Improved Guarantees for Learning GMMs

by

Allen Liu

Submitted to the Department of Electrical Engineering and Computer Science
in partial fulfillment of the requirements for the degree of

Masters of Science in Computer Science and Engineering

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

September 2022

Author .....
Department of Electrical Engineering and Computer Science
August 26, 2022

Certified by .....
Ankur Moitra
Norbert Wiener Professor of Mathematics, MIT
Thesis Supervisor

Accepted by .....
Leslie A. Kolodziejski
Professor of Electrical Engineering and Computer Science
Chair, Department Committee on Graduate Students

Improved Guarantees for Learning GMMs

by

Allen Liu

Submitted to the Department of Electrical Engineering and Computer Science
on August 26, 2022, in partial fulfillment of the
requirements for the degree of
Masters of Science in Computer Science and Engineering

Abstract

Mixtures of Gaussians (GMMs) are one of the most commonly used statistical models. They are typically used to model data coming from two or more heterogenous sources and have applications in a wide variety of fields including statistics, biology, physics and computer science. A fundamental task at the core of many of these applications is to learn the parameters of a mixture of Gaussians from samples. Starting with the seminal work of Karl Pearson in 1894 [81], there has been a long line of work on this problem [32, 6, 93, 48, 63, 78, 59, 49, 46, 39, 13, 65].

Despite extensive work, several important questions have remained open for decades. We address two of those here. First, we study the problem of clustering in polynomial time, in terms of both the dimension d and number of components k. While an exponential dependence on k is necessary for learning in the worst case, it is possible to achieve a better dependence if one assumes that the components are clusterable. More precisely, for a mixture of k isotropic Gaussians in $\mathbb{R}^d$, as long as the means are separated by $\Omega(\sqrt{\log k})$, then it is information-theoretically possible to cluster and learn the parameters in polynomial time. Despite recent advances [67, 55, 46], existing polynomial time algorithms all require a larger separation of $\Omega(k^\delta)$ for some $\delta > 0$. In this work, we give an algorithm that has runtime and sample complexity $\mathrm{poly}(k, d)$ and provably works with essentially minimal separation.

Second, we seek to address robustness. In particular, real data generally does not come from a distribution that is exactly a mixture of Gaussians, but rather a distribution that is close to a mixture of Gaussians. To address this, we consider a more challenging setting, that is now ubiquitous in the field of robust statistics, where an ϵ-fraction of the datapoints may be arbitrarily altered, potentially adversarially. There has been a flurry of recent work towards developing robust algorithms for learning mixtures of Gaussians [39, 13, 65], but these results all require restrictions on the class of mixtures considered. In this work, we give an algorithm that attains provable robustness guarantees and works in full generality.

Thesis Supervisor: Ankur Moitra
Title: Norbert Wiener Professor of Mathematics, MIT

Acknowledgments

I would first like to thank my advisor, Ankur Moitra, for his invaluable advice and support. His guidance has been instrumental in my growth ever since I began working on research with him many years ago, when I was still an undergrad. I would also like to thank Jerry Li, who mentored me for two summers as an intern at Microsoft Research. He has helped me expand my research horizon and introduced me to many new and exciting research directions.

I would also like to thank many other wonderful mentors that I have had throughout the years. In particular, I would like to thank Alex Iosevich, Zeev Dvir, Jon Schneider, Renato Paes Leme and Morteza Zadimoghaddam for their support and guidance over the past several years. I would also like to thank the many great collaborators that I have been fortunate to work with including, but not limited to, Anders Aamand, Sara Ahmadian, Ainesh Bakshi, Clément Canonne, Zachary Chase, Sitan Chen, Sam Hopkins, Brice Huang, Jonathan Kelner, Shyam Narayanan, Martin Pal, Binghui Peng, Mark Sellke, Aaron Sidford, Balasubramanian Sivan, Kevin Tian, and Morris Yau. It has been a pleasure working together on all of our projects, both successful and unsuccessful.

I would like to thank the NSF and the Fannie and John Hertz Foundation for generously supporting my graduate studies. I would also like to thank all of my friends from undergrad, grad school and elsewhere. While there are too many to name here, I want to make sure to express my appreciation for everyone that has helped me keep my morale up and made my experience infinitely more enjoyable. Finally, and most importantly, I would like to thank my parents for their unwavering support.

Contents

Chapter 1

Introduction

1.1 Background

Mixtures of Gaussians (GMMs) are one of the most enduring, well-weathered models of applied statistics. First introduced by Karl Pearson in his seminal work in 1894 [82], since then, GMMs have found a wide variety of applications spanning statistics, biology, physics and computer science. Formally, a mixture of Gaussians $\mathcal{M}$ with k components (abbreviated k-GMM) in d dimensions is a distribution specified by k non-negative mixing weights $w_1, \ldots, w_k$ which sum to 1, and k component d-dimensional Gaussians $N(\mu_1, \Sigma_1), \ldots, N(\mu_k, \Sigma_k)$ where $N(\mu_i, \Sigma_i)$ denotes a Gaussian with mean μ_i and covariance Σ_i. The probability density function of the mixture $\mathcal{M}$ is given by

$$\mathcal{M} = \sum_{i=1}^{k} w_i N(\mu_i, \Sigma_i) \ .$$

In other words, to draw a sample from $\mathcal{M}$, we select the i-th component with probability w_i, and then draw an independent sample from $N(\mu_i, \Sigma_i)$. Henceforth, we will use k to denote the number of components and d to denote the dimension of the space.

Typically, GMMs are used to model data that may come from several possible sources, e.g. subpopulations within a population. The data from each source is modeled by a single

Gaussian, corresponding to a component in the mixture. Usually, the data we receive is unlabelled and mixed between the various sources. Given this unlabelled data, a question that often arises is what can we learn about the components of the original mixture? There are myriad of possible approaches to this problem. However, many early approaches were heuristic or only asymptotic in nature i.e. it is not clear how many samples they require to learn the components up to some desired accuracy or whether the estimates can be computed efficiently. In a seminal work, Sanjoy Dasgupta [32] introduced the problem to theoretical computer science and asked:

Is there an efficient algorithm for provably learning the parameters of the mixture?

1.2 The Method of Moments

Over the past few decades, there has been extensive work on learning GMMs. The method of moments is a general framework for learning latent variable models, dating back to Pearson's work in 1894 [82], that has been at the heart of many advances [64, 79, 21, 53, 67, 55, 46, 39, 44]. At a high-level, the method of moments proceeds as follows. Given samples from an unknown distribution $\mathcal{M}$, we can estimate the moments $\mathbb{E}_{x\sim\mathcal{M}}[x^t]$ empirically, using the samples. We refer to t as the degree of the moment; note that $t = 1$ corresponds to the mean and $t = 2$ is related to the variance, while higher degree moments contain more precise information about the tails of the distribution. On the other hand, if we knew the true parameters of the distribution $\mathcal{M} = \sum_{i=1}^{k} w_i N(\mu_i, \Sigma_i)$, we could write the moments as explicit polynomials in terms of the parameters. We could write

$$\mathop{\mathbb{E}}_{x\sim\mathcal{M}}[x^t] = Q_t(\{w_i\}, \{\mu_i\}, \{\Sigma_i\})$$

for some polynomial Q_t. In fact, we can explicitly compute these polynomials Q_t. Thus, to learn the parameters, it suffices to empirically estimate some number of moments and then set up and solve a polynomial system of equations for the unknowns $\{w_i\}, \{\mu_i\}, \{\Sigma_i\}$. Of course, there are still many complications to deal with. It is necessary to analyze the stability

of this system as we only have polynomially many samples and thus can only estimate the moments of $\mathcal{M}$ to a certain accuracy. Also, it is necessary to give an efficient algorithm for solving this polynomial system as generic methods would require exponential time. Most importantly, to learn the true parameters, we need to argue about the uniqueness of solutions to this polynomial system – we need to prove that any valid solution for the unknowns must be close to the true parameters. In other words, we need to prove *identifiability* – that any two mixtures $\mathcal{M}, \mathcal{M}'$ that are close on a certain number of moments must actually be close in their parameters. It turns out that a proof of identifiability is at the heart of virtually all parameter learning algorithms.

Over the decades, progress on learning GMMs has generally relied on gaining a better understanding of a few fundamental questions concerning identifiability. Firstly, how many moments are necessary for identifiability? Since we only have polynomially many samples and runtime, we can only afford to estimate a finite number of moments. Furthermore, high-degree moments also require more samples to estimate to good accuracy. Thus, it is crucial that learning algorithms and their associated proofs of identifiability only require a bounded number of moments. Understanding the trade-offs between the number of moments being used and the sample complexity and computational efficiency is at the core of improving algorithms for learning GMMs.

Secondly, how strong is the quantitative relationship in the identifiability? To get an algorithm with polynomial sample complexity, it is necessary to prove a polynomial relationship for the distance between the moments of two mixtures $\mathcal{M}, \mathcal{M}'$ and the distance between their parameters. This polynomial relationship may depend on ϵ (the accuracy), k and d. On the other hand, we can view this relationship as the "robustness" of the algorithm. It roughly governs how much the moment estimates can be perturbed while still guaranteeing to learn the parameters. If one hopes to develop robust algorithms (in a sense that will be more precisely specified in Section 1.4.1), it is important to prove quantitatively stronger forms of identifiability where the quantitative relationship does not depend on the dimension d.

With these overarching questions in mind, we now delve into more concrete learning goals and discuss our contributions.

1.3 Clustering

Many early works on learning GMMs were based on clustering i.e. grouping the samples into which component generated them [34, 6, 93, 3, 26, 68]. Formally, given n independent samples $\{X_1, \ldots, X_n\}$ drawn from an unknown k-GMM, the goal of clustering is to recover a partition of the data into k parts $S_1, \ldots, S_k$ so that each part corresponds to exactly one component of the original mixture and contains essentially all of the samples generated from that component.

Of course, clustering is only possible if one assumes that there is some separation between the components. An overarching theme in clustering is to understand what separation assumptions are necessary and how these assumptions affect the sample complexity and efficiency of clustering algorithms. We focus on an important special class of GMMs, namely *isotropic* GMMs (also sometimes called spherical), where all of the covariances are equal to the identity matrix i.e. $\Sigma_1, \ldots, \Sigma_k = I$. In this case, separation between components amounts to separation between their means. In other words, we assume $\|\mu_i - \mu_j\|_2 \geq \Delta$ for all $i \neq j$ for some parameter Δ. The question now becomes how small can we take Δ so that we can still cluster?

1.3.1 Separation Assumptions

Even isotropic GMMs have proven to be remarkably complex and there is a rich literature on clustering algorithms for this case. Information-theoretically, it is known (see e.g. [84]) that $\Delta = \Theta(\sqrt{\log k})$ is both necessary and sufficient (as long as the mixing weights are lower bounded by say $w_i \geq 1/\text{poly}(k)$). However, the landscape becomes more complicated if we ask for efficient algorithms i.e. sample complexity and runtime $\text{poly}(k, d)$. The first noteworthy algorithmic result by Dasgupta in 1999 [32] gives a polynomial time algorithm when

the separation is at least $\Omega(\sqrt{k})$. Follow-up works [6, 93] improve the required separation to $\Omega(k^{1/4})$. These works can be viewed as using only degree-2 moment information – the algorithms use only very simple low-degree tests to cluster the data.

Recent advances [67, 55, 46], based on the sum-of-squares hierarchy, make further improvements. Their algorithms have sample complexity and runtime $\mathrm{poly}(d^{1/\gamma}, k)$ when the separation is $\Delta = \Omega(k^{\gamma})$ for some $\gamma > 0$. These works can be viewed as using degree-$1/\gamma$ moment information to cluster down to separation $\sim k^{\gamma}$. Unfortunately, if we wish to achieve the optimal separation of $\Omega(\sqrt{\log k})$—or indeed, any polylogarithmic amount of separation—these algorithms would require quasipolynomial runtime and sample complexity.

The barrier at quasipolynomial time The previously mentioned algorithms all get stuck at quasipolynomial time when $\Delta = \Theta(\mathrm{poly}(\log k))$ for the same reason. Fundamentally, they all rely on the following geometric identifiability fact:

> *Given enough samples from an isotropic k-GMM with separation $\Delta = \Omega(k^{\gamma})$, then any sufficiently large subset of samples whose empirical moments of degree up to $t = \Theta(1/\gamma)$ approximately match those of a standard Gaussian must essentially recover one of the true clusters.*

Algorithmically, this amounts to finding a subset of points whose empirical t-th moment tensor is close to that of a standard Gaussian (i.e. $N(0, I)$) in the appropriate norm. Since this results in a computationally hard optimization problem whenever $t > 2$, these algorithms often solve some suitable relaxation of this using e.g. the sum-of-squares hierarchy. Crucially though, as the separation decreases, the number of moments that must be matched grows. In particular, to achieve $\Delta = \Theta(\mathrm{poly}\log k)$, one must set $t = \mathrm{poly}\log k$, that is, one must match polylogarithmically many moments. However, even writing down the degree $t = \mathrm{poly}\log k$ moment tensor requires quasipolynomial time, and guaranteeing that the empirical moment tensor is a reasonable approximation to the truth requires quasipolynomially many samples. As a result, the aforementioned algorithms all require quasipolynomial time and sample

complexity, as they need to not only write down the moment tensor, but perform some fairly complex optimization tasks on top of it.

On the flip side, there is no concrete reason for pessimism either. While there are lower bounds against large classes of efficient algorithms for clustering mixtures of arbitrary Gaussians, see e.g. [45, 25], none of these apply when the components are isotropic. In particular, this leaves open the appealing possibility that one could even cluster down to separation $\Delta = \Theta(\sqrt{\log k})$ in polynomial time. Stated another way:

Can we cluster any clusterable mixture of isotropic Gaussians in polynomial time?

Our Results In this work, we (almost) resolve this question in the affirmative. Namely, for any constant $c > 0$, we give an algorithm which takes polynomially many samples and time, and which can cluster with high probability, so long as the separation satisfies $\Delta = \Omega(\log^{1/2+c} k)$. In other words, our algorithm can almost match the information theoretically optimal separation, up to sub-polylogarithmic factors. The main conceptual contribution of our work that allows us to circumvent the quasipolynomial barrier for previous algorithms is a novel way to implicitly access degree $t = O(\log k / \log\log k)$ moment information with polynomially many samples and time. We do so by carefully constructing an implicitly maintained projection that maps moment tensors living in a d^t-dimensional space (which are too large to store naively) down to a k-dimensional subspace, while still preserving all of the necessary information about the components. In other words, we still need to use moment information of degree $\sim \log k$ but develop a novel way to process this information that requires only polynomial sample complexity and runtime. More broadly, we believe that our techniques for implicitly representing and manipulating moment tensors may have further applications in speeding up other moment-based algorithms such as for tensor decomposition [57, 75, 85, 56] and refuting random CSPs [83]. Our main result for clustering GMMs is stated below.

Theorem 1.3.1 (Informal)**.** *Let $c > 0$ be fixed but arbitrary. Let $\mathcal{M}$ be a mixture of k isotropic Gaussians in $\mathbb{R}^d$ with minimum mixing weight lower bounded by $1/\mathrm{poly}(k)$, and*

minimum mean separation at least $\Delta = \Omega(\log^{1/2+c} k)$*. Then, there is an algorithm which, given samples* $X_1, \ldots, X_n \sim \mathcal{M}$ *for* $n = \mathrm{poly}(k, d)$*, outputs a clustering which is correct for all of the points, with high probability. Moreover, this algorithm runs in time* $\mathrm{poly}(d, k)$*.*

We briefly remark that we can handle arbitrary mixing weights as well, but for simplicity we only state the theorem here in the more natural regime where all the mixing weights are not too small. Our results and proofs are discussed in more detail in Chapter 2. Our main theorem for clustering is stated formally in Theorem 2.2.5. We also generalize our result beyond GMMs to mixtures of translates of a distribution satisfying the Poincaré inequality under a mild additional condition (see Theorem 2.2.3).

1.4 Learning without Clustering

Given samples from an unknown GMM, clustering is a natural goal. Unfortunately, clustering may be asking for too much – when the components of a GMM overlap with each other, clustering is no longer possible. Nevertheless, we may still hope to learn the parameters of the mixture, namely the weights $w_1, \ldots, w_k$ and means and covariances $\mu_1, \ldots, \mu_k, \Sigma_1, \ldots, \Sigma_k$, even when the components overlap. In this vein, Kalai, Moitra and Valiant [63] gave an algorithm for learning the parameters of a mixture of two Gaussians that works even if the components are almost entirely overlapping. Their approach was based on reducing the high-dimensional problem to a series of one-dimensional problems and exploiting the structure of the moments in one dimension. In particular, they proved that every one-dimensional mixture of two Gaussians is uniquely determined by its first six moments. Subsequently, Moitra and Valiant [78] and Belkin and Sinha [20] were able to give an algorithm for learning the parameters of a mixture of any constant number of Gaussians in high dimensions. These algorithms crucially make use of even higher moments along with several new ingredients like methods for recursively separating out submixtures that are difficult to directly learn via one-dimensional projections. Compared to clustering, a crucial difference is that these results assume that the number of components is a constant. In other words, the runtime and

sample complexity may grow with say d^k in contrast to the previous section where we ask for runtime and sample complexity $\text{poly}(k, d)$. Actually, an exponential dependence on k is information-theoretically necessary for learning when the components may overlap [53] and thus throughout this section, we will treat k as a constant and only worry about dependence on d.

1.4.1 Robustness

The aforementioned algorithms have a key shortcoming – they break down when the data does not exactly come from a mixture of Gaussians. In fact in Karl Pearson's original application [81], and in many others, mixtures of Gaussians are only intended as an approximation to the true data generating process. The field of robust statistics was launched by the seminal works of John Tukey [90, 91] and Peter Huber [60] and seeks to address this kind of shortcoming by designing estimators that are provably robust to some ϵ-fraction of their data being altered, potentially even adversarially. The field of robust statistics had many successes and explicated some of the general principles behind what makes estimators robust [61, 51]. Provably robust estimators were discovered for fundamental tasks such as estimating the mean and covariance of a distribution and for linear regression. There are a variety of types of robustness guarantees but the crucial point is that these estimators can all tolerate a constant fraction of corruptions that is *independent of the dimension*. However all of these estimators turn out to be hard to compute in high-dimensions [62, 22, 52].

Recently Diakonikolas et al. [40] and Lai et al. [70] designed the first provably robust and computationally efficient estimators for the mean and covariance. They operate under some kind of assumption on the uncorrupted data – either that they come from a simple generative model like a single Gaussian or that they have bounded moments. To put this in perspective, without corruptions this is a trivial learning task because it is possible to learn the mean and covariance of any distribution with bounded moments by simply using the empirical mean and empirical covariance respectively. Algorithmic robust statistics has transformed into a highly active area [42, 29, 15, 66, 41, 55, 67, 73, 86, 43, 14, 30] with many

successes. Since then, a much sought-after goal has been to answer the following challenge:

> *Is there a provably robust and computationally efficient algorithm for learning mixtures of Gaussians? Can we robustify the existing learning results?*

There has been steady progress on this problem. Diakonikolas et al. [40] gave a robust algorithm for learning mixtures of spherical Gaussians. In recent breakthroughs Bakshi and Kothari [13] and Diakonikolas et al. [39] gave a robust algorithm for learning clusterable mixtures of Gaussians using the powerful sum-of-squares hierarchy [80]. Building on this, Kane [65] gave a robust algorithm for learning mixtures of two Gaussians. We note that these works do place some mild restrictions on the mixing weights and the component covariance matrices.

1.4.2 Key Challenges

Robust Identifiability As mentioned before, behind every polynomial time algorithm for learning the parameters of a mixture model is a proof of identifiability. It needs to be proven that if two mixtures are ϵ close on some family of test functions (usually moments), then they must be poly(d, ϵ) close in their parameters. This is called *polynomial identifiability* [89, 77]. Because this relationship allows a polynomial dependence on d, the precise choices for how to measure distance between moments and distance between parameters generally do not matter.

However we need much stronger bounds when it comes to robust learning problems where we want to be able to tolerate a constant fraction of corruptions that is dimension-independent. In particular, we need to prove that for a certain measure of distance, whenever we have two mixtures that are ϵ-close on their moments with respect to this distance, then their parameters must be poly(ϵ)-close. Crucially, this relationship cannot involve dependence on the dimension d. We will call this *robust identifiability*. Recall that the non-robust learning algorithms for mixtures of Gaussians reduce to a series of one-dimensional problems. Unfortunately this strategy inherently introduces polynomial factors in d and it cannot give what we are after. For the special case of clusterable mixtures of Gaussians, Bakshi and

Kothari [13] and Diakonikolas et al. [39] proved robust identifiability and their approach was based on classifying the ways in which two single Gaussians can be very separated (i.e. have total variation distance close to one). When it comes to the more general problem of handling mixtures where the components can overlap non-trivially it seems difficult to follow the same route because we can no longer match components from the two mixtures to each other and peel them off.

Reasoning About the Sum-of-Squares Relaxation Even if we had proved robust identifiability in the general case, we would still need to devise an efficient algorithm for learning the parameters. Recall that at a high-level, we can measure the moments of the distribution empirically and then set up a polynomial system in the parameters that we are trying to solve for. To solve this system, we rely on the sum-of-squares (SoS) hierarchy [80] which is a general framework for relaxing and solving polynomial systems via semidefinite programming and has been at the heart of recent progress on robustly learning GMMs [13, 39] and many other problems [17, 18, 58, 19]. Of course, solving polynomial systems in general requires exponential time so instead SoS solves a relaxation. One view is that the SoS hierarchy treats each low-degree monomial in the variables as an independent unknown and then computes a pseudo-expectation which assigns values to these monomials while enforcing certain consistency constraints. Roughly, the consistency constraints force these assigned values to, in some vague sense, behave like a real solution i.e. as if we had actually assigned values to the original variables and simply evaluated the corresponding monomials. The SoS hierarchy gives a natural way to incorporate systems of polynomial constraints into a relaxation which can model complex primitives. It can often be used to turn proofs (e.g. of identifiability) into algorithms. In particular, if we prove some property about the actual solution to a polynomial system and the proof only uses certain types of allowable steps, then the proof can be translated into the SoS framework and the same properties must hold for the pseudo-expectation obtained by solving the SoS relaxation of the polynomial system. Furthermore, the pseudo-expectation can be efficiently computed. Thus at a high-level, we can hope to translate our proof of robust identifiability into the SoS framework to deduce a

sort of robust identifiability of the pseudo-expectation obtained from the SoS relaxation. We can then extract information about the components of the original mixture from this pseudo-expectation. Of course, it is often quite challenging to set up the polynomial system in a way that all of the necessary constraints are enforced and so that the proof of identifiability can actually be translated into the SoS framework [57, 54]. Previous works [13, 39] have a comparatively easier task because they can set up the system to find individual components of the mixture. However, in our setting this is no longer possible and our system must involve all of the components of the mixture simultaneously. It is clear that we need to exploit the structure of the moments of a mixture of Gaussians but we cannot reduce to low dimensions as in [79, 64]. The structure of moments in high dimensions is very complex so how exactly do we leverage them in our analysis?

Our Results In this work, we solve the question of robustly learning mixtures of Gaussians in full generality modulo mild assumptions on the mixing weights. We overcome both of the aforementioned obstacles using the same approach. We store the relevant moments and variables we would like to solve for in certain generating functions. We use the structure of the moments of a mixture of Gaussians to write a concise generating function that stores all of the necessary moment information. Then by manipulating the generating functions using differential operators to algebraically isolate the parameters of individual components, we are able to prove robust identifiability. Furthermore, these manipulations are all formal algebraic manipulations that can be translated into the SoS framework and thus we can reason about the SoS relaxation of the polynomial system that encodes the parameters that we want to learn. In particular, we show that from the solution to this SoS relaxation, we can still extract the parameters of the original mixture. More broadly, learning GMMs may just be a first step. We believe that our technique of using generating functions and differential operators to leverage structure in the moments is quite general and could have further applications for learning other types of latent variable models. Our main result is stated below.

Theorem 1.4.1. *[Informal] Let k be a constant. Let $\mathcal{M} = w_1 G_1 + \cdots + w_k G_k$ be a mixture*

of Gaussians in $\mathbb{R}^d$ whose components are non-degenerate and such that the mixing weights and TV distances between different components are lower bounded. Given $n = \mathrm{poly}(d/\epsilon)$ samples from $\mathcal{M}$ of which an ϵ-fraction may be arbitrarily corrupted, there is an algorithm that runs in time $\mathrm{poly}(n)$ and with high probability outputs a set of mixing weights $\widetilde{w_1}, \ldots, \widetilde{w_k}$ and components $\widetilde{G_1}, \ldots, \widetilde{G_k}$ that are $\mathrm{poly}(\epsilon)$-close to the true components (up to some permutation).

Our results and proofs are discussed in more detail in Chapter 3. Our main theorem for robustly learning mixtures of Gaussians is stated formally in Theorem 3.8.3. In Section 3.1.1, we include a more detailed discussion of the mixing weight assumptions and concurrent work [11] which obtains similar results and subsequent improvements in [74] and [12].

1.5 Other Related Work

The literature on learning GMMs is vast and here we briefly survey some other lines of work. There are several lines of work that seek to circumvent the exponential in k lower bound of [53] for learning general mixtures of Gaussians by weakening the learning goal. Firstly, there are approaches based on tensor decompositions [59, 23, 49] that get $\mathrm{poly}(k, d)$ runtime and sample complexity and use only moments up to degree 3 or 4 but need to assume that the parameters are non-degenerate and subject to some kind of random smoothing.

There are also lines of work that consider easier learning goals such as *proper* or *semi-proper* learning, where the goal is to output a mixture of Gaussians which is close to the unknown mixture in statistical distance (but not necessarily with the same components). A learning algorithm is said to be proper if its output is a mixture of k Gaussians, where k is the number of components in the true mixture, and semi-proper if it outputs a mixture of $k' \geq k$ Gaussians. While the sample complexity of proper learning is polynomial in all parameters, all known algorithms still incur a runtime which is exponential in k, even in the univariate setting [47, 36, 2, 72, 8]. When the hypothesis is only constrained to be semi-proper, polynomial time algorithms are known in the univariate setting [38, 24, 71], but these

do not extend to high dimensional settings. In the more challenging high dimensional regime, a remarkable recent result of [44] demonstrate that for a mixture of isotropic Gaussians, one can achieve semi-proper learning with quasipolynomial sample complexity and runtime. They do this by explicitly constructing a small cover for the candidate means, by techniques inspired by algebraic geometry.

An even weaker goal that has been commonly considered is that of *density estimation*, where the objective is to output any hypothesis which is close to the unknown mixture in statistical distance. Efficient (in fact, nearly optimal) algorithms are again known for this problem in low dimensions, see e.g. [38, 28, 27, 1], but as was the case with proper learning, these techniques do not extend nicely to high dimensional settings. In high dimensions, there has been a line of work [10, 9] that focuses on achieving optimal sample complexity even with robustness. However, these works do not give efficient algorithms as they are based on discretization and brute-force search, requiring exponential time.

We also mention a line of work studying the theoretical behavior of the popular *expectation-maximization* (EM) algorithm for learning mixtures of Gaussians [35, 37, 96]. However, while the above works can prove that the dynamics of EM converge in limited settings, it is known that EM fails to converge in general, even for mixtures of three Gaussians [95, 37].

Chapter 2

Clustering Mixtures of Gaussians

2.1 Overview

In this chapter, we focus on clustering isotropic GMMs. Recall that this means we receive samples say $X_1, \ldots, X_n$ from an unknown mixture of k isotropic Gaussians in $\mathbb{R}^d$

$$\mathcal{M} = w_1 N(\mu_1, I) + \cdots + w_k N(\mu_k, I)$$

with some mean separation $\|\mu_i - \mu_j\|_2 \geq \Delta$ and our goal is to group the samples into clusters corresponding to the component that generated them. Here, we give a polynomial time algorithm that can cluster with separation that nearly matches the information-theoretic optimum of $\Theta(\sqrt{\log k})$. In particular, for any constant $c > 0$, we give an algorithm that takes polynomially many samples and time, and which can cluster with high probability, so long as the separation satisfies $\Delta = \Omega(\log^{1/2+c} k)$. This is the first polynomial time algorithm that works with any poly-logarithmic separation. Our main theorem is:

Theorem 2.1.1. *[Informal, see Theorem 2.2.5, Corollary 2.2.6] Let $c > 0$ be fixed but arbitrary. Let $\mathcal{M}$ be a mixture of k isotropic Gaussians with minimum mixing weight lower bounded by $1/\mathrm{poly}(k)$, and minimum mean separation at least $\Delta = \Omega(\log^{1/2+c} k)$. Then, there is an algorithm which, given samples $X_1, \ldots, X_n \sim \mathcal{M}$ for $n = \mathrm{poly}(d, k)$, outputs a*

clustering which is correct for all of the points, with high probability. Moreover, this algorithm runs in time $\mathrm{poly}(d, k)$.

We briefly remark that we can handle arbitrary mixing weights as well, but for simplicity we only state the theorem here in the more natural regime where all the mixing weights are not too small. We also remark that a simple corollary of this is that we can also estimate the parameters of $\mathcal{M}$ to good accuracy in polynomial time. In fact, by using our algorithm as a warm start for the method proposed in [84], we can achieve arbitrarily good accuracy for parameter estimation, in polynomial time. It is known that $\Delta = \Omega(\sqrt{\log k})$ is also necessary to achieve nontrivial parameter estimation with polynomially many samples [84], so our results for parameter estimation are also almost-optimal, in terms of the separation that they handle. We note that our formal theorems are actually stated for parameter estimation, rather than clustering, however, in light of [84], these two problems are equivalent in the regime we consider.

Our main technique (as we will discuss in more detail later) extends to beyond Gaussians, and in fact also allows us to cluster any mixture of translations of a distribution $\mathcal{D}$, so long as this distribution satisfies the *Poincaré inequality*, under a mild technical condition. Recall that a distribution $\mathcal{D}$ over $\mathbb{R}^d$ is said to be σ-Poincaré if for all differentiable functions $f : \mathbb{R}^d \to \mathbb{R}$, we have

$$\operatorname*{\mathsf{Var}}_{X \sim \mathcal{D}}[f(X)] \leq \sigma^2 \cdot \mathop{\mathbb{E}}_{X \sim \mathcal{D}}\left[\|\nabla f(X)\|_2^2\right] .$$

This condition is widely studied in probability theory, and is satisfied by many natural distribution classes. For instance, isotropic Gaussians are 1-Poincaré, and any isotropic logconcave distribution is ψ-Poincaré, where ψ is the value of the KLS constant (see e.g. [88] for an overview of the KLS conjecture).

In fact, the family of Poincaré distributions is the most general family of distributions for which the SoS-based methods for clustering [67, 55, 46] discussed in the introduction are known to work. For a mixture of 1-Poincaré distributions, [67] shows that if the minimum mean separation is $\Delta = \Omega(k^\gamma)$, then their algorithm successfully clusters the points in time $O(d^{\mathrm{poly}(1/\gamma)})$. As before, when the separation is polylogarithmic, the runtime and sample

complexity of their method is once again quasipolynomial.

We show that one can improve the runtime and sample complexity to polynomial time, under two additional assumptions: first, the mixture must consist of translated versions of the same Poincaré distribution which we can get samples from, and second, the maximum and minimum separations between any two components must be polynomially related. More concretely, we show:

Theorem 2.1.2 (informal, see Theorem 2.2.3, Corollary 2.2.4)**.** *Let $c > 0$ be fixed but arbitrary. Let $\mathcal{D}$ be a fixed distribution with mean zero over $\mathbb{R}^d$ which is 1-Poincaré. Let $\mathcal{M}$ be a mixture of k distributions where each component is of the form $\mu_i + \mathcal{D}$. Assume that the minimum mixing weight in this distribution is at least $1/\mathrm{poly}(k)$, and moreover, assume that*

$$\min_{i \neq j} \|\mu_i - \mu_j\|_2 \geq \Omega(\log^{1+c} k)$$
$$\max_{i \neq j} \|\mu_i - \mu_j\|_2 \leq \mathrm{poly}\left(\min_{i \neq j} \|\mu_i - \mu_j\|_2\right) .$$

Then, there is an algorithm which, given samples $X_1, \ldots, X_n \sim \mathcal{M}$ and samples $z_1, \ldots, z_n \sim \mathcal{D}$ for $n = \mathrm{poly}(k, d)$, outputs a clustering of $X_1, \ldots, X_n$ which is correct for all of the points, with high probability. Moreover, this algorithm runs in time $\mathrm{poly}(d, k)$.

Remark. *Note that we only need the samples from $\mathcal{D}$. We do not need to actually know the distribution or access the p.d.f.*

We note that separation $\Omega(\log k)$ is optimal for general Poincaré distributions, as Poincaré distributions include some distributions with exponential tails, for which $\Omega(\log k)$ separation is necessary to cluster. Thus, the separation that we require is almost optimal for general Poincaré distributions. We also note that one immediate consequence of this theorem, alongside Chen's recent breakthrough result [31] for KLS that $\psi \leq \exp(C\sqrt{\log d \log\log d})$ for some universal constant C, and a simple application of PCA, is that we can cluster a mixture of translates of an isotropic logconcave distribution in polynomial time, as long as the separation is $\Omega(\exp(C\sqrt{\log k \log\log k}))$.

2.1.1 Our Techniques

In this section, we describe how our techniques work at a high level. Our goal will be to devise a method which can, given samples $X, X' \sim \mathcal{M}$, detect whether or not X and X' are from the same components or from different ones.

We first make the following reduction. Notice that if $\mathcal{M}$ is a mixture with separation Δ, then $(X - X')/\sqrt{2}$ can be thought of as a sample from the *difference mixture* $\mathcal{M}'$. This is a mixture with $\binom{k}{2} + 1$ components, each with covariance I. It has one component with mean zero, and the means of the remaining components all have norm at least $\Delta/\sqrt{2}$. Moreover, given $X, X' \sim \mathcal{M}$, we have that $X - X'$ is drawn from the mean zero component of the difference mixture if and only if X, X' were drawn from the same component in the original mixture. Thus, to check if two samples from the original mixture X, X' are from the same component, it suffices to be able to detect, given a sample from the difference mixture, whether or not this sample comes from the mean zero component or not.

In the remainder of this section, in a slight abuse of notation, we will let $\mathcal{M}$ denote the difference mixture, we will assume it has k components with nonzero mean, and we will assume that all nonzero means of the difference mixture have norm at least Δ. We will henceforth refer to the components with nonzero mean as the nonzero components of the mixture. This reparameterization of the problem only changes things by polynomial factors, which do not impact our qualitative results.

Rough clustering via implicit moment tensors

The main conceptual contribution of our work is a novel way to implicitly access degree $t = O(\log k / \log \log k)$ moment information with polynomially many samples and time. We do so by carefully constructing an implicitly maintained projection map from $\mathbb{R}^{d^t}$ down to a subspace of dimension k, which still preserves meaningful information about the nonzero components. For now, let us first focus on the case where the maximum norm of any mean in the difference mixture is upper bounded by $\text{poly}(\Delta)$, or equivalently, the maximum separation between any two components in the original mixture is at most polynomially

larger than the minimum separation.

Low rank estimators for Hermite polynomials Central to our methods is the *t-th Hermite polynomial tensor*, a classical object in probability theory, which we denote $h_t : \mathbb{R}^d \to \mathbb{R}^{d^t}$. These are explicit polynomials, and are the natural analog of the univariate Hermite polynomials to high dimensions. A well-known fact about the Hermite polynomial tensor is that

$$\mathop{\mathbb{E}}_{X \sim N(\mu, I)}[h_t(X)] = \mu^{\otimes t} \,. \tag{2.1}$$

Throughout the introduction, we will treat all tensors as flattened into vectors in $\mathbb{R}^{d^t}$ (in a canonical way) e.g. we view the RHS of the above as a vector in $\mathbb{R}^{d^t}$. One simple but important implication of (2.1) is that

$$\mathop{\mathbb{E}}_{X \sim \mathcal{M}}[h_t(X)] = \sum_{i=1}^{k} w_i \mu_i^{\otimes t} \,. \tag{2.2}$$

This fact will be crucial for us going forward.

However, a major bottleneck for algorithmically using the Hermite polynomial tensors is that we cannot write down $h_t(X)$ in polynomial time when t gets large, e.g. when $t = \Omega(\log k / \log \log k)$, which is the regime we will require.

To get around this, we will use a modification of the Hermite polynomial tensors that can still be used to estimate the RHS of (2.2) but can also be easily manipulated implicitly. In particular, we will construct a *random* polynomial $R_t : \mathbb{R}^d \to \mathbb{R}^{d^t}$ i.e. we can imagine R_t is actually a polynomial in x whose coefficients are randomly chosen. The polynomial R_t (constructed in Corollary 2.9.11) satisfies two key properties.

1. $R_t(x)$ is an unbiased estimator for $h_t(x)$ with bounded variance i.e. for a fixed x, $\mathbb{E}[R_t(x)] = h_t(x)$ where the expectation is over the random coefficients of R_t.

2. For any choice of the randomness in the coefficients, the polynomial $R_t(x)$ can be

written as a *sum of polynomially many rank*-1 *tensors* i.e. tensors of the form

$$v = v_1 \otimes \ldots \otimes v_t \ , \text{ where } v_i \in \mathbb{R}^d \text{ for all } i = 1, \ldots, t \ .$$

Note that the first property implies that

$$\mathop{\mathbb{E}}_{R_t, X \sim N(\mu, I)}[R_t(X)] = \mu^{\otimes t} \ , \text{ and } \quad \mathop{\mathbb{E}}_{R_t, X \sim \mathcal{M}}[R_t(X)] = \sum_{i=1}^{k} w_i \mu_i^{\otimes t} \ . \tag{2.3}$$

The second property is the main motivation behind the definition of R_t, as it implies that we can have efficient, but restricted access to it. The key point is that if our algorithm can be implemented with techniques that only require accesses to rank-1 tensors, we can implicitly access R_t in polynomial time.

Implicitly finding the span of the tensorized means Motivated by the above discussion, our goal will be to find a projection matrix $\Pi : \mathbb{R}^{d^t} \to \mathbb{R}^k$ with the following properties:

(i) **Efficient application** If $v \in \mathbb{R}^{d^t}$ is a rank-1 tensor, then Πv can be evaluated in polynomial time.

(ii) **Zero component is small** If $X \sim N(0, I)$, then $\|\Pi R_t(X)\|_2 < k^{50}$ with high probability.

(iii) **Nonzero components are large** If $X \sim \mathcal{M}$ is a sample from a nonzero component of the difference mixture, then $\|\Pi R_t(X)\|_2 \geq k^{100}$ with high probability.

Given such a projection map, our clustering procedure is straightforward: given two samples X, X' from the original mixture, we apply the projection map to many copies of $R_t((X - X')/\sqrt{2})$, and cluster them in the same component if and only if their projected norm is small on average. We show that the above properties, as well as some facts about the concentration of R_t, imply that this clustering algorithm succeeds with high probability, assuming we have access to Π.

It thus remains how to construct Π. In fact, there is a natural candidate for such a map. Let $\mu_1, \ldots, \mu_k$ denote the means of the nonzero components, and let

$$S_t = \text{span}\left(\left\{\mu_i^{\otimes t}\right\}_{i=1}^k\right) .$$

If we could find the projection $\Gamma_t : \mathbb{R}^{d^t} \to \mathbb{R}^k$ onto the subspace S_t, it can be verified that this projection map would satisfy Conditions (ii) and (iii).[1]

Moreover, there is a natural estimator for this subspace. In particular (2.3) implies that $\mathbb{E}[R_{2t}(X)]$ rearranged as a $d^t \times d^t$ matrix in a canonical way is exactly

$$\sum_{i=1}^k w_i \left(\mu_i^{\otimes t}\right)\left(\mu_i^{\otimes t}\right)^\top .$$

Notice that this matrix is rank k, and moreover, the span of its nonzero eigenvectors is exactly S_t. Consequently, if we could estimate this quantity, then find the projection onto the span of the top k eigenvectors, we would be done.

As alluded to earlier, the difficulty is that doing this naively would not be efficient; writing down any of these objects would take quasipolynomial time. Instead, we seek to find an implicit representation of Γ_t with the key property that it can be applied to rank-1 tensors in polynomial time.

We will do so by iteratively building an approximation to this subspace. Namely, we give a method which, given a good approximation to $\Gamma_{s-1} : \mathbb{R}^{d^{s-1}} \to \mathbb{R}^k$ which can be efficiently applied to flattenings of rank-1 tensors, constructs a good approximation to Γ_s with the same property. We do so by obtaining a good approximation to the $dk \times dk$ sized matrix

$$M_s = \sum_{i=1}^k w_i \left(\mu_i \otimes \Gamma_{s-1}\mu_i^{\otimes(s-1)}\right)\left(\mu_i \otimes \Gamma_{s-1}\mu_i^{\otimes(s-1)}\right)^\top . \tag{2.4}$$

[1]Here and throughout the introduction, we will assume for simplicity of exposition that the vectors $\mu_1^{\otimes s}, \ldots, \mu_k^{\otimes s}$ are linearly independent, for all $s = 1, \ldots, t$, so that S_s is always a k-dimensional subspace, for all $s = 1, \ldots, t$. In general, our algorithms work even if they are not linearly independent, and will always find a subspace which contains S_t, which will suffice for our purposes.

Notice that M_s has rank k, and moreover, the span of its k largest eigenvectors is equal to the span of $\left\{\mu_i \otimes \Gamma_{s-1}\mu_i^{\otimes(s-1)}\right\}_{i=1}^{k}$. Therefore, if we let $\Pi_s : \mathbb{R}^{dk} \to \mathbb{R}^k$ denote the projection onto the span of the k largest eigenvectors of M_s, then one can easily verify that

$$\Gamma_s = \Pi_s \left(I \otimes \Gamma_{s-1}\right) \ . \tag{2.5}$$

Moreover, if Γ_{s-1} can be efficiently applied to flattenings of rank-1 tensors in $\mathbb{R}^{d^{s-1}}$, then the form of (2.5) immediately implies that Γ_s can also be applied efficiently to flattenings of rank-1 tensors in $\mathbb{R}^{d^s}$.

It remains to demonstrate how to efficiently approximate M_s, given Γ_{s-1}. There is again a fairly natural estimator for this matrix. Namely, each component of the sum in (2.4) can be formed by rearranging the length-$(dk)^2$ vector

$$\left(I \otimes \Gamma_{s-1}\right)^{\otimes 2} \mu_i^{\otimes 2s} = \mathop{\mathbb{E}}_{R_{2s}, X \sim N(\mu_i, I)} \left[\left(I \otimes \Gamma_{s-1}\right)^{\otimes 2} R_{2s}(X)\right] \ .$$

into a $dk \times dk$ sized matrix in the canonical way. In particular, this implies that the overall matrix is the rearrangement of the length-$(dk)^2$ vector

$$\sum_{i=1}^{k} w_i \left(I \otimes \Gamma_{s-1}\right)^{\otimes 2} \mu_i^{\otimes 2s} = \mathop{\mathbb{E}}_{R_{2s}, X \sim \mathcal{M}} \left[\left(I \otimes \Gamma_{s-1}\right)^{\otimes 2} R_{2s}(X)\right] \tag{2.6}$$

into a $dk \times dk$ sized matrix in the canonical way. Since R_{2s} is a sum of polynomially many rank-1 tensors, and by our inductive hypothesis, Γ_{s-1} can be efficiently applied to rank-1 tensors, we can efficiently estimate the right hand side of (2.6), given samples from $\mathcal{M}$.

Putting it all together, this allows us to approximate M_s efficiently given Γ_{s-1}, which, by (2.5), gives us the desired expression for Γ_s. Iterating this procedure gives us a way to estimate Γ_t as a sequence of nested projection maps, i.e.

$$\Gamma_t \approx \Pi_t \left(I \otimes \Pi_{t-1} \left(I \otimes \ldots\right)\right) \ ,$$

where we can compute $\Pi_1, \ldots, \Pi_t$ efficiently, given samples from $\mathcal{M}$. This form allows us

to evaluate Γ_t efficiently on any flattening of a rank-1 tensor, thus satisfying Condition (i), and we previously argued that Γ_t satisfies Conditions (ii) and (iii). Combining all of these ingredients gives us our clustering algorithm, when the minimum and maximum separations are at most polynomially separated.

Implicit moment tensors for Poincaré distributions So far, we have only discussed how to do this implicit moment estimation for isotropic Gaussians. It turns out that all of the quantities discussed above have very natural analogues for *any* Poincaré distribution. For instance, given any Poincaré distribution $\mathcal{D}$ with zero mean, there is an explicit polynomial tensor we call the $\mathcal{D}$-*adjusted polynomial* $P_{t,\mathcal{D}}$ (see Section 2.4) that essentially satisfies all the same properties that we needed above. If we use these polynomials instead of the Hermite polynomial tensor, it turns out that all of these proofs directly lift to any Poincaré distribution. In fact, in the actual technical sections, we directly work with arbitrary Poincaré distributions, as everything is stated very naturally there. The resulting clustering algorithm immediately gives us Theorem 2.2.3.

Sample complexity Previous approaches always needed to estimate high degree moment tensors, and as a result, their sample complexity was quasipolynomial. While we may also need to estimate fairly high degree polynomials, notice that all quantities that we will deal with will be polynomially bounded. This is because we can terminate our procedure at any t which satisfies Conditions (ii) and (iii). Therefore, all the quantities that we need are polynomially large. As a result, one can verify that the polynomials we construct will also only ever have polynomially large range, with high probability. Therefore, all quantities we need to estimate can be estimated using polynomially many samples. We defer the detailed proofs outlined in this discussion to Sections 2.4, 2.5, 2.6 and 2.7. Note that in those sections, we work with a general Poincaré distribution but the outline follows the description here.

Fine-grained clustering for Gaussians

We now discuss how to handle general mixtures of isotropic Gaussians, without any assumption on the maximum separation. The problem with applying the implicit moment estimation method outlined above to this general setting is that the signal from the components in the difference mixture with relatively small mean will be drowned out by the signal from the components with much larger norm. Consequently, we can reliably cluster points from the components with large mean, but we could obtain an imperfect clustering for some components with somewhat smaller mean, and we will be unable to detect components with very small mean.

To overcome this, we devise a recursive clustering strategy. One somewhat simple approach is as follows. We first use our rough clustering algorithm described above to find a "signal direction" $v \in \mathbb{R}^d$. This direction will have the property that there is a pair of well-separated means along this direction. Thus, if we project the data points on this direction, and take only points which lie within a randomly chosen small interval on this interval, we can guarantee that with reasonable probability, we only accept points from at most half of the components of the mixture. Of course, after restricting to this interval, the resulting distribution is no longer a mixture of Gaussians. However, if we consider the projection of these accepted points to the subspace orthogonal to v, the resulting distribution will again be a mixture of fewer isotropic Gaussians. We can then recurse on this mixture with fewer components. Here, we are crucially using the fact that isotropic Gaussians remain isotropic Gaussians after slicing and projecting orthogonally.

While this strategy described above, when implemented carefully, would work down to $\Delta = \mathrm{poly}(\log k)$, it would not be able to achieve the nearly optimal separation in Theorem 2.2.5. To achieve the nearly optimal separation, there are several more technical details that need to be dealt with and thus there will be several additional steps in the algorithm. We defer the details of this to Sections 2.9, 2.10 and 2.11.

2.1.2 Related Work

The literature on mixture models—and Gaussian mixture models in particular—is incredibly vast and has already been discussed in Chapter 1 so we will focus only on the most related papers here and discuss those in more detail. Our results are most closely related to the line of work on studying efficient algorithms for clustering and parameter estimation under mean-separation conditions [33, 35, 7, 94, 46, 55, 84, 67, 44]. There are also a number of papers also generalize from mixtures of Gaussians to mixtures of more general classes of distributions [4, 69, 76, 55, 67]. These algorithms fall into two classes: either they require separation which is at least $\Omega(k^{1/2})$ or even larger, but they can handle general subgaussian distributions, or they require more structure on the higher moments of the distribution, but they can tolerate much less separation.

The most general condition under which the latter is known to work is the condition commonly referred to as *certifiable hypercontractivity*, which roughly states that the Sum-of-Squares hierarchy can certify that the distribution has bounded tails. While there is no complete characterization of what distributions satisfy this condition, the most general class of distributions for which it is known to hold is the class of Poincaré distributions [67], which is also the class of distributions we consider here.

We note that our work bears some vague resemblance to the general line of work that uses spectral-based methods to speed up Sum-of-Squares (SoS) algorithms. Spectral techniques have been used to demonstrate to speed up SoS-based algorithms in various settings such as tensor decomposition [57, 75, 85, 56] and refuting random CSPs [83]. Similarly, it has been observed that in some settings, SoS-based algorithms can be sped up, when the SoS proofs are much smaller than the overall size of the program [50, 87]. Our algorithm shares some qualitative similarities with some of these approaches—for instance, it is based on a (fairly complicated) spectral algorithm. However, we do not know of a concrete connection between our algorithm and this line of work. It is possible, for instance, that our algorithm could be interpreted as extracting a specific randomized polynomially-sized SoS proof of identifiability, but we leave further investigations of this to future work.

2.2 Formal Problem Setup and Results

In this section, we formally define the problems we consider, and state our formal results. For the remainder of this paper, we will always let $\|\cdot\|$ denote the ℓ_2 norm.

2.2.1 Clustering Mixtures of Poincare Distributions

The general problem that we study involves clustering mixtures of Poincare distributions. We begin with a few definitions.

Definition 2.2.1 (Poincare Distribution)**.** *For a parameter σ, we say a distribution $\mathcal{D}$ on $\mathbb{R}^d$ is σ-Poincare if for all differentiable functions $f : \mathbb{R}^d \to \mathbb{R}$,*

$$\operatorname*{\mathsf{Var}}_{z\sim\mathcal{D}}[f(z)] \leq \sigma^2 \operatorname*{\mathbb{E}}_{z\sim\mathcal{D}}[\|\nabla f(z)\|^2]\,.$$

Definition 2.2.2. *Let $\mathcal{D}$ be a distribution on $\mathbb{R}^d$. We use $\mathcal{D}(\mu_1)$ for $\mu_1 \in \mathbb{R}^d$ to denote the distribution obtained by shifting $\mathcal{D}$ by the vector μ_1.*

We assume that there is some σ-Poincare distribution $\mathcal{D}$ on $\mathbb{R}^d$ that we have sample access to. Since everything will be scale invariant, it will suffice to focus on the case $\sigma = 1$. For simplicity we assume that $\mathcal{D}$ has mean 0 (it is easy to reduce to this case since we can simply estimate the mean of $\mathcal{D}$ and subtract it out). We also assume that we have sample access to a mixture

$$\mathcal{M} = w_1\mathcal{D}(\mu_1) + \cdots + w_k\mathcal{D}(\mu_k)$$

where the mixing weights $w_1, \ldots, w_n$ and means $\mu_1, \ldots, \mu_k$ are unknown. We will assume that we are given a lower bound on the mixing weights $w_{\min}$. We consider the setting where there is some minimum separation between all pairs of means μ_i, μ_j so that the mixture is clusterable. In the proceeding sections, when we say an event happens with high probability, we mean that the failure probability is smaller than any inverse polynomial in $k, 1/w_{\min}$. Our main theorem is stated below.

Theorem 2.2.3. *Let $\mathcal{D}$ be a 1-Poincare distribution on $\mathbb{R}^d$. Let*

$$\mathcal{M} = w_1\mathcal{D}(\mu_1) + \cdots + w_k\mathcal{D}(\mu_k)$$

be a mixture of translated copies of $\mathcal{D}$. Let $w_{\min}, s$ be parameters such that $w_i \geq w_{\min}$ for all i and $\|\mu_i - \mu_j\| \geq s$ for all $i \neq j$. Let $\alpha = (w_{\min}/k)^{O(1)}$ be some desired accuracy (that is inverse polynomial) [2]. Assume that

$$s \geq (\log(k/w_{\min}))^{1+c}$$

for some $0 < c < 1$. Also assume that

$$\max \|\mu_i - \mu_j\| \leq s^C$$

for some C. There is an algorithm that takes $n = \text{poly}((kd/(w_{\min}\alpha))^{C/c})$ samples from $\mathcal{M}$ and $\mathcal{D}$ and runs in $\text{poly}(n)$ time and with high probability, outputs estimates

$$\widetilde{w_1}, \ldots, \widetilde{w_k}, \widetilde{\mu_1}, \ldots, \widetilde{\mu_k}$$

such that for some permutation π on $[k]$,

$$|w_i - \widetilde{w_{\pi(i)}}| \leq \alpha, \left\|\mu_i - \widetilde{\mu_{\pi(i)}}\right\| \leq \alpha$$

for all i.

