Sample-Optimal Density Estimation in Nearly-Linear Time

Jayadev Acharya*
EECS, MIT
jayadev@csail.mit.edu

Ilias Diakonikolas†
Informatics, U. of Edinburgh
ilias.d@ed.ac.uk

Jerry Li‡
EECS, MIT
jerryzli@csail.mit.edu

Ludwig Schmidt§
EECS, MIT
ludwigs@mit.edu

November 15, 2015

Abstract

We design a new, fast algorithm for agnostically learning univariate probability distributions whose densities are well approximated by piecewise polynomial functions. Let f be the density function of an arbitrary univariate distribution, and suppose that f is OPT close in L_1-distance to an unknown piecewise polynomial function with t interval pieces and degree d. For any $\gamma > 0$, our algorithm draws $n = O_\gamma(t(d+1)/\epsilon^2)$ samples from f, runs in time $\widetilde{O}(n \cdot \mathrm{poly}(d))$, and with probability at least $9/10$ outputs an $O_\gamma(t)$-piecewise degree-d hypothesis h that is $(3+\gamma) \cdot \mathrm{OPT} + \epsilon$ close to f. Our approximation factor almost matches the best known information–theoretic (but computationally inefficient) upper bound of 3.

Our general algorithm yields (nearly) sample-optimal and *nearly-linear time* estimators for a wide range of structured distribution families over both continuous and discrete domains in a unified way. For most of our applications, these are the *first* sample-optimal and nearly-linear time estimators in the literature. As a consequence, our work resolves the sample and computational complexities of a broad class of inference tasks via a single "meta-algorithm". Moreover, we experimentally demonstrate that our algorithm performs very well in practice.

Our algorithm consists of three "levels": (i) At the top level, we employ an iterative greedy algorithm for finding a good partition of the real line into the pieces of a piecewise polynomial. (ii) For each piece, we show that the sub-problem of finding a good polynomial fit on the current interval can be solved efficiently with a separation oracle method. (iii) We reduce the task of finding a separating hyperplane to a combinatorial problem and give an efficient algorithm for this problem. Combining these three procedures gives a density estimation algorithm with the claimed guarantees.

*Supported by a grant from the MIT-Shell Energy Initiative.

†Supported by a Marie Curie CIG, EPSRC grant EP/L021749/1 and a SICSA grant.

‡Supported by NSF grant CCF-1217921, DOE grant DE-SC0008923, NSF CAREER Award CCF-145326, and a NSF Graduate Research Fellowship

§Supported by MADALGO and a grant from the MIT-Shell Energy Initiative.

Contents

1 Introduction

Estimating an unknown probability density function based on observed data is a classical problem in statistics that has been studied since the late nineteenth century, starting with the pioneering work of Karl Pearson [Pea95]. Distribution estimation has become a paradigmatic and fundamental unsupervised learning problem with a rich history and extensive literature (see e.g., [BBBB72, DG85, Sil86, Sco92, DL01]). A number of general methods for estimating distributions have been proposed in the mathematical statistics literature, including histograms, kernels, nearest neighbor estimators, orthogonal series estimators, maximum likelihood, and more. We refer the reader to [Ize91] for a survey of these techniques. During the past few decades, there has been a large body of work on this topic in computer science with a focus on *computational efficiency* [KMR$^+$94, FM99, FOS05, BS10, KMV10, MV10, KSV08, VW02, DDS12a, DDS12b, DDO$^+$13, CDSS14a].

Suppose that we are given a number of samples from an unknown distribution that belongs to (or is well-approximated by) a given family of distributions $\mathcal{C}$, e.g., a mixture of a small number of Gaussians. Our goal is to estimate the unknown distribution in a precise and well-defined way. In this work, we focus on the problem of *density estimation* (non-proper learning), where the objective is to output an approximation of the unknown density without any constraints on its representation. That is, the output hypothesis is not necessarily a member of the family $\mathcal{C}$. In this setting, the "gold standard" is to design learning algorithms that achieve all of the following four goals:

Statistical efficiency. For a given error tolerance, the algorithm should require a small number of samples, ideally matching the information-theoretic minimum.

Computational efficiency. The algorithm should produce a hypothesis with the desired error tolerance as quickly as possible. In the best case, the algorithm has a time complexity that is (nearly-)linear in the number of samples provided as input.

Robustness. The algorithm should offer guarantees even if our samples do not come from a distribution in the family $\mathcal{C}$. In that case, our goal is to be close to the best approximation of the unknown distribution with a distribution in $\mathcal{C}$.

Generality. Ideally, the algorithm achieves the above goals for many families of distributions, i.e., the algorithm is not overly specialized to the properties of a specific family of distributions $\mathcal{C}$.

We give the first algorithm that achieves *all four* of these objectives for density estimation of univariate distributions, i.e., distributions with a density $f : \Omega \to \mathbb{R}_+$ where the sample space Ω is a subset of the real line. While density estimation of univariate distributions has been studied for several decades, both the sample and time complexity were not yet well understood before this work, even for surprisingly simple classes of distributions such as mixtures of Binomials and mixtures of Gaussians. Our main result is a general learning algorithm that can be used to estimate a wide variety of structured distribution families over both continuous and discrete domains. For each such family, our algorithm has a nearly-optimal sample complexity (up to logarithmic factors) and runs in nearly-linear time. Moreover, our algorithm is agnostic and almost matches the best known lower bounds on the approximation ratio.

Our approach is based on learning a piecewise polynomial function that approximates the unknown density. The key idea of using piecewise polynomials for learning is that the *existence* of good piecewise polynomial approximations for a family of distributions $\mathcal{C}$ can be leveraged for the design of efficient learning algorithms for the family $\mathcal{C}$. The approach of using piecewise polynomial approximation has been employed in this context before (e.g., [WW83, Sto94, SHKT97, WN07, CDSS14a]) — our main contributions are new

algorithmic tools for learning with piecewise polynomials. In contrast to prior work, our algorithmic techniques achieve all four goals outlined above. In particular, we show that a natural greedy merging algorithm, combined with an efficiently solvable convex program, yields sample-optimal and nearly-linear time estimators for a wide range of structured distribution families. For most of our applications, these are the *first* sample-optimal and nearly-linear time estimators in the literature. As a consequence, our work resolves the sample and time complexity of a broad class of inference tasks via a single "meta-algorithm". Moreover, we demonstrate experimentally that our algorithm has very good empirical performance.

