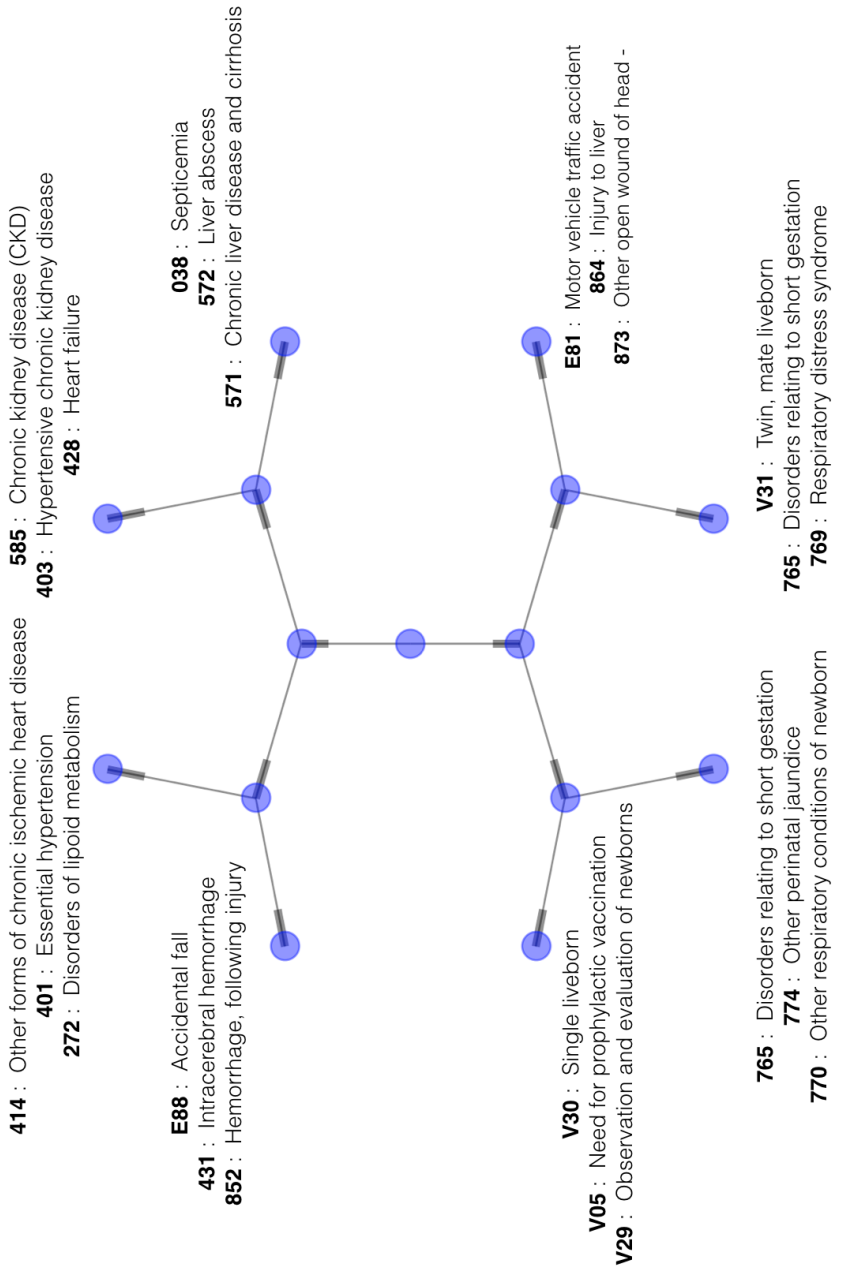


Figure 6.2: The hierarchical tree for the MIMIC dataset.

Method	Stability Mean	Stability St.dev.	Predictive Accuracy	Run Time
3V-TPM	0.06	0.03	0.67	5.35 sec.
3V-MoM	0.06	0.01	0.68	**0.70** sec.
D-SIDIWO	**0.89**	**0.01**	**0.7**	7 sec.
3V-TPM+EM	0.67	0.08	0.68	214sec.
3V-MoM+EM	0.71	0.10	0.69	97 sec.
ASVTD+EM	0.65	0.07	0.69	50 sec.
D-SIDIWO+EM	**0.86**	**0.3**	**0.7**	**30** sec.

Table 6.2: The clustering performance of flat methods of moments compared with those of D-SIDIWO on the Tertiarism dataset.

Comparison with other methods

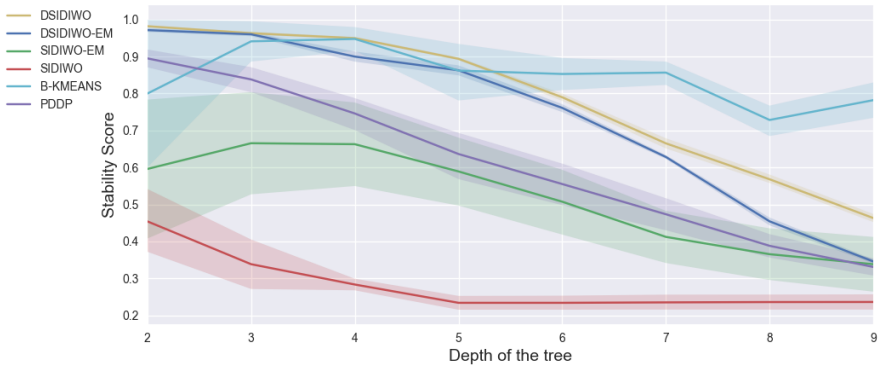
We perform the same evaluation performed with MIMIC, presenting the comparison with flat methods of moments in Table 6.2 and those with hierarchical methods in Figures 6.3a, 6.3b and 6.3c. Again, D-SIDIWO outperforms existing flat spectral methods, both with and without EM. At the same time it results stable under random perturbations of the dataset, while keeping conveniently fast running times.

Analysis of the clusters

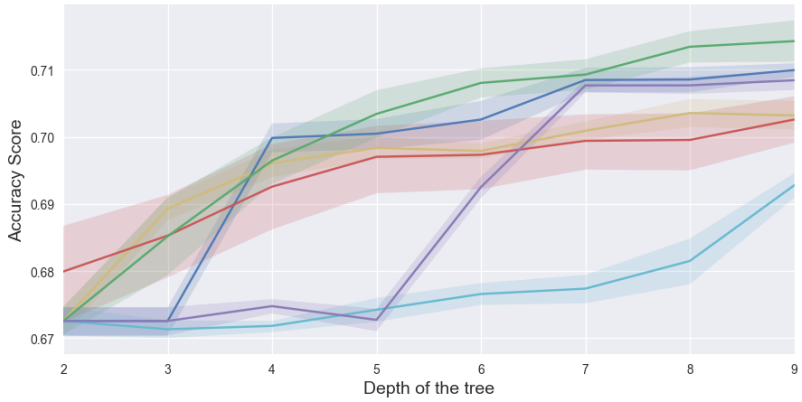
Figure 6.4 presents the content of the clusters provided by D-SIDIWO when we ask for 8 leaves. Again, the clusters make sense and relate with major traumatic events, neoplasms, hearth and kidney-related problems and complicated labors.