Remark. *If we could remove the assumption $\|\mu_i - \mu_j\| \leq s^C$, then we would get a complete polynomial time learning result. Still, our learning algorithm works in polynomial time for mixtures where the maximum separation is polynomially bounded in terms of the minimum separation.*

[2] If d is much larger than k and we wanted inverse polynomial accuracy like $1/d$ then we can simply decrease the parameter $w_{\min}$ (and then we would need separation $\log(dk)$ instead of $\log k$)

An immediate consequence of Theorem 2.2.3 is that we can cluster samples from the mixture with accuracy better than any inverse polynomial.

Corollary 2.2.4. *Under the same assumptions as Theorem 2.2.3, we can recover the ground-truth clustering of the samples with high probability i.e. we output k clusters $S_1, \dots S_k$ such that for some permutation π on $[k]$, the set $S_{\pi(i)}$ consists precisely of the samples from the component $\mathcal{D}(\mu_i)$ for all i.*

2.2.2 Clustering Mixtures of Gaussians

In the case where the distribution $\mathcal{D}$ in the setup in Section 2.2.1 is a Gaussian, we can obtain a stronger result that works in full generality, without any assumption about the maximum separation. The result for Gaussians also works with a smaller separation of $(\log(k/w_{\min}))^{1/2+c}$ which, as mentioned before, is essentially optimal for Gaussians.

Theorem 2.2.5. *Let $\mathcal{M} = w_1 N(\mu_1, I) + \dots + w_k N(\mu_k, I)$ be an unknown mixture of Gaussians in $\mathbb{R}^d$ such that $w_i \geq w_{\min}$ for all i and $\|\mu_i - \mu_j\| \geq (\log(k/w_{\min}))^{1/2+c}$ for some constant $c > 0$. Then for any desired (inverse polynomial) accuracy $\alpha \geq (w_{\min}/k)^{O(1)}$, given $n = \text{poly}((dk/(w_{\min}\alpha))^{1/c})$ samples and $\text{poly}(n)$ runtime, there is an algorithm that with high probability outputs estimates $\{\widetilde{\mu_1}, \dots, \widetilde{\mu_k}\}$ and $\{\widetilde{w_1}, \dots, \widetilde{w_k}\}$ such that for some permutation π on $[k]$, we have*

$$|w_i - \widetilde{w_{\pi(i)}}|, \left\|\mu_i - \widetilde{\mu_{\pi(i)}}\right\| \leq \alpha$$

for all $i \in [k]$.

Again, once we have estimated the parameters of $\mathcal{M}$, it is easy to cluster samples from $\mathcal{M}$ into each of the components with accuracy better than any inverse polynomial.

Corollary 2.2.6. *Under the same assumptions as Theorem 2.2.5, with high probability, we can recover the ground-truth clustering of the samples i.e. we output k clusters $S_1, \dots S_k$ such that for some permutation π on $[k]$, the set $S_{\pi(i)}$ consists precisely of the samples from the component $N(\mu_i, I)$ for all i.*

Note that throughout this paper, we do not actually need to know the true number of components k. All of the algorithms that we write will work if instead of the number of components being k, the number of components is upper bounded by k i.e. we only need to be told an upper bound on the number of components. In fact, we can simply use $1/w_{\min}$ as the upper bound on the number of components.

2.2.3 Organization

In Section 2.3, we introduce basic notation and prove a few basic facts that will be used later on.

Clustering Test: In Sections 2.4 - 2.7, we develop our key clustering test i.e. we show how to test if two samples are from the same component or not. Note that throughout these sections, when we work with a mixture

$$\mathcal{M} = w_1\mathcal{D}(\mu_1) + \cdots + w_k\mathcal{D}(\mu_k)\,,$$

this will correspond to the "difference mixture" of the mixture that we are trying to learn and thus our goal will be to test whether a sample came from a component with $\mu_i = 0$ or μ_i far from 0.

In Section 2.4, we discuss how to construct estimators for the moments of a Poincare distribution that can be manipulated implicitly. In the end, we construct a random polynomial R_t such that

$$\mathop{\mathbb{E}}_{x\sim\mathcal{D}(\mu)}[R_t(x)] = \mu^{\otimes t}$$

and such that R_t can be written as the sum of polynomially many rank-1 tensors (see Corollary 2.4.10). In Section 2.5, we describe our iterative projection technique and explain how it can be used to implicitly store and apply a projection map $\Gamma : \mathbb{R}^{d^t} \to \mathbb{R}^k$ in polynomial time to rank-1 tensors. In Section 2.6, we combine the techniques in Section 2.4 and Section

2.5 to achieve the following: given samples from

$$\mathcal{M} = w_1\mathcal{D}(\mu_1) + \cdots + w_k\mathcal{D}(\mu_k)$$

we can find a $k \times d^t$ projection matrix Γ_t (where $t \sim \log k/\log\log k$) whose row span essentially contains all of $\mu_1^{\otimes t}, \ldots \mu_k^{\otimes t}$ (see Lemma 2.6.5). Finally, in Section 2.7, we use the projection map computed in the previous step and argue that if $x \sim \mathcal{D}(0)$ then $\|\Gamma_t R_t(x)\|$ is small and if $x \sim \mathcal{D}(\mu_i)$ for μ_i far from 0, then $\|\Gamma_t R_t(x)\|$ is large with high probability. Thus, to test a sample x, it suffices to measure the length of $\|\Gamma_t R_t(x)\|$.

Main Result for Mixtures of Poincare Distributions: In Section 2.8, we prove Theorem 2.2.3, our main result for mixtures of Poincare distributions. It will follow fairly easily from the guarantees of the clustering test in Section 2.7.

Main Result for Mixtures of Gaussians: Proving our main result for mixtures of Gaussians requires some additional work although all of the machinery from Sections 2.4 - 2.7 can still be used. In particular, we will need a few quantitatively stronger versions of the bounds in Sections 2.4 - 2.7 that exploit special properties of Gaussians in order to get down from $\log^{1+c} k$ to $\log^{1/2+c} k$ separation. We prove these stronger bounds in Section 2.9. Then in Sections 2.10 and 2.11, we show how to do recursive clustering to eliminate the assumption that the maximum separation is polynomially bounded in terms of the minimum separation. In Section 2.10, we introduce a few basic building blocks in our recursive clustering algorithm and we put them together in Section 2.11.

2.3 Preliminaries

We now introduce some notation that will be used throughout the paper. We use I_n to denote the $n \times n$ identity matrix. For matrices A, B we define $A \otimes_{\mathsf{kr}} B$ to be their Kronecker product. This is to avoid confusion with our notation for tensor products. For a tensor T,

we use $\mathsf{flatten}(T)$ to denote the flattening of T into a vector. We assume that this is done in a canonical way throughout this paper.

2.3.1 Manipulating Tensors

We will need to do many manipulations with tensors later on so we first introduce some notation for working with tensors.

Definition 2.3.1 (Tensor Notation)**.** *For an order-t tensor, we index its dimensions $\{1, 2, \ldots, t\}$. For a partition of $[t]$ into subsets $S_1, \ldots, S_a$ and tensors $T_1, \ldots, T_a$ of orders $|S_1|, \ldots, |S_a|$ respectively we write*

$$T_1^{(S_1)} \otimes \cdots \otimes T_a^{(S_a)}$$

to denote the tensor obtained by taking the tensor product of T_1 in the dimensions indexed by S_1, T_2 in the dimensions indexed by S_2, and so on for $T_3, \ldots, T_a$.

Definition 2.3.2. *For a vector x (viewed as an order-1 tensor), we will use the shorthand $x^{\otimes S}$ to denote $(x^{\otimes |S|})^{(S)}$ (i.e. the product of copies x in dimensions indexed by elements of S). For example,*

$$x^{\otimes \{1,3\}} \otimes y^{\otimes \{2,4\}} = x \otimes y \otimes x \otimes y\,.$$

Definition 2.3.3 (Tensor Slicing)**.** *For an order-t tensor T, we can imagine indexing its entries with indices $(\eta_1, \ldots, \eta_t) \in [d_1] \times \cdots \times [d_t]$ where $d_1, \ldots, d_t$ are the dimensions of T. We use the notation*

$$T_{\eta_{a_1}=b_1,\ldots,\eta_{a_j}=b_j}$$

to denote the slice of T of entries whose indices satisfy the constraints $\eta_{a_1} = b_1, \ldots, \eta_{a_j} = b_j$.

Definition 2.3.4 (Unordered Partitions)**.** *We use $Z_t(S)$ to denote all partitions of S into t unordered, possibly empty, parts.*

Remark. *Note the partitions are not ordered so $\{\{1\}, \{2\}\}$ is the same as $\{\{2\}, \{1\}\}$.*

Definition 2.3.5. *For a collection of sets $S_1, \ldots, S_t$, we define $\mathcal{C}(\{S_1, \ldots, S_t\})$ to be the number of sets among $S_1, \ldots, S_t$ that are nonempty.*

Definition 2.3.6 (Symmetrization)**.** *Let $A_1, \ldots, A_n$ be tensors such that A_i is an order a_i tensor having dimensions $\underbrace{d \times \cdots \times d}_{a_i}$ for some integers $a_1, \ldots, a_n$. We define*

$$\sum_{sym}(A_1 \otimes \cdots \otimes A_n) = \sum_{\substack{S_1 \cup \cdots \cup S_n = [a_1 + \cdots + a_n] \\ S_i \cap S_j = \emptyset \\ |S_i| = a_i}} A_1^{(S_1)} \otimes \cdots \otimes A_n^{(S_n)} .$$

In other words, we sum over all ways to tensor $A_1, \ldots, A_n$ together to form a tensor of order $a_1 + \cdots + a_n$.

2.3.2 Properties of Poincare Distributions

Here we state a few standard facts about Poincare distributions that will be used later on.

Fact 2.3.7. *Poincare distributions satisfy the following properties:*

- ***Direct Products:*** *If $\mathcal{D}_1$ and $\mathcal{D}_2$ are σ-Poincare distributions then their product $\mathcal{D}_1 \times \mathcal{D}_2$ is σ-Poincare*
- ***Linear Mappings:*** *If $\mathcal{D}$ is σ-Poincare and A is a linear mapping then the distribution Ax for $x \sim \mathcal{D}$ is $\sigma \|A\|_{op}$-Poincare*
- ***Concentration:*** *If $\mathcal{D}$ is σ-Poincare then for any L-Lipchitz function f and any parameter $t \geq 0$, we have*

$$\Pr_{z \sim \mathcal{D}}[|f(z) - \mathbb{E}[f(z)]| \geq t] \leq 6e^{-t/(\sigma L)} .$$

The following concentration inequality for samples from a Poincare distribution is also standard.

Claim 2.3.8. *Let $\mathcal{D}$ be a distribution in $\mathbb{R}^d$ that is 1-Poincare. Let $0 < \epsilon < 0.1$ be some parameter. Given $n \geq (d/\epsilon)^8$ independent samples $z_1, \ldots, z_n \sim \mathcal{D}$, with probability at least $1 - 2^{-d/\epsilon}$, we have*

$$\left\| \frac{z_1 + \cdots + z_n}{n} - \mathop{\mathbb{E}}_{z \sim \mathcal{D}}[z] \right\| \leq \epsilon .$$

2.3.3 Basic Observations

Before we begin with the main proofs of Theorem 2.2.3 and Theorem 2.2.5, it will be useful to make a few simple reductions that allow us to make the following simplifying assumptions:

- **Means Polynomially Bounded:** For all i, we have $\|\mu_i - \mu_j\| \leq O((k/w_{\min})^2)$, and
- **Dimension Not Too High:** We have $d \leq k$.

Since reducing to the case when the above assumptions hold is straight-forward, we defer the details to Section 2.12. The reductions work in both settings (general Poincare distributions and Gaussians) so in all future sections, we will be able to work assuming that the above properties hold.

2.4 Moment Estimation

We will now work towards proving our result for Poincare distributions. A key ingredient in our algorithm will be estimating the moment tensor of a mixture $\mathcal{M}$ i.e. for an unknown mixture

$$\mathcal{M} = w_1\mathcal{D}(\mu_1) + \cdots + w_k\mathcal{D}(\mu_k)$$

we would like to estimate the tensor

$$w_1\mu_1^{\otimes t} + \cdots + w_k\mu_k^{\otimes t}$$

for various values of t using samples from $\mathcal{M}$. Naturally, it suffices to consider the case where we are given samples from $\mathcal{D}(\mu)$ for some unknown μ and our goal is to estimate the tensor $\mu^{\otimes t}$. For our full algorithm, we will need to go up to $t \sim \log k/\log\log k$ but of course for such t, our estimate has to be implicit because we cannot write down the full tensor in polynomial time. In this section, we address this task by constructing an unbiased estimator with bounded variance that can be easily manipulated implicitly.

We make the following definition to simplify notation later on.

Definition 2.4.1. *For integers t and a distribution $\mathcal{D}$, we define the tensor*

$$D_{t,\mathcal{D}} = \mathop{\mathbb{E}}_{z\sim\mathcal{D}}[z^{\otimes t}]\,.$$

We will drop the subscript $\mathcal{D}$ when it is clear from context.

2.4.1 Adjusted Polynomials

First, we just construct an unbiased estimator for $\mu^{\otimes t}$ (without worrying about making it implicit). This estimator is given in the definition below.

Definition 2.4.2. *Let $\mathcal{D}$ be a distribution on $\mathbb{R}^d$. For $x \in \mathbb{R}^d$, define the polynomials $P_{t,\mathcal{D}}(x)$ for positive integers t as follows. $P_{0,\mathcal{D}}(x) = 1$ and for $t \geq 1$,*

$$P_{t,\mathcal{D}}(x) = x^{\otimes t} - \sum_{sym} D_{1,\mathcal{D}} \otimes P_{t-1,\mathcal{D}}(x) - \sum_{sym} D_{2,\mathcal{D}} \otimes P_{t-2,\mathcal{D}}(x) - \cdots - D_{t,\mathcal{D}}\,. \tag{2.7}$$

We call $P_{t,\mathcal{D}}$ the $\mathcal{D}$-adjusted polynomials and will sometimes drop the subscript $\mathcal{D}$ when it is clear from context.

We now prove that the $\mathcal{D}$-adjusted polynomials give an unbiased estimator for $\mu^{\otimes t}$ when given samples from $\mathcal{D}(\mu)$.

Claim 2.4.3. *For any $\mu \in \mathbb{R}^d$,*

$$\mathop{\mathbb{E}}_{z\sim\mathcal{D}(\mu)}[P_{t,\mathcal{D}}(z)] = \mu^{\otimes t}\,.$$

Proof. To simplify notation, we will drop all of the $\mathcal{D}$ from the subscripts as there will be no ambiguity. We prove the claim by induction on t. The base case for $t = 1$ is clear. Now

for the inductive step, note that

$$\begin{aligned}
\mathop{\mathbb{E}}_{z\sim\mathcal{D}(\mu)}[P_t(z)] &= \mathop{\mathbb{E}}_{z\sim\mathcal{D}(\mu)}\left[z^{\otimes t} - \sum_{sym} D_1 \otimes P_{t-1}(z) - \sum_{sym} D_2 \otimes P_{t-2}(z) - \cdots - D_t\right] \\
&= \mathop{\mathbb{E}}_{x\sim\mathcal{D}}[(x+\mu)^{\otimes t}] - \mathop{\mathbb{E}}_{z\sim\mathcal{D}(\mu)}\left[\sum_{sym} D_1 \otimes P_{t-1}(z) + \sum_{sym} D_2 \otimes P_{t-2}(z) + \cdots + D_t\right] \\
&= \mathop{\mathbb{E}}_{x\sim\mathcal{D}}\left[\mu^{\otimes t} - \sum_{sym}(D_1 - x^{\otimes 1}) \otimes \mu^{t-1} - \sum_{sym}(D_2 - x^{\otimes 2}) \otimes \mu^{t-2} - \cdots - (D_t - x^{\otimes t})\right] \\
&= \mu^{\otimes t}\,.
\end{aligned}$$

where we used the induction hypothesis and then the definition of D_t in the last two steps. ■

2.4.2 Variance Bounds for Poincare Distributions

In the previous section, we showed that the $\mathcal{D}$-adjusted polynomials give an unbiased estimator for $\mu^{\otimes t}$. We now show that they also have bounded variance when $\mathcal{D}$ is Poincare. This will rely on the following claim which shows that the $\mathcal{D}$-adjusted polynomials recurse under differentiation.

Claim 2.4.4. *Let $\mathcal{D}$ be a distribution on $\mathbb{R}^d$. Then*

$$\frac{\partial P_{t,\mathcal{D}}(x)}{\partial x_i} = \sum_{sym} e_i \otimes P_{t-1,\mathcal{D}}(x)$$

where we imagine $x = (x_1, \ldots, x_d)$ so x_i is the i^{th} coordinate of x and

$$e_i = (\underbrace{0, \ldots, 1}_{i}, \ldots, 0)$$

denotes the i^{th} coordinate basis vector.

Proof. We will prove this by induction on t. The base case for $t = 1$ is clear. In the proceeding computations, we drop the $\mathcal{D}$ from all subscripts as there will be no ambiguity.

Differentiating the definition of $P_{t,\mathcal{D}}$ and using the induction hypothesis, we get

$$\begin{aligned}\frac{\partial P_t(x)}{\partial x_i} &= \sum_{sym} e_i \otimes x^{\otimes t-1} - \sum_{sym} D_1 \otimes e_i \otimes P_{t-2}(x) - \cdots - \sum_{sym} D_{t-2} \otimes e_i \otimes P_1(x) - \sum_{sym} D_{t-1} \otimes e_i \\ &= \sum_{sym} e_i \otimes \left(x^{\otimes t-1} - \sum_{sym} D_1 \otimes P_{t-2}(x) - \cdots - D_{t-1} \right) \\ &= \sum_{sym} e_i \otimes P_{t-1}(x)\end{aligned}$$

where in the last step we again used (2.7), the recursive definition of P_{t-1}. This completes the proof. ■

Now, by using the Poincare property, we can prove a bound on the variance of the estimator $P_{t,\mathcal{D}}(x)$.

Claim 2.4.5. *Let $\mathcal{D}$ be a distribution on $\mathbb{R}^d$ that is 1-Poincare. Let $v \in \mathbb{R}^{d^t}$ be a vector. Then*

$$\mathop{\mathbb{E}}_{z\sim\mathcal{D}(\mu)}[(v \cdot \mathit{flatten}(P_{t,\mathcal{D}}(z)))^2] \leq (\|\mu\|^2 + t^2)^t \|v\|^2 .$$

Proof. We will prove the claim by induction on t. The base case for $t = 1$ follows because

$$\begin{aligned}\mathop{\mathbb{E}}_{z\sim\mathcal{D}(\mu)}[(v \cdot \mathsf{flatten}(P_{1,\mathcal{D}}(z)))^2] = (v \cdot \mu)^2 + \mathop{\mathsf{Var}}_{z\sim\mathcal{D}(\mu)}(v \cdot \mathsf{flatten}(P_{1,\mathcal{D}}(z))) &\leq (v \cdot \mu)^2 + \|v\|^2 \\ &\leq (\|\mu\|^2 + 1)\|v\|^2\end{aligned}$$

where we used the fact that $\mathcal{D}$ is 1-Poincare. Now for the inductive step, we have

$$\begin{aligned}
\mathop{\mathbb{E}}_{z\sim\mathcal{D}(\mu)}[(v\cdot\mathsf{flatten}(P_{t,\mathcal{D}}(z)))^2] &= (v\cdot\mathsf{flatten}(\mu^{\otimes t}))^2 + \mathop{\mathsf{Var}}_{z\sim\mathcal{D}(\mu)}(v\cdot\mathsf{flatten}(P_{t,\mathcal{D}}(z))) \\
&\le \|\mu\|^{2t}\|v\|^2 + \mathop{\mathbb{E}}_{z\sim\mathcal{D}(\mu)}\left[\sum_{i=1}^{d}\left(v\cdot\frac{\partial P_{t,\mathcal{D}}(x)}{\partial x_i}\right)^2\right] \\
&= \|\mu\|^{2t}\|v\|^2 + \mathop{\mathbb{E}}_{z\sim\mathcal{D}(\mu)}\left[\sum_{i=1}^{d}\left(\sum_{j=1}^{t} v_{\eta_j=i}\cdot P_{t-1,\mathcal{D}}(z)\right)^2\right] \\
&\le \|\mu\|^{2t}\|v\|^2 + \mathop{\mathbb{E}}_{z\sim\mathcal{D}(\mu)}\left[t\sum_{j=1}^{t}\sum_{i=1}^{d}\left(v_{\eta_j=i}\cdot P_{t-1,\mathcal{D}}(z)\right)^2\right] \\
&\le \|\mu\|^{2t}\|v\|^2 + t\sum_{j=1}^{t}\sum_{i=1}^{d}(\|\mu\|^2+(t-1)^2)^{t-1}\left\|v_{\eta_j=i}\right\|^2 \\
&= \|\mu\|^{2t}\|v\|^2 + t^2(\|\mu\|^2+(t-1)^2)^{t-1}\|v\|^2 \\
&\le (\|\mu\|^2+t^2)^t\|v\|^2
\end{aligned}$$

where in the above manipulations, we first used Claim 2.4.3, then the fact that $\mathcal{D}$ is 1-Poincare, then Claim 2.4.4, then Cauchy Schwarz, then the inductive hypothesis, and finally some direct manipulation. This completes the inductive step and we are done. ■

2.4.3 Efficient Implicit Representation

In the previous section, we showed that for $x\sim\mathcal{D}(\mu)$ for unknown μ, $P_{t,\mathcal{D}}(x)$ gives us an unbiased estimator of $\mu^{\otimes t}$ with bounded variance. Still, it is not feasible to actually compute $P_{t,\mathcal{D}}(x)$ in polynomial time because we cannot write down all of its entries and there is no nice way to implicitly work with terms such as $D_{t,\mathcal{D}}$ that appear in $P_{t,\mathcal{D}}(x)$. In this section, we construct a modified estimator that is closely related to $P_{t,\mathcal{D}}(x)$ but is also easy to work with implicitly because all of the terms will be rank-1 i.e. of the form $v_1\otimes\cdots\otimes v_t$ for some vectors $v_1,\ldots,v_t\in\mathbb{R}^d$. Throughout this section, we will assume that the distribution $\mathcal{D}$ that we are working with is fixed and we will drop it from all subscripts as there will be no ambiguity.

Roughly, the way that we construct this modified estimator is that we take multiple variables $x_1, \dots, x_t \in \mathbb{R}^d$. We start with $P_t(x_1)$. We then add various products

$$P_{a_1}(x_1) \otimes \cdots \otimes P_{a_t}(x_t)$$

to it in a way that when expanded as monomials, only the leading terms, which are rank-1 since they are a direct product of the form $x_1^{\otimes a_1} \otimes \cdots \otimes x_t^{\otimes a_t}$, remain. If we then take $x_1 \sim \mathcal{D}(\mu)$ and $x_2, \dots, x_t \sim \mathcal{D}$, then Claim 2.4.3 will immediately imply that the expectation is $\mu^{\otimes t}$. The key properties are stated formally in Corollary 2.4.10, Corollary 2.4.11 and Corollary 2.4.12.

First, we write out an explicit formula for $P_t(x)$.

Claim 2.4.6. *We have*

$$P_t(x) =$$
$$\sum_{S_0 \subseteq [t]} \left(x^{\otimes S_0}\right) \otimes \left(\sum_{\{S_1,\dots,S_t\} \in Z_t([t] \setminus S_0)} (-1)^{\mathcal{C}\{S_1,\dots,S_t\}} (\mathcal{C}\{S_1, \dots, S_t\})! (D_{|S_1|})^{(S_1)} \otimes \cdots \otimes (D_{|S_t|})^{(S_t)} \right)$$

Proof. We use induction. The base case is trivial. Now, it suffices to compute the coefficient of some "monomial" in P_t. Note that the coefficient of the monomial $x^{\otimes t}$ is clearly 1 which matches the desired formula. Otherwise, consider a monomial

$$A = \left(x^{\otimes S_0}\right) \otimes (D_{|S_1|})^{(S_1)} \otimes \cdots \otimes (D_{|S_a|})^{(S_a)}$$

where $S_1, \dots, S_a$ are nonempty for some $1 \le a \le t$. Recall the recursive definition of P_t in (2.7). There are exactly a terms on the RHS of (2.7) that can produce the monomial A. These terms are

$$-D_{|S_1|}^{(S_1)} \otimes P_{t-|S_1|}(x)^{([t]\setminus S_1)}, \dots, -D_{|S_a|}^{(S_a)} \otimes P_{t-|S_a|}(x)^{([t]\setminus S_a)}.$$

However, by the inductive hypothesis, the coefficient of A produced by each of these terms is exactly $(-1)^a(a-1)!$. Thus, combining over all a terms, the resulting coefficient is $(-1)^a a!$ which matches the desired formula. This completes the proof. ■

Now, we are ready to write out our estimator that can be written as a sum of few rank-1 terms. The key identity is below.

Definition 2.4.7. *For $x_1, \dots, x_t \in \mathbb{R}^d$, define the polynomial*

$$Q_t(x_1, \dots, x_t) = \sum_{\substack{S_1 \cup \cdots \cup S_t = [t] \\ |S_i \cap S_j| = 0}} \frac{(-1)^{\mathcal{C}\{S_1,\dots,S_t\}}}{\binom{t-1}{\mathcal{C}\{S_1,\dots,S_t\}-1}} \left(P_{|S_1|}(x_1)\right)^{(S_1)} \otimes \cdots \otimes \left(P_{|S_t|}(x_t)\right)^{(S_t)} . \tag{2.8}$$

Lemma 2.4.8. *All nonzero monomials in Q_t either have total degree t in the variables $x_1, \dots, x_t$ or are constant.*

Proof. Consider substituting the formula in Claim 2.4.6 into the RHS for all occurrences of P. Consider a monomial

$$A = \left(x_{i_1}^{\otimes U_1}\right) \otimes \cdots \otimes \left(x_{i_a}^{\otimes U_a}\right) \otimes \left(D_{|V_1|}\right)^{(V_1)} \otimes \cdots \otimes \left(D_{|V_b|}\right)^{(V_b)}$$

where $U_1, \dots, U_a, V_1, \dots, V_b$ are nonempty. Note that when we expand out the RHS, all monomials are of this form. It now suffices to compute the coefficient of this monomial A in the expansion of the RHS.

The terms in the sum on the RHS are of the form

$$\frac{(-1)^c}{\binom{t-1}{c-1}} P_{|S_{j_1}|}(x_{j_1})^{(S_{j_1})} \otimes \cdots \otimes P_{|S_{j_c}|}(x_{j_c})^{(S_{j_c})}$$

where $S_{j_1}, \dots, S_{j_c}$ are nonempty. We now consider summing over all such terms that can produce the monomial A and sum the corresponding coefficient to get the overall coefficient of A.

In order for the monomial A to appear in the expansion of this term, we need $\{i_1, \dots, i_a\} \subseteq$

$\{j_1, \ldots, j_c\}$. Once the indices $j_1, \ldots, j_c$ are fixed, it remains to assign each of the terms

$$(D_{|V_1|})^{(V_1)}, \ldots, (D_{|V_b|})^{(V_b)}$$

to one of the variables $x_{j_1}, \ldots, x_{j_c}$. This will correspond to which of the polynomials

$$P_{|S_{j_1}|}(x_{j_1})^{(S_{j_1})}, \ldots, P_{|S_{j_c}|}(x_{j_c})^{(S_{j_c})}$$

that the term came from. Specifically, for each integer f with $1 \leq f \leq c$, let $B_f \subset [b]$ be the indices of the terms that are assigned to the variable x_{j_f}. The sets $B_1, \ldots, B_c$ uniquely determine the sets $S_{j_1}, \ldots, S_{j_c}$ but also need to satisfy the constraint that if $j_f \notin \{i_1, \ldots, i_a\}$ then $B_f \neq \emptyset$. Once these sets are all fixed, by Claim 2.4.6, the desired coefficient is simply

$$\frac{(-1)^c}{\binom{t-1}{c-1}}(-1)^{|B_1|+\cdots+|B_c|}|B_1|! \cdots |B_c|! = \frac{(-1)^c}{\binom{t-1}{c-1}}(-1)^b |B_1|! \cdots |B_c|!\,.$$

Now overall, the desired coefficient is

$$\sum_{c=a}^{a+b} \frac{(-1)^c}{\binom{t-1}{c-1}} \binom{t-a}{c-a} (-1)^b \sum_{\substack{B_1 \cup \cdots \cup B_c = [b] \\ B_i \cap B_j = \emptyset \\ B_{a+1}, \ldots, B_c \neq \emptyset}} |B_1|! \cdots |B_c|!\,. \tag{2.9}$$

This is because there are $\binom{t-a}{c-a}$ to choose the set $\{j_1, \ldots, j_c\}$ (since it must contain $\{i_1, \ldots, i_a\}$). Once we have chosen this set, WLOG we can label $j_1 = i_1, \ldots, j_a = i_a$ so that $j_{a+1}, \ldots, j_c$ are the elements that are not contained in $\{i_1, \ldots, i_a\}$ and thus the constraint on $B_1, \ldots, B_c$ is simply that $B_{a+1}, \ldots, B_c$ are nonempty.

To evaluate (2.9), we will evaluate the inner sum differently. Imagine first choosing the sizes $s_1 = |B_1|, \ldots, s_c = |B_c|$ and then choosing the sets $B_1, \ldots, B_c$ to satisfy these size

constraints. The inner sum can then be rewritten as

$$\sum_{\substack{s_1+\cdots+s_c=b\\ s_{a+1},\ldots,s_c>0}} \binom{b}{s_1,\ldots,s_c} s_1!\cdots s_c! = b! \sum_{\substack{s_1+\cdots+s_c=b\\ s_{a+1},\ldots,s_c>0}} 1 = b!\binom{b+a-1}{c-1}$$

where the last equality follows from counting using stars and bars. Now we can plug back into (2.9). Assuming that $a, b \geq 1$, the coefficient of the monomial A is

$$\begin{aligned}\sum_{c=a}^{a+b} \frac{(-1)^c}{\binom{t-1}{c-1}}\binom{t-a}{c-a}(-1)^b b!\binom{b+a-1}{c-1} &= (-1)^b b! \sum_{c=a}^{a+b} \frac{(-1)^c (t-a)!(b+a-1)!}{(c-a)!(t-1)!(b+a-c)!} \\ &= (-1)^b \frac{(t-a)!(b+a-1)!}{(t-1)!} \sum_{c=a}^{a+b} (-1)^c \binom{b}{c-a} \\ &= 0\,.\end{aligned}$$

Thus, the only monomials that have nonzero coefficient either have $a = 0$ (meaning they are constant) or $b = 0$ (meaning they have degree t). This completes the proof. ■

Now, we can easily eliminate the constant term by subtracting off $Q(x_{t+1}, \ldots, x_{2t})$ for some additional variables $x_{t+1}, \ldots, x_{2t}$ and we will be left with only degree-t terms. It will be immediate that the degree-t terms are all rank-1 and this will give us an estimator that can be efficiently manipulated implicitly.

Definition 2.4.9. *For $x_1, \ldots, x_{2t} \in \mathbb{R}^d$, define the polynomial*

$$R_t(x_1, \ldots, x_{2t}) = -Q_t(x_1, \ldots, x_t) + Q_t(x_{t+1}, \ldots, x_{2t})\,.$$

Corollary 2.4.10. *We have the identity*

$$R_t(x_1, \ldots, x_{2t}) = \sum_{\substack{S_1 \cup \cdots \cup S_t = [t]\\ |S_i \cap S_j| = 0}} \frac{(-1)^{\mathcal{C}\{S_1,\ldots,S_t\}-1}}{\binom{t-1}{\mathcal{C}\{S_1,\ldots,S_t\}-1}} \left(x_1^{\otimes S_1} \otimes \cdots \otimes x_t^{\otimes S_t} - x_{t+1}^{\otimes S_1} \otimes \cdots \otimes x_{2t}^{\otimes S_t}\right)\,.$$

Proof. This follows immediately from Lemma 2.4.8 and the definition of $R_t(x_1, \ldots, x_{2t})$ be-

cause the constant terms cancel out and the degree-t terms clearly match the RHS of the desired expression. ■

Corollary 2.4.10 gives us a convenient representation for working implicitly with $R_t(x_1, \ldots, x_{2t})$. We now show why this polynomial is actually useful. In particular, we show that for $x_1 \sim \mathcal{D}(\mu)$ and $x_2, \ldots, x_{2t} \sim \mathcal{D}$, $R_t(x_1, \ldots, x_{2t})$ is an unbiased estimator of $\mu^{\otimes t}$ and furthermore that its variance is bounded. These properties will follow directly from the definitions of R_t, Q_t combined with Claim 2.4.3 and Claim 2.4.5.

Corollary 2.4.11. *We have*

$$\mathop{\mathbb{E}}_{z_1 \sim \mathcal{D}(\mu), z_2, \ldots, z_{2t} \sim \mathcal{D}} [R_t(z_1, \ldots, z_{2t})] = \mu^{\otimes t}$$

and for fixed z_1, we have

$$\mathop{\mathbb{E}}_{z_2, \ldots, z_{2t} \sim \mathcal{D}} [R_t(z_1, \ldots, z_{2t})] = P_t(z_1)\,.$$

Proof. Using the definition of Q_t in (2.8) and Claim 2.4.3, the expectations of all of the terms are 0 except for the leading term $P_t(z_1)$. Thus,

$$\mathop{\mathbb{E}}_{z_2, \ldots, z_{2t} \sim \mathcal{D}} [R_t(z_1, \ldots, z_{2t})] = P_t(z_1)\,.$$

Also by Claim 2.4.3,

$$\mathop{\mathbb{E}}_{z_1 \sim \mathcal{D}(\mu)} [P_t(z_1)] = \mu^{\otimes t}$$

and this gives us the two desired identities. ■

Corollary 2.4.12. *Let $\mathcal{D}$ be a distribution that is 1-Poincare. We have*

$$\mathop{\mathbb{E}}_{z_1 \sim \mathcal{D}(\mu), z_2, \ldots, z_{2t} \sim \mathcal{D}} \left[\mathsf{flatten}(R_t(z_1, \ldots, z_{2t}))^{\otimes 2} \right] \preceq (20t)^{2t} (\|\mu\|^{2t} + 1) I_{d^t}\,.$$

where recall I_{d^t} denotes the d^t-dimensional identity matrix.

Proof. Using the definition of R_t and Q_t and Cauchy Schwarz, we have

$$\mathop{\mathbb{E}}_{z_1\sim\mathcal{D}(\mu),z_2,\ldots,z_{2t}\sim\mathcal{D}}\left[\mathsf{flatten}(R_t(z_1,\ldots,z_{2t}))^{\otimes 2}\right]$$
$$\preceq 2\mathop{\mathbb{E}}_{z_1\sim\mathcal{D}(\mu),z_2,\ldots,z_t\sim\mathcal{D}}\left[\mathsf{flatten}\left(Q_t(z_1,\ldots,z_t)\right)^{\otimes 2}\right]+2\mathop{\mathbb{E}}_{z_{t+1},\ldots,z_{2t}\sim\mathcal{D}}\left[\mathsf{flatten}\left(Q_t(z_{t+1},\ldots,z_{2t})\right)^{\otimes 2}\right].$$

Now by Cauchy Schwarz again,

$$\mathop{\mathbb{E}}_{z_1\sim\mathcal{D}(\mu),z_2,\ldots,z_t\sim\mathcal{D}}\left[\mathsf{flatten}\left(Q_t(z_1,\ldots,z_t)\right)^{\otimes 2}\right]\preceq\left(\sum_{\substack{S_1\cup\cdots\cup S_t=[t]\\|S_i\cap S_j|=0}}|S_1|!\cdots|S_t|!\right)$$
$$\cdot\left(\sum_{\substack{S_1\cup\cdots\cup S_t=[t]\\|S_i\cap S_j|=0}}\frac{\mathbb{E}_{z_1\sim\mathcal{D}(\mu),z_2\ldots,z_t\sim\mathcal{D}}\left[\mathsf{flatten}\left(\left(P_{|S_1|}(z_1)\right)^{(S_1)}\otimes\cdots\otimes\left(P_{|S_t|}(z_t)\right)^{(S_t)}\right)^{\otimes 2}\right]}{|S_1|!\cdots|S_t|!}\right).$$

Let the first term above be C_1 and the second term be C_2. We can rearrange the sum over partitions of $[t]$ as follows. We can first choose the sizes $s_1=|S_1|,\ldots,s_t=|S_t|$ and then choose the partition according to these constraints. We get

$$C_1=\sum_{s_1+\cdots+s_t=t}\binom{t}{s_1,\ldots,s_t}s_1!\cdots s_t!=t!\binom{2t-1}{t}\leq(2t)^t$$

and similarly (and also using Claim 2.4.5)

$$\begin{aligned}
C_2 &\preceq \left(\sum_{\substack{S_1 \cup \cdots \cup S_t = [t] \\ |S_i \cap S_j| = 0}} \frac{(\|\mu\|^2 + |S_1|^2)^{|S_1|} |S_2|^{2|S_2|} \cdots |S_t|^{2|S_t|}}{|S_1|! \cdots |S_t|!} \right) I_{d^t} \\
&\preceq \left(20^t \sum_{\substack{S_1 \cup \cdots \cup S_t = [t] \\ |S_i \cap S_j| = 0}} \frac{\left(\|\mu\|^{2|S_1|} + (|S_1|!)^2 \right) (|S_2|!)^2 \cdots (|S_t|!)^2}{|S_1|! \cdots |S_t|!} \right) I_{d^t} \\
&\preceq \left(20^t (\|\mu\|^{2t} + 1) \sum_{s_1 + \cdots + s_t = t} \binom{t}{s_1, \ldots, s_t} s_1! s_2! \cdots s_t! \right) I_{d^t} \\
&\preceq (40t)^t (\|\mu\|^{2t} + 1) I_{d^t} \,.
\end{aligned}$$

Note that in the first step above, we also used the fact that $z_1, \ldots, z_t$ are drawn independently. Thus, overall we have shown

$$\mathop{\mathbb{E}}_{z_1 \sim \mathcal{D}(\mu), z_2, \ldots, z_t \sim \mathcal{D}} \left[\mathsf{flatten} \left(Q_t(z_1, \ldots, z_t) \right)^{\otimes 2} \right] \preceq (10t)^{2t} (\|\mu\|^{2t} + 1) I_{d^t} \,.$$

Similarly, we have

$$\mathop{\mathbb{E}}_{z_{t+1}, \ldots, z_{2t} \sim \mathcal{D}} \left[\mathsf{flatten} \left(Q_t(z_1, \ldots, z_t) \right)^{\otimes 2} \right] \preceq (10t)^{2t} I_{d^t}$$

and putting everything together, we conclude

$$\mathop{\mathbb{E}}_{z_1 \sim \mathcal{D}(\mu), z_2, \ldots, z_{2t} \sim \mathcal{D}} \left[\mathsf{flatten} (R_t(z_1, \ldots, z_{2t}))^{\otimes 2} \right] \preceq (20t)^{2t} (\|\mu\|^{2t} + 1) I_{d^t}$$

and we are done. ■

2.5 Iterative Projection

In this section, we explain our technique for implicitly working with tensors that have too many entries to write down. Recall that we would like to estimate the moment tensor

$$w_1\mu_1^{\otimes t} + \cdots + w_k\mu_k^{\otimes t}$$

for

$$t \sim \frac{\log(k/w_{\min})}{\log\log(k/w_{\min})}\,.$$

However doing this directly requires quasipolynomial time (because there are quasipolynomially many entries). Roughly, the way we get around this issue is by, iteratively for each t, computing a k-dimensional subspace that contains the span of $\mu_1^{\otimes t}, \ldots, \mu_k^{\otimes t}$. We then only need to compute the projection of $w_1\mu_1^{\otimes t} + \cdots + w_k\mu_k^{\otimes t}$ onto this subspace. Of course, the subspace and projection need to be computed implicitly because we cannot explicitly write out these expressions in polynomial time.

2.5.1 Nested Projection Maps

At a high level, to implicitly estimate the span of $\mu_1^{\otimes t}, \ldots, \mu_k^{\otimes t}$, we will first estimate the span of $\mu_1^{\otimes t-1}, \ldots, \mu_k^{\otimes t-1}$ and then bootstrap this estimate to estimate the span of $\mu_1^{\otimes t}, \ldots, \mu_k^{\otimes t}$. Since we cannot actually write down the span even though it is k-dimensional (because the vectors have super-polynomial length), we will store the span implicitly through a sequence of projections. We explain the details below.

Definition 2.5.1 (Nested Projection)**.** *Let $c_0 = 1$ and $c_1, \ldots, c_t$ be positive integers. Let $\Pi_1 \in \mathbb{R}^{c_1 \times dc_0}, \Pi_2 \in \mathbb{R}^{c_2 \times dc_1}, \ldots, \Pi_t \in \mathbb{R}^{c_t \times dc_{t-1}}$ be matrices. Define the $c_t \times d^t$ nested projection matrix*

$$\Gamma_{\Pi_t,\ldots,\Pi_1} = \Pi_t\left(I_d \otimes_{kr} \left(\Pi_{t-1}\left(I_d \otimes_{kr} \cdots\right)\right)\right)\,.$$

It is not hard to verify (see below) that when $\Pi_1, \ldots, \Pi_t$ are projection matrices then $\Gamma_{\Pi_t,\ldots,\Pi_1}$ is as well.

Claim 2.5.2. *Let $c_0 = 1$ and $c_1, \ldots, c_t$ be positive integers. Let $\Pi_1 \in \mathbb{R}^{c_1 \times dc_0}, \Pi_2 \in \mathbb{R}^{c_2 \times dc_1}, \ldots, \Pi_t \in \mathbb{R}^{c_t \times dc_{t-1}}$ be matrices whose rows are orthonormal. Then $\Gamma_{\Pi_t,\ldots,\Pi_1}$ has orthonormal rows.*

Proof. We prove the claim by induction on t. The base case is clear. Next, by the induction hypothesis, the matrix

$$\Gamma_{\Pi_{t-1},\ldots,\Pi_1} = \Pi_{t-1}\left(I_d \otimes_{\mathsf{kr}} \left(\Pi_{t-2}\left(I_d \otimes_{\mathsf{kr}} \cdots\right)\right)\right)$$

has orthonormal rows. Thus, the matrix

$$\Pi_t\left(I_d \otimes_{\mathsf{kr}} \Gamma_{\Pi_{t-1},\ldots,\Pi_1}\right)$$

has orthonormal rows as well, completing the induction. ■

Note that in our paper, $\Pi_1, \ldots, \Pi_t$ will always have orthonormal rows so $\Gamma_{\Pi_t,\ldots,\Pi_1}$ always does as well. This fact will often be used without explicitly stating it. The key point about the construction of $\Gamma_{\Pi_t,\ldots,\Pi_1}$ is that instead of storing a full $c_t \times d^t$-sized matrix, it suffices to store the individual matrices $\Pi_1, \ldots, \Pi_t$ which are all polynomially sized. The next important observation is that for certain vectors $v \in \mathbb{R}^{d^t}$ that are "rank-1" i.e. those that can be written in the form

$$v = \mathsf{flatten}(v_t \otimes \cdots \otimes v_1)\,,$$

the expression $\Gamma_{\Pi_t,\ldots,\Pi_1} v$ can be computed efficiently. This is shown in the following claim.

Claim 2.5.3. *Let $c_0 = 1$ and $c_1, \ldots, c_t$ be positive integers. Let $\Pi_1 \in \mathbb{R}^{c_1 \times dc_0}, \Pi_2 \in \mathbb{R}^{c_2 \times dc_1}, \ldots, \Pi_t \in \mathbb{R}^{c_t \times dc_{t-1}}$ be matrices. Let $v \in \mathbb{R}^{d^t}$ satisfy $v = \mathsf{flatten}(v_1 \otimes \cdots \otimes v_t)$ for some $v_1, \ldots, v_t \in \mathbb{R}^d$. Then in $\mathrm{poly}(d, t, \max(c_i))$ time, we can compute $\Gamma_{\Pi_t,\ldots,\Pi_1} v$.*

Proof. We will prove the claim by induction on t. For each $t' = 1, 2, \ldots, t$, we compute

$$\Gamma_{\Pi_{t'},\ldots,\Pi_1}\mathsf{flatten}(v_{t'} \otimes \cdots \otimes v_1)\,.$$

The base case of the induction is clear. To do the induction step, note that

$$\Gamma_{\Pi_{t'+1},\dots,\Pi_1}\mathsf{flatten}(v_{t'+1}\otimes\cdots\otimes v_1) = \Pi_{t'+1}\mathsf{flatten}\left(v_{t'+1}\otimes\left(\Gamma_{\Pi_{t'},\dots,\Pi_1}\mathsf{flat}(v_{t'}\otimes\cdots\otimes v_1)\right)\right).$$

It is clear that this computation can be done in $\mathrm{poly}(d,t,\max(c_i))$ time so iterating this operation completes the proof. ■

As a trivial consequence of the above, we can also compute direct products of nested projections applied to a "rank-1" vector v.

Corollary 2.5.4. *Let $c_0 = 1$ and $c_1,\dots,c_t$ be positive integers. Let $\Pi_1 \in \mathbb{R}^{c_1\times dc_0}, \Pi_2 \in \mathbb{R}^{c_2\times dc_1},\dots,\Pi_t \in \mathbb{R}^{c_t\times dc_{t-1}}$ be matrices. Let $v\in\mathbb{R}^{d^t}$ satisfy $v = \mathsf{flatten}(v_1\otimes\cdots\otimes v_t)$ for some $v_1,\dots,v_t\in\mathbb{R}^d$. Then for any integers $1<s_1<s_2<\cdots<s_n<t$, in $\mathrm{poly}(d,t,\max(c_i))$ time, we can compute the expression*

$$\left(\Gamma_{\Pi_t,\dots,\Pi_{s_n+1}}\otimes_{kr}\cdots\otimes_{kr}\Gamma_{\Pi_{s_1},\dots,\Pi_1}\right)v\,.$$

Throughout our paper, we will only compute nested projections of the above form so it will be easy to verify that all steps can be implemented in polynomial time.

2.6 Implicitly Estimating the Moment Tensor

In this section, we combine the iterative projection techniques from Section 2.5 with the estimators from Section 2.4 to show how to implicitly estimate the moment tensor given sample access to the distribution $\mathcal{D}$ and the mixture

$$\mathcal{M} = w_1\mathcal{D}(\mu_1)+\cdots+w_k\mathcal{D}(\mu_k)\,.$$

By implicitly estimate, we mean that we will compute projection matrices $\Pi_t,\dots,\Pi_1$ such that the row-span of $\Gamma_{\Pi_t,\dots,\Pi_1}$ essentially contains all of the flattenings of $\mu_1^{\otimes t},\dots,\mu_k^{\otimes t}$.

Remark. *Once we have these matrices, we will also be able to estimate expressions such as*

$$\Gamma_{\Pi_t,\ldots,\Pi_1} \mathsf{flatten}\left(w_1\mu_1^{\otimes t} + \cdots + w_k\mu_k^{\otimes t}\right) .$$

It will be convenient to make the following definitions.

Definition 2.6.1. *For a mixture $\mathcal{M} = w_1\mathcal{D}(\mu_1) + \cdots + w_k\mathcal{D}(\mu_k)$, we use $T_{t,\mathcal{M}}$ to denote the tensor $w_1\mu_1^{\otimes t} + \cdots + w_k\mu_k^{\otimes t}$. We may drop the subscript $\mathcal{M}$ and just write T_t when it is clear from context.*

Definition 2.6.2. *For a mixture $\mathcal{M} = w_1\mathcal{D}(\mu_1) + \cdots + w_k\mathcal{D}(\mu_k)$, we define $M_{2s,\mathcal{M}}$ to be the tensor $T_{2s,\mathcal{M}}$ rearranged (in a canonical way) as a $d^s \times d^s$ square matrix. Again, we may drop the subscript $\mathcal{M}$ when it is clear from context.*

We define $\mu_{\max} = \max(1, \|\mu_1\|, \ldots, \|\mu_k\|)$. We do not assume that we know $\mu_{\max}$ in advance. However, the reduction in Section 2.3.3 means that it suffices to consider when $\mu_{\max}$ is polynomially bounded. Also we can assume $d = k$ i.e. the dimension of the underlying space is equal to the number of components. This is because we can use the reduction in Section 2.3.3 and if $d < k$, then we can simply add independent standard Gaussian entries in the remaining $k - d$ dimensions.

We now describe our algorithm for implicitly estimating the moment tensor. For the remainder of this section, we will only work with a fixed mixture $\mathcal{M}$ so we will drop it from all subscripts e.g. in Definitions 2.6.1 and 2.6.2. At a high level, we will recursively compute a sequence of projection matrices $\Pi_1 \in \mathbb{R}^{k\times d}, \Pi_2, \ldots, \Pi_s \in \mathbb{R}^{k\times dk}$. Our goal will be to ensure that $\Gamma_{\Pi_s,\ldots,\Pi_1}$ (which is a $k \times d^s$ matrix) essentially contains the flattenings of $\mu_1^{\otimes s}, \ldots, \mu_k^{\otimes s}$ in its row span.

To see how to do this, assume that we have computed $\Pi_{s-1}, \ldots, \Pi_1$ so far. By the inductive hypothesis, $\Gamma_{\Pi_{s-1},\ldots,\Pi_1}$ tells us a k-dimensional subspace that essentially contains the flattenings of $\mu_1^{\otimes s-1}, \ldots, \mu_k^{\otimes s-1}$. Thus, we trivially have a dk dimensional subspace, given by the rows of $\left(I_d \otimes_{\mathsf{kr}} \Gamma_{\Pi_{s-1},\ldots,\Pi_1}\right)$ that must essentially contain all of the flattenings of

$\mu_1^{\otimes s}, \ldots, \mu_k^{\otimes s}$. It remains to reduce from this dk-dimensional space back to a k-dimensional space. However, we can now write everything out in this dk-dimensional space and simply run PCA and take the top-k singular subspace. Formally, we estimate the $dk \times dk$ matrix

$$A_{2s} = \left(I_d \otimes_{\mathsf{kr}} \Gamma_{\Pi_{s-1},\ldots,\Pi_1}\right) M_{2s} \left(I_d \otimes_{\mathsf{kr}} \Gamma_{\Pi_{s-1},\ldots,\Pi_1}\right)^T = \sum_{i=1}^{k} w_i \left(\left(I_d \otimes_{\mathsf{kr}} \Gamma_{\Pi_{s-1},\ldots,\Pi_1}\right) \mathsf{flatten}(\mu_i)^{\otimes s}\right)^{\otimes 2}$$

using techniques from Section 2.4 and then simply set Π_s to have rows given by the top k singular vectors of A_{2s}. To gain some intuition for why this works, imagine that the subspace spanned by the rows of $\Gamma_{\Pi_{s-1},\ldots,\Pi_1}$ exactly contains the flattenings of $\mu_1^{\otimes s-1}, \ldots, \mu_k^{\otimes s-1}$. Also assume that our estimate of A_{2s} is exact. Then A_{2s} has rank at most k and the top-k singular subspace must contain all of the vectors

$$\left(I_d \otimes_{\mathsf{kr}} \Gamma_{\Pi_{s-1},\ldots,\Pi_1}\right) \mathsf{flatten}(\mu_i)^{\otimes s} = \mathsf{flatten}\left(\mu_i \otimes \Gamma_{\Pi_{s-1},\ldots,\Pi_1} \mathsf{flatten}(\mu_i)^{\otimes s-1}\right) .$$

The above then immediately implies that the subspace spanned by the rows of $\Gamma_{\Pi_s,\ldots,\Pi_1}$ exactly contains the flattenings of $\mu_1^{\otimes s}, \ldots, \mu_k^{\otimes s}$. Of course, the actual analysis will need to be much more precise quantitatively in tracking the errors in each step.

Our algorithm is described in full below. The main algorithm, Algorithm 1, computes the projection matrices $\Pi_1, \ldots \Pi_s$ following the outline above. As a subroutine, it needs to estimate the matrix A_{2s}. This is done in Algorithm 2 which relies on the results in Section 2.4, namely Corollary 2.4.11 and Corollary 2.4.12.