1.1 Our main result

Preliminaries. We consider univariate probability density functions (pdfs) defined over a known finite interval $I \subseteq \mathbb{R}$. (This assumption is without loss of generality and our results easily apply to densities defined over the entire real line.) We focus on a standard notion of learning an unknown probability distribution from samples [KMR$^+$94], which is a natural analogue of Valiant's well-known PAC model for learning Boolean functions [Val84]. (Our definition is essentially equivalent to the notion of the L_1-minimax rate of convergence in statistics [DL01].) A distribution learning problem is defined by a class $\mathcal{C}$ of probability distributions over a domain Ω. Given an error tolerance $\epsilon > 0$ and sample access to an unknown distribution with density f, the goal of an *agnostic learning algorithm for* $\mathcal{C}$ is to compute a hypothesis h such that, with probability at least $9/10$, it holds that $\|h - f\|_1 \leq C \cdot \mathrm{OPT}_{\mathcal{C}}(f) + \epsilon$, where $\mathrm{OPT}_{\mathcal{C}}(f) := \inf_{q \in \mathcal{C}} \|q - f\|_1$, i.e., $\mathrm{OPT}_{\mathcal{C}}(f)$ is the L_1-distance between the unknown density f and the closest distribution to it in $\mathcal{C}$, and C is a universal constant.

We say that a function f over an interval I is a *t-piecewise degree-d polynomial* if there is a partition of I into t disjoint intervals $I_1, \ldots, I_t$ such that $f(x) = f_j(x)$ for all $x \in I_j$, where each of the $f_1, \ldots, f_t$ is a polynomial of degree at most d. Let $\mathcal{P}_{t,d}(I)$ be the class of all t-piecewise degree-d polynomials over I.

Our Results. Our main algorithmic result is the following:

Theorem 1 (Main)**.** *Let $f : I \to \mathbb{R}_+$ be the density of an unknown distribution over I, where I is either an interval on the real line or the discrete set $[N]$. There is an algorithm with the following performance guarantee: Given parameters $t, d \in \mathbb{Z}_+$, an error tolerance $\epsilon > 0$, and any $\gamma > 0$, the algorithm draws $n = O_\gamma(t(d+1)/\epsilon^2)$ samples from the unknown distribution, runs in time $\widetilde{O}(n \cdot \mathrm{poly}(d+1))$, and with probability at least $9/10$ outputs an $O_\gamma(t)$-piecewise degree-d hypothesis h such that $\|f - h\|_1 \leq (3+\gamma)\mathrm{OPT}_{t,d}(f) + \epsilon$, where $\mathrm{OPT}_{t,d}(f) := \inf_{r \in \mathcal{P}_{t,d}(I)} \|f - r\|_1$ is the error of the best t-piecewise degree-d approximation to f.*

Our algorithm simultaneously achieves all four goals outlined above. (i) First, the algorithm is *general*: for a wide range of structured distribution families over continuous and discrete domains, density estimation can be reduced to learning a piecewise polynomial (see Subsection 1.2). (ii) Second, this reduction is *statistically efficient*. In essentially all cases, the sample complexity $n = O_\gamma(t(d+1)/\epsilon^2)$ of our algorithm matches the information-theoretic optimum for the respective family of distributions up to logarithmic factors. (iii) Moreover, our algorithm makes this reduction *computationally efficient*. For almost all families of distributions, the degree of the polynomial approximation is only $O(\log \frac{1}{\epsilon})$. As a result, our running time $\widetilde{O}(n \cdot \mathrm{poly}(d+1))$ has only logarithmic overhead compared to the information-theoretically optimal sample complexity. In other words, our algorithm runs in nearly-linear time. (iv) Finally, our algorithm is *agnostic* and its approximation ratio $3+\gamma$ almost matches the approximation ratio 3 of a natural information-theoretic algorithm [CDSS14a]. Furthermore, a lower bound of 2 is already known for the special case of histograms ($d = 0$) [CDSS14b], and in this special case our algorithm even achieves an approximation ratio of $2 + \gamma$. In both cases our algorithm only has a mild (inverse linear) dependence on γ in the sample complexity.

As a result, our new algorithmic techniques for learning with piecewise polynomials lead to the first (nearly) sample-optimal and *nearly-linear time* estimators for a wide range of natural and well-studied distribution families. Interestingly, our algorithm shows that there is essentially no (not even a polynomial) trade-off between the computational and statistical efficiency of these problems, i.e., it is possible to achieve *both* the optimal sample and time complexity up to poly-logarithmic factors. In addition, the computational efficiency of our algorithm for *improper* learning has already lead to advances in *proper* learning of Gaussian mixtures [LS15]. It is worth noting that our new algorithm for *univariate* learning also leads to progress for learning *multivariate* mixtures of Gaussians.

Our algorithm has a clean and modular probabilistic analysis that invokes the well-known VC Inequality exactly once. As a result, our algorithm is not plagued by large constants or polylogarithmic factors in the sample complexity and exhibits a very good empirical performance (see Section 8 in the full paper). Moreover, this modularity has already lead to applications of our algorithm in private learning [DHS15].

1.2 Applications of our algorithm

Density estimation. We now explain how to use Theorem 1 in order to agnostically learn structured distribution families. Note that we do not aim to exhaustively cover all possible applications of Theorem 1, but rather to give some selected applications that are indicative of the generality and power of our method.

Given a class $\mathcal{C}$ that we wish to learn, we proceed as follows: (i) Prove that any distribution in $\mathcal{C}$ is $\epsilon/2$-close in L_1-distance to a t-piecewise degree-d polynomial, for appropriate values of t and d. (ii) Instantiate Theorem 1 for these values of t and d to agnostically learn the target distribution up to error $\epsilon/2$. Note that t and d will depend on the desired error ϵ and the underlying class $\mathcal{C}$. We emphasize that there are many combinations of t and d that guarantee an $\epsilon/2$-approximation of $\mathcal{C}$ in Step (i). To minimize the sample complexity of our learning algorithm in Step (ii), we would like to determine the values of t and d that minimize the product $t(d+1)$. This is, of course, an approximation theory problem that only depends on the structure of the family $\mathcal{C}$.