6.5 Conclusions

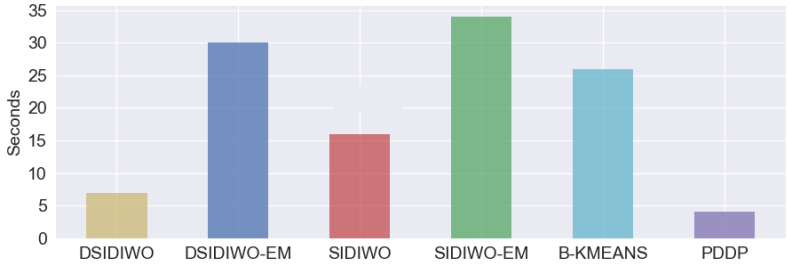
In this chapter, we have presented D-SIDIWO, a novel clustering algorithm based on a data-oriented interpretation of the method of moments.



(a)



(b)



(c)

Figure 6.3: Analysis of the stability 6.3a and of the predictive power 6.3b for the Tertiary dataset. Shaded areas represent the standard deviation of the results over 5 experiments. Figure 6.3c contains the average running time in seconds of the various methods.

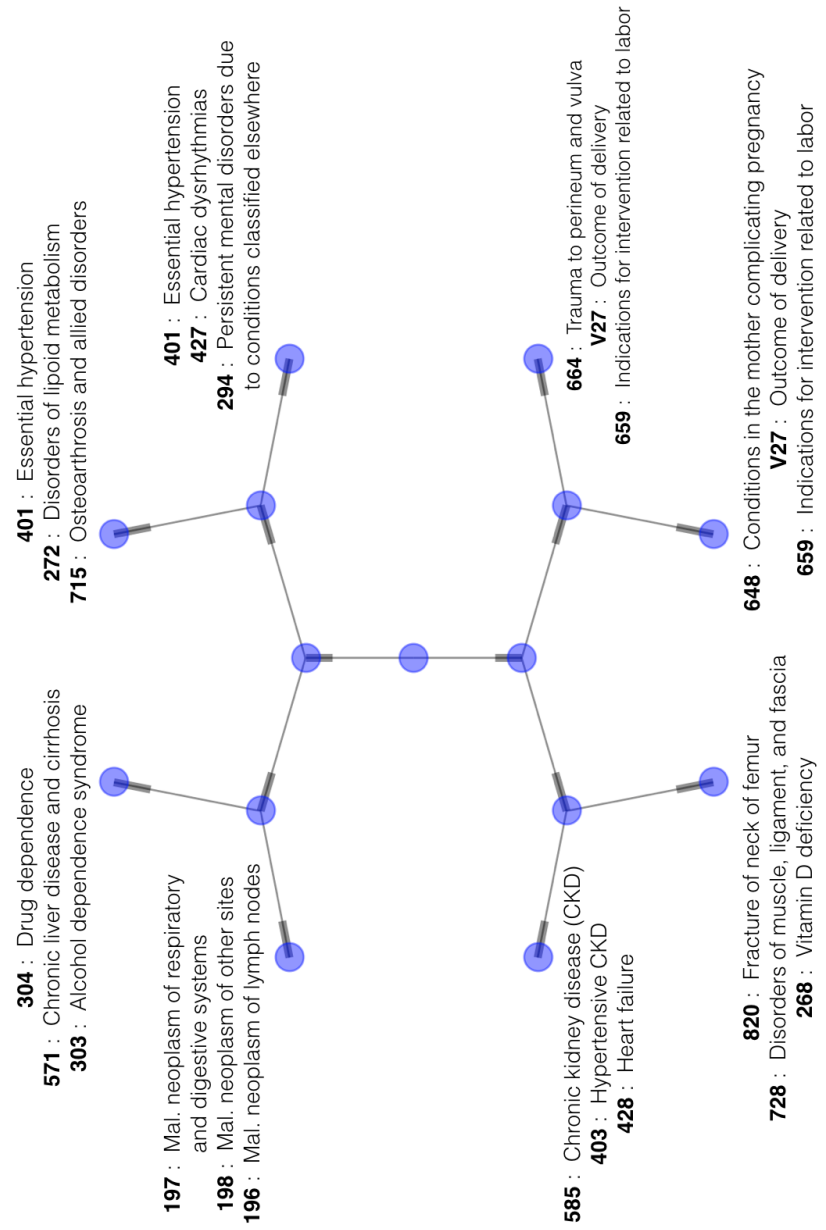


Figure 6.4: The hierarchical tree for the Tertiarism dataset.

Unlike existing methods, D-SIDIWO is grounded on a explicit relation between the input data and the output parameters of a method of moments, providing guaranteed results that are optimal according to an intuitive geometric interpretation which hold regardless the model generating the data. We tested the proposed algorithm on two real-world datasets, showing that D-SIDIWO provides meaningful results on real data even without the aid of EM, outperforming other methods in terms of speed, clusters quality and stability.

Conclusions, Open Problems and Future Work

In this thesis, we have investigated several aspects of methods of moments, analyzing the theory on which these methods are grounded, and presented their most notable limitations and possible improvements. Also, we have provided applications of these tools to two prominent areas of applied machine learning: topic modeling and healthcare analytics. Importantly, in this thesis we have shown that the aspects of interest of methods of moments go beyond their theoretical properties and they can be applied with success to real world problems – an area poorly explored in previous literature. We have provided implementations of these techniques – which require little or no parameter tuning and provide superior computational and learning performance with respect to existing methods – enabling also non-specialized practitioners to use them in their research activities.

Despite the recent progresses, the theory studying methods of moments is still not exhaustive, their understanding is still limited and several problems remain open. In this chapter, we will outline some open problems arising from the theory studied in this thesis. Furthermore, we will present some possible ideas of applications for the models and the approaches presented in this thesis.

Theoretical Open Problems

To improve the perturbation theorem for SVTD

In Chapter 2, we have introduced SVTD, a decomposition algorithm for methods of moments. Together with SVTD, we have also presented a

perturbation theorem (Theorem 2.2.1) that analyzes how the noise on input moments propagates to the output model returned by SVTD. This theorem ensures that, under certain hypotheses, SVTD is guaranteed to return a model that gets closer to the one generating the data as the sample size increases. Intuitively, this theorem explains why SVTD works in practice – for example in the experimental Section 2.3. The bounds that this theorem provides are very similar to the ones presented by Anandkumar et al. (2012b) for the SVD method; however, the experiments performed in Section 2.3 prove that SVTD has a better learning accuracy than the SVD method. This fact suggests that the bounds provided by Theorem 2.2.1 can be widely improved. One possibility to provide such improvement may consist in analyzing the perturbations under a different and more structured point of view, separating the contribution to the final error coming from the problem from those coming from the algorithm. Those coming from the problem can be studied using the condition number, for which we could rely on the recently developed theory on Waring decompositions (Breiding and Vannieuwenhoven, 2018), which is an extension of the decompositions studied here. Understanding the exact contribution to the error that is given by the problem may enable us to understand which part of the noise we are seeing is introduced by the algorithm, and to perform a better analysis leading to bounds tighter than those in Theorem 2.2.1.

