Small Covers for Near-Zero Sets of Polynomials and Learning Latent Variable Models

Ilias Diakonikolas
Department of Computer Sciences
University of Wisconsin, Madison
U.S.A.
ilias@cs.wisc.edu

Daniel M. Kane
Departments of CSE & Mathematics
University of California, San Diego
U.S.A.
dakane@ucsd.edu

***Abstract*—Let V be any vector space of multivariate degree-d homogeneous polynomials with co-dimension at most k, and S be the set of points where all polynomials in V *nearly* vanish. We establish a qualitatively optimal upper bound on the size of ϵ-covers for S, in the ℓ_2-norm. Roughly speaking, we show that there exists an ϵ-cover for S of cardinality $M = (k/\epsilon)^{O_d(k^{1/d})}$. Our result is constructive yielding an algorithm to compute such an ϵ-cover that runs in time $\mathrm{poly}(M)$.**

Building on our structural result, we obtain significantly improved learning algorithms for several fundamental high-dimensional probabilistic models with hidden variables. These include density and parameter estimation for k-mixtures of spherical Gaussians (with known common covariance), PAC learning one-hidden-layer ReLU networks with k hidden units (under the Gaussian distribution), density and parameter estimation for k-mixtures of linear regressions (with Gaussian covariates), and parameter estimation for k-mixtures of hyperplanes. Our algorithms run in time *quasi-polynomial* in the parameter k. Previous algorithms for these problems had running times exponential in $k^{\Omega(1)}$.

At a high-level our algorithms for all these learning problems work as follows: By computing the low-degree moments of the hidden parameters, we are able to find a vector space of polynomials that nearly vanish on the unknown parameters. Our structural result allows us to compute a quasi-polynomial sized cover for the set of hidden parameters, which we exploit in our learning algorithms.

***Keywords*-component; machine learning; style; styling;**

I. Introduction

A. Background and Motivation

The main motivation behind this work is the problem of designing efficient learning algorithms for high-dimensional probabilistic models with latent (hidden) variables. This general question has a long history in statistics, starting with the pioneering work of Karl Pearson [Pea94] on learning Gaussian mixtures, that introduced the method of moments in this context. During the past decades, an extensive line of work in theoretical computer science and machine learning has made significant progress on various statistical and computational aspects of this broad question.

In this paper, we focus our attention on high-dimensional latent variable models with a large number k of hidden components[1]. In the settings we study, previously known learning algorithms have running times that scale exponentially with k. Roughly speaking, this exponential dependence is typically due to some form of "brute-force" search, after the high-dimensional problem is reduced down to a k-dimensional one. It should be noted that, in certain regimes, the exponential dependence on k is inherent, due to either information-theoretic (see, e.g., [MV10], [HP15]) or computational (see, e.g., [DKS17]) bottlenecks. For the problems we study here, there is no (known) a priori reason ruling out $\mathrm{poly}(k)$ time algorithms, while current algorithms have an $\exp(k^{\Omega(1)})$ dependence.

Motivated by this huge gap in our understanding, we develop new algorithms for several high-dimensional probabilistic models with running times *quasi-polynomial* in the number k of hidden components. More specifically, we design new algorithms for the following fundamental statistical tasks: density estimation and parameter learning for k-mixtures of spherical Gaussians, PAC learning one-hidden-layer neural networks with k hidden ReLU gates (and other well-behaved activation functions) under the Gaussian distribution, density estimation and parameter estimation for k-mixtures of linear regressions (under Gaussian covariates), and parameter learning for k-mixtures of hyperplanes. See Section I-D for detailed statements of our results and comparison to prior work.

All our learning algorithms are based on a new technique that we develop in this work. The key common ingredient is a new result in algebraic geometry that we believe is of independent interest. In more detail, we establish the following: Let V be any vector space of multivariate degree-d homogeneous polynomials with co-dimension at most k and S be the set of points where all polynomials in V *nearly* vanish. Then the set S has an ϵ-cover, in ℓ_2-norm, of size at most $M = (k/\epsilon)^{O_d(k^{1/d})}$. Importantly, our proof is constructive immediately giving an algorithm to compute such a cover that runs in $\mathrm{poly}(M)$ time.

[1]By this we mean that $k = \omega(1)$, in which case an algorithm with runtime exponential in k is not deemed satisfactory.

2575-8454/20/$31.00 ©2020 IEEE
DOI 10.1109/FOCS46700.2020.00026

With this structural result in hand, all our learning algorithms follow a common recipe: First, given a set of samples from our distribution there is an efficient procedure to approximate the degree-$2d$ moments of the hidden parameters. Then we use our structural result to compute a small ϵ-cover for the set of hidden parameters. Once we have a cover of the parameters, we leverage problem-specific techniques to perform density estimation or parameter estimation.

B. Overview for Our Approach

In this section, we give an overview of our approach with a focus on the problem of learning mixtures of spherical Gaussians. In particular, we explain how our aforementioned structural result (regarding covers of near-zero sets of polynomials) naturally comes into play to find a cover for the set of hidden parameters.

Suppose we have access to i.i.d. samples from an unknown k-mixture of identity covariance Gaussians on $\mathbb{R}^m$, $X = \sum_{i=1}^k w_i N(\mu_i, I)$, where $w_i \geq 0$ are the mixing weights, satisfying $\sum_{i=1}^k w_i = 1$, and $\mu_i \in \mathbb{R}^m$ are the mean vectors. There are two versions of the learning problem: (1) Density estimation, where the goal is to compute a hypothesis distribution H that is ϵ-close to X in total variation distance, and (2) Parameter estimation, where the goal is to approximate the parameters w_i, μ_i within small error ϵ. Our approach yields significantly improved algorithms for both these problems via a common technique. In particular, we develop a method to efficiently find a cover for the set of hidden mean vectors, i.e., a set $\mathcal{C} \subset \mathbb{R}^m$ such that for any μ_i, $i \in [k]$, with w_i not too small, there exists $c \in C$ such that the ℓ_2-distance $\|c - \mu_i\|_2$ is small.

A natural approach to learn a k-mixture of Gaussians is to use the method of moments. This method has two steps: (i) We draw sufficiently many samples to accurately approximate the first d moments of the mixture X. (ii) We use our approximations to the moments to compute an approximation of the distribution or its parameters. Unfortunately, the method of moments faces the following obstacle in our context: There exists two k-mixtures of spherical Gaussians, X and X', that are far from each other, but have their first k moments exactly matching. This means that one cannot compute an approximation to X from the first $d < k$ moments alone.

The above moment-matching statement might suggest that *any* moment-based method cannot lead to learning algorithms with running time $2^{o(k)}$ for our problem. However, looking at the structure of these moment-matching distributions gives us hope. Essentially, these instances are based on a one-dimensional construction that matches k moments, which is then embedded into a higher dimensional space. If X and X' are constructed by having all of their Gaussian components centered on an unknown line L, one might not be able to distinguish X and X' directly by using their low-degree moments, but looking at second moments should suffice to approximately determine the line L. Once this line is determined, it would allow us to reduce down to a one-dimensional problem, which can be efficiently solved by other means. Of course, the task of finding the hidden line L could be made more difficult by adding more components to each of X and X', but it is not clear whether or not this could successfully disguise this critical line.

In order to obtain a truly insurmountable hard instance, we would need to construct a k-mixture X, such that not only do the higher moment tensors of X agree with those of some other k-mixture X' (that is far from X), but in addition the higher moment tensors of X are *rotation-invariant*. Such a (hypothetical) construction would imply that the low-degree moments of X are indistinguishable from any rotation of X, and therefore it would be impossible to locate lower-dimensional sub-structures, like the line L above.

Our approach is motivated by the fact that such a hypothetical hard instance is in fact impossible. In particular, we can write our unknown k-mixture X as a convolution $D * G$, where $G \sim N(0, I)$ is the standard Gaussian, and D is a discrete distribution on $\mathbb{R}^m$ with support size at most k. By de-convolving, we can use the moments of X to compute the moments of D. Now, if $\binom{m+d}{d} > k$, a dimension counting argument implies that there exists a non-trivial degree-d polynomial p that vanishes on the support of D. This means that $\mathbf{E}[p^2(D)]$ is also 0. But if we know the first $2d$ moments of X, we can in principle find such a polynomial p, which would imply that p must be identically zero on the support of D. That is, if we know the first $2d$ moments of X, we can find a polynomial p that vanishes on the support of D, and unless $p(x)$ is a function of $\|x\|_2^2$, this will not be a rotationally invariant condition, implying that the moments of X cannot be rotationally invariant.

