Clustering Mixtures with Almost Optimal Separation in Polynomial Time

Jerry Li[*]   Allen Liu[†]

Abstract

We consider the problem of clustering mixtures of mean-separated Gaussians in high dimensions. We are given samples from a mixture of k identity covariance Gaussians, so that the minimum pairwise distance between any two pairs of means is at least Δ, for some parameter $\Delta > 0$, and the goal is to recover the ground truth clustering of these samples. It is folklore that separation $\Delta = \Theta(\sqrt{\log k})$ is both necessary and sufficient to recover a good clustering, at least information theoretically. However, the estimators which achieve this guarantee are inefficient. We give the first algorithm which runs in polynomial time, and which almost matches this guarantee. More precisely, we give an algorithm which takes polynomially many samples and time, and which can successfully recover a good clustering, so long as the separation is $\Delta = \Omega(\log^{1/2+c} k)$, for any $c > 0$. Previously, polynomial time algorithms were only known for this problem when the separation was polynomial in k, and all algorithms which could tolerate poly $\log k$ separation required quasipolynomial time. We also extend our result to mixtures of translations of a distribution which satisfies the Poincaré inequality, under additional mild assumptions. Our main technical tool, which we believe is of independent interest, is a novel way to implicitly represent and estimate high degree moments of a distribution, which allows us to extract important information about high-degree moments without ever writing down the full moment tensors explicitly.

1 Introduction

Gaussian mixture models are some of the most popular generative models in both theory and practice. A k-Gaussian mixture model $\mathcal{M}$ (henceforth k-GMM) is a distribution specified by k non-negative mixing weights $w_1, \ldots, w_k$ which sum to 1, and k component Gaussians $N(\mu_1, \Sigma_1), \ldots, N(\mu_k, \Sigma_k)$, and is given by the probability density function

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

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 $N(\mu_i, \Sigma_i)$. An important special case of this model is the *isotropic* (also sometimes referred to as *spherical*) case, where $\Sigma_i = I$ for all $i = 1, \ldots, k$.

One of the most important and well-studied problems for GMMs and isotropic GMMs in particular is *clustering*. 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 points into k parts $S_1, \ldots, S_k$ so that (a) almost all the points in every cluster are drawn from the same component Gaussian, and (b) for every component, there is some cluster which contains almost all of the samples drawn from it. Of course, not every isotropic GMM is clusterable: for instance, if two components are identical, then it is information-theoretically impossible to detect which component a sample came from, and so recovering a good clustering is also impossible. In the isotropic setting, a necessary and sufficient condition for the GMM to be clusterable is some amount of mean separation, namely, that $\|\mu_i - \mu_j\|_2 \geq \Delta$ for all $i \neq j$, for some parameter Δ.

The question then becomes: how small can we take Δ so that we can still cluster? For simplicity, for the remainder of this section, let us assume that $w_i \geq 1/\text{poly}(k)$ for all $i = 1, \ldots, k$, that is, all the components have a nontrivial amount of weight. Information-theoretically, it is well-known (see e.g. [RV17]) that $\Delta = \Theta(\sqrt{\log k})$ separation is necessary and sufficient to achieve a clustering which correctly clusters more than an $1 - \epsilon$ fraction of the points with high probability, for all $\epsilon > 0$. Unfortunately, all known efficient algorithms for clustering at this separation rely on brute force methods, and run in exponential time.

If we restrict our attention to efficient estimators, the state of affairs is a bit more complicated. In fact, for over 20 years, the smallest separation that polynomial time algorithms could tolerate was $\Delta = \Omega(k^{1/4})$ [VW04]. It was not until recently that three concurrent papers [DKS18, HL18, KSS18] gave algorithms which could handle separation $\Delta = \Omega(k^\gamma)$ with runtime and sample complexity $O(d^{\text{poly}(1/\gamma)})$. In particular, whenever the separation is some fixed polynomial in k, these algorithms run in polynomial time. 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 time and sample complexity.

The barrier at quasipolynomial time These algorithms all get stuck at quasipolynomial time when $\Delta = \Theta(\text{poly} \log k)$ for the same reason. Fundamentally, all current algorithmic approaches to this problem 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 $t = \Theta(1/\gamma)$ moments approximately match those of a Gaussian along all projections 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 Gaussian in the appropriate norm. Since this results in a hard optimization problem whenever $t > 2$, these algorithms often solve some suitable relaxation of this using something like the Sum of Squares hierarchy. But, as the separation decreases, the algorithms must match more and more moments. In particular, to achieve $\Delta = \Theta(\text{poly} \log k)$, one must set $t = \text{poly} \log k$, that is, one must match polylogarithmically many moments. However, even writing down the degree $t = \text{poly} \log k$ moment tensor requires quasipolynomial time, and guaranteeing that the empirical moment tensor concentrates 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. [DKS17, BBH+20], 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, the question becomes:

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

1.1 Our Results

In this paper, we (almost) resolve this question in the affirmative. Namely, for all constants $c > 0$, 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. Our main theorem is:

Theorem 1.1 (informal, see Theorem 2.5, Corollary 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 $1/\text{poly}(k)$, 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 $n = \text{poly}(k, d)$, outputs a clustering which is correct for all of the points, with high probability. Moreover, this algorithm runs in time $\text{poly}(d, k)$.*

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 [RV17], 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 [RV17], 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 [RV17], 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

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

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. [TLV18] for an overview of the KLS conjecture).

In fact, the family of Poincaré distributions is the most general family of distributions for which the previously mentioned Sum of Squares-based methods for clustering are known to work. For any well-separated mixture of 1-Poincaré distributions, [KSS18] shows that one can recover the same guarantees as mentioned above: 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 1.2 (informal, see Theorem 2.3, Corollary 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 [Che21] 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 a logconcave distribution in polynomial time, as long as the separation is $\Omega(\exp(C\sqrt{\log k \log \log k}))$.