To make SVTD more general, reducing its requirements

Another improvement point related to SVTD is the fact that in Theorem 2.1.1, we have assumed that at least one feature of the model we want to learn has different conditional expectations across the various values of the latent state. When this requirement is violated SVTD does not fail, but the returned parameters have no guarantees to properly solve the equations of the moments – even in the setting where the theoretical, unperturbed moments are available[2]. Even if this requirement is mild, it is interesting to study how to modify SVTD in order to remove this limitation, so to see if improved experimental performance is obtained. For example, the SVD method (Anandkumar et al., 2012b) overcomes this issue by performing a random transformation of the model to be learned; this approach could be ported to our case by finding the matrix O (line 4 of Algorithm 1) from

[2]Simple experiments running SVTD where this condition is violated show that it may actually fail to provide the right solution of the moment equations.

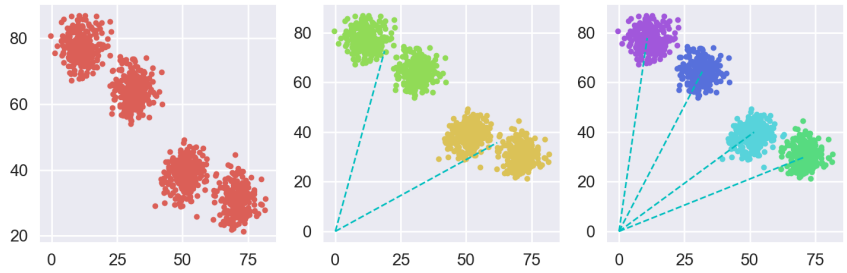


Figure 6.5: Leftmost figure: an overcomplete naive Bayes model with Gaussian features, where we have $d = 2$ and $k = 4$. Center figure: the two pseudocenters learned by SIDIWO allow to bisect the data into two clusters. Rightmost figure: iterating, we are able to recover four pseudocenters, coinciding with the four centers of the model.

the singular vectors of a random linear combination of the various matrices H_r, which would ensure the almost certain uniqueness of the results. It may be interesting to see if a similar approach – which has the cost of adding artificial random noise to the algorithm – would improve the performance of SVTD. Ideally, it may guarantee similar or improved performance, with less theoretical limitations.

To use SIDIWO to learn hierarchical models

In Chapter 5, we have introduced a technique to retrieve hierarchical representations of latent variable models. That method enables to retrieve a hierarchical tree whose nodes describe portions of the analyzed data with increasing accuracy. One interesting extension of this work may consist in analyzing whether there exist hierarchical latent variable models that can be accurately learned with the proposed hierarchical method of moments – like for example binary latent trees.

To use SIDIWO to learn overcomplete models

Another interesting open question is whether the hierarchical approach introduced by SIDIWO may enable to learn overcomplete latent variable models – models where the number of latent states is higher than the number of observable features, so with $d < k$. A simple example may come from

a naive Bayes model (see Chapter 1) with Gaussian features – like the one depicted in the leftmost chart of Figure 6.5 – with bidimensional features, $d = 2$, and $k = 4$ latent states/clusters. This simple model would be not learnable with a traditional approach to method of moments – which would require $d \geq k$. Conversely, with the hierarchical approach proposed in Chapter 5, we are able to first learn a model with two pseudo-centers, which allows to bisect the data as done in the center chart of Figure 6.5, and then to iterate the approach on the two clusters, to obtain four pseudo-centers, which are very close to the actual centers of the model. Our intuition is that the pseudo-centers obtained in this way asymptotically approximate those of the model generating the data, allowing SIDIWO to learn model otherwise not learnable with standard methods of moments. A possible next step may consist in studying if and under which conditions this intuition is confirmed; if confirmed, this would make SIDIWO a robust learning technique for certain overcomplete latent variable models.

To discover explicit relations between the data and the outputs of a method of moments

An important open problem is the one of finding explicit links between inputs and outputs of methods of moments; a problem that, if solved, could dramatically increase our understanding of these techniques. In Chapter 6 we did a first step in this direction, geometrically linking the input data with the optima of SIDIWO and using the optima to perform clustering. However, this required to modify the purpose of methods of moments, renouncing to explicitly learn a latent variable model and working with a standardized, not model-dependent definition of the moments. An interesting next step consists in continuing the analysis carried on in Chapter 6 under a model-dependent perspective, with the objective of discovering an explicit relation between the input data and the models that are returned by methods of moments.

To connect methods of moments with maximum likelihood

A interesting path that is worth exploring consists in finding theoretical relations linking methods of moments with maximum-likelihood based approaches. For example, consider a dataset $\mathcal{X}$, and assume it to be generated by a certain latent variable model, parameterizable with a pair (M, ω) and let $(\tilde{M}, \tilde{\omega})$ be the pair of parameters returned by a method of moments run on

the observed data. Now, assume to have an oracle, able to return the model maximizing the likelihood of the observed data: $(\tilde{M}_{ML}, \tilde{\omega}_{ML})$. This model will probably be different from the one returned by the method of moments, as they are optimizing two different cost functions: the method of moments is trying to match the empirical moments, while the likelihood-based one is trying to maximize the likelihood of the observed data. Studying a relation between the maximum-likelihood model and the one learned with a method of moments is a poorly explored research area. Intuitively, the two models are expected to converge to the same model (M, ω) as the sample size increases, but it would be interesting to understand if the two optimization problems are in some way related to each other. For example, can we say that, in certain scenarios, the optimization problem solved by a method of moments is a relaxation of the maximum-likelihood problem? The fact that methods of moments typically return good initializers for likelihood-maximizing heuristics (Balle et al., 2014; Zhang et al., 2014) seems to foster the intuition that some kind of relation may exists, however an explicit intuition on the relation between the two optimization problems is missing and remains an interesting open problem to be explored in future works.

Model selection and connections with the *minimum description length* principle

In the first part of this thesis, we have presented a set of methods to learn latent variable models assuming that the number of latent states k was known to the algorithm, overcoming this requirement in part II, with a series of hierarchical approaches. Additionally, in Chapters 5 and 6, we proposed a set of algorithms that are guaranteed to return a meaningful model for any value of the input k; nevertheless, finding which is the *best* value of k to enable a latent variable model to properly describe a dataset, remains an interesting open problem. Consider for example the patient clustering problem studied in Chapter 4; in that case, the value k represents the number of clusters in which we want to group our population of patients. While the results in Chapters 5 and 6 tell us that with the proposed algorithms we will likely obtain a meaningful clustering for any value of k, it is interesting to understand whether it exists a value k that is better than the others, and to find a principled way to determine it.