The above paragraph naturally leads to an idea for an algorithm. Note that, for any d, the space of degree-d polynomials on $\mathbb{R}^m$ has dimension $N = \binom{m+d}{d}$. By the same dimension counting argument, there exists a subspace V of degree-d polynomials with dimension at least $N - k$ that vanishes on the support of D. On the other hand, given the first $2d$ moments of D, we can identify V as the space of polynomials p so that $\mathbf{E}[p^2(D)] = 0$. (We note that this is indeed a subspace, since the quadratic form $q \to \mathbf{E}[q^2(D)]$ is positive semi-definite). If we know V, we know that all the component means of our mixture must lie on the variety $\mathbb{V}$ defined by the polynomials in V. It is not hard to show that this variety $\mathbb{V}$ will have relatively small dimension. This holds because the space of degree-d polynomials on $\mathbb{V}$ is (degree-d polynomials on $\mathbb{R}^m$)/V,which has dimension at most k. This implies that $\binom{\dim(\mathbb{V})+d}{d} \leq k$, and in particular that $\dim(\mathbb{V}) = O(dk^{1/d})$. This allows us to reduce our problem to one on a variety of small dimension that we can hopefully brute force in time exponential in $k^{1/d}$. Indeed, we are able to show that the variety $\mathbb{V}$ will have a small cover. (Of course, having a variety with small dimension does not

imply the existence of a small cover in general. But our variety has additional properties that our proof exploits.)

The biggest technical obstacle to the approach outlined above is, of course, that we cannot have access to the *exact* moments of X (and thus D), but can only hope to approximate them. However, if we have sufficiently accurate approximations to the moments of D, we can still find a vector space V of degree-d polynomials such that for all $p \in V$ we have that $\mathbf{E}[p(D)^2]$ is *small*. This implies that for any point x in the support of D, with reasonable mass, $p(x)$ must *nearly* vanish for all $p \in V$. At this point, we will need a robust version of the aforementioned structural result, which is the main geometric result of this work (Theorem 1). This result essentially says the following: Given such a V and a unit ball B, if we define S to be the set of all points x in B such that $|p(x)|$ is small for all $p \in V$, then S can be covered by approximately $\exp(O(k^{1/d}))$ many small balls. Moreover, there is an efficient algorithm to compute such a cover. This allows us to compute an explicit set of (not too many) hypotheses means x_i such that each center of a Gaussian in X with reasonable weight is close to some x_i.

Given our cover for the set of possible parameters, we can solve both the density estimation and the parameter estimation problems as follows: For density estimation, we note that X can be approximated as a mixture of the $N(x_i, I)$'s. We can thus draw samples from X and use convex optimization to compute appropriate mixing coefficients (Proposition 14). For parameter estimation, if we assume separation of the components of X, we can use the list of hypotheses means to do clustering and learn approximations of the true means using techniques from [DKS18].

More broadly, our technique can also be applied to a number of other high-dimensional learning problems. The key requirement is that the unknown distribution in question is determined by a set of k vectors $v_i \in \mathbb{R}^m$ and non-negative weights w_i, and that we can efficiently approximate the quantity $\sum_{i=1}^k w_i p(v_i)$, for any low degree polynomial p. Given this primitive, we can use our Theorem 1 to find a subspace V of polynomials that almost vanish on the v_i's, and from there compute a small list of hypotheses so that each relevant v_i must be close to at least one such hypothesis. From this point on, we can use efficient algorithms operating on the final cover and/or problem-specific techniques to complete the learning algorithm.

C. Main Result: Small Covers for Near-Zero Sets of Polynomials

Let V be any vector space of homogeneous degree-d real polynomials on $\mathbb{R}^m$ with co-dimension k. We use $\mathbb{R}_{[d]}[x_1, \ldots, x_m]$ for the vector space of all homogeneous degree-d real polynomials on $\mathbb{R}^m$. Let S be the set of points where all polynomials in V are close to zero. Our main result shows that S has a small cover that can be computed efficiently. Specifically, we show:

Theorem 1 (informal)**.** *Let V be any vector space of homogeneous degree-d real polynomials on $\mathbb{R}^m$ with codimension at most k within $\mathbb{R}_{[d]}[x_1, \ldots, x_m]$. For $\delta, R > 0$, let*

$$S = S(V, R, \delta) = \{x \in \mathbb{R}^m : \|x\|_2 \le R \text{ and } |p(x)| \le \delta \|p\|_{\ell_2} \text{ for all } p \in V\} .$$

Then, for sufficiently small $\delta > 0$, there exists an ϵ-cover of S with size at most $M = (2(R/\epsilon)dk)^{O(d^2 k^{1/d})}$. Moreover, there exists an algorithm to compute such a cover in $\mathrm{poly}(M)$ time.

See Theorems 9 and 11 for more detailed formal statements.

Very Brief Proof Overview: The proof of Theorem 1 is elementary, but quite technically involved. At a very high level, we consider what happens when we fix the first m' coordinates of a point $x \in \mathbb{R}^m$. Plugging in these values will change V from a space of polynomials in m variables to a space of polynomials in $m - m'$ variables. Since the latter space is much smaller, generically we should expect that this restriction of V produces a very large space of such polynomials, implying (by way of an inductive application of our theorem), that there are very few ways to fill in the remaining coordinates and still lie in S. This will hold unless the chosen values satisfy the unusual property that when plugged into polynomials of V they cause many of them to vanish or nearly vanish. We prove that this circumstance is in fact rare by showing that all points for which this holds must lie near a low-dimensional hyperplane. By restricting our functions to this hyperplane, we can again use our theorem inductively to handle these bad points.

Discussion: It is instructive to consider Theorem 1 in the special case where $\delta = 0$. Here S is the intersection of a variety $\mathbb{V}$ with a ball of ℓ_2-radius R, and we are asking the natural question of how many balls are needed to cover the real points of an algebraic variety. For sufficiently nice varieties, we should expect to have a cover of size approximately $O(R/\epsilon)^{\dim(\mathbb{V})}$. The constraint that the generating set V is so large does imply strong bounds on the dimension of $\mathbb{V}$. In particular, the fact that V has codimension k implies that the restriction of the space of degree-d homogenous polynomials to $\mathbb{V}$ has dimension at most k, which in turn implies that $\binom{\dim(\mathbb{V})+d-1}{d} \le k$, and therefore $\dim(\mathbb{V}) = O(dk^{1/d})$. Note that this bound is actually tight in the case that $\mathbb{V}$ is a hyperplane, thus requiring covers of size $(R/\epsilon)^{\Omega(dk^{1/d})}$ (for $d \ll \log(k)$) even in the $\delta = 0$ case.

The above argument allows one to show that the dependence of our cover size upper bound on R/ϵ is approximately best possible, as the dimension should equal the metric dimension which is the limit of the logarithm of the cover size over $\log(1/\epsilon)$. However, in order to prove this for finite values of (R/ϵ), one needs to have information not just about the dimension of $\mathbb{V}$, but also about the geometric complexity of the variety. It is perhaps not surprising that such bounds can be obtained (for example because the

codimension of V should bound the *degree* of the variety $\mathbb{V}$), but it seems technically highly non-trivial to do so. Further technical complications arise when one considers the case of $\delta > 0$, i.e., one needs to consider points that are merely *close* to satisfying the equations in $\mathbb{V}$.

Another instructive example here is the case where $d = 1$. In this case, V is a space of linear functions that all vanish on a hyperplane H with dimension at most k. It is easy to see that only points within distance δ of H will lie in S, thus making it easy to produce a cover of size $O(R/\epsilon)^k$. Given the way that we will use Theorem 1 in our applications, the degree-1 case will end up looking very similar to the dimension reduction techniques already known for many of these problems. These techniques involve computing the second moments of the object in question and noting that the second moment matrix will have small singular values in directions perpendicular to the span of the k hidden parameters. This provides us with an $(m-k)$-dimensional subspace of directions on which none of the (significant) parameters has a large projection, allowing one to find a subspace H that nearly passes through all of them. From this point, one can usually reduce to a k-dimensional problem by restricting to or projecting onto H.

In our setting, instead of computing second moments, we compute degree-$2d$ moments. This allows us to compute not just linear functions that nearly vanish on our points, but many functions of degree up to d. This gives us a much-smaller dimensional variety which our points must lie near. Unfortunately, since this new variety is potentially much more complicated than a subspace, we cannot generally project onto it and reduce to a lower dimensional version of the same problem. However, Theorem 1 will allow us to find a small cover of this variety, which can then be used in a brute force manner to solve many of our problems.