Algorithm 1 ITERATIVE PROJECTION STEP

Input: Samples $z_1, \ldots, z_n$ from unknown mixture

$$\mathcal{M} = w_1\mathcal{D}(\mu_1) + \cdots + w_k\mathcal{D}(\mu_k)$$

Input: integer $t > 0$

Split samples into t sets $S_1, \ldots, S_t$ of equal size

Let $\Pi_1 = I_d$ (recall $k = d$)

for s = 2, ..., t **do**

Run ESTIMATE MOMENT TENSOR using samples S_s to get approximation $\widetilde{A_{2s}} \in \mathbb{R}^{dk\times dk}$ to

$$A_{2s} = \left(I_d \otimes_{\mathsf{kr}} \Gamma_{\Pi_{s-1},\ldots,\Pi_1}\right) M_{2s} \left(I_d \otimes_{\mathsf{kr}} \Gamma_{\Pi_{s-1},\ldots,\Pi_1}\right)^T$$

Let $\Pi_s \in \mathbb{R}^{k\times dk}$ have rows forming an orthonormal basis of the top k singular subspace of $\widetilde{A_{2s}}$

Output: $(\Pi_t, \ldots, \Pi_1)$

Algorithm 2 ESTIMATE MOMENT TENSOR

Input: Samples $z_1, \ldots, z_n$ from unknown mixture

$$\mathcal{M} = w_1\mathcal{D}(\mu_1) + \cdots + w_k\mathcal{D}(\mu_k)$$

Input: Integer $s > 0$

Input: Matrices $\Pi_{s-1} \in \mathbb{R}^{k\times dk}, \ldots, \Pi_2 \in \mathbb{R}^{k\times dk}, \Pi_1 \in \mathbb{R}^{k\times d}$

for i = 1,2, ..., n **do**

Independently draw samples $x_1, \ldots, x_{4s-1}$ from $\mathcal{D}$

Compute the $(kd)^2$-dimensional vector (recall Definition 2.4.9)

$$X_i = \left(\left(I_d \otimes_{\mathrm{kr}} \Gamma_{\Pi_{s-1},\ldots,\Pi_1}\right) \otimes_{\mathrm{kr}} \left(I_d \otimes_{\mathrm{kr}} \Gamma_{\Pi_{s-1},\ldots,\Pi_1}\right)\right)\mathsf{flatten}\left(R_{2s}(z_i, x_1, \ldots, x_{4s-1})\right) .$$

Let K_i be the rearrangement of X_i into a square $dk \times dk$-dimensional matrix

Output: $A = (K_1 + \cdots + K_n)/n$

2.6.1 Efficient Implementation

A naive implementation of Algorithm 1 requires d^t time, which is too large. However, in this section, we show that we can implement all of the steps more efficiently using only $\text{poly}(ndk, t^t)$ time.

Remark. *We will later show that it suffices to consider*

$$t \sim O\left(\frac{\log(k/w_{\min})}{\log\log(k/w_{\min})}\right)$$

so this runtime is actually polynomial in all parameters that we need.

Claim 2.6.3. *Algorithm 1 can be implemented to run in* $\text{poly}(n, d, k, t^t)$ *time.*

Proof. First we analyze the runtime of Algorithm 2. Note that by Corollary 2.4.10, we can write

$$R_s(z_i, x_1, \ldots, x_{4s-1}) = V_1 + \cdots + V_l$$

where $V_1, \ldots, V_l$ are all rank-1 tensors and $l = s^{O(s)}$. Now by Corollary 2.5.4, this means that computing

$$\begin{aligned} X_i &= ((I_d \otimes_{\text{kr}} \Gamma_{\Pi_{s-1},\ldots,\Pi_1}) \otimes_{\text{kr}} (I_d \otimes_{\text{kr}} \Gamma_{\Pi_{s-1},\ldots,\Pi_1}))\mathsf{flatten}\,(Q_{2s}(z_i, x_1, \ldots, x_{4s-1})) \\ &= ((I_d \otimes_{\text{kr}} \Gamma_{\Pi_{s-1},\ldots,\Pi_1}) \otimes_{\text{kr}} (I_d \otimes_{\text{kr}} \Gamma_{\Pi_{s-1},\ldots,\Pi_1}))(\mathsf{flatten}(V_1) + \cdots + \mathsf{flatten}(V_l)) \end{aligned}$$

can be done in $\text{poly}(d, k, s^s)$ time by expanding out the RHS and computing each term separately. It is then immediate that all of Algorithm 2 runs in $\text{poly}(n, d, k, s^s)$ time. Now in Algorithm 1, the only additional steps involve computing an SVD of the matrices $\widetilde{A_{2s}}$ (which are polynomially sized) so we conclude that the entire algorithm runs in $\text{poly}(n, d, k, t^t)$ time. ■

2.6.2 Accuracy Analysis

Now, we analyze the correctness of Algorithm 1 namely that the span of the rows of the matrix $\Gamma_{\Pi_t,\ldots,\Pi_1}$ indeed essentially contain all of $\mu_1^{\otimes t}, \ldots, \mu_k^{\otimes t}$. To simplify notation, we make the following definition.

Definition 2.6.4. *For all s, we define the matrix*

$$A_{2s} = \left(I_d \otimes_{kr} \Gamma_{\Pi_{s-1},\ldots,\Pi_1}\right) M_{2s} \left(I_d \otimes_{kr} \Gamma_{\Pi_{s-1},\ldots,\Pi_1}\right)^T .$$

Remark. *Note that in the execution of Algorithm 1, $\widetilde{A_{2s}}$ is intended to be an estimate of A_{2s}.*

The main result that we will prove is stated below. Note that this lemma does not require any assumptions about minimum mixing weights or means in the mixture. Instead, it simply says that the subspace spanned by the rows of $\Gamma_{\Pi_s,\ldots,\Pi_1}$ essentially contains $\mathsf{flatten}(\mu_i^{\otimes s})$ for all components $\mathcal{D}(\mu_i)$ with mean and mixing weight bounded away from 0.

Lemma 2.6.5. *Let $\mathcal{D}$ be a distribution on $\mathbb{R}^d$ that is 1-Poincare and let $\mathcal{M} = w_1\mathcal{D}(\mu_1) + \cdots + w_k\mathcal{D}(\mu_k)$ be a mixture of translations of $\mathcal{D}$. Let $w^*, \epsilon > 0$ be parameters. Assume that the number of samples satisfies*

$$n \geq \left(\frac{t^t \mu_{\max}^t kd}{w^*\epsilon}\right)^C$$

for some sufficiently large universal constant C. Then with probability at least $1 - n^{-0.2}$, in the execution of Algorithm 1, the following condition holds: for all $i \in [k]$ such that $\|\mu_i\| \geq 1$ and $w_i \geq w^*$, we have

$$\left\|\Gamma_{\Pi_s,\ldots,\Pi_1} \mathsf{flatten}(\mu_i^{\otimes s})\right\| \geq (1 - s\epsilon) \|\mu_i\|^s$$

for all $s = 1, 2, \ldots, t$.

Remark. *The parameter ϵ represents the desired accuracy and the parameter w^* is a weight cutoff threshold where we guarantee to recover "significant" components whose mixing weight is at least w^*.*

Roughly, the proof of Lemma 2.6.5 will involve following the outline at the beginning of this section but quantitatively tracking the errors more precisely. Before we prove Lemma 2.6.5, we first prove a preliminary claim that our estimation error $\left\|A_{2s} - \widetilde{A_{2s}}\right\|_F$ is small.

Claim 2.6.6. *Assume that for a fixed integer s, Algorithm 2 is run with a number of samples*

$$n \geq \left(\frac{s^s \mu_{\max}^s kd}{w^* \epsilon}\right)^C$$

for some sufficiently large universal constant C. Then with probability at least $1 - n^{-0.4}$, its output $\widetilde{A_{2s}}$ satisfies

$$\left\|\widetilde{A_{2s}} - A_{2s}\right\|_F \leq 0.5 w^* \epsilon^2 .$$

Proof. Recall that we are trying to estimate A_{2s} which is a $dk \times dk$ matrix. It will suffice for us to obtain a concentration bound for our estimate of each entry and then union bound. Recall that in Algorithm 2, we estimate A_{2s} by averaging $K_1, \ldots, K_n$. By Corollary 2.4.11, we have that

$$\mathbb{E}[K_i] = A_{2s} \tag{2.10}$$

so our estimator is unbiased. Next, observe that each entry of K_i, say $K_i[a, b]$ where $1 \leq a, b \leq dk$ can be written as

$$K_i[a, b] = v \cdot \mathsf{flatten}(R_{2s}(z_i, x_1, \ldots, x_{4s-1}))$$

where $v \in \mathbb{R}^{d^{2s}}$ is some unit vector (this is by Claim 2.5.2). By Corollary 2.4.12, we have

$$\mathbb{E}\left[(K_i[a, b] - A_{2s}[a, b])^2\right] \leq \mathbb{E}[K_i[a, b]^2] \leq (20s)^{2s}(\mu_{\max}^{2s} + 1)$$

where the first inequality above is true by (2.10). Since our final estimate is obtained by averaging over n independent samples, we have

$$\mathbb{E}\left[(\widetilde{A_{2s}}[a, b] - A_{2s}[a, b])^2\right] \leq \frac{(20s)^{2s}(\mu_{\max}^{2s} + 1)}{n} .$$

Thus, with probability at least $1 - n^{-0.5}$, we must have

$$\left|\widetilde{A_{2s}}[a,b] - A_{2s}[a,b]\right| \leq \frac{(20s)^{2s}(\mu_{\max}^{2s}+1)}{\sqrt{n}} \leq \frac{0.5w^*\epsilon^2}{dk}$$

where the last inequality holds as long as we choose n sufficiently large. Union bounding the above over all entries (there are only $(dk)^2$ entries to union bound over) and ensuring that n is sufficiently large, we get the desired bound. ■

Now we are ready to prove Lemma 2.6.5.

Proof of Lemma 2.6.5. We will prove the claim by induction on s. The base case for $s = 0$ is clear. Now let $i \in [k]$ be such that $\|\mu_i\| \geq 1$ and $w_i \geq w^*$. Define the vector $v_{i,s} \in \mathbb{R}^{dk}$ as

$$v_{i,s} = \mathsf{flatten}\left(\mu_i \otimes \Gamma_{\Pi_{s-1},\ldots,\Pi_1}\mathsf{flatten}(\mu_i^{s-1})\right) .$$

Note that this allows us to rewrite the matrix A_{2s} as

$$A_{2s} = w_1(v_{1,s} \otimes v_{1,s}) + \cdots + w_k(v_{k,s} \otimes v_{k,s}) .$$

Let $u_{i,s}$ be the projection of $v_{i,s}$ onto the orthogonal complement of Π_s. Note that

$$u_{i,s} \cdot v_{i,s} = \|u_{i,s}\|^2 .$$

Thus, we must have

$$u_{i,s}^T A_{2s} u_{i,s} \geq w_i \|u_{i,s}\|^4 .$$

On the other hand, note that A_{2s} has rank at most k. Assuming that the hypothesis of Claim 2.6.6 holds, the $k+1$st singular value of $\widetilde{A_{2s}}$ has size at most $w^*\epsilon^2$. Thus,

$$u_{i,s}^T \widetilde{A_{2s}} u_{i,s} \leq 0.5w^*\epsilon^2 \|u_{i,s}\|^2 .$$

Finally, using the hypothesis of Claim 2.6.6 again, we must have

$$\left|u_{i,s}^T(A_{2s} - \widetilde{A_{2s}})u_{i,s}\right| \leq 0.5w^*\epsilon^2 \left\|u_{i,s}\right\|^2 .$$

Putting the previous three inequalities together, we deduce that we must have

$$w_i \left\|u_{i,s}\right\|^4 \leq w^*\epsilon^2 \left\|u_{i,s}\right\|^2$$

which implies $\|u_{i,s}\| \leq \epsilon$. Also, the induction hypothesis implies that

$$\left\|\Gamma_{\Pi_{s-1},\ldots,\Pi_1}\mathsf{flatten}(\mu_i^{s-1})\right\| \geq (1-(s-1)\epsilon)\left\|\mu_i\right\|^{s-1}$$

and thus

$$\|v_{i,s}\| \geq (1-(s-1)\epsilon)\left\|\mu_i\right\|^s .$$

Finally, note that

$$\|\Gamma_{\Pi_s,\ldots,\Pi_1}\mathsf{flatten}(\mu_i^s)\| = \|v_{i,s} - u_{i,s}\| \geq (1-(s-1)\epsilon)\left\|\mu_i\right\|^s - \epsilon \geq (1-s\epsilon)\left\|\mu_i\right\|^s .$$

This completes the inductive step. Finally, it remains to note that the overall failure probability can be bounded by union bounding over all applications of Claim 2.6.6 and is clearly at most $n^{-0.2}$ as long as we choose n sufficiently large. This completes the proof. ■

2.7 Testing Samples Using Implicit Moments

Now we show how to use the projection maps $\Pi_t, \ldots, \Pi_1$ computed by Algorithm 1 to test whether a sample came from a component with mean close to 0 or mean far away from 0. Roughly, given a sample z, the test simply works by computing $R_t(z, z_1, \ldots, z_{2t-1})$ for $z_1, \ldots, z_{2t-1} \sim \mathcal{D}$ and computing

$$\|\Gamma_{\Pi_t,\ldots,\Pi_1}\mathsf{flatten}(R_t(z, z_1, \ldots, z_{2t-1}))\| .$$

We output FAR if the above is larger than some threshold and otherwise we output CLOSE. For technical reasons, we will actually average over multiple independent draws for $z_1, \ldots, z_{2t-1} \sim \mathcal{D}$.

Roughly, the intuition for why this test works is as follows. Note that if $z \sim \mathcal{D}$ then by Corollary 2.4.11,

$$\mathbb{E}\left[\Gamma_{\Pi_t,\ldots,\Pi_1}\mathsf{flatten}(R_t(z, z_1, \ldots, z_{2t-1}))\right] = 0$$

and if we control the variance using Corollary 2.4.12, then we can upper bound the length with reasonable probability. On the other hand if $z \sim \mathcal{D}(\mu_i)$ for some μ_i with large norm, then

$$\mathbb{E}\left[\Gamma_{\Pi_t,\ldots,\Pi_1}\mathsf{flatten}(R_t(z, z_1, \ldots, z_{2t-1}))\right] = \Gamma_{\Pi_t,\ldots,\Pi_1}\mathsf{flatten}(\mu_i^{\otimes t})$$

and since the algorithm in the previous section can ensure that $\mu_i^{\otimes t}$ is essentially contained in the row span of $\Gamma_{\Pi_t,\ldots,\Pi_1}$, the RHS above has large norm. The details of our algorithm for testing samples are described below.

Algorithm 3 TEST SAMPLES

Input: Projection matrices $\Pi_t, \ldots, \Pi_2 \in \mathbb{R}^{k \times dk}, \Pi_1 \in \mathbb{R}^{k \times d}$

Input: Sample $z \in \mathbb{R}^d$ to test

Input: Threshold τ, desired accuracy δ

Set $n = ((10^3 t)^t/\delta)^3$

for $i = 1, 2, \ldots, n$ **do**

Draw samples $z_1, \ldots, z_{2t-1} \sim \mathcal{D}$

Let $A_i = \Gamma_{\Pi_t,\ldots,\Pi_1}\mathsf{flatten}(R_t(z, z_1, \ldots, z_{2t-1}))$

Set $A = (A_1 + \cdots + A_n)/n$

if $\|A\| \geq \tau$ **then**

Output: FAR

else

Output: CLOSE

2.7.1 Analysis of TEST SAMPLES

Now we analyze the behavior of Algorithm 3. The key properties that the test satisfies are summarized in the following two lemmas. Lemma 2.7.2 say that with $1-\delta$ probability the test will successfully output FAR for samples from a component with mean far from 0 and Lemma 2.7.1 says that with $1-\delta$ probability, the test will successfully output CLOSE for samples from a component with mean 0.

Note that Lemma 2.7.2 requires that the row span of $\Gamma_{\Pi_t,\dots,\Pi_1}$ essentially contains $\mathsf{flatten}(\mu_i^{\otimes t})$ (which can be guaranteed by Algorithm 1 and Lemma 2.6.5). Lemma 2.7.1 actually does not require anything about $\Pi_t,\dots,\Pi_1$ (other than the fact that they are actually projections).

Lemma 2.7.1. *Let $\mathcal{D}$ be a distribution that is 1-Poincare. Let $t \in \mathbb{N}$ and $0 < \delta < 0.01$ be some parameters. Let $\Pi_t,\dots,\Pi_2 \in \mathbb{R}^{k\times dk}, \Pi_1 \in \mathbb{R}^{k\times d}$ be any matrices whose rows are orthonormal. Let τ be some parameter satisfying $\tau \geq (20t)^t k/\delta$. Let $z \sim \mathcal{D}$. Then with probability at least $1-\delta$, Algorithm 3 run with these parameters outputs* CLOSE *where the randomness is over z and the random choices within Algorithm 3.*

Lemma 2.7.2. *Let $\mathcal{D}$ be a distribution that is 1-Poincare. Let $t \in \mathbb{N}$ and $0 < \delta < 0.01$ be some parameters. Let $z \sim \mathcal{D}(\mu_i)$ where $\|\mu_i\| \geq 10^4(\log 1/\delta + t)$. Let τ be some parameter satisfying $\tau \leq (0.4\|\mu_i\|)^t$. Assume that the matrices $\Pi_t,\dots,\Pi_2 \in \mathbb{R}^{k\times dk}, \Pi_1 \in \mathbb{R}^{k\times d}$ satisfy that*

$$\left\|\Gamma_{\Pi_t,\dots,\Pi_1}\mathsf{flatten}(\mu_i^{\otimes t})\right\| \geq (1-t\epsilon)\|\mu_i\|^t \ .$$

where

$$\epsilon < \frac{\delta}{(10t\|\mu_i\|)^{4t}} \ .$$

Then with probability at least $1-\delta$, Algorithm 3 run with these parameters outputs FAR *(where the randomness is over z and the random choices within Algorithm 3).*

Remark. *Note that we will want to run the test with some inverse polynomial failure probability i.e. $\delta = \mathrm{poly}(k/w^*)$ for some weight threshold w^*. In order to be able to combine*

Lemma 2.7.1 and Lemma 2.7.2 meaningfully, we need

$$(0.4\,\|\mu_i\|)^t \geq (20t)^t k/\delta\,.$$

If $\|\mu_i\| \geq (\log(k/w^*))^{1+c}$ *for some constant* $c > 0$ *then setting* $t \sim O\left(c^{-1}\log(k/w^*)/\log\log(k/w^*)\right)$ *ensures that the above inequality is true. Note that for this setting,* $t^t = \mathrm{poly}(k/w^*)$ *(where we treat* c *as a constant) and thus we will be able to ensure that our overall runtime is polynomial.*

We will first prove Lemma 2.7.1 (which is much easier to prove than Lemma 2.7.2). In fact, a direct variance bound using Corollary 2.4.12 will suffice.

Proof of Lemma 2.7.1. Note that the matrix $\Gamma_{\Pi_t,\ldots,\Pi_1}$ has (up to) k orthonormal rows, say $v_1,\ldots,v_k$. Let R be the average of $R_t(z,z_1,\ldots,z_{2t-1})$ over $n = ((10^3t)^t/\delta)^3$ trials where z is drawn once and $z_1,\ldots,z_{2t-1}$ are sampled independently in each trial. By Cauchy Schwarz, for any vector v,

$$\mathbb{E}[(v\cdot\mathsf{flatten}(R))^2] \leq \mathop{\mathbb{E}}_{z\sim\mathcal{D}(\mu_i),z_1,\ldots,z_{2t-1}\sim\mathcal{D}}[(v\cdot\mathsf{flatten}(R_t(z,z_1,\ldots,z_{2t-1})))^2]\,.$$

In other words, the covariance matrix of R is smaller than the covariance of $R_t(z,z_1,\ldots,z_{2t-1})$ (in the semidefinite ordering). Now fix $i\in[k]$. By Corollary 2.4.12 and Markov's inequality, we have with probability at least $1-\delta/k$

$$|v_i\cdot\mathsf{flatten}(R)| \leq \sqrt{\frac{k}{\delta}}(20t)^t\,.$$

Union bounding over all i, with probability at least $1-\delta$, we have

$$\|\Gamma_{\Pi_t,\ldots,\Pi_1}\mathsf{flatten}(R)\| \leq \frac{k}{\delta}(20t)^t\,.$$

Note that the expression $\Gamma_{\Pi_t,\ldots,\Pi_1}\mathsf{flatten}(R)$ is exactly equivalent to the vector A that is tested by Algorithm 3 so with probability at least $1-\delta$, the final output is Close. ■

A direct variance bound will not work for Lemma 2.7.2. This is because we want the norm to be large with $1-\delta$ probability but the variance is comparable to the squared length of the mean so we cannot get strong enough concentration with just a variance bound. Instead, we need a more precise argument. We will first, in the next claim, prove a bound on evaluations of the polynomial $P_{t,\mathcal{D}}$ (recall Definition 2.4.2). Essentially, we will argue that under the conditions of Lemma 2.7.2, for $z \sim \mathcal{D}(\mu_i)$, with high probability, the tensor $P_{t,\mathcal{D}}(z)$ has large inner product with $\mu_i^{\otimes t}$. Since $\mathsf{flatten}(\mu_i^{\otimes t})$ is essentially contained in the row span of $\Gamma_{\Pi_t,\dots,\Pi_1}$, this implies that

$$\Gamma_{\Pi_t,\dots,\Pi_1}\mathsf{flatten}(P_{t,\mathcal{D}}(z))$$

must have large norm. By Corollary 2.4.11,

$$\mathop{\mathbb{E}}_{z_1,\dots,z_{2t-1}\sim\mathcal{D}}[R_t(z,z_1,\dots,z_{2t-1})] = P_{t,\mathcal{D}}(z)$$

so if we average over enough independent samples for $z_1,\dots,z_{2t-1}$ then the estimator A computed in Algorithm 3 will concentrate around its mean and also have large norm which is exactly what we want. From now on, we will drop the subscript $\mathcal{D}$ as the adjusted polynomial will always be defined with respect to $\mathcal{D}$.

Claim 2.7.3. *Let $x \in \mathbb{R}^d$. Let $v \in \mathbb{R}^d$ be a vector such that $v \cdot x \geq 200t\,\|v\|$. Then*

$$\langle P_t(x), v^{\otimes t}\rangle \geq (0.9v\cdot x)^t\,.$$

Proof. WLOG we may assume v is a unit vector. Let $a = v \cdot x$. We will use Claim 2.4.6 to rewrite $P_t(x)$. Note that for any integer s, $\langle D_s, v^{\otimes s}\rangle = \mathbb{E}_{z\sim\mathcal{D}}[(z\cdot v)^s]$. Combining with Fact 2.3.7, we have

$$|\langle D_s, v^{\otimes s}\rangle| \leq 6\cdot s!\,.$$

Now by Claim 2.4.6, we have

$$
\begin{aligned}
&\langle P_t(x), v^{\otimes t}\rangle \\
&= \left\langle v^{\otimes t}, \sum_{S_0\subset[t]} \left(x^{\otimes S_0}\right)\otimes \right. \\
&\qquad \left. \left(\sum_{\{S_1,\ldots,S_t\}\in Z_t([t]\setminus S_0)} (-1)^{\mathcal{C}\{S_1,\ldots,S_t\}}(\mathcal{C}\{S_1,\ldots,S_t\})!(D_{|S_1|})^{(S_1)}\otimes\cdots\otimes(D_{|S_t|})^{(S_t)}\right)\right\rangle \\
&\geq a^t - \sum_{S_0\subset[t],S_0\neq[t]} a^{|S_0|}\left(\sum_{\{S_1,\ldots,S_t\}\in Z_t([t]\setminus S_0)} 6^{\mathcal{C}\{S_1,\ldots,S_t\}}(\mathcal{C}\{S_1,\ldots,S_t\})!|S_1|!\cdots|S_t|!\right) \\
&\geq a^t - \sum_{S_0\subset[t],S_0\neq[t]} a^{|S_0|}6^{t-|S_0|}\left(\sum_{c=1}^{t-|S_0|}\sum_{\substack{S_1\cup\cdots\cup S_c=[t]\setminus S_0\\ S_i\cap S_j=\emptyset, S_i\neq\emptyset}} |S_1|!\cdots|S_c|!\right) \\
&= a^t - \sum_{S_0\subset[t],S_0\neq[t]} a^{|S_0|}6^{t-|S_0|}\left(\sum_{c=1}^{t-|S_0|}\sum_{\substack{s_1+\cdots+s_c=t-|S_0|\\ s_i>0}} (t-|S_0|)!\right) \\
&\geq a^t - \sum_{S_0\subset[t],S_0\neq[t]} a^{|S_0|}6^{t-|S_0|}(t-|S_0|)!2^{t-|S_0|} \\
&\geq a^t - \sum_{c=1}^{t}\binom{t}{c}c!12^c a^{t-c} \geq a^t - \sum_{c=1}^{t}(12t)^c a^{t-c} = a^t\left(1-\sum_{c=1}^{t}\left(\frac{12t}{a}\right)^c\right) \geq 0.9a^t\,.
\end{aligned}
$$

This completes the proof. ■

Now we are ready to prove Lemma 2.7.2.

Proof of Lemma 2.7.2. Let $v = \mu_i/\left\|\mu_i\right\|$. By Fact 2.3.7, with probability at least $1-\delta/10$, we have

$$v\cdot z \geq 0.9\left\|\mu_i\right\|\,. \tag{2.11}$$

Now by Claim 2.7.3, we have

$$\langle P_t(z), v^{\otimes t}\rangle \geq \left(0.8\left\|\mu_i\right\|\right)^t. \tag{2.12}$$

Now by assumption (and the fact that v is a scalar multiple of μ_i), we have

$$\Gamma^T_{\Pi_t,\dots,\Pi_1}\Gamma_{\Pi_t,\dots,\Pi_1}\mathsf{flatten}(v^{\otimes t}) = \mathsf{flatten}(v^{\otimes t}) + u$$

where u is a vector with

$$\|u\| \le \sqrt{1-(1-t\epsilon)^2} \le \sqrt{2t\epsilon}\,.$$

By Claim 2.4.5 and Markov's inequality, we have that with probability at least $1-\delta/10$ over the choice of z,

$$|u \cdot \mathsf{flatten}(P_t(z))| \le \sqrt{10/\delta}\cdot(t+\|\mu_i\|)^t\,\|u\| \le \sqrt{20t\epsilon/\delta}(2t\,\|\mu_i\|)^t \le 1$$

where the last step follows from our condition on ϵ. Now we can combine this with (2.12) to get that

$$\langle \mathsf{flatten}(P_t(z)), \Gamma^T_{\Pi_t,\dots,\Pi_1}\Gamma_{\Pi_t,\dots,\Pi_1}\mathsf{flatten}(v^{\otimes t})\rangle \ge (0.8\,\|\mu_i\|)^t - 1 \ge (0.5\,\|\mu_i\|)^t\,. \tag{2.13}$$

Now let

$$w = \Gamma^T_{\Pi_t,\dots,\Pi_1}\Gamma_{\Pi_t,\dots,\Pi_1}\mathsf{flatten}(v^{\otimes t})\,.$$

Note that $\|w\| \le 1$ because it is the projection of a unit vector $\mathsf{flatten}(v^{\otimes t})$ onto some subspace. By Corollary 2.4.12 and Markov's inequality, with probability at least $1-\delta/10$ over the randomness in z, we have

$$\mathop{\mathbb{E}}_{z_1,\dots,z_{2t-1}\sim\mathcal{D}}\left[(w\cdot\mathsf{flatten}(R(z,z_1,\dots,z_{2t-1})))^2\right] \le (10/\delta)(20t)^{2t}(\|\mu_i\|^{2t}+1)\,.$$

Assuming that the above holds, we now treat z as fixed. Note that the mean of $R(z,z_1,\dots,z_{2t-1})$ is $P_t(z)$ by Corollary 2.4.11. Let R be obtained by averaging $R(z,z_1,\dots,z_{2t-1})$ over $n = ((10^3t)^t/\delta)^3$ independent trials for $z_1,\dots,z_{2t-1}$. Of course the mean of R is also $P_t(z)$. Using

the above, we have that

$$\begin{aligned}
&\mathbb{E}\left[(w \cdot (\mathsf{flatten}(R) - \mathsf{flatten}(P_t(z))))^2\right] \\
&= \frac{1}{n} \mathop{\mathbb{E}}_{z_1,\ldots,z_{2t-1}\sim\mathcal{D}} \left[(w \cdot (\mathsf{flatten}(R(z, z_1, \ldots, z_{2t-1}) - P_t(z))))^2\right] \\
&\leq \frac{1}{n} \mathop{\mathbb{E}}_{z_1,\ldots,z_{2t-1}\sim\mathcal{D}} \left[(w \cdot \mathsf{flatten}(R(z, z_1, \ldots, z_{2t-1})))^2\right] \\
&\leq (\delta/(10^3 t)^t)^2 (20t)^{2t} (\|\mu_i\|^{2t} + 1) \\
&\leq \frac{\delta}{10} \left(\frac{\|\mu_i\|}{20}\right)^{2t}.
\end{aligned}$$

Thus, with probability at least $1 - \delta/10$, we have

$$|w \cdot (\mathsf{flatten}(R) - \mathsf{flatten}(P_t(z)))| \leq (0.05 \|\mu_i\|)^t.$$

If this holds, then combining with (2.13) implies

$$w \cdot \mathsf{flatten}(R) \geq (0.4 \|\mu_i\|)^t$$

but since $v^{\otimes t}$ is a unit vector and w is its projection onto the subspace spanned by the rows of $\Gamma_{\Pi_t,\ldots,\Pi_1}$, the above implies

$$\|\Gamma_{\Pi_t,\ldots,\Pi_1} \mathsf{flatten}(R)\| \geq (0.4 \|\mu_i\|)^t.$$

Note that the expression $\Gamma_{\Pi_t,\ldots,\Pi_1} \mathsf{flatten}(R)$ is exactly equivalent to the vector A that is tested by Algorithm 3 so combining all of the failure probabilities, we conclude that with probability at least $1 - \delta$, the output of Algorithm 3 will be FAR. ■

2.8 Learning Mixtures of Poincare Distributions

We are now ready to prove Theorem 2.2.3, our full result for mixtures of Poincare distributions. Let $\mathcal{D}' = (\mathcal{D} - \mathcal{D})/\sqrt{2}$ i.e. $\mathcal{D}'$ is the distribution of the difference between

two independent samples from $\mathcal{D}$ scaled down by $\sqrt{2}$. The $\sqrt{2}$ scaling ensures that $\mathcal{D}'$ is 1-Poincare by Fact 2.3.7. Note that we can take pairwise differences between samples to simulate access to the mixture

$$\mathcal{M}' = (w_1^2 + \cdots + w_k^2)\mathcal{D}' + \sum_{i \neq j} w_i w_j \mathcal{D}'((\mu_i - \mu_j)/\sqrt{2}) \,.$$

We will run Algorithm 1 and Algorithm 3 for the distribution $\mathcal{D}'$ and mixture $\mathcal{M}'$. Note that the test in Algorithm 3 will test for a pair of samples say z, z', from components $\mathcal{D}(\mu_i), \mathcal{D}(\mu_j)$ of the original mixture, whether $\|\mu_i - \mu_j\|$ is large or zero. Running this test between all pairs of samples will let us form clusters of samples that correspond to each of the components. We begin with a lemma that more precisely specifies the guarantees of this test and then Theorem 2.2.3 will follow easily from it.

Lemma 2.8.1. *Let $\mathcal{D}$ be a 1-Poincare distribution on $\mathbb{R}^d$. Let*

$$\mathcal{M} = w_1\mathcal{D}(\mu_1) + \cdots + w_k\mathcal{D}(\mu_k)$$

be a mixture of translated copies of $\mathcal{D}$. Let w^*, s, δ be parameters and assume that $s \geq (\log k/(w^*\delta))^{1+c}$ for some $0 < c < 1$. Also assume that

$$\max \|\mu_i - \mu_j\| \leq \min((k/(w^*\delta))^2, s^C)$$

for some C. There is an algorithm that takes $n = \mathrm{poly}((kd/(w^*\delta))^{C/c})$ samples from $\mathcal{M}$ and $\mathcal{D}$ and runs in $\mathrm{poly}(n)$ time and achieves the following testing guarantees: for a pair of (independent) samples $z \sim \mathcal{D}(\mu_i), z' \sim \mathcal{D}(\mu_j)$,

- *If $i = j$ and $w_i \geq w^*$ then with probability $1 - \delta$ the output is* accept
- *If $w_i, w_j \geq w^*$ and $\|\mu_i - \mu_j\| \geq s$ then with probability $1 - \delta$ the output is* reject

where the randomness is over the samples z, z' and the random choices in the algorithm.

Remark. *Note that Lemma 2.8.1 actually does not require a minimum separation or minimum mixing weight but instead just guarantees to detect when two samples come from components that have nontrivial mixing weights and whose means are far apart. We will focus on achieving inverse polynomial accuracy i.e.* $\delta = (w^*/k)^{O(1)}$ *so the required separation will be* $(\log k/w^*)^{1+c}$ *and the runtime will be polynomial in* $k, d, 1/w^*$.

Proof. Let t be the minimum integer such that

$$\left(\frac{s}{\log(k/(w^*\delta))}\right)^t \geq (k/(w^*\delta))^{10}\,.$$

Note that we clearly must have that

$$\max \|\mu_i - \mu_j\|^t \leq s^{tC} \leq \left(\frac{s}{\log(k/(w^*\delta))}\right)^{Ct/c} \leq (k/(w^*\delta))^{20C/c}$$
$$t \leq \frac{20\log(k/(w^*\delta))}{c\log\log(k/(w^*\delta))}\,.$$

Now we run Algorithm 1 with distribution $\mathcal{D}'$ and mixture $\mathcal{M}'$ and parameter t to obtain projection matrices $\Pi_t, \ldots, \Pi_1$. Set

$$\epsilon = \frac{\delta}{(10ts^C)^{4t}}\,.$$

Note that by the above, the runtime and sample complexity of $n = \mathrm{poly}((kd/(w^*\delta))^{C/c})$ suffices for us to apply Lemma 2.6.5 with accuracy parameter ϵ and weight threshold $(w^*)^2$. This implies that for all $i \neq j$ such that $\|\mu_i - \mu_j\| \geq \sqrt{2}$ and $w_i, w_j \geq w^*$, we have

$$\left\|\Gamma_{\Pi_s,\ldots,\Pi_1}\mathsf{flatten}((\mu_i - \mu_j)^{\otimes s})\right\| \geq (1 - s\epsilon)\,\|\mu_i - \mu_j\|^s \tag{2.14}$$

for all $s = 1, 2, \ldots, t$. Now we run Algorithm 3 with projection matrices $\Pi_t, \ldots, \Pi_1$, accuracy δ and threshold

$$\tau = (0.2s)^t\,.$$

We use Algorithm 3 to test $(z - z')/\sqrt{2}$ and we output Accept if Algorithm 3 returns

CLOSE and we output REJECT if Algorithm 3 returns FAR. Note that runtime and sample complexity of $n = \text{poly}((kd/(w^*\delta))^{C/c})$ suffice. Also $(z - z')/\sqrt{2}$ is equivalent to a sample from $\mathcal{D}'((\mu_i - \mu_j)/\sqrt{2})$. We now apply Lemma 2.7.1 to verify the first guarantee of our test. Note that it suffices to verify that

$$\tau \geq \frac{(20t)^t k}{\delta}$$

but this clearly holds because

$$\frac{\tau}{(20t)^t} = \left(\frac{0.2s}{20t}\right)^t \geq \left(\frac{cs}{10^4 \log(k/(w^*\delta))}\right)^t \geq \frac{k}{\delta},.$$

Thus, we conclude that if z, z' are drawn from the same component then the output is ACCEPT with probability at least $1 - \delta$. Now we use Lemma 2.7.2 to verify the second guarantee of our test. Note that

$$\frac{\|\mu_i - \mu_j\|}{\sqrt{2}} \geq \frac{s}{\sqrt{2}} \geq 10^4(\log 1/\delta + t)\,.$$

Also note that the threshold clearly satisfies $\tau \leq (0.4\,\|\mu_i - \mu_j\|/\sqrt{2})^t$ (note that the $\sqrt{2}$ comes from the fact that we scale the difference down by $\sqrt{2}$). Combined with the guarantee in (2.14), we conclude that the output is REJECT with probability at least $1 - \delta$ and we are done. ■

Using Lemma 2.8.1, it is not difficult to prove Theorem 2.2.3.

Proof of Theorem 2.2.3. Recall by the reduction in Section 2.3.3 that we may assume $d \leq k$ and that $\|\mu_i - \mu_j\| \leq O((k/w_{\min})^2)$ for all i, j. Now we will do the following process to estimate the means of the components.

1. Draw a sample $z \sim \mathcal{M}$
2. Take $m = (k/(w_{\min}\alpha))^{10^2}$ samples $z_1, \ldots, z_m \sim \mathcal{M}$
3. Use Lemma 2.8.1 with parameters $w^* = w_{\min}$ and $\delta = (w_{\min}\alpha/k)^{10^4}$ to test the pair of samples z, z_i for all for all $i \in [m]$

4. Let $S \subset [m]$ be the set of all i that are ACCEPTED and compute $\mu = \frac{1}{|S|}\sum_{i \in S} z_i$

We first argue that if z is a sample from $\mathcal{D}(\mu_i)$ for some i, then with $1 - (w_{\min}\alpha/k)^{10^2}$ probability, the procedure returns μ such that $\|\mu - \mu_i\| \leq 0.1\alpha$. This is because by the guarantees of Lemma 2.8.1, with probability at least $1 - (w_{\min}\alpha/k)^{10^3}$ the test will accept all samples from among $z_1, \ldots, z_m$ that are from the component $\mathcal{D}(\mu_i)$ and reject all of the others. Also, with high probability, there will be at least $(k/(w_{\min}\alpha))^{99}$ samples from the component $\mathcal{D}(\mu_i)$ so by Claim 2.3.8 and union bounding all of the failure probabilities, we have that if z is a sample from $\mathcal{D}(\mu_i)$ then $\|\mu - \mu_i\| \leq 0.1\alpha$ with probability at least $1 - (w_{\min}\alpha/k)^{10^2}$.

Now it suffices to repeat the procedure in steps $1-4$ for $l = (k/(w_{\min}\alpha))^{10^2}$ independent samples $z \sim \mathcal{M}$. This gives us a list of means say $S = \{\widetilde{\mu}_1, \ldots, \widetilde{\mu}_l\}$. The previous argument implies that most of these estimates will be close to one of the true means and that all of the true means will be represented. To ensure that with high probability we output exactly one estimate corresponding to each true mean and no extraneous estimates, we do a sort of majority voting.

We will inspect the estimates in S one at a time and decide whether to output them or not. Let T be the set of estimates that we will output. Note that T is initially empty. Now for each i, let S_i be the subset of $\{\widetilde{\mu}_1, \ldots, \widetilde{\mu}_l\}$ consisting of all means with $\|\widetilde{\mu}_j - \widetilde{\mu}_i\| \leq 0.2\alpha$. If $|S_i| \geq 0.9 w_{\min} l$ and $\widetilde{\mu}_i$ is not within α of any element of T then add $\widetilde{\mu}_i$ to T. Otherwise do nothing.

We claim that with high probability, this procedure returns one estimate corresponding to each true means and nothing extraneous. First, with high probability there will be no extraneous outputs because if say $\widetilde{\mu}_i$ is at least 0.5α away from all of the true means, then with high probability we will have $|S_i| < 0.9 w_{\min} l$. Now it is also clear that with high probability we output exactly one estimate corresponding to each true mean (since l is sufficiently large that with high probability we get enough samples from each component). Thus, with high probability, the final output will be a set of means that are within α of the true means up to some permutation. Once we have learned the means, we can learn the mixing weights

by simply taking fresh samples from $\mathcal{M}$ and clustering since with high probability, we can uniquely identify which component a sample came from. This completes the proof. ■

The clustering guarantee in Corollary 2.2.4 follows as an immediate consequence of Theorem 2.2.3.

Proof of Corollary 2.2.4. Let the estimated means computed by Theorem 2.2.3 for $\alpha = (w_{\min}/k)^{10}$ be $\widetilde{\mu_1}, \ldots, \widetilde{\mu_k}$. Now for all $j_1, j_2 \in [k]$ with $j_1 \neq j_2$, let

$$v_{j_1 j_2} = \frac{\widetilde{\mu_{j_1}} - \widetilde{\mu_{j_2}}}{\|\widetilde{\mu_{j_1}} - \widetilde{\mu_{j_2}}\|}.$$

Now given a sample z from $\mathcal{M}$, we compute the index j such that for all j_1, j_2, we have

$$|v_{j_1 j_2} \cdot (\widetilde{\mu_j} - z)| \leq (\log(k/w_{\min}))^{1+0.5c}.$$

Note that by the guarantees of Theorem 2.2.3, there is a permutation π such that $\left\|\widetilde{\mu_{\pi(i)}} - \mu_i\right\| \leq \alpha$ for all i. If z is a sample from $\mathcal{D}(\mu_i)$, then by the tail bound in Fact 2.3.7, with high probability the unique index j that satisfies the above is exactly $j = \pi(i)$ and thus, we recover the ground truth clustering with high probability. ■

2.9 Sharper Bounds for Gaussians

For spherical Gaussians, i.e. when $\mathcal{D} = N(0, I)$, we can obtain stronger results. Our results are stronger in two ways. First, we can improve the minimum separation to $(\log(k/w_{\min}))^{1/2+c}$ instead of $(\log(k/w_{\min})^{1+c}$. Secondly, we can remove the assumption about the maximum separation by using a recursive clustering routine. We first focus on improving the minimum separation. To get this improvement, we will prove sharper quantitative bounds in our implicit moment tensor and testing subroutines.

2.9.1 Hermite Polynomials

Naturally, for the standard Gaussian $N(0, I)$, the adjusted polynomials $P_{t,N(0,I)}$ are exactly the Hermite polynomials. In this section, we define and go over a few basic properties of the Hermite polynomials. Many of these are from [65]. In the next subsection, we will show how to exploit specific properties of the Hermite polynomials to get sharper versions of the results in Sections 2.6 and 2.7.

Definition 2.9.1. *We will use $P(S)$ to denote the set of partitions of a set S. We use $P^2(S)$ to denote the set of partitions of S into subsets of size 2. We use $P^{1,2}(S)$ to denote the set of partitions of S into subsets of size 1 and 2.*

Definition 2.9.2 (Hermite Polynomial Tensor (see [65]))**.** *Let $X = (X_1, \ldots, X_d)$ be a vector of d formal variables. We define the Hermite polynomial tensor*

$$h_t(X) = \sum_{P \in P^{1,2}([t])} \bigotimes_{\{a,b\} \in P} (-I)^{(a,b)} \otimes \bigotimes_{\{c\} \in P} X^{(c)}$$

where I denotes the $d \times d$ identity matrix.

Below we summarize several well-known properties that the Hermite polynomials satisfy.

Claim 2.9.3 (See [65])**.** *Let $G = N(\mu, I)$. Then*

$$\mathop{\mathbb{E}}_{z \sim G}[h_t(z)] = \mu^{\otimes t}\,.$$

Claim 2.9.4 (See [65])**.** *We have the identity*

$$\mathop{\mathbb{E}}_{z \sim N(0,I)}[z^{\otimes t}] = \sum_{P \in P^2([t])} \bigotimes_{\{a,b\} \in P} I^{(a,b)}\,.$$

We also have that the Hermite polynomials are exactly the adjusted polynomials (recall Definition 2.4.2) for the standard Gaussian.

Claim 2.9.5. *For any $x \in \mathbb{R}^d$ and integer t, we have*

$$P_{t,N(0,I)}(x) = h_t(x)\,.$$

Proof. We can verify the desired statement by induction. The base cases for $t = 1, 2$ are clear. To finish, we can simply plug Claim 2.9.4 into the recursion in (2.7) to verify the inductive step. ■

We could get a bound on the variance of the Hermite polynomials using Claim 2.4.5. However, since our goal for Gaussians will be to get separation $(\log(k/w_{\min}))^{1/2+c}$ instead of separation $(\log(k/w_{\min}))^{1+c}$, we will obtain a stronger bound by hand. Fortunately, we will only need the stronger bound for when the mean is 0 and can get away with using the general bounds from Section 2.4 for when the mean μ is nonzero.

Claim 2.9.6 (From [65])**.** *We have*

$$\mathop{\mathbb{E}}_{z\sim N(\mu,I)}[h_t(z) \otimes h_t(z)] = \sum_{S_1,S_2\subset[t],|S_1|=|S_2|} \sum_{\substack{\text{Matchings } \mathcal{P} \\ \text{of } S_1,S_2}} \bigotimes_{\{a,b\}\in\mathcal{P}} I^{(a,t+b)} \bigotimes_{c\notin S_1} \mu^{(c)} \bigotimes_{c\notin S_2} \mu^{(t+c)}\,.$$

Flattening the above expression for $\mathbb{E}_{z\sim G}[h_t(z)\otimes h_t(z)]$ into a $d^t \times d^t$ matrix in the natural way, we immediately get a bound on the covariance of $h_t(z)$ for $z \sim G$

Corollary 2.9.7. *Let*

$$M(z) = \mathsf{flat}(h_t(z)) \otimes \mathsf{flat}(h_t(z))\,.$$

Then

$$\mathop{\mathbb{E}}_{z\sim N(0,I)}[M(z)] \preceq t!I$$

where I is the $d^t \times d^t$ identity matrix.

Proof. This follows immediately from the formula in Claim 2.9.6 since there are $t!$ terms that are a tensor product of t copies of the identity matrix and each of these has spectral norm 1. ■

We will also need the following property, that the Hermite polynomials are an orthogonal family. Again, this property was not true for general Poincare distributions but will be used in getting stronger quantitative bounds that will let us deal with smaller separation.

Claim 2.9.8. *For integers $t \neq t'$, we have*

$$\mathop{\mathbb{E}}_{z\sim N(0,I)}[\mathsf{flat}(h_t(z)) \otimes \mathsf{flat}(h_{t'}(z))] = 0\,.$$

Proof. WLOG $t' < t$. We first prove

$$\mathop{\mathbb{E}}_{z\sim N(0,I)}[h_t(z) \otimes z^{\otimes t'}] = 0\,.$$

Substitute the expression in Definition 2.9.2 into the LHS and then use Claim 2.9.4. The above then follows from direct computation. Next, since the above holds for all $t' < t$ and $h'_t(z)$ is a sum of monomials of degree at most t', we immediately get that

$$\mathop{\mathbb{E}}_{z\sim N(0,I)}[h_t(z) \otimes h_{t'}(z)]$$

as desired. ■

2.9.2 Implicit Representations of Hermite Polynomials

Similar to Section 2.4.3, we will need implicit representations of the Hermite polynomial tensors. While technically the representation in Definition 2.9.2 is already implicit, the identity matrices $I^{a,b}$ do not behave well when we try to apply our iterative projection map in Section 2.5 and thus we will again, similar to Section 2.4.3, need to obtain a representation as a sum of a polynomial number of "rank-1" tensors.

The next set of definitions exactly parallel those in Section 2.4.3.

Definition 2.9.9. *For $x_1, \dots, x_t \in \mathbb{R}^d$, define the polynomial*

$$Q_t(x_1, \dots, x_t) = \sum_{\substack{S_1 \cup \cdots \cup S_t = [t] \\ |S_i \cap S_j| = 0}} \frac{(-1)^{\mathcal{C}\{S_1,\dots,S_t\}}}{\binom{t-1}{\mathcal{C}\{S_1,\dots,S_t\}-1}} \left(h_{|S_1|}(x_1)\right)^{(S_1)} \otimes \cdots \otimes \left(h_{|S_t|}(x_t)\right)^{(S_t)} . \tag{2.15}$$

Definition 2.9.10. *For $x_1, \dots, x_{2t} \in \mathbb{R}^d$, define the polynomial*

$$R_t(x_1, \dots, x_{2t}) = -Q_t(x_1, \dots, x_t) + Q_t(x_{t+1}, \dots, x_{2t}) .$$

Remark. *Note we will use the same letters Q_t, R_t as in the general case. All of the proceeding sections deal specifically with GMMs and we will only consider the corresponding Q_t, R_t so there will be no ambiguity.*

By Claim 2.9.5, we have the following (which is the exact same as Corollary 2.4.10).

Corollary 2.9.11. *We have the identity*

$$R_t(x_1, \dots, x_{2t}) = \sum_{\substack{S_1 \cup \cdots \cup S_t = [t] \\ |S_i \cap S_j| = 0}} \frac{(-1)^{\mathcal{C}\{S_1,\dots,S_t\}-1}}{\binom{t-1}{\mathcal{C}\{S_1,\dots,S_t\}-1}} \left(x_1^{\otimes S_1} \otimes \cdots \otimes x_t^{\otimes S_t} - x_{t+1}^{\otimes S_1} \otimes \cdots \otimes x_{2t}^{\otimes S_t}\right) .$$

Claim 2.9.5 also implies that Corollary 2.4.11 and Corollary 2.4.12 still hold and can be used in the analysis. We now prove the one strengthened bound that we will need.

Lemma 2.9.12. *We have*

$$\mathop{\mathbb{E}}_{z_1,\dots,z_{2t} \sim N(0,I)} [\mathit{flatten}(R_t(z_1, \dots, z_{2t}))^{\otimes 2}] \preceq (2t)^t I_{d^t} .$$

Proof. Note that by Claim 2.9.8 and Corollary 2.9.7,

$$\begin{aligned}
&\mathop{\mathbb{E}}_{z_1,\ldots,z_{2t}\sim N(0,I)}[\mathsf{flatten}(R_t(z_1,\ldots,z_{2t}))^{\otimes 2}] \\
&= 2\sum_{\substack{S_1\cup\cdots\cup S_t=[t]\\ |S_i\cap S_j|=0}} \frac{1}{\binom{t-1}{\mathcal{C}\{S_1,\ldots,S_t\}-1}^2}\mathsf{flatten}\left(\left(h_{|S_1|}(x_1)\right)^{(S_1)}\otimes\cdots\otimes\left(h_{|S_t|}(x_t)\right)^{(S_t)}\right)^{\otimes 2} \\
&\preceq 2\sum_{\substack{S_1\cup\cdots\cup S_t=[t]\\ |S_i\cap S_j|=0}} |S_1|!|S_2|\cdots|S_t|!I_{d^t} \\
&\preceq 2I_{d^t}\sum_{\substack{s_1+\cdots+s_t=t\\ s_i\geq 0}}\binom{t}{s_1,\ldots,s_t}s_1!\cdots s_t! \\
&\preceq (2t)^t I_{d^t}
\end{aligned}$$

where as before we use the trick of reindexing the sum so that $s_1,\ldots,s_t$ are the sizes of $S_1,\ldots,S_t$ respectively. This completes the proof. ■

Note that the key difference of the above compared to Corollary 2.4.12 is that the RHS is t^t instead of t^{2t}. This is the key to improving the separation.

2.9.3 Testing Samples

All of the algorithms and results in Section 2.6 still hold (with the distribution $\mathcal{D}$ set to $N(0,I)$). We also run the testing algorithm in Section 2.7. It remains to prove that the distinguishing power of the test is stronger when specialized to Gaussians. Recall that the key lemmas were Lemma 2.7.1 and Lemma 2.7.2. The strengthened versions are stated below.

Lemma 2.9.13. *Let $t\in\mathbb{N}$ and $0<\delta<0.01$ be some parameters. Let $\Pi_t,\ldots,\Pi_2\in\mathbb{R}^{k\times dk},\Pi_1\in\mathbb{R}^{k\times d}$ be any matrices with orthonormal rows. Let τ be some parameter satisfying $\tau\geq(2t)^{t/2}k/\delta$. Let $z\sim N(0,I)$. Then with probability at least $1-\delta$, Algorithm 3 run with these parameters outputs* Close *where the randomness is over z and the random choices within Algorithm 3.*

Lemma 2.9.14. *Let $t \in \mathbb{N}$ and $0 < \delta < 0.01$ be some parameters. Let $z \sim N(\mu_i, I)$ where $\|\mu_i\| \geq 10^4(\sqrt{\log 1/\delta} + \sqrt{t})$. Let τ be some parameter satisfying $\tau \leq (0.4\|\mu_i\|)^t$. Assume that the matrices $\Pi_t, \ldots, \Pi_2 \in \mathbb{R}^{k \times dk}, \Pi_1 \in \mathbb{R}^{k \times d}$ satisfy that*

$$\left\|\Gamma_{\Pi_t,\ldots,\Pi_1} \mathsf{flatten}(\mu_i^{\otimes t})\right\| \geq (1 - t\epsilon)\|\mu_i\|^t \,.$$

where

$$\epsilon < \frac{\delta}{(10t\|\mu_i\|)^{4t}} \,.$$

Then with probability at least $1 - \delta$, Algorithm 3 run with these parameters outputs FAR *(where the randomness is over z and the random choices within Algorithm 3).*

There are two key difference compared to Lemma 2.7.1 and Lemma 2.7.2. First, in Lemma 2.9.13, the bound on τ involves $t^{t/2}$ instead of t^t. Second, in Lemma 2.9.14, we require separation $O(\sqrt{\log 1/\delta} + \sqrt{t})$ instead of $O(\log 1/\delta + t)$. When we combine the two, the inequality that we need for our test to be meaningful is

$$(0.4\|\mu_i\|)^t \geq (2t)^{t/2}k/\delta \,.$$

All of the settings of parameters will be the same as before i.e. $t \sim O\left(c^{-1}\log(k/w^*)/\log\log(k/w^*)\right)$ and $\delta = \mathrm{poly}(w^*/k)$ except now we can get away with $\|\mu_i\| \geq (\log(k/w^*))^{1/2+c}$ for some constant $c > 0$ which allows us to improve the minimum separation.