An interesting approach may come from the so-called Minimum Description Length (MDL) principle, which is an information-theoretical formalization of

Occam's razor (Grunwald, 2004). Loosely speaking, the MDL principle relies on the fact that regularities occurring in the data can be used to retrieve a compressed encoding of the data itself, and on the idea that between two encodings equally able to capture these regularities, the one providing the highest compression should be preferable. Phrasing this under a probabilistic perspective, a latent variable model could be interpreted as a synthetic encoding of the training set, an entity able to capture most of the regularities appearing in the data, while keeping a conveniently compressed representation. Increasing the number of latent states, would increase the ability of the model to capture these regularities, but would also increase the size of the model we use to describe the data, yielding to a less compressed representation. As a consequence, the MDL principle translates to the probabilistic framework as a model-selection tool, where the model is selected by thresholding between how well a model describes the data and how big the model is; too big models will be probably accurate, but will be penalized by their high dimension, while very small models will be not enough accurate to be selected.

The MDL is commonly used in the literature to select the number of states in a latent variable model (Roberts, 1999), and the predominant formalization (Grunwald, 2004) consists of building a cost function summing the likelihood of a model with respect to the training data to some non-decreasing function of the number of parameters of the model – with the first term indicating the goodness of fit, and the second acting as a regularizer, preferring in this way smaller models. This approach is well studied, it can be successfully used in practice to select the value of k (Roberts et al., 1998) and generalizes other well-known model-selection tools like the Bayesian Information Criterion (BIC) (Schwarz et al., 1978) – but it is based on the likelihood function and, implicitly, on the maximum likelihood principle, whose relation with the outputs of a method of moments are not always clear.

An alternative approach may come from the observation that the method of moments is essentially solving a low-rank tensor factorization problem, and the model selection task consists in finding the proper rank to be used to perform the decomposition. This problem can easily be framed under the MDL perspective: a rank-k matrix decomposing a tensor would be seen as a synthetic representation of the tensor itself, and the optimal rank could be found by tresholding between the size of the matrix – obviously depending on k – and the decomposition error. Under the model perspective this would mean to select k using a cost function composing the error in the tensor decomposition task – which would indicate how well the model fits the data – with an

additional cost penalizing models where k is too big. This approach seems more aligned with the spirit of methods of moments than the likelihood-based formalization; additionally, literature provides several MDL-based approaches to find the optimal rank for the tensor decomposition task (Zarowski, 1998; Ramírez and Sapiro, 2012; Liu et al., 2016; Squires et al., 2017), which suggest that this approach is feasible. Extending these works under the probabilistic model-selection perspective may lead to the development of a principled approach to model-selection in methods of moments, which is an interesting area for future research.

Methods of moments to learn interpretable generative models, and relations with deep-learning

During all this thesis, Latent Variable Models have been seen as probabilistic models able to approximately describe the training data. However, in order to keep tractable the equations arising during the learning task, limitations have been posed over the complexity of a model: in Chapter 4 for example, we performed the assumption of conditionally independent diseases, given the latent medical status of a patient; in Chapter 3, while modeling texts, we assumed exchangeable words, claiming that each word has no direct relation with the words in its context. These limitations allowed us to calculate straightforward equations of the moments, enabling learning techniques with provable guarantees of global optimality; at the same time, the models we retrieved were able to describe the training data in a satisfying way and were simple – ensuring interpretability of their behavior when used in practical tasks like clustering (see Chapter 4). On the other hand, it is natural to wonder whether more complex and sophisticated models would have better captured the patterns and the regularities arising in the data – perhaps at the cost of sacrificing the interpretability of the model or the efficiency and optimality of the learning task.

Understanding how much a model hypothesis is limiting to accurately describe a dataset is an interesting area for future research, which nicely intersects with another branch of machine learning: the theory of generative processes. A latent variable model in fact can be seen as a generative process able to generate samples according to a prescribed probability distribution. Ideally, if a model accurately describes the data and captures all its characteristics, it is expected to generate synthetic samples indistinguishable from the training ones; the similarity between the training data and the ones

sampled from the model distribution may thus be an indicator of how well a model describes the data. A potential future line of research consists of comparing synthetic samples generated from models learned with methods of moments with the training data, checking how well the synthetic data resembles the training one. This opens the problem on how to determine whether or not two datasets are indistinguishable, and what *indistinguishable* means; to this extent one could leverage on some statistical hypothesis tests, like the one introduced by Gretton et al. (2012) – which uses a metric called Maximum Mean Discrepancy (MMD) – as an indicator of how much the distributions of two samples differ. Alternatively, one could train a classifier (called *discriminator*) to distinguish whether a sample is real or synthetic and consider its generalization accuracy as an indicator of the distinguishability of the two datasets.

In a recent work (Aviñó et al., 2018), we used these two approaches to evaluate the model describing patient medical records introduced in Chapter 4: after having learned the model, we used it to generate synthetic ICD9 medical records, and compared the generated data with the real one – using both the MMD and a discriminator – showing that for a high number of clusters, synthetic and real data are hard to distinguish under both the criteria. On one hand this was a positive result, showing that that methods of moments can learn highly descriptive model. However, ICD9 records are highly structured data, and it is not a surprise that a simple model like a naive Bayes model is able to accurately describe their behavior. Extending this analysis to more challenging scenarios is a interesting next step, which would help us in setting the boundaries on which modeling tasks can be accomplished using an understandable model and which tasks instead can not. Consider for example the task of modeling texts: while we can expect that models like the Single Topic Model and the LDA will not be able to generate credible synthetic texts, it would be interesting to understand if a more complex – but still understandable and interpretable – model would be capable of this task.

Beside the theoretical implications described above, it is important to observe that generating synthetic but plausible data is a task with several practical applications: synthetic data can be used for example to establish cross-industry benchmarks, to stress-test a system by trying it on larger datasets than available, or as the shareable version of data otherwise protected by privacy constraints. Finding latent variable models able to generate realistic synthetic data in various contexts would have the benefit of providing an

interpretable description of the data – via the structure of the LVM – and an effective generator for synthetic data. It is interesting to observe that not-interpretable models able to generate plausible data in challenging contexts exist, and have been object of an increasing interest in the deep learning community during the last few years. These models, called *implicit generative models* (Mohamed and Lakshminarayanan, 2016), are probabilistic parametric models whose likelihood function is not explicitly accessible but from which it is possible to generate synthetic samples. Unlike the LVMs we have seen up to now, they are not interpretable, and they are typically represented via a deep neural network with millions of parameters, which propagates the outcomes of a prior to generate synthetic, structured data. These networks are trained with the explicit objective of generating data indistinguishable from the training one: they use objective functions aimed for example at minimizing the MMD (Li et al., 2015) or at matching the moments between synthetic and training data (Ravuri et al., 2018) or at making a discriminator unable to distinguish between synthetic and real data (Goodfellow et al., 2014). Implicit generative models have been successfully used to generate synthetic data from a variety of scenarios, including images (Goodfellow et al., 2014), texts (Yu et al., 2017) and medical data (Esteban et al., 2017; Choi et al., 2017; Yoon et al., 2018) and, while they seem to be able to simulate arbitrarily complex models, they do not provide any interpretation on the process generating the data; additionally, these models typically require longer training times in comparison with the simpler latent variable models studied in this thesis (see Aviñó et al., 2018). Understanding in which contexts and in which modeling tasks implicit generative models could be substituted by more interpretable LVMs is an interesting area of future research.