More formally, by computing the first $2d$ moments of our distribution, we can solve some equations to compute the first $2d$ moments of our parameters. This will allow us to approximate the values of $\sum_{i=1}^{k} q(v_i)$ for any degree-$2d$ polynomial q. In particular, we look for degree-d polynomials p for which $\sum_{i=1}^{k} p^2(v_i)$ is small. We note that this will hold if and only if p nearly vanishes on all v_i. However, we are guaranteed that a large space of such polynomials will exist and we can find it by an appropriate singular value decomposition. These p's will provide the subspace V needed by Theorem 1, which in turn will provide us with a small cover $\mathcal{C}$. The elements of $\mathcal{C}$ can be thought of as hypotheses for our parameters, and we are guaranteed that each v_i will be close to at least one of our hypotheses. From this point, we can make use of various algorithms to solve our problem that will run in time polynomial in the cover size.

D. Applications: Learning Latent Variable Models

Our main structural result has the following algorithmic applications.

1) Learning Mixtures of Spherical Gaussians: A k-mixture of spherical Gaussians is a distribution on $\mathbb{R}^m$ with density function $F(x) = \sum_{j=1}^{k} w_j N(\mu_j, I)$, where $\mu_j \in \mathbb{R}^m$ are the unknown mean vectors and $w_j \geq 0$, with $\sum_{j=1}^{k} w_j = 1$, are the mixing weights. We assume that the components have the same known covariance matrix, which we can take for simplicity to be the identity matrix.

We will consider both density estimation and parameter estimation. In density estimation, we want to output a hypothesis distribution with small total variation distance from the target. In parameter estimation, we assume that the component means are sufficiently separated, and the goal is to recover the unknown mixing weights and mean vectors to small error.

Prior Work on Learning Mixtures of Spherical Gaussians: Gaussian mixture models are one of the most extensively studied latent variable models, starting with the pioneering work of Karl Pearson [Pea94]. In this paper, we focus on the important special case where each component is spherical. Here we survey the most relevant prior work on density estimation and parameter estimation for this distribution family.

In density estimation, the goal is to output some hypothesis that is close to the unknown mixture in total variation distance. Density estimation for mixtures of spherical Gaussians in both low and high dimensions has been studied in a series of works [FOS06], [MV10], [CDSS13], [CDSS14], [SOAJ14], [DK14], [BSZ15], [HP15], [ADLS17], [DKK$^+$16], [LS17], [ABH$^+$18]. The sample complexity of this learning task for k-mixtures on $\mathbb{R}^m$, for variation distance error ϵ, is easily seen to be $\mathrm{poly}(mk/\epsilon)$, and a nearly tight bound of $\tilde{\Theta}(mk/\epsilon^2)$ was recently shown [ABH$^+$18]. Unfortunately, all previously known algorithms for this learning problem have running times that scale exponentially with the number of components k. Specifically, [SOAJ14] gave a proper density estimation algorithm that uses $\mathrm{poly}(mk/\epsilon)$ samples and runs in time $\mathrm{poly}(mk/\epsilon) + m^2(k/\epsilon)^{O(k^2)}$.

In parameter estimation, the goal is to output the parameters of the data generating distribution, up to small error. For this problem to be information-theoretically solvable with polynomial sample complexity, some further assumptions are needed. The typical assumption involves some kind of pairwise separation between the component means. The algorithmic problem of parameter estimation for high-dimensional Gaussian mixtures under separation conditions was first studied by Dasgupta [Das99], followed by a long series of works [AK01], [VW02], [AM05], [KSV08], [BV08], [RV17], [HL18], [KS17], [DKS18]. For the simplicity of this discussion, we focus on the case of uniform mix-

tures with identity covariance components. [RV17] showed that, in order for the problem to be information-theoretically solvable with $\mathrm{poly}(m,k)$ samples, the minimum pairwise ℓ_2-mean separation should be $\Theta(\sqrt{\log k})$. Subsequently, three independent works [HL18], [KS17], [DKS18] gave parameter estimation algorithms with sample complexities and running times $\mathrm{poly}(m,k^{\mathrm{polylog}(k)})$ that succeed under the optimal separation of $\Theta(\sqrt{\log k})$.

Finally, a related line of work [HK13], [BCMV14], [ABG+14], [GHK15] that studied parameter estimation in a smoothed-like setting, where (instead of separation conditions) one makes certain condition number assumptions about the parameters. These results are incomparable to ours, as we make no such assumptions.

We are now ready to state our algorithmic contributions for this problem. For the task of density estimation, we prove:

Theorem 2 (Density Estimation for Spherical k-GMMs)**.** *There is an algorithm that on input $\epsilon > 0$, and $\tilde{O}(m^2)\mathrm{poly}(k/\epsilon)+(k/\epsilon)^{O(\log^2 k)}$ samples from an unknown k-mixture of spherical Gaussians F on $\mathbb{R}^m$, it runs in time $\mathrm{poly}(mk/\epsilon)+(k/\epsilon)^{O(\log^2 k)}$ and outputs a hypothesis distribution H such that with high probability $d_{\mathrm{TV}}(H,F)\leq\epsilon$.*

Prior to this work, the fastest known algorithm for this learning problem had running time exponential in k, in particular $\mathrm{poly}(m)(k/\epsilon)^{O(k^2)}$ [SOAJ14]. Interestingly, our density estimation algorithm is not proper. The hypothesis H it outputs is an ℓ-mixture of identity covariance Gaussians, where $\ell \gg k$.

For the task of parameter estimation, we prove:

Theorem 3 (Parameter Estimation for Spherical k-GMMs)**.** *There is an algorithm that on input $\epsilon>0$, $d\in\mathbb{Z}_+$, and $N=\tilde{O}(m^2)\mathrm{poly}(k)+\mathrm{poly}(k/\epsilon)+k^{O(d)}$ samples from a uniform k-mixture $F=(1/k)\sum_{i=1}^k N(\mu_i,I)$ on $\mathbb{R}^m$ with pairwise mean separation $\Delta=\min_{i\neq j}\|\mu_i-\mu_j\|_2\geq C\sqrt{\log k}$, where C is a sufficiently large constant, the algorithm runs in time $\mathrm{poly}(N)+k^{O(d^2k^{1/d})}$, and outputs a list of candidate means $\widetilde{\mu}_i$ such that with high probability we have that $\|\mu_i-\widetilde{\mu}_{\pi(i)}\|\leq\epsilon$, $i\in[k]$, for some permutation $\pi\in\mathbf{S}_k$.*

(See the full version for a more detailed statement handling non-uniform mixtures as well.) Prior to this work, [HL18], [KS17], [DKS18] gave algorithms for this problem with sample complexities and runtimes $\mathrm{poly}(m/\epsilon,k^{\mathrm{polylog}(k)})$. Our algorithm provides a tradeoff between sample complexity and running time (by increasing the parameter d from constant to $\log k$). In particular, for $d=\log k$, the algorithm of Theorem 3 matches the best known (quasi-polynomial in k) sample complexity and runtime. More importantly, by taking d to be a large universal constant, we obtain an algorithm with polynomial sample complexity $\mathrm{poly}(m/\epsilon)k^c$, $c>0$, and sub-exponential time $\mathrm{poly}(m/\epsilon)2^{O_c(k^{1/c})}$. No algorithm with polynomial sample complexity and $2^{o(k)}$ time was previously known under any $\mathrm{polylog}(k)$ separation.

Additional Discussion: In this paragraph, we provide two remarks that are useful to put our algorithmic contributions (Theorems 2 and 3) in context.

[DKS17] gave a Statistical Query (SQ) lower bound of $m^{\Omega(k)}$ on the complexity of density estimation for k-mixtures of Gaussians in $\mathbb{R}^m$. The hard instances constructed in that work are far from spherical. A question posed in [DKS17] was whether 2^{k^c}, for some constant $0<c<1$, or even $k^{\omega(1)}$ SQ lower bounds can be shown for learning k-mixtures of spherical Gaussians. The algorithmic results of this paper were inspired by our unsuccessful efforts to prove such lower bounds. In particular, an SQ lower bound of the form 2^{k^c} is ruled out by Theorem 2. An SQ lower bound of the form $k^{\omega(1)}$ is still possible, in principle. Given our quasi-polynomial upper bound, it is a plausible conjecture that a $\mathrm{poly}(k)$ time algorithm is attainable.

The list-decodable Gaussian mean estimator of [DKS18], with runtime $m^{O(\log(1/\alpha))}$, combined with a known dimension-reduction [VW02] and a post-processing clustering step, gives a $\mathrm{poly}(m/\epsilon,k^{\log k})$ sample and time algorithm for parameter learning of spherical k-GMMs, under the information-theoretically optimal mean separation. Due to an SQ lower bound shown in [DKS18] for list-decodable mean estimation, Theorem 3 cannot be obtained via a reduction to list-decoding.