For example, if $\mathcal{C}$ is the family of log-concave distributions, the optimal t-histogram approximation with accuracy ϵ requires $\Theta(1/\epsilon)$ intervals. This leads to an algorithm with sample complexity $\Theta(1/\epsilon^3)$. On the other hand, it can be shown that any log-concave distribution has a piecewise *linear* ϵ-approximation with $\Theta(1/\epsilon^{1/2})$ intervals [CDSS14a, DK15], which yields an algorithm with sample complexity $\Theta(1/\epsilon^{5/2})$. Perhaps surprisingly, this sample bound cannot be improved using higher degrees; one can show an information-theoretic lower bound of $\Omega(1/\epsilon^{5/2})$ for learning log-concave densities [DL01]. Hence, Theorem 1 gives a sample-optimal and nearly-linear time agnostic learning algorithm for this fundamental problem. We remark that piecewise polynomial approximations are "closed" under taking mixtures. As a corollary, Theorem 1 also yields an $O(k/\epsilon^{5/2})$ sample and nearly-linear time algorithm for learning an arbitrary mixture of k log-concave distributions. Again, there exists a matching information-theoretic lower bound of $\Omega(k/\epsilon^{5/2})$.

As a second example, let $\mathcal{C}$ be the class of mixtures of k Gaussians in one dimension. It is not difficult to show that learning such a mixture of Gaussians up to L_1-distance ϵ requires $\Omega(k/\epsilon^2)$ samples. By approximating the corresponding probability density functions with piecewise polynomials of degree $O(\log(1/\epsilon))$, we obtain an agnostic learning algorithm for this class that uses $n = \widetilde{O}(k/\epsilon^2)$ samples and runs in time $\widetilde{O}(n)$. Similar bounds can be obtained for several other natural parametric mixture families.

Note that for a wide range of structured families, the optimal choice of the degree d (i.e., the choice minimizing $t(d+1)$ among all $\epsilon/2$-approximations) is *at most poly-logarithmic in* $1/\epsilon$. For several classes (such as unimodal, monotone hazard rate, and log-concave distributions), the degree d is even a constant. As a consequence, Theorem 1 yields (nearly) sample optimal and nearly-linear time estimators for all these families in a unified way. In particular, we obtain nearly-optimal estimators for natural distributions such as

multi-modal, concave, convex, log-concave, monotone hazard rate, Gaussian, Poisson, Binomial, functions in Besov spaces, mixtures of these distributions, and others. See Table 1 for a summary of these applications.

Class of distributions	Sample complexity	Time complexity	Reference	Optimality
t-histograms	$\widetilde{O}(\frac{t}{\epsilon^2})$	$\widetilde{O}(\frac{t}{\epsilon^2})$	[CDSS14b]	
	$O(\frac{t}{\epsilon^2})$	$O(\frac{t}{\epsilon^2}\log(1/\epsilon))$	Theorem 10	$\mathcal{SO}, \mathcal{TOS}$
t-piecewise degree-d polynomials	$\widetilde{O}(\frac{t\cdot d}{\epsilon^2})$	$\widetilde{O}\left(t^3\cdot(\frac{d^{3.5}}{\epsilon^{3.5}}+\frac{d^{6.5}}{\epsilon^{2.5}})\right)$	[CDSS14a]	
	$O(\frac{t\cdot d}{\epsilon^2})$	$\widetilde{O}(\frac{t\cdot d^{\omega+3}}{\epsilon^2})$	Theorem 1	$\mathcal{NSO}$
k-mixture of log-concave	$\widetilde{O}(\frac{k}{\epsilon^{5/2}})$	$\widetilde{O}(\frac{k^3}{\epsilon^5})$	[CDSS14a]	
	$O(\frac{k}{\epsilon^{5/2}})$	$\widetilde{O}(\frac{k}{\epsilon^{5/2}})$	Theorem 42	$\mathcal{SO}, \mathcal{NTO}$
k-mixture of Gaussians	$\widetilde{O}(\frac{k}{\epsilon^2})$	$\widetilde{O}(\frac{k^3}{\epsilon^{3.5}})$	[CDSS14a]	
	$O(\frac{k\log(1/\epsilon)}{\epsilon^2})$	$\widetilde{O}(\frac{k}{\epsilon^2})$	Theorem 43	$\mathcal{NSO}, \mathcal{NTO}$
Besov space $B^\alpha_q(L_p([0,1]))$	$O_\alpha\left(\frac{\log^2(1/\epsilon)}{\epsilon^{2+1/\alpha}}\right)$	$\widetilde{O}_\alpha\left(\frac{1}{\epsilon^{6+3/\alpha}}\right)$	[WN07]	
	$O_\alpha\left(\frac{1}{\epsilon^{2+1/\alpha}}\right)$	$\widetilde{O}_\alpha\left(\frac{1}{\epsilon^{2+1/\alpha}}\right)$	Theorem 44	$\mathcal{SO}, \mathcal{NTO}$
k-mixture of t-monotone	$\widetilde{O}(\frac{t\cdot k}{\epsilon^{2+1/t}})$	$\widetilde{O}(\frac{k^3}{\epsilon^{3/t}}\cdot(\frac{t^{3.5}}{\epsilon^{3.5}}+\frac{t^{6.5}}{\epsilon^{2.5}}))$	[CDSS14a]	
	$O(\frac{t\cdot k}{\epsilon^{2+1/t}})$	$\widetilde{O}(\frac{k\cdot t^{2+\omega}}{\epsilon^{2+1/t}})$	Theorem 45	$\mathcal{SO}, \mathcal{NTO}$ for $t=1,2$
k-mixture of t-modal	$\widetilde{O}(\frac{t\cdot k\log(N)}{\epsilon^3})$	$\widetilde{O}(\frac{t\cdot k\log(N)}{\epsilon^3})$	[CDSS14b]	
	$O(\frac{t\cdot k\log(N)}{\epsilon^3})$	$O(\frac{t\cdot k\log(N)}{\epsilon^3}\log(1/\epsilon))$	Theorem 46	$\mathcal{SO}, \mathcal{TOS}$
k-mixture of MHR	$\widetilde{O}(\frac{k\log(N/\epsilon)}{\epsilon^3}))$	$\widetilde{O}(\frac{k\log(N/\epsilon)}{\epsilon^3}))$	[CDSS14b]	
	$O(\frac{k\log(N/\epsilon)}{\epsilon^3})$	$O(\frac{k\log(N/\epsilon)}{\epsilon^3}\log(1/\epsilon))$	Theorem 47	$\mathcal{SO}, \mathcal{TOS}$
k-mixture of Binomial, Poisson	$\widetilde{O}(\frac{k}{\epsilon^3})$	$\widetilde{O}(\frac{k}{\epsilon^3})$	[CDSS14b]	
	$O(\frac{k\log(1/\epsilon)}{\epsilon^2})$	$\widetilde{O}(\frac{k}{\epsilon^2})$	Theorem 48	$\mathcal{NSO}, \mathcal{NTO}$