Ideas for Applications

Clustering patients using multiple sources of data

In Chapter 4, we have presented a technique to cluster patients in groups with homogeneous clinical profiles, starting from ICD9 data. In our modeling, each patient is characterized by a latent state, which represents his unobservable clinical status; observable features, like the ICD9 diagnostics, are the real-world manifestation of this latent state. This model can be easily extended to include other observable features, without changing its intrinsic

structure. For example, modern healthcare systems collect for each patient data coming from several sources, other than ICD9 record, like lab results, medical notes, X-ray images, ethnic and genetic information about the patient. An interesting model, would consider these sources of data as different *views* on the patient latent state. Each view will follow its own distribution, which will depend on the latent patient status; for a certain status, the views may be considered to be conditionally independent. For example, imagine to add doctors notes to the model described in Chapter 4. Then, we would have two views: one representing the ICD9 records, and another one the doctor annotations on patient status. The first view will be distributed as a mixture of independent Bernoulli variable, while the second, for example as a single topic model; both depending on the same latent variable, representing the patient's medical status. This kind of modeling may be powerful to provide improved and more granular clusters, which consider several, otherwise hardly joinable datasets. Two interesting challenges may arise in this setting: first, not all the patients may have data regarding all the views – some patients may have X-ray images but not lab tests, some other blood tests but no medical annotations for examples – and second, views may present conflicting clusters – a clustering done on the sole ICD9 records for example may group patients by clinical profiles, while the one done only on medical notes may group patients by the prescriptions recommended to the patients by the doctors (see for example He et al. 2017). Providing a method able to deal with missing and conflicting views and yet return meaningful clusters is an interesting future line of applications.

Sequential models to describe medical time series

The clustering approaches described above and in Chapter 4 assume to have data representing a snapshot of patient status at a certain point of his life. During his life, however, a patient may move across several different statuses, visiting a hospital several times and producing at each visit different data records. An interesting next step may thus consist in studying the evolution of patient status along time, while several records – possibly from different views – are provided at different points in time. This may require to introduce more complex latent variable models (see for example Alaa and Van Der Schaar 2018; Li et al. 2018), studying the relations between the data observed at a certain time with those of the subsequent observation, or studying how the waiting times between an observation and the next are

distributed.

Methods of moments and privacy-preserving machine learning

A setting where methods of moments may give an important contribution is privacy-preserving machine learning, and its applications to healthcare analytics. Imagine for example that several hospitals (called *parties* in privacy jargon) have their own private data from certain patients. The parties do not want to exchange the data between them, but do want to use all the available data to develop a certain latent variable model (for example, to perform clustering like in Chapter 4), which will be more reliable as developed on a bigger sample and less biased on a specific hospital population. This scenario may comprehend two different settings.

1. One – called vertically partitioned (VP) setting – where the parties share different kinds of data for the same set of patients. For example one hospital may have lab-results data, while another ICD9 records for the same set of patients. This kind of data, can be modeled using a multi-view approach as described above, with the additional constraint that the views can not be shared between the parties.

2. The second setting – called the horizontally partitioned (HP) setting– is when the hospitals share the same kind of data – like ICD9 records – but for different populations of patients.

In both the settings, the objective is to use all the data available to the various hospitals to learn a certain latent variable model, without actually exchanging these data between the parties, never compromising the privacy of the patients; the learned model will then be shared between the hospitals, so to be used for the tasks it is needed for. An interesting future application of the work presented in this thesis is to develop methods of moments able to work robustly in the two settings described above. This may be almost automatic for the HP setting: here, each party should compute their own moments, add noise to guarantee differential privacy of the moments, and then pass the moments to a party that has to aggregate and decompose the moments to retrieve the model parameters, which are then shared back with the hospitals. Also dealing with the VP setting looks doable, even if more difficult: methods of moments for multi-view models exist (Anandkumar et al., 2012b) but require to perform heavy inner products between the views,

which is not straightforward in a privacy-preserving setting. However, privacy-preserving inner products exist (Gascón et al., 2017), and can help to deal also with this more challenging scenarios.

Methods of moments and genetic data

A last interesting area where latent variable models and methods of moments can have a positive impact is the handling and the modeling of genetic data. These kinds of data are in fact typically high dimensional, sparse and with a number of available features that is in general way larger than the sample size (Cavalli-Sforza and Bodmer, 1999), which is the ideal setting for methods of moments. Examples of application may be clustering – taking genotypes as inputs to discover partitions in human populations (Rosenberg et al., 2002) – or analysis of genotype-phenotype relations – where model-based causal relations between genetic structure and phenotype are investigated (Rakitsch et al., 2013). In both the cases, methods of moments can help in learning interpretable yet high dimensional models with high efficiency, following for example an approach similar to the one described in Chapter 4.

Bibliography

Aggarwal, C. C., Hinneburg, A., and Keim, D. A. (2001). On the surprising behavior of distance metrics in high dimensional space. In *International Conference on Database Theory*, pages 420–434.

Alaa, A. M. and Van Der Schaar, M. (2018). A hidden absorbing semi-Markov model for informatively censored temporal data: learning and inference. *The Journal of Machine Learning Research*, 19(1):108–169.

Alighieri, D. (1979). *La Divina Commedia, a cura di N. Sapegno*. Nuova Italia, Firenze.

Anandkumar, A., Foster, D. P., Hsu, D. J., Kakade, S. M., and Liu, Y.-K. (2012a). A spectral algorithm for latent Dirichlet allocation. In *Neural Information Processing Systems*, pages 917–925.

Anandkumar, A., Ge, R., Hsu, D., Kakade, S. M., and Telgarsky, M. (2014). Tensor decompositions for learning latent variable models. *The Journal of Machine Learning Research*, 15(1):2773–2832.

Anandkumar, A., Hsu, D., and Kakade, S. M. (2012b). A method of moments for mixture models and hidden Markov models. In *Conference on Learning Theory*, pages 33.1–33.34.

Appellof, C. J. and Davidson, E. R. (1981). Strategies for analyzing data from video fluorometric monitoring of liquid chromatographic effluents. *Analytical Chemistry*, 53(13):2053–2056.