2) Learning One-hidden-layer ReLU Networks: A one-hidden-layer ReLU network with k hidden units is any function $F:\mathbb{R}^m\to\mathbb{R}$ that can be expressed in the form $F(x)=\sum_{i=1}^k a_i\mathrm{ReLU}(w_i\cdot x)$, for some unit vectors $w_i\in\mathbb{R}^m$ and $a_i\in\mathbb{R}_+$, where $\mathrm{ReLU}(t)=\max\{0,t\}$, $t\in\mathbb{R}$. We will denote by $\mathcal{C}_{m,k}$ the class of all such functions.

The PAC learning problem for $\mathcal{C}_{m,k}$ is the following: The input is a multiset of i.i.d. labeled examples (x,y), where $x\sim N(0,I)$ and $y=F(x)+\xi$, for some $F\in\mathcal{C}_{m,k}$ and $\xi\sim N(0,\sigma^2)$, with ξ independent of x. We will call such an (x,y) a *noisy sample* from F. The goal is to output a hypothesis $H:\mathbb{R}^m\to\mathbb{R}$ that with high probability is close to F in L_2-norm.

Prior Work on Learning One-hidden-layer ReLU Networks: In recent years, there has been an explosion of research on provable algorithms for learning neural networks in various settings, see, e.g., [JSA15], [SJA16], [DFS16], [ZLJ16], [ZSJ+17], [GLM18], [GKLW19], [BJW19], [GKKT17], [MR18], [GK19], [VW19] for some works on the topic. Many of these works focused on parameter learning—the problem of recovering the weight matrix of the data generating neural network. We also note that PAC learning of simple classes of neural networks has been studied in a number of recent works [GKKT17], [MR18], [GK19], [VW19].

The work of [GLM18] studies the parameter learning of positive linear combinations of ReLUs under the Gaussian

distribution in the presence of additive noise. It is shown in [GLM18] that the parameters can be approximately recovered efficiently, under the assumption that the weight matrix is full-rank with bounded condition number. The sample complexity and running time of their algorithm scales polynomially with the condition number. More recently, [BJW19], [GKLW19] obtained efficient parameter learning algorithms for vector-valued depth-2 ReLU networks under the Gaussian distribution. Similarly, the algorithms in these works have sample complexity and running time scaling polynomially with the condition number.

In contrast to parameter estimation, PAC learning one-hidden-layer ReLU networks does not require any assumptions on the structure of the weight matrix. The PAC learning problem for this class is information-theoretically solvable with polynomially many samples. The question is whether a computationally efficient algorithm exists. Until recently, the problem of PAC learning positive linear combinations of ReLUs had remained open, even under Gaussian marginals and for $k = 3$, and had been posed as an open problem by Klivans [Kli17]. Recent work [DKKZ20] gave the first non-trivial PAC algorithm for this problem. The algorithm in [DKKZ20] uses $\mathrm{poly}(mk/\epsilon)$ samples, and has runtime $\mathrm{poly}(mk/\epsilon) + (k/\epsilon)^{O(k^2)}$.

Our main result for this learning problem is the following:

Theorem 4 (PAC Learning $\mathcal{C}_{m,k}$)**.** *There is a PAC learning algorithm for $\mathcal{C}_{m,k}$ with respect to $N(0, I)$ with the following performance guarantee: Given $\epsilon > 0$, and $O(m^2k^2/\epsilon^6) + (k/\epsilon)^{O(\log k)}$ noisy samples from an unknown $F \in \mathcal{C}_{m,k}$, the algorithm runs in time $\mathrm{poly}(mk/\epsilon)+(k/\epsilon)^{O(\log^2 k)}$, and outputs a hypothesis $H : \mathbb{R}^m \to \mathbb{R}$ that with high probability satisfies $\|H - F\|_2^2 \le \epsilon^2(\|F\|_2^2 + \sigma^2)$.*

Interestingly, our PAC learning algorithm is not proper. The hypothesis H it outputs is a positive linear combination of ℓ ReLUs, for some $\ell \gg k$.

Our algorithm establishing Theorem 4 does not make crucial use of the assumption that the activation function is a ReLU. The only properties we require is that our activation function has bounded higher moments and non-vanishing even-degree Fourier coefficients. Even though we do not prove theorems for this more general setting, we note that our algorithmic ideas can be extended to other activation functions satisfying these properties.

3) Learning Mixtures of Linear Regressions: A k-mixture of linear regressions (k-MLR), specified by mixing weights $w_i \ge 0$, where $\sum_{i=1}^k w_i = 1$, and regressors $\beta_i \in \mathbb{R}^m$, $i \in [k]$, is the distribution Z on pairs $(x, y) \in \mathbb{R}^m \times \mathbb{R}$, where $x \sim N(0, I)$ and with probability w_i we have that $y = \beta_i \cdot x + \nu$, where $\nu \sim N(0, \sigma^2)$ is independent of x.

We study both density estimation and parameter learning for k-MLRs. For simplicity of the presentation, we will assume in this section that $\max_i \|\beta_i\|_2 \le 1$ and that the mixing weights are uniform.

Prior Work on Learning Mixtures of Linear Regressions: Mixtures of linear regressions are a natural probabilistic model introduced in [DeV89], [JJ94] and have been extensively studied in machine learning. Prior work on this problem is quite extensive. The reader is referred to Section 1.2 of [CLS19] for a detailed summary of prior work on this problem. Here we focus on the prior work that is most closely related to the results of this paper.

Most prior work on learning MLRs has focused on the parameter estimation problem. A line of work (see, e.g., [ZJD16], [LL18], [KC19] and references therein) has focused on analyzing non-convex methods (including expectation maximization and alternating minimization). These works establish local convergence guarantees: Given a sufficiently accurate solution (warm start), these non-convex methods can efficiently boost this to a solution with arbitrarily high accuracy. The focus of our algorithmic results in this section is to provide such a warm start. We note that the local convergence result of [LL18] applies for the noiseless case, while the more recent result of [KC19] can handle non-trivial regression noise when the weights of the unknown mixture are known.

The prior works most closely related to ours are [LL18], [CLS19]. The work of [LL18] focuses on the noiseless setting ($\sigma = 0$) and provides an algorithm with sample complexity and running time scaling exponentially with k. The main bottleneck of their algorithm lies in a univariate parameter estimation step, which relies on the method of moments and requires $k^{O(k)}$ samples and time. The recent work [CLS19] pointed out that the exponential dependence on k is inherent in this approach: One can construct a pair of k-MLRs whose moment tensors of degree up to $\Omega(k)$ match, but their parameters are far from each other. [CLS19] concludes that "any moment-based estimator" would therefore require runtime $\exp(\Omega(k))$. Our approach also uses moments, but exploits the underlying symmetry to circumvent this obstacle.

The fastest previously known algorithm for the parameter estimation problem of k-MLRs was given in [CLS19]. This work circumvents the aforementioned exponential barrier by considering moments of carefully chosen projections of the Fourier transform. Roughly speaking, [CLS19] gives algorithms whose sample complexity and running time scales with $\exp(\tilde{O}(k^{1/2}))$. In more detail, for the noiseless ($\sigma = 0$) and uniform weights case with separation $\Delta > 0$, the algorithm of [CLS19] has sample complexity and runtime of the form $\mathrm{poly}(mk/\Delta)\,(k\ln(1/\Delta))^{\tilde{O}(k^{1/2})}$. For the noisy case, when $\sigma = O(\epsilon)$ and the weights are uniform, the algorithm of [CLS19] has sample complexity and runtime of the form $\mathrm{poly}(mk/(\epsilon\Delta))\,(k/\epsilon)^{\tilde{O}(k^{1/2}/\Delta^2)}$.

In summary, prior to this work, the best known learning algorithms for k-MLRs had sample complexity and running times scaling exponentially with $k^{1/2}$ [CLS19].

We are now ready to state our results for this problem.

For density estimation, we show:

Theorem 5 (Density Estimation for k-MLR)**.** *There is an algorithm that on input $\epsilon > 0$, and $N = \left(m^2\mathrm{poly}(k) + k^{O(\log k)}\right)\tilde{O}(\log(1/\sigma)) + (k/\epsilon)^{O(\log^2 k)}$ samples from an unknown k-MLR Z on $\mathbb{R}^m \times \mathbb{R}$, it runs in $\mathrm{poly}(N)$ time and outputs a hypothesis distribution H such with high probability $d_{\mathrm{TV}}(H, Z) \leq \epsilon$.*

(See the full version for a detailed statement handling general mixtures.) To the best of our knowledge, this is the first algorithm for density estimation of k-MLRs with running time sub-exponential in k.