$\mathcal{SO}$: Sample complexity is optimal up to a constant factor.
$\mathcal{NSO}$: Sample complexity is optimal up to a poly-logarithmic factor.
$\mathcal{TOS}$: Time complexity is optimal (up to sorting the samples).
$\mathcal{NTO}$: Time complexity is optimal up to a poly-logarithmic factor.

Table 1: A list of applications to agnostically learning specific families of distributions. For each class, the first row is the best known previous result and the second row is our result. Note that for most of the examples, our algorithm runs in time that is nearly-linear in the information-theoretically optimal sample complexity. The last three classes are over discrete sets, and N denotes the size of the support.

Proper learning. Our non-proper learning algorithm is also useful for proper learning. Theorem 1 has recently been used [LS15] as a crucial component in the fastest known algorithm for properly learning a mixture of univariate Gaussian distributions. Note that non-proper and proper learning for a family $\mathcal{C}$ are equivalent in terms of sample complexity: given any hypothesis, we can perform a brute-force search to

find its closest approximation in the class $\mathcal{C}$. The challenging part is to perform this computation efficiently. Roughly speaking, given a piecewise polynomial hypothesis, [LS15] design an efficient algorithm to find the closest mixture of k Gaussians. Moreover, [LS15] show that our algorithm for *univariate* density estimation is useful for attaining some of the best known bounds on proper learning of *multivariate* Gaussian mixtures.

Private learning. Our algorithm relies only on a single assumption about the input samples which is provided by the VC Inequality (see Theorem 2). As a result, our algorithm is modular and can easily be adapted to other settings that provide a similar guarantee about the input. One example is private learning, in which we want to ensure both a good learning error and that the final hypothesis returned by our algorithm is differentially private, i.e., does not reveal information about individual samples. Recent work in differential privacy shows that it is possible to efficiently construct a differentially private and accurate representation of the empirical distribution [BNSV15]. This "privatized" empirical distribution still satisfies the guarantee required by our approach, and hence our algorithm can easily be adapted to give the best known results for differentially private learning of structured distributions [DHS15].

1.3 Our techniques

We now provide a brief overview of our algorithm. A key definition is the following: for any $k \geq 1$ and an interval $I \subseteq \mathbb{R}$, let the $\mathcal{A}_k$-norm of a function $g : I \to \mathbb{R}$ be $\|g\|_{\mathcal{A}_k} \stackrel{\text{def}}{=} \sup_{I_1,\ldots,I_k} \sum_{i=1}^{k} |g(I_i)|$, where the supremum is over all sets of k disjoint intervals $I_1, \ldots, I_k$ in I, and $g(J) \stackrel{\text{def}}{=} \int_J g(x)\,\mathrm{d}x$ for any measurable set $J \subseteq I$. Our main probabilistic tool is the following well-known version of the VC inequality:

Theorem 2 (VC Inequality [VC71, DL01])**.** *Let $f : I \to \mathbb{R}_+$ be an arbitrary pdf over I, and let $\widehat{f}$ be the empirical pdf obtained after taking n i.i.d. samples from f. Then $\mathbb{E}[\|f - \widehat{f}\|_{\mathcal{A}_k}] \leq O(\sqrt{k/n})$.*

The probabilistic part of our analysis simply conditions on the event $\mathbb{E}[\|f - \widehat{f}\|_{\mathcal{A}_k}] \leq O(\sqrt{k/n})$ for $k = O(t(d+1))$. The remainder of our analysis only uses implications of this $\mathcal{A}_k$-guarantee. In particular, the following two-step procedure is an agnostic learning algorithm for $\mathcal{P}_{t,d}$:

(1) Draw a set of $n = \Theta(t(d+1)/\epsilon^2)$ samples from f.
(2) Output the piecewise-polynomial hypothesis $h \in \mathcal{P}_{t,d}$ that minimizes the quantity $\|h - \widehat{f}\|_{\mathcal{A}_k}$ up to an additive error of $O(\epsilon)$.

While the optimization problem in Step (2) is non-convex, it has sufficient structure so that it can be solved in polynomial time. Intuitively, an algorithm for Step (2) involves two main components:

(2.1) An efficient procedure to find a good set of t intervals.
(2.2) An efficient procedure to agnostically learn a degree-d polynomial in a given interval. The procedure for (2.1) will use the procedure for (2.2) multiple times as a subroutine.

Our algorithm follows this outline. It is worth noting that our techniques for *both* subroutines (2.1) and (2.2) are novel. At a high-level, we implement (2.1) using an *iterative greedy* algorithm. The main idea is to iteratively merge the right pairs of intervals by calling an oracle for procedure (2.2) in every step until the number of intervals becomes $O(t)$. Our iterative algorithm and its subtle analysis are directly inspired by the VC inequality. In each iteration, the algorithm estimates the contribution to an appropriate notion of error when two consecutive intervals are merged, and it then merges pairs of intervals with small error. This procedure ensures that the number of intervals in our partition decreases geometrically.

Our algorithm for procedure (2.2) is based on *convex programming* and runs in time $O(s \cdot \mathrm{poly}(d+1))$, where s is the number of samples in the current interval. To ensure that the overall running time of our

algorithm is nearly-linear for many families of distributions, it is crucial that procedure (2.2) has a time complexity with nearly-optimal dependence on s, i.e., the time complexity must be nearly-linear in the number of samples. At a high level, we achieve this by exploiting the problem structure inherent in the $\mathcal{A}_k$-norm in order to separate the problem dimension d from the problem dimension $1/\epsilon$. We then solve a convex program in dimension d, and access the samples only through a subroutine running in nearly-linear time. More specifically, we consider the convex set of non-negative polynomials with $\mathcal{A}_{d+1}$-distance at most τ from the empirical distribution. While this set is defined through a large number of constraints, we show that it is possible to design a combinatorial *separation oracle* with time complexity that is *nearly-linear* in the number of samples. Combined with tools from convex optimization, such as the Ellipsoid method or Vaidya's algorithm, this gives an efficient algorithm for procedure (2.2).