The proof of Lemma 2.9.13 is essentially exactly the same as the proof of Lemma 2.7.1.

Proof of Lemma 2.9.13. Exactly the same as the proof of Lemma 2.7.1 except use Lemma 2.9.12 instead of Corollary 2.4.12 to bound the variance of R_t. ■

The proof of Lemma 2.9.14 requires one additional modification. In particular, we will use a stronger version of Claim 2.7.3 that we now prove.

Claim 2.9.15. *Let $x \in \mathbb{R}^d$. Let $v \in \mathbb{R}^d$ be a vector such that $v \cdot x \geq 20\sqrt{t}\,\|v\|$. Then*

$$\langle h_t(x), v^{\otimes t} \rangle \geq (0.9v \cdot x)^t \,.$$

Proof. WLOG v is a unit vector. Let $a = v \cdot x$. Using Definition 2.9.2 and elementary counting, we have

$$\langle h_t(x), v^{\otimes t} \rangle = \sum_{j=0}^{\lfloor t/2 \rfloor} (-1)^j a^{t-2j} \frac{t!}{2^j j!(t-2j)!} \,.$$

The RHS is a polynomial in a and is exactly the standard univariate Hermite polynomial, which we denote $H_t(a)$. Note it can be easily verified that the univariate Hermite polynomials satisfy the recurrence

$$H_t(a) = aH_{t-1}(a) - (t-1)H_{t-2}(a) \,.$$

Thus, we can write

$$H_t(a) = \det \begin{bmatrix} a & 1 & & & \\ 1 & a & \sqrt{2} & & \\ & \sqrt{2} & a & \ddots & \\ & & \ddots & \ddots & \sqrt{t-1} \\ & & & \sqrt{t-1} & a \end{bmatrix}$$

where the blank entries are 0. Let the matrix on the RHS be M. M is a $t \times t$ matrix that has a on the diagonal and adjacent to the diagonal the entries are $M_{i(i+1)} = M_{(i+1)i} = \sqrt{i}$ and 0 everywhere else. Note that this implies that H_t has t real roots (since it is the characteristic polynomial of a symmetric matrix). Also, all roots have magnitude at most $2\sqrt{t}$ since if $|a| \geq 2\sqrt{t}$ then the matrix is diagonally dominant and cannot have determinant 0. Thus, we can write $H_t(a) = (a - r_1) \cdots (a - r_t)$ where $-2\sqrt{t} \leq r_1, \ldots, r_t \leq 2\sqrt{t}$. For $a \geq 20\sqrt{t}$, we clearly have $H_t(a) \geq (0.9a)^t$ and this completes the proof. ■

Proof of Lemma 2.9.14. Exactly the same as the proof of Lemma 2.7.2 except using Claim 2.9.15 instead of Claim 2.7.3. ■

2.10 Clustering Part 1: Building Blocks

In order to prove our full result for GMMs, we need to be able to deal with mixtures for which the maximum separation is much larger (superpolynomial) than the minimum separation. To do this, we will have a recursive clustering procedure where we are able to remove half of the components each time. Roughly, we find a direction v for which there are two components whose separation along direction v is large (say at least $(\log(k/w_{\min}))^{1/2+c}$). We call this a signal direction.

Next, imagine projecting all of our samples onto the direction v. For each component, essentially all of its samples will be in an interval of width $O(\sqrt{\log(k/w_{\min})})$. Thus, since there are two components that are sufficiently separated, we can find an interval of width $O(\sqrt{\log(k/w_{\min})})$ for which

- For at least one component, essentially all of its samples are in this interval
- At most half of the components ever generate samples in this interval

We then imagine going back to $\mathbb{R}^d$ but only keeping the samples whose projection onto v lies in this interval. Of course, after this restriction, the distribution is no longer a mixture of Gaussians because there could be a component that is cut in half by the boundary of the interval. The key observation is that we can project all of the remaining samples (after the restriction) onto the orthogonal complement of v. This new distribution will be a mixture of Gaussians in $d-1$ dimensions with at most half as many components. Furthermore, because we restricted to an interval of width $O(\sqrt{\log(k/w_{\min})})$, we can argue that this projection onto the orthogonal complement of v does not reduce separations by too much.

The actual clustering algorithm will be more complicated than the outline above for technical reasons. There are several additional details that we need to deal with such as the fact that we may create components with very small mixing weights when we restrict to samples in an interval. Also, the procedure for halving the number of remaining components only works when there is some pair of means whose separation is at least $\text{poly}(\log(k/w_{\min}))$ (which is larger, but polynomially related to the minimum separation). Thus, after running

the halving procedure sufficiently many times, we then have to run a full clustering routine that has guarantees similar to Theorem 2.2.3.

Remark. *Note that for general Poincare distributions, this type of approach seems infeasible because after restricting to an interval in direction v, projecting onto the orthogonal complement still may not "fix" all of the components of the mixture be translations of the same distribution again.*

In this section, we introduce the building blocks for our complete clustering algorithm. The two main components in this section are Lemma 2.10.11 which roughly allows us to, in an arbitrary mixture, find a signal direction along which some two components have separation comparable to the maximum separation and Lemma 2.10.12 which allows us to do full clustering when the maximum separation is at most $\mathrm{poly}(\log(k/w_{\min}))$.

2.10.1 Notation and Terminology

Throughout the proceeding sections, we will use $0 < c < 1$ do denote an arbitrary (small) positive constant that does not change with the other problem parameters.

Definition 2.10.1. *For a subspace $V \subset \mathbb{R}^d$ and a point $x \in \mathbb{R}^d$, we use $\mathsf{Proj}_V(x)$ to denote the projection of x onto V.*

Definition 2.10.2. *For a subspace $V \subset \mathbb{R}^d$ we use $V^\perp$ to denote its orthogonal complement.*

Definition 2.10.3. *For a Gaussian with mean μ and covariance Σ, we use $N_V(\mu, \Sigma)$ to denote its projection onto a subspace V.*

We will now introduce terminology for describing mixtures and how well-behaved they are.

Definition 2.10.4. *We say a GMM $\mathcal{M} = w_1 N(\mu_1, I) + \cdots + w_k N(\mu_k, I)$ is s-separated if for all $i \neq j$, $\|\mu_i - \mu_j\| \geq s$.*

Definition 2.10.5. *For a GMM $\mathcal{M} = w_1 N(\mu_1, I) + \cdots + w_k N(\mu_k, I)$ and parameter $w^* > 0$, we say that $\mathcal{M}$ is w^*-reasonable if the following holds:*

$$\max_{\substack{i,j \\ w_i, w_j \geq w^*}} \|\mu_i - \mu_j\| \geq \sqrt{\max_{i,j} \|\mu_i - \mu_j\|}\,.$$

We also use the convention that a trivial mixture consisting of one component is reasonable.

Remark. *Note the choice that the RHS is a square root is arbitrary. Any constant power would suffice.*

Roughly, the inequality in the definition says that there are two components whose mixing weights are at least w^* such that the separation between their means is comparable to the maximum separation between any two means. Note that the notion of comparable is very loose (the two quantities can be off by a square).

Definition 2.10.6. *For a distribution $\mathcal{M}$ and parameters p, Δ, we say that a direction, given by a unit vector $v \in \mathbb{R}^d$, is a (p, Δ)-signal direction for $\mathcal{M}$ if there is a real number θ such that*

$$\Pr_{z \sim \mathcal{M}}[v \cdot z \leq \theta - \Delta] \geq p$$
$$\Pr_{z \sim \mathcal{M}}[v \cdot z \geq \theta + \Delta] \geq p\,.$$

Roughly, a direction is a signal direction if the distribution $\mathcal{M}$ is nontrivially spread out along that direction. It will be important to note that we can easily check whether a direction is a signal direction.

Claim 2.10.7. *Let $\mathcal{M}$ be a distribution on $\mathbb{R}^d$ and let $v \in \mathbb{R}^d$ be a unit vector. For parameters p, Δ, given $\mathrm{poly}(d, 1/p)$ samples from $\mathcal{M}$, we can distinguish with high probability when v is a (p, Δ)-signal direction and when v is not a $(0.9p, \Delta)$-signal direction.*

Proof. We can simply take sufficiently many samples and check if v is a $(0.95p, \Delta)$-signal direction for the empirical distribution on the samples. ■

We will also need the following concentration inequalities about tails of a Gaussian. The first one is standard.

Claim 2.10.8. *Consider a Gaussian $N(\mu, I)$ in $\mathbb{R}^d$. For any parameter β, we have*

$$\Pr_{z\sim N(\mu,I)}[\|z-\mu\| \le \sqrt{2d+\beta}] \ge 1 - 2^{-\beta/8}\,.$$

The next inequality is closely connected to the notion of stability (see [43]) that has recently proved very useful in the field of robust statistics.

Claim 2.10.9 (See [43])**.** *Consider the standard Gaussian $N(0, I)$ in $\mathbb{R}^d$. Let $\epsilon > 0$ be some parameter. Given $n \ge (d/\epsilon)^8$ samples $z_1, \ldots, z_n$ from $N(0, I)$, with probability at least $1 - 2^{-d/\epsilon}$ the following property holds: for any subset $S \subset [n]$ with $|S| \ge \epsilon n$, we have*

$$\left\| \frac{1}{|S|} \sum_{i\in S} z_i \right\| \le 10\sqrt{\log 1/\epsilon}\,.$$

2.10.2 Clustering Test

Recall that for arbitrary Poincare distributions, one of the key ingredients was Lemma 2.8.1, an algorithm for testing whether two samples from a mixture $\mathcal{M}$ are from the same component or not. The analog (with strengthened quantitative bounds) that we will need for GMMs is stated below.

Lemma 2.10.10. *Let*

$$\mathcal{M} = w_1 N(\mu_1, I) + \cdots + w_k N(\mu_k, I)$$

be a GMM in $\mathbb{R}^d$. Let w^*, s, δ be parameters and assume that $s \ge (\log k/(w^*\delta))^{1/2+c}$ for some $0 < c < 1$. Also assume that

$$\max_{i,j} \|\mu_i - \mu_j\| \le \min((k/(w^*\delta))^2, s^C)$$

for some C. There is an algorithm that takes $n = \mathrm{poly}((kd/(w^*\delta))^{C/c})$ samples from $\mathcal{M}$ and

runs in poly(n) *time and achieves the following testing guarantees: for a pair of (independent) samples* $z \sim N(\mu_i, I), z' \sim N(\mu_j, I)$,

- *If* $i = j$ *and* $w_i \geq w^*$ *then with probability* $1 - \delta$ *the output is* accept
- *If* $w_i, w_j \geq w^*$ *and* $\|\mu_i - \mu_j\| \geq s$ *then with probability* $1 - \delta$ *the output is* reject

where the randomness is over the samples z, z' *and the random choices in the algorithm.*

Proof. The proof is exactly the same as the proof of Lemma 2.8.1 except we use Lemma 2.9.14 and Lemma 2.9.13 in place of Lemma 2.7.2 and Lemma 2.7.1 respectively. ■

2.10.3 Finding a Signal Direction

As mentioned before, to remove the constraint on the maximum separation, we will need to do recursive clustering. The key ingredient in the recursive clustering is finding a signal direction. In the next lemma, we show how to find a signal direction in a reasonable mixture. Note that if a mixture is not reasonable, then we have no guarantees. Roughly this is because if there are some components with means μ_i, μ_j such that $\|\mu_i - \mu_j\|$ is very large but whose mixing weights are very small then the few samples from these components will drastically affect all of our estimators. Fortunately, in the next section (see Claim 2.11.7), we show how to ensure that we always work with a reasonable mixture.

The main algorithm for finding a signal direction is summarized below. Roughly, we just randomly take two samples z, z' from $\mathcal{M}$. We then take the mean μ of samples z_i such that (z, z_i) passes the test in Lemma 2.10.10 and the mean μ' of samples z_i' such that (z, z_i') passes the test in Lemma 2.10.10. We then test whether the direction $\mu - \mu'$ is indeed a good enough signal direction and otherwise just try again with new samples.

Algorithm 4 FINDING A SIGNAL DIRECTION

Input: Sample access to GMM $\mathcal{M}$

Input: Parameters k, w^*, positive constant $c > 0$

Input: Desired separation Δ

Take two samples $z, z' \sim \mathcal{M}$

Take samples $z_1, \ldots, z_m, z'_1, \ldots, z'_m$ from $\mathcal{M}$ where $m = (k/w^*)^{10^2}$

for i = 1,2, ..., m **do**

Test pairs z, z_i and z', z'_i using Lemma 2.10.10 with parameters $w^*, \delta = (w^*/k)^{10^4}, s = 0.01\Delta$

Let S be the set of i such that the pair z, z_i is accepted.

Let S' be the set of i such that the pair z', z'_i is accepted.

Compute $\mu = \frac{1}{|S|}\sum_{i\in S} z_i$ and $\mu' = \frac{1}{|S'|}\sum_{i\in S'} z'_i$

Set $v = (\mu - \mu')/\left\|\mu - \mu'\right\|$

Test if v is a $(0.8w^*, 0.8\Delta)$ signal direction using Claim 2.10.7

Output v if test passes

Lemma 2.10.11. *Let $\mathcal{M} = w_1 N(\mu_1, I) + \cdots + w_k N(\mu_k, I)$ be a GMM. Let w^* be a parameter and assume that $\mathcal{M}$ is w^*-reasonable and $\|\mu_i - \mu_j\| \le O((k/w^*)^2)$ for all i, j. Also assume that there are i, j such that $w_i, w_j \ge w^*$ and $\|\mu_i - \mu_j\| \ge (\log(k/w^*))^{1/2+c}$ for some positive constant c. There is an algorithm that takes $n = \mathrm{poly}((dk/w^*)^{1/c})$ samples from $\mathcal{M}$ and $\mathrm{poly}(n)$ runtime and with high probability outputs a unit vector v such that v is a $(w^*/2, \max_{w_i, w_j \ge w^*} \|\mu_i - \mu_j\|/2)$-signal direction.*

Proof. Let

$$\Delta = \max_{w_i, w_j \ge w^*} \|\mu_i - \mu_j\| .$$

It suffices now to find a $(w^*/2, \Delta/2)$-signal direction. While we technically do not know Δ, we can guess Δ within a factor of 1.1 by using a multiplicative grid. We then try to find a $(0.8w^*, 0.8\Delta)$ signal direction for various guesses of Δ and test them using Claim 2.10.7 and output the one for the largest Δ that succeeds. Thus, we can essentially assume that

we know Δ up to a factor of 1.1.

Now we can just run Algorithm 4 repeatedly. It suffices to prove that with some inverse polynomial probability, the output v is a $(0.8w^*, 0.8\Delta)$-signal direction. We can then just repeat polynomially many times and test each output using Claim 2.10.7 to guarantee success with high probability. Let i, i' be indices such that $\|\mu_i - \mu_i'\| = \Delta$. With at least $(1/w^*)^2$ probability, z is drawn from $N(\mu_i, I)$ and z' is drawn from $N(\mu_i', I)$. Now using the guarantee of Lemma 2.10.10 and union bounding (note this is valid because $\mathcal{M}$ is w^*-reasonable), with at least $1 - (w^*/k)^{10^2}$ probability, we have the following properties

- For all $i \in S$, z_i is a sample from $N(\mu_j, I)$ where $\|\mu_j - \mu_i\| \leq 0.01\Delta$
- For all $i \in S'$, z_i is a sample from $N(\mu_j, I)$ where $\|\mu_j - \mu_{i'}\| \leq 0.01\Delta$
- $|S|, |S'| \geq 0.9w^*m$

However, combining these properties with Claim 2.10.9 implies that with high probability $\|\mu - \mu_i\|, \|\mu' - \mu_{i'}\| \leq 0.02\Delta$. This then implies that v has correlation 0.9 with the unit vector in the direction $\mu_i - \mu_{i'}$ which clearly implies that it is a $(0.8w^*, 0.8\Delta)$-signal direction. Overall, the success probability of a single trial is at least $0.9 \cdot (1/w^*)^2$ so repeating over polynomially many trials guarantees that we succeed with high probability. ■

2.10.4 Full Clustering with Bounded Maximum Separation

Similar to Theorem 2.2.3, we can do full clustering if the maximum separation is polynomially bounded in terms of the minimum separation. We have a slightly more technical requirement here that there may be some components with very small mixing weights but we only need to recover the means of the components with substantial mixing weights. This result will be used as the final step in our complete clustering algorithm when we have reduced to a submixture where the maximum separation is sufficiently small and then we can just fully cluster the remaining components.

Lemma 2.10.12. *Let $\mathcal{M} = w_1 N(\mu_1, I) + \cdots + w_k N(\mu_k, I)$ be a GMM. Let $c > 0$ be a positive constant and let w^* be a parameter that we are given. Assume that $\mathcal{M}$ is s-separated where $s = (\log(k/w^*))^{1/2+c}$ for some positive constant c. Also assume that $\max_{i,j} \|\mu_i - \mu_j\| \leq (\log(k/w^*))^4$. Given $n = \text{poly}((dk/w^*)^{1/c})$ samples from $\mathcal{M}$ and $\text{poly}(n)$ runtime, there is an algorithm that with high probability outputs a set of means $\{\widetilde{\mu}_1, \ldots, \widetilde{\mu}_r\}$ such that the following properties hold:*

- $r \leq k$
- *For all $i \in [k]$ such that $w_i \geq w^*$, there is some $j \in [r]$ such that $\|\mu_i - \widetilde{\mu}_j\| \leq 0.1$*
- *We have $\|\widetilde{\mu}_i - \widetilde{\mu}_j\| \geq s/2$ for all $i \neq j$*

Proof. The proof will be very similar to the proof of Theorem 2.2.3 except using the test in Lemma 2.10.10. Now we will do the following process to estimate the means of the components.

1. Draw a sample $z \sim \mathcal{M}$
2. Take $m = (k/w^*)^{10^2}$ samples $z_1, \ldots, z_m \sim \mathcal{M}$
3. Use Lemma 2.10.10 with parameters $w^* = (w^*/k)^{10}, \delta = (w^*/k)^{10^4}, s$ to test the pair of samples z, z_i for all for all $i \in [m]$
4. Let $S \subset [m]$ be the set of all i that are ACCEPTED and compute $\mu = \frac{1}{|S|} \sum_{i \in S} z_i$

Note that if z is a sample from $N(\mu_i, I)$ for some i with $w_i \geq (w^*/k)^5$, then with $1 - (w^*/k)^{10^2}$ probability, the procedure returns μ such that $\|\mu - \mu_i\| \leq 0.01$. This is because by the guarantees of Lemma 2.10.10, with probability at least $1 - (w^*/k)^{10^3}$ the test will accept all samples from among $z_1, \ldots, z_m$ that are from the component $N(\mu_i, I)$. Also, the only other samples that are accepted must be from components $N(\mu_j, I)$ with $w_j \leq (w^*/k)^{10}$. With high probability, among $z_1, \ldots, z_m$, there will be at least $0.9 w_i m \geq 0.9(w^*/k)^5 m$ samples from the component $N(\mu_i, I)$ and at most $2k(w^*/k)^{10} m \leq (w^*/k)^9 m$ from other components $N(\mu_j, I)$ with mixing weight smaller than $(w^*/k)^{10}$. Since we have a bound on the maximum

separation $\|\mu_i - \mu_j\| \leq (\log(k/w^*))^4$, with high probability, the mean μ of all of these samples satisfies $\|\mu - \mu_i\| \leq 0.01$.

Now we repeat the procedure in steps $1-4$ for $l = (k/w^*)^{10^2}$ independent samples $z \sim \mathcal{M}$. This gives us a list of means say $S = \{\widehat{\mu}_1, \ldots, \widehat{\mu}_l\}$. The previous argument implies that most of these estimates will be close to μ_i for some i with $w_i \geq (w^*/k)^5$ and furthermore that all such components will be represented. As in Theorem 2.2.3, to ensure that with high probability we output exactly one estimate corresponding to each true mean and no extraneous estimates, we do a sort of majority voting.

We will inspect the estimates in S one at a time and decide whether to output them or not. Let T be the set of estimates that we will output. Note that T is initially empty. Now for each i, let S_i be the subset of $\{\widehat{\mu}_1, \ldots, \widehat{\mu}_l\}$ consisting of all means with $\|\widehat{\mu}_j - \widehat{\mu}_i\| \leq 0.02$. If $|S_i| \geq 0.9w^*l$ and $\widehat{\mu}_i$ is not within 0.1 of any element of T then add $\widehat{\mu}_i$ to T. Otherwise do nothing. At the end we output everything in T.

We now argue that with high probability, the set T satisfies the desired properties. First, note that with high probability, all elements of T must be within 0.05 of one of the true means μ_i since otherwise, there would not be enough elements in the set S_j. Also it is clear that there can be at most one element $\widehat{\mu}_j \in T$ corresponding to each true mean μ_i. It remains to argue that with high probability, for all $i \in [k]$ such that $w_i \geq w^*$, there must be some element in T within 0.05 of μ_i. This is because for such an i, with high probability there will be at least $0.9w_il$ elements $\widehat{\mu}_j$ in S such that $\|\widehat{\mu}_j - \mu_i\| \leq 0.01$. Thus, if there are no elements already in T that are close to μ_i, then one of these $\widehat{\mu}_j$ will be added to T. Overall, we have shown that with high probability, the set T satisfies the desired properties and we are done. ■

2.11 Clustering Part 2: Recursive Clustering

In this section, we put together the building blocks from Section 2.10 in our complete clustering algorithm and complete the proof of Theorem 2.2.5.

2.11.1 Clustering Checkers

First, we introduce the concept of a checker. This will ease notation for later on when we need to consider various restrictions of a mixture involving restricting to samples whose projection onto a subspace V is close to a certain point $p \in V$.

Definition 2.11.1 (Checker)**.** *A checker, denoted (V, p, r) consists of a subspace $V \subset \mathbb{R}^d$, a point $p \in V$ and a positive real number r.*

Definition 2.11.2. *We say that a checker (V, p, r) contains a point x if $\|\mathsf{Proj}_V(x) - p\| \leq r$. We write $\mathsf{chk}_{V,p,r}(x) = 1$ if the checker contains the point x and $\mathsf{chk}_{V,p,r}(x) = 0$ otherwise.*

Definition 2.11.3. *Given a distribution $\mathcal{M}$ and a checker (V, p, r), we may take samples from $\mathcal{M}$, delete all of them that do not lie inside the checker (V, p, r), and then project the remaining samples onto $V^\perp$. We call the resulting distribution the reduction of $\mathcal{M}$ by the checker (V, p, r) and denote it $\mathsf{red}_{V,p,r}(\mathcal{M})$.*

2.11.2 Basic Properties

The key observation is that reducing a mixture of Gaussians by a checker (V, p, r) results in a new mixture with the same components (projected onto $V^\perp$) but with different mixing weights.

Claim 2.11.4. *Assume that we have a checker (V, p, r) and let a be the dimension of V. Given a GMM $\mathcal{M} = w_1 N(\mu_1, I) + \cdots + w_k N(\mu_k, I)$, the reduction of $\mathcal{M}$ is*

$$\mathsf{red}_{V,p,r}(\mathcal{M}) = \frac{\sum_{i=1}^k w_i \Pr_{z \sim N(\mu_i, I)}[\mathsf{chk}_{V,p,r}(z) = 1] N_{V^\perp}(\mu_i, I)}{\sum_{i=1}^k w_i \Pr_{z \sim N(\mu_i, I)}[\mathsf{chk}_{V,p,r}(z) = 1]}\,.$$

Proof. This follows immediately from the fact that for a sample z from a standard Gaussian $N(0, I)$, for any subspace V, the projections $\mathsf{Proj}_V(z)$ and $\mathsf{Proj}_{V^\perp}(z)$ are independent and distributed as standard Gaussians in the respective subspaces. ■

In light of Claim 2.10.8, reducing a GMM by a checker (V, p, r) where V has dimension a essentially removes all components whose means μ_i satisfy $\|\mathsf{Proj}_V(\mu_i) - p\| \geq r +$

$\omega(\sqrt{a+\log k})$. We now make this notion more formal by defining a truncation of a reduced mixture that involves deleting such components. We then argue that the truncation only affects the overall distribution by a negligible amount.

Definition 2.11.5. *Let $\mathcal{M} = w_1 N(\mu_1, I) + \cdots + w_k N(\mu_k, I)$ be a GMM and θ be some parameter. For a checker (V, p, r), define the truncated reduction of $\mathcal{M}$ as*

$$\mathsf{red}^{(\theta)}_{V,p,r}(\mathcal{M}) = \frac{\sum_{i \in S} w_i \Pr_{z \sim N(\mu_i, I)}[\mathsf{chk}_{V,p,r}(z) = 1] N_{V^\perp}(\mu_i, I)}{\sum_{i \in S} w_i \Pr_{z \sim N(\mu_i, I)}[\mathsf{chk}_{V,p,r}(z) = 1]}$$

where S is defined as the set of $i \in [k]$ such that μ_i is in the checker $(V, p, r+\theta)$. We say that the set S is the set of relevant components in the truncation $\mathsf{red}^{(\theta)}_{V,p,r}(\mathcal{M})$.

Combining Claim 2.10.8 and Claim 2.11.4, we deduce that we can truncate after reducing by a checker while changing the distribution by only a negligible amount.

Corollary 2.11.6. *Let $\mathcal{M} = w_1 N(\mu_1, I) + \cdots + w_k N(\mu_k, I)$ be a GMM. Let w^*, δ be some parameters and let $\theta \geq (\log(k/(w^*\delta)))^{(1+c)/2}$ for some positive constant $c > 0$. Let (V, p, r) be a checker where V has dimension at most $(\theta/10)^2$. Assume that for some i, we have $w_i \geq w^*$ and $\mathsf{chk}_{V,p,r-\theta}(\mu_i) = 1$. Then*

$$d_{\mathsf{TV}}\left(\mathsf{red}^{(\theta)}_{V,p,r}(\mathcal{M}), \mathsf{red}_{V,p,r}(\mathcal{M})\right) \leq 2^{-(\log(k/(w^*\delta)))^{1+0.1c}}.$$

Proof. Note that by Claim 2.10.8, for any $\mu \in \mathbb{R}^d$ we have

$$\Pr_{z \sim N(\mu, I)}[\|\mathsf{Proj}_V(z - \mu)\| \geq 0.5\theta] \leq 2^{-0.01\theta^2}.$$

In particular, for any μ_j that is not in the checker $(V, p, r+\theta)$, the probability that a sample from $N(\mu_j, I)$ lands in (V, p, r) is at most $2^{-0.01\theta^2}$. Also we assume that there is some i such that μ_i is in $(V, p, r-\theta)$ and for this component $N(\mu_i, I)$, the probability that a sample lands in (V, p, r) is at least $1 - 2^{-0.01\theta^2}$. Combining these observations gives the desired inequality. ∎

Note that we can simulate samples from $\mathsf{red}_{V,p,r}(\mathcal{M})$ (using samples from $\mathcal{M}$) and the above implies that this is essentially equivalent to simulating samples from $\mathsf{red}^{(\theta)}_{V,p,r}(\mathcal{M})$ since their TV distance is negligible. One more important claim that we will need is that if a checker contains the mean of one of the components, then by adjusting the radius slightly, we can find a truncation for which the resulting mixture is reasonable (recall Definition 2.10.5). This will allow us to apply Lemma 2.10.12 to find a new signal direction in $V^{\perp}$.

Claim 2.11.7. *Let $\mathcal{M} = w_1 N(\mu_1, I) + \cdots + w_k N(\mu_k, I)$ be a GMM. Let w^* be some parameter and let $\theta \geq (\log(k/w^*))^{(1+c)/2}$ for some positive constant $c > 0$. Assume that*

- $\max \|\mu_i - \mu_j\| \leq O((k/w^*)^2)$
- *The mixture $\mathcal{M}$ is $\theta \cdot (\log(k/w^*))^{0.1c/2}$-separated*
- *$w_i \geq w^*$ for all i*

Let V be a subspace of dimension at most $(\theta/10)^2$ and p be a point in V. Assume that for some i, the checker (V, p, θ) contains μ_i. Then for a random integer γ chosen from among $\{1, \ldots, \lceil 10^4 \log\log(k/w^*) \rceil\}$, with probability at least $1/2$, the mixture

$$\mathsf{red}^{(\theta)}_{V,p,\gamma\theta}(\mathcal{M})$$

is $0.9w^*$-reasonable.

Proof. For any real number l, define α_l as follows.

$$\alpha_l = \max_{\substack{\mathsf{chk}_{V,p,l}(\mu_i)=1 \\ \mathsf{chk}_{V,p,l}(\mu_j)=1}} \|\mathsf{Proj}_{V^{\perp}}(\mu_i - \mu_j)\| \ .$$

Now for any γ, the only way that

$$\mathsf{red}^{(\theta)}_{V,p,\gamma\theta}(\mathcal{M})$$

is not reasonable is if

$$\alpha^2_{(\gamma-1)\theta} \leq \alpha_{(\gamma+1)\theta} \, . \tag{2.16}$$

This is because by Claim 2.10.8, for all means μ_i that are in $(V, p, (\gamma-1)\theta)$, essentially all of the samples from $N(\mu_i, I)$ are contained in $(V, p, \gamma\theta)$. However, we now consider the sequence

$$\alpha_\theta, \alpha_{2\theta}, \ldots .$$

We can lower bound the first nonzero element of the sequence as follows. If there are two distinct means μ_i, μ_j both contained in $(V, p, \gamma\theta)$ for $\gamma \leq \lceil 10^4 \log\log(k/w^*) \rceil$, we must have

$$\alpha_{\gamma\theta} \geq \|\mu_i - \mu_j\| - \|\mathsf{Proj}_V(\mu_i - \mu_j)\| \geq \theta \cdot (\log(k/w^*))^{0.1c/2} - 2\gamma\theta \geq 2\,.$$

Thus, since α_l is upper bounded by $O((k/w^*)^2)$, the condition in (2.16) can fail for at most half of the choices of γ and we are done. ■

We will need one more preliminary result. It simply states that we can correctly cluster any sample with high probability if we are given a candidate set of means $S = \{\widetilde{\mu_1}, \ldots, \widetilde{\mu_t}\}$ that are all separated and such that there is one that is close to each true mean.

Claim 2.11.8. *Let $\mathbb{N}(\mu_1, I), \ldots, N(\mu_l.I)$ be some unknown Gaussians such that $\|\mu_i - \mu_j\| \geq s$ for all $i \neq j$ where $s = (\log l)^{(1+c)/2}$ and $c > 0$ is some constant. Assume we are given a candidate set of means $S = \{\widetilde{\mu_1}, \ldots, \widetilde{\mu_r}\}$ such that*

- *$r \leq l$*
- *For all $i \in [l]$, there is some $f(i) \in [r]$ such that $\left\|\widetilde{\mu_{f(i)}} - \mu_i\right\| \leq 0.1$*
- *For all distinct $j_1, j_2 \in [r]$, $\|\widetilde{\mu_{j_1}} - \widetilde{\mu_{j_2}}\| \geq 0.5s$*

Then there is an efficient algorithm that with probability at least $1 - 2^{-0.01s^2}$, given a sample from $N(\mu_i, I)$ for any $i \in [k]$, returns $f(i)$.

Proof. For all $j_1, j_2 \in [r]$ with $j_1 \neq j_2$, define

$$v_{j_1 j_2} = \frac{\widetilde{\mu_{j_1}} - \widetilde{\mu_{j_2}}}{\|\widetilde{\mu_{j_1}} - \widetilde{\mu_{j_2}}\|}\,.$$

Given a sample z, we find a j such that for all $j_1, j_2 \in [r]$, we have

$$|v_{j_1 j_2} \cdot (z - \widetilde{\mu_j})| \leq 0.1s\,.$$

If $z \sim N(\mu_i, I)$, then with probability at least $1 - 2^{-0.02s^2}$, setting $j = f(i)$ clearly satisfies the above. Also, by the assumption that $\|\widetilde{\mu_{j_1}} - \widetilde{\mu_{j_2}}\| \geq 0.5s$ for all distinct j_1, j_2, it is clear that with probability at least $1 - 2^{-0.02s^2}$ that no other $j' \in [r]$ will satisfy the above. This completes the proof. ■

2.11.3 Putting Everything Together

We can now put everything together and describe our complete clustering algorithm. At a high level there will be two phases.

In the first phase, we will keep track of a subspace V and point $p \in V$ and keep refining it. In particular, in each step we will add a dimension to V to get a higher dimensional subspace V' and we will compute a new point $p' \in V'$. Our goal in each step will be to maintain (roughly) the following properties

- For a certain radius $r = O((\log(k/w^*))^{(1+c)/2})$, the checker (V, p, r) always contains one of the true means μ_i
- If there are two means μ_i, μ_j in (V, p, r) with separation at least $0.1 \log^4(k/w_{\min})$, then with at least 0.1 probability, the refinement V', p' satisfies that (V', p', r') contains at most half as many means μ_i as (V, p, r') for some larger radius r'.

The first guarantee is not difficult to achieve as we simply need to check that there are enough samples in the checker. To achieve the second guarantee, we rely on Lemma 2.10.11 to find a signal direction and add this direction to V to get V'. We then argue that because this new direction is a signal direction, the set of true means in (V, p, r) can be split into two parts along this new direction and one of these parts will have at most half as many. The guarantees are stated formally in Lemma 2.11.9.

We will run the first phase sufficiently many times that with high probability, at some point, we must have V, p such that all means in (V, p, r) have separation at most $\log^4(k/w_{\min})$. We can actually test this condition (with some slack) using Lemma 2.11.10. If the test passes, in the second phase, we can simply use Lemma 2.10.12 to fully cluster the remaining sub-mixture obtained by reducing $\mathcal{M}$ by (V, p, r). This allows us to learn one of the components of the mixture. In fact, we can obtain a stronger guarantee and actually identify all of the samples from this component. See Lemma 2.11.11 for more details. Once we have done this, we can remove these samples and recurse on the remaining mixture. This completes the entire clustering algorithm. Below is an outline that summarizes our algorithm.

Algorithm 5 Complete Clustering Algorithm (Outline)

Initialize $V = 0$ (i.e. 0-dimensional subspace), $p = 0$

Set $r = O((\log(k/w_{\min}))^{(1+c)/2})$

for $j = 1, 2, \ldots, (\log(k/w_{\min}))^{1+0.1c}$ **do**

Refine $(V, p) \leftarrow (V', p')$ using Lemma 2.11.9

Test if means in (V, p, r) have max-separation at most $\log^4(k/w_{\min})$ using Lemma 2.11.10

Break if above test passes

Fully cluster samples in (V, p, r) and identify samples from one component $N(\mu_i, I)$ (see Lemma 2.11.11)

Remove samples from $N(\mu_i, I)$ and recurse on remaining

The next lemma formalizes the guarantees of the refinement step.

Lemma 2.11.9. *Let $\mathcal{M} = w_1 N(\mu_1, I) + \cdots + w_k N(\mu_k, I)$ be a GMM. Let $w^* > 0$ be a parameter and $c > 0$ be a positive constant. Assume that $\mathcal{M}$ is s-separated where $s = (\log(k/w^*))^{1/2+c}$ and satisfies $w_i \geq w^*$ for all i. Also assume that $\max \|\mu_i - \mu_j\| \leq O((k/w^*)^2)$.*

Assume that we are given a subspace V and a point $p \in V$ where the dimension of V is $a < (\log(k/w^*))^{1+0.1c}$. Assume that the checker $(V, p, 10(\log(k/w^*))^{(1+c)/2})$ contains some

μ_i. *Also let* C *be the number of indices* i *such that* μ_i *is contained in the checker*

$$\left(V, p, (\log(k/w^*))^2((\log(k/w^*))^{1+0.1c} - a)\right) .$$

Then there exists an algorithm that takes $n = \text{poly}((dk/w^*)^{1/c})$ *samples and* $\text{poly}(n)$ *runtime and returns a subspace* V' *and a point* $p' \in V'$ *with the following properties*

- V' *has dimension* $a + 1$ *and is obtained by adding one orthogonal direction to* V
- *The checker*

  $$\left(V', p', (\log(k/w^*))^2((\log(k/w^*))^{1+0.1c} - a - 1)\right)$$

  contains at most C *of the* μ_i.
- *With high probability, the checker* $(V', p', 10(\log(k/w^*))^{(1+c)/2})$ *contains some* μ_i
- *If there are two* μ_{i_1}, μ_{i_2} *contained in the checker* $(V, p, (\log(k/w^*))^{(1+1.1c)/2})$ *such that*

  $$\|\mu_{i_1} - \mu_{i_2}\| \geq 0.1(\log(k/w^*))^4$$

  then with probability at least 0.1*, the checker*

  $$\left(V', p', (\log(k/w^*))^2((\log(k/w^*))^{1+0.1c} - a - 1)\right)$$

  contains at most $C/2$ *of the* μ_i.

Proof. We first focus on the last guarantee as it will not be difficult to maintain the first three guarantees. Let $\theta = (\log(k/w^*))^{(1+c)/2}$ and $\beta = (\log(k/w^*))^{(1+1.1c)/2}$. Choose some γ randomly from $\{1, \ldots, \lceil 10^4 \log\log(k/w^*)\rceil\}$. Now by Claim 2.11.7, with probability at least $1/2$, the mixture

$$\mathsf{red}^{(\theta)}_{V,p,\beta+\gamma\theta}(\mathcal{M})$$

is $0.9w^*$-reasonable. Also note that the two components μ_{i_1} and μ_{i_2} promised in the last

condition must satisfy

$$\|\mathsf{Proj}_{V^\perp}(\mu_{i_2}) - \mathsf{Proj}_{V^\perp}(\mu_{i_1})\| \geq 0.09(\log(k/w^*))^4 .$$

and furthermore, their respective mixing weights in $\mathsf{red}^{(\theta)}_{V,p,\beta+\gamma\theta}(\mathcal{M})$ are at least $0.9w^*$. Now by Corollary 2.11.6 we can, with high probability, simulate a polynomial number of samples from $\mathsf{red}^{(\theta)}_{V,p,\beta+\gamma\theta}(\mathcal{M})$ by taking samples from $\mathsf{red}_{V,p,\beta+\gamma\theta}(\mathcal{M})$ (which we can simulate using samples from $\mathcal{M}$). Also note that since the checker

$$(V, p, 10(\log(k/w^*))^{(1+c)/2})$$

contains some μ_i, samples from $\mathcal{M}$ are contained in the checker $(V, p, \beta+\gamma\theta)$ with probability at least $0.9w^*$. Thus we can apply Lemma 2.10.11 and with high probability we can find a signal direction. Note this signal direction is a unit vector $v \in V^\perp$ such that it is a $(0.4w^*, 0.04(\log(k/w^*))^4)$-signal direction for the distribution $\mathsf{red}^{(\theta)}_{V,p,\beta+\gamma\theta}(\mathcal{M})$. Note that we can check this last condition so if we repeat the above polynomially many times (for random choices of γ), we can guarantee that we have found such a signal direction.

We let $V' = V + v$ (i.e. V' is obtained by adding v to the span of V). Note that by the assumption that v is a signal direction, we can find a real number t such that there must be two components μ_i and μ_j that are relevant in $\mathsf{red}^{(\theta)}_{V,p,\beta+\gamma\theta}(\mathcal{M})$ such that

$$t - v \cdot \mu_i \geq 0.01`(\log(k/w^*))^4$$
$$v \cdot \mu_j - t \geq 0.01(\log(k/w^*))^4 .$$

Now to find the new center p', we do the following. Draw a fresh set of $\mathrm{poly}(dk/w^*)$ samples from $\mathcal{M}$. For each sample, we keep it if and only if it is contained in the checker $(V, p, \beta + (\gamma+2)\theta)$. We say a sample z is good if at least $0.9w^*$-fraction of other samples z' satisfy

$$\|\mathsf{Proj}_{V'}(z') - \mathsf{Proj}_{V'}(z)\| \leq (\log(k/w^*))^{(1+c)/2} .$$

Note that with high probability, we will be able to find two samples z_1, z_2 that are both good and such that $|v \cdot z_1 - v \cdot z_2| \geq 0.01(\log(k/w^*))^4$ (this will happen as long as we take a sample from μ_i and a sample from μ_j for the two separated components promised in the previous paragraph). Now with $1/2$ probability take $p' = z_1$ and with $1/2$ probability take $p' = z_2$. Note that the checkers

$$\left(V', z_1, (\log(k/w^*))^2((\log(k/w^*))^{1+0.1c} - a - 1)\right)$$
$$\left(V', z_2, (\log(k/w^*))^2((\log(k/w^*))^{1+0.1c} - a - 1)\right)$$

are disjoint since z_1, z_2 are sufficiently separated along direction v. Also,

$$\left\|\mathsf{Proj}_V(z_1) - p\right\|, \left\|\mathsf{Proj}_V(z_2) - p\right\| \leq \beta + (\gamma + 2)\theta$$

and thus both of the above checkers are contained in the checker

$$\left(V, p, (\log(k/w^*))^2((\log(k/w^*))^{1+0.1c} - a)\right) .$$

Thus, with $1/2$ probability, the number of means contained in the new checker is at most $C/2$. It remains to verify the first three conditions. The first is trivial. The second is also trivial. To see why the third is true, note that since z_1 is good, with high probability, there must be some μ_{i_1} such that $\|\mathsf{Proj}_{V'}(\mu_{i_1}) - \mathsf{Proj}_{V'}(z_1)\| \leq 2(\log(k/w^*))^{(1+c)/2}$ by Claim 2.10.8 and similar for z_2. This immediately implies the third condition. ■

Next, we show that we can actually check the termination condition (with some slack), that the maximum separation of any two means in (V, p, r) is at most $O(\log^4(k/w^*))$ where $r = O((\log(k/w^*))^{(1+c)/2})$.

Lemma 2.11.10. *Let $\mathcal{M} = w_1 N(\mu_1, I) + \cdots + w_k N(\mu_k, I)$ be a GMM. Let $w^* > 0$ be a parameter and $c > 0$ be a positive constant. Assume that $\mathcal{M}$ is s-separated where $s = (\log(k/w^*))^{1/2+c}$ and satisfies $w_i \geq w^*$ for all i. Also assume that $\|\mu_i - \mu_j\| \leq O((k/w^*)^2)$ for all i, j. Say that we are given a subspace V and a point $p \in V$ where the dimension of V*

is $a < (\log(k/w^*))^{1+0.1c}$. *Assume that the checker* $(V, p, 10(\log(k/w^*))^{(1+c)/2})$ *contains some* μ_i. *Then there is an algorithm that takes* $n = \mathrm{poly}((dk/w^*)^{1/c})$ *samples and runs in* $\mathrm{poly}(n)$ *time, and with high probability,*

- *Outputs* REJECT *if there are* i, j *such that* μ_i *and* μ_j *are both contained in the checker*

$$(V, p, 20(\log(k/w^*))^{(1+c)/2})$$

  and that $\|\mu_i - \mu_j\| \geq (\log(k/w^*))^4$.

- *Outputs* ACCEPT *if for all* i, j *such that* μ_i *and* μ_j *are both contained in the checker*

$$(V, p, (\log(k/w^*))^{(1+1.1c)/2})$$

  we have that $\|\mu_i - \mu_j\| \leq 0.1(\log(k/w^*))^4$.

Proof. Let $\theta = (\log(k/w^*))^{(1+c)/2}$. We consider $\mathsf{red}^{(\theta)}_{V,p,(30+\gamma)\theta}(\mathcal{M})$ for all

$$\gamma = \{1, \ldots, \lceil 10^4 \log\log(k/w^*) \rceil\}.$$

We simulate samples from $\mathsf{red}^{(\theta)}_{V,p,(30+\gamma)\theta}(\mathcal{M})$ using samples from $\mathsf{red}_{V,p,(30+\gamma)\theta}(\mathcal{M})$ (which can be simulated using samples from $\mathcal{M}$), which by Corollary 2.11.6 is equivalent with high probability for polynomially many samples. We attempt to find a signal direction in this reduced mixture for each γ using Lemma 2.10.11. If for any γ, we find a direction that we can check is a $(0.4w^*, 0.4(\log(k/w^*))^4)$-signal direction using Claim 2.10.7, then we output REJECT. Otherwise we output ACCEPT. To see why this works, first note that clearly we will always output ACCEPT when we are supposed to accept. Now to see that we reject when we are supposed to reject, consider the choice of γ guaranteed by Claim 2.11.7 for which the mixture $\mathsf{red}^{(\theta)}_{V,p,(30+\gamma)\theta}(\mathcal{M})$ is reasonable. For this choice of γ, by the guarantees of Lemma 2.10.11, we will find a $(0.4w^*, 0.4(\log(k/w^*))^4)$-signal direction with high probability and thus we will REJECT. ■

Finally, we show how to do the final step where we do full clustering and isolate samples from a single component.

Lemma 2.11.11. *Let $\mathcal{M} = w_1 N(\mu_1, I) + \cdots + w_k N(\mu_k, I)$ be a GMM. Let $w^* > 0$ be a parameter and $c > 0$ be a positive constant. Assume that $\mathcal{M}$ is s-separated where $s = (\log(k/w^*))^{1/2+c}$ and satisfies $w_i \geq w^*$ for all i. Assume that we are given a subspace V and a point $p \in V$ where the dimension of V is $a < (\log(k/w^*))^{1+0.1c}$. Assume that the checker $(V, p, 10(\log(k/w^*))^{(1+c)/2})$ contains some μ_i. Also assume that for all i, j such that μ_i and μ_j are both contained in the checker $(V, p, 20(\log(k/w^*))^{(1+c)/2})$, we have that $\|\mu_i - \mu_j\| \leq (\log(k/w^*))^4$. Then there is an algorithm that takes $n = \mathrm{poly}((dk/w^*)^{1/c})$ samples and runs in $\mathrm{poly}(n)$ time, and with high probability, returns a test (which we denote test) that can be computed in $\mathrm{poly}(dk)$ time and has the following properties*

- *There is some i such that for a random sample $z \sim N(\mu_i, I)$, with high probability $\mathsf{test}(z) = \mathsf{accept}$*
- *For all other $i' \neq i$, for a random sample $z \sim N(\mu_i, I)$, with high probability $\mathsf{test}(z) = \mathsf{reject}$*

Proof. Let $\theta = (\log(k/w^*))^{(1+c)/2}$. Consider the truncated reduced mixture

$$\mathsf{red}^{(\theta)}_{V,p,19\theta}(\mathcal{M})\,.$$

By Corollary 2.11.6, we can simulate a polynomial number of samples from this mixture by instead taking samples from $\mathsf{red}_{V,p,19\theta}(\mathcal{M})$ (which we can simulate using samples from $\mathcal{M}$). Now, by assumption, all relevant components μ_i, μ_j in $\mathsf{red}^{(\theta)}_{V,p,19\theta}(\mathcal{M})$ must have

$$0.9(\log(k/w^*))^{1/2+c} \leq \|\mathsf{Proj}_{V^\perp}(\mu_i) - \mathsf{Proj}_{V^\perp}(\mu_j)\| \leq (\log(k/w^*))^4\,.$$

Thus, we can apply Lemma 2.10.12 to this mixture with parameter $w^* = (w^*/k)^{10}$. With high probability, we obtain a list of candidate means $\{\widetilde{\mu_1}, \ldots, \widetilde{\mu_r}\}$ such that $r \leq k$, all of the candidate means are $0.4(\log(k/w^*))^{1/2+c}$-separated and for all $i \in [k]$ such that μ_i is

contained in the checker $(V, p, 18\theta)$, there is some $f(i) \in [r]$ such that

$$\left\|\mathsf{Proj}_{V^\perp}(\widetilde{\mu_{f(i)}}) - \mathsf{Proj}_{V^\perp}(\mu_i)\right\| \leq 0.1\,.$$

Note the last condition is because any component whose mean μ_i is contained in the checker $(V, p, 18\theta)$ must be almost entirely contained in the checker $(V, p, 19\theta)$ and thus will have significant mixing weight in $\mathsf{red}^{(\theta)}_{V,p,19\theta}(\mathcal{M})$.

Now we draw $\mathrm{poly}(k/w^*)$ fresh samples from $\mathcal{M}$ and restrict to those that are contained in the checker $(V, p, 17\theta)$. With high probability, this is equivalent to drawing samples from the mixture

$$\mathsf{red}^{(\theta)}_{V,p,17\theta}(\mathcal{M})\,.$$

All of the relevant components in this mixture have mean μ_i that is contained in the checker $(V, p, 18\theta)$. Thus, we can apply Claim 2.11.8 to fully cluster these samples with high probability. Recall that by assumption, there is some true mean μ_i contained in the checker $(V, p, 10\theta)$. Thus, we can find one of the clusters with weight at least $0.5w^*$ and such that at least half of the points in the cluster are contained in the checker $(V, p, 11\theta)$. Say that this cluster corresponds to μ_j and estimated mean $\widetilde{\mu_{f(j)}}$. This cluster must be almost entirely contained in the checker $(V, p, 17\theta)$. To isolate exactly the points from $N(\mu_j, I)$ with high probability, we can first restrict to points contained in the checker $(V, p, 17\theta)$ and then apply Claim 2.11.8 and restrict to points for which the output is $f(j)$. Thus, we have an efficiently computable test that, with high probability, isolates exactly the points from the cluster $N(\mu_j, I)$ and we are done. ■

Now can complete the proof of our main theorem, Theorem 2.2.5, for learning clusterable mixtures of Gaussians.

Proof of Theorem 2.2.5. First we apply the reductions in Section 2.3.3 to ensure that $d \leq k$ and $\|\mu_i - \mu_j\| \leq O((k/w_{\min})^2)$ for all i, j. Now we apply the algorithm in Lemma 2.11.9 for $(\log(k/w_{\min}))^{1+0.1c}$ times (where we set $w^* = w_{\min}$). Note that the lemma works with

a trivial initialization where V is 0-dimensional and p is just 0. Using the guarantees of Lemma 2.11.9, with high probability at some point, we will get V, p (where V has dimension $(\log(k/w_{\min}))^{1+0.1c}$) which satisfy that

- The checker $(V, p, 10(\log(k/w_{\min}))^{(1+c)/2})$ contains at least one of the μ_i
- For any μ_i, μ_j contained in the checker $(V, p, (\log(k/w_{\min}))^{(1+1.1c)/2})$, we have

$$\|\mu_i - \mu_j\| \leq 0.1(\log(k/w_{\min}))^4 .$$

Thus, when we run Lemma 2.11.10 to check this pair V, p, we will ACCEPT this pair of V, p and move to the final step. Note that Lemma 2.11.10 also implies that for any pair V, p that we ACCEPT and move to the final step, we have the slightly weaker guarantees that

- The checker $(V, p, 10(\log(k/w_{\min}))^{(1+c)/2})$ contains at least one of the μ_i
- For any μ_i, μ_j contained in the checker $(V, p, 20(\log(k/w_{\min}))^{(1+c)/2})$, we have

$$\|\mu_i - \mu_j\| \leq (\log(k/w_{\min}))^4$$

These weaker guarantees still suffice to apply the algorithm in Lemma 2.11.11 to isolate one of the components of $\mathcal{M}$ (again with $w^* = w_{\min}$). With high probability, we can take poly$(k/w_{\min})$ samples from this component and estimate its mean and mixing weight to within α (since $\alpha = (k/w_{\min})^{O(1)}$). We can also remove all of the samples from this component from the mixture and recurse on a mixture of $k-1$ Gaussians. Note that we can do this because our test for checking whether a sample belongs to the removed component succeeds with high probability (meaning its failure probability is smaller than any inverse polynomial) and the recursive call only takes polynomially many samples. Thus overall the algorithm succeeds with high probability and we are done. ■

The clustering guarantee in Corollary 2.2.6 follows as an immediate consequence of Theorem 2.2.5.

Proof of Corollary 2.2.6. Let the estimated means computed by Theorem 2.2.5 for $\alpha = (w_{\min}/k)^{10}$ be $\widetilde{\mu_1}, \ldots, \widetilde{\mu_k}$. Now for all $j_1, j_2 \in [k]$ with $j_1 \neq j_2$, let

$$v_{j_1 j_2} = \frac{\widetilde{\mu_{j_1}} - \widetilde{\mu_{j_2}}}{\|\widetilde{\mu_{j_1}} - \widetilde{\mu_{j_2}}\|} .$$

Now given a sample z from $\mathcal{M}$, we compute the index j such that for all j_1, j_2, we have

$$|v_{j_1 j_2} \cdot (\widetilde{\mu_j} - z)| \leq (\log(k/w_{\min}))^{(1+c)/2} .$$

Note that by the guarantees of Theorem 2.2.5, there is a permutation π such that $\left\|\widetilde{\mu_{\pi(i)}} - \mu_i\right\| \leq \alpha$ for all i. If z is a sample from $N(\mu_i, I)$, then with high probability the unique index j that satisfies the above is exactly $j = \pi(i)$ and thus, we recover the ground truth clustering with high probability. ■

2.12 Omitted Proofs from Section 2.3.3

Here we explain the reductions for Section 2.3.3. When explaining the reductions, we will work with a mixture of Poincare distributions

$$\mathcal{M} = w_1 \mathcal{D}(\mu_1) + \cdots + w_k \mathcal{D}(\mu_k)$$

but it will be obvious that these reductions also work for Gaussians.