For the parameter estimation problem, we provide two algorithmic results – one for the noiseless case (corresponding to $\sigma = 0$) and one for the noisy case (corresponding to $\sigma > 0$). We note that the $\sigma = 0$ case is already quite challenging, and most prior work (with provable guarantees) for the large k regime focuses on this case (see, e.g., [ZJD16], [LL18], [CLS19].

For the noiseless case, we achieve exact recovery (see the full version for a more detailed statement):

Theorem 6 (Parameter Estimation for k-MLR, Noiseless Case)**.** *There is an algorithm that given $N = \left(m^2\mathrm{poly}(k) + k^{O(\log k)}\right)\tilde{O}(\log(k\log(m)/\Delta))$ samples from an unknown k-MLR Z on $\mathbb{R}^m \times \mathbb{R}$ with uniform weights and pairwise separation $\Delta = \min_{i\neq j}\|\beta_i - \beta_j\|_2 > 0$, the algorithm runs in time $\mathrm{poly}(N, k^{\log^2 k})$, and outputs a list of hypothesis vectors $\widetilde{\beta}_i$ such that with high probability we have that $\beta_i = \widetilde{\beta}_{\pi(i)}$, $i \in [k]$, for some permutation $\pi \in \mathbf{S}_k$.*

Our second result can handle additive noise (see the full version for a more detailed statement).

Theorem 7 (Parameter Estimation for k-MLR, Noisy Case)**.** *There is an algorithm that on input $\epsilon > 0$, $N = \left(m^2\mathrm{poly}(k) + k^{O(\log k)}\right)\tilde{O}(\log(k\log(m)/\Delta)) + \tilde{O}(m)\mathrm{poly}(k, \log(1/\epsilon))$ samples from an unknown k-MLR Z with uniform weights and mean separation $\Delta = \min_{i\neq j}\|\mu_i - \mu_j\|_2$ such that Δ/σ at least an appropriate polynomial in $k\log(m)$, the algorithm runs in $\mathrm{poly}(N, k^{\log^2 k})$ time, and outputs a list of hypothesis vectors $\widetilde{\beta}_i$ such that with high probability we have that $\|\beta_i - \widetilde{\beta}_{\pi(i)}\| \leq \epsilon$, $i \in [k]$, for some permutation $\pi \in \mathbf{S}_k$.*

4) Learning Mixtures of Hyperplanes: Our final learning application is for the problem of parameter estimation for mixtures of hyperplanes. A k-mixture of hyperplanes is a distribution on $\mathbb{R}^m$ with density function $F(x) = \sum_{j=1}^k w_j N(0, I - v_j v_j^T)$, where $v_j \in \mathbb{R}^m$ with $\|v_j\|_2 = 1$ and $w_j \geq 0$ with $\sum_{j=1}^k w_j = 1$.

We study parameter estimation for this probabilistic model under Δ pairwise separation for the v_j's. Specifically, we will assume that we know some $\Delta > 0$ such that for all $i \neq j$ and $\sigma_i, \sigma_j \in \{\pm 1\}$, we have that $\|\sigma_i v_i - \sigma_j v_j\|_2 \geq \Delta$. Note that the v_j's are only identifiable up to sign, which motivates this definition.

For simplicity, we will assume uniform weights in this section, i.e., that all the w_i's are $1/k$. The goal of parameter learning in this context is to output a list of unit vectors $\{\tilde{v}_j\}_{j=1}^k$ such that there is a permutation $\pi \in \mathbf{S}_k$ and a list of signs $\sigma_j \in \{\pm 1\}$ for which $v_j = \sigma_j \tilde{v}_{\pi(j)}$ for all $j \in [k]$.

Prior Work on Learning Mixtures of Hyperplanes: Mixtures of hyperplanes is a natural probabilistic model that was recently studied in [CLS19], motivated by its connection to the subspace clustering problem (see, e.g., [PHL04], [Vid11] for overviews). In the subspace clustering problem, the data is assumed to be drawn from a union of linear subspaces, and the algorithmic problem is to identify the hidden subspaces. The mixtures of hyperplanes model can be viewed as a hard instance of subspace clustering, but is also of interest in its own right as it arises in various contexts (see [CLS19] for a detailed discussion).

The fastest previously known algorithm for the parameter estimation problem of k-mixtures of hyperplanes was given in [CLS19]. In more detail, for uniform weights and separation $\Delta > 0$, the algorithm of [CLS19] has sample complexity and runtime of the form $\mathrm{poly}(mk/\Delta)\,(k\ln(1/\Delta))^{\tilde{O}(k^{3/5})}$.

Our main result in this section is the following theorem:

Theorem 8 (Parameter Learning for k-mixtures of Hyperplanes)**.** *There is an algorithm that on input $N = O(k/\Delta)^{O(\log k)} + O(m^2)\mathrm{poly}(k\log(m)/\Delta)$ samples from a uniform k-mixture of hyperplanes on $\mathbb{R}^m$ with pairwise separation $\Delta > 0$, the algorithm runs in time $\mathrm{poly}(N) + m^2\log(\log(m)/\Delta)k^{O(\log^2 k)}$ and with high probability outputs an ϵ-approximation to the unknown parameter vectors.*

E. Organization

The structure of this extended abstract is as follows: In Section II, we sketch the proof of our novel geometric result. Section III describes how our geometric result is used for our learning theory applications. Due to space limitations, the details of our learning algorithms and most proofs have been deferred to the full version of this paper.

II. Main Geometric Result

In Section II-A, we show the existence of small covers for near-zero sets of polynomials. In Section II-B, we point out how to turn our existence proof into an efficient algorithm to compute such a small cover.

A. Existence of Small Covers

Our main geometric result is the following theorem:

Theorem 9. *There exists a universal constant $C > 0$ such that the following holds: Let $d, k, m \in \mathbb{Z}_+$ and V be any vector space of homogeneous degree-d real polynomials in m variables with codimension at most k within $\mathbb{R}_{[d]}[x_1, \ldots, x_m]$. For $\delta, R > 0$, let*

$$S = S(V, R, \delta) = \{x \in \mathbb{R}^m : \|x\|_2 \leq R \text{ and } |p(x)| \leq \delta \|p\|_{\ell_2} \text{ for all } p \in V\}.$$

Then, for any $\epsilon \geq \delta^{\frac{1}{(C+1)d}} (2Rdkm)^{\frac{C}{C+1}}$, there exists an ϵ-cover of S with size at most $(2(R/\epsilon)dkm)^{C^2 d^2 k^{1/d}}$.

Detailed Proof Overview.: The proof of Theorem 9 is elementary though highly technical. Before we give the formal proof, we start by explaining the main ideas here.

Fundamentally, the proof is recursive, and we show that given V, δ, ϵ, and R, we can find an appropriate cover of the corresponding set S by reducing to a number of smaller and similar looking problems. The first step in this reduction involves writing $\mathbb{R}^m$ as $\mathbb{R}^{m'} \times \mathbb{R}^{m-m'}$, for m' a sufficient multiple of $k^{1/d}$. We then cover $B_m(0, R)$ by a number of cylinders of the form $B_{m'}(x, \epsilon') \times B_{m-m'}(0, R)$ (ϵ' a carefully chosen constant on the order of δ). Our basic plan is to show that for most x that there is a small cover of the intersection of S with the associated cylinder, and then to show that the bad x all lie close to a hyperplane of codimension at least $m'/2$.

For each cylinder, we note that for $x' \in B(x, \epsilon')$ and $y \in \mathbb{R}^{m-m'}$ not too large that $p(x', y)$ is close to $p(x, y)$ for all p. This means that in order to cover $B(x, \epsilon') \times \mathbb{R}^{m-m'}$, it suffices to find a cover of just $\{x\} \times B_{m-m'}(0, R)$ with slightly stronger parameters. The set that needs to be covered is the set of points that nearly vanish on every polynomial in V, where V is a set of degree-d polynomials in m variables. We restrict our attention to those polynomials $p \in V$ that when restricted to x in their first m' coordinates leave a degree $d-1$ polynomial in $m - m'$ variables. We note that for any such polynomial p, if substituting x into its first m' coordinates does not decrease its L_2 norm by too much, the resulting polynomial $q(y) := p(x, y)$ must nearly vanish on every point of $S \cap \{x\} \times \mathbb{R}^{m-m'}$. One way of formalizing this is as follows. We let W be the subspace of V consisting of polynomials that are homogeneous degree-1 in the x-coordinates. We define a linear transformation A_x mapping W to degree-$(d-1)$ polynomials in the y-coordinates by evaluation on the x coordinates. We note that all points in $S \cap \{x\} \times \mathbb{R}^{m-m'}$ nearly vanish on all polynomials in U, where U is the span of the eigenvectors of A_x with not-too-small eigenvalues. If the number of such eigenvectors is large, then we can recursively find a small cover of $S \cap \{x\} \times \mathbb{R}^{m-m'}$. In particular, if the number of small eigenvectors is less than $k' = k^{(d-1)/d}$, the recursive bounds should suffice. We call such x good. We will need a different technique for finding a cover of the bad points.