1.4 Related work

There is a long history of research in statistics on estimating structured families of distributions. For distributions over continuous domains, a very natural type of structure to consider is some sort of "shape constraint" on the probability density function (pdf) defining the distribution. Statistical research in this area started in the 1950's, and the reader is referred to the book [BBBB72] for a summary of the early work. Most of the literature in shape-constrained density estimation has focused on one-dimensional distributions, with a few exceptions during the past decade. Various structural restrictions have been studied over the years, starting from monotonicity, unimodality, convexity, and concavity [Gre56, Bru58, Rao69, Weg70, HP76, Gro85, Bir87a, Bir87b, Fou97, CT04, JW09], and more recently focusing on structural restrictions such as log-concavity and k-monotonicity [BW07, DR09, BRW09, GW09, BW10, KM10, Wal09, DW13, CS13, KS14, BD14, HW15]. The reader is referred to [GJ14] for a recent book on the subject. Mixtures of structured distributions have received much attention in statistics [Lin95, RW84, TSM85, LB99] and, more recently, in theoretical computer science [Das99, DS00, AK01, VW02, FOS05, AM05, MV10].

The most common method used in statistics to address density estimation problems is the Maximum Likelihood Estimator (MLE). While the MLE is very popular and quite natural, we note that it is not agnostic, and it may in general require solving an intractable optimization problem (e.g., for mixture models).

Piecewise polynomials (splines) have been extensively used as tools for inference tasks, including density estimation, see, e.g., [WW83, Sto94, SHKT97, WN07]. We note that these piecewise polynomial estimators are fundamentally different than ours in a number of ways: they rely on a different optimization problem (the MLE and its variants), they use polynomials whose degree is bounded by a small constant (2 or 3), and are largely of a heuristic nature. Indeed, these works do not provide any theoretical analysis of the computational complexity of their estimators. A notable exception is the work of Willett and Nowak [WN07], which gives a polynomial time algorithm to compute (a variant of) the MLE for continuous piecewise polynomial densities. While this method yields estimators that are near-sample optimal (up to polylogarithmic factors) for some families of distributions, the degree of their polynomial in their running time is quite high, making it inapplicable for large datasets. We remark that our approach automatically improves on both the sample complexity and, importantly, the running time for all their applications.

A related line of work in mathematical statistics [KP92, DJKP95, KPT96, DJKP96, DJ98] uses non-linear estimators based on wavelet techniques to learn continuous distributions whose densities satisfy various smoothness constraints, such as Triebel and Besov-type smoothness. We remark that the focus of these works is usually on the statistical efficiency of the proposed estimators.

Comparison to Prior Work. We now provide a detailed comparison between our algorithm and related prior work. Two recent papers [CDSS14a, CDSS14b] study the problem of agnostically learning piecewise

polynomial densities over a continuous domain. [CDSS14a] gives an algorithm that uses $n = \widetilde{O}(t(d+1)/\epsilon^2)$ samples, runs in $\widetilde{O}(\frac{t^2 \cdot d^{3.5}}{\epsilon^{4.5}}(\frac{1}{\epsilon} + d^3))$ time, and outputs a hypothesis with error $14 \cdot \mathrm{OPT}_{t,d} + \epsilon$. [CDSS14b] improves the running time of the latter for the $d = 0$ (i.e., piecewise constant) case to $\widetilde{O}(n)$ and has error of $O(\mathrm{OPT}_{t,d}) + \epsilon$ for an unspecified (large) constant hidden in the $O(\cdot)$.

At a high-level, the starting point of both our work and [CDSS14a, CDSS14b] is the notion of A_k-distance and the related VC-inequality, a classical tool in empirical process theory [DL01, SW09]. As explained in Section 1.3, this machinery directly implies an information-theoretic (but computationally inefficient) upper bound of $O(t(d+1)/\epsilon^2)$ on the sample complexity of agnostic learning. The key challenge is making this bound algorithmic, and this is where our work departs from previous approaches.

Recall that we require two main algorithmic components for solving the learning problem: (2.1), an efficient procedure for finding a good set of $O(t)$ intervals; and (2.2), an efficient algorithm for fitting a degree-d polynomial to the samples in a given interval with respect to the $\mathcal{A}_k$-distance. Our new algorithmic techniques for both components are significantly different from the aforementioned works [CDSS14a, CDSS14b] and lead to an overall learning algorithm that improves in all four goals outlined above (statistical efficiency, computational efficiency, robustness, and generality).

The algorithm in [CDSS14b] (for the $d = 0$ case) has a superficial resemblance to our greedy merging scheme but is in fact considerably different. In particular, their algorithm proceeds by iteratively merging pairs of intervals, but its merging decisions are solely based on the ratio between the means of two adjacent intervals. This criterion is merely local and heavily specialized to piecewise constant functions. As a result, there does not seem to be a way to generalize this idea even to the piecewise linear case ($d = 1$). In contrast, our algorithm crucially exploits *global* information in each iteration to decide which intervals to merge and naturally generalizes to piecewise polynomials of any degree in a clean and modular way.

The algorithm in [CDSS14a] runs a dynamic program (DP) over the sample points in order to find a good set of $O(t)$ intervals that minimizes the $\mathcal{A}_k$-distance between the piecewise polynomial hypothesis and the empirical distribution. This approach requires $\widetilde{O}(t^2/\epsilon^2)$ calls to a polynomial fitting subroutine. Thus, even if we had a polynomial fitting routine that runs in sample-linear time, this DP would lead to a total running time that is suboptimal, both as a function of t and ϵ. In contrast, our novel greedy merging strategy requires only a linear number of calls to the polynomial fitting subroutine, and thus our runtime is nearly optimal in t and ϵ.