Arabshahi, F. and Anandkumar, A. (2017). Spectral Methods for Correlated Topic Models. In *International Conference on Artificial Intelligence and Statistics*, pages 1439–1447.

Arora, S., Ge, R., and Moitra, A. (2012). Learning topic models–going beyond SVD. In *Annual IEEE Symposium on Foundations of Computer Science*, pages 1–10.

Aviñó, L., Ruffini, M., and Gavaldà, R. (2018). Generating Synthetic but Plausible Healthcare Record Datasets. *arXiv preprint arXiv:1807.01514*.

Bailly, R. (2011). Quadratic weighted automata: Spectral algorithm and likelihood maximization. *The Journal of Machine Learning Research*, 20:147–162.

Balle, B., Hamilton, W., and Pineau, J. (2014). Methods of moments for learning stochastic languages: unified presentation and empirical comparison. In *International Conference on Machine Learning*, pages 1386–1394.

Balle, B. and Mohri, M. (2012). Spectral learning of general weighted automata via constrained matrix completion. In *Neural Information Processing Systems*, pages 2159–2167.

Balle, B., Quattoni, A., and Carreras, X. (2012). Local loss optimization in operator models: a new insight into spectral learning. In *International Conference on Machine Learning*, pages 1819–1826.

Bishop, C. M. (2006). *Pattern Recognition and Machine Learning (Information Science and Statistics)*. Springer-Verlag New York, Inc.

Blei, D. M. and Lafferty, J. D. (2005). Correlated topic models. In *Neural Information Processing Systems*, pages 147–154.

Blei, D. M. and Lafferty, J. D. (2006). Dynamic topic models. In *International Conference on Machine Learning*, pages 113–120.

Blei, D. M., Ng, A. Y., and Jordan, M. I. (2003). Latent Dirichlet Allocation. *The Journal of machine Learning research*, 3(Jan):993–1022.

Boley, D. (1998). Principal direction divisive partitioning. *Data Mining and Knowledge discovery*, 2(4):325–344.

Breiding, P. and Vannieuwenhoven, N. (2018). The condition number of join decompositions. *SIAM Journal on Matrix Analysis and Applications*, 39(1):287–309.

Bunse-Gerstner, A., Byers, R., and Mehrmann, V. (1993). Numerical methods for simultaneous diagonalization. *SIAM journal on matrix analysis and applications*, 14(4):927–949.

Cardoso, J.-F. and Souloumiac, A. (1996). Jacobi angles for simultaneous diagonalization. *SIAM journal on matrix analysis and applications*, 17(1):161–164.

Carroll, J. D. and Chang, J.-J. (1970). Analysis of individual differences in multidimensional scaling via an N-way generalization of "Eckart-Young" decomposition. *Psychometrika*, 35(3):283–319.

Cavalli-Sforza, L. L. and Bodmer, W. F. (1999). *The genetics of human populations*. Courier Corporation.

Chaganty, A. T. and Liang, P. (2013). Spectral experts for estimating mixtures of linear regressions. In *International Conference on Machine Learning*, pages 1040–1048.

Chang, J. T. (1996). Full reconstruction of Markov models on evolutionary trees: identifiability and consistency. *Mathematical Biosciences*, 137(1):51–73.

Chaudhuri, K., Kakade, S. M., Livescu, K., and Sridharan, K. (2009). Multi-view clustering via canonical correlation analysis. In *International Conference on Machine Learning*, pages 129–136.

Chen, X. S., Li, W., and Xu, W. W. (2012). Perturbation analysis of the eigenvector matrix and singular vector matrices. *Taiwanese Journal of Mathematics*, 16(1):pp–179.

Chiantini, L., Ottaviani, G., and Vannieuwenhoven, N. (2017). On generic identifiability of symmetric tensors of subgeneric rank. *Transactions of the American Mathematical Society*, 369(6):4021–4042.

Choi, E., Biswal, S., Malin, B., Duke, J., Stewart, W. F., and Sun, J. (2017). Generating Multi-label Discrete Patient Records using Generative Adversarial Networks. In *Machine Learning for Healthcare Conference*, pages 286–305.

Colombo, N. and Vlassis, N. (2016). Tensor decomposition via joint matrix Schur decomposition. In *International Conference on Machine Learning*, pages 2820–2828.

Comon, P., Qi, Y., and Usevich, K. (2017). Identifiability of an X-rank decomposition of polynomial maps. *SIAM Journal on Applied Algebra and Geometry*, 1(1):388–414.

De Lathauwer, L., De Moor, B., and Vandewalle, J. (2004). Computation of the canonical decomposition by means of a simultaneous generalized Schur decomposition. *SIAM journal on Matrix Analysis and Applications*, 26(2):295–327.

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

Esteban, C., Hyland, S. L., and Rätsch, G. (2017). Real-valued (Medical) Time Series Generation with Recurrent Conditional GANs. *arXiv preprint arXiv:1706.02633*.

Feldman, J., Servedio, R. A., and O'Donnell, R. (2006). PAC Learning Axis-Aligned Mixtures of Gaussians with No Separation Assumption. *Learning Theory*, page 20.

Gascón, A., Schoppmann, P., Balle, B., Raykova, M., Doerner, J., Zahur, S., and Evans, D. (2017). Privacy-preserving distributed linear regression on high-dimensional data. *Proceedings on Privacy Enhancing Technologies*, 2017(4):345–364.

Ge, R., Huang, Q., and Kakade, S. M. (2015). Learning mixtures of Gaussians in high dimensions. In *Symposium on Theory of Computing*, pages 761–770.

Geraci, J. M., Ashton, C. M., Kuykendall, D. H., Johnson, M. L., and Wu, L. (1997). International Classification of Diseases, 9th Revision, Clinical Modification codes in discharge abstracts are poor measures of complication occurrence in medical inpatients. *Medical care*, 35(6):589–602.

Golub, G. H. and Reinsch, C. (1970). Singular value decomposition and least squares solutions. *Numerische mathematik*, 14(5):403–420.

Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., and Bengio, Y. (2014). Generative adversarial nets. In *Neural Information Processing Systems*, pages 2672–2680.

Gretton, A., Borgwardt, K. M., Rasch, M. J., Schölkopf, B., and Smola, A. (2012). A kernel two-sample test. *The Journal of Machine Learning Research*, 13(Mar):723–773.

Griffiths, T. L. and Steyvers, M. (2004). Finding scientific topics. *Proceedings of the National academy of Sciences*, 101(suppl 1):5228–5235.

Grunwald, P. (2004). A tutorial introduction to the minimum description length principle. *arXiv preprint math/0406077*.

Halko, N., Martinsson, P.-G., and Tropp, J. A. (2011). Finding structure with randomness: Probabilistic algorithms for constructing approximate matrix decompositions. *SIAM review*, 53(2):217–288.