3

1.2 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.

1.2.1 Rough clustering via implicit moment tensors

The main conceptual contribution of our paper 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 $\mathrm{poly}(\Delta)$, 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

$$\mathbb{E}_{X \sim N(\mu, I)}[h_t(X)] = \mu^{\otimes t} \ . \tag{1}$$

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 (1) is that

$$\mathbb{E}_{X \sim \mathcal{M}}[h_t(X)] = \sum_{i=1}^{k} w_i \mu_i^{\otimes t} \ . \tag{2}$$

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) 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 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

$$\mathbb{E}_{R_t, X \sim N(\mu, I)} [R_t(X)] = \mu^{\otimes t} \ , \ \text{and} \ \mathbb{E}_{R_t, X \sim \mathcal{M}} [R_t(X)] = \sum_{i=1}^{k} w_i \mu_i^{\otimes t} \ . \tag{3}$$

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

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

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 (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.

---

[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.

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

$$M_t = \sum_{i=1}^k w_i \left(\mu_i \otimes \Gamma_{s-1} \mu_i^{\otimes(s-1)} \right) \left(\mu_i \otimes \Gamma_{s-1} \mu_i^{\otimes(s-1)} \right)^\top . \tag{4}$$

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

$$\Gamma_s = \Pi_s \left(I \otimes \Gamma_{s-1} \right) . \tag{5}$$

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 (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 (4) can be formed by rearranging the length-$(dk)^2$ vector

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

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

$$\sum_{i=1}^k w_i \left(I \otimes \Gamma_{s-1} \right)^{\otimes 2} \mu_i^{\otimes 2s} = \underset{R_{2s}, X \sim \mathcal{M}}{\mathbb{E}} \left[(I \otimes \Gamma_{s-1})^{\otimes 2} R_{2s}(X) \right] \tag{6}$$

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 (6), given samples from $\mathcal{M}$.

Putting it all together, this allows us to approximate M_s efficiently given Γ_{s-1}, which, by (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 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 2.3.

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 4, 5, 6 and 7. Note that in those sections, we work with a general Poincaré distribution but the outline follows the description here.

1.2.2 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 $\Delta = \text{poly}(\log k)$, it would not be able to achieve the nearly optimal separation in Theorem 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 9, 10 and 11.

1.3 Related work

The literature on mixture models—and Gaussian mixture models in particular—dates back to seminal work of Pearson [Pea94] and is incredibly vast. For conciseness, we will focus only on the most related papers here. Our results are most closely related to the aforementioned line of work on studying efficient algorithms for clustering and parameter estimation under mean-separation conditions [Das99, DS07, AK05, VW04, DKS18, HL18, RV17, KSS18, DK20]. However, none of these algorithms can handle polylogarithmic separation in polynomial time.

We also note that a number of papers also generalize from mixtures of Gaussians to mixtures of more general classes of distributions [AM05, KK10, MVW17, HL18, KSS18]. 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 [KSS18], which is also the class of distributions we consider here.

Another line of work focuses on parameter estimation for mixtures models without separation conditions [KMV10, BS15, MV10, HP15, DK20]. These papers typically make much weaker assumptions on the components, namely, that they are statistically distinguishable, and the goal is to recover the parameters of the mixture, even in settings where clustering is impossible. However, typically, these algorithms incur a sample complexity and runtime which is exponential in the number of components; indeed, [HP15] demonstrate that this is tight, even in one dimension, when the Gaussians can have different variances.

To circumvent this, researchers have also considered the easier notion of *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. 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 [FOS08, DK14, AJOS14, LS17, ABDH+18]. When the hypothesis is only constrained to be semi-proper, polynomial time algorithms are known in the univariate setting [DL01, BSZ15, LLM21], but these do not extend to high dimensional settings. In the more challenging high dimensional regime, a remarkable recent result of [DK20] 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. [DL01, CDSS14b, CDSS14a, ADLS17], but as was the case with proper learning, these techniques do not extend nicely to high dimensional settings.

An alternate assumption that has been considered in the literature is that the means satisfy some algebraic non-degeneracy assumptions. For instance, such assumptions are satisfied in smoothed analysis settings [HK13, ABG+14, BCMV14, GHK15]. Often in these settings, access to constant order moments suffice (e.g. 3rd or 4th moments), although we note that [BCMV14] is a notable exception. In contrast, we do not make any non-degeneracy assumptions, and our methods need to access much higher moments.

We also mention a line of work studying the theoretical behavior of the popular *expectation-maximization* (EM) algorithm for learning mixtures of Gaussians [DS07, DTZ17, XHM16]. 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 3 Gaussians [Wu83, DTZ17].

Finally, 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 [HSSS16, MSS16, SS17, HSS19] and refuting random CSPs [RRS17]. 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 [GS12, ST21]. 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 Formal Problem Setup and Our 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.

2.1 Clustering Mixtures of Poincare Distributions

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

Definition 2.1 (Poincare Distribution). *For a parameter σ, we say a distribution $\mathcal{D}$ on $\mathbb{R}^d$ is σ-Poincare if for all differentiable functions $f : \mathbb{R}^d \to \mathbb{R}$,*

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

Definition 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 2.3. *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 $n = \operatorname{poly}((kd/(w_{\min}\alpha))^{C/c})$ samples from $\mathcal{M}$ and $\mathcal{D}$ and runs in $\operatorname{poly}(n)$ 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.*

An immediate consequence of Theorem 2.3 is that we can cluster samples from the mixture with accuracy better than any inverse polynomial.