To agnostically learn a single degree-d polynomial, [CDSS14a] implicitly approximates the A_k-distance by a different metric (using an adaptive interval partition of the space) and formulates a linear program that optimizes this modified metric. This approximation step has the following consequences: (i) an inherent loss of poly-logarithmic factors in the sample complexity, and (ii) a linear program with $\Omega(d/\epsilon)$ variables and $\Omega(d^2/\epsilon^2 + d^5/\epsilon)$ constraints. Using the fastest known linear programming solver for their instances [LS14] gives a time complexity of $\Omega(d^{3.5}/\epsilon^{3.5} + d^{6.5}/\epsilon^{2.5})$. Hence, the overall running time of their linear program is a polynomial of prohibitively large degree for most applications. In contrast, our polynomial fitting algorithm directly optimizes the $\mathcal{A}_k$-distance over the set of non-negative normalized polynomials. Naively, one would think that this approach leads to an asymptotically slower algorithm as it involves solving an SDP with exponentially many constraints. However, by exploiting the structure of the $\mathcal{A}_k$-distance, we construct a novel combinatorial separation oracle for the problem that runs in nearly *sample-linear* time.

The running time improvements outlined above also lead to large empirical gains. For a medium-scale problem of fitting a piecewise-linear density estimate to $n = 10^6$ samples, a simple back-of-the-envelope calculation shows that the approach of [CDSS14a] would require more than a *week* of computation on a modern CPU, while our algorithm runs in about 0.3 *seconds*. We remark that the back-of-the-envelope calculation is very much in favor of [CDSS14a] (ignoring all logarithmic factors and large constants in their

time complexity), and the running time of our algorithm is based on an actual implementation on a laptop computer from 2010. Hence the gap between the large polynomial time complexity of [CDSS14a] and our nearly-linear running time dictates which algorithm can actually be run on a non-trivial number of samples (see Section 8 in the full paper for further details on the experimental evaluation). Moreover, we note that our dependence on d is better than [CDSS14a], so this gap only widens as d increases. Preliminary experiments indicate that our algorithm still has a reasonable running time (i.e., at most a few minutes) for say $d \leq 6$, at which point [CDSS14a] would likely require more than a year of computation.

As a final point of comparison, we remark that our algorithm (both components) directly optimizes the $\mathcal{A}_k$-distance and *exclusively relies on the $\mathcal{A}_k$-distance* to guide its decision-making. As a result, the probabilistic part of its correctness analysis is extremely simple, using the VC inequality exactly once. This modular character of our approach makes it easily applicable in other settings, e.g., see [DHS15] for an application in differentially private density estimation. In contrast, the previous approaches [CDSS14a, CDSS14b] have a complicated probabilistic analysis involving several additional tools. As a consequence, when $d > 0$, their analysis requires an *anti-concentration* property of the underlying density, namely that it does not assign large probability mass on a single point. Hence, their approaches are restricted to continuous densities and do not extend to discrete distributions beyond using piecewise constant approximation. Our analysis has no such requirements and easily generalizes to discrete piecewise polynomial distributions. As a result, we achieve the first nearly sample-optimal and nearly-linear time algorithms for learning some classical families of discrete distributions (including arbitrary mixtures of Poisson and binomial distributions).

To summarize, our algorithmic approach in this paper provides both quantitative and qualitative improvements over [CDSS14a, CDSS14b]. In addition to its significantly improved *running time*, our algorithm has the following advantages: (1) It achieves the *optimal sample complexity* for the piecewise polynomial learning problem (up to a constant factor). (2) It is more *robust* to noise, achieving an approximation factor arbitrarily close to 3, matching the natural information-theoretic argument. (3) It is more *general* and applies to a wider class of distributions, e.g., families of discrete distributions.

In joint work with Hegde [ADH$^+$15], the authors of the current paper have shown that an analogous approach yields sample optimal and efficient algorithms for agnostically learning discrete distributions with piecewise polynomial functions under the ℓ_2-distance metric. We view this as indication of the robustness of our iterative merging algorithm. We note that learning under the ℓ_2-distance is easier than under the L_1-distance, and that the analysis of [ADH$^+$15] is significantly simpler than the analysis in the current paper. We believe that our iterative greedy approach may find additional applications. In upcoming work, we use our merging technique to improve upon the best-known algorithms for segmented linear regression.

1.5 Paper structure

After some preliminaries in Section 2, we give an outline of our algorithm in Section 3. Sections 4 – 6 contain the various components of our algorithm. Section 7 gives a detailed description of our applications to learning structured distribution families, and we conclude in Section 8 with our experimental evaluation.

2 Preliminaries

We consider univariate probability density functions (pdf's) defined over a known finite interval $I \subseteq \mathbb{R}$. For an interval $J \subseteq I$ and a positive integer k, we will denote by $\mathfrak{I}_J^k$ the family of all sets of k disjoint intervals $I_1, \ldots, I_k$ where each $I_i \subseteq J$. For a measurable function $g : I \to \mathbb{R}$ and a measurable set S, let $g(S) \stackrel{\text{def}}{=} \int_S g$. The L_1-norm of g over a subinterval $J \subseteq I$ is defined as $\|g\|_{1,J} \stackrel{\text{def}}{=} \int_J |g(x)|dx$. More

generally, for any set of disjoint intervals $\mathcal{J} \in \mathfrak{I}_I^k$, we define $\|g\|_{1,\mathcal{J}} = \sum_{J\in\mathcal{J}} \|g\|_{1,J}$.

We now define a norm which induces a corresponding distance metric that will be crucial for this paper:

Definition 3 ($\mathcal{A}_k$-norm)**.** *Let k be a positive integer and let $g : I \to \mathbb{R}$ be measurable. For any subinterval $J \subseteq I$,* the $\mathcal{A}_k$-norm of g on J *is defined as*

$$\|g\|_{\mathcal{A}_k,J} \stackrel{\text{def}}{=} \sup_{\mathcal{I}\in\mathfrak{I}_J^k} \sum_{M\in\mathcal{I}} |g(M)| \ .$$

When $J = I$, we omit the second subscript and simply write $\|g\|_{\mathcal{A}_k}$.