Reducing to all Means Polynomially Bounded

By Fact 2.3.7, with $1 - 2^{-10(d+k)/w_{\min}}$ probability, a sample $z \sim \mathcal{D}(\mu_i)$ satisfies

$$\|z - \mu_i\| \leq 10^4 \cdot (d+k)/w_{\min} .$$

Now, we look for a pair of samples z, z' such that

$$\|z - z'\| \geq 10^6((d+k)/w_{\min})^2 .$$

Note that with high probability, such a pair exists if

$$\|\mu_i - \mu_j\| \geq 1.1 \cdot 10^6((d+k)/w_{\min})^2$$

for some i, j. Let v be the unit vector in the direction $z - z'$ and let μ_i, μ_j be the components that z, z' were drawn from. We must have that

$$|\langle v, \mu_i - \mu_j \rangle| \geq 0.99 \cdot 10^6((d+k)/w_{\min})^2 .$$

Now imagine projecting all of the samples onto the direction v. Note that by Fact 2.3.7, with high probability all of the samples from a given component will lie in an interval of width at most $10^2(d+k)/w_{\min}$ after projecting onto the direction v. Thus, there must be an empty interval of width at least $10^3(d+k)/w_{\min}$ between the projections of z and z'. This means that no μ_i has projection in this interval and we can subdivide the mixture into two submixtures with strictly fewer components by cutting via a hyperplane normal to v through the middle of this interval. Note that for each of the two submixtures, a sample will be on the wrong side of this cut with exponentially small probability but our algorithm will only use polynomially many samples. Thus, we can simply run our learning algorithm on each submixture.

We have reduced to $\|\mu_i - \mu_j\| \leq O(((d+k)/w_{\min})^2)$. Now we can simply estimate the mean of the distribution $\mathcal{M}$ and subtract it out. Note that we have proved the reduction with a bound of $O(((d+k)/w_{\min})^2)$. To reduce to $O((k/w_{\min})^2)$, we can apply the reduction in the next section, which will ensure that $d \leq k$ and then we can apply this reduction again.

Reducing the Dimension

Next, we show that we can reduce to the case when $d \leq k$. We can estimate the empricial covariance of $\mathcal{D}$, say $\Sigma_{\mathcal{D}}$. We can also estimate the empirical covariance of $\mathcal{M}$, which is

$$\Sigma_{\mathcal{M}} = w_1(\mu_1 \otimes \mu_1) + \cdots + w_k(\mu_k \otimes \mu_k) + \Sigma_{\mathcal{D}}\,.$$

Note that since all means $\|\mu_i\|$ are polynomially bounded by the previous reduction, we can obtain estimates for $\Sigma_{\mathcal{M}}$ and $\Sigma_{\mathcal{D}}$ that are accurate to within ϵ in Frobenius norm for any inverse polynomial ϵ. Thus, we have an estimate $\widetilde{M}$ such that

$$\left\|\widetilde{M} - (w_1(\mu_1 \otimes \mu_1) + \cdots + w_k(\mu_k \otimes \mu_k))\right\|_F \leq 2\epsilon$$

for any desired inverse polynomial ϵ. We can then take V to be the subspace spanned by the top k principal components of $\widetilde{M}$. Using the above, we can ensure that all of the μ_i are within distance $0.1\delta \cdot (w_{\min}/k)^{10}$ of the subspace V. Let $\mathcal{D}_{\mathsf{Proj},V}$ be the projection of the distribution $\mathcal{D}$ onto the subspace V. If we project all of our samples onto the subspace V, we would have a mixture of translated copies of $\mathcal{D}_{\mathsf{Proj},V}$ where the separations are decreased by at most $0.2\delta \cdot (w_{\min}/k)^{10}$. Thus, we have reduced to a k-dimensional problem.

Chapter 3

Robustly Learning Mixtures of Gaussians

3.1 Overview

In this chapter, we consider the problem of robustly learning a mixture of Gaussians. Formally, there is some unknown GMM in d dimensions

$$\mathcal{M} = w_1 N(\mu_1, \Sigma_1) + \cdots + w_k N(\mu_k, \Sigma_k)\,.$$

We receive samples from $\mathcal{M}$ except an ϵ-fraction of them may be corrupted, possibly adversarially. Our goal is to learn the parameters $w_1, \ldots, w_k, \mu_1, \ldots, \mu_k, \Sigma_1, \ldots, \Sigma_k$. We treat k as a constant and focus on the dependence on d and ϵ. In particular, we ask for sample complexity and runtime $\mathrm{poly}(d/\epsilon)$ and accuracy $\mathrm{poly}(\epsilon)$ and independent of d.

3.1.1 Key Techniques

We now give an overview of our techniques. For simplicity, assume that the mixture is in isotropic position. First, we have the unknown parameters of the mixture. Let

$$\mathcal{M} = w_1 N(\mu_1, I + \Sigma_1) + \cdots + w_k N(\mu_k, I + \Sigma_k)$$

where w_i are the mixing weights and μ_i and $I + \Sigma_i$ denote the mean and covariance of the ith component – note that we have re-parametrized the covariances in a way that will be useful later on. Second, we have the indeterminates we would like to solve for. These will be denoted $\widetilde{\mu}_i$ and $\widetilde{\Sigma}_i$ and our intention is for these to be the means and covariances of a hypothetical mixture of Gaussians [1]. We will also guess the mixing weights of the hypothetical mixture $\widetilde{w}_i$. Finally, we have a d-dimensional vector $X = (X_1, \ldots, X_d)$ of formal variables and one auxiliary formal variable y. These will mostly be used to help us organize everything in a convenient way. Roughly, we would like to solve for $\widetilde{\mu}_i$ and $\widetilde{\Sigma}_i$ so that the hypothetical mixture

$$\widetilde{\mathcal{M}} = \widetilde{w_1} N(\widetilde{\mu}_1, I + \widetilde{\Sigma_1}) + \cdots + \widetilde{w_k} N(\widetilde{\mu_k}, \widetilde{I + \Sigma_k})$$

is close to $\mathcal{M}$ on the family of test functions (which will be low-degree multivariate Hermite polynomials). It turns out that this amounts to solving a polynomial system for the indeterminates.

Now we explain in more detail how to actually reason about and solve the polynomial system. It will be useful to work with the following generating functions. First let

$$F(y) = \sum_{i=1}^{k} w_i e^{\mu_i(X)y + \frac{1}{2}\Sigma_i(X)y^2}$$

[1] Technically our setup is slightly different in that we solve for vectors $u_1, \ldots, u_k$ and $v_1, \ldots, v_k$ that are supposed to form an orthonormal basis for the span of the $\{\widetilde{\mu}_i\}$ and $\{\widetilde{\Sigma_i}\}$ respectively. Regardless, the argument is conceptually the same.

Here we have used the notation that $\mu_i(X)$ denotes the inner product of μ_i with the d-dimensional vector X and that $\Sigma_i(X)$ denotes the quadratic form of X on Σ_i. Second let

$$\widetilde{F}(y) = \sum_{i=1}^{k} \widetilde{w}_i e^{\widetilde{\mu_i}(X)y + \frac{1}{2}\widetilde{\Sigma_i}(X)y^2}$$

As is familiar from elementary combinatorics we can tease out important properties of the generating function by applying carefully chosen operators that involve differentiation. This requires a lot more bookkeeping than usual because there are unknown parameters of the mixture, indeterminates and formal variables. But it turns out that there are simple differential operators we can apply which can isolate components. To gain some intuition, consider the operator

$$\mathcal{D}_i = \partial_y - (\mu_i(X) + \Sigma_i(X)y)\,.$$

Note that

$$\mathcal{D}_i \left(e^{\mu_i(X)y + \frac{1}{2}\Sigma_i(X)y^2} \right) = 0\,,$$

in other words, this operator annihilates the i^{th} component. Thus, by composing such operators, we can annihilate all but one of the components in F [2]. On the other hand, note that applying differential operators is just a rearrangement of the polynomials that show up in the infinite sum representation of the generating function but using differential operators and generating functions in exponential form gives a particularly convenient way to derive useful expressions that would otherwise be extremely complex to write down.

Ultimately we derive a symbolic identity

$$\widetilde{w_k} \prod_{i=1}^{k} (\widetilde{\Sigma_k}(X) - \Sigma_i(X))^{2^{i-1}} \prod_{i=1}^{k-1} (\widetilde{\Sigma_k}(X) - \widetilde{\Sigma}_i(X))^{2^{k+i-1}} = \sum_{i=1}^{m} P_i(X)(\widetilde{h}_i(X) - h_i(X)) \quad (3.1)$$

where m is a function of k. In the above, the $h_i(X)$'s are the expectations of the multivariate Hermite polynomials for the true mixture $\mathcal{M}$ and the $\widetilde{h}_i$'s are the expectations of the multi-

[2] There are some additional details because applying $\mathcal{D}_i$ to a different component creates an extra polynomial factor in front. Section 3.4.3 shows how to deal with this complication.

variate Hermite polynomials for the hypothetical mixture $\widetilde{\mathcal{M}}$. A more detailed explanation of Hermite polynomials is given in Section 3.3 but for now we may simply think of them as modified moments. The reason that we use Hermite polynomials instead of standard moments is that they can be robustly estimated without losing dimension-dependent factors (see e.g. [65]).

The above identity (when combined with a few others of a similar form) allows us to deduce robust identifiability. Roughly, this is because if we have a mixture $\widetilde{\mathcal{M}}$ with means $\widetilde{\mu}_i$ and covariances $I + \widetilde{\Sigma}_i$ that don't match those of $\mathcal{M}$, then the LHS of (3.1) will be bounded away from 0, implying that some term on the RHS must also be bounded away from 0. This means that there must be some $i \leq m = O_k(1)$ such that $h_i(X)$ and $\widetilde{h_i(X)}$ are substantially different – i.e. there is some test function that is a low-degree multivariate Hermite polynomial that distinguishes the two mixtures.

Robust identifiability alone does not give us a polynomial time learning algorithm. However, it turns out that we can use SOS to obtain a polynomial time learning algorithm from the argument for robust identifiability i.e. we essentially get the learning algorithm "for free". The key point is that the $h_i(X)$ can be estimated using our samples and the coefficients of the $\widetilde{h}_i$ are explicit polynomials in the indeterminates that we can write down. Also the P_i's are polynomials in *everything*: the unknown parameters, the indeterminates and the formal variables (except for y). To set up an SOS system, we obtain robust estimates $\overline{h_i}$ for the expectations of the Hermite polynomials $h_i(X)$ for the true mixture that we can compute from existing techniques in the literature. We then enforce that the expectations of the Hermite polynomials for the hypothetical mixture $\widetilde{h}_i(X)$ are close to these robust estimates where closeness is defined in terms of the distance between their coefficient vectors.

It is not immediately clear why the expression in (3.1) ought to be useful for solving the SOS system that we set up. After all, we cannot explicitly compute it because it depends on things we do not have, like the true parameters, the μ_i's and Σ_i's. However the sum-of-squares relaxation enforces that the pseudo-expectation operator assigns values to polynomials in the indeterminates in a way that behaves like an actual distribution on solutions when we are

evaluating certain types of low degree expressions that contain the one above. So even though we do not know the actual polynomials in the identity, they exist and the fact that they are enforced is enough to ensure that we can estimate the covariance of the kth component We stress that this is just the high-level picture and many more details are needed to fill it in.

Using these techniques, we come to the main result of our paper, which is a polynomial-time algorithm for robustly learning the parameters of a high-dimensional mixture of a constant number of Gaussians. Our main theorem is (informally) stated below. A formal statement can be found in Theorem 3.8.3.

Theorem 3.1.1. *Let k be a constant. Let $\mathcal{M} = w_1 G_1 + \cdots + w_k G_k$ be a mixture of Gaussians in $\mathbb{R}^d$ whose components are non-degenerate and such that the mixing weights have bounded fractionality and TV distances between different components are lower bounded. (Both of these bounds can be any function of k). Given $n = \mathrm{poly}(d/\epsilon)$ samples from $\mathcal{M}$ that are ϵ-corrupted, there is an algorithm that runs in time $\mathrm{poly}(n)$ and with high probability outputs a set of mixing weights $\widetilde{w_1}, \ldots, \widetilde{w_k}$ and components $\widetilde{G_1}, \ldots, \widetilde{G_k}$ that are $\mathrm{poly}(\epsilon)$-close to the true components (up to some permutation).*

Discussion of Assumptions and Later Improvements

Bounded Fractionality of Mixing Weights: The assumption of bounded fractionality stems from an issue in previous work [39] about learning clusterable mixtures of Gaussians i.e. when the components have essentially no overlap. We use some of their subroutines in our algorithm for clustering the mixture into submixtures where the components are not too far apart from each other (see Section 3.6). While [39] claims to handle general mixing weights, the analysis of the algorithm in [39] is only done in detail for the case of uniform mixing weights and the argument in Appendix C for reducing from general mixing weights to uniform mixing weights does not work. The authors of [39] along with the authors of [14] were able to fix these arguments to handle general mixing weights in [11]. The modified proof uses essentially the same techniques. Plugging this improved clustering result into our analysis, instead of using the weaker guarantees for uniform mixing weights, we are able

to straightforwardly remove the bounded fractionality assumption. See Section 3.6.3 for a formal statement and explanation.

Separation Assumption: While our original proof required constant separation between components, we show in a follow-up paper, [74], that this assumption can be replaced with a separation of $\epsilon^{\Omega_k(1)}$ (which is a necessary assumption for parameter learning) using standard tricks (see Theorem 9.2 in [74]). This argument works independently of the improvement for the mixing weight assumption. See Section 3.8.1 for a more formal statement and explanation.

Non-degeneracy of Components: This assumption was included so there are no bit complexity issues. In fact, dealing with potentially degenerate covariance matrices requires a formal specification of the model of computation.

To make the timeline of events clear, we stated our original result above. However, plugging in these improvements (for removing the bounded fractionality and separation assumption), we obtain an improved result, stated below. The formal statement can be found in Theorem 3.8.6. We emphasize that these modifications are *completely independent* of our main contributions, but are rather tools that we employ to reduce to the case where the components are not too far from each other. We reduce to this case by running a preprocessing step where we cluster the mixture into such submixtures. All of the assumptions stem from the clustering step, which is done via modifications to the techniques for learning fully clusterable mixtures in [39, 13].

Theorem 3.1.2. *Let k be a constant. Let $\mathcal{M} = w_1 G_1 + \cdots + w_k G_k$ be a mixture of Gaussians in $\mathbb{R}^d$ whose components are non-degenerate and such that the mixing weights are lower bounded by some function of k. Also assume that the TV distances between components are at least $\epsilon^{\Omega_k(1)}$. Then given $n = \mathrm{poly}(d/\epsilon)$ samples from $\mathcal{M}$ that are ϵ-corrupted, there is an algorithm that runs in time $\mathrm{poly}(n)$ and with high probability outputs a set of mixing weights $\widetilde{w_1}, \ldots, \widetilde{w_k}$ and components $\widetilde{G_1}, \ldots, \widetilde{G_k}$ that are $\mathrm{poly}(\epsilon)$-close to the true components (up to*

some permutation).

Remark. *The improved clustering arguments of [11] are able to get polynomial dependence on the minimum mixing weight instead of exponential dependence so they are actually able to deal with mixing weights that are $\epsilon^{\Omega_k(1)}$. This improvement also plugs into our result as well.*

3.1.2 Proof Overview

The proof of our main theorem can be broken down into several steps. We first present our main contribution, an algorithm for learning mixtures of Gaussians when no pair of components are too far apart. We introduce the necessary generating function machinery in Section 3.3 and then present our algorithm in Sections 3.4 and 3.5. Specifically, in Section 3.4 we show how to learn the parameters once we have estimates for the Hermite polynomials of the true mixture. And in Section 3.5, we show how to robustly estimate the Hermite polynomials, using similar techniques to [65].

To complete our full algorithm for learning general mixtures of Gaussians, we combine our aforementioned results with a clustering algorithm similar to [39]. Combining these algorithms, we prove that our algorithm outputs a mixture that is close to the true mixture in TV distance. This is done in Sections 3.6 and 3.7. We then prove identifiability in Section 3.8, implying that our algorithm actually learns the true parameters.

3.1.3 Concurrent and Subsequent Work

There are three main pieces of work that we discuss. The first by Bakshi et. al [11], was independent and concurrent to this one. The next is a subsequent work by Bakshi et. al [12] that improves their earlier result but also borrows techniques from our paper. We also discuss our follow-up work [74] that was after [11] but before [12] (and the contributions are essentially disjoint).

Bakshi et. al in [11] obtain a result that is similar to our main result Theorem 3.1.1, but using rather different techniques. There are a few key differences, which we discuss

below. We learn the mixture to accuracy $\epsilon^{\Omega_k(1)}$ while their result only achieves accuracy $(1/\log(1/\epsilon))^{\Omega_k(1)}$, an exponentially worse guarantee. Also, our result solves parameter learning – i.e. we estimate the parameters of the true mixture – while their algorithm solves proper density estimation – i.e. it outputs a mixture of k Gaussians that is close to the true density. Their algorithm does not need any lower bound on the mixing weights or on the pairwise separation of the components. In fact lower bounds on these quantities are necessary for parameter learning. However, our original result makes an even stronger assumption about the bounded fractionality of the mixing weights and pairwise separation.

Subsequently in [12], Bakshi et. al improve their earlier result to achieve parameter learning and accuracy $\epsilon^{\Omega_k(1)}$. In fact the analysis of their new parameter learning algorithm relies crucially on the robust identifiability result from our paper (see Section 9 in [12]). Thus, all known algorithms for robust parameter learning go through our machinery and robust identifiability. Compared to our result, the main improvement in [12] is an improved, polynomial dependence on the minimum mixing weight.

Our follow-up work [74] improves the results here in a different direction. We are able to obtain an algorithm that solves density estimation to accuracy $\widetilde{O}(\epsilon)$ (instead of $\epsilon^{\Omega_k(1)}$). Note that parameter learning to this accuracy is information-theoretically impossible. The work in [74] does need a stronger assumption that the components have variances in all directions lower and upper bounded by a constant and the learning algorithm is improper, outputting a mixture of polynomials times Gaussians, instead of just a mixture of Gaussians. The main relevance of [74] to this paper is that as a first step, [74] shows how to improve the separation assumption in Theorem 3.1.1 from some constant to $\epsilon^{\Omega_k(1)}$.

3.2 Preliminaries

3.2.1 The Model

We use $N(\mu, \Sigma)$ to denote a Gaussian with mean μ and covariance Σ. We use $d_{\mathsf{TV}}(\mathcal{D}, \mathcal{D}')$ to denote the total variation distance between two distributions $\mathcal{D}, \mathcal{D}'$. We begin by formally

defining the problem that we will study. First we define the contamination model. This is a standard definition from robust learning (see e.g. [39]).

Definition 3.2.1 (Strong Contamination Model)**.** *We say that a set of vectors $Y_1, \ldots, Y_n$ is an ϵ-corrupted sample from a distribution $\mathcal{D}$ over $\mathbb{R}^d$ if it is generated as follows. First $X_1, \ldots, X_n$ are sampled i.i.d. from $\mathcal{D}$. Then a (malicious, computationally unbounded) adversary observes $X_1, \ldots, X_n$ and replaces up to ϵn of them with any vectors it chooses. The adversary may then reorder the vectors arbitrarily and output them as $Y_1, \ldots, Y_n$*

In this paper, we study the following problem. There is an unknown mixture of Gaussians

$$\mathcal{M} = w_1 G_1 + \cdots + w_k G_k$$

where $G_i = N(\mu_i, \Sigma_i)$. We receive an ϵ-corrupted sample $Y_1, \ldots, Y_n$ from $\mathcal{M}$ where $n = \text{poly}(d/\epsilon)$. The goal is to output a set of parameters $\widetilde{w_1}, \ldots, \widetilde{w_k}$ and $(\widetilde{\mu_1}, \widetilde{\Sigma_1}), \ldots, (\widetilde{\mu_k}, \widetilde{\Sigma_k})$ that are $\text{poly}(\epsilon)$ close to the true parameters in the sense that there exists a permutation π on $[k]$ such that for all i

$$|w_i - \widetilde{w_{\pi(i)}}|, d_{\mathsf{TV}}\left(N(\mu_i, \Sigma_i), N(\widetilde{\mu_{\pi(i)}}, \widetilde{\Sigma_{\pi(i)}})\right) \leq \text{poly}(\epsilon).$$

Throughout our paper, we will assume that all of the Gaussians that we consider have variance at least $\text{poly}(\epsilon/d)$ and at most $\text{poly}(d/\epsilon)$ in all directions i.e. they are not too flat. This implies that their covariance matrices are invertible so we may write expressions such as Σ_i^{-1}. We will also make the following assumptions about the mixture:

- The w_i are rational with denominator at most A
- For all $i \neq j$, $d_{\mathsf{TV}}(G_i, G_j) > b$

for some positive constants A, b. Note that a lower bound on the minimum mixing weight and a lower bound on the TV distance between components is necessary for parameter learning. Throughout this paper, we treat k, A, b as constants – i.e. A and b could be any function of

k – and when we say polynomial, the exponent may depend on these parameters. We are primarily interested in dependence on ϵ and d (the dimension of the space).

3.2.2 Sum of Squares Proofs

We will make repeated use of the sum-of-squares (SOS) proof system. We review a few basic facts here (see [16] for a more extensive treatment). Our exposition here closely mirrors [39].

Definition 3.2.2 (Symbolic Polynomials)**.** *A degree-t symbolic polynomial P is a collection of indeterminates $\widehat{P}(\alpha)$, one for each multiset $\alpha \subseteq [n]$ of size at most t. We think of it as representing a polynomial $P : \mathbb{R}^n \to \mathbb{R}$ in the sense that*

$$P(x) = \sum_{\alpha \subseteq [n], |\alpha| \leq t} \widehat{P}(\alpha) x^\alpha$$

Definition 3.2.3 (SOS proof)**.** *Let $x_1, \ldots, x_n$ be indeterminates and let $\mathcal{A}$ be a set of polynomial inequalities*

$$\{p_1(x) \geq 0, \ldots, p_m(x) \geq 0\}$$

An SOS proof of an inequality $r(x) \geq 0$ from constraints $\mathcal{A}$ is a set of polynomials $\{r_S(x)\}_{S \subseteq [m]}$ such that each r_S is a sum of squares of polynomials and

$$r(x) = \sum_{S \subseteq [m]} r_S(x) \prod_{i \in S} p_i(x)$$

The degree of this proof is the maximum of the degrees of $r_S(x) \prod_{i \in S} p_i(x)$ over all S. We write

$$\mathcal{A} \vdash_k r(x) \geq 0$$

to denote that the constraints $\mathcal{A}$ give an SOS proof of degree k for the inequality $r(x) \geq 0$. Note that we can represent equality constraints in $\mathcal{A}$ by including $p(x) \geq 0$ and $-p(x) \geq 0$.

The dual objects to SOS proofs are pseudoexpectations. We will repeatedly make use of pseudoexpectations later on.

Definition 3.2.4. *Let $x_1, \ldots, x_n$ be indeterminates. A degree-k pseudoexpectation $\widetilde{\mathbb{E}}$ is a linear map*

$$\widetilde{\mathbb{E}} : \mathbb{R}[x_1, \ldots, x_n]_{\leq k} \to \mathbb{R}$$

from degree-k polynomials to $\mathbb{R}$ such that $\widetilde{\mathbb{E}}[p(x)^2] \geq 0$ for any p of degree at most $k/2$ and $\widetilde{\mathbb{E}}[1] = 1$. For a set of polynomial constraints $\mathcal{A} = \{p_1(x) \geq 0, \ldots, p_m(x) \geq 0\}$, we say that $\widetilde{\mathbb{E}}$ satisfies $\mathcal{A}$ if

$$\widetilde{\mathbb{E}}[s^2(x)p_i(x)] \geq 0$$

for all polynomials $s(x)$ and $i \in [m]$ such that $s(x)^2 p_i(x)$ has degree at most k.

The key fact is that given a set of polynomial constraints, we can solve for a constant-degree pseudoexpectation that satisfies those constraints (or determine that none exist) in polynomial time as it reduces to solving a polynomially sized SDP.

Theorem 3.2.5 (SOS Algorithm [16])**.** *There is an algorithm that takes a natural number k and a satisfiable system of polynomial inequalities $\mathcal{A}$ in varibles $x_1, \ldots, x_n$ with coefficients at most 2^n containing an inequality of the form $\|x\|^2 \leq M$ for some real number M and returns in time $n^{O(k)}$ a degree-k pseudoexpectation $\widetilde{\mathbb{E}}$ which satisfies $\mathcal{A}$ up to error 2^{-n}.*

Note that there are a few technical details with regards to only being able to compute a pseudoexpectation that nearly satisfies the constraints. These technicalities do not affect our proof (as 2^{-n} errors will be negligible) so we will simply assume that we can compute a pseudoexpectation that exactly satisfies the constraints. See [16] for more details about these technicalities.

Finally, we state a few simple inequalities for pseudoexpectations that will be used repeatedly later on.

Claim 3.2.6 (Cauchy Schwarz for Pseudo-distributions)**.** *Let f, g be polynomials of degree at most k in indeterminates $x = (x_1, \ldots, x_n)$. Then for any degree k pseudoexpectation,*

$$\widetilde{\mathbb{E}}[fg] \leq \sqrt{\widetilde{\mathbb{E}}[f^2]}\sqrt{\widetilde{\mathbb{E}}[g^2]}.$$

Corollary 3.2.7. *Let $f_1, g_1, \ldots, f_m, g_m$ be polynomials of degree at most k in indeterminates $x = (x_1, \ldots, x_n)$. Then for any degree k pseudoexpectation,*

$$\widetilde{\mathbb{E}}[f_1 g_1 + \cdots + f_m g_m] \leq \sqrt{\widetilde{\mathbb{E}}[f_1^2 + \cdots + f_m^2]} \sqrt{\widetilde{\mathbb{E}}[g_1^2 + \cdots + g_m^2]}.$$

Proof. Note

$$\begin{aligned}\widetilde{\mathbb{E}}[f_1 g_1 + \cdots + f_m g_m] &\leq \sqrt{\widetilde{\mathbb{E}}[f_1^2]} \sqrt{\widetilde{\mathbb{E}}[g_1^2]} + \cdots + \sqrt{\widetilde{\mathbb{E}}[f_m^2]} \sqrt{\widetilde{\mathbb{E}}[g_m^2]} \\ &\leq \sqrt{\widetilde{\mathbb{E}}[f_1^2 + \cdots + f_m^2]} \sqrt{\widetilde{\mathbb{E}}[g_1^2 + \cdots + g_m^2]}\end{aligned}$$

where the first inequality follows from Cauchy Schwarz for pseudoexpectations and the second follows from standard Cauchy Schwarz. ■

3.3 Fun with Generating Functions

We now introduce the generating function machinery that we will use in our learning algorithm. We begin with a standard definition.

Definition 3.3.1. *Let $\mathcal{H}_m(x)$ be the univariate Hermite polynomials $\mathcal{H}_0 = 1, \mathcal{H}_1 = x, \mathcal{H}_2 = x^2 - 1 \cdots$ defined by the recurrence*

$$\mathcal{H}_m(x) = x\mathcal{H}_{m-1}(x) - (m-1)\mathcal{H}_{m-2}(x)$$

Note that in $\mathcal{H}_m(x)$, the degree of each nonzero monomials has the same parity as m. In light of this, we can write the following:

Definition 3.3.2. *Let $\mathcal{H}_m(x, y^2)$ be the homogenized Hermite polynomials e.g. $\mathcal{H}_2(x, y^2) = x^2 - y^2, \mathcal{H}_3(x, y^2) = x^3 - 3xy^2$.*

It will be important to note the following fact:

Claim 3.3.3. *We have*

$$e^{xz-\frac{1}{2}y^2z^2} = \sum_{m=0}^{\infty} \frac{1}{m!}\mathcal{H}_m(x, y^2)z^m$$

where the RHS is viewed as a formal power series in z whose coefficients are polynomials in x, y.

Now we define a multivariate version of the Hermite polynomials. This is similar to the definition in Section 2.9.1 except instead of representing these objects using a tensor, we will represent them as a polynomial as this will be more useful for the modes of analysis that we use here.

Definition 3.3.4. *Let $H_m(X, z)$ be a formal polynomial in variables $X = X_1, \ldots, X_d$ whose coefficients are polynomials in d variables $z_1, \ldots, z_d$ that is given by*

$$H_m(X, z) = \mathcal{H}_m(z_1X_1 + \cdots + z_dX_d, X_1^2 + \cdots + X_d^2)$$

Note that H_m is homogeneous of degree m as a polynomial in $X_1, \ldots, X_d$

Definition 3.3.5. *For a distribution D on $\mathbb{R}^d$, we let*

$$h_{m,D}(X) = \mathop{\mathbb{E}}_{(z_1,\ldots,z_d)\sim D}[H_m(X, z)]$$

where we take the expectation of H_m over $(z_1, \ldots, z_d)$ drawn from D. Note that $h_{m,D}(X)$ is a polynomial in $(X_1, \ldots, X_d)$. We will omit the D in the subscript when it is clear from context. Moreover for a mixture of Gaussians

$$\mathcal{M} = w_1N(\mu_1, \Sigma_1) + \ldots w_kN(\mu_k, \Sigma_k)$$

we will refer to the Hermite polynomials $h_{m,\mathcal{M}}$ as the Hermite polynomials of the mixture.

We remark that if there is a mixture $\mathcal{M} = w_1N(\mu_1, \Sigma_1) + \ldots w_kN(\mu_k, \Sigma_k)$ where instead of real numbers, the w_i, μ_i, Σ_i are given in terms of indeterminates, the Hermite polynomials

will be polynomials in those indeterminates. We will repeatedly make use of this abstraction later on.

The first important observation is that the Hermite polynomials for Gaussians can be written in a simple closed form via generating functions.

Claim 3.3.6. *Let $D = N(\mu, I + \Sigma)$. Let $a(X) = \mu \cdot X$ and $b(X) = X^T \Sigma X$. Then*

$$e^{a(X)y + \frac{1}{2}b(X)y^2} = \sum_{m=0}^{\infty} \frac{1}{m!} \cdot h_{m,D}(X) y^m$$

as formal power series in y.

Proof. By Claim 3.3.3, we have

$$e^{a(X)y + \frac{1}{2}b(X)y^2} = \sum_{m=0}^{\infty} \frac{1}{m!} \mathcal{H}_m(a(X), -b(X)) y^m$$

It now suffices to verify that

$$\mathop{\mathbb{E}}_{(z_1, \ldots z_d) \sim D} \left[\mathcal{H}_m(z_1 X_1 + \cdots + z_d X_d, X_1^2 + \cdots + X_d^2) \right] = \mathcal{H}_m(a(X), -b(X))$$

This can be verified through straight-forward computations using the moment tensors of a Gaussian (see Lemma 2.7 in [65]). ■

We now have two simple corollaries to the above.

Corollary 3.3.7. *Let $\mathcal{M} = w_1 N(\mu_1, I + \Sigma_1) + \ldots w_k N(\mu_k, I + \Sigma_k)$. Let $a_i(X) = \mu_i \cdot X$ and $b_i(X) = X^T \Sigma_i X$. Then*

$$\sum_{m=0}^{\infty} \frac{1}{m!} \cdot h_{m,\mathcal{M}}(X) y^m = w_1 e^{a_1(X)y + \frac{1}{2}b_1(X)y^2} + \cdots + w_k e^{a_k(X)y + \frac{1}{2}b_k(X)y^2}$$

Corollary 3.3.8. *Let $\mathcal{M} = w_1 N(\mu_1, I + \Sigma_1) + \ldots w_k N(\mu_k, I + \Sigma_k)$. Let $a_i(X) = \mu_i \cdot X$ and $b_i(X) = X^T \Sigma_i X$. Then the Hermite polynomials $h_{m,\mathcal{M}}(X)$ can be written as a linear combination of products of the $a_i(X), b_i(X)$ such that the number of terms in the sum, the*

number of terms in each product, and the coefficients in the linear combination are all bounded as functions of m, k.

The next important insight is that the generating functions for the Hermite polynomials behave nicely under certain differential operators. We can use these differential operators to derive identities that the Hermite polynomials must satisfy and these identities will be a crucial ingredient in our learning algorithm.

The proceeding claims all follow from direct computation.

Claim 3.3.9. *Let ∂ denote the differential operator with respect to y. If*

$$f(y) = P(y, X)e^{a(X)y+\frac{1}{2}b(X)y^2}$$

where P is a polynomial in y of degree k (whose coefficients are polynomials in X) then

$$(\partial - (a(X) + yb(X)))f(y) = Q(y, X)e^{a(X)y+\frac{1}{2}b(X)y^2}$$

where Q is a polynomial in y with degree exactly $k-1$ whose leading coefficient is k times the leading coefficient of P.

Corollary 3.3.10. *Let ∂ denote the differential operator with respect to y. If*

$$f(y) = P(y, X)e^{a(X)y+\frac{1}{2}b(X)y^2}$$

where P is a polynomial in y of degree k then

$$(\partial - (a(X) + yb(X)))^{k+1}f(y) = 0.$$

Claim 3.3.11. *Let ∂ denote the differential operator with respect to y. Let*

$$f(y) = P(y, X)e^{a(X)y+\frac{1}{2}b(X)y^2}$$

where P is a polynomial in y of degree k. Let the leading coefficient of P (viewed as a polynomial in y) be $L(X)$. Let $c(X), d(X)$ be a linear and quadratic polynomial in the X variables respectively such that $\{a(X), b(X)\} \neq \{c(X), d(X)\}$. If $b(X) \neq d(X)$ then

$$(\partial - (c(X) + yd(X)))^{k'} f(y) = Q(y, X)e^{a(X)y + \frac{1}{2}b(X)y^2}$$

where Q is a polynomial of degree $k + k'$ in y with leading coefficient

$$L(x)(b(X) - d(X))^{k'}$$

and if $b(X) = d(X)$ then

$$(\partial - (c(X) + yd(X)))^{k'} f(y) = Q(y, X)e^{a(X)y + \frac{1}{2}b(X)y^2}$$

where Q is a polynomial of degree k in y with leading coefficient

$$L(X)(a(X) - c(X))^{k'}$$

3.3.1 Polynomial Factorizations

The analysis of our SOS-based learning algorithm will rely on manipulations of Hermite polynomials. An important piece of our analysis is understanding how the coefficients of polynomials behave under addition and (polynomial) multiplication. Specifically, if we have two polynomials $f(X), g(X)$ and we have bounds on the coefficients of f and g, we now want to give bounds on the coefficients of the polynomials $f(X) + g(X)$ and $f(X)g(X)$. Most of these bounds are easy to obtain. The one that is somewhat nontrivial is lower bounding the coefficients of $f(X)g(X)$ i.e. if the coefficients of f and g are not all small, then the product $f(X)g(X)$ cannot have all of its coefficients be too small.

Definition 3.3.12. *For a polynomial $f(X)$ in the d variables $X_1, \ldots, X_d$ with real coefficients define $v(f)$ to be the vectorization of the coefficients. (We will assume this is done in*

a consistent manner so that the same coordinate of vectorizations of two polynomials corresponds to the coefficient of the same monomial.) We will frequently consider expressions of the form $\|v(f)\|$ i.e. the L^2 norm of the coefficient vector.

Definition 3.3.13. *For a polynomial $A(X)$ of degree k in d variables $X_1, \ldots, X_d$ and a vector $v \in \mathbb{R}^d$ with nonnegative integer entries summing to at most k, we use A_v to denote the corresponding coefficient of A.*

First, we prove a simple result about the norm of the vectorization of a sum of polynomials.

Claim 3.3.14. *Let $f_1, \ldots, f_m$ be polynomials in $X_1, \ldots, X_d$ whose coefficients are polynomials in formal variables $u_1, \ldots, u_n$ of degree $O_k(1)$. Then*

$$\|v(f_1 + \cdots + f_m)\|^2 \leq m(\|v(f_1)\|^2 + \cdots + \|v(f_m)\|^2)$$

Furthermore, the difference can be written as a sum of squares of polynomials of degree $O_k(1)$ in $u_1, \ldots, u_n$.

Proof. Note

$$(a_1 + \cdots + a_m)^2 \leq m(a_1^2 + \cdots + a_m^2)$$

and the difference between the two sides can be written as a sum of squares

$$\sum_{i \neq j} (a_i - a_j)^2$$

The desired inequality can now be obtained by summing expressions of the above form over all coefficients. ■

Next, we upper bound the norm of the vectorization of a product of polynomials.

Claim 3.3.15. *Let $f, g, h_1, \ldots, h_k$ be polynomials in $X_1, \ldots, X_d$ of degree at most k with coefficients that are polynomials in formal variables $u_1, \ldots, u_n$ of degree $O_k(1)$ Then for any*

pseudoexpectation $\widetilde{\mathbb{E}}$ *of degree* C_k *for some sufficiently large constant* C_k *depending only on* k*,*

$$\widetilde{\mathbb{E}}[\|v(h_1)\|^2 \dots \|v(h_k)\|^2 \|v(fg)\|^2] \leq O_k(1)\widetilde{\mathbb{E}}[\|v(h_1)\|^2 \dots \|v(h_k)\|^2 \|v(f)\|^2 \|v(g)\|^2]$$

where the pseudoexpectation operates on polynomials in $u_1, \dots, u_n$*.*

Proof. Note that each monomial in the product fg has degree at most $2k$ and thus can only be split in $O_k(1)$ ways. Specifically, each entry of $v(fg)$ can be written as a sum of $O_k(1)$ entries of $v(f) \otimes v(g)$ so

$$\|v(fg)\|^2 \leq O_k(1) \|v(f)\|^2 \|v(g)\|^2$$

where the difference between the two sides can be written as a sum of squares. This implies the desired inequality. ■

Before we prove the final result in this section, we introduce a few definitions.

Definition 3.3.16. *For a vector* $v \in \mathbb{R}^d$ *with integer coordinates, we define* $\tau(v)$ *to be the multiset formed by the coordinates of* v*. We call* τ *the type of* v*.*

Definition 3.3.17. *For a monomial say* $X_1^{a_1} \dots X_d^{a_d}$*, we call* $(a_1, \dots, a_d) \in \mathbb{R}^d$ *its degree vector.*

Now we can prove a lower bound on the norm of the vectorization of the product of polynomials.

Claim 3.3.18. *Let* $f, g, h_1, \dots, h_k$ *be polynomials in* $X_1, \dots, X_d$ *of degree at most* k *with coefficients that are polynomials in formal variables* $u_1, \dots, u_n$ *of degree* $O_k(1)$*. Then for any pseudoexpectation* $\widetilde{\mathbb{E}}$ *of degree* C_k *for some sufficiently large constant* C_k *depending only on* k*,*

$$\widetilde{\mathbb{E}}[\|v(h_1)\|^2 \dots \|v(h_k)\|^2 \|v(fg)\|^2] \geq \Omega_k(1)\widetilde{\mathbb{E}}[\|v(h_1)\|^2 \dots \|v(h_k)\|^2 \|v(f)\|^2 \|v(g)\|^2]$$

where the pseudoexpectation operates on polynomials in $u_1, \dots, u_n$*.*

Proof. We will first prove the statement for $h_1 = \cdots = h_k = 1$.

Let S be the set of all types that can be obtained by taking the sum of two degree vectors for monomials of degree at most k and let T be the set of all types that can be obtained by taking the difference of two degree vectors for monomials of degree at most k. Note that $|S|, |T| = O_k(1)$. Now

$$\widetilde{\mathbb{E}}[\|v(fg)\|^2] = \widetilde{\mathbb{E}}\left[\sum_a \left(\sum_{u+v=a} f_u g_v\right)^2\right] = \widetilde{\mathbb{E}}\left[\sum_{u_1+v_1=u_2+v_2} f_{u_1} g_{v_1} f_{u_2} g_{v_2}\right]$$
$$= \widetilde{\mathbb{E}}\left[\sum_{u_1-v_2=u_2-v_1} f_{u_1} g_{v_1} f_{u_2} g_{v_2}\right] = \widetilde{\mathbb{E}}\left[\sum_b \left(\sum_{u-v=b} f_u g_v\right)^2\right]$$

where the sums in the above expression are over all a and all b that are vectors in $\mathbb{Z}^d$ for which the inner summands are nonempty. Let $T = \{t_1, \ldots, t_n\}$ where the types $t_1, \ldots, t_n$ are sorted in non-increasing order of their L^2 norm. Recall that T consists of all types that can be obtained by taking the difference of two degree vectors corresponding to monomials of degree at most k. Now first note

$$\widetilde{\mathbb{E}}[\|v(fg)\|^2] \geq \widetilde{\mathbb{E}}\left[\sum_{b,\tau(b)=t_1} \left(\sum_{u-v=b} f_u g_v\right)^2\right] = \widetilde{\mathbb{E}}\left[\sum_{b,\tau(b)=t_1} \left(\sum_{u-v=b} (f_u g_v)^2\right)\right]$$

since t_1 corresponds to the type $(k, -k)$ and each of the inner summands only contains one term. Now consider t_i for $i > 1$.

$$\widetilde{\mathbb{E}}\left[\sum_{b,\tau(b)=t_i} \left(\sum_{u-v=b} f_u g_v\right)^2\right] = \widetilde{\mathbb{E}}\left[\sum_{b,\tau(b)=t_i} \left(\sum_{u-v=b} (f_u g_v)^2\right)\right]$$
$$+ 2\widetilde{\mathbb{E}}\left[\sum_{b,\tau(b)=t_i} \left(\sum_{\substack{\{u_1,v_1\}\neq\{u_1,v_2\} \\ u_1-v_1=u_2-v_2=b}} f_{u_1} f_{u_2} g_{v_1} g_{v_2}\right)\right]$$

Note that in the second sum, either $u_1 - v_2 \in t_j$ for $j < i$ or $u_2 - v_1 \in t_j$ for $j < i$. To see

this, let $a = v_1 - v_2$. Then $u_1 - v_2 = b + a$ and $u_2 - v_1 = b - a$. Now

$$\|b - a\|_2^2 + \|b + a\|_2^2 > \|b\|_2^2$$

since $a \neq 0$ so one of the differences must be of an earlier type.

Next, note that for a fixed u_1, v_2, there are at most $O_k(1)$ possible values for u_2', v_1' such that the term $f_{u_1} f_{u_2'} g_{v_1'} g_{v_2}$ appears. This is because we must have $u_1 + v_2 = u_2' + v_1'$ and there are only $O_k(1)$ ways to achieve this. Thus, by Cauchy Schwarz

$$\widetilde{\mathbb{E}}\left[\sum_{b,\tau(b)=t_i}\left(\sum_{u-v=b} f_u g_v\right)^2\right] \geq \widetilde{\mathbb{E}}\left[\sum_{b,\tau(b)=t_i}\left(\sum_{u-v=b}(f_u g_v)^2\right)\right]$$
$$-O_k(1)\sqrt{\widetilde{\mathbb{E}}\left[\sum_{j<i}\sum_{b,\tau(b)=t_j}\left(\sum_{u-v=b}(f_u g_v)^2\right)\right]} \cdot \sqrt{\widetilde{\mathbb{E}}[\|v(f)\|^2 \|v(g)\|^2]}$$

Now combining the above with the fact that $|T| = n = O_k(1)$ and that

$$\widetilde{\mathbb{E}}[\|v(f)\|^2 \|v(g)\|^2] = \widetilde{\mathbb{E}}\left[\sum_{i=1}^{n}\sum_{b,\tau(b)=t_i}\left(\sum_{u-v=b}(f_u g_v)^2\right)\right]$$

we can complete the proof. To see this, for each i, let

$$Q_i = \widetilde{\mathbb{E}}\left[\sum_{b,\tau(b)=t_i}\left(\sum_{u-v=b}(f_u g_v)^2\right)\right]$$
$$R_i = \widetilde{\mathbb{E}}\left[\sum_{b,\tau(b)=t_i}\left(\sum_{u-v=b} f_u g_v\right)^2\right]$$

Also normalize so that

$$\widetilde{\mathbb{E}}[\|v(f)\|^2 \|v(g)\|^2] = 1.$$

Let δ be some suitably chosen constant depending only on k. If $Q_1 \geq \delta$ then we are done.

Otherwise, we have an upper bound on the square root terms that are subtracted in the expression for R_2. If $Q_2 \geq \Omega_1(k)\sqrt{\delta}$ then we are again done (since we have now reduced to the case where $Q_1 \leq \delta$). Iteratively repeating this procedure, we are done whenever one of the Q_i is sufficiently large compared to $Q_1, \ldots, Q_{i-1}$. However, not all of $Q_1, \ldots, Q_n$ can be small since their sum is 1. Choosing δ to be a sufficiently small constant but depending only on k we conclude that

$$\widetilde{\mathbb{E}}[\|v(fg)\|^2] \geq \Omega_k(1)\widetilde{\mathbb{E}}[\|v(f)\|^2 \|v(g)\|^2]$$

as desired.

For the general case when not all of the h_i are 1, we can multiply the insides of all of the pseudoexpectations above by $\|v(h_1)\|^2 \ldots \|v(h_k)\|^2$ and the same argument will work. ■

3.4 Components Are Not Far Apart

Now we are ready to present our main contribution: an algorithm that learns the parameters of a mixture of Gaussians $\mathcal{M} = w_1 G_1 + \cdots + w_k G_k$ from an ϵ-corrupted sample when the components are not too far apart. In this section, we will assume that the mixture is in nearly isotropic position and that we have estimates for the Hermite polynomials. We will show how to learn the parameters from these estimates. In the next section, Section 3.5, we show how to actually place the mixture in isotropic position and obtain estimates for the Hermite polynomials.

We use the following conventions:

- The true means and covariances are given by $(\mu_1, I + \Sigma_1), \ldots, (\mu_k, I + \Sigma_k)$
- The true mixing weights are $w_1, \ldots, w_k$ and are all bounded below by some value $w_{\min}$
- Δ is an upper bound that we have on $\|\mu_i\|$ and $\|\Sigma_i\|$ i.e. the components are not too

far separated.

- $\|\mu_i - \mu_j\|_2 + \|\Sigma_i - \Sigma_j\|_2 \geq c$ for all $i \neq j$ i.e. no pair of components is too close
- We should think of $w_{\min}, c$ as being at least ϵ^r and Δ being at most ϵ^{-r} for some sufficiently small value of $r > 0$.
- Let the Hermite polynomials for the true mixture be given by $h_1 = h_{1,\mathcal{M}}, h_2 = h_{2,\mathcal{M}}, \ldots$ where

$$\mathcal{M} = w_1 N(\mu_1, I + \Sigma_1) + \cdots + w_k N(\mu_k, I + \Sigma_k)$$

In this section we assume that we have the following:

- Estimates $\overline{h}_i(X)$ for the Hermite polynomials such that $\left\|v(\overline{h_i}(X) - h_i(X))\right\|^2 \leq \epsilon' = \text{poly}(\epsilon)$

and our only interaction with the actual samples is through these estimates. We will show how to obtain these estimates in Section 3.5 (closely mirroring the method in [65]).

The main theorem that we prove in this section is as follows.

Theorem 3.4.1. *Let ϵ' be a parameter that is sufficiently small in terms of k. There is a sufficiently small function $f(k)$ and a sufficiently large function $F(k)$ such that if*

$$\mathcal{M} = w_1 N(\mu_1, I + \Sigma_1) + \cdots + w_k N(\mu_k, I + \Sigma_k)$$

is a mixture of Gaussians with

- $\|\mu_i\|_2, \|\Sigma_i\|_2 \leq \Delta$ *for all* i
- $\|\mu_i - \mu_j\|_2 + \|\Sigma_i - \Sigma_j\|_2 \geq c$ *for all* $i \neq j$
- $w_1, \ldots, w_k \geq w_{\min}$

for parameters $w_{\min}, c \geq \epsilon'^{f(k)}$ *and* $\Delta \leq \epsilon'^{-f(k)}$ *and we are given estimates* $\overline{h}_i(X)$ *for the Hermite polynomials for all* $i \leq F(k)$ *such that*

$$\left\| v(\overline{h_i}(X) - h_i(X)) \right\|^2 \leq \epsilon'$$

where h_i *are the Hermite polynomials for the true mixture* $\mathcal{M}$*, then there is an algorithm that returns* $\text{poly}(1/\epsilon')^{O_1(k)}$ *candidate mixtures, at least one of which satisfies*

$$\|w_i - \widetilde{w}_i\| + \|\mu_i - \widetilde{\mu}_i\|_2 + \left\|\Sigma_i - \widetilde{\Sigma}_i\right\|_2 \leq \epsilon'^{f(k)}$$

for all i*.*

Informally, assuming that the parameters of the components of the mixture are bounded by $\text{poly}(1/\epsilon')$ and that their separation is at least $\text{poly}(\epsilon')$, given ϵ'-accurate estimates for the Hermite polynomials, we can learn the parameters of the mixture to within Frobenius error $\text{poly}(\epsilon')$.

3.4.1 Reducing to all pairs of parameters equal or separated

We claim that it suffices to work under the following assumption. All pairs of parameters are either separated of equal. More specifically, for each pair of parameters μ_i, μ_j (and same for Σ_i, Σ_j), either $\mu_i = \mu_j$ or

$$\|\mu_i - \mu_j\|_2 \geq c$$

We now prove that it suffices to work with the above simplification. For any function $0 < f(k) < 1$ depending only on k, there is some $C \geq (f(k))^{k^2}$ such that there is no pair of parameters μ_i, μ_j or Σ_i, Σ_j whose distance is in the interval $[\epsilon'^C, \epsilon'^{f(k)C}]$. Now consider the graph on the k nodes where i, j are connected if and only if

$$\|\mu_i - \mu_j\| \leq \epsilon'^{f(k)C}$$

We now construct a new mixture $N(\mu'_i, \Sigma'_i)$. For each connected component say $\{i_1, \ldots, i_j\}$

, pick a representative and set $\mu'_{i_1} = \mu'_{i_2} = \cdots = \mu'_{i_j} = \mu_{i_1}$. Do this for all connected components and similar in the graph on covariance matrices. For all i, we have

$$\|\mu'_i - \mu_i\|, \|\Sigma'_i - \Sigma_i\| \leq O_k(1)\epsilon'^C$$

because there is a path of length at most k connecting i to the representative in its component that it is rounded to, and all edges correspond to pairs within distance of ϵ'^C.

The Hermite polynomials of this new mixture satisfy

$$\|v(h'_m - h_m)\|^2 \leq O_k(1)\Delta^{O_k(1)}\epsilon'^{\Omega_k(1)C}$$

as long as m is bounded as a function of k. If we pretend that the new mixture is the true mixture, we have estimates $\overline{h_i}(X)$ such that

$$\left\|v(h'_i - \overline{h_i})\right\|^2 \leq O_k(1)\Delta^{O_k(1)}\epsilon'^{\Omega_k(1)C}$$

and all pairs of parameters in the new mixture are either equal or $\epsilon'^{f(k)C}$ separated. If we prove Theorem 3.4.1 with the assumption that the pairs of parameters are separated or equal, then we can choose $f(k)$ accordingly and then we deduce that the theorem holds in the general case (with worse, but still polynomial, bounds on $\Delta, c, w_{\min}$ and the accuracy of our output as a function of ϵ').

From now on we will work with the assumption that each pair of parameters is either equal or separated by c.

3.4.2 SOS Program Setup

Our algorithm for learning the parameters when given estimates of the Hermite polynomials involves solving an SOS program. Here we set up the SOS program that we will solve.

We will let $D = \binom{d}{2} + d$. We think of mapping between symmetric $d \times d$ matrices and $\mathbb{R}^D$ as

$$\begin{bmatrix} a_{11} & \dots & a_{1d} \\ \vdots & \ddots & \vdots \\ a_{d1} & \dots & a_{dd} \end{bmatrix} \leftrightarrow (a_{11}, 2a_{12}, 2a_{13}, \dots, a_{dd})$$

Definition 3.4.2 (Parameter Solving Program $\mathcal{S}$)**.** *We will have the following variables*

- $u_1 = (u_{11}, \dots, u_{1d}), \dots, u_k = (u_{k1}, \dots, u_{kd})$
- $v_1 = (v_{1,(1,1)}, v_{1,(1,2)}, \dots, v_{1,(d,d)}), \dots, v_k = (v_{k,(1,1)}, v_{k,(1,2)}, \dots, v_{k,(d,d)})$

In the above $u_1, \dots, u_k \in \mathbb{R}^d$ and $v_1, \dots v_k \in \mathbb{R}^D$. Our goal will be to solve for these variables in a way so that the solutions form orthonormal bases for the span of the μ_i and the span of the Σ_i. Note $v_1, \dots, v_k$ live in $\mathbb{R}^D$ because the Σ_i must be symmetric.