Corollary 2.4. *Under the same assumptions as Theorem 2.3, we can recover the ground-truth clustering of the samples with high probability i.e. we output k clusters $S_1, \ldots S_k$ 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] 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 $\log k$)

2.2 Clustering Mixtures of Gaussians

In the case where the distribution $\mathcal{D}$ in the setup in Section 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 2.5. *Let $\mathcal{M} = w_1 N(\mu_1, I) + \cdots + w_k N(\mu_k, I)$ 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 2.6. *Under the same assumptions as Theorem 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.

2.3 Organization

In Section 3, we introduce basic notation and prove a few basic facts that will be used later on.

Clustering Test: In Sections 4 - 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 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

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

and such that R_t can be written as the sum of polynomially many rank-1 tensors (see Corollary 4.10). In Section 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 6, we combine the techniques in Section 4 and Section 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 $\mu_1^{\otimes t}, \dots \mu_k^{\otimes t}$ (see Lemma 6.5). Finally, in Section 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 8, we prove Theorem 2.3, our main result for mixtures of Poincare distributions. It will follow fairly easily from the guarantees of the clustering test in Section 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 4 - 7 can still be used. In particular, we will need a few quantitatively stronger versions of the bounds in Sections 4 - 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 9. Then in Sections 10 and 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 10, we introduce a few basic building blocks in our recursive clustering algorithm and we put them together in Section 11.

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.

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 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 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 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 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 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.*

11

Definition 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 $a_1 + \cdots + a_n$.

3.2 Properties of Poincare Distributions

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

Fact 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 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*

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

3.3 Basic Observations

Before we begin with the main proofs of Theorem 2.3 and Theorem 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 Appendix A. 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.

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 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.

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 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 $t \geq 1$,*

$$P_{t,\mathcal{D}}(x) = x^{\otimes t} - \sum_{sym} D_{1,\mathcal{D}} \otimes P_{t-1,\mathcal{D}}(x) - \sum_{sym} D_{2,\mathcal{D}} \otimes P_{t-2,\mathcal{D}}(x) - \cdots - D_{t,\mathcal{D}}. \tag{7}$$

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 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. ∎

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

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

where we imagine $x = (x_1, \ldots, x_d)$ so x_i is the ith coordinate of x and

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

denotes the ith 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

$$\frac{\partial P_t(x)}{\partial x_i} = \sum_{\text{sym}} e_i \otimes x^{\otimes t-1} - \sum_{\text{sym}} D_1 \otimes e_i \otimes P_{t-2}(x) - \cdots - \sum_{\text{sym}} D_{t-2} \otimes e_i \otimes P_1(x) - \sum_{\text{sym}} D_{t-1} \otimes e_i$$

$$= \sum_{\text{sym}} e_i \otimes \left(x^{\otimes t-1} - \sum_{\text{sym}} D_1 \otimes P_{t-2}(x) - \cdots - D_{t-1} \right)$$

$$= \sum_{\text{sym}} e_i \otimes P_{t-1}(x)$$

where in the last step we again used (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 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*

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

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

$$\mathbb{E}_{z \sim \mathcal{D}(\mu)} [(v \cdot \textsf{flatten}(P_{1,\mathcal{D}}(z)))^2] = (v \cdot \mu)^2 + \underset{z \sim \mathcal{D}(\mu)}{\textsf{Var}} (v \cdot \textsf{flatten}(P_{1,\mathcal{D}}(z))) \leq (v \cdot \mu)^2 + \|v\|^2$$

$$\leq (\|\mu\|^2 + 1) \|v\|^2$$

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

$$\mathbb{E}_{z \sim \mathcal{D}(\mu)} [(v \cdot \textsf{flatten}(P_{t,\mathcal{D}}(z)))^2] = (v \cdot \textsf{flatten}(\mu^{\otimes t}))^2 + \underset{z \sim \mathcal{D}(\mu)}{\textsf{Var}} (v \cdot \textsf{flatten}(P_{t,\mathcal{D}}(z)))$$

$$\leq \|\mu\|^{2t} \|v\|^2 + \mathbb{E}_{z \sim \mathcal{D}(\mu)} \left[\sum_{i=1}^{d} \left(v \cdot \frac{\partial P_{t,\mathcal{D}}(x)}{\partial x_i} \right)^2 \right]$$

$$= \|\mu\|^{2t} \|v\|^2 + \mathbb{E}_{z \sim \mathcal{D}(\mu)} \left[\sum_{i=1}^{d} \left(\sum_{j=1}^{t} v_{\eta_j = i} \cdot P_{t-1,\mathcal{D}}(z) \right)^2 \right]$$

$$\leq \|\mu\|^{2t} \|v\|^2 + \mathbb{E}_{z \sim \mathcal{D}(\mu)} \left[t \sum_{j=1}^{t} \sum_{i=1}^{d} \left(v_{\eta_j = i} \cdot P_{t-1,\mathcal{D}}(z) \right)^2 \right]$$

$$\leq \|\mu\|^{2t} \|v\|^2 + t \sum_{j=1}^{t} \sum_{i=1}^{d} (\|\mu\|^2 + (t-1)^2)^{t-1} \|v_{\eta_j = i}\|^2$$

$$= \|\mu\|^{2t} \|v\|^2 + t^2 (\|\mu\|^2 + (t-1)^2)^{t-1} \|v\|^2$$

$$\leq (\|\mu\|^2 + t^2)^t \|v\|^2$$

14

where in the above manipulations, we first used Claim 4.3, then the fact that $\mathcal{D}$ is 1-Poincare, then Claim 4.4, then Cauchy Schwarz, then the inductive hypothesis, and finally some direct manipulation. This completes the inductive step and we are done. ∎

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 $x_1, \ldots, x_t \in \mathbb{R}^d$. 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 $x_2, \ldots, x_t \sim \mathcal{D}$, then Claim 4.3 will immediately imply that the expectation is $\mu^{\otimes t}$. The key properties are stated formally in Corollary 4.10, Corollary 4.11 and Corollary 4.12.