For this, we claim that there is a hyperplane H of codimension at least $m'/2$ so that all of the bad x lie close to H. If we can show this, we can cover all of the bad cylinders by recursively finding a cover of $S \cap H$ (considering only the polynomials in the variables along H). To prove this statement, we proceed by contradiction. If no H can be found, there must be many bad x_i so that x_{i+1} is far from the span of $\{x_1, \ldots, x_i\}$. To each x_i we associate degree-$(d-1)$ polynomials $p_{i1}, \ldots, p_{ik'}$ corresponding to the small eigenvectors of A_{x_i}. We let $q_i(x)$ be the linear function $q_i(x) = x \cdot x_i$, and consider the set U of polynomials $q_i(x) p_{ij}(y)$. It is not hard to see that each polynomial in U has a large component orthogonal to the previous elements, and thus their Gram matrix must have a relatively large determinant. On the other hand, $|U| \gg m'k' \gg k$, so there must be many linear combinations of polynomials in U that lie in W. It is not hard to see that any polynomial in W will have relatively small inner product with any polynomial in U, and this will imply that the Gram matrix of U has relatively small determinant, yielding a contradiction with our previous bound.

We note that Theorem 9 has poor dependence on m. This will not matter for our applications, which all begin by reducing to the case $m \leq k$. However, if one wants a better bound in general, there is a black-box way to remove most of the dependence on m.

Proposition 10. *If $\delta < \epsilon^d/2$, then $f(R, d, \epsilon, \delta, k, m) \leq f(R, d, \epsilon/2, \delta, k, O(k^2 (R/\epsilon)^2)) + 1$.*

B. Algorithmic Version of Theorem 9

The proof of Theorem 9 can be made algorithmic in a straightforward manner, yielding the following:

Theorem 11. *In the context of Theorem 9, given a basis for the vector space V, there is an algorithm to compute an ϵ-cover of S with size at most $M = (2(R/\epsilon)dkm)^{C^2 d^2 k^{1/d}}$ that runs in $\mathrm{poly}(M)$ time.*

III. Overall Strategy for Learning Applications

In Section III-A, we explain how and under what conditions one can use Theorem 11 to obtain an ϵ-cover for the set of parameters in a given learning application. Section III-B presents a template for all our applications.

A. From Covers of Near-Zero Sets of Polynomials to Covers of the Parameters

The overall strategy of our algorithmic applications is as follows. We have an underlying learning problem that is defined by a collection of k vectors $v_i \in \mathbb{R}^m$ and corresponding non-negative weights $w_i \geq 0$, $i \in [k]$. We assume that we have an efficient method for computing the weighted low-degree moments of the v_i's. That is, we assume that we can efficiently obtain a sufficiently good approximation to the tensors $\sum_{i=1}^k w_i v_i^{\otimes 2d}$, or equivalently

that we can approximate $\sum_{i=1}^k w_i p(v_i)$ for any polynomial p of degree $2d$.

Let $p : \mathbb{R}^m \to \mathbb{R}$ be a real degree-d homogeneous polynomial. We consider the quadratic form $Q : \mathbb{R}_{[d]}[x_1, \ldots, x_m] \to \mathbb{R}$ defined by letting $Q(p)$ be our aforementioned approximation to $\sum_{i=1}^k w_i p^2(v_i)$. Note that Q has the following crucial property: If p vanishes on all of the v_i's, then $Q(p)$ *nearly* vanishes.

This property allows us to efficiently compute a subspace V of $\mathbb{R}_{[d]}[x]$, so that for every $p \in V$ we will have that $|p(v_i)|$ is very small for all $i \in [k]$ such that the corresponding weight w_i is not negligibly small. It is not hard to see that V will have small codimension, so using Theorem 11, we can efficiently compute a small cover for the set of possible values for such v_i's.

In particular, we show:

Proposition 12. *Let $m, k \in \mathbb{Z}_+, R > 0$, $v_i \in \mathbb{R}^m$ and $w_i \in \mathbb{R}_+$, for all $i \in [k]$. There is an algorithm that takes as input R, k, m, parameters $\delta, \epsilon > 0$ with $\delta \leq \epsilon^{2d}((\epsilon/R)/(2kmd))^{2Cd}$ for $C > 0$ a sufficiently large constant, and a tensor $T : \mathbb{R}^d \to \mathbb{R}$ such that $\|T - \sum_{i=1}^k w_i v_i^{\otimes 2d}\|_2 \leq \delta$, runs in time $(2(R/\epsilon)kmd)^{O(d^2 k^{1/d})}$, and outputs a set $\mathcal{C} \subset \mathbb{R}^m$ of cardinality at most $(2(R/\epsilon)kmd)^{O(d^2 k^{1/d})}$, satisfying the following property: For any $i \in [k]$ with $w_i \geq ((\epsilon/R)/(2kmd))^{Cd}$ and $\|v_i\|_2 \leq R$, there is a $c \in \mathcal{C}$ with $\|v_i - c\|_2 \leq \epsilon$.*

Proof: The algorithm is described below:

1) Define the quadratic form $Q : \mathbb{R}_{[d]}[x] \to \mathbb{R}$, where $Q(p)$ is defined as follows: We can write $p^2(x) = A_p(x, x, \ldots, x)$, for some uniquely defined rank-$2d$ symmetric tensor A_p. We define $Q(p) = \langle A_p, T \rangle$.
2) Let $V \subseteq \mathbb{R}_{[d]}[x]$ be the subspace spanned by all but the top-k eigenvectors of Q (with respect to our $\|\cdot\|_{\ell_2}$ norm on polynomials).
3) Run the algorithm from Theorem 11 on input $V, \epsilon, k, m, \delta, R$ to obtain the set $\mathcal{C}$.

To show that this algorithm works, we first note that Q is in fact a quadratic form, as the A_p are quadratic in p. In fact, if $p(x) = B_p(x, x, \ldots, x)$, for a symmetric tensor B_p of rank s, then A_p is the symmetrization of $B_p \otimes B_p$. It then follows from the Cauchy-Schwartz inequality that $\|A_p\|_2 \leq \|B_p\|_2^2 = \|p\|_{\ell_2}^2$. Therefore, we have that

$$\begin{aligned} Q(p) &= \langle T, A_p \rangle \\ &= \left\langle \sum_{i=1}^k w_i v_i^{\otimes t}, A_p \right\rangle + \left\langle T - \sum_{i=1}^k w_i v_i^{\otimes 2d}, A_p \right\rangle \\ &= \sum_{i=1}^k w_i p^2(v_i) + O\left(\left\| T - \sum_{i=1}^k w_i v_i^{\otimes 2d} \right\|_2 \|A_p\|_2 \right) \\ &= \sum_{i=1}^k w_i p^2(v_i) + O\left(\delta \|p\|_{\ell_2}^2\right) . \end{aligned}$$

Therefore, $Q(p)$ is indeed a good approximation to the quadratic form $p \to \sum_{i=1}^k w_i p^2(v_i)$, as desired.

We next show that Q has many small eigenvalues. In particular, let U be the space of polynomials $p \in \mathbb{R}_{[d]}(x)$ so that $p(v_i) = 0$ for all $i \in [k]$. Note that U is the kernel of the map $E : \mathbb{R}_{[d]}(x) \to \mathbb{R}^k$ given by $E(p) = (p(v_1), \ldots, p(v_k))$. Therefore, U has co-dimension at most k in $\mathbb{R}_{[d]}[x]$. On the other hand, for $p \in U$, we have that $\sum_{i=1}^k w_i p^2(v_i) = 0$, and therefore $|Q(p)| = O(\delta \|p\|_{\ell_2}^2)$. Thus, the $(k+1)^{st}$ largest eigenvalue of Q is at most $O(\delta \|p\|_{\ell_2}^2)$. In particular, this implies that V is spanned by eigenvalues of at most this size. Therefore, for all $p \in V$, we have that

$$\sum_{i=1}^k w_i p^2(v_i) + O(\delta \|p\|_{\ell_2}^2) = Q(p) = O(\delta \|p\|_{\ell_2}^2) .$$

That is, for $p \in V$, we have that

$$\sum_{i=1}^k w_i p^2(v_i) = O(\delta \|p\|_{\ell_2}^2) .$$

In particular, this means that for all $p \in V$ and all $w_i \geq ((\epsilon/R)/(2kmd))^{Cd}$, we have that

$$|p(v_i)| = O\left(\epsilon^d ((\epsilon/R)/(2kmd))^{Cd/2} \|p\|_{\ell_2} \right) .$$

This implies that v_i satisfies the condition for being in the set S of Theorem 9, and therefore there exists $c \in \mathcal{C}$ so that $\|v_i - c\|_2 \leq \epsilon$. ∎

This cover will prove useful to us, however, the requirement that we learn these m-dimensional tensors for potentially large values of m is suboptimal. We show by a similar technique that we can often reduce the problem to an at most k-dimensional one.