More generally, for any set of disjoint intervals $\mathcal{J} = \{J_1, \ldots, J_\ell\}$ where each $J_i \subseteq I$, we define

$$\|g\|_{\mathcal{A}_k,\mathcal{J}} = \sup_{\mathcal{I}} \sum_{J\in\mathcal{I}} |g(J)|$$

where the supremum is taken over all $\mathcal{I} \in \mathfrak{I}_J^k$ such that for all $J \in \mathcal{I}$ there is a $J_i \in \mathcal{J}$ with $J \subseteq J_i$.

We note that the definition of the $\mathcal{A}_k$-norm in this work is slightly different than that in [DL01, CDSS14a] but is easily seen to be essentially equivalent. The VC inequality (Theorem 2) along with uniform convergence bounds (see, e.g., Theorem 2.2. in [CDSS13] or p. 17 in [DL01]), yields the following:

Corollary 4. *Fix $0 < \epsilon$ and $\delta < 1$. Let $f : I \to \mathbb{R}_+$ be an arbitrary pdf over I, and let $\widehat{f}$ be the empirical pdf obtained after taking $n = \Theta((k + \log 1/\delta)/\epsilon^2)$ i.i.d. samples from f. Then with probability at least $1 - \delta$,*

$$\|f - \widehat{f}\|_{\mathcal{A}_k} \leq \epsilon \ .$$

Definition 5. *Let $g : I \to \mathbb{R}$. We say that g has at most k sign changes if there exists a partition of I into intervals $I_1, \ldots, I_{k+1}$ such that for all $i \in [k + 1]$ either $g(x) \geq 0$ for all $x \in I_i$ or $g(x) \leq 0$ for all $x \in I_i$.*

We will need the following elementary facts about the $\mathcal{A}_k$-norm.

Fact 6. *Let $J \subseteq I$ be an arbitrary interval or a finite set of intervals. Let $g : I \to \mathbb{R}$ be a measurable function.*

(a) If g has at most $k - 1$ sign changes in J, then $\|g\|_{1,J} = \|g\|_{\mathcal{A}_k,J}$.

(b) For all $k \geq 1$, we have $\|g\|_{\mathcal{A}_k,J} \leq \|g\|_{1,J}$.

(c) Let α be a positive integer. Then, $\|g\|_{\mathcal{A}_{\alpha\cdot k},I} \leq \alpha \cdot \|g\|_{\mathcal{A}_k,I}$.

(d) Let $f : I \to \mathbb{R}_+$ be a pdf over I, and let $\mathcal{J}_1, \ldots, \mathcal{J}_\ell$ be finite sets of disjoint subintervals of I, such that for all i, i' and for all $I \in \mathcal{J}_i$ and $I' \in \mathcal{J}_{i'}$, I and I' are disjoint. Then, for all positive integers $m_1, \ldots, m_\ell$, $\sum_{i=1}^{\ell} \|f\|_{\mathcal{A}_{m_i},\mathcal{J}_i} \leq \|f\|_{\mathcal{A}_M}$, where $M = \sum_{i=1}^{\ell} m_i$.

3 Paper outline

In this section, we give a high-level description of our algorithm for learning t-piecewise degree-d polynomials. Our algorithm can be divided into three layers.

Level 1: General merging (Section 4). At the top level, we design an iterative merging algorithm for finding the closest piecewise polynomial approximation to the unknown target density. Our merging algorithm applies more generally to broad classes of piecewise hypotheses. Let $\mathcal{D}$ be a class of hypotheses satisfying the following: (i) The number of intersections between any two hypotheses in $\mathcal{D}$ is bounded. (ii) Given an interval J and an empirical distribution $\widehat{f}$, we can efficiently find the best fit to $\widehat{f}$ from functions in $\mathcal{D}$ with respect to the $\mathcal{A}_k$-distance. (iii) We can efficiently compute the $\mathcal{A}_k$-distance between the empirical distribution and any hypothesis in $\mathcal{D}$. Under these assumptions, our merging algorithm agnostically learns piecewise hypotheses where each piece is in the class $\mathcal{D}$.

In Section 4.1, we start by presenting our merging algorithm for the case of piecewise constant hypotheses. This interesting special case captures many of the ideas of the general case. In Section 4.2, we proceed to present our general merging algorithm that applies all classes of distributions satisfying properties (i)-(iii).

When we adapt the general merging algorithm to a new class of piecewise hypotheses, the main algorithmic challenge is constructing a procedure for property (ii). More formally, we require a procedure with the following guarantee.

Definition 7. *Fix $\eta > 0$. An algorithm $\mathcal{O}_p(\widehat{f}, J, \eta)$ is an* η-approximate $\mathcal{A}_k$-projection oracle *for $\mathcal{D}$ if it takes as input an interval J and $\widehat{f}$, and returns a hypothesis $h \in \mathcal{D}$ such that*

$$\|h - \widehat{f}\|_{\mathcal{A}_k} \leq \inf_{h' \in \mathcal{D}} \|h' - \widehat{f}\|_{\mathcal{A}_k, J} + \eta \, .$$

One of our main contributions is an efficient $\mathcal{A}_k$-projection oracle for the class of degree-d polynomials, which we describe next.

Level 2: $\mathcal{A}_k$-projection for polynomials (Section 5). Our $\mathcal{A}_k$-projection oracle computes the coefficients $c \in \mathbb{R}^{d+1}$ of a degree-d polynomial p_c that approximately minimizes the $\mathcal{A}_k$-distance to the empirical distribution $\widehat{f}$ in the given interval J. Moreover, our oracle ensures that p_c is non-negative on J.

At a high-level, we formulate the $\mathcal{A}_k$-projection as a convex optimization problem. A key insight is that we can construct an efficient, approximate *separation oracle* for the set of polynomials that have an $\mathcal{A}_k$-distance of at most τ to the empirical distribution $\widehat{f}$. Combining this separation oracle with existing convex optimization algorithms allows us to solve the feasibility problem of checking whether we can achieve a given $\mathcal{A}_k$-distance τ. We then convert the feasibility problem to the optimization variant via a binary search over τ.