We guess coefficients a_{ij}, b_{ij} where $i, j \in [k]$ expressing the means and covariances in this orthonormal basis. We ensure that the guesses satisfy the property that for every pair of vectors $A_i = (a_{i1}, \dots, a_{ik}), A_j = (a_{j1}, \dots, a_{jk})$ either $A_i = A_j$ or

$$\|A_i - A_j\|_2 \geq \frac{c}{2}$$

and similarly for B_i, B_j. We ensure that

$$\|A_i\|_2 \leq 2\Delta$$

Ensure similar conditions for the $\{B_i\}$. We also guess the mixing weights $\widetilde{w_1}, \dots, \widetilde{w_k}$ and ensure that our guesses are all at least $w_{\min}/2$.

Now we set up the constraints. Let C be a sufficiently large integer depending only on k. Define $\widetilde{\mu}_i = a_{i1}u_1 + \cdots + a_{ik}u_k$ and define $\widetilde{\Sigma}_i$ similarly. These are linear expressions in the variables that we are solving for. Now consider the hypothetical mixture with mixing weights $\widetilde{w}_i$, means $\widetilde{\mu}_i$, and covariances $I + \widetilde{\Sigma}_i$. The Hermite polynomials for this hypothetical mixture $\widetilde{h}_i(X)$ can be written as formal polynomials in $X = (X_1, \dots, X_d)$ with coefficients that are

polynomials in u, v. Note that we can explicitly write down these Hermite polynomials. The set of constraints for our SOS system is as follows:

- $\|u_i\|_2^2 = 1$ *for all* $1 \leq i \leq k$
- $\|v_i\|_2^2 = 1$ *for all* $1 \leq i \leq k$
- $u_i \cdot u_j = 0$ *for all* $i \neq j$
- $v_i \cdot v_j = 0$ *for all* $i \neq j$
- *For all* $p = 1, 2, \ldots, C$
$$\left\| v(\widetilde{h_p}(X) - \overline{h_p}(X)) \right\|^2 \leq 100\epsilon'$$

Note that we can explicitly write down the last set of constraints because we have estimates $\overline{h_i}$.

It is important to note that the $\widetilde{w_i}, A_i, B_i$ are real numbers. We will attempt to solve the system for each of our guesses and show that for some set of guesses, we obtain a solution from which we can recover the parameters. We can brute-force search over an ϵ'-net because there are only $O_k(1)$ parameters to guess. We call the SOS program that we set up $\mathcal{S}$.

3.4.3 Analysis

We now prove a set of properties that must be satisfied by any pseudoexpectation of degree C_k satisfying $\mathcal{S}$ where C_k is a sufficiently large constant depending only on k. What we would ideally want to show is that

- The span of the $\widetilde{\Sigma_i}$ is close to the span of the Σ_i
- The span of the $\widetilde{\mu_i}$ is close to the span of the μ_i

However, it appears to be difficult to prove a statement of the above form within an SOS framework. Instead, we will look at the pseudoexpectations of the matrices

$$M_i = \widetilde{\mathbb{E}}[\widetilde{\Sigma}_i \widetilde{\Sigma}_i^T]$$

(where $\widetilde{\Sigma}_i$ is viewed as a length-D vector so $\widetilde{\Sigma}_i \widetilde{\Sigma}_i^T$ is a $D \times D$ matrix.) The two key properties that we will prove about these matrices are in Lemmas 3.4.11 and 3.4.12.

Roughly Lemma 3.4.11 says that any singular vector that corresponds to a large singular value of M_i must be close to the span of the $\{\Sigma_i\}$. Lemma 3.4.12 says that any vector v that has large projection onto the subspace spanned by the $\{\Sigma_i\}$ must have the property that $v^T M_i v$ is large for some i. Putting these together, we can take the the top-k principal components of each of $M_1, \ldots, M_k$ and show that the span of these essentially contains the span of the $\{\Sigma_i\}$ (this last step is done outside the SOS framework). We can now brute-force over an ϵ'-net and guess the Σ_i (since we have narrowed them down to an $O_k(1)$-dimensional subspace). We can then plug in real values for the covariances and solve for the means using a similar method.

Algebraic Identities

First we will prove several purely algebraic identities. We will slightly abuse notation and for $\mu \in \mathbb{R}^d$, we use $\mu(X)$ to denote the inner product of μ with the formal variables $(X_1, \ldots, X_d)$ and for $\Sigma \in \mathbb{R}^D$, we will use $\Sigma(X)$ to denote the quadratic form in formal variables $(X_1, \ldots, X_d)$ given by $X^T \Sigma X$ (when Σ is converted to a symmetric $d \times d$ matrix). It will be useful to consider the following two formal power series (in y)

$$F(y) = \sum_{i=1}^{k} w_i e^{\mu_i(X)y + \frac{1}{2}\Sigma_i(X)y^2}$$

$$\widetilde{F}(y) = \sum_{i=1}^{k} \widetilde{w}_i e^{\widetilde{\mu_i}(X)y + \frac{1}{2}\widetilde{\Sigma_i}(X)y^2}$$

We view these objects in the following way: the coefficients of $1, y, y^2, \cdots$ are formal polynomials in $(X_1, \ldots, X_d)$. In the first expression, the coefficients of these polynomials are (unknown) constants. In the second, the coefficients are polynomials in the variables $u_1, \ldots, u_k, v_1, \ldots, v_k$. In fact, the coefficients in the first power series are precisely $h_1, h_2, \ldots$ while the coefficients in the second power series are precisely $\widetilde{h}_1, \widetilde{h}_2, \ldots$. The key insight is the following:

> *After taking derivatives and polynomial combinations of either of the above formal power series, the coefficients can still be expressed as polynomial combinations of their respective Hermite polynomials.*

Definition 3.4.3. *Let $\mathcal{D}_i$ denote the differential operator $(\partial - (\mu_i(X) + \Sigma_i(X)y))$ and $\widetilde{\mathcal{D}_i}$ denote the differential operator $(\partial - (\widetilde{\mu}_i(X) + \widetilde{\Sigma}_i(X)y))$. As usual, the partial derivatives are taken with respect to y.*

To simplify the exposition, we make the following definition:

Definition 3.4.4. *Consider a polynomial $P(X)$ that is a formal polynomial in $X_1, \ldots, X_d$ whose coefficients are polynomials in the indeterminates $u_1, \ldots, u_k, v_1, \ldots, v_k$. We say P is m-simple if P can be written as a linear combination of a constant number of terms that are a product of some of $\{\mu_i(X)\}, \{\Sigma_i(X)\}, \{\widetilde{\mu}_i(X)\}, \{\widetilde{\Sigma}_i(X)\}$ where*

1. *The coefficients in the linear combination are bounded by a constant depending only on m, k*
2. *The number of terms in the sum depends only on m and k*
3. *The number of terms in each product depends only on m and k*

Claim 3.4.5. *Consider the power series*

$$\widetilde{\mathcal{D}_{k-1}}^{2^{2k-2}} \ldots \widetilde{\mathcal{D}_1}^{2^k} \mathcal{D}_k^{2^{k-1}} \ldots \mathcal{D}_1^1(\widetilde{F})$$

For any m, the coefficient of y^m when the above is written as a formal power series can be written in the form

$$P_0(X) + P_1(X)\widetilde{h_1}(X) + \cdots + P_{m'}(X)\widetilde{h_{m'}}(X)$$

where

- *m' depends only on m and k*
- *Each of the P_i is m-simple*
- *We have*

$$P_0(X) + P_1(X)h_1(X) + \cdots + P_{m'}(X)h_{m'}(X) = 0$$

as an algebraic identity over formal variables $X_1, \ldots, X_d, \{u_i\}, \{v_i\}$.

Proof. Note the coefficients of $\widetilde{F}$ (as a formal power series in y) are exactly given by the $\widetilde{h_i}$. Now the number of differential operators we apply is $O_k(1)$. The first two statements can be verified through straightforward computations since when applying each of the differential operators, we are simply multiplying the coefficients by some of $\{\mu_i(X)\}, \{\Sigma_i(X)\}, \{\widetilde{\mu_i}(X)\}, \{\widetilde{\Sigma_i}(X)\}$ and taking a linear combination. Next, note that by Corollary 3.3.10

$$\mathcal{D}_k^{2^{k-1}} \ldots \mathcal{D}_1^1(F) = 0.$$

To see this, we prove by induction that the differential operator

$$\mathcal{D}_j^{2^{j-1}} \ldots \mathcal{D}_1^1(F)$$

annihilates the components of F corresponding to Gaussians $N(\mu_1, I+\Sigma_1), \ldots, N(\mu_j, I+\Sigma_j)$. The base case is clear. To complete the induction step, note that by Corollary 3.3.10, the above operator puts polynomials of degree at most $1 + 2 + \cdots + 2^{j-1} = 2^j - 1$ in front of the

other components. Thus the operator

$$\mathcal{D}_{j+1}^{2^j} \dots \mathcal{D}_1^1(F)$$

annhiliates the first $j+1$ components, completing the induction. We now conclude that

$$\widetilde{\mathcal{D}_{k-1}}^{2^{2k-2}} \dots \widetilde{\mathcal{D}_1}^{2^k} \mathcal{D}_k^{2^{k-1}} \dots \mathcal{D}_1^1(F) = 0$$

implying that if the coefficients of $\widetilde{F}$ were $h_1, \dots, h_m$, then the result would be identically zero. ■

Claim 3.4.6. *Consider the power series*

$$\mathcal{D}_{k-1}^{2^{2k-2}} \dots \mathcal{D}_1^{2^k} \widetilde{\mathcal{D}_k}^{2^{k-1}} \dots \widetilde{\mathcal{D}_1}^1(F)$$

For any m, the coefficient of y^m when the above is written as a formal power series can be written in the form

$$P_0(X) + P_1(X)h_1(X) + \dots + P_{m'}(X)h_{m'}(X)$$

where

- *m' depends only on m and k*
- *Each of the P_i is m-simple*
- *We have*

$$P_0(X) + P_1(X)\widetilde{h_1}(X) + \dots + P_{m'}(X)\widetilde{h_{m'}}(X) = 0$$

  as an algebraic identity over formal variables $X_1, \dots, X_d, \{u_i\}, \{v_i\}$.

Proof. The proof is identicial to the proof of Claim 3.4.5. ■

Note that the polynomials P_i in Claim 3.4.5 and Claim 3.4.6 are *not* necessarily the same.

Warm-up: All Pairs of Parameters are Separated

As a warm-up, we first analyze the case where all pairs of true parameters μ_i, μ_j and Σ_i, Σ_j satisfy $\|\mu_i - \mu_j\|_2 \geq c$ and $\|\Sigma_i - \Sigma_j\|_2 \geq c$. We will show how to deal with the general case where parameters may be separated or equal in Section 3.4.3.

We can assume that our guesses satisfy $\|A_i - A_j\|_2 \geq c/2$ and $\|B_i - B_j\|_2 \geq c/2$ for all i, j. The key expressions to consider are applying the following differential operators

$$\mathcal{D} = \widetilde{\mathcal{D}_k}^{2^{2k-1}-1}\widetilde{\mathcal{D}_{k-1}}^{2^{2k-2}} \dots \widetilde{\mathcal{D}_1}^{2^k}\mathcal{D}_k^{2^{k-1}} \dots \mathcal{D}_1^1$$
$$\widetilde{\mathcal{D}} = \mathcal{D}_k^{2^{2k-1}-1}\mathcal{D}_{k-1}^{2^{2k-2}} \dots \mathcal{D}_1^{2^k}\widetilde{\mathcal{D}_k}^{2^{k-1}} \dots \widetilde{\mathcal{D}_1}^{1}$$

to F and $\widetilde{F}$ respectively. The reason these differential operators are so useful is that $\mathcal{D}$ zeros out the generating function for the true mixture and also zeros out all but one component of the generating function for the hypothetical mixture with parameters $\widetilde{w}_i, \widetilde{\mu}_i, I + \widetilde{\Sigma}_i$. For the one component that is not zeroed out, only the leading coefficient remains and we can use Claim 3.3.11 to explicitly compute the leading coefficient. Thus, we can compare the results of applying these operators on the generating functions for the true and hypothetical mixtures and, using the fact that the Hermite polynomials for these mixtures must be close, we obtain algebraic relations that allow us to extract information about individual components.

We begin by explicitly computing the relevant leading coefficients.

Claim 3.4.7. *Write*

$$\widetilde{\mathcal{D}_k}^{2^{2k-1}-1}\widetilde{\mathcal{D}_{k-1}}^{2^{2k-2}} \dots \widetilde{\mathcal{D}_1}^{2^k}\mathcal{D}_k^{2^{k-1}} \dots \mathcal{D}_1^1(\widetilde{F})$$

as a formal power series in y. Its evaluation at $y = 0$ is

$$C_k\widetilde{w_k}\prod_{i=1}^{k}(\widetilde{\Sigma_k}(X) - \Sigma_i(X))^{2^{i-1}}\prod_{i=1}^{k-1}(\widetilde{\Sigma_k}(X) - \widetilde{\Sigma}_i(X))^{2^{k+i-1}}$$

where C_k is a constant depending only on k.

Proof. Write

$$\widetilde{F}(y) = \sum_{i=1}^{k} \widetilde{w_i} e^{\widetilde{\mu_i}(X)y + \frac{1}{2}\widetilde{\Sigma_i}(X)y^2}$$

When applying the differential operator, by Corollary 3.3.10, all of the terms become 0 except for

$$\widetilde{w_k} e^{\widetilde{\mu_k}(X)y + \frac{1}{2}\widetilde{\Sigma_i}(X)y^2}.$$

We now use Claim 3.3.11 and Claim 3.3.9 to analyze what happens when applying the differential operator to this term. We know that

$$\widetilde{\mathcal{D}_{k-1}}^{2^{2k-2}} \dots \widetilde{\mathcal{D}_1}^{2^k} \mathcal{D}_k^{2^{k-1}} \dots \mathcal{D}_1^1(\widetilde{F}) = P(y) e^{\widetilde{\mu_k}(X)y + \frac{1}{2}\widetilde{\Sigma_i}(X)y^2}$$

where P has leading coefficient

$$\widetilde{w_k} \prod_{i=1}^{k} (\widetilde{\Sigma_k}(X) - \Sigma_i(X))^{2^{i-1}} \prod_{i=1}^{k-1} (\widetilde{\Sigma_k}(X) - \widetilde{\Sigma}_i(X))^{2^{k+i-1}}$$

and degree $2^{2k-1} - 1$. Thus,

$$\begin{aligned}
&\widetilde{\mathcal{D}_k}^{2^{2k-1}-1} \widetilde{\mathcal{D}_{k-1}}^{2^{2k-2}} \dots \widetilde{\mathcal{D}_1}^{2^k} \mathcal{D}_k^{2^{k-1}} \dots \mathcal{D}_1^1(\widetilde{F}) \\
&= (2^{2k-1} - 1)! \widetilde{w_k} \prod_{i=1}^{k} (\widetilde{\Sigma_k}(X) - \Sigma_i(X))^{2^{i-1}} \prod_{i=1}^{k-1} (\widetilde{\Sigma_k}(X) - \widetilde{\Sigma}_i(X))^{2^{k+i-1}} e^{\widetilde{\mu_k}(X)y + \frac{1}{2}\widetilde{\Sigma_i}(X)y^2}
\end{aligned}$$

and plugging in $y = 0$, we are done. ∎

Claim 3.4.8. *Write*

$$\mathcal{D}_k^{2^{2k-1}-1} \mathcal{D}_{k-1}^{2^{2k-2}} \dots \mathcal{D}_1^{2^k} \widetilde{\mathcal{D}_k}^{2^{k-1}} \dots \widetilde{\mathcal{D}_1}^1(F)$$

as a formal power series in y. Its evaluation at $y = 0$ is

$$C_k w_k \prod_{i=1}^{k} (\Sigma_k(X) - \widetilde{\Sigma}_i(X))^{2^{i-1}} \prod_{i=1}^{k-1} (\Sigma_k(X) - \Sigma_i(X))^{2^{k+i-1}}$$

where C_k is a constant depending only on k.

Proof. This can be proved using the same method as Claim 3.4.7. ■

Combining the previous two claims with Claim 3.4.5 and Claim 3.4.6, we can write the expressions for the leading coefficients as polynomial combinations of the Hermite polynomials.

Lemma 3.4.9. *Consider the polynomial*

$$\widetilde{w_k}\prod_{i=1}^{k}(\widetilde{\Sigma_k}(X)-\Sigma_i(X))^{2^{i-1}}\prod_{i=1}^{k-1}(\widetilde{\Sigma_k}(X)-\widetilde{\Sigma}_i(X))^{2^{k+i-1}}$$

It can be written in the form

$$P_0(X)+P_1(X)\widetilde{h_1}(X)+\cdots+P_m(X)\widetilde{h_m}(X)$$

where

- *m is a function of k*
- *Each of the P_i is m-simple*
- *We have*

$$P_0(X)+P_1(X)h_1(X)+\cdots+P_m(X)h_m(X)=0$$

as an algebraic identity over formal variables $X_1,\ldots,X_d,\{u_i\},\{v_i\}$.

Proof. Consider the power series

$$\widetilde{\mathcal{D}_k}^{2^{2k-1}-1}\widetilde{\mathcal{D}_{k-1}}^{2^{2k-2}}\ldots\widetilde{\mathcal{D}_1}^{2^k}\mathcal{D}_k^{2^{k-1}}\ldots\mathcal{D}_1^1(\widetilde{F})$$

Now using Claim 3.4.7 and repeating the proof of Claim 3.4.5, we get the desired. ■

Similarly, we have:

Lemma 3.4.10. *Consider the polynomial*

$$w_k \prod_{i=1}^{k} (\Sigma_k(X) - \widetilde{\Sigma}_i(X))^{2^{i-1}} \prod_{i=1}^{k-1} (\Sigma_k(X) - \Sigma_i(X))^{2^{k+i-1}}$$

It can be written in the form

$$P_0(X) + P_1(X)h_1(X) + \cdots + P_m(X)h_m(X)$$

where

- *m is a function of k*
- *Each of the P_i is m-simple*
- *We have*

$$P_0(X) + P_1(X)\widetilde{h_1}(X) + \cdots + P_m(X)\widetilde{h_m}(X) = 0$$

as an algebraic identity over formal variables $X_1, \ldots, X_d, \{u_i\}, \{v_i\}$.

Everything we've done so far has been symbolic manipulations and the claims in this section are all true as algebraic identities. We are now ready to analyze the SOS program. Note the polynomials $P_0, \ldots, P_m$ in Lemma 3.4.9 are unknown because they depend on the true parameters. This is fine because we will simply use their existence to deduce properties of pseudoexpectations that solve the SOS-system $\mathcal{S}$.

Let U be the subspace spanned by the true $\mu_1, \ldots, \mu_k$ and let V denote the subspace spanned by the true (flattened) $\Sigma_1, \ldots, \Sigma_k$. We will use $\Gamma_V, \Gamma_{V^\perp}$ to denote projections onto V and the orthogonal complement of V (and similar for $U, U^\perp$). Note that these are linear maps.

Our goal now will be to show that V is essentially contained within the span of the union of the top k principal components of the matrices

$$\widetilde{\mathbb{E}}[\widetilde{\Sigma_1}\widetilde{\Sigma_1}^T], \ldots, \widetilde{\mathbb{E}}[\widetilde{\Sigma_k}\widetilde{\Sigma_k}^T]$$

This gives us a k^2-dimensional space that essentially contains V and then we can guess the true covariance matrices via brute force search. In the first key lemma, we prove that the matrix $\widetilde{\mathbb{E}}[\widetilde{\Sigma}_i\widetilde{\Sigma}_i^T]$ lives almost entirely within the subspace V.

Lemma 3.4.11. *Let $\widetilde{\mathbb{E}}$ be a pseudoexpectation of degree C_k for some sufficiently large constant C_k depending only on k that solves $\mathcal{S}$. Consider the matrix*

$$M = \widetilde{\mathbb{E}}[\widetilde{\Sigma_k}\widetilde{\Sigma_k}^T]$$

where by this we mean we construct the $D \times D$ matrix $\widetilde{\Sigma_k}\widetilde{\Sigma_k}^T$ whose entries are quadratic in the variables $\{u\}, \{v\}$ and then take the entry-wise pseudoexpectation. Then

$$Tr_{V^\perp}(M) \leq \epsilon'^{2^{-k}} O_k(1) \left(\frac{\Delta}{w_{\min} c}\right)^{O_k(1)}$$

where $Tr_{V^\perp}(M)$ denotes the trace of M on the subspace $V^\perp$.

Proof. Using Lemma 3.4.9, we may write

$$\begin{aligned}
&\widetilde{w_k}\prod_{i=1}^{k}(\widetilde{\Sigma_k}(X) - \Sigma_i(X))^{2^{i-1}} \prod_{i=1}^{k-1}(\widetilde{\Sigma_k}(X) - \widetilde{\Sigma}_i(X))^{2^{k+i-1}} \\
&= P_1(X)(\widetilde{h_1}(X) - h_1(X)) + \cdots + P_m(X)(\widetilde{h_m}(X) - h_m(X))
\end{aligned}$$

where $m = O_k(1)$

Now we bound

$$\widetilde{\mathbb{E}}\left[\left\|v\left(P_1(X)(\widetilde{h_1}(X) - h_1(X)) + \cdots + P_m(X)(\widetilde{h_m}(X) - h_m(X))\right)\right\|^2\right]$$

Using Claim 3.3.15 and Claim 3.3.14,

$$\begin{aligned}
&\widetilde{\mathbb{E}}\left[\left\|v\left(P_1(X)(\widetilde{h_1}(X)-h_1(X))+\cdots+P_m(X)(\widetilde{h_m}(X)-h_m(X))\right)\right\|^2\right]\\
&\leq O_k(1)\sum_{i=1}^{m}\widetilde{\mathbb{E}}\left[\|v(P_i(X))\|^2\cdot\left\|v(\widetilde{h_i}(X)-h_i(X))\right\|^2\right]\\
&\leq O_k(1)\sum_{i=1}^{m}\widetilde{\mathbb{E}}\left[\|v(P_i(X))\|^2\cdot 2\left(\left\|v(\widetilde{h_i}(X)-\overline{h_i}(X))\right\|^2+\left\|v(\overline{h_i}(X)-h_i(X))\right\|^2\right)\right]
\end{aligned}$$

Where the last step is true because Claim 3.3.14 allows us to write the difference between the two sides as a sum of squares.

Now $\left\|v(\overline{h_i}(X)-h_i(X))\right\|^2$ is just a real number and is bounded above by ϵ' by assumption. We also have the constraint that

$$\left\|v(\widetilde{h_i}(X)-\overline{h_i}(X))\right\|^2\leq 100\epsilon'$$

so

$$\begin{aligned}
&\widetilde{\mathbb{E}}\left[\left\|v\left(P_1(X)(\widetilde{h_1}(X)-h_1(X))+\cdots+P_m(X)(\widetilde{h_m}(X)-h_m(X))\right)\right\|^2\right]\\
&\leq O_k(1)\epsilon'\sum_{i=1}^{m}\widetilde{\mathbb{E}}\left[\|v(P_i(X))\|^2\right]
\end{aligned}$$

Now we use the properties from Lemma 3.4.9 that each of the P_i can be written as a linear combination of a constant number of terms that are a product of some of

$$\{\mu_i(X)\},\{\Sigma_i(X)\},\{\widetilde{\mu}_i(X)\},\{\widetilde{\Sigma}_i(X)\}$$

where

- The coefficients in the linear combination are bounded by a constant depending only on k

- The number of terms in the sum depends only on k
- The number of terms in each product depends only on k

Note for each $\widetilde{\mu_i}(X)$, since we ensured that our guesses for the coefficients that go with the orthonormal basis $u_1, \ldots, u_k$ are at most Δ and we have the constraints $\|u_i\|_2^2 = 1, u_i \cdot u_j = 0$, we have

$$\|v(\widetilde{\mu_i}(X))\|^2 \preceq_{SOS} O_k(1)\Delta^2$$

where $\preceq_{SOS}$ means the difference can be written as a sum of squares. We can make similar arguments for $\widetilde{\Sigma_i}(X), \mu_i(X), \Sigma_i(X)$. Now using Claim 3.3.14 and Claim 3.3.15 we can deduce

$$\widetilde{\mathbb{E}}\left[\|v(P_i(X))\|^2\right] \leq O_k(1)\Delta^{O_k(1)}$$

Overall, we have shown

$$\widetilde{\mathbb{E}}\left[\left\|v\left(P_1(X)(\widetilde{h_1}(X) - h_1(X)) + \cdots + P_m(X)(\widetilde{h_m}(X) - h_m(X))\right)\right\|^2\right] \leq O_k(1)\epsilon'\Delta^{O_k(1)}$$

Now we examine the expression

$$\widetilde{\mathbb{E}}\left[\left\|v\left(\widetilde{w_k}\prod_{i=1}^{k}(\widetilde{\Sigma_k}(X) - \Sigma_i(X))^{2^{i-1}}\prod_{i=1}^{k-1}(\widetilde{\Sigma_k}(X) - \widetilde{\Sigma_i}(X))^{2^{k+i-1}}\right)\right\|^2\right]$$

By Claim 3.3.18 (recall $\widetilde{w_k}$ is a constant that we guess),

$$\begin{aligned}&\widetilde{\mathbb{E}}\left[\left\|v\left(\widetilde{w_k}\prod_{i=1}^{k}(\widetilde{\Sigma_k}(X) - \Sigma_i(X))^{2^{i-1}}\prod_{i=1}^{k-1}(\widetilde{\Sigma_k}(X) - \widetilde{\Sigma_i}(X))^{2^{k+i-1}}\right)\right\|^2\right] \\ &\geq \widetilde{w_k}\Omega_k(1)\widetilde{\mathbb{E}}\left[\prod_{i=1}^{k}\left(\left\|v(\widetilde{\Sigma_k}(X) - \Sigma_i(X))\right\|^2\right)^{2^{i-1}}\prod_{i=1}^{k-1}\left(\left\|v(\widetilde{\Sigma_k}(X) - \widetilde{\Sigma_i}(X))\right\|^2\right)^{2^{k+i-1}}\right]\end{aligned}$$

Note that

$$\left\|v(\widetilde{\Sigma_k}(X) - \Sigma_i(X))\right\|^2 \succeq_{SOS} \left\|\Gamma_{V^\perp}(\widetilde{\Sigma_k})\right\|^2$$

(recall that $\Gamma_{V^\perp}$ is a projection map with unknown but constant coefficients). Next, since we ensure that the coefficients B_i that we guess for the orthonormal basis satisfy $\|B_i - B_j\|_2 \geq \frac{c}{2}$, we have

$$\left\| v(\widetilde{\Sigma_k}(X) - \widetilde{\Sigma}_i(X)) \right\|^2 \succeq_{SOS} \frac{c^2}{4}$$

where we use the constraints in $\mathcal{S}$ that $\|v_i\|_2^2 = 1, v_i \cdot v_j = 0$. Overall, we conclude

$$\begin{aligned}
&\widetilde{\mathbb{E}} \left[\left\| v \left(\widetilde{w_k} \prod_{i=1}^{k} (\widetilde{\Sigma_k}(X) - \Sigma_i(X))^{2^{i-1}} \prod_{i=1}^{k-1} (\widetilde{\Sigma_k}(X) - \widetilde{\Sigma}_i(X))^{2^{k+i-1}} \right) \right\|^2 \right] \\
&\geq \Omega_k(1) \, \mathbb{E} \left[\left\| \Gamma_{V^\perp}(\widetilde{\Sigma_k}) \right\|^{2^{k+1}-2} \right] (\widetilde{w_k} c)^{O_k(1)}
\end{aligned}$$

Note

$$\begin{aligned}
&\widetilde{\mathbb{E}} \left[\left\| v \left(\widetilde{w_k} \prod_{i=1}^{k} (\widetilde{\Sigma_k}(X) - \Sigma_i(X))^{2^{i-1}} \prod_{i=1}^{k-1} (\widetilde{\Sigma_k}(X) - \widetilde{\Sigma}_i(X))^{2^{k+i-1}} \right) \right\|^2 \right] = \\
&\widetilde{\mathbb{E}} \left[\left\| v \left(P_1(X)(\widetilde{h_1}(X) - h_1(X)) + \cdots + P_m(X)(\widetilde{h_m}(X) - h_m(X)) \right) \right\|^2 \right]
\end{aligned}$$

because the inner expressions are equal symbolically. Thus

$$\widetilde{\mathbb{E}} \left[\left\| \Gamma_{V^\perp}(\widetilde{\Sigma_k}) \right\|^{2^{k+1}-2} \right] \leq O_k(1) \epsilon' \left(\frac{\Delta}{w_{\min} c} \right)^{O_k(1)}$$

Thus

$$\widetilde{\mathbb{E}} \left[\left\| \Gamma_{V^\perp}(\widetilde{\Sigma_k}) \right\|^2 \right] \leq \epsilon'^{2^{-k}} O_k(1) \left(\frac{\Delta}{w_{\min} c} \right)^{O_k(1)}$$

It remains to note that

$$\mathrm{Tr}_{V^\perp}(M) = \widetilde{\mathbb{E}} \left[\left\| \Gamma_{V^\perp}(\widetilde{\Sigma_k}) \right\|^2 \right]$$

and we are done. ■

In the next key lemma, we prove that any vector that has nontrivial projection onto V must also have nontrivial projection onto $\widetilde{\mathbb{E}}[\widetilde{\Sigma}_i \widetilde{\Sigma}_i^T]$ for some i.

Lemma 3.4.12. *Let $\widetilde{\mathbb{E}}$ be a pseudoexpectation of degree C_k for some sufficiently large constant C_k depending only on k that solves $\mathcal{S}$. Consider the matrix*

$$N = \sum_{i=1}^{k} \widetilde{\mathbb{E}}[\widetilde{\Sigma}_i \widetilde{\Sigma}_i{}^T]$$

where by this we mean we construct the $D \times D$ matrix whose entries are quadratic in the variables $\{u\}, \{v\}$ and then take the entry-wise pseudoexpectation. Then for any unit vector $z \in \mathbb{R}^D$,

$$z^T N z \geq \left(\frac{w_{\min}(z \cdot \Sigma_k)^{O_k(1)} - O_k(1)\epsilon' \Delta^{O_k(1)}}{O_k(1)\Delta^{O_k(1)}} \right)^2$$

as long as

$$w_{\min}(z \cdot \Sigma_k)^{O_k(1)} > O_k(1)\epsilon' \Delta^{O_k(1)}$$

Proof. Using Lemma 3.4.10, we may write

$$\begin{aligned} &w_k \prod_{i=1}^{k} (\Sigma_k(X) - \widetilde{\Sigma}_i(X))^{2^{i-1}} \prod_{i=1}^{k-1} (\Sigma_k(X) - \Sigma_i(X))^{2^{k+i-1}} \\ &= P_1(X)(\widetilde{h_1}(X) - h_1(X)) + \cdots + P_m(X)(\widetilde{h_m}(X) - h_m(X)) \end{aligned}$$

where $m = O_k(1)$.

Using the same method as the proof in Lemma 3.4.11, we have

$$\widetilde{\mathbb{E}} \left[\left\| v \left(P_1(X)(\widetilde{h_1}(X) - h_1(X)) + \cdots + P_m(X)(\widetilde{h_m}(X) - h_m(X)) \right) \right\|^2 \right] \leq O_k(1)\epsilon' \Delta^{O_k(1)}$$

Now by Claim 3.3.18,

$$\begin{aligned}
&\widetilde{\mathbb{E}}\left[\left\|v\left(w_k\prod_{i=1}^{k}(\Sigma_k(X)-\widetilde{\Sigma}_i(X))^{2^{i-1}}\prod_{i=1}^{k-1}(\Sigma_k(X)-\Sigma_i(X))^{2^{k+i-1}}\right)\right\|^2\right]\\
&\geq w_k\widetilde{\mathbb{E}}\left[\prod_{i=1}^{k}\left(\left\|v\left(\Sigma_k(X)-\widetilde{\Sigma}_i(X)\right)\right\|^2\right)^{2^{i-1}}\prod_{i=1}^{k-1}\left(\|(\Sigma_k(X)-\Sigma_i(X))\|^2\right)^{2^{k+i-1}}\right]\\
&\geq w_k\widetilde{\mathbb{E}}\left[\prod_{i=1}^{k}\left((z\cdot\Sigma_k-z\cdot\widetilde{\Sigma}_i)^2\right)^{2^{i-1}}c^{O_k(1)}\right]
\end{aligned}$$

where the second inequality is true because

$$\left\|v\left(\Sigma_k(X)-\widetilde{\Sigma}_i(X)\right)\right\|^2\succeq_{SOS}(z\cdot\Sigma_k-z\cdot\widetilde{\Sigma}_i)^2$$

Now we claim

$$\widetilde{\mathbb{E}}\left[\prod_{i=1}^{k}\left((z\cdot\Sigma_k-z\cdot\widetilde{\Sigma}_i)^2\right)^{2^{i-1}}\right]\geq(z\cdot\Sigma_k)^{O_k(1)}-O_k(1)\Delta^{O_k(1)}\sqrt{\widetilde{\mathbb{E}}\left[\sum_i(z\cdot\widetilde{\Sigma}_i)^2\right]}$$

To see this, first recall that $z\cdot\Sigma_k$ is just a constant. Next, we can expand the LHS into a sum of monomials in the $z\cdot\widetilde{\Sigma}_i$. In particular, we can write the expansion in the form

$$(z\cdot\Sigma_k)^{O_k(1)}+\sum_i(z\cdot\widetilde{\Sigma}_i)P_i(z\cdot\widetilde{\Sigma_1},\ldots,z\cdot\widetilde{\Sigma_k})$$

wjhere P is some polynomial in k variables. We can upper bound the coefficients of the polynomial in terms of Δ, k and we also know that

$$(z\cdot\widetilde{\Sigma}_i)^2\preceq_{SOS}O_k(1)\Delta^{O(1)}$$

due to the constraints in our system. Thus, we can bound the pseudoexpectation

$$-\widetilde{\mathbb{E}}\left[\sum_i(z\cdot\widetilde{\Sigma}_i)P_i(z\cdot\widetilde{\Sigma_1},\ldots,z\cdot\widetilde{\Sigma_k})\right]\leq O_k(1)\Delta^{O_k(1)}\sqrt{\widetilde{\mathbb{E}}\left[\sum_i(z\cdot\widetilde{\Sigma}_i)^2\right]}$$

via Cauchy Schwarz. Putting everything together the same way as in Lemma 3.4.11, we deduce

$$\widetilde{\mathbb{E}}\left[\sum_i (z \cdot \widetilde{\Sigma}_i)^2\right] \geq \left(\frac{w_{\min}(z \cdot \Sigma_k)^{O_k(1)} - O_k(1)\epsilon' \Delta^{O_k(1)}}{O_k(1)\Delta^{O_k(1)}}\right)^2$$

and now we are done. ■

Putting Lemmas 3.4.11 and 3.4.12 together, we now prove that V is essentially contained within the span of the union of the top principal components of $\widetilde{\mathbb{E}}[\widetilde{\Sigma}_i \widetilde{\Sigma}_i^T]$ over all i.

Lemma 3.4.13. *For each i, let M_i be the $D \times D$ matrix given by*

$$M_i = \widetilde{\mathbb{E}}[\widetilde{\Sigma}_i \widetilde{\Sigma_i^T}].$$

Assume that for a sufficiently small function f depending only on k,

$$\Delta \leq \epsilon'^{-f(k)}$$

$$w_{\min}, c \geq \epsilon'^{f(k)}$$

Let V_i be the subspace spanned by the top k singular vectors of M_i. Then for all i, the projection of the true covariance matrix Σ_i onto the orthogonal complement of $\mathsf{spn}(V_1, \ldots, V_k)$ has length at most $\epsilon'^{\Omega_k(1)}$.

Proof. Assume for the sake of contradiction that the desired statement is false for Σ_i. Let z be the projection of Σ_i onto the orthogonal complement of $\mathsf{spn}(V_1, \ldots, V_k)$. By Lemma 3.4.12,

$$\sum_j z^T M_j z \geq \left(\frac{w_{\min}(z \cdot \Sigma_i)^{O_k(1)} - O_k(1)\epsilon' \Delta^{O_k(1)}}{O_k(1)\Delta^{O_k(1)}}\right)^2 \tag{3.2}$$

so there is some j for which

$$z^T M_j z \geq \frac{1}{k}\left(\frac{w_{\min}(z \cdot \Sigma_i)^{O_k(1)} - O_k(1)\epsilon' \Delta^{O_k(1)}}{O_k(1)\Delta^{O_k(1)}}\right)^2$$

On the other hand, Lemma 3.4.11 implies that the sum of the singular values of M_j outside

the top k is at most

$$\epsilon'^{2^{-k}} O_k(1) \left(\frac{\Delta}{w_{\min} c}\right)^{O_k(1)}$$

Since z is orthogonal to the span of the top-k singular vectors of M_j, we get

$$z^T M_j z \leq \epsilon'^{2^{-k}} O_k(1) \left(\frac{\Delta}{w_{\min} c}\right)^{O_k(1)} \|z\|_2^2 \tag{3.3}$$

Note $z \cdot \Sigma_i = \|z\|_2^2$ since z is a projection of Σ_i onto a subspace. Now combining (3.2) and (3.3) we get a contradiction unless

$$\|z\|_2 \leq \epsilon'^{\Omega_k(1)}$$

■

Finishing Up: Finding the Covariances and then the Means

Now we can brute-force search over the subspace spanned by the union of the top k singular vectors of $M_1, \ldots, M_k$. Note that the SOS system $\mathcal{S}$ is clearly feasible as it is solved when the u_i, v_i form orthonormal bases for the true subspaces and the $\widetilde{w}_i, A_i, B_i$ are within $\epsilon'^{O_k(1)}$ of the true values (i.e. the values needed to express the true means and covariances in the orthonormal basis given by the u_i, v_i).

Thus, brute forcing over an $\epsilon'^{O_k(1)}$-net for the $\widetilde{w}_i, A_i, B_i$, we will find a feasible solution. By Lemma 3.4.12 and Lemma 3.4.13, once we find any feasible solution, we will be able to obtain a set of $(1/\epsilon')^{O_k(1)}$ estimates at least one of which, say $\overline{\Sigma_1}, \ldots, \overline{\Sigma_k}$, satisfies

$$\left\|\Sigma_i - \overline{\Sigma_i}\right\|_2^2 \leq \epsilon'^{\Omega_k(1)}$$

for all i. With these estimates we will now solve for the means. Note we can assume that our covariance estimates are exactly correct because we can pretend that the true mixture is actually $N(\mu_1, \overline{\Sigma_1}), \ldots, N(\mu_k, \overline{\Sigma_k})$ and our estimates for the Hermite polynomials of this mixture will be off by at most $O_k(1)\epsilon'^{\Omega_k(1)}$. Thus, making this assumption will only affect the dependence on ϵ' that we get at the end. From now on we can write Σ_i to denote the

true covariances and treat these as known quantities.

Now we set up the same system as in Section 3.4.2 except we no longer have the variables $v_1, \ldots, v_k$ and no longer have the $\widetilde{\Sigma}_i$. These will instead be replaced by real values from Σ_i. Formally:

Definition 3.4.14 (SOS program for learning means)**.** *We will have the following variables*

- $u_1 = (u_{11}, \ldots, u_{1d}), \ldots, u_k = (u_{k1}, \ldots, u_{kd})$

In the above $u_1, \ldots, u_k \in \mathbb{R}^d$. We guess coefficients a_{ij} where $i, j \in [k]$ expressing the means in this orthonormal basis. We ensure that the guesses satisfy the property that for every pair of vectors $A_i = (a_{i1}, \ldots, a_{ik}), A_j = (a_{j1}, \ldots, a_{jk})$ either $A_i = A_j$ or

$$\|A_i - A_j\|_2 \geq \frac{c}{2}$$

We ensure that

$$\|A_i\|_2 \leq 2\Delta$$

We also guess the mixing weights $\widetilde{w_1}, \ldots, \widetilde{w_k}$ and ensure that our guesses are all at least $w_{\min}/2$.

Now we set up the constraints. Let C be a sufficiently large integer depending only on k. Define $\widetilde{\mu}_i = a_{i1}u_1 + \cdots + a_{ik}u_k$. These are linear expressions in the variables that we are solving for. Now consider the hypothetical mixture with mixing weights $\widetilde{w}_i$, means $\widetilde{\mu}_i$, and covariances $I + \Sigma_i$. The Hermite polynomials for this hypothetical mixture $\widetilde{h}_i(X)$ can be written as formal polynomials in $X = (X_1, \ldots, X_d)$ with coefficients that are polynomials in u. Note that we can explicitly write down these Hermite polynomials. The set of constraints for our SOS system is as follows:

- $\|u_i\|_2^2 = 1$ *for all* $1 \leq i \leq k$
- $u_i \cdot u_j = 0$ *for all* $i \neq j$

- *For all* $p = 1, 2, \ldots, C$

$$\left\| v(\widetilde{h_p}(X) - \overline{h_p}(X)) \right\|^2 \leq 100\epsilon'$$

Now we can repeat the same arguments from Section 3.4.3 to prove that once we find a feasible solution, we can recover the span of the μ_i. The important generating functions are

$$F(y) = \sum_{i=1}^{k} w_i e^{\mu_i(X)y + \frac{1}{2}\Sigma_i(X)y^2}$$
$$\widetilde{F}(y) = \sum_{i=1}^{k} \widetilde{w}_i e^{\widetilde{\mu_i}(X)y + \frac{1}{2}\Sigma_i(X)y^2}$$

Define the differential operators as before except with $\widetilde{\Sigma}_i$ replaced with Σ_i. Let $\mathcal{D}_i$ denote the differential operator $(\partial - (\mu_i(X) + \Sigma_i(X)y))$ and $\widetilde{\mathcal{D}_i}$ denote the differential operator $(\partial - (\widetilde{\mu}_i(X) + \Sigma_i(X)y))$. All derivatives are taken with respect to y. The two key differential operators to consider are

$$\widetilde{\mathcal{D}_k}^{2^{2k-1}-2^{k-1}-1}\widetilde{\mathcal{D}_{k-1}}^{2^{2k-2}} \ldots \widetilde{\mathcal{D}_1}^{2^k}\mathcal{D}_k^{2^{k-1}} \ldots \mathcal{D}_1^1$$
$$\mathcal{D}_k^{2^{2k-1}-2^{k-1}-1}\mathcal{D}_{k-1}^{2^{2k-2}} \ldots \mathcal{D}_1^{2^k}\widetilde{\mathcal{D}_k}^{2^{k-1}} \ldots \widetilde{\mathcal{D}_1}^1$$

Note the change to $2^{2k-1} - 2^{k-1} - 1$ from $2^{2k-1} - 1$ in the exponent of the first term. This is because when operating on $P(y)e^{\mu_k(X)y + \frac{1}{2}\Sigma_k(X)y^2}$ for some polynomial P, the operator $\mathcal{D}_k$ reduces the degree of P by 1 while the operator $\widetilde{D_k}$ does not change the degree of P (whereas before this operator increased the degree of the formal polynomial P). Similar to Claim 3.4.7 and Claim 3.4.8 in Section 3.4.3, we have

Claim 3.4.15. *Write*

$$\widetilde{\mathcal{D}_k}^{2^{2k-1}-2^{k-1}-1}\widetilde{\mathcal{D}_{k-1}}^{2^{2k-2}} \ldots \widetilde{\mathcal{D}_1}^{2^k}\mathcal{D}_k^{2^{k-1}} \ldots \mathcal{D}_1^1(\widetilde{F})$$

as a formal power series in y. Its evaluation at $y = 0$ is

$$C_k \widetilde{w_k}(\widetilde{\mu_k}(X) - \mu_k(X))^{2^{k-1}} \prod_{i=1}^{k-1} (\Sigma_k(X) - \Sigma_i(X))^{2^{i-1}} \prod_{i=1}^{k-1} (\Sigma_k(X) - \Sigma_i(X))^{2^{k+i-1}}$$

where C_k is a constant depending only on k.

Claim 3.4.16. *Write*

$$\mathcal{D}_k^{2^{2k-1}-2^{k-1}-1} \mathcal{D}_{k-1}^{2^{2k-2}} \dots \mathcal{D}_1^{2^k} \widetilde{\mathcal{D}_k}^{2^{k-1}} \dots \widetilde{\mathcal{D}_1}^1(F)$$

as a formal power series in y. Its evaluation at $y = 0$ is

$$C_k w_k(\widetilde{\mu_k}(X) - \mu_k(X))^{2^{k-1}} \prod_{i=1}^{k-1} (\Sigma_k(X) - \Sigma_i(X))^{2^{i-1}} \prod_{i=1}^{k-1} (\Sigma_k(X) - \Sigma_i(X))^{2^{k+i-1}}$$

where C_k is a constant depending only on k.

Now repeating the arguments in Lemmas 3.4.9, 3.4.10, 3.4.11,3.4.12, 3.4.13, we can prove that for any feasible solution, the subspace spanned by the top k singular vectors of each of $\widetilde{\mathbb{E}}[\widetilde{\mu_1}\widetilde{\mu_1}^T], \dots, \widetilde{\mathbb{E}}[\widetilde{\mu_k}\widetilde{\mu_k}^T]$ approximately contains all of $\mu_1, \dots, \mu_k$. We can now brute force search over this subspace (and since we are already brute-force searching over the mixing weights), we will output some set of candidate components that are close to the true components.

All Pairs of Parameters are Equal or Separated

In the case where some pairs of parameters may be equal (but pairs (μ_i, Σ_i) and (μ_j, Σ_j) cannot be too close), we can repeat essentially the same arguments from the previous section but with minor adjustments in the number of times we are applying each differential operator.

We can assume that our guesses for the coefficients A_i, B_i satisfy the correct equality pattern in the sense that $A_i = A_j$ if and only if $\mu_i = \mu_j$ and otherwise $\|A_i - A_j\| \geq c/2$ and similar for the parameters B_i. This is because there are only $O_k(1)$ different equality patterns.

Now without loss of generality let $\{\Sigma_1, \ldots, \Sigma_j\}$ $(j < k)$ be the set of covariance matrices that are equal to Σ_k. The key differential operators to consider are

$$\widetilde{\mathcal{D}_k}^{2^{2k-1}-1-2^k-\cdots-2^{k+j}} \widetilde{\mathcal{D}_{k-1}}^{2^{2k-2}} \ldots \widetilde{\mathcal{D}_1}^{2^k} \mathcal{D}_k^{2^{k-1}} \ldots \mathcal{D}_1^1$$
$$\mathcal{D}_k^{2^{2k-1}-1-2^0-\cdots-2^j} \mathcal{D}_{k-1}^{2^{2k-2}} \ldots \mathcal{D}_1^{2^k} \widetilde{\mathcal{D}_k}^{2^{k-1}} \ldots \widetilde{\mathcal{D}_1}^1$$

Similar to Claim 3.4.7 and Claim 3.4.8, we get

Claim 3.4.17. *Let $\{\Sigma_1, \ldots, \Sigma_j\}$ $(j < k)$ be the set of covariance matrices that are equal to Σ_k. Note this also implies $\{\widetilde{\Sigma_1}, \ldots, \widetilde{\Sigma_j}\}$ are precisely the subset of $\{\widetilde{\Sigma}_i\}$ that are equal to $\widetilde{\Sigma_k}$. Write*

$$\widetilde{\mathcal{D}_k}^{2^{2k-1}-1-2^k-\cdots-2^{k+j-1}} \widetilde{\mathcal{D}_{k-1}}^{2^{2k-2}} \ldots \widetilde{\mathcal{D}_1}^{2^k} \mathcal{D}_k^{2^{k-1}} \ldots \mathcal{D}_1^1(\widetilde{F})$$

as a formal power series in y. Its evaluation at $y = 0$ is

$$C_k \widetilde{w_k} \prod_{i=1}^{k} (\widetilde{\Sigma_k}(X) - \Sigma_i(X))^{2^{i-1}} \prod_{i=1}^{j} (\widetilde{\mu_k}(X) - \widetilde{\mu}_i(X))^{2^{k+i-1}} \prod_{i=j+1}^{k-1} (\widetilde{\Sigma_k}(X) - \widetilde{\Sigma}_i(X))^{2^{k+i-1}}$$

where C_k is a constant depending only on k.

Claim 3.4.18. *Let $\{\Sigma_1, \ldots, \Sigma_j\}$ $(j < k)$ be the set of covariance matrices that are equal to Σ_k. Note this also implies $\{\widetilde{\Sigma_1}, \ldots, \widetilde{\Sigma_j}\}$ are precisely the subset of $\{\widetilde{\Sigma}_i\}$ that are equal to $\widetilde{\Sigma_k}$. Write*

$$\mathcal{D}_k^{2^{2k-1}-1-2^k-\cdots-2^{k+j-1}} \mathcal{D}_{k-1}^{2^{2k-2}} \ldots \mathcal{D}_1^{2^k} \widetilde{\mathcal{D}_k}^{2^{k-1}} \ldots \widetilde{\mathcal{D}_1}^1(F)$$

as a formal power series in y. Its evaluation at $y = 0$ is

$$C_k \widetilde{w_k} \prod_{i=1}^{k} (\Sigma_k(X) - \widetilde{\Sigma}_i(X))^{2^{i-1}} \prod_{i=1}^{j} (\mu_k(X) - \mu_i(X))^{2^{k+i-1}} \prod_{i=j+1}^{k-1} (\Sigma_k(X) - \Sigma_i(X))^{2^{k+i-1}}$$

where C_k is a constant depending only on k.

Now we can repeat the arguments in Lemmas 3.4.9, 3.4.10, 3.4.11,3.4.12, 3.4.13. The key

point is that the constraints in our SOS program give explicit values for

$$\|v(\widetilde{\mu}_i(X) - \widetilde{\mu}_j(X))\|^2$$
$$\left\|v(\widetilde{\Sigma}_i(X) - \widetilde{\Sigma_j}(X))\right\|^2$$

in terms of A_i, B_i (which are explicit real numbers). We can then repeat the arguments in Section 3.4.3 (with appropriate modifications to the number of times we apply each differential operator) to find the means.

3.5 Robust Moment Estimation

In Section 3.4, we showed how to learn the parameters of a mixture of Gaussians $\mathcal{M}$ with components that are not too far apart when we are given estimates for the Hermite polynomials. In this section, we show how to estimate the Hermite polynomials from an ϵ-corrupted sample. Putting the results together, we will get a robust learning algorithm in the case when the components are not too far apart.

While the closeness of components in Section 3.4 is defined in terms of parameter distance, we will need to reason about TV-distance between components in order to integrate our results into our full learning algorithm. We begin with a definition.

Definition 3.5.1. *We say a mixture of Gaussians $w_1G_1 + \cdots + w_kG_k$ is δ-well-conditioned if*

1. *Let $\mathcal{G}$ be the graph on $[k]$ obtained by connecting two nodes i, j if $d_{\mathsf{TV}}(G_i, G_j) \leq 1 - \delta$. Then $\mathcal{G}$ is connected*

2. *$d_{\mathsf{TV}}(G_i, G_j) \geq \delta$ for all $i \neq j$*

3. *$w_{\min} \geq \delta$*

The main theorem that we will prove in this section is as follows.

Theorem 3.5.2. *There is a function $f(k) > 0$ depending only on k such that given an ϵ-corrupted sample from a δ-well-conditioned mixture of Gaussians*

$$\mathcal{M} = w_1 N(\mu_1, \Sigma_1) + \cdots + w_k N(\mu_k, \Sigma_k)$$

where $\delta \geq \epsilon^{f(k)}$, there is a polynomial time algorithm that outputs a set of $(1/\epsilon)^{O_k(1)}$ candidate mixtures $\{\widetilde{w_1 N}(\widetilde{\mu_1}, \widetilde{\Sigma_1}) + \cdots + \widetilde{w_k} N(\widetilde{\mu_k}, \widetilde{\Sigma_k}\}$ and with high probability, at least one of them satisfies that for all i:

$$|w_i - \widetilde{w_i}| + d_{\mathsf{TV}}(N(\mu_i, \Sigma_i), N(\widetilde{\mu_i}, \widetilde{\Sigma_i})) \leq \mathrm{poly}(\epsilon)$$

3.5.1 Distance between Gaussians

As mentioned earlier, we will first introduce a few tools for relating parameter distance and TV distance between Gaussians.

The following is a standard fact.