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

Claim 4.6. *We have*

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

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 $S_1, \ldots, S_a$ are nonempty for some $1 \leq a \leq t$. Recall the recursive definition of P_t in (7). There are exactly a terms on the RHS of (7) that can produce the monomial A. These terms are

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

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 4.7. *For $x_1, \ldots, x_t \in \mathbb{R}^d$, define the polynomial*

$$Q_t(x_1, \ldots, x_t) = \sum_{\substack{S_1 \cup \cdots \cup S_t = [t] \\ |S_i \cap S_j| = 0}} \frac{(-1)^{\mathcal{C}\{S_1, \ldots, S_t\}}}{\binom{t-1}{\mathcal{C}\{S_1, \ldots, S_t\} - 1}} \left(P_{|S_1|}(x_1) \right)^{(S_1)} \otimes \cdots \otimes \left(P_{|S_t|}(x_t) \right)^{(S_t)} . \tag{8}$$

15

Lemma 4.8. *All nonzero monomials in Q_t either have total degree t in the variables $x_1, \ldots, x_t$ or are constant.*

Proof. Consider substituting the formula in Claim 4.6 into the RHS for all occurrences of P. Consider a monomial

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

where $U_1, \ldots, U_a, V_1, \ldots, V_b$ 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 $S_{j_1}, \ldots, S_{j_c}$ 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 $\{i_1, \ldots, i_a\} \subseteq \{j_1, \ldots, j_c\}$. 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 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

$$\sum_{c=a}^{a+b} \frac{(-1)^c}{\binom{t-1}{c-1}}\binom{t-a}{c-a}(-1)^b \sum_{\substack{B_1 \cup \cdots \cup B_c = [b] \\ B_i \cap B_j = \emptyset \\ B_{a+1}, \ldots, B_c \neq \emptyset}} |B_1|! \cdots |B_c|! \,. \tag{9}$$

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 (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

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

16

where the last equality follows from counting using stars and bars. Now we can plug back into (9). Assuming that $a, b \geq 1$, the coefficient of the monomial A is

$$\sum_{c=a}^{a+b} \frac{(-1)^c}{\binom{t-1}{c-1}} \binom{t-a}{c-a}(-1)^b b! \binom{b+a-1}{c-1} = (-1)^b b! \sum_{c=a}^{a+b} \frac{(-1)^c (t-a)!(b+a-1)!}{(c-a)!(t-1)!(b+a-c)!}$$

$$= (-1)^b \frac{(t-a)!(b+a-1)!}{(t-1)!} \sum_{c=a}^{a+b} (-1)^c \binom{b}{c-a}$$

$$= 0 \, .$$

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 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 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 4.8 and the definition of $R_t(x_1, \ldots, x_{2t})$ because the constant terms cancel out and the degree-t terms clearly match the RHS of the desired expression. ∎

Corollary 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 4.3 and Claim 4.5.

Corollary 4.11. *We have*

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

and for fixed z_1, we have

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

Proof. Using the definition of Q_t in (8) and Claim 4.3, the expectations of all of the terms are 0 except for the leading term $P_t(z_1)$. Thus,

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

Also by Claim 4.3,

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

and this gives us the two desired identities. ∎

Corollary 4.12. *Let $\mathcal{D}$ be a distribution that is 1-Poincare. We have*

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

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

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

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

Now by Cauchy Schwarz again,

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

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

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 4.5)

$$C_2 \preceq \left(\sum_{\substack{S_1 \cup \cdots \cup S_t = [t] \\ |S_i \cap S_j| = 0}} \frac{(\|\mu\|^2 + |S_1|^2)^{|S_1|} |S_2|^{2|S_2|} \cdots |S_t|^{2|S_t|}}{|S_1|! \cdots |S_t|!} \right) I_{d^t}$$

$$\preceq \left(20^t \sum_{\substack{S_1 \cup \cdots \cup S_t = [t] \\ |S_i \cap S_j| = 0}} \frac{\left(\|\mu\|^{2|S_1|} + (|S_1|!)^2 \right) (|S_2|!)^2 \cdots (|S_t|!)^2}{|S_1|! \cdots |S_t|!} \right) I_{d^t}$$

$$\preceq \left(20^t (\|\mu\|^{2t} + 1) \sum_{s_1 + \cdots + s_t = t} \binom{t}{s_1, \ldots, s_t} s_1! s_2! \cdots s_t! \right) I_{d^t}$$

$$\preceq (40t)^t (\|\mu\|^{2t} + 1) I_{d^t}.$$

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

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

Similarly, we have

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

and putting everything together, we conclude

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

and we are done. $\blacksquare$

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.

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 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*

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

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 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 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

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

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 5.4. *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 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*

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

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.

6 Implicitly Estimating the Moment Tensor

In this section, we combine the iterative projection techniques from Section 5 with the estimators from Section 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 $\Pi_t, \ldots, \Pi_1$ such that the row-span of $\Gamma_{\Pi_t, \ldots, \Pi_1}$ essentially contains all of the flattenings of $\mu_1^{\otimes t}, \ldots, \mu_k^{\otimes t}$.