Proposition 13. *Let $v_i, w_i, k, m, T, \delta, R$ be as in Proposition 12 with $d = 1$ and $w > 0$. There is an algorithm that, given this input, computes a subspace U of dimension at most k so that, for all $w_i \geq w$, we have that v_i is within ℓ_2-distance $O((\delta/w)^{1/2})$ of U. Furthermore, this algorithm runs in polynomial time.*

Proof: The algorithm is as follows:

1) Compute Q and V as in Proposition 12.
2) Let U be the set of points $x \in \mathbb{R}^m$ so that $p(x) = 0$ for all polynomials p in V.

To show correctness, it is not hard to see that, since V has co-dimension at most k, U will have dimension at most k. In particular, if $p_1, \ldots, p_{m-k}$ is a basis for V, U is the subspace defined by these $m - k$ linear constraints, and so will have dimension k.

On the other hand, suppose that we have an $i \in [k]$ with $w_i \geq w$. We note that there is a unit vector u_i orthogonal to U so that the ℓ_2-distance from v_i to U is $u_i \cdot v_i$. Let $p(x)$ be the polynomial $p(x) = x \cdot u_i$. Since p vanishes on U, it

must be in V. This means that the ℓ_2-distance from v_i to U is

$$|u_i \cdot v_i| = |p(v_i)| = O((\delta/w)^{1/2}\|p\|_{\ell_2}) = O((\delta/w)^{1/2}) \ ,$$

as desired. ∎

A common application of this technique is where we are learning a high-dimensional distribution D that is given as a mixture $D = \sum_{i=1}^{k} w_i\theta(v_i)$, where θ is some family of distributions parameterized by the vectors v_i. Proposition 12 will allow us to get a large list of hypotheses that will include approximations to all of the large components in this mixture. It will usually be the case therefore, that we can approximate D by another distribution $D' = \sum_{i=1}^{k} w_i'\theta(c_i)$, where the c_i's are in our cover. In particular, if $\|v-c\|_2 \leq \epsilon$ implies that $d_{\mathrm{TV}}(\theta(v),\theta(c)) < \eta$, then by replacing v_i by the closest c_i and letting $w_i = w_i'$, it is not hard to see that $d_{\mathrm{TV}}(D,D') \leq \eta + k((\epsilon/R)/(2kmd))^{Cd}$, as the L_1-distance between $w_i\theta(v_i)$ and $w_i\theta(c_i)$ will always be at most $\min(((\epsilon/R)/(2kmd))^{Cd}, w_i\eta)$.

This shows that D can be approximated by a mixture of the distributions $\theta(c_i)$. It turns out that we can always find such a distribution efficiently:

Proposition 14. *Let $p_1,\ldots,p_n$ be explicit probability distributions and let X be a probability distribution such that, for some $w_1,\ldots,w_n$, we have $d_{\mathrm{TV}}(X,\sum_{i=1}^{n} w_ip_i) \leq \epsilon$. Then there exists a $\mathrm{poly}(n/\epsilon)$-time algorithm that given $p_1,\ldots,p_n$ and $N > n/\epsilon^2$ samples from X, returns a distribution p such that with probability at least $2/3$, we have that $d_{\mathrm{TV}}(X,p) = O(\sqrt{\epsilon\log(n/\epsilon)})$.*

B. Template Approach for Learning Applications

In this section, we describe at a high-level how the preceding theorems are used to make our applications work.

1) Setup: First, we need to define our problem in the context described. In particular, we have access to an object parameterized by k vectors v_i and k non-negative weights w_i.

2) Moment Computation: Critically, we need a way to compute approximations of the moments $\sum_{i=1}^{k} w_iv_i^{\otimes 2d}$. For this, it suffices for every degree-$2d$ monomial $p(x)$ to be able to approximate $\sum_{i=1}^{k} w_ip(v_i)$ to error δ/m^d. This is usually done by finding some polynomial function P of our samples that is an unbiased estimator of $\sum_{i=1}^{k} w_ip(v_i)$ and computing an empirical mean.

3) (Optional) Rough Clustering: One issue with this technique is that our requirements on error are often dependent on the upper bound R we have on the ℓ_2-norm of the v_i's. As having large v_i will usually also make our sample complexity to approximate moments higher as well, it is often important to reduce to the case where R is relatively small. This can often be done by performing some kind of rough clustering of samples to split our problem into components whose v_i all lie in a relatively small ball.

4) (Optional) Dimension Reduction: We will often want to use Proposition 13 to reduce to the case where the underlying problem is k-dimensional. This is because the m-dimensional version of the problem will often incur runtime and sample complexity proportional to m^d.

5) Covering: Next we compute the weighted moments of the v_i's and use Proposition 12 to efficiently find an appropriate cover.

6) From Covers to Learning: Finally, we use this cover to learn. For density estimation, this entails some sort of algorithm with time polynomial in the cover size, analogous to Proposition 14. For parameter estimation, we need to employ additional problem-specific algorithmic ideas.

REFERENCES

[ABG+14] J. Anderson, M. Belkin, N. Goyal, L. Rademacher, and J. R. Voss. The more, the merrier: the blessing of dimensionality for learning large gaussian mixtures. In *Proceedings of The 27th Conference on Learning Theory, COLT 2014*, pages 1135–1164, 2014.

[ABH+18] H. Ashtiani, S. Ben-David, N. J. A. Harvey, C. Liaw, A. Mehrabian, and Y. Plan. Nearly tight sample complexity bounds for learning mixtures of gaussians via sample compression schemes. In *Advances in Neural Information Processing Systems 31: Annual Conference on Neural Information Processing Systems 2018, NeurIPS 2018*, pages 3416–3425, 2018.

[ADLS17] J. Acharya, I. Diakonikolas, J. Li, and L. Schmidt. Sample-optimal density estimation in nearly-linear time. In *Proc. 28th Annual Symposium on Discrete Algorithms (SODA)*, pages 1278–1289, 2017.

[AK01] S. Arora and R. Kannan. Learning mixtures of arbitrary Gaussians. In *Proceedings of the 33rd Symposium on Theory of Computing*, pages 247–257, 2001.

[AM05] D. Achlioptas and F. McSherry. On spectral learning of mixtures of distributions. In *Proceedings of the Eighteenth Annual Conference on Learning Theory (COLT)*, pages 458–469, 2005.

[BCMV14] A. Bhaskara, M. Charikar, A. Moitra, and A. Vijayaraghavan. Smoothed analysis of tensor decompositions. In *Symposium on Theory of Computing, STOC 2014*, pages 594–603, 2014.

[BJW19] A. Bakshi, R. Jayaram, and D. P. Woodruff. Learning two layer rectified neural networks in polynomial time. In *Conference on Learning Theory, COLT 2019*, pages 195–268, 2019.

[BSZ15] A. Bhaskara, A. T. Suresh, and M. Zadimoghaddam. Sparse solutions to nonnegative linear systems and applications. In *Proceedings of the Eighteenth International Conference on Artificial Intelligence and Statistics, AISTATS 2015*, volume 38 of *JMLR Workshop and Conference Proceedings*. JMLR.org, 2015.

[BV08] S. C. Brubaker and S. Vempala. Isotropic PCA and Affine-Invariant Clustering. In *Proc. 49th IEEE Symposium on Foundations of Computer Science*, pages 551–560, 2008.

[CDSS13] S. Chan, I. Diakonikolas, R. Servedio, and X. Sun. Learning mixtures of structured distributions over discrete domains. In *Proc. 24th Annual Symposium on Discrete Algorithms (SODA)*, pages 1380–1394, 2013.

[CDSS14] S. Chan, I. Diakonikolas, R. Servedio, and X. Sun. Efficient density estimation via piecewise polynomial approximation. In *Proc. 46th Annual ACM Symposium on Theory of Computing (STOC)*, pages 604–613, 2014.