Claim 3.5.3. *For two Gaussians $N(\mu_1, \Sigma_1), N(\mu_2, \Sigma_2)$*

$$d_{\mathsf{TV}}(N(\mu_1, \Sigma_1), N(\mu_2, \Sigma_2)) = O\left(\left((\mu_1 - \mu_2)^T \Sigma_1^{-1} (\mu_1 - \mu_2)\right)^{1/2} + \left\|\Sigma_1^{-1/2} \Sigma_2 \Sigma_1^{-1/2} - I\right\|_F\right)$$

Proof. See e.g. Fact 2.1 in [65]. ■

Next, we will prove a bound in the opposite direction, that when Gaussians are not too far apart in TV distance, then their parameters also cannot be too far apart.

Lemma 3.5.4. *Let $\mathcal{M}$ be a mixture of k Gaussians that is δ-well conditioned. Let Σ be the covariance matrix of the mixture. Then*

1. *$\Sigma_i \leq \mathrm{poly}(\delta)^{-1}\Sigma$ for all components of the mixture*
2. *$\Sigma_i \geq \mathrm{poly}(\delta)\Sigma$ for all components of the mixture*
3. *For any two components i, j, we have $\left\|\Sigma^{-1/2}(\mu_i - \mu_j)\right\| \leq \mathrm{poly}(\delta)^{-1}$*

4. For any two components i, j, we have $\left\|\Sigma^{-1/2}(\Sigma_i - \Sigma_j)\Sigma^{-1/2}\right\|_2 \leq \mathrm{poly}(\delta)^{-1}$

where the coefficients and degrees of the polynomials may depend only on k.

Proof. The statements are invariant under linear transformations so without loss of generality let $\Sigma = I$. Assume for the sake of contradiction that the first condition is failed. Then there is some direction v such that say

$$v^T \Sigma_1 v \geq \delta^{-10k}$$

There must be some $i \in [k]$ such that $v^T \Sigma_i v \leq 1$ since otherwise the variance of the mixture in direction v would be bigger than 1. Now we claim that i and 1 cannot be connected in $\mathcal{G}$, the graph defined in Definition 3.5.1. To see this, if they were connected, then there must be two vertices j_1, j_2 that are consecutive along the path between 1 and i such that

$$\frac{v^T \Sigma_{j_1} v}{v^T \Sigma_{j_2} v} \geq \delta^{-10}$$

But then $d_{\mathsf{TV}}(G_{j_1}, G_{j_2}) \geq 1 - \delta$. To see this, let $\sqrt{v^T \Sigma_{j_2} v} = c$. We can project both Gaussians onto the direction v and note that the Gaussian G_{j_1} is spread over width $\delta^{-5}c$ whereas the Gaussian G_{j_2} is essentially contained in a strip of width $O(\log 1/\delta)c$.

Now we may assume that the first condition is satisfied. Now we consider when the third condition is failed. Assume that

$$\|(\mu_i - \mu_j)\| \geq k\delta^{-20k}$$

Now let v be the unit vector in direction $\mu_i - \mu_j$. Projecting the Gaussians G_i, G_j onto direction v and considering the path between them, we must find j_1, j_2 that are connected such that

$$\|(\mu_{j_1} - \mu_{j_2})\| \geq \delta^{-20k}$$

Now, using the fact that the first condition must be satisfied (i.e. $v^T \Sigma_{j_1} v, v^T \Sigma_{j_2} v \leq \delta^{-10k}$) we get that $d_{\mathsf{TV}}(G_{j_1}, G_{j_2}) \geq 1 - \delta$, a contradiction.

Now we may assume that the first and third conditions are satisfied. Assume now that the second condition is not satisfied. Without loss of generality, there is some vector v such that

$$v^T \Sigma_1 v \leq (\delta/k)^{10^2 k}$$

If there is some component i such that

$$v^T \Sigma_i v \geq (\delta/k)^{50k}$$

then comparing the Gaussians along the path between i and 1 in the graph $\mathcal{G}$, we get a contradiction. Thus, we now have

$$v^T \Sigma_i v \leq (\delta/k)^{50}$$

for all components. Note that the covariance of the entire mixture is the identity. Thus, there must be two components with

$$|v \cdot \mu_i - v \cdot \mu_j| \geq \frac{1}{2k}.$$

Taking the path between i and j, we must be able to find two consecutive vertices j_1, j_2 such that

$$|v \cdot \mu_{j_1} - v \cdot \mu_{j_2}| \geq \frac{1}{2k^2}.$$

However, we then get $d_{\mathsf{TV}}(G_{j_1}, G_{j_2}) > 1 - \delta$, a contradiction.

Now we consider when the first three conditions are all satisfied. Using the first two conditions, we have bounds on the smallest and largest singular value of $\Sigma_i^{1/2} \Sigma_j^{-1/2}$ for all i, j. Thus,

$$\|\Sigma_i - \Sigma_j\|_2 \leq \mathrm{poly}(\delta)^{-1} \left\| I - \Sigma_i^{-1/2} \Sigma_j \Sigma_i^{-1/2} \right\|_2$$

for all i, j. However if for some i, j that are connected in $\mathcal{G}$, we have

$$\|(\Sigma_i - \Sigma_j)\|_2 \geq (k/\delta)^{10^4}$$

then we would have

$$\left\|I - \Sigma_i^{-1/2}\Sigma_j\Sigma_i^{-1/2}\right\|_2 \geq (k/\delta)^{10^3}$$

and this would contradict the assumption that $d_{\mathsf{TV}}(G_i, G_j) \leq 1 - \delta$ (this follows from the same argument as in Lemma 3.2 of [65]). Now using triangle inequality along each path, we deduce that for all i, j

$$\|(\Sigma_i - \Sigma_j)\|_2 \leq (k/\delta)^{10^5}$$

completing the proof. ■

As a corollary to the previous lemma, in a δ-well conditioned mixture, all component means and covariances are close to the mean and covariance of the overall mixture.

Corollary 3.5.5. *Let $\mathcal{M}$ be a mixture of k Gaussians that is δ-well conditioned. Let μ, Σ be the mean and covariance matrix of the mixture. Then we have for all i*

- $\left\|\Sigma^{-1/2}(\mu - \mu_i)\right\|_2 \leq \mathrm{poly}(\delta)^{-1}$
- $\left\|\Sigma^{-1/2}(\Sigma - \Sigma_i)\Sigma^{-1/2}\right\|_2 \leq \mathrm{poly}(\delta)^{-1}$

Proof. The statement is invariant under linear transformation so we may assume $\Sigma = I$ and $\mu = 0$. Then noting

$$\mu_i = \mu + w_1(\mu_i - \mu_1) + \cdots + w_k(\mu_i - \mu_k)$$

and using Lemma 3.5.4, we have proved the first part. Now for the second part, note $\Sigma = \sum_{i=1}^k w_i(\Sigma_i + \mu_i\mu_i^T)$ and hence we have

$$\Sigma = \Sigma_i + w_1(\Sigma_1 - \Sigma_i) + \cdots + w_k(\Sigma_k - \Sigma_i) + \sum_{i=1}^k w_i\mu_i\mu_i^T$$

and using Lemma 3.5.4 and the first part, we are done.

■

3.5.2 Hermite Polynomial Estimation

Now we show how to estimate the Hermite polynomials of a δ-well-conditioned mixture $\mathcal{M}$ if we are given an ϵ-corrupted sample (where $\delta \geq \epsilon^{f(k)}$ for some sufficiently small function $f(k) > 0$ depending only on k). Our algorithm will closely mirror the algorithm in [65].

The first step will be to show that we can robustly estimate the mean and covariance of the mixture $\mathcal{M}$ and then we will use these estimates to compute a linear transformation to place the mixture in isotropic position.

Lemma 3.5.6. *There is a sufficiently small function $f(k)$ depending only on k such that given a ϵ-corrupted sample from a δ-well-conditioned mixture of Gaussians $\mathcal{M}$ with true mean and covariance μ, Σ respectively, where $\delta \geq \epsilon^{f(k)}$, then with high probability we can output estimates $\widehat{\mu}$ and $\widehat{\Sigma}$ such that*

1. $\left\|\Sigma^{-1/2}(\widehat{\Sigma} - \Sigma)\Sigma^{-1/2}\right\|_2 \leq \epsilon^{\Omega_k(1)}$
2. $\left\|\Sigma^{-1/2}(\widehat{\mu} - \mu)\right\|_2 \leq \epsilon^{\Omega_k(1)}$

Proof. This can be proven using a similar argument to Proposition 4.1 in [65]. First we will estimate the covariance of the mixture. Note that the statement is invariant under linear transformation (and the robust estimation algorithtm that we will use, Theorem 2.4 in [65], is also invariant under linear transformation), so it suffices to consider when $\Sigma = I$. Let the components of the mixture be $G_1, \ldots, G_k$. Note that by pairing up our samples, we have access to a 2ϵ-corrupted sample from the distribution $\mathcal{M} - \mathcal{M}'$ (i.e. the difference of two independent samples from $\mathcal{M}$). For each such sample say $Y \sim \mathcal{M} - \mathcal{M}'$, $\Sigma = 0.5\,\mathbb{E}[YY^T]$. We will now show that $Z = YY^T$ where Z is flattened into a vector, has bounded covariance. Note that we can view Y as being sampled from a mixture of $O(k^2)$ Gaussians $G_i - G_j$ (where we may have $i = j$). We now prove that

- For $Y \sim G_i - G_j$ and $Z = YY^T$, $\mathbb{E}[Z \otimes Z] - \mathbb{E}[Z] \otimes \mathbb{E}[Z] \leq \text{poly}(\delta)^{-1} I$

- For $Y \sim G_i - G_j, Y' \sim G_{i'} - G_{j'}$ and $Z = YY^T, Z' = Y'Y'^T$, $\|\mathbb{E}[Z - Z']\|_2^2 = \text{poly}(\delta)^{-1}$

Using Lemma 3.5.4 and Corollary 3.5.5, we have $\text{poly}(\delta)^{-1}$ bounds on $\|\mu_i\|_2$, $\|\Sigma_i\|_{\mathsf{op}}$ and $\|\Sigma_i - \Sigma_j\|_2$ for all i, j. We can now follow the same argument as Proposition 4.1 in [65] to bound the above two quantities. With these bounds, by Theorem 2.4 in [65], we can robustly estimate the covariance. Once we have an estimate for the covariance $\widehat{\Sigma}$, we can apply the linear transformation $\widehat{\Sigma}^{-1/2}$ and robustly estimate the mean (which now has covariance close to identity). ■

Using the above, we can place our mixture in isotropic position. This mirrors Proposition 4.2 in [65].

Corollary 3.5.7. *There is a sufficiently small function $f(k)$ depending only on k such that given a ϵ-corrupted sample from a δ-well-conditioned mixture of Gaussians $\mathcal{M} = w_1 G_1 + \cdots + w_k G_k$ with mean and covariance μ, Σ where $\delta \geq \epsilon^{f(k)}$, there is a polynomial time algorithm that with high probability outputs an invertible linear transformation L so that*

1. $\|L(\mu)\|_2 \leq \text{poly}(\epsilon)$

2. $\|I - L(\Sigma)\|_2 \leq \text{poly}(\epsilon)$

Proof. We can first obtain estimates $\widehat{\mu}$ and $\widehat{\Sigma}$ using Lemma 3.5.6. We can then apply the linear transformation

$$L(x) = \widehat{\Sigma}^{-1/2}(x - \widehat{\mu})$$

It follows from direct computation that this transformation satisfies the desired properties. ■

Once our mixture is placed in isotropic position, we will estimate the Hermite polynomials and then we will be able to use Theorem 3.4.1. The following lemma can be easily derived from the results in [65] (see Lemmas 2.7,2.8 and 5.2 there).

Lemma 3.5.8. *Let $\mathcal{M}$ be a mixture of Gaussians $w_1 N(\mu_1, I + \Sigma_1) + \cdots + w_k N(\mu_k, I + \Sigma_k)$. Then*

$$\left\| \mathop{\mathbb{E}}_{z \sim \mathcal{M}} (v_X(H_m(X.z)) \otimes v_X(H_m(X.z)))] \right\|_2 = O_m(1 + \max \|\Sigma_i\|_2 + \max \|\mu_i\|)^{2m}$$

where $H_m(X, z)$ is defined as in definition 3.3.4 and $v_X(H_m(X.z))$ denotes vectorizing as a polynomial in X so that the entries of the vector are polynomials in z.

Kane [65] works with Hermite polynomial tensors, which are tensorized versions of the Hermite polynomials we are using. It is clear that these two notions are equivalent up to $O_k(1)$ factors as long as m is $O_k(1)$ (writing them as formal polynomials instead of tensors simply collapses symmetric entries of the tensor but this collapses at most $O_m(1)$ entries together at once).

We can now combine everything in this section with Theorem 3.4.1 to complete the proof of Theorem 3.5.2.

Proof of Theorem 3.5.2. We can split the samples into $O(1)$ parts that are each $O(1)\epsilon$ corrupted samples. First, we use Corollary 3.5.7 to compute a transformation L that places the mixture in nearly isotropic position. Now Lemma 3.5.4 and Corollary 3.5.5 gives us bounds on how far each of the means is from 0 and how far each of the covariances is from I. We can apply Lemma 3.5.8 and standard results from robust estimation of bounded covariance distributions (see e.g. Theorem 2.2 in [65]) to obtain estimates $\overline{h_{m,L(\mathcal{M})}}(X)$ for the Hermite polynomials of the mixture $L(\mathcal{M})$ such that

$$\left\| v\left(\overline{h_{m,L(\mathcal{M})}}(X) - h_{m,L(\mathcal{M})}(X)\right) \right\|_2 \leq \text{poly}(\epsilon)$$

where m is bounded as a function of k. Now we must verify that the remaining hypotheses of Theorem 3.4.1 are satisfied with $\epsilon' = \text{poly}(\epsilon)$ for the transformed mixture $L(\mathcal{M})$.

- Corollary 3.5.5 gives the required upper bound on $\|L(\mu_i)\|$ and $\|L(\Sigma_i) - I\|$

- The first two conditions of Lemma 3.5.4, combined with Claim 3.5.3, imply the condition that no pair of components has essentially the same mean and covariance
- Finally, the mixing weights are unchanged by the linear transformation so the third condition is easily verified (since the original mixture is δ-well-conditioned)

Thus, by Theorem 3.4.1 we can obtain a list of $(1/\epsilon)^{O_k(1)}$ candidate mixtures at least one of which satisfies

$$\|w_i - \widetilde{w}_i\| + \|L(\mu_i) - \widetilde{\mu}_i\|_2 + \left\|L(\Sigma_i) - \widetilde{\Sigma}_i\right\|_2 \leq \mathrm{poly}(\epsilon)$$

for all i. By Claim 3.5.3, we know that the components we compute are close in TV to the true components. Now applying the inverse transformation L^{-1} to all of the components, we are done.

■

3.6 Rough Clustering

As mentioned earlier in the proof overview, the first step in our full algorithm will be to cluster the points. We present our clustering algorithm in this section. This section closely mirrors the work in [39]. We first define a measure of closeness between Gaussians that we will use throughout the paper.

Definition 3.6.1. *We say that two Gaussians $N(\mu, \Sigma)$ and $N(\mu', \Sigma')$ are C-close if all of the following conditions hold*

1. *(mean condition) For all unit vectors $v \in \mathbb{R}^d$, we have $(v \cdot \mu - v \cdot \mu')^2 \leq C v^T(\Sigma + \Sigma')v$*
2. *(variance condition) For all unit vectors $v \in \mathbb{R}^d$, we have*

$$\max(v^T\Sigma v, v^T\Sigma' v) \leq C \min(v^T\Sigma v, v^T\Sigma' v)$$

3. *(covariance condition) Finally, we have $\left\|I - \Sigma'^{-1/2}\Sigma\Sigma'^{-1/2}\right\|_2^2 \leq C$*

The main theorem that we aim to prove in this section is the following, which implies that if the true mixture can be well-clustered into submixtures, then we can recover this clustering with constant-accuracy.

Theorem 3.6.2. *Let k, D, γ be parameters. Assume we are given ϵ-corrupted samples from a mixture of Gaussians $w_1G_1 + \cdots + w_kG_k$ where the mixing weights w_i are all rational numbers with denominator bounded by a constant A. Let $A_1, \ldots, A_l$ be a partition of the components such that*

1. *For any j_1, j_2 in the same piece of the partition G_{j_1}, G_{j_2} are D-close*
2. *For any j_1, j_2 in different pieces of the partition, G_{j_1}, G_{j_2} are not D'-close*

where $D' > F(k, A, D, \gamma)$ for some sufficiently large function F. Assume that $t > F(k, A, D, \gamma)$ and $\eta, \epsilon, \delta < f(k, A, D, \gamma)$ for some sufficiently small function f. Then with probability at least $1 - \gamma$, if $X_1, \ldots, X_n$ is an ϵ-corrupted sample from the mixture $w_1G_1 + \cdots + w_kG_k$ with $n \geq \mathrm{poly}(1/\epsilon, 1/\eta, 1/\delta, d)^{O(k,A)}$, then one of the clusterings returned by Rough Clustering *(see Algorithm 6) gives a γ-corrupted sample of each of the submixtures given by $A_1, \ldots, A_l$.*

Remark. *Note that the last statement is well defined because the assumption about the partition essentially implies that all pairs of components in different submixtures are separated so γ-corrupted sample simply means correctly recovering a $1-\gamma$-fraction of the original points that were drawn from the corresponding submixture.*

In this section, it will suffice to consider when the mixing weights are equal as we can subdivide one component into many identical ones so from now on we assume $w_1 = \cdots = w_k = 1/k$ and all dependencies on A become dependencies on k.

We begin with a few preliminaries. The following claim is a simple consequence of the definition.

Claim 3.6.3. *Let G_1, G_2, G_3 be Gaussians such that G_1 and G_2 are C-close and G_2 and G_3 are C-close. Then G_1 and G_3 are* $\mathrm{poly}(C)$*-close.*

Proof. The second condition follows immediately from the fact that G_1 and G_2 are C-close and G_2 and G_3 are C-close. Now we know that for all vectors v, $v^T\Sigma_1 v, v^T\Sigma_2 v, v^T\Sigma_3 v$ are all within a poly(C) factor of each other. This means that the singular values of $\Sigma_i^{1/2}\Sigma_j^{-1/2}$ are all bounded above and below by poly(C). From this and the triangle inequality, we get the first and third conditions. ■

The next claim follows immediately from Lemma 3.6 in [39].

Claim 3.6.4. *There is a decreasing function f such that $f(C) > 0$ for all $C > 0$ such that if two Gaussians G_1, G_2 are C-close then*

$$d_{TV}(G_1, G_2) \leq 1 - f(C)$$

We will now show that either all pairs in the mixture are not too far apart, or there exists a nontrivial partition of the mixture into two parts that are separated in either mean, variance in some direction, or covariance. This parallels Corollary 3.7 in [39]. However, we require a slightly different statement because their paper specializes to the case where all pairs of components are separated. We use μ, Σ to denote the mean and covariance of the overall mixture.

Claim 3.6.5. *Let $C > 100$ be a constant. Let C_k be a sufficiently large constant depending only on C and k. Assume that there are $i, j \in [k]$ such that $N(\mu_i, \Sigma_i)$ and $N(\mu_j, \Sigma_j)$ are not C_k-close. Then there exists a partition of $[k]$ into two disjoint sets S, T such that for any $a \in S, b \in T$, $N(\mu_a, \Sigma_a)$ is not k^C-close to $N(\mu_b, \Sigma_b)$ and at least one of the following holds:*

1. *There is a direction v such that for all $a \in S, b \in T$,*

$$((\mu_a - \mu_b) \cdot v) \geq \max\left(k^C(v^T(\Sigma_a + \Sigma_b)v), \frac{v^T\Sigma v}{k^2}\right)$$

2. *There is a direction v such that for all $a \in S, b \in T$,*

$$\frac{v^T\Sigma_a v}{v^T\Sigma_b v} \geq k^C \text{ and } \frac{v^T\Sigma_a v}{v^T\Sigma v} \geq \frac{1}{k^{2Ck}}$$

3. We have

$$\left\|I - \Sigma_a^{-1/2}\Sigma_b\Sigma_a^{-1/2}\right\|^2 \geq k^C \max\left(\left\|\Sigma_a^{1/2}A_{ab}\Sigma_a^{1/2}\right\|, \left\|\Sigma_b^{1/2}A_{ab}\Sigma_b^{1/2}\right\|, \left\|\Sigma^{1/2}A_{ab}\Sigma^{1/2}\right\|\right)$$

where $A_{ab} = \Sigma_a^{-1/2}\left(I - \Sigma_a^{-1/2}\Sigma_b\Sigma_a^{-1/2}\right)\Sigma_a^{-1/2}$

Proof. We break into a few cases:

Case 1: Suppose that there is a v such that for some a, b

$$((\mu_a - \mu_b)\cdot v)^2 \geq 10k^2 \cdot k^C \max_i(v^T\Sigma_i v)$$

then we claim we are done. To see this, first observe that

$$v^T\Sigma v = \frac{1}{k^2}\sum_{i\neq j}((\mu_i - \mu_j)\cdot v)^2 + \frac{1}{k}\sum v^T\Sigma_i v$$

so then choosing a, b such that $((\mu_a - \mu_b)\cdot v)^2$ is maximal, we have $((\mu_a - \mu_b)\cdot v)^2 \geq 0.1v^T\Sigma v$. Now we can partition the components based on the value of $\mu_i \cdot v$. We can ensure that the gap between the clusters has size at least $\frac{(\mu_a-\mu_b)\cdot v}{k}$. This will imply for all $a \in S, b \in T$

$$((\mu_a - \mu_b)\cdot v)^2 \geq k^C v^T(\Sigma_a + \Sigma_b)v)$$

i.e. the corresponding components are not k^C-close. Since we can choose C_k sufficiently large, the first condition is also satisfied and we are done in this case.

Case 2: Alternatively suppose there is a v such that

$$\frac{\max_i(v^t\Sigma_i v)}{\min_i(v^T\Sigma_i v)} \geq k^{4Ck}$$

In this case, we can partition the components based on the value of $v^T\Sigma_i v$. Without loss of generality we have

$$v^T\Sigma_1 v \geq \cdots \geq v^T\Sigma_k v$$

Note that since we are not in the first case $v^T\Sigma_1 v \geq \frac{v^T\Sigma v}{20k^{2+C}}$. Next, because $\frac{v^T\Sigma_k v}{v^T\Sigma v} \leq \frac{1}{k^{2Ck}}$ there must be some consecutive $i, i+1$ such that

$$\left(\frac{v^T\Sigma_i v}{v^T\Sigma v}\right) \geq k^C \left(\frac{v^T\Sigma_{i+1} v}{v^T\Sigma v}\right) \text{ and } \left(\frac{v^T\Sigma_i v}{v^T\Sigma v}\right) \geq \frac{1}{k^{2Ck}}$$

partitioning into $S = \{1, 2, \ldots, i\}$ and $T = \{i+1, \ldots, k\}$, we immediately verify that the desired conditions (second condition) are satisfied.

Case 3: Finally, it remains to consider the situation where neither the condition in Case 1 nor the condition in Case 2 holds. Note that by assumption, there is some pair $a, b \in [k]$ for which $N(\mu_a\Sigma_a), N(\mu_b, \Sigma_b)$ are not C_k-close. Since we can choose

$$C_k > (kC)^{10kC}$$

this pair cannot fail the variance condition in any direction (second condition of Definition 3.6.1). This pair also cannot fail the mean condition in any direction (first condition of Definition 3.6.1) because then we would have

$$((\mu_a - \mu_b) \cdot v)^2 \geq C_k v^T\Sigma_a v \geq \frac{C_k}{k^{4Ck}} \max_i (v^T\Sigma_i v)$$

and we would be in the first case. Thus, we must actually have

$$\left\| I - \Sigma_a^{-1/2}\Sigma_b\Sigma_a^{-1/2} \right\|_2^2 \geq C_k$$

Next, we claim that for all i, j, $\Sigma_i^{1/2}\Sigma_j^{-1/2}$ has smallest and largest singular value in the interval

$$\mathcal{I} \triangleq \left[\frac{1}{k^{4Ck}}, k^{4Ck}\right]$$

If this were not true, without loss of generality we can find a unit vector v such that $\left\|\Sigma_i^{1/2}\Sigma_j^{-1/2}v\right\|_2 \geq k^{4Ck}$. But this implies

$$\frac{(\Sigma_j^{-1/2}v)^T\Sigma_i(\Sigma_j^{-1/2}v)}{(\Sigma_j^{-1/2}v)^T\Sigma_j(\Sigma_j^{-1/2}v)} \geq k^{8Ck}$$

meaning we are actually in case 2. Similarly, we can show that $\Sigma_i^{1/2}\Sigma^{-1/2}$ has smallest and largest singular value in the interval $\mathcal{I}$ or else we would be in Case 1.

To complete the proof, let a_0, b_0 be indices corresponding to a pair of components that are not C_k-close and construct the following graph. Two nodes i, j are connected if and only if

$$\left\|\Sigma_{a_0}^{-1/2}\Sigma_j\Sigma_{a_0}^{-1/2} - \Sigma_{a_0}^{-1/2}\Sigma_i\Sigma_{a_0}^{-1/2}\right\|_2^2 \leq \frac{C_k}{k^2}$$

This graph must not be connected since otherwise there would be a path of length at most k between a_0 and b_0 and summing the above inequalities along this path, this would contradict the fact that

$$\left\|I - \Sigma_{a_0}^{-1/2}\Sigma_{b_0}\Sigma_{a_0}^{-1/2}\right\|_2^2 \geq C_k.$$

We claim that it suffices to take S and T to be two connected components of the graph. Indeed, for any $a \in S, b \in T$, we have

$$\left\|\Sigma_{a_0}^{-1/2}\Sigma_a\Sigma_{a_0}^{-1/2} - \Sigma_{a_0}^{-1/2}\Sigma_b\Sigma_{a_0}^{-1/2}\right\|_2^2 \geq \frac{C_k}{k^2}$$

Now observe

$$I - \Sigma_a^{-1/2}\Sigma_b\Sigma_a^{-1/2} = (\Sigma_a^{-1/2}\Sigma_{a_0}^{1/2})\left(\Sigma_{a_0}^{-1/2}\Sigma_a\Sigma_{a_0}^{-1/2} - \Sigma_{a_0}^{-1/2}\Sigma_b\Sigma_{a_0}^{-1/2}\right)(\Sigma_{a_0}^{1/2}\Sigma_a^{-1/2})$$

and combining with the singular value bounds we showed for $\Sigma_i^{1/2}\Sigma_j^{-1/2}$ and $\Sigma_i^{1/2}\Sigma^{-1/2}$, we have

$$\left\|I - \Sigma_a^{-1/2}\Sigma_b\Sigma_a^{-1/2}\right\|_2^2 \geq \max\left(k^C, k^C \left\|A_{ab}\right\|\right)$$

for any a, b on different sides of the partition. The other quantities in the third condition can be bounded similarly as long as C_k is chosen to be sufficiently large. ■

3.6.1 SOS Program

To solve the clustering problem, we set up the same polynomial constraints as in Diakonikolas et al. [39]. Recall that Definition 3.2.4 gives a recipe for turning this into an SDP relaxation.

Definition 3.6.6 (Clustering Program $\mathcal{A}$, restated from [39])**.** *Let $X_1, \ldots, X_n \in \mathbb{R}^d$ represent the samples. Let $w_1, \ldots, w_n, z_1, \ldots, z_n, X_1', \ldots, X_n'$ and $\Sigma, \Sigma^{1/2}, \Sigma^{-1/2} \in \mathbb{R}^{d\times d}$ (we think of the Σ as $d \times d$ matrices whose entries are variables) be indeterminates that we will solve for in the system. We think of the w variables as weights on the points and the z variables as representing whether points are outliers. We will enforce that the subset of points weighted by w has moments that are approximately Gaussian. The full system of polynomial constraints is given below:*

1. *We have parameters $t \in \mathbb{N}$ that is even and $\delta, \epsilon > 0$.*

2. *Let $\mathcal{A}_{\textsf{corruptions}} = \{z_i^2 = z_i\}_{i\in[n]}, \{z_i(X_i - X_i') = 0\}_{i\in[n]}, \{\sum_{i\in[n]} z_i = (1-\epsilon)n/k\}$*

3. *Let $\mathcal{A}_{\textsf{subset}} = \{w_i^2 = w_i\}_{i\in[n]}, \{\sum_{i\in[n]} w_i = n/k\}$*

4. *Let $\mu(w) = \frac{k}{n}\sum_{i\in[n]} w_i X_i'$*

5. *Let $\Sigma(w) = \frac{k}{n}\sum_{i\in[n]} w_i(X_i' - \mu(w))(X_i' - \mu(w))^T$*

6. *Let $\mathcal{A}_{\textsf{matrices}} = \{(\Sigma^{1/2})^2 = \Sigma(w)\}, \{(\Sigma^{-1/2}\Sigma^{1/2})^2 = \Sigma^{-1/2}\Sigma^{1/2}\}, \{\Sigma^{-1/2}\Sigma^{1/2}w_i(X_i' - \mu(w)) = w_i(X_i' - \mu(w))\}_{i\in[n]}$*

7. *Let $\mathcal{A}_{\text{moments}}$ be the following set of polynomial inequalities for all $s \le t$*

$$\left\| \frac{k}{n} \sum_{i \in [n]} w_i [\Sigma^{-1/2}(X_i' - \mu(w))]^{\otimes s} - M_s \right\|^2 \le \delta d^{-2t}$$

where $M_s = \mathbb{E}_{g \in N(0,I)}[g^{\otimes s}]$ is the moment tensor of a standard Gaussian.

We will work with the same set of deterministic conditions on the samples as in Diakonikolas et al. [39]. These conditions hold with high probability for the uncorrupted points.

Definition 3.6.7 (Deterministic conditions, restated from [39])**.** *Fix Gaussians $G_1, \dots, G_k$ on $\mathbb{R}^d$. For $\delta, \psi > 0$ and $t \in \mathbb{N}$. The (δ, ψ, t)-deterministic conditions with respect to $G_1, \dots, G_k$ on a set of samples $X_1, \dots, X_n \in \mathbb{R}^d$ are*

1. *There is a partition of $\{X_1, \dots, X_n\}$ into k pieces $S_1, \dots, S_k$ each of size n/k such that for all $i \in [k]$ and $s \le t$*

$$\left\| \frac{k}{n} \sum_{j \in S_i} [\overline{\Sigma}_i^{-1/2}(X_j - \overline{\mu}_i)]^{\otimes s} - M_s \right\|_F^2 \le d^{-2t}\delta$$

where $\overline{\Sigma}_i$ and $\overline{\mu}_i$ denote the empirical mean and covariance of the uniform distribution over elements of S_i and $M_s = \mathbb{E}_{g \in N(0,I)}[g^{\otimes s}]$ is the moment tensor of a standard Gaussian.

2. *For $a \in [k], v \in \mathbb{R}^d, A \in \mathbb{R}^{d \times d}$ we define*

   (a) $E_a(v) = \{X_i \in S_a | ((X_i - \mu_a) \cdot v)^2 \le O(1) \log(1/\psi) v^T \Sigma_a v\}$

   (b) $F_a(v) = \{(X_i, X_j) \in S_a^2 | ((X_i - X_j) \cdot v)^2 \ge \Omega(1) \cdot \psi v^T \Sigma_a v\}$

   (c) $G_a(A) = \{(X_i, X_j) \in S_a^2 | (X_i - X_j)^T A (X_i - X_j) = 2\langle \Sigma_a, A \rangle \pm O(1) \log(1/\psi) \cdot \|\Sigma_a A\|_F\}$.

   Then for every $v \in \mathbb{R}^d, A \in \mathbb{R}^{d \times d}$ we have

- $|E_a(v)| \geq (1-\psi)(n/k)$
- $|F_a(v)|, |G_a(A)| \geq (1-\psi)(n/k)^2$

Claim 3.6.8 (Restated from [39]). *For all even t, if*

$$n \geq \log(1/\gamma)^{Ct} d^{10kt}/\delta^2$$

for some sufficiently large constant C and $\psi \geq \delta$, then $X_1, \ldots, X_n$ drawn i.i.d from $\frac{1}{k}\sum_{i=1}^k G_i$ satisfy Definition 3.6.7 with probability at least $1-\gamma$.

We will use the following key lemmas from [39]. The setup is exactly the same. Let $X_1, \ldots, X_n \in \mathbb{R}^d$ satisfy the (δ, ψ, t)-deterministic conditions (Definition 3.6.7) with respect to Gaussians $G_1, \ldots, G_k$. Let $S_1, \ldots, S_k$ be the partition guaranteed in the definition. Let $Y_1, \ldots, Y_n$ be an ϵ-corruption of $X_1, \ldots, X_n$ and let $\mathcal{A}$ be the clustering program (Definition 3.6.6) for $Y_1, \ldots, Y_n$. For indeterminates $w_1, \ldots, w_n$, define

$$\alpha_i(w) = \sum_{j \in S_i} w_j.$$

Below we will assume ψ, τ are smaller than some universal constants $\psi_0, \tau_0 > 0$.

Recall in Claim 3.6.5 that there are essentially three different ways that two Gaussians can be separated in TV distance. We call these mean separation, variance separation, and covariance separation. The lemmas below roughly assert that if two Gaussians are separated in one of these ways, then a valid solution to the clustering program $\mathcal{A}$ cannot assign significant weight to both of them.

Lemma 3.6.9 (Mean Separation, restated from [39]). *For every $\tau > 0$, there is $s = \widetilde{O}(1/\tau^2)$ such that if $\epsilon, \delta \leq s^{-O(s)} k^{-20}$ then for all $a, b \in [k]$, all $v \in \mathbb{R}^d$ and all sufficiently small $\rho > 0$, if*

$$\langle \mu_a - \mu_b, v \rangle^2 \geq \rho \mathop{\mathbb{E}}_{X, X' \sim \frac{1}{k}\sum G_i} \langle X - X', v \rangle^2,$$

then

$$\mathcal{A} \vdash_{O(s)} \left(\frac{\alpha_a(w)\alpha_b(w)}{n^2}\right)^s \leq (s \log 1/\psi)^{O(s)} \cdot \left(\frac{\langle v, \Sigma_a v\rangle + \langle v, \Sigma_b v\rangle}{\langle \mu_a - \mu_b, v\rangle^2}\right)^{\Omega(s)} + \rho^{-O(s)}(\tau^{\Omega(s)} + \epsilon^{\Omega(s)} k^{O(s)} s^{O(s^2)} + \psi^{\Omega(s)})$$

Lemma 3.6.10 (Variance Separation, restated from [39])**.** *For every $\tau > 0$, there is $s = \widetilde{O}(1/\tau^2)$ such that if $\epsilon, \delta \leq s^{-O(s)} k^{-20}$ then for all $a, b \in [k]$, all $v \in \mathbb{R}^d$ and all sufficiently small $\rho > 0$, if*

$$\langle v, \Sigma_b v\rangle \geq \rho \mathop{\mathbb{E}}_{X, X' \sim \frac{1}{k}\sum G_i} \langle X - X', v\rangle^2,$$

then

$$\mathcal{A} \vdash_{O(s)} \left(\frac{\alpha_a(w)\alpha_b(w)}{n^2}\right)^s \leq \psi^{-O(s)} \cdot \left(s^{O(s)} \left(\frac{\langle v, \Sigma_a v\rangle}{\langle v, \Sigma_b v\rangle}\right)^{\Omega(s)} + \rho^{-O(s)}(\tau^{\Omega(s)} + \epsilon^{\Omega(s)} k^{O(s)} s^{O(s^2)})\right) + \rho^{-O(s)} \psi^{\Omega(s)}$$

Lemma 3.6.11 (Covariance Separation, restated from [39])**.** *Let Σ be the covariance of the mixture $\frac{1}{k}\sum G_i$. If $\epsilon, \delta < k^{-O(1)}$, then for all $a, b \in [k]$ and $A \in \mathbb{R}^{d \times d}$,*

$$\mathcal{A} \vdash_{O(1)} \left(\frac{\alpha_a(w)\alpha_b(w)}{n^2}\right)^{16} \leq O(\log 1/\psi)^8 \cdot \frac{\left\|\Sigma^{1/2} A \Sigma^{1/2}\right\|_F^8 + \left\|\Sigma_a^{1/2} A \Sigma_a^{1/2}\right\|_F^8 + \left\|\Sigma_b^{1/2} A \Sigma_b^{1/2}\right\|_F^8}{\langle \Sigma_a - \Sigma_b A\rangle^8} + O(\psi^4) + O(\epsilon^2 k^{20})$$

3.6.2 Clustering Algorithm

We use essentially the same clustering algorithm as [39].

Proof of Theorem 3.6.2. We can use Claim 3.6.8 to ensure that with $1 - \gamma/2$ probability, the deterministic conditions in Definition 3.6.7 are satisfied for all submixtures and the various values of δ, ψ, t that we will need.

First, if all pairs of components are D' close, then returning the entire sample as one

Algorithm 6 ROUGH CLUSTERING

Input: ϵ-corrupted samples $X_1, \dots, X_n$ and parameters $t, \delta, \epsilon, k, \eta$
Initialize a list of subsets $L = \{\}$
for $\mathsf{count} = 0, 1, \dots, 100k \log 1/\eta$ **do**
Let $\mathcal{A}$ be the clustering program (Definition 3.6.6) for $X_1, \dots, X_n$
Compute the pseudoexpectation $\widetilde{E}$ that satisfies the constraints $\mathcal{A}$ (Definition 3.6.6) and maximizes

$$\widetilde{E}\left[\sum_{i \notin \cup_{R \in L} R} w_i\right]$$

Choose a random $i \sim [n]$ with probability $p_i = \frac{\widetilde{\mathbb{E}}[w_i]}{\sum \widetilde{\mathbb{E}}[w_i]}$
Create set R by adding each element $j \in [n]$ independently with probability $\frac{\widetilde{\mathbb{E}}[w_i w_j]}{\widetilde{\mathbb{E}}[w_i]}$
Add R to the list L
Let $L = \{R_1, \dots, R_m\}$
for all subsets $S \subset L$ **do**
Recurse on $\cup_{i \in S} R_i$ for each of $k \to 1, 2, \dots, k-1$ and $t, \delta, \epsilon, \eta$ unchanged
Return $\{X_1, \dots, X_n\}$ (as one cluster) and all unions of some combination of the clusters returned in each computation branch

cluster suffices. Now, we may assume that there is some pair that is not D'-close. We apply Claim 3.6.5 and let U, V be the partition of the components given by the claim. Let C be a sufficiently large function of k, D, γ that we will set later. We can do this as long as we ensure that D' is a sufficiently large function of k, C. We ensure that $k^C > D$. Note that each of the pieces $A_1, \dots, A_l$, must be entirely in U or in V because of our assumption about closeness between the components. We claim that

$$\widetilde{\mathbb{E}}\left[\left(\sum_{i \in \cup_{j \in U} S_j} w_i\right)\left(\sum_{i \in \cup_{j \in V} S_j} w_i\right)\right] \leq \gamma' n^2 \tag{3.4}$$

where we can make γ' sufficiently small in terms of γ, D, k by choosing D' and the functions f, F suitably.

Below we will let a, b be indices such that $a \in U$ and $b \in V$. If the first clause of Claim 3.6.5 is satisfied, then we can take $\rho = poly(1/k)$ and for τ sufficiently small in terms of γ, k, D, C,

we have

$$\widetilde{\mathbb{E}}\left(\frac{\alpha_a(w)\alpha_b(w)}{n^2}\right)^s \leq k^{-C/2s}$$

Summing over all $a \in U, b \in V$, this gives (3.4).

If the second clause of Claim 3.6.5 is satisfied, then we can take

$$\rho = \min_{a \in U} \frac{v^T \Sigma_a v}{v^T \Sigma v} \geq \frac{1}{k^{2Ck}}$$

We choose τ sufficiently small in terms of γ, k, C, D and combining with the fact that $(v^t \Sigma_a v) \geq k^C (v^T \Sigma_b v)$ for all $a \in U, b \in V$ we get

$$\widetilde{\mathbb{E}}\left(\frac{\alpha_a(w)\alpha_b(w)}{n^2}\right)^s \leq k^{-\Omega(C)s}$$

Finally, when the third clause of Claim 3.6.5 is satisfied follows similarly after setting $A = A_{ab}$. In all cases, we now have (3.4). The next step will be to analyze our random sampling to select the subset R. First note

$$\widetilde{\mathbb{E}}[|R|] = \frac{\sum_{i,j} \widetilde{\mathbb{E}}[w_i w_j]}{\sum_i \widetilde{\mathbb{E}}[\sum_i w_i]} = \frac{n}{k}$$

Next we analyze the intersections with the two sides of the partition U, V. We will slightly abuse notation and use $i \in U$ when $i \in \cup_{j \in U} S_j$ and it is clear from context that we are indexing the samples. Conditioned on the first index that is randomly chosen satisfying $i \in U$ then

$$\mathbb{E}[|R \cap V|] = \frac{\sum_{i_1 \in U, i_2 \in V} \widetilde{\mathbb{E}}[w_{i_1} w_{i_2}]}{\sum_{i \in U} \widetilde{\mathbb{E}}[w_i]} \leq \frac{\gamma' n^2}{\sum_{i \in U} \widetilde{\mathbb{E}}[w_i]}$$

repeating the same argument for when $i \in V$, we have $\mathbb{E}[\min(|R \cap U|, |R \cap V|)] \leq \gamma' kn$. Finally, we lower bound the expected number of new elements that R adds to the list L. This quantity is

$$\frac{\widetilde{\mathbb{E}}[\sum_{i \in [n], j \notin L} w_i w_j]}{n/k} = \sum_{j \notin L} \widetilde{\mathbb{E}}[w_j]$$

where by $j \notin L$ we mean j is not in the union of all previous subsets in the list L. Note that indicator functions of the components $S_1, \ldots, S_k$ are all valid pseudoexpectations and since we are picking the pseudoexpectation that maximizes $\sum_{j \notin L} \widetilde{\mathbb{E}}[w_j]$, the expected number of new elements added to L is at least

$$\frac{n - |\cup_{R \in L} R|}{k}$$

Now we analyze the recombination step once we finalize $L = \{R_1, \ldots, R_m\}$. For any sufficiently small function $h(k, \gamma, D)$, we claim that by choosing D' and the functions f, F appropriately, we can ensure with $1 - h(k, D, \gamma)$ probability, there is some recombination that gives a $1 - h(k, D, \gamma)$-corrupted sample of the submixture corresponding to U. To see this, it suffices to set $\eta < h(k, D, \gamma)$ and then look at the first $m' = 100k \log 1/h(k, D, \gamma)$ subsets in L. Their union has expected size $(1 - h(k, D, \gamma)^{100})n$. Next, among $R_1, \ldots, R_{m'}$,

$$\mathbb{E}\left[\sum_{i=1}^{m'} \min(|R_i \cap U|, |R_i \cap V|)\right] \leq \gamma' k m' n$$

If we ensure that γ' is sufficiently small in terms of γ, D, k, then using Markov's inequality, with $1 - h(k, D, y)$ probability, there is some recombination that gives a $1 - h(k, D, \gamma)$-corrupted sample of the submixture corresponding to U. We can make the same argument for V. Now we can recurse and repeat the argument because each of these submixtures only contains at most $k - 1$ true components. ■

3.6.3 Improved Clustering Result from [11]

In [11], the authors obtain a rough clustering result similar to Theorem 3.6.2 but are able to remove the bounded fractionality assumption. Their result is restated using our notation below.

Theorem 3.6.12 ([11])**.** *Let k, D, γ be parameters. Assume we are given ϵ-corrupted samples from a mixture of Gaussians $w_1 G_1 + \cdots + w_k G_k$ where the mixing weights w_i are all at least*

$1/A$ *for some constant* A*. Let* $A_1, \dots, A_l$ *be a partition of the components such that*

1. *For any* j_1, j_2 *in the same piece of the partition* G_{j_1}, G_{j_2} *are* D*-close*

2. *For any* j_1, j_2 *in different pieces of the partition,* G_{j_1}, G_{j_2} *are not* D'*-close*

where $D' > F(k, A, D, \gamma)$ *for some sufficiently large function* F*. Assume that* $t > F(k, A, D, \gamma)$ *and* $\eta, \epsilon, \delta < f(k, A, D, \gamma)$ *for some sufficiently small function* f*. Then with probability at least* $1 - \gamma$*, if* $X_1, \dots, X_n$ *is an* ϵ*-corrupted sample from the mixture* $w_1 G_1 + \cdots + w_k G_k$ *with* $n \geq \text{poly}(1/\epsilon, 1/\eta, 1/\delta, d)^{O(k,A)}$*, then there is an algorithm that runs in* $\text{poly}(n)$ *time and returns* $O_k(1)$ *candidate clusterings, at least one of which gives a* γ*-corrupted sample of each of the submixtures given by* $A_1, \dots, A_l$*.*

Observe that Theorem 3.6.12 is the same as Theorem 3.6.2 but with the bounded fractionality assumption removed. Theorem 3.6.2 is the only source of the bounded fractionality assumption in our paper. In the subsequent sections, replacing all uses of Theorem 3.6.2 with Theorem 3.6.12 allows us to remove the bounded fractionality assumption from our main result.

3.7 Putting Everything Together

We can now combine our clustering results and our results for learning mixtures of Gaussians that are not too separated to get a learning algorithm in the fully general case. Our main theorem is stated below.

Theorem 3.7.1. *Let* $k, A, b > 0$ *be constants. There is a sufficiently large function* G *and a sufficiently small function* g *depending only on* k, A, b *(with* $G(k, A, b), g(k, A, b) > 0$*) such that given an* ϵ*-corrupted sample* $X_1, \dots, X_n$ *from a mixture of Gaussians* $\mathcal{M} = w_1 G_1 + \cdots + w_k G_k \in \mathbb{R}^d$ *where the* G_i *have variance at least* $\text{poly}(\epsilon/d)$ *and at most* $\text{poly}(d/\epsilon)$ *in all directions and*

- *The* w_i *are all rational with denominator at most* A

- $d_{\mathsf{TV}}(G_i, G_j) \geq b$

and $n \geq (d/\epsilon)^{G(k,A,b)}$*, then there is an algorithm that runs in time* $\mathrm{poly}(n)$ *and with* 0.99 *probability outputs a mixture*

$$\widetilde{\mathcal{M}} = \widetilde{w_1}\widetilde{G_1} + \cdots + \widetilde{w_k}\widetilde{G_k}$$

such that $d_{\mathsf{TV}}(\widetilde{\mathcal{M}}, \mathcal{M}) \leq \epsilon^{g(k,A,b)}$.

3.7.1 Distance Between Gaussians

We will need to prove a few preliminary results. The main lemma we prove in this section is the following, which gives a stronger bound than the triangle inequality for TV distance between Gaussians.

Lemma 3.7.2. *Let* λ *be a constant. Let* A, B, C *be Gaussian distributions. Assume that* $d_{\mathsf{TV}}(A, B) \leq 1 - \lambda$*. If* $d_{\mathsf{TV}}(A, C) \geq 1 - \epsilon$ *and* ϵ *is sufficiently small then*

$$d_{\mathsf{TV}}(B, C) \geq 1 - \mathrm{poly}(\epsilon)$$

(where the RHS may depend on λ*).*

Note that this result is not true for arbitrary distributions A, B, C. We actually need to exploit the fact that A, B, C are Gaussian.

Our proof will parallel results in Section 8 of [39]. First, a definition:

Definition 3.7.3. *For two distributions* P, Q *let*

$$h(P, Q) = -\log(1 - d_{\mathsf{TV}}(P, Q)).$$

The key ingredient is the following result from [39]:

Lemma 3.7.4 (Restated from [39])**.** *Let* A *and* B *be two Gaussians with* $h(A, B) = O(1)$*. If* $D \in \{A, B\}$ *then*

$$P_{x \sim D}\left[\epsilon \leq \frac{A(x)}{B(x)} \leq \frac{1}{\epsilon}\right] \geq 1 - \mathrm{poly}(\epsilon)$$

Proof of Lemma 3.7.2. Note that

$$P_{x\sim A}\left[\epsilon^{0.5} \le \frac{A(x)}{C(x)} \le \frac{1}{\epsilon^{0.5}}\right] \le \epsilon^{0.5}$$

If this weren't the case, then A and C would have more than ϵ overlap, contradicting our assumption. Next, by Lemma 3.7.4,

$$P_{x\sim A}\left[\epsilon^{0.1} \le \frac{A(x)}{B(x)} \le \frac{1}{\epsilon^{0.1}}\right] \ge 1 - \text{poly}(\epsilon) \tag{3.5}$$

Combining the above two inequalities, we deduce

$$P_{x\sim A}\left[\epsilon^{0.4} \le \frac{C(x)}{B(x)} \le \frac{1}{\epsilon^{0.4}}\right] \le \text{poly}(\epsilon) \tag{3.6}$$

Let $0 < c < 0.1$ be a constant such that the RHS of (3.6) is at most ϵ^c. By Lemma 3.7.4

$$P_{x\sim B}\left[\epsilon^{c/2} \le \frac{A(x)}{B(x)} \le \frac{1}{\epsilon^{c/2}}\right] \ge 1 - \text{poly}(\epsilon)$$

and combining with (3.6), we deduce

$$P_{x\sim B}\left[\epsilon^{0.4} \le \frac{C(x)}{B(x)} \le \frac{1}{\epsilon^{0.4}}\right] \le \text{poly}(\epsilon)$$

which implies $d_{\mathsf{TV}}(B, C) \ge 1 - \text{poly}(\epsilon)$. ■

3.7.2 Full Algorithm

We are now ready to prove Theorem 3.7.1. We begin by describing the algorithm. Our full algorithm consists of several phases.

1. Cluster with constant accuracy into constant-separated submixtures
2. Learn parameters of submixtures to constant accuracy
3. Recluster all points and form new poly(ϵ)-separated submixtures

4. Learn parameters of submixtures to poly(ϵ) accuracy

The algorithm Learn Parameters (well-conditioned) for learning the parameters of a well-conditioned mixture of Gaussians (see Theorem 3.5.2) is summarized in Algorithm 7.

Algorithm 7 Learn Parameters (well-conditioned)

Input: ϵ-corrupted sample $X_1, \dots, X_n$ from δ-well-conditioned mixture of Gaussians $\mathcal{M} = w_1 G_1 + \cdots + w_k G_k$
Estimate Hermite polynomials of $\mathcal{M}$
Solve for parameters using SOS (see Section 3.4)

Our full algorithm is described in the next block Algorithm 8.

Algorithm 8 Full Algorithm

Input: ϵ-corrupted sample $X_1, \dots, X_n$ from mixture of Gaussians $\mathcal{M} = w_1 G_1 + \cdots + w_k G_k$
Run Rough Clustering Algorithm to split sample into subsamples for submixtures where all pairs are D-close for constant D
for each candidate clustering **do**
    Run Learn Parameters (well-conditioned) for each submixture
    Output candidate components
for each set of candidate components $\widetilde{G_1}, \dots \widetilde{G_k}$ **do**
    Assign samples to components according to maximum likelihood to form sets of samples $\{\widetilde{S_1}, \dots, \widetilde{S_k}\}$
    for all partitions of $[k]$ into sets $R_1, \dots, R_l$ **do**
        Run Learn Parameters (well-conditioned) on each of $\cup_{i \in R_j} S_i$ for all $j \in [l]$
        Output candidate components
Hypothesis test over all candidate components to find a mixture $\widetilde{\mathcal{M}}$ that is poly(ϵ)-close to $\mathcal{M}$

3.7.3 Analysis of Full Algorithm

The first step will be to show that among the first set of candidate components that we output, there are some that are within constant distance (say $< c(k)$ for some sufficiently small function c) of the true components.

Lemma 3.7.5. *Let $k, A, b > 0$ be constants and θ be a desired accuracy. There is a sufficiently large function G and a sufficiently small function g depending only on k, A, b, θ such that given an ϵ-corrupted sample $X_1, \ldots, X_n$ from a mixture of Gaussians $\mathcal{M} = w_1 G_1 + \cdots + w_k G_k \in \mathbb{R}^d$ where*

- *The w_i are all rational with denominator at most A*
- *$d_{\mathsf{TV}}(G_i, G_j) \geq b$*

and

- *$\epsilon < g(k, A, b, \theta)$*
- *$n \geq (d/\epsilon)^{G(k,A,b,\theta)}$*

then there is an algorithm that runs in time $\mathrm{poly}(n)$ *and with* 0.999 *probability outputs a set of $(1/\theta)^{G(k,A,b,\theta)}$ candidate mixtures at least one of which satisfies*

$$\max\left(d_{\mathsf{TV}}(\widetilde{G_1}, G_1), \ldots, d_{\mathsf{TV}}(\widetilde{G_k}, G_k)\right) \leq \theta$$
$$\max\left(|\widetilde{w_1} - w_1|, \ldots, |\widetilde{w_k} - w_k|\right) \leq \theta$$

Proof. We will use Theorem 3.6.2 to argue that the clustering algorithm finds some set of candidate clusters that can then be used to learn the parameters via Theorem 3.5.2. The main thing we need to prove is that we can find the D, D' satisfying the hypotheses of Theorem 3.6.2. In the argument below, all functions may depend on k, A, b, θ but we may omit writing some of these variables in order to highlight the important dependences.