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 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 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 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 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 6.1 and 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 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 4, namely Corollary 4.11 and Corollary 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 4.9)

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

    Let K_i be the rearrangement of X_i into a square $dk \times dk$-dimensional matrix
Output: $A = (K_1 + \cdots + K_n)/n$

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 $\mathrm{poly}(ndk, t^t)$ 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 6.3. *Algorithm 1 can be implemented to run in $\mathrm{poly}(n, d, k, t^t)$ time.*

Proof. First we analyze the runtime of Algorithm 2. Note that by Corollary 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 5.4, this means that computing

$$X_i = ((I_d \otimes_{\mathrm{kr}} \Gamma_{\Pi_{s-1},\ldots,\Pi_1}) \otimes_{\mathrm{kr}} (I_d \otimes_{\mathrm{kr}} \Gamma_{\Pi_{s-1},\ldots,\Pi_1}))\mathsf{flatten}\left(Q_{2s}(z_i, x_1, \ldots, x_{4s-1})\right)$$
$$= ((I_d \otimes_{\mathrm{kr}} \Gamma_{\Pi_{s-1},\ldots,\Pi_1}) \otimes_{\mathrm{kr}} (I_d \otimes_{\mathrm{kr}} \Gamma_{\Pi_{s-1},\ldots,\Pi_1}))(\mathsf{flatten}(V_1) + \cdots + \mathsf{flatten}(V_l))$$

can be done in $\mathrm{poly}(d, k, s^s)$ time by expanding out the RHS and computing each term separately. It is then immediate that all of Algorithm 2 runs in $\mathrm{poly}(n, d, k, s^s)$ 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 $\mathrm{poly}(n, d, k, t^t)$ time. ∎

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 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 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*

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

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 6.5 will involve following the outline at the beginning of this section but quantitatively tracking the errors more precisely. Before we prove Lemma 6.5, we first prove a preliminary claim that our estimation error $\left\|A_{2s} - \widetilde{A_{2s}}\right\|_F$ is small.

Claim 6.6. *Assume that for a fixed integer s, Algorithm 2 is run with a number of samples*

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

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 4.11, we have that

$$\mathbb{E}[K_i] = A_{2s} \tag{10}$$

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 5.2). By Corollary 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 (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

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

23

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 6.5.

Proof of Lemma 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

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

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 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 6.6 again, we must have

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

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

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

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

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

and thus

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

Finally, note that

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

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 6.6 and is clearly at most $n^{-0.2}$ as long as we choose n sufficiently large. This completes the proof. ∎

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 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 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

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 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 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 7.2 requires that the row span of $\Gamma_{\Pi_t, \ldots, \Pi_1}$ essentially contains $\mathsf{flatten}(\mu_i^{\otimes t})$ (which can be guaranteed by Algorithm 1 and Lemma 6.5). Lemma 7.1 actually does not require anything about $\Pi_t, \ldots, \Pi_1$ (other than the fact that they are actually projections).

Lemma 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 $\Pi_t, \ldots, \Pi_2 \in \mathbb{R}^{k \times dk}, \Pi_1 \in \mathbb{R}^{k \times d}$ 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 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 $\Pi_t, \ldots, \Pi_2 \in \mathbb{R}^{k \times dk}, \Pi_1 \in \mathbb{R}^{k \times d}$ 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).*

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 7.1 and Lemma 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 7.1 (which is much easier to prove than Lemma 7.2). In fact, a direct variance bound using Corollary 4.12 will suffice.

Proof of Lemma 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 $n = ((10^3 t)^t/\delta)^3$ 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 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. $\blacksquare$

A direct variance bound will not work for Lemma 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 4.2). Essentially, we will argue that under the conditions of Lemma 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 $\Gamma_{\Pi_t, \ldots, \Pi_1}$, this implies that

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

must have large norm. By Corollary 4.11,

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

26

so if we average over enough independent samples for $z_1, \ldots, z_{2t-1}$ 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 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*

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

Proof. WLOG we may assume v is a unit vector. Let $a = v \cdot x$. We will use Claim 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 3.7, we have

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

Now by Claim 4.6, we have

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

This completes the proof. ∎

Now we are ready to prove Lemma 7.2.

Proof of Lemma 7.2. Let $v = \mu_i / \|\mu_i\|$. By Fact 3.7, with probability at least $1 - \delta/10$, we have

$$v \cdot z \geq 0.9 \, \|\mu_i\| \, . \tag{11}$$

Now by Claim 7.3, we have

$$\langle P_t(z), v^{\otimes t} \rangle \geq (0.8 \, \|\mu_i\|)^t \, . \tag{12}$$

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

$$\Gamma_{\Pi_t, \ldots, \Pi_1}^T \Gamma_{\Pi_t, \ldots, \Pi_1} \mathsf{flatten}(v^{\otimes t}) = \mathsf{flatten}(v^{\otimes t}) + u$$

where u is a vector with

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

27

By Claim 4.5 and Markov's inequality, we have that with probability at least $1 - \delta/10$ over the choice of z,

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

where the last step follows from our condition on ϵ. Now we can combine this with (12) to get that

$$\langle \mathsf{flatten}(P_t(z)), \Gamma^T_{\Pi_t,\ldots,\Pi_1}\Gamma_{\Pi_t,\ldots,\Pi_1}\mathsf{flatten}(v^{\otimes t})\rangle \geq (0.8\|\mu_i\|)^t - 1 \geq (0.5\|\mu_i\|)^t. \tag{13}$$

Now let

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

Note that $\|w\| \leq 1$ because it is the projection of a unit vector $\mathsf{flatten}(v^{\otimes t})$ onto some subspace. By Corollary 4.12 and Markov's inequality, with probability at least $1 - \delta/10$ over the randomness in z, we have