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

He, X., Li, L., Roqueiro, D., and Borgwardt, K. M. (2017). Multi-view spectral clustering on conflicting views. In *European Conference on Machine Learning and Knowledge Discovery in Databases*, pages 826–842.

Hitchcock, F. L. (1927). The expression of a tensor or a polyadic as a sum of products. *Studies in Applied Mathematics*, 6(1-4):164–189.

Hitchcock, F. L. (1928). Multiple Invariants and Generalized Rank of a P-Way Matrix or Tensor. *Studies in Applied Mathematics*, 7(1-4):39–79.

Ho, J. C., Ghosh, J., Steinhubl, S. R., Stewart, W. F., Denny, J. C., Malin, B. A., and Sun, J. (2014a). Limestone: High-throughput candidate phenotype generation via tensor factorization. *Journal of biomedical informatics*, 52:199–211.

Ho, J. C., Ghosh, J., and Sun, J. (2014b). Marble: high-throughput phenotyping from electronic health records via sparse nonnegative tensor factorization. In *SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 115–124.

Hofmann, T. (2017). Probabilistic latent semantic indexing. *SIGIR Forum*, 51(2):211–218.

Hsu, D. and Kakade, S. M. (2013). Learning mixtures of spherical Gaussians: moment methods and spectral decompositions. In *Innovations in Theoretical Computer Science*, pages 11–20.

Hsu, D., Kakade, S. M., and Zhang, T. (2012). A spectral algorithm for learning hidden Markov models. *Journal of Computer and System Sciences*, 78(5):1460–1480.

Hubert, L. and Arabie, P. (1985). Comparing partitions. *Journal of classification*, 2(1):193–218.

Jain, P. and Oh, S. (2014). Learning Mixtures of Discrete Product Distributions using Spectral Decompositions. In *Conference on Learning Theory*, pages 824–856.

Johnson, A. E., Pollard, T. J., Shen, L., Li-wei, H. L., Feng, M., Ghassemi, M., Moody, B., Szolovits, P., Celi, L. A., and Mark, R. G. (2016). MIMIC-III, a freely accessible critical care database. *Scientific data*, 3:160035.

Kolda, T. G. (2001). Orthogonal tensor decompositions. *SIAM Journal on Matrix Analysis and Applications*, 23(1):243–255.

Kolda, T. G. and Bader, B. W. (2009). Tensor decompositions and applications. *SIAM review*, 51(3):455–500.

Kriegel, H.-P., Kröger, P., and Zimek, A. (2009). Clustering high-dimensional data: A survey on subspace clustering, pattern-based clustering, and correlation clustering. *ACM Transactions on Knowledge Discovery from Data*, 3(1):1.

Kruskal, J. B. (1977). Three-way arrays: rank and uniqueness of trilinear decompositions, with application to arithmetic complexity and statistics. *Linear algebra and its applications*, 18(2):95–138.

Kshetri, K. B. (2011). *Modelling patient states in intensive care patients*. PhD thesis, Massachusetts Institute of Technology.

Kuleshov, V., Chaganty, A., and Liang, P. (2015). Tensor factorization via matrix factorization. In *Artificial Intelligence and Statistics*, pages 507–516.

Kulesza, A., Jiang, N., and Singh, S. (2015). Low-rank spectral learning with weighted loss functions. In *Artificial Intelligence and Statistics*, pages 517–525.

Kulesza, A., Rao, N. R., and Singh, S. (2014). Low-rank spectral learning. In *Artificial Intelligence and Statistics*, pages 522–530.

Leurgans, S., Ross, R., and Abel, R. (1993). A decomposition for three-way arrays. *SIAM Journal on Matrix Analysis and Applications*, 14(4):1064–1083.

Li, T., Rabusseau, G., and Precup, D. (2018). Nonlinear Weighted Finite Automata. In *Artificial Intelligence and Statistics*, pages 679–688.

Li, Y., Swersky, K., and Zemel, R. (2015). Generative moment matching networks. In *International Conference on Machine Learning*, pages 1718–1727.

Liu, K., Da Costa, J. P. C., So, H. C., Huang, L., and Ye, J. (2016). Detection of number of components in CANDECOMP/PARAFAC models via minimum description length. *Digital Signal Processing*, 51:110–123.

Macqueen, J. (1967). Some methods for classification and analysis of multivariate observations. In *Berkeley Symposium on Mathematical Statistics and Probability*, pages 281–297.

Marin, J.-M., Mengersen, K., and Robert, C. P. (2005). Bayesian modelling and inference on mixtures of distributions. *Handbook of statistics*, 25:459–507.

McDiarmid, C. (1998). Concentration. In *Probabilistic methods for algorithmic discrete mathematics*, pages 195–248. Springer.

Melnykov, V., Maitra, R., et al. (2010). Finite mixture models and model-based clustering. *Statistics Surveys*, 4:80–116.

Mimno, D., Wallach, H. M., Talley, E., Leenders, M., and McCallum, A. (2011). Optimizing semantic coherence in topic models. In *Conference on empirical methods in natural language processing*, pages 262–272.

Mohamed, S. and Lakshminarayanan, B. (2016). Learning in implicit generative models. *arXiv preprint arXiv:1610.03483*.

Mossel, E. and Roch, S. (2005). Learning nonsingular phylogenies and hidden Markov models. In *Symposium on Theory of Computing*, pages 366–375.

Newman, D., Asuncion, A., Smyth, P., and Welling, M. (2009). Distributed algorithms for topic models. *The Journal of Machine Learning Research*, 10(Aug):1801–1828.

Ng, A. Y., Jordan, M. I., and Weiss, Y. (2002). On spectral clustering: Analysis and an algorithm. In *Neural Information Processing Systems*, pages 849–856.

Pearson, K. (1894). Contributions to the mathematical theory of evolution. *Philosophical Transactions of the Royal Society of London. A*, 185:71–110.

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., et al. (2011). Scikit-learn: Machine learning in python. *The Journal of machine learning research*, 12(Oct):2825–2830.

Pérez, L. M., Inzitari, M., Quinn, T. J., Montaner, J., Gavaldà, R., Duarte, E., Coll-Planas, L., Cerdà, M., Santaeugenia, S., Closa, C., et al. (2016). Rehabilitation profiles of older adult stroke survivors admitted to intermediate care units: A multi-centre study. *PloS one*, 11(11).

Perrone, V., Jenkins, P. A., Spanò, D., and Teh, Y. W. (2017). Poisson random fields for dynamic feature models. *The Journal of Machine Learning Research*, 18(1):4626–4670.

Perros, I., Papalexakis, E. E., Wang, F., Vuduc, R., Searles, E., Thompson, M., and Sun, J. (2017). SPARTan: Scalable PARAFAC2 for Large & Sparse Data. In *SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 375–384.

Qi, Y., Comon, P., and Lim, L.-H. (2016). Semialgebraic geometry of nonnegative tensor rank. *SIAM Journal on Matrix Analysis and Applications*, 37(4):1556–1580.