[CLS19] S. Chen, J. Li, and Z. Song. Learning mixtures of linear regressions in subexponential time via fourier moments. *CoRR*, abs/1912.07629, 2019.

[Das99] S. Dasgupta. Learning mixtures of Gaussians. In *Proceedings of the 40th Annual Symposium on Foundations of Computer Science*, pages 634–644, 1999.

[DeV89] R. D. DeVeaux. Mixtures of linear regressions. *Computational Statistics & Data Analysis*, 8(3):227–245, November 1989.

[DFS16] A. Daniely, R. Frostig, and Y. Singer. Toward deeper understanding of neural networks: The power of initialization and a dual view on expressivity. In *Advances in Neural Information Processing Systems 29: Annual Conference on Neural Information Processing Systems 2016*, pages 2253–2261, 2016.

[DK14] C. Daskalakis and G. Kamath. Faster and sample near-optimal algorithms for proper learning mixtures of Gaussians. In *Proc. 27th Annual Conference on Learning Theory (COLT)*, pages 1183–1213, 2014.

[DKK+16] I. Diakonikolas, G. Kamath, D. M. Kane, J. Li, A. Moitra, and A. Stewart. Robust estimators in high dimensions without the computational intractability. In *Proc. 57th IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 655–664, 2016.

[DKKZ20] I. Diakonikolas, D. M. Kane, V. Kontonis, and N. Zarifis. Algorithms and SQ lower bounds for PAC learning one-hidden-layer relu networks. In Jacob D. Abernethy and Shivani Agarwal, editors, *Conference on Learning Theory, COLT 2020*, volume 125 of *Proceedings of Machine Learning Research*, pages 1514–1539. PMLR, 2020.

[DKS17] I. Diakonikolas, D. M. Kane, and A. Stewart. Statistical query lower bounds for robust estimation of high-dimensional Gaussians and Gaussian mixtures. In *Proc. 58th IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 73–84, 2017.

[DKS18] I. Diakonikolas, D. M. Kane, and A. Stewart. List-decodable robust mean estimation and learning mixtures of spherical Gaussians. In *Proc. 50th Annual ACM Symposium on Theory of Computing (STOC)*, pages 1047–1060, 2018.

[FOS06] J. Feldman, R. O'Donnell, and R. Servedio. PAC learning mixtures of Gaussians with no separation assumption. In *Proc. 19th Annual Conference on Learning Theory (COLT)*, pages 20–34, 2006.

[GHK15] R. Ge, Q. Huang, and S. M. Kakade. Learning mixtures of gaussians in high dimensions. In *Proceedings of the Forty-Seventh Annual ACM on Symposium on Theory of Computing, STOC 2015*, pages 761–770, 2015.

[GK19] S. Goel and A. R. Klivans. Learning neural networks with two nonlinear layers in polynomial time. In *Conference on Learning Theory, COLT 2019*, pages 1470–1499, 2019.

[GKKT17] S. Goel, V. Kanade, A. R. Klivans, and J. Thaler. Reliably learning the relu in polynomial time. In *Proceedings of the 30th Conference on Learning Theory, COLT 2017*, pages 1004–1042, 2017.

[GKLW19] R. Ge, R. Kuditipudi, Z. Li, and X. Wang. Learning two-layer neural networks with symmetric inputs. In *7th International Conference on Learning Representations, ICLR 2019*, 2019.

[GLM18] R. Ge, J. D. Lee, and T. Ma. Learning one-hidden-layer neural networks with landscape design. In *6th International Conference on Learning Representations, ICLR 2018*, 2018.

[HK13] D. Hsu and S. M. Kakade. Learning mixtures of spherical gaussians: moment methods and spectral decompositions. In *Innovations in Theoretical Computer Science, ITCS '13*, pages 11–20, 2013.

[HL18] S. B. Hopkins and J. Li. Mixture models, robustness, and sum of squares proofs. In *Proc. 50th Annual ACM Symposium on Theory of Computing (STOC)*, pages 1021–1034, 2018.

[HP15] M. Hardt and E. Price. Tight bounds for learning a mixture of two gaussians. In *Proceedings of the Forty-Seventh Annual ACM on Symposium on Theory of Computing, STOC 2015*, pages 753–760, 2015.

[JJ94] M. I. Jordan and R. A. Jacobs. Hierarchical mixtures of experts and the EM algorithm. *Neural Computation*, 6(2):181–214, 1994.

[JSA15] M. Janzamin, H. Sedghi, and A. Anandkumar. Beating the perils of non-convexity: Guaranteed training of neural networks using tensor methods, 2015.

[KC19] J. Kwon and C. Caramanis. EM converges for a mixture of many linear regressions. *CoRR*, abs/1905.12106, 2019.

[Kli17] A. Klivans. Talk at stoc'17 workshop on new challenges in machine learning – robustness and nonconvexity, 2017.

[KS17] P. K. Kothari and D. Steurer. Outlier-robust moment-estimation via sum-of-squares. *CoRR*, abs/1711.11581, 2017.

[KSV08] R. Kannan, H. Salmasian, and S. Vempala. The spectral method for general mixture models. *SIAM J. Comput.*, 38(3):1141–1156, 2008.

[LL18] Y. Li and Y. Liang. Learning mixtures of linear regressions with nearly optimal complexity. In *Conference On Learning Theory, COLT 2018*, volume 75 of *Proceedings of Machine Learning Research*, pages 1125–1144. PMLR, 2018.

[LS17] J. Li and L. Schmidt. Robust and proper learning for mixtures of gaussians via systems of polynomial inequalities. In *Proceedings of the 30th Conference on Learning Theory, COLT 2017*, volume 65 of *Proceedings of Machine Learning Research*, pages 1302–1382. PMLR, 2017.

[MR18] P. Manurangsi and D. Reichman. The computational complexity of training relu(s), 2018.

[MV10] A. Moitra and G. Valiant. Settling the polynomial learnability of mixtures of Gaussians. In *FOCS*, pages 93–102, 2010.

[Pea94] K. Pearson. Contribution to the mathematical theory of evolution. *Phil. Trans. Roy. Soc. A*, 185:71–110, 1894.

[PHL04] L. Parsons, E. Haque, and H. Liu. Subspace clustering for high dimensional data: a review. *SIGKDD Explorations*, 6(1):90–105, 2004.

[RV17] O. Regev and A. Vijayaraghavan. On learning mixtures of well-separated gaussians. In *58th IEEE Annual Symposium on Foundations of Computer Science, FOCS 2017*, pages 85–96, 2017.

[SJA16] H. Sedghi, M. Janzamin, and A. Anandkumar. Provable tensor methods for learning mixtures of generalized linear models. In *Proceedings of the 19th International Conference on Artificial Intelligence and Statistics, AISTATS 2016*, pages 1223–1231, 2016.

[SOAJ14] A. T. Suresh, A. Orlitsky, J. Acharya, and A. Jafarpour. Near-optimal-sample estimators for spherical Gaussian mixtures. In *Proc. 29th Annual Conference on Neural Information Processing Systems (NIPS)*, pages 1395–1403, 2014.

[Vid11] R. Vidal. Subspace clustering. *IEEE Signal Process. Mag.*, 28(2):52–68, 2011.

[VW02] S. Vempala and G. Wang. A spectral algorithm for learning mixtures of distributions. In *Proc. 43rd IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 113–122, 2002.

[VW19] S. Vempala and J. Wilmes. Gradient descent for one-hidden-layer neural networks: Polynomial convergence and SQ lower bounds. In *Conference on Learning Theory, COLT 2019*, pages 3115–3117, 2019. Full version available at https://arxiv.org/abs/1805.02677.

[ZJD16] K. Zhong, P. Jain, and I. S. Dhillon. Mixed linear regression with multiple components. In Daniel D. Lee, Masashi Sugiyama, Ulrike von Luxburg, Isabelle Guyon, and Roman Garnett, editors, *Advances in Neural Information Processing Systems 29: Annual Conference on Neural Information Processing Systems 2016*, pages 2190–2198, 2016.

[ZLJ16] Y. Zhang, J. D. Lee, and M. I. Jordan. L1-regularized neural networks are improperly learnable in polynomial time. In *Proceedings of the 33nd International Conference on Machine Learning, ICML 2016*, pages 993–1001, 2016.

[ZSJ+17] K. Zhong, Z. Song, P. Jain, P. L. Bartlett, and I. S. Dhillon. Recovery guarantees for one-hidden-layer neural networks. In *Proceedings of the 34th International Conference on Machine Learning, ICML 2017*, pages 4140–4149, 2017.