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

Assuming that the above holds, we now treat z as fixed. Note that the mean of $R(z, z_1, \ldots, z_{2t-1})$ is $P_t(z)$ by Corollary 4.11. Let $\overline{R}$ be obtained by averaging $R(z, z_1, \ldots, z_{2t-1})$ over $n = ((10^3 t)^t/\delta)^3$ independent trials for $z_1, \ldots, z_{2t-1}$. Of course the mean of $\overline{R}$ is also $P_t(z)$. Using the above, we have that

$$\mathbb{E}\left[(w \cdot (\mathsf{flatten}(\overline{R}) - \mathsf{flatten}(P_t(z))))^2\right] = \frac{1}{n}\mathbb{E}_{z_1,\ldots,z_{2t-1}\sim\mathcal{D}}\left[(w \cdot (\mathsf{flatten}(R(z, z_1, \ldots, z_{2t-1}) - P_t(z))))^2\right]$$

$$\leq \frac{1}{n}\mathbb{E}_{z_1,\ldots,z_{2t-1}\sim\mathcal{D}}\left[(w \cdot \mathsf{flatten}(R(z, z_1, \ldots, z_{2t-1})))^2\right]$$

$$\leq (\delta/(10^3 t)^t)^2(20t)^{2t}(\|\mu_i\|^{2t} + 1)$$

$$\leq \frac{\delta}{10}\left(\frac{\|\mu_i\|}{20}\right)^{2t}.$$

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

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

If this holds, then combining with (13) implies

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

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

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

Note that the expression $\Gamma_{\Pi_t,\ldots,\Pi_1}\mathsf{flatten}(\overline{R})$ 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. ∎

8 Learning Mixtures of Poincare Distributions

We are now ready to prove Theorem 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 3.7. Note that we can take pairwise differences between samples to simulate access to the mixture

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

28

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 2.3 will follow easily from it.

Lemma 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 $n = \text{poly}((kd/(w^* \delta))^{C/c})$ samples from $\mathcal{M}$ and $\mathcal{D}$ and runs in $\text{poly}(n)$ 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 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

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

Note that by the above, the runtime and sample complexity of $n = \text{poly}((kd/(w^* \delta))^{C/c})$ suffices for us to apply Lemma 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

$$\left\| \Gamma_{\Pi_s, \ldots, \Pi_1} \text{flatten}((\mu_i - \mu_j)^{\otimes s}) \right\| \geq (1 - s\epsilon) \|\mu_i - \mu_j\|^s \tag{14}$$

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

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

29

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 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 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 (14), we conclude that the output is REJECT with probability at least $1 - \delta$ and we are done. ∎

Using Lemma 8.1, it is not difficult to prove Theorem 2.3.

Proof of Theorem 2.3. Recall by the reduction in Section 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 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 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 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 $|S_i| \geq 0.9w_{\min}l$ 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 $|S_i| < 0.9w_{\min}l$.

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 2.4 follows as an immediate consequence of Theorem 2.3.

Proof of Corollary 2.4. Let the estimated means computed by Theorem 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 2.3, there is a permutation π such that $\|\widetilde{\mu_{\pi(i)}} - \mu_i\| \leq \alpha$ for all i. If z is a sample from $\mathcal{D}(\mu_i)$, then by the tail bound in Fact 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. ■

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.

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 [Kan20]. 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 6 and 7.

Definition 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 9.2 (Hermite Polynomial Tensor (see [Kan20])). *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 9.3 (See [Kan20]). *Let $G = N(\mu, I)$. Then*

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

Claim 9.4 (See [Kan20]). *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 4.2) for the standard Gaussian.

Claim 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 9.4 into the recursion in (7) to verify the inductive step. ∎

We could get a bound on the variance of the Hermite polynomials using Claim 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 4 for when the mean μ is nonzero.

Claim 9.6 (From [Kan20]). *We have*

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

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 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 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 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 9.2 into the LHS and then use Claim 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. ∎

32

9.2 Implicit Representations of Hermite Polynomials

Similar to Section 4.3, we will need implicit representations of the Hermite polynomial tensors. While technically the representation in Definition 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 5 and thus we will again, similar to Section 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 4.3.

Definition 9.9. *For $x_1, \ldots, x_t \in \mathbb{R}^d$, define the polynomial*

$$Q_t(x_1, \ldots, x_t) = \sum_{\substack{S_1 \cup \cdots \cup S_t = [t] \\ |S_i \cap S_j| = 0}} \frac{(-1)^{\mathcal{C}\{S_1, \ldots, S_t\}}}{\binom{t-1}{\mathcal{C}\{S_1, \ldots, S_t\}-1}} \left(h_{|S_1|}(x_1)\right)^{(S_1)} \otimes \cdots \otimes \left(h_{|S_t|}(x_t)\right)^{(S_t)} . \tag{15}$$

Definition 9.10. *For $x_1, \ldots, x_{2t} \in \mathbb{R}^d$, define the polynomial*

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

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 9.5, we have the following (which is the exact same as Corollary 4.10).

Corollary 9.11. *We have the identity*

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

Claim 9.5 also implies that Corollary 4.11 and Corollary 4.12 still hold and can be used in the analysis. We now prove the one strengthened bound that we will need.

Lemma 9.12. *We have*

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

Proof. Note that by Claim 9.8 and Corollary 9.7,

$$\mathbb{E}_{z_1, \ldots, z_{2t} \sim N(0,I)} \left[\mathsf{flatten}(R_t(z_1, \ldots, z_{2t}))^{\otimes 2}\right]$$

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

$$\preceq 2 \sum_{\substack{S_1 \cup \cdots \cup S_t = [t] \\ |S_i \cap S_j| = 0}} |S_1|! |S_2| \cdots |S_t|! I_{d^t}$$

$$\preceq 2 I_{d^t} \sum_{\substack{s_1 + \cdots + s_t = t \\ s_i \geq 0}} \binom{t}{s_1, \ldots, s_t} s_1! \cdots s_t!$$

$$\preceq (2t)^t I_{d^t}$$

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 4.12 is that the RHS is t^t instead of t^{2t}. This is the key to improving the separation.

9.3 Testing Samples

All of the algorithms and results in Section 6 still hold (with the distribution $\mathcal{D}$ set to $N(0, I)$). We also run the testing algorithm in Section 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 7.1 and Lemma 7.2. The strengthened versions are stated below.

Lemma 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 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*

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

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 7.1 and Lemma 7.2. First, in Lemma 9.13, the bound on τ involves $t^{t/2}$ instead of t^t. Second, in Lemma 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 $\delta = \text{poly}(w^*/k)$ 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 9.13 is essentially exactly the same as the proof of Lemma 7.1.