Quattoni, A., Balle, B., Carreras, X., and Globerson, A. (2014). Spectral regularization for max-margin sequence tagging. In *International Conference on Machine Learning*, pages 1710–1718.

Rakitsch, B., Lippert, C., Stegle, O., and Borgwardt, K. (2013). A lasso multi-marker mixed model for association mapping with population structure correction. *Bioinformatics*, 29(2):206–214.

Ramírez, I. and Sapiro, G. (2012). Low-rank data modeling via the minimum description length principle. In *IEEE International Conference on Acoustics, Speech and Signal Processing*, pages 2165–2168.

Ravuri, S., Mohamed, S., Rosca, M., and Vinyals, O. (2018). Learning Implicit Generative Models with the Method of Learned Moments. In *International Conference on Machine Learning*, pages 4311–4320.

Rixen, D., Siegel, J. H., and Friedman, H. P. (1996). " Sepsis/SIRS," physiologic classification, severity stratification, relation to cytokine elaboration and outcome prediction in posttrauma critical illness. *Journal of Trauma and Acute Care Surgery*, 41(4):581–598.

Roberts, S. J. (1999). Novelty detection using extreme value statistics. *IEE Proceedings-Vision, Image and Signal Processing*, 146(3):124–129.

Roberts, S. J., Husmeier, D., Rezek, I., and Penny, W. (1998). Bayesian approaches to gaussian mixture modeling. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 20(11):1133–1142.

Robeva, E. M. (2016). *Decomposing matrices, tensors, and images*. PhD thesis, University of California, Berkeley.

Roch, S. (2006). A short proof that phylogenetic tree reconstruction by maximum likelihood is hard. *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, 3(1):92–94.

Rosenberg, N. A., Pritchard, J. K., Weber, J. L., Cann, H. M., Kidd, K. K., Zhivotovsky, L. A., and Feldman, M. W. (2002). Genetic structure of human populations. *Science*, 298(5602):2381–2385.

Ruffini, M. (2018). Hierarchical Methods of Moments for Clustering High Dimensional Binary Data. *Submitted*.

Ruffini, M., Casanellas, M., and Gavaldà, R. (2018). A new method of moments for latent variable models. *Machine Learning*, 107(8-10):1431–1455.

Ruffini, M., Gavalda, R., and Limon, E. (2017a). Clustering patients with tensor decomposition. In *Machine Learning for Healthcare Conference*, pages 126–146.

Ruffini, M., Rabusseau, G., and Balle, B. (2017b). Hierarchical methods of moments. In *Neural Information Processing Systems*, pages 1901–1911.

Sanchez, E. and Kowalski, B. R. (1990). Tensorial resolution: a direct trilinear decomposition. *Journal of Chemometrics*, 4(1):29–45.

Savaresi, S. M. and Boley, D. L. (2001). On the performance of bisecting K-means and PDDP. In *SIAM International Conference on Data Mining*, pages 1–14. SIAM.

Schwarz, G. et al. (1978). Estimating the dimension of a model. *The annals of statistics*, 6(2):461–464.

Sidiropoulos, N. D., De Lathauwer, L., Fu, X., Huang, K., Papalexakis, E. E., and Faloutsos, C. (2017). Tensor decomposition for signal processing and machine learning. *IEEE Transactions on Signal Processing*, 65(13):3551–3582.

Sievert, C. and Shirley, K. (2014). LDAvis: A method for visualizing and interpreting topics. In *Workshop on interactive language learning, visualization, and interfaces*, pages 63–70.

Squires, S., Prügel-Bennett, A., and Niranjan, M. (2017). Rank selection in nonnegative matrix factorization using minimum description length. *Neural computation*, 29(8):2164–2176.

Steinbach, M., Karypis, G., Kumar, V., et al. (2000). A comparison of document clustering techniques. In *KDD workshop on text mining*, volume 400, pages 525–526.

Stewart, G. (1990). Perturbation Theory for the Singular Value Decomposition. In *SVD and Signal Processing, II: Algorithms, Analysis, and Applications*.

Stewart, G. and Sun, J.-g. (1990). *Matrix Perturbation Theory*. Computer science and scientific computing. Academic Press.

Sun, Z., Rosen, O., and Sampson, A. R. (2007). Multivariate Bernoulli mixture models with application to postmortem tissue studies in schizophrenia. *Biometrics*, 63(3):901–909.

Tomasi, G. and Bro, R. (2006). A comparison of algorithms for fitting the parafac model. *Computational Statistics & Data Analysis*, 50(7):1700–1734.

Tucker, L. R. (1966). Some mathematical notes on three-mode factor analysis. *Psychometrika*, 31(3):279–311.

Van Der Veen, A.-J. and Paulraj, A. (1996). An analytical constant modulus algorithm. *IEEE Transactions on Signal Processing*, 44(5):1136–1155.

von Luxburg, U. (2007). A tutorial on spectral clustering. *Statistics and Computing*, 17(4):395–416.

Walt, S. v. d., Colbert, S. C., and Varoquaux, G. (2011). The NumPy array: a structure for efficient numerical computation. *Computing in Science & Engineering*, 13(2):22–30.

Wang, Y., Chen, R., Ghosh, J., Denny, J. C., Kho, A., Chen, Y., Malin, B. A., and Sun, J. (2015). Rubik: Knowledge guided tensor factorization and completion for health data analytics. In *SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 1265–1274.

World Health Organization (2004). *International statistical classification of diseases and related health problems*, volume 1. World Health Organization.

Yoon, J., Jordon, J., and van der Schaar, M. (2018). RadialGAN: Leveraging multiple datasets to improve target-specific predictive models using generative adversarial networks. In *International Conference on Machine Learning*, pages 5699–5707.

Yu, L., Zhang, W., Wang, J., and Yu, Y. (2017). SeqGAN: Sequence Generative Adversarial Nets with Policy Gradient. In *AAAI Conference on Artificial Intelligence*, volume 31, pages 2852–2858.

Yu, Y., Wang, T., Samworth, R. J., et al. (2015). A useful variant of the Davis–Kahan theorem for statisticians. *Biometrika*, 102(2):315–323.

Zarowski, C. J. (1998). The MDL criterion for rank determination via effective singular values. *IEEE transactions on signal processing*, 46(6):1741–1744.

Zhang, T. and Golub, G. H. (2001). Rank-one approximation to high order tensors. *SIAM Journal on Matrix Analysis and Applications*, 23(2):534–550.

Zhang, Y., Chen, X., Zhou, D., and Jordan, M. I. (2014). Spectral methods meet EM: A provably optimal algorithm for crowdsourcing. In *Neural Information Processing Systems*, pages 1260–1268.

Zou, J. Y., Hsu, D. J., Parkes, D. C., and Adams, R. P. (2013). Contrastive learning using spectral methods. In *Neural Information Processing Systems*, pages 2238–2246.