Note that Claim 3.6.4 combined with Theorem 3.5.2 imply that if we have a γ-corrupted sample of a submixture of $\mathcal{M}$ where all pairs are D-close and $\gamma < f(D, \theta)$ for some sufficiently small function f then we can learn the components of the submixture to the desired accuracy. Now if the separation condition of Theorem 3.6.2 were satisfied with $\gamma = f(D, \theta)$ and $D' > F(k, D, \gamma)$ then we would be done.

We now show that there is some constant D depending only on k, A, b, θ for which this is true. Assume that the condition does not hold for some value of D_0. Then construct a graph G_{D_0} on nodes $1, 2, \ldots, k$ where two nodes are connected if and only if they are D-close. Take the connected components in this graph. Note that by Claim 3.6.3, all pairs in the same connected component are $\text{poly}(D_0)$-close. Thus, there must be an edge between two components such that G_i and G_j are D_1-close for

$$D_0 < D_1 < F(k, \text{poly}(D_0), f(\text{poly}(D_0), \theta))$$

Now the graph G_{D_1} has one less connected component than G_{D_0}. Starting from say $D_0 = 2$, we can iterate this argument and deduce that the entire graph will be connected for some constant value of D depending only on k, A, b, θ. Now by Claim 3.6.3 it suffices to treat the entire mixture as one mixture and we can apply Claim 3.6.4 and Theorem 3.5.2 to complete the proof. ■

Our next step is to show that if our algorithm starts with component estimates that are accurate within some constant and guesses a good set of clusters, then the resulting subsamples (after assigning according to maximum likelihood) are equivalent to $\text{poly}(\epsilon)$-corrupted samples from the corresponding submixtures. First, we prove a preliminary claim which implies that a good set of clusters exists.

Claim 3.7.6. *Let $\mathcal{M} = w_1 G_1 + \cdots + w_k G_k$ be a mixture of Gaussians. For any constant $c > 0$ and parameter ϵ, there exists a function $f(c, k)$ such that there exists a partition (possibly trivial) of $[k]$ into sets $R_1, \ldots, R_l$ such that*

- *If we draw edges between all i, j such that $d_{\mathsf{TV}}(G_i, G_j) \leq 1 - \epsilon^{c\kappa}$ then each piece of the partition is connected*
- *For any i, j in different pieces of the partition $d_{\mathsf{TV}}(G_i, G_j) \geq 1 - \epsilon^{\kappa}$*

and $f(c, k) < \kappa < 1$.

Proof. For a real number f, let $\mathcal{G}_f$ be the graph on $[k]$ obtained by connecting two nodes i, j if and only if $d_{\mathsf{TV}}(G_i, G_j) \leq 1 - f$. Consider $\mathcal{G}_{\epsilon^{c^k}}$. Consider the partition formed by taking all connected components in this graph. If this partition does not satisfy the desired condition, then there are some two G_i, G_j in different components such that

$$d_{\mathsf{TV}}(G_i, G_j) \leq 1 - \epsilon^{c^{k-1}}$$

Thus, the graph $\mathcal{G}_{\epsilon^{c^{k-1}}}$ has strictly fewer connected components than $\mathcal{G}_{\epsilon^{c^k}}$. We can now repeat this argument on $\mathcal{G}_{\epsilon^{c^{k-1}}}$. However, the number of connected components in $\mathcal{G}_{\epsilon^{c^k}}$ is at most k so we conclude that there must be some $c^k < \kappa < 1$ for which the desired condition is satisfied. ■

We will also need the following results about the VC-dimension of hypotheses obtained by comparing the density functions of two mixtures of Gaussians. The reason we need these VC dimension bounds is that we will need to argue that given *any* constant-accuracy estimates, we can obtain a clustering that is poly(ϵ) accurate. While naively this would require union bounding over infinitely many possibilities for the initial estimates, the VC dimension bound allows to get around this and obtain uniform convergence over all possible initial estimates.

Technically for our clustering result, we only need the VC dimension bound for single Gaussians (instead of mixtures of k Gaussians). However, we will need the VC dimension bound for mixtures of Gaussians later when we do hypothesis testing so we state the full result below. First we need a definition.

Definition 3.7.7. *Let $\mathcal{F}$ be a family of distributions on some domain $\mathcal{X}$. Let $\mathcal{H}_{\mathcal{F},a}$ be the set of functions of the form $f_{\mathcal{M}_1,\mathcal{M}_2,\ldots,\mathcal{M}_a}$ where $\mathcal{M}_1, \mathcal{M}_2, \ldots, \mathcal{M}_a \in \mathcal{F}$ and*

$$f_{\mathcal{M}_1,\mathcal{M}_2,\ldots,\mathcal{M}_a}(x) = \begin{cases} 1 \text{ if } \mathcal{M}_1(x) \geq \mathcal{M}_2(x), \ldots, \mathcal{M}_a(x) \\ 0 \text{ otherwise} \end{cases}$$

where $\mathcal{M}_i(x)$ denotes the pdf of the corresponding distribution at x.

Lemma 3.7.8 (Theorem 8.14 in [5]). *Let $\mathcal{F}_k$ be the family of distributions that are a mixture of at most k Gaussians in $\mathbb{R}^d$. Then the VC dimension of $\mathcal{H}_{\mathcal{F}_k,a}$ is* $\mathrm{poly}(d, a, k)$.

It is a standard result in learning theory that for a hypothesis class with bounded VC dimension, taking a polynomial number of samples suffices to get a good approximation for all hypotheses in the class.

Lemma 3.7.9 ([92]). *Let $\mathcal{H}$ be a hypothesis class of functions from some domain $\mathcal{X}$ to $\{0,1\}$ with VC dimension V. Let $\mathcal{D}$ be a distribution on $\mathcal{X}$. Let $\epsilon, \delta > 0$ be parameters. Let S be a set of $n = \mathrm{poly}(V, 1/\epsilon, \log 1/\delta)$ i.i.d samples from $\mathcal{D}$. Then with $1-\delta$ probability, for all $f \in \mathcal{H}$*

$$\left| \mathop{\mathbb{E}}_{x\sim S}[f(x)] - \mathop{\mathbb{E}}_{x\sim \mathcal{D}}[f(x)] \right| \leq \epsilon\,.$$

Now we can prove our lemma about obtaining a $\mathrm{poly}(\epsilon)$-accurate clustering into submixtures when given constant-accuracy estimates for the components.

Lemma 3.7.10. *Let $\mathcal{M} = w_1 G_1 + \cdots + w_k G_k \in \mathbb{R}^d$ be a mixture of Gaussians where*

- *The w_i are all rational with denominator at most A*
- *$d_{\mathsf{TV}}(G_i, G_j) \geq b$*

There exists a sufficiently small function $g(k, A, b) > 0$ depending only on k, A, b such that the following holds. Let $X_1, \ldots, X_n$ be an ϵ-corrupted sample from the mixture $\mathcal{M}$ where $\epsilon < g(k, A, b)$ and $n = \mathrm{poly}(d/\epsilon)$ for some sufficiently large polynomial. Let $S_1, \ldots, S_k \subset \{X_1, \ldots, X_n\}$ denote the sets of samples from each of the components $G_1, \ldots, G_k$ respectively. Let $R_1, \ldots, R_l$ be a partition such that for $i_1 \in R_{j_1}, i_2 \in R_{j_2}$ with $j_1 \neq j_2$,

$$d_{\mathsf{TV}}(G_{i_1}, G_{i_2}) \geq 1 - \epsilon'$$

where $\epsilon \leq \epsilon' \leq g(k, A, b)$. Let $\widetilde{G_1}, \ldots, \widetilde{G_k}$ be any Gaussians such that $d_{\mathsf{TV}}(G_i, \widetilde{G_i}) \leq g(k, A, b)$ for all i. Let $\widetilde{S_1}, \ldots, \widetilde{S_k} \subset \{X_1, \ldots, X_n\}$ be the subsets of samples obtained by assigning each

sample to the component $\widetilde{G_i}$ *that gives it the maximum likelihood. Then with probability at least* 0.999*,*

$$\left|\left(\cup_{i\in R_j} S_i\right) \cap \left(\cup_{i\in R_j} \widetilde{S_i}\right)\right| \geq (1 - \mathrm{poly}(\epsilon')) \left|\left(\cup_{i\in R_j} S_i\right)\right|$$

for all j*.*

Proof. First, we will upper bound the expected number of uncorrupted points that are misclassified for each $j \in [l]$ when the Gaussians $\widetilde{G_1}, \ldots, \widetilde{G_k}$ are fixed. This quantity can be upper bounded by

$$\sum_{j_1 \neq j_2} \sum_{\substack{i_1 \in R_{j_1} \\ i_2 \in R_{j_2}}} \int 1_{\widetilde{G_{i_1}}(x) > \widetilde{G_{i_2}}(x)} dG_{i_2}(x)$$

Clearly we can ensure $d_{\mathsf{TV}}(G_i, \widetilde{G_i}) \leq 1/2$. Thus, by Lemma 3.7.2 and the assumption about $R_1, \ldots, R_l$, $d_{\mathsf{TV}}(\widetilde{G_{i_1}}, G_{i_2}) \geq 1 - \mathrm{poly}(\epsilon')$ for all G_{i_2} where i_2 is not in the same piece of the partition as i_1. Let c be such that

$$d_{\mathsf{TV}}(\widetilde{G_{i_1}}, G_{i_2}) \geq 1 - \epsilon'^c$$

By Lemma 3.7.4,

$$\Pr_{x \in G_{i_2}}\left[\epsilon'^{c/2} \leq \frac{\widetilde{G_{i_2}}(x)}{G_{i_2}(x)} \leq \epsilon'^{c/2}\right] \geq 1 - \mathrm{poly}(\epsilon')$$

and combining the above two inequalities, we deduce

$$\int 1_{\widetilde{G_{i_1}}(x) > \widetilde{G_{i_2}}(x)} dG_{i_2}(x) \leq \mathrm{poly}(\epsilon')$$

Since we are only summing over $O_k(1)$ pairs of components, as long as ϵ' is sufficiently small compared to k, A, b, the expected fraction of misclassified points is $\mathrm{poly}(\epsilon')$.

Next, note that the clustering depends only on the comparisons between the values of the pdfs of the Gaussians $\widetilde{G_1}, \ldots, \widetilde{G_k}$ at each of the samples $X_1, \ldots, X_n$. Since $n = \mathrm{poly}(d/\epsilon)$ for some sufficiently large polynomial, applying Lemma 3.7.8 and Lemma 3.7.9 completes the proof (note that the fraction of corrupted points is at most ϵ so it does not matter how they

are clustered). ■

Combining Lemma 3.7.5, Claim 3.7.6, Lemma 3.7.10 and Theorem 3.5.2, we can show that at least one of the sets of candidate parameters that our algorithm outputs is close to the true parameters.

Lemma 3.7.11. *Let $k, A, b > 0$ be constants. There is a sufficiently large function G and a sufficiently small function g depending only on k, A, b such that given an ϵ-corrupted sample $X_1, \ldots, X_n$ from a mixture of Gaussians $\mathcal{M} = w_1 G_1 + \cdots + w_k G_k \in \mathbb{R}^d$ where*

- *The w_i are all rational with denominator at most A*
- *$d_{\mathsf{TV}}(G_i, G_j) \geq b$*

and $n \geq (d/\epsilon)^{G(k,A,b)}$, with 0.999 probability, among the set of candidates output by Full Algorithm*, there is some $\{\widetilde{w_1}, \widetilde{G_1}, \ldots, \widetilde{w_k}, \widetilde{G_k}\}$ such that for all i we have*

$$|w_i - \widetilde{w_i}| + d_{\mathsf{TV}}(G_i, \widetilde{G_i}) \leq \mathrm{poly}(\epsilon)$$

Proof. This follows from combining Lemma 3.7.5, Claim 3.7.6, Lemma 3.7.10 and finally applying Theorem 3.5.2. Note we can choose c in Claim 3.7.6 so that when combined with Lemma 3.7.10, the resulting accuracy that we get on each submixture is high enough that we can then apply Theorem 3.5.2 (we can treat the subsample corresponding to each submixture as a $\mathrm{poly}(\epsilon')$-corrupted sample). We apply Lemma 3.7.10 with $\epsilon' = \epsilon^\kappa$ where the κ is obtained from Claim 3.7.6. ■

We have shown that our algorithm recovers a list of candidate mixtures, at least one of which is close to the true mixture. The last result that we need is a hypothesis testing routine. This is similar to the hypothesis testing result in [65]. However, there is a subtle difference that the samples we use to hypothesis test may not be independent of the hypotheses. This is because the adversary sees all of the data points and may corrupt the data in a way to affect the list of hypotheses that we output. Thus, we must prove that given an ϵ-corrupted sample

and *any* list of hypotheses with the promise that at least one of the hypotheses is close to the true distribution, we must output a hypothesis that is close to the true distribution.

Lemma 3.7.12. *Let $\mathcal{F}$ be a family of distributions on some domain $\mathcal{X}$ with explicitly computable density functions that can be efficiently sampled from. Let V be the VC dimension of $\mathcal{H}_{\mathcal{F},2}$ (recall Definition 3.7.7). Let $\mathcal{D}$ be an unknown distribution in $\mathcal{F}$. Let m be a parameter. Let $X_1, \ldots, X_n$ be an ϵ-corrupted sample from $\mathcal{D}$ with $n \geq \text{poly}(m, \epsilon, V)$ for some sufficiently large polynomial. Let $H_1, \ldots, H_m$ be distributions in $\mathcal{F}$ given to us by an adversary with the promise that*

$$\min(d_{\mathsf{TV}}(\mathcal{D}, H_i)) \leq \epsilon \,.$$

Then there exists an algorithm that runs in time $\text{poly}(n, \epsilon)$ *and outputs an i with $1 \leq i \leq m$ such that with* 0.999 *probability*

$$d_{TV}(\mathcal{D}, H_i) \leq O(\epsilon) \,.$$

Proof. The proof will be very similar to the proof in [65] except we will use the VC dimension bound and Lemma 3.7.9 to obtain a bound over all possible hypothesis distributions given to us by the adversary.

For each i, j, define $A_{i,j}$ to be the subset of $\mathcal{X}$ where $H_i(x) \geq H_j(x)$ (where we abuse notation and use H_i, H_j to denote their respective probability density functions). Note $d_{\mathsf{TV}}(H_i, H_j) = |H_i(A_{i,j}) - H_j(A_{i,j})|$. By Lemma 3.7.9, we can ensure that with high probability, for all i, j, the empirical estimates of $A_{i,j}$ are close to their true values, i.e.

$$|\mathcal{D}(A_{i,j}) - X(A_{i,j})| \leq 2\epsilon \,.$$

Now, since the distributions $H_1, \ldots, H_m$ can be efficiently sampled from, we can obtain estimates $\widehat{H_l}(A_{i,j})$ that are within ϵ of $H_l(A_{i,j})$ for all i, j, l. Now, it suffices to return any l such that for all i, j,

$$|X(A_{i,j}) - \widehat{H_l}(A_{i,j})| \leq 4\epsilon \,.$$

Note that any l such that $d_{\mathsf{TV}}(\mathcal{D}, H_l) \leq \epsilon$ must satisfy the above by the triangle inequality. Next, we argue that any such l must be sufficient. To see this, let l' be such that $d_{\mathsf{TV}}(\mathcal{D}, H_{l'}) \leq$

ϵ. Then

$$d_{\mathsf{TV}}(\mathcal{D}, H_l) \leq \epsilon + d_{\mathsf{TV}}(H_l, H_{l'}) = \epsilon + |H_l(A_{l,l'}) - H_{l'}(A_{l,l'})| \leq 2\epsilon + |H_l(A_{l,l'}) - \mathcal{D}(A_{l,l'})|$$
$$\leq 2\epsilon + |X(A_{l,l'}) - \mathcal{D}(A_{l,l'})| + |X(A_{l,l'}) - \widehat{H_l}(A_{l,l'})| + |\widehat{H_l}(A_{l,l'}) - H_l(A_{l,l'})| = O(\epsilon)\,.$$

■

We can now complete the proof of our main theorem.

Proof of Theorem 3.7.1. Combining Lemma 3.7.11, Lemma 3.7.12 and Lemma 3.7.8, we immediately get the desired bound. ■

3.8 Identifiability

Theorem 3.7.1 implies that we can learn a mixture that is close to the true mixture in TV distance. In order to prove that we recover the individual components, it suffices to prove identifiability. In this section we prove the following.

Theorem 3.8.1. *Let $\mathcal{M} = w_1 G_1 + \cdots + w_{k_1} G_{k_1}$ and $\mathcal{M}' = w'_1 G'_1 + \cdots + w'_{k_2} G'_{k_2}$ be mixtures of Gaussians such that $TV(\mathcal{M}, \mathcal{M}') \leq \epsilon$ and the G_i, G'_i have variance at least $\mathrm{poly}(\epsilon/d)$ and at most $\mathrm{poly}(d/\epsilon)$ in all directions. Further assume,*

- *$d_{\mathsf{TV}}(G_i, G_j) \geq b, d_{\mathsf{TV}}(G'_i, G'_j) \geq b$ for all $i \neq j$*
- *$w_i, w'_i \geq w_{\min}$*

where $w_{\min} \geq f(k)$, $b \geq \epsilon^{f(k)}$ where $k = \max(k_1, k_2)$ and $f(k) > 0$ is sufficiently small function depending only on k. Then $k_1 = k_2$ and there exists a permutation π such that

$$|w_i - w'_{\pi(i)}| + d_{\mathsf{TV}}(G_i, G'_{\pi(i)}) \leq \mathrm{poly}(\epsilon)$$

While technically, we do not need to prove identifiability in an algorithmic manner, our proof will mirror our main algorithm. We will first prove identifiability in the case where the two mixtures are δ-well conditioned for $\delta = \mathrm{poly}(\epsilon)$.

Lemma 3.8.2. *Let $\mathcal{M} = w_1 G_1 + \cdots + w_{k_1} G_{k_1}$ and $\mathcal{M}' = w'_1 G'_1 + \cdots + w'_{k_2} G'_{k_2}$ be two δ-well conditioned mixtures of Gaussians such that $d_{\mathsf{TV}}(\mathcal{M}, \mathcal{M}') \leq \epsilon$ and $\delta \geq \epsilon^{f(k)}$ where $k = \max(k_1, k_2)$ and $f(k) > 0$ is sufficiently small function depending only on k. Then $k_1 = k_2$ and there exists a permutation π such that*

$$|w_i - w'_{\pi(i)}| + d_{\mathsf{TV}}(G_i, G'_{\pi(i)}) \leq \mathrm{poly}(\epsilon)$$

Proof. Let $\mu, \Sigma, \mu', \Sigma'$ be the means and covariances of the mixtures $\mathcal{M}$ and $\mathcal{M}'$. Let $\mu_i, \Sigma_i, \mu'_i, \Sigma'_i$ be the means and covariances of the respective components. Without loss of generality we may assume $\mu = 0$, $\Sigma = I$. The results in Section 3.5, namely Corollary 3.5.7, imply that

$$\|I - \Sigma'\| = \mathrm{poly}(\epsilon)$$
$$\|\mu'\| = \mathrm{poly}(\epsilon)$$

This is because we can simulate an ϵ-corrupted sample from $\mathcal{M}'$ by just sampling from $\mathcal{M}$ (since $d_{\mathsf{TV}}(\mathcal{M}, \mathcal{M}') \leq \epsilon$) and then robustly estimate the mean and covariance of this sample. Thus, by Corollary 3.5.5, we have for all i,

$$\|\mu_i\|, \|\mu'_i\| \leq \mathrm{poly}(\delta)^{-1}$$
$$\|\Sigma_i - I\|, \|\Sigma'_i - I\| \leq \mathrm{poly}(\delta)^{-1}$$

Now, we can use Lemma 3.5.8 to estimate the Hermite polynomials of the mixtures $\mathcal{M}, \mathcal{M}'$. Since we can robustly estimate the means of bounded-covariance distributions (see Theorem 2.2 in [65], Lemma 3.5.8), we must have

$$\|v\left(h_{m,\mathcal{M}}(X) - h_{m,\mathcal{M}'}(X)\right)\|_2 \leq \mathrm{poly}(\epsilon)$$

Also note that since each of the mixtures is δ-well conditioned, using Claim 3.5.3 and Lemma

3.5.4 implies that

$$\|\mu_i - \mu_j\|_2 + \|\Sigma_i - \Sigma_j\|_2 \geq \text{poly}(\delta)$$

and similar for the components of the mixture $\mathcal{M}'$. Repeating the argument in Section 3.4.1, it suffices to prove the lemma in the case when all pairs of parameters are separated or equal i.e. among the sets $\{\mu_i\} \cup \{\mu'_i\}$ and $\{\Sigma_i\} \cup \{\Sigma'_i\}$, each pair of parameters is either equal or separated by at least $\text{poly}(\delta)$. If we prove this. we can then deduce the statement of the lemma in the general case with worse, but still polynomial dependencies on ϵ.

Now we consider the generating functions

$$F = \sum_{i=1}^{k_1} w_i e^{\mu_i(X)y + \frac{1}{2}\Sigma_i(X)y^2} = \sum_{m=0}^{\infty} \frac{1}{m!} h_{m,\mathcal{M}}(X) y^n$$
$$F' = \sum_{i=1}^{k_2} w'_i e^{\mu'_i(X)y + \frac{1}{2}\Sigma'_i(X)y^2} = \sum_{m=0}^{\infty} \frac{1}{m!} h_{m,\mathcal{M}'}(X) y^n$$

where similar to in Section 3.4, $\mu_i(X) = \mu_i \cdot X, \Sigma_i(X) = X^T \Sigma_i X$. Consider the pair $(\mu'_{k_2}, \Sigma'_{k_2})$. We claim that there must be some i such that

$$(\mu_i, \Sigma_i) = (\mu'_{k_2}, \Sigma'_{k_2})$$

Assume for the sake of contradiction that this is not the case. Let S_1 be the subset of $[k_1]$ such that $\Sigma_i = \Sigma_{k'_2}$ and let S_2 be the subset of $[k_2 - 1]$ such that $\Sigma'_j = \Sigma'_{k_2}$. Define the differential operators

$$\mathcal{D}_i = \partial - \mu_i(X) - \Sigma_i(X)y$$
$$\mathcal{D}'_i = \partial - \mu'_i(X) - \Sigma'_i(X)y$$

where as before, partial derivatives are taken with respect to y. Now consider the differential operator

$$\mathcal{D} = \left(\mathcal{D}'_{k_2-1}\right)^{2^{k_1+k_2-2}} \left(\mathcal{D}'_1\right)^{2^{k_1}} \mathcal{D}_{k_1}^{2^{k_1-1}} \mathcal{D}_1^1$$

Note by Claim 3.3.9, $\mathcal{D}(F) = 0$. Using Claim 3.3.11,

$$\mathcal{D}(F') = P(y, X)e^{\mu'_{k_2}(X)y + \frac{1}{2}\Sigma'_{k_2}(X)y^2}$$

where P is a polynomial of degree

$$\deg(P) = 2^{k_1+k_2-1} - 1 - \sum_{i \in S_1} 2^{i-1} - \sum_{i \in S_2} 2^{k_1+i-2}$$

and has leading coefficient

$$C_0 = w'_{k_2} \prod_{i \in [k_1] \setminus S_1} (\Sigma'_{k_2} - \Sigma_i)^{2^{i-1}} \prod_{i \in S_1} (\mu'_{k_2} - \mu_i)^{2^{i-1}} \prod_{i \in [k_2-1] \setminus S_2} (\Sigma'_{k_2} - \Sigma'_i)^{2^{k_1+i-2}} \prod_{i \in S_2} (\mu'_{k_2} - \mu'_i)^{2^{k_1+i-2}}.$$

If there is no i such that $(\mu_i, \Sigma_i) = (\mu'_{k_2}, \Sigma'_{k_2})$ then

$$C_0 \geq \delta^{O_k(1)}$$

We can now compare

$$\left(\mathcal{D}'_{k_2}\right)^{\deg(P)} \mathcal{D}(F)$$
$$\left(\mathcal{D}'_{k_2}\right)^{\deg(P)} \mathcal{D}(F')$$

evaluated at $y = 0$. The first quantitiy is 0 because $\mathcal{D}(F)$ is identically 0 as a formal power series. The second expression is equal to $\Omega_k(1)C_0$. However, the coefficients of the formal power series F, F' are the Hermite polynomials $h_{m,\mathcal{M}}(X)$ and $h_{m,\mathcal{M}'}(X)$. We assumed that

$$\|v\left(h_{m,\mathcal{M}}(X) - h_{m,\mathcal{M}'}(X)\right)\|_2 \leq \text{poly}(\epsilon)$$

so this is a contradiction as long as ϵ is smaller than $\delta^{F(k)}$ for some sufficiently large function F depending only on k. Thus, there must be some component of the mixture $\mathcal{M}$ that matches each component of $\mathcal{M}'$. We can then repeat the argument in reverse to conclude

that $\mathcal{M}$ and $\mathcal{M}'$ have the same components. Finally, assume that we have two mixtures $\mathcal{M} = w_1 G_1 + \cdots + w_k G_k$ and $\mathcal{M}' = w'_1 G_1 + \cdots + w'_k G_k$ on the same set of components. WLOG

$$w_1 - w'_1 < \cdots < w_l - w'_l < 0 < w_{l+1} - w'_{l+1} < \cdots < w_k - w'_k$$

Then we can consider

$$(w'_1 - w_1)G_1 + \cdots + (w'_l - w_l)G_l \text{ and}$$
$$(w_{l+1} - w'_{l+1})G_{l+1} + \cdots + (w_k - w'_k)G_k$$

each treated as a mixture. If

$$\sum_{i=1}^{k} |w_i - w'_i| > \epsilon^{\zeta}$$

for some sufficiently small ζ depending only on k, we can then normalize each of the above into a mixture (i.e. make the mixing weights sum to 1) and repeat the same argument, using the fact that pairs of components cannot be too close, to obtain a contradiction. Thus, actually the components and mixing weights of the two mixtures must be poly(ϵ)-close and this completes the proof. ■

To complete the proof of Theorem 3.8.1, we will prove a sort of cluster identifiability that mirrors our algorithm and then combine with Lemma 3.8.2.

Proof of Theorem 3.8.1. Let c be a sufficiently small constant that we will set later. We apply Claim 3.7.6 on the mixture $\mathcal{M}$ with parameter c to find a partition $R_1, \ldots, R_l$. Let κ be the parameter obtained from the statement of Claim 3.7.6 i.e. κ depends on k and c. First, we claim that each of the components $G'_1, \ldots, G'_{k_2}$ must be essentially contained within one of the clusters. To see this, for each $j \in [k_2]$ there must be some i such that

$$d_{\mathsf{TV}}(G_i, G'_j) \leq 1 - \frac{w_{\min}}{2k} \leq 1 - \Omega_k(1)$$

without loss of generality $i \in R_1$. Then by Lemma 3.7.2, for all $a \notin R_1$,

$$d_{\mathsf{TV}}(G_a, G'_j) \geq 1 - \mathrm{poly}(\epsilon^\kappa)$$

where the polynomial may depend on k *but does not depend on* c. The above implies that we can match each of the components $G'_1, \ldots, G'_{k_2}$ uniquely to one of the clusters $R_1, \ldots, R_l$ where it has constant overlap with $\cup_{i \in R_j} G_i$. Let S_1 be the subset of $[k_2]$ corresponding to the components among $G'_1, \ldots, G'_{k_2}$ that are matched to R_1. Consider the mixtures

$$\mathcal{M}_1 = \frac{\sum_{i \in R_1} w_i G_i}{\sum_{i \in R_1} w_i}$$
$$\mathcal{M}'_1 = \frac{\sum_{i \in S_1} w'_i G'_i}{\sum_{i \in S_1} w'_i}$$

The above (combined with our assumed lower bound on the minimum mixing weight) implies that

$$d_{\mathsf{TV}}(\mathcal{M}_1, \mathcal{M}'_1) \leq \mathrm{poly}(\epsilon^\kappa)$$

where again the polynomial may depend on k but not c. Now if we choose c sufficiently small, we can apply Lemma 3.8.2 to deduce that the components and mixing weights of $\mathcal{M}_1, \mathcal{M}'_1$ must be close. We can then repeat this argument for all of the clusters $R_1, \ldots, R_l$ to complete the proof. ■

Combing Theorem 3.7.1 and Theorem 3.8.1. we have

Theorem 3.8.3. *Let $k, A, b > 0$ be constants. There is a sufficiently large function G and a sufficiently small function g depending only on k, A, b (with $G(k, A, b), g(k, A, b) > 0$) such that given an ϵ-corrupted sample $X_1, \ldots, X_n$ from a mixture of Gaussians $\mathcal{M} = w_1 G_1 + \cdots + w_k G_k \in \mathbb{R}^d$ where the G_i have variance at least $\mathrm{poly}(\epsilon/d)$ and at most $\mathrm{poly}(d/\epsilon)$ in all directions and*

- *The w_i are all rational with denominator at most A*
- *$d_{\mathsf{TV}}(G_i, G_j) \geq b$*

and $n \geq (d/\epsilon)^{G(k,A,b)}$*, then there is an algorithm that runs in time* $\mathrm{poly}(n)$ *and with* 0.99 *probability outputs a set of components* $\widetilde{G_1}, \ldots, \widetilde{G_k}$ *and mixing weights* $\widetilde{w_1}, \ldots, \widetilde{w_k}$ *such that there exists a permutation* π *on* $[k]$ *with*

$$|w_i - \widetilde{w}_{\pi(i)}| + d_{\mathsf{TV}}(G_i, \widetilde{G}_{\pi(i)}) \leq \epsilon^{g(k,A,b)}$$

for all i*.*

3.8.1 Improving the Separation Assumption

With simple modifications to the analysis, we obtain the following improvement of Theorem 3.7.1 in [74].

Theorem 3.8.4 ([74])**.** *Let* $k, A > 0$ *be constants. There is a sufficiently large function* G *and a sufficiently small function* g *depending only on* k, A *such that given an* ϵ*-corrupted sample* $X_1, \ldots, X_n$ *from a mixture of Gaussians* $\mathcal{M} = w_1 G_1 + \cdots + w_k G_k \in \mathbb{R}^d$ *where* $\epsilon < g(k, A)$*, the* w_i *are all rational with denominator at most* A*, and* $n \geq (d/\epsilon)^{G(k,A)}$*, there is an algorithm that runs in time* $\mathrm{poly}(n)$ *and with* 0.999 *probability, outputs a set of* $(1/\epsilon)^{O_{k,A}(1)}$ *candidate mixtures such that for at least one of these candidates,* $\{\widetilde{w_1}, \widetilde{G_1}, \ldots, \widetilde{w_k}, \widetilde{G_k}\}$*, we have*

$$|w_i - \widetilde{w_i}| + d_{\mathsf{TV}}(G_i, \widetilde{G_i}) \leq \epsilon^{g(k,A)}$$

for all $i \in [k]$*.*

To go from Theorem 3.7.1 to Theorem 3.8.4, the main idea to remove the constant separation assumption is just that we can find a scale δ such that all pairs of components either have $d_{\mathsf{TV}}(G_i, G_j) \leq \delta$ or $d_{\mathsf{TV}}(G_i, G_j) \geq \delta'$ for $\delta' \gg \delta$. This is possible because the number of components k is a constant. We can then merge components whose TV distance is less than δ, treating them as the same component and the remaining components will be sufficiently separated. See [74] for more details.

The results of [74] were obtained using the previous clustering subroutine of [39]. If we instead plug in the updated clustering results of [11] (see Theorem 3.6.12 vs Theorem

3.6.2), we can remove the bounded fractionality assumption on the mixing weights. Also, we can ensure that the algorithm outputs a unique mixture instead of a list by running the hypothesis testing routine in Lemma 3.7.12.

Theorem 3.8.5. *Let $k, A > 0$ be constants. There is a sufficiently large function G and a sufficiently small function g depending only on k, A such that given an ϵ-corrupted sample $X_1, \ldots, X_n$ from a mixture of Gaussians $\mathcal{M} = w_1 G_1 + \cdots + w_k G_k \in \mathbb{R}^d$ where $\epsilon < g(k, A)$, the w_i are all at least $1/A$, and $n \geq (d/\epsilon)^{G(k,A)}$, there is an algorithm that runs in time* $\mathrm{poly}(n)$ *and with* 0.999 *probability, outputs a mixture $\widetilde{M} = \widetilde{w_1}\widetilde{G_1} + \cdots + \widetilde{w_k}, \widetilde{G_k}$, such that*

$$d_{\mathsf{TV}}(\mathcal{M}, \widetilde{\mathcal{M}}) \leq \epsilon^{g(k,A)} .$$

Finally, combining the above with identifiability (Theorem 3.8.1), we immediately get an improved version of our main theorem for parameter learning.

Theorem 3.8.6. *Let $k, A > 0$ be constants. There is a sufficiently large function G and a sufficiently small function g depending only on k, A (with $G(k, A), g(k, A) > 0$) such that given an ϵ-corrupted sample $X_1, \ldots, X_n$ from a mixture of Gaussians $\mathcal{M} = w_1 G_1 + \cdots + w_k G_k \in \mathbb{R}^d$ where the G_i have variance at least* $\mathrm{poly}(\epsilon/d)$ *and at most* $\mathrm{poly}(d/\epsilon)$ *in all directions and*

- *The w_i are all at least $1/A$*
- *$d_{\mathsf{TV}}(G_i, G_j) \geq \epsilon^{g(k,A)}$*

and $n \geq (d/\epsilon)^{G(k,A)}$, then there is an algorithm that runs in time $\mathrm{poly}(n)$ *and with* 0.99 *probability outputs a set of components $\widetilde{G_1}, \ldots, \widetilde{G_k}$ and mixing weights $\widetilde{w_1}, \ldots, \widetilde{w_k}$ such that there exists a permutation π on $[k]$ with*

$$|w_i - \widetilde{w}_{\pi(i)}| + d_{\mathsf{TV}}(G_i, \widetilde{G}_{\pi(i)}) \leq \epsilon^{g(k,A)}$$

for all i.

Bibliography

[1] Jayadev Acharya, Ilias Diakonikolas, Jerry Li, and Ludwig Schmidt. Sample-optimal density estimation in nearly-linear time. In *Proceedings of the Twenty-Eighth Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 1278–1289. SIAM, 2017.

[2] Jayadev Acharya, Ashkan Jafarpour, Alon Orlitsky, and Ananda Theertha Suresh. Near-optimal-sample estimators for spherical gaussian mixtures. *arXiv preprint arXiv:1402.4746*, 2014.

[3] Dimitris Achlioptas and Frank McSherry. On spectral learning of mixtures of distributions. In *Learning Theory*, pages 458–469. Springer, 2005.

[4] Dimitris Achlioptas and Frank McSherry. On spectral learning of mixtures of distributions. In *International Conference on Computational Learning Theory*, pages 458–469. Springer, 2005.

[5] Martin Anthony and Peter L Bartlett. *Neural network learning: Theoretical foundations*. cambridge university press, 2009.

[6] Sanjeev Arora and Ravi Kannan. Learning mixtures of arbitrary gaussians. In *Proceedings of the thirty-third annual ACM symposium on Theory of computing*, pages 247–257, 2001.

[7] Sanjeev Arora and Ravi Kannan. Learning mixtures of separated nonspherical gaussians. *The Annals of Applied Probability*, 15(1A):69–92, 2005.

[8] Hassan Ashtiani, Shai Ben-David, Nicholas JA Harvey, Christopher Liaw, Abbas Mehrabian, and Yaniv Plan. Nearly tight sample complexity bounds for learning mixtures of gaussians via sample compression schemes. In *Proceedings of the 32nd International Conference on Neural Information Processing Systems*, pages 3416–3425, 2018.

[9] Hassan Ashtiani, Shai Ben-David, Nicholas JA Harvey, Christopher Liaw, Abbas Mehrabian, and Yaniv Plan. Near-optimal sample complexity bounds for robust learning of gaussian mixtures via compression schemes. *Journal of the ACM (JACM)*, 67:1–42, 2020.

[10] Hassan Ashtiani, Shai Ben-David, and Abbas Mehrabian. Sample-efficient learning of mixtures. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 32, 2018.

[11] Ainesh Bakshi, Ilias Diakonikolas, He Jia, Daniel M Kane, Pravesh K Kothari, and Santosh S Vempala. Robustly learning mixtures of k arbitrary gaussians. *arXiv preprint arXiv:2012.02119*, 2020. This version may be found at `https://arxiv.org/abs/2012.02119v2`.

[12] Ainesh Bakshi, Ilias Diakonikolas, He Jia, Daniel M Kane, Pravesh K Kothari, and Santosh S Vempala. Robustly learning mixtures of k arbitrary gaussians. *arXiv preprint arXiv:2012.02119*, 2020. This version may be found at `https://arxiv.org/abs/2012.02119v3`.

[13] Ainesh Bakshi and Pravesh Kothari. Outlier-robust clustering of non-spherical mixtures. *arXiv preprint arXiv:2005.02970*, 2020.

[14] Ainesh Bakshi and Adarsh Prasad. Robust linear regression: Optimal rates in polynomial time. *arXiv preprint arXiv:2007.01394*, 2020.

[15] Sivaraman Balakrishnan, Simon S Du, Jerry Li, and Aarti Singh. Computationally efficient robust sparse estimation in high dimensions. In *Conference on Learning Theory*, pages 169–212, 2017.

[16] Boaz Barak. Proofs, beliefs, and algorithms through the lens of sum-of-squares.

[17] Boaz Barak, Jonathan A Kelner, and David Steurer. Rounding sum-of-squares relaxations. In *Proceedings of the forty-sixth annual ACM symposium on Theory of computing*, pages 31–40, 2014.

[18] Boaz Barak, Jonathan A Kelner, and David Steurer. Dictionary learning and tensor decomposition via the sum-of-squares method. In *Proceedings of the forty-seventh annual ACM symposium on Theory of computing*, pages 143–151, 2015.

[19] Boaz Barak and Ankur Moitra. Noisy tensor completion via the sum-of-squares hierarchy. In *Conference on Learning Theory*, pages 417–445, 2016.

[20] Mikhail Belkin and Kaushik Sinha. Polynomial learning of distribution families. In *Foundations of Computer Science (FOCS), 2010 51st Annual IEEE Symposium on*, pages 103–112. IEEE, 2010.

[21] Mikhail Belkin and Kaushik Sinha. Polynomial learning of distribution families. *SIAM Journal on Computing*, 44(4):889–911, 2015.

[22] Thorsten Bernholt. Robust estimators are hard to compute. Technical report, Technical Report, 2006.

[23] Aditya Bhaskara, Moses Charikar, Ankur Moitra, and Aravindan Vijayaraghavan. Smoothed analysis of tensor decompositions. In *Proceedings of the forty-sixth annual ACM symposium on Theory of computing*, pages 594–603, 2014.

[24] Aditya Bhaskara, Ananda Suresh, and Morteza Zadimoghaddam. Sparse solutions to nonnegative linear systems and applications. In *Artificial Intelligence and Statistics*, pages 83–92. PMLR, 2015.

[25] Matthew Brennan, Guy Bresler, Samuel B Hopkins, Jerry Li, and Tselil Schramm. Statistical query algorithms and low-degree tests are almost equivalent. *arXiv preprint arXiv:2009.06107*, 2020.

[26] S Charles Brubaker and Santosh S Vempala. Isotropic pca and affine-invariant clustering. In *Building Bridges*, pages 241–281. Springer, 2008.

[27] Siu-On Chan, Ilias Diakonikolas, Rocco A Servedio, and Xiaorui Sun. Efficient density estimation via piecewise polynomial approximation. In *Proceedings of the forty-sixth annual ACM symposium on Theory of computing*, pages 604–613, 2014.

[28] Siu-On Chan, Ilias Diakonikolas, Rocco A Servedio, and Xiaorui Sun. Near-optimal density estimation in near-linear time using variable-width histograms. *arXiv preprint arXiv:1411.0169*, 2014.

[29] Moses Charikar, Jacob Steinhardt, and Gregory Valiant. Learning from untrusted data. In *Proceedings of the 49th Annual ACM SIGACT Symposium on Theory of Computing*, pages 47–60. ACM, 2017.

[30] Sitan Chen, Frederic Koehler, Ankur Moitra, and Morris Yau. Online and distribution-free robustness: Regression and contextual bandits with huber contamination. *arXiv preprint arXiv:2010.04157*, 2020.

[31] Yuansi Chen. An almost constant lower bound of the isoperimetric coefficient in the kls conjecture. *Geometric and Functional Analysis*, 31(1):34–61, 2021.

[32] Sanjoy Dasgupta. Learning mixtures of gaussians. In *Foundations of Computer Science, 1999. 40th Annual Symposium on*, pages 634–644. IEEE, 1999.

[33] Sanjoy Dasgupta. Learning mixtures of gaussians. In *40th Annual Symposium on Foundations of Computer Science (Cat. No. 99CB37039)*, pages 634–644. IEEE, 1999.

[34] Sanjoy Dasgupta and Leonard Schulman. A two-round variant of em for gaussian mixtures. *arXiv preprint arXiv:1301.3850*, 2013.

[35] Sanjoy Dasgupta and Leonard J Schulman. A probabilistic analysis of em for mixtures of separated, spherical gaussians. *Journal of Machine Learning Research*, 8:203–226, 2007.

[36] Constantinos Daskalakis and Gautam Kamath. Faster and sample near-optimal algorithms for proper learning mixtures of gaussians. In *Conference on Learning Theory*, pages 1183–1213. PMLR, 2014.

[37] Constantinos Daskalakis, Christos Tzamos, and Manolis Zampetakis. Ten steps of em suffice for mixtures of two gaussians. In *Conference on Learning Theory*, pages 704–710. PMLR, 2017.

[38] Luc Devroye and Gábor Lugosi. *Combinatorial methods in density estimation*. Springer Science & Business Media, 2001.

[39] Ilias Diakonikolas, Samuel B Hopkins, Daniel Kane, and Sushrut Karmalkar. Robustly learning any clusterable mixture of gaussians. *arXiv preprint arXiv:2005.06417*, 2020.

[40] Ilias Diakonikolas, Gautam Kamath, Daniel Kane, Jerry Li, Ankur Moitra, and Alistair Stewart. Robust estimators in high-dimensions without the computational intractability. *SIAM Journal on Computing*, 48(2):742–864, 2019.

[41] Ilias Diakonikolas, Gautam Kamath, Daniel Kane, Jerry Li, Jacob Steinhardt, and Alistair Stewart. Sever: A robust meta-algorithm for stochastic optimization. In *International Conference on Machine Learning*, pages 1596–1606, 2019.

[42] Ilias Diakonikolas, Gautam Kamath, Daniel M Kane, Jerry Li, Ankur Moitra, and Alistair Stewart. Being robust (in high dimensions) can be practical. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pages 999–1008. JMLR. org, 2017.

[43] Ilias Diakonikolas and Daniel M. Kane. Recent advances in algorithmic high-dimensional robust statistics, 2019.

[44] Ilias Diakonikolas and Daniel M Kane. Small covers for near-zero sets of polynomials and learning latent variable models. In *2020 IEEE 61st Annual Symposium on Foundations of Computer Science (FOCS)*, pages 184–195. IEEE, 2020.

[45] Ilias Diakonikolas, Daniel M Kane, and Alistair Stewart. Statistical query lower bounds for robust estimation of high-dimensional gaussians and gaussian mixtures. In *2017 IEEE 58th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 73–84. IEEE, 2017.

[46] Ilias Diakonikolas, Daniel M Kane, and Alistair Stewart. List-decodable robust mean estimation and learning mixtures of spherical gaussians. In *Proceedings of the 50th Annual ACM SIGACT Symposium on Theory of Computing*, pages 1047–1060. ACM, 2018.

[47] Jon Feldman, Ryan O'Donnell, and Rocco A Servedio. Learning mixtures of product distributions over discrete domains. *SIAM Journal on Computing*, 37(5):1536–1564, 2008.

[48] Jon Feldman, Rocco A. Servedio, and Ryan O'Donnell. Pac learning axis-aligned mixtures of gaussians with no separation assumption. In *Proceedings of the 19th annual conference on Learning Theory*, COLT'06, pages 20–34, Berlin, Heidelberg, 2006. Springer-Verlag.

[49] Rong Ge, Qingqing Huang, and Sham M Kakade. Learning mixtures of gaussians in high dimensions. In *Proceedings of the forty-seventh annual ACM symposium on Theory of computing*, pages 761–770, 2015.

[50] Venkatesan Guruswami and Ali Kemal Sinop. Faster sdp hierarchy solvers for local rounding algorithms. In *2012 IEEE 53rd Annual Symposium on Foundations of Computer Science*, pages 197–206. IEEE, 2012.

[51] Frank R Hampel, Elvezio M Ronchetti, Peter J Rousseeuw, and Werner A Stahel. *Robust statistics: the approach based on influence functions*, volume 196. John Wiley & Sons, 2011.

[52] Moritz Hardt and Ankur Moitra. Algorithms and hardness for robust subspace recovery. In *Conference on Learning Theory*, pages 354–375, 2013.

[53] Moritz Hardt and Eric Price. Tight bounds for learning a mixture of two gaussians. In *Proceedings of the forty-seventh annual ACM symposium on Theory of computing*, pages 753–760, 2015.

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

[55] Samuel B Hopkins and Jerry Li. Mixture models, robustness, and sum of squares proofs. In *Proceedings of the 50th Annual ACM SIGACT Symposium on Theory of Computing*, pages 1021–1034. ACM, 2018.

[56] Samuel B Hopkins, Tselil Schramm, and Jonathan Shi. A robust spectral algorithm for overcomplete tensor decomposition. In *Conference on Learning Theory*, pages 1683–1722. PMLR, 2019.

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

[58] Samuel B Hopkins, Jonathan Shi, and David Steurer. Tensor principal component analysis via sum-of-square proofs. In *Conference on Learning Theory*, pages 956–1006, 2015.

[59] Daniel Hsu and Sham M Kakade. Learning mixtures of spherical gaussians: moment methods and spectral decompositions. In *Proceedings of the 4th conference on Innovations in Theoretical Computer Science*, pages 11–20. ACM, 2013.

[60] Peter J Huber. Robust estimation of a location parameter. *The Annals of Mathematical Statistics*, pages 73–101, 1964.

[61] Peter J Huber. *Robust statistics*, volume 523. John Wiley & Sons, 2004.

[62] David S. Johnson and Franco P Preparata. The densest hemisphere problem. *Theoretical Computer Science*, 6(1):93–107, 1978.

[63] Adam Tauman Kalai, Ankur Moitra, and Gregory Valiant. Efficiently learning mixtures of two gaussians. In *Proceedings of the 42nd ACM symposium on Theory of computing*, pages 553–562. ACM, 2010.

[64] Adam Tauman Kalai, Ankur Moitra, and Gregory Valiant. Efficiently learning mixtures of two gaussians. In *Proceedings of the forty-second ACM symposium on Theory of computing*, pages 553–562, 2010.

[65] Daniel M Kane. Robust learning of mixtures of gaussians. *arXiv preprint arXiv:2007.05912*, 2020.

[66] Adam Klivans, Pravesh K Kothari, and Raghu Meka. Efficient algorithms for outlier-robust regression. In *Conference On Learning Theory*, pages 1420–1430, 2018.

[67] Pravesh K Kothari, Jacob Steinhardt, and David Steurer. Robust moment estimation and improved clustering via sum of squares. In *Proceedings of the 50th Annual ACM SIGACT Symposium on Theory of Computing*, pages 1035–1046, 2018.

[68] Amit Kumar and Ravindran Kannan. Clustering with spectral norm and the k-means algorithm. In *Foundations of Computer Science (FOCS), 2010 51st Annual IEEE Symposium on*, pages 299–308. IEEE, 2010.

[69] Amit Kumar and Ravindran Kannan. Clustering with spectral norm and the k-means algorithm. In *2010 IEEE 51st Annual Symposium on Foundations of Computer Science*, pages 299–308. IEEE, 2010.

[70] Kevin A Lai, Anup B Rao, and Santosh Vempala. Agnostic estimation of mean and covariance. In *2016 IEEE 57th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 665–674. IEEE, 2016.

[71] Jerry Li, Allen Liu, and Ankur Moitra. Sparsification for sums of exponentials and its algorithmic applications. *arXiv preprint arXiv:2106.02774*, 2021.

[72] Jerry Li and Ludwig Schmidt. Robust and proper learning for mixtures of gaussians via systems of polynomial inequalities. In *Conference on Learning Theory*, pages 1302–1382. PMLR, 2017.

[73] Jerry Zheng Li. *Principled approaches to robust machine learning and beyond.* PhD thesis, Massachusetts Institute of Technology, 2018.

[74] Allen Liu and Ankur Moitra. Learning gmms with nearly optimal robustness guarantees. *arXiv preprint arXiv:2104.09665*, 2021.

[75] Tengyu Ma, Jonathan Shi, and David Steurer. Polynomial-time tensor decompositions with sum-of-squares. In *2016 IEEE 57th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 438–446. IEEE, 2016.

[76] Dustin G Mixon, Soledad Villar, and Rachel Ward. Clustering subgaussian mixtures by semidefinite programming. *Information and Inference: A Journal of the IMA*, 6(4):389–415, 2017.

[77] Ankur Moitra. *Algorithmic aspects of machine learning.* Cambridge University Press, 2018.

[78] Ankur Moitra and Gregory Valiant. Settling the polynomial learnability of mixtures of gaussians. In *Foundations of Computer Science (FOCS), 2010 51st Annual IEEE Symposium on*, pages 93–102. IEEE, 2010.

[79] Ankur Moitra and Gregory Valiant. Settling the polynomial learnability of mixtures of gaussians. In *2010 IEEE 51st Annual Symposium on Foundations of Computer Science*, pages 93–102. IEEE, 2010.

[80] Pablo A Parrilo. *Structured semidefinite programs and semialgebraic geometry methods in robustness and optimization.* PhD thesis, California Institute of Technology, 2000.

[81] Karl Pearson. Contributions to the mathematical theory of evolution. *Philosophical Transactions of the Royal Society of London. A*, 185:71–110, 1894.

[82] Karl Pearson. Contributions to the mathematical theory of evolution. *Philosophical Transactions of the Royal Society of London. A*, 185:71–110, 1894.

[83] Prasad Raghavendra, Satish Rao, and Tselil Schramm. Strongly refuting random csps below the spectral threshold. In *Proceedings of the 49th Annual ACM SIGACT Symposium on Theory of Computing*, pages 121–131, 2017.

[84] Oded Regev and Aravindan Vijayaraghavan. On learning mixtures of well-separated gaussians, 2017.

[85] Tselil Schramm and David Steurer. Fast and robust tensor decomposition with applications to dictionary learning. In *Conference on Learning Theory*, pages 1760–1793. PMLR, 2017.

[86] Jacob Steinhardt. *Robust Learning: Information Theory and Algorithms.* PhD thesis, Stanford University, 2018.

[87] David Steurer and Stefan Tiegel. Sos degree reduction with applications to clustering and robust moment estimation. In *Proceedings of the 2021 ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 374–393. SIAM, 2021.

[88] Yin Tat Lee and Santosh S Vempala. The kannan-lovász-simonovits conjecture. *arXiv e-prints*, pages arXiv–1807, 2018.

[89] Henry Teicher. Identifiability of mixtures. *The annals of Mathematical statistics*, 32(1):244–248, 1961.

[90] John W Tukey. A survey of sampling from contaminated distributions. *Contributions to probability and statistics*, pages 448–485, 1960.

[91] John W Tukey. Mathematics and the picturing of data. In *Proceedings of the International Congress of Mathematicians, Vancouver, 1975*, volume 2, pages 523–531, 1975.

[92] Vladimir N Vapnik and A Ya Chervonenkis. On the uniform convergence of relative frequencies of events to their probabilities. In *Measures of complexity*, pages 11–30. Springer, 2015.

[93] Santosh Vempala and Grant Wang. A spectral algorithm for learning mixture models. *Journal of Computer and System Sciences*, 68(4):841–860, 2004.

[94] Santosh Vempala and Grant Wang. A spectral algorithm for learning mixture models. *Journal of Computer and System Sciences*, 68(4):841–860, 2004.

[95] CF Jeff Wu. On the convergence properties of the em algorithm. *The Annals of statistics*, pages 95–103, 1983.

[96] Ji Xu, Daniel Hsu, and Arian Maleki. Global analysis of expectation maximization for mixtures of two gaussians. *arXiv preprint arXiv:1608.07630*, 2016.