Proof of Lemma 9.13. Exactly the same as the proof of Lemma 7.1 except use Lemma 9.12 instead of Corollary 4.12 to bound the variance of R_t. ∎

The proof of Lemma 9.14 requires one additional modification. In particular, we will use a stronger version of Claim 7.3 that we now prove.

Claim 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*

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

Proof. WLOG v is a unit vector. Let $a = v \cdot x$. Using Definition 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

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

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 9.14. Exactly the same as the proof of Lemma 7.2 except using Claim 9.15 instead of Claim 7.3. ∎

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 2.3.

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 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 10.12 which allows us to do full clustering when the maximum separation is at most $\text{poly}(\log(k/w_{\min}))$.

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

Definition 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 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 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 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 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 $\text{poly}(d, 1/p)$ 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 10.8. *Consider a Gaussian $N(\mu, I)$ in $\mathbb{R}^d$. For any parameter β, we have*

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

The next inequality is closely connected to the notion of stability (see [DK19]) that has recently proved very useful in the field of robust statistics.

Claim 10.9 (See [DK19]). *Consider the standard Gaussian $N(0, I)$ in $\mathbb{R}^d$. Let $\epsilon > 0$ be some parameter. Given $n \geq (d/\epsilon)^8$ 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 $|S| \geq \epsilon n$, we have*

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

10.2 Clustering Test

Recall that for arbitrary Poincare distributions, one of the key ingredients was Lemma 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 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 $s \geq (\log k/(w^* \delta))^{1/2+c}$ for some $0 < c < 1$. Also assume that

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

for some C. There is an algorithm that takes $n = \mathrm{poly}((kd/(w^* \delta))^{C/c})$ samples from $\mathcal{M}$ and runs in $\mathrm{poly}(n)$ 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 8.1 except we use Lemma 9.14 and Lemma 9.13 in place of Lemma 7.2 and Lemma 7.1 respectively. ∎

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 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 10.10 and the mean μ' of samples z_i' such that (z, z_i') passes the test in Lemma 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 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 $v = (\mu - \mu')/ \|\mu - \mu'\|$
Test if v is a $(0.8w^*, 0.8\Delta)$ signal direction using Claim 10.7
Output v if test passes

Lemma 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 $\|\mu_i - \mu_j\| \leq O((k/w^*)^2)$ for all i, j. Also assume that there are i, j such that $w_i, w_j \geq w^*$ and $\|\mu_i - \mu_j\| \geq (\log(k/w^*))^{1/2+c}$ for some positive constant c. There is an algorithm that takes $n = \text{poly}((dk/w^*)^{1/c})$ samples from $\mathcal{M}$ and $\text{poly}(n)$ runtime and with high probability outputs a unit vector v such that v is a $(w^*/2, \max_{w_i, w_j \geq w^*} \|\mu_i - \mu_j\| /2)$-signal direction.*

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

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 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 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 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 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. ∎

10.4 Full Clustering with Bounded Maximum Separation

Similar to Theorem 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 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 $n = \mathrm{poly}((dk/w^*)^{1/c})$ samples from $\mathcal{M}$ and $\mathrm{poly}(n)$ 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 2.3 except using the test in Lemma 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 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 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 2.3, 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 $|S_i| \geq 0.9 w^* l$ 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 $0.9 w_i l$ 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. ∎

11 Clustering Part 2: Recursive Clustering

In this section, we put together the building blocks from Section 10 in our complete clustering algorithm and complete the proof of Theorem 2.5.

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

Definition 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 $red_{V,p,r}(\mathcal{M})$.*

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 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*

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

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 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 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*

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

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 $red_{V,p,r}^{(\theta)}(\mathcal{M})$.

Combining Claim 10.8 and Claim 11.4, we deduce that we can truncate after reducing by a checker while changing the distribution by only a negligible amount.

Corollary 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 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 10.5). This will allow us to apply Lemma 10.12 to find a new signal direction in $V^\perp$.

Claim 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

$$\alpha^2_{(\gamma-1)\theta} \leq \alpha_{(\gamma+1)\theta}. \tag{16}$$

This is because by Claim 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 (16) can fail for at most half of the choices of γ and we are done. ∎

41

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 $S = \{\widetilde{\mu_1}, \dots, \widetilde{\mu_t}\}$ that are all separated and such that there is one that is close to each true mean.

Claim 11.8. *Let $\mathbb{N}(\mu_1, I), \dots, N(\mu_l.I)$ 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 $S = \{\widetilde{\mu_1}, \dots, \widetilde{\mu_r}\}$ 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. ∎

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 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 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 11.10. If the test passes, in the second phase, we can simply use Lemma 10.12 to fully cluster the remaining submixture 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 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 11.9
    Test if means in (V, p, r) have max-separation at most $\log^4(k/w_{\min})$ using Lemma 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 11.11)
Remove samples from $N(\mu_i, I)$ and recurse on remaining

The next lemma formalizes the guarantees of the refinement step.

Lemma 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 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 $\mathsf{red}_{V,p,\beta+\gamma\theta}^{(\theta)}(\mathcal{M})$ are at least $0.9w^*$. Now by Corollary 11.6 we can, with high probability, simulate a polynomial number of samples from $\mathsf{red}_{V,p,\beta+\gamma\theta}^{(\theta)}(\mathcal{M})$ 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 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 $\mathsf{red}_{V,p,\beta+\gamma\theta}^{(\theta)}(\mathcal{M})$. 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 $\mathsf{red}_{V,p,\beta+\gamma\theta}^{(\theta)}(\mathcal{M})$ 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

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

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

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

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

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

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 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 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 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 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 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 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 10.11, we will find a $(0.4w^*, 0.4(\log(k/w^*))^4)$-signal direction with high probability and thus we will REJECT. $\blacksquare$

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

Lemma 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* test$(z) =$ accept

- *For all other* $i' \neq i$, *for a random sample* $z \sim N(\mu_i, I)$, *with high probability* test$(z) =$ 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 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 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 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 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 2.5, for learning clusterable mixtures of Gaussians.

Proof of Theorem 2.5. First we apply the reductions in Section 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 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 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 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 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 11.11 to isolate one of the components of $\mathcal{M}$ (again with $w^* = w_{\min}$). With high probability, we can take $\mathrm{poly}(k/w_{\min})$ 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 2.6 follows as an immediate consequence of Theorem 2.5.

Proof of Corollary 2.6. Let the estimated means computed by Theorem 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 2.5, there is a permutation π such that $\|\widetilde{\mu_{\pi(i)}} - \mu_i\| \leq \alpha$ 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. ∎

References

[ABDH+18] 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.

[ABG+14] Joseph Anderson, Mikhail Belkin, Navin Goyal, Luis Rademacher, and James Voss. The more, the merrier: the blessing of dimensionality for learning large gaussian mixtures. In *Conference on Learning Theory*, pages 1135–1164. PMLR, 2014.

[ADLS17] 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.

[AJOS14] Jayadev Acharya, Ashkan Jafarpour, Alon Orlitsky, and Ananda Theertha Suresh. Near-optimal-sample estimators for spherical gaussian mixtures. *arXiv preprint arXiv:1402.4746*, 2014.

[AK05] Sanjeev Arora and Ravi Kannan. Learning mixtures of separated nonspherical gaussians. *The Annals of Applied Probability*, 15(1A):69–92, 2005.

[AM05] Dimitris Achlioptas and Frank McSherry. On spectral learning of mixtures of distributions. In *International Conference on Computational Learning Theory*, pages 458–469. Springer, 2005.

[BBH+20] 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.

[BCMV14] 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.

[BS15] Mikhail Belkin and Kaushik Sinha. Polynomial learning of distribution families. *SIAM Journal on Computing*, 44(4):889–911, 2015.

[BSZ15] 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.

[CDSS14a] 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.

[CDSS14b] 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.

[Che21] Yuansi Chen. An almost constant lower bound of the isoperimetric coefficient in the kls conjecture. *Geometric and Functional Analysis*, 31(1):34–61, 2021.

[Das99] Sanjoy Dasgupta. Learning mixtures of gaussians. In *40th Annual Symposium on Foundations of Computer Science (Cat. No. 99CB37039)*, pages 634–644. IEEE, 1999.

[DK14] 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.

[DK19] Ilias Diakonikolas and Daniel M. Kane. Recent advances in algorithmic high-dimensional robust statistics, 2019.

[DK20] 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.

[DKS17] 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.

[DKS18] 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, 2018.

[DL01] Luc Devroye and Gábor Lugosi. *Combinatorial methods in density estimation*. Springer Science & Business Media, 2001.

[DS07] 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.

[DTZ17] 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.

[FOS08] 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.

[GHK15] 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.

[GS12] 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.

[HK13] 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, 2013.

[HL18]   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, 2018.

[HP15]   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.

[HSS19]   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.

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

[Kan20]   Daniel M. Kane. Robust learning of mixtures of gaussians. *CoRR*, abs/2007.05912, 2020.

[KK10]   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.

[KMV10]   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.

[KSS18]   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.

[LLM21]   Jerry Li, Allen Liu, and Ankur Moitra. Sparsification for sums of exponentials and its algorithmic applications. *arXiv preprint arXiv:2106.02774*, 2021.

[LS17]   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.

[MSS16]   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.

[MV10]   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.

[MVW17]   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.

[Pea94]   Karl Pearson. Contributions to the mathematical theory of evolution. *Philosophical Transactions of the Royal Society of London. A*, 185:71–110, 1894.

[RRS17]   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.

[RV17]   Oded Regev and Aravindan Vijayaraghavan. On learning mixtures of well-separated gaussians, 2017.

[SS17]   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.

[ST21]   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.

[TLV18]   Yin Tat Lee and Santosh S Vempala. The kannan-lovász-simonovits conjecture. *arXiv e-prints*, pages arXiv–1807, 2018.

[VW04]   Santosh Vempala and Grant Wang. A spectral algorithm for learning mixture models. *Journal of Computer and System Sciences*, 68(4):841–860, 2004.

[Wu83]   CF Jeff Wu. On the convergence properties of the em algorithm. *The Annals of statistics*, pages 95–103, 1983.

[XHM16]   Ji Xu, Daniel Hsu, and Arian Maleki. Global analysis of expectation maximization for mixtures of two gaussians. *arXiv preprint arXiv:1608.07630*, 2016.

A   Basic Reductions

Here we explain the reductions for Section 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 3.7, with $1 - 2^{-10(d+k)/w_{\min}}$ probability, a sample $z \sim \mathcal{D}(\mu_i)$ satisfies
$$\|z - \mu_i\| \le 10^4 \cdot (d+k)/w_{\min} \,.$$
Now, we look for a pair of samples z, z' such that
$$\|z - z'\| \ge 10^6 ((d+k)/w_{\min})^2 \,.$$
Note that with high probability, such a pair exists if
$$\|\mu_i - \mu_j\| \ge 1.1 \cdot 10^6 ((d+k)/w_{\min})^2$$
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
$$|\langle v, \mu_i - \mu_j \rangle| \ge 0.99 \cdot 10^6 ((d+k)/w_{\min})^2 \,.$$

Now imagine projecting all of the samples onto the direction v. Note that by Fact 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 $\|\mu_i - \mu_j\| \le O(((d+k)/w_{\min})^2)$. 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 $d \le k$ 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.