Note that the set of non-negative polynomials is a spectrahedron (the feasible set of a semidefinite program). After restricting the set of coefficients to non-negative polynomials, we can simplify the definition of the $\mathcal{A}_k$-distance: it suffices to consider sets of intervals with endpoints at the locations of samples. Hence, we can replace the supremum in the definition of the $\mathcal{A}_k$-distance by a maximum over a finite set, which shows that the set of polynomials that are both non-negative and τ-close to $\widehat{f}$ in $\mathcal{A}_k$-distance is also a spectrahedron. This suggests that the $\mathcal{A}_k$-projection problem could be solved by a black-box application of an SDP solver. However, this would lead to a running time that is *exponential* in k because there are more than $\binom{s}{2k}$ possible sets of intervals, where s is the number of sample points in the current interval J.[1]

Instead of using black-box SDP or LP solvers, we construct an algorithm that exploits additional structure in the $\mathcal{A}_k$-projection problem. Most importantly, our algorithm separates the dimension of the desired degree-d polynomial from the number of samples (or equivalently, the error parameter ϵ). This allows us

[1]While the authors of [CDSS14a] introduce an encoding of the $\mathcal{A}_k$-constraint with fewer linear inequalities, their approach increases the number of variables in the optimization problem to depend polynomially on $1/\epsilon$, which leads to an $\Omega(\mathrm{poly}(d+1)/\epsilon^{3.5})$ running time. In contrast, our approach achieves a nearly optimal dependence on ϵ that is $\widetilde{O}(\mathrm{poly}(d+1)/\epsilon^2)$.

to achieve a running time that is *nearly-linear* for a wide range of distributions. Interestingly, we can solve our SDP significantly faster than the LP which has been proposed in [CDSS14a] for the same problem. We achieve this by combining Vaidya's cutting plane method [Vai96] with an efficient separation oracle that leverages the structure of the $\mathcal{A}_k$-distance. This separation oracle is the third level of our algorithm, which we describe next.

Level 3: $\mathcal{A}_k$-separation oracle for polynomials (Section 6). Our separation oracle efficiently tests two properties for a given polynomial p_c with coefficients $c \in \mathbb{R}^{d+1}$: (i) Is the polynomial p_c non-negative on the given interval J? (ii) Is the $\mathcal{A}_k$-distance between p_c and the empirical distribution $\widehat{f}$ at most τ? We implement Test (i) by using known algorithms for finding roots of real polynomials efficiently [Pan01]. Note, however, that root-finding algorithms cannot be exact for degrees larger than 4. Hence, we can only approximately Test (i), which necessarily leads to an *approximate* separation oracle. Nevertheless, we show that such an approximate oracle is still sufficient for solving the convex program outlined above.

At a high level, our algorithm proceeds as follows. We first verify that our current candidate polynomial p_c is "nearly" non-negative at every point in J. Assuming that p_c passes this test, we then focus on the problem of computing the $\mathcal{A}_k$-distance between p_c and $\widehat{f}$. We reduce this problem to a discrete variant by showing that the endpoints of intervals jointly maximizing the $\mathcal{A}_k$-distance are guaranteed to coincide with sample points of the empirical distribution (assuming p_c is nearly non-negative on the current interval). Our discrete variant of this problem is related to a previously studied question in computational biology, namely finding maximum-scoring DNA segment sets [Csu04]. We exploit this connection and give a combinatorial algorithm for this discrete variant that runs in time nearly-linear in the number of sample points in J and the degree d. Once we have found a set of intervals maximizing the $\mathcal{A}_k$-distance, we can convert it to a separating hyperplane for the polynomial coefficients c and the set of non-negative polynomials with $\mathcal{A}_k$-distance at most τ to $\widehat{f}$.

Combining these ingredients yields our general algorithm with the performance guarantees stated in Theorem 1.

4 Iterative merging algorithm

In this section, we describe and analyze our iterative merging algorithm. We start with the case of histograms and then provide the generalization to piecewise polynomials.

4.1 The histogram merging algorithm

A *t-histogram* is a function $h : I \to \mathbb{R}$ that is piecewise constant with at most t interval pieces, i.e., there is a partition of I into intervals $I_1, \ldots, I_{t'}$ with $t' \leq t$ such that h is constant on each I_i. Given sample access to an arbitrary pdf f over I and a positive integer t, we would like to efficiently compute a good t-histogram approximation to f. Namely, if $\mathcal{H}_t = \mathcal{H}_t(I)$ denotes the set of t-histogram probability density functions over I and $\mathrm{OPT}_t = \inf_{g \in \mathcal{H}_t} \|g - f\|_1$, our goal is to output an $O(t)$-histogram $h : I \to \mathbb{R}$ that satisfies $\|h - f\|_1 \leq C \cdot \mathrm{OPT}_t + O(\epsilon)$ with high probability over the samples, where C is a universal constant.

The following notion of flattening a function over an interval will be crucial for our algorithm:

Definition 8. *For a function $g : I \to \mathbb{R}$ and an interval $J = [u, v] \subseteq I$, we define the* flattening of g over J, *denoted $\overline{g}_J$, to be the constant function defined on J as*

$$\overline{g}_J(x) \stackrel{\mathrm{def}}{=} \frac{g(J)}{v - u} \quad \textit{for all } x \in J.$$

For a set $\mathcal{I}$ of disjoint intervals in I, we define the flattening of g over $\mathcal{I}$ *to be the function $\overline{g}_{\mathcal{I}}$ on $\cup_{J\in\mathcal{I}} J$ which for each $J \in \mathcal{I}$ satisfies $\overline{g}_{\mathcal{I}}(x) = \overline{g}_J(x)$ for all $x \in J$.*

We start by providing an intuitive explanation of our algorithm followed by a proof of correctness. The algorithm draws $n = \Theta((t+\log 1/\delta)/\epsilon^2)$ samples $x_1 \le x_2 \le \ldots \le x_n$ from f. We start with the following partition of $I = [a, b]$:

$$\mathcal{I}_0 = \{[a, x_1), [x_1, x_1], (x_1, x_2), [x_2, x_2], \ldots, (x_{n-1}, x_n), [x_n, x_n], (x_n, b]\}. \tag{1}$$

This is the partition where each interval is either a single sample point or the interval between two consecutive samples. Starting from this partition, our algorithm greedily merges pairs of consecutive intervals in a sequence of iterations. When deciding which interval pairs to merge, the following notion of approximation error will be crucial:

Definition 9. *For a function $g : I \to \mathbb{R}$ and an interval $J \subseteq I$, define $e(g, J) = \|g - \overline{g}_J\|_{\mathcal{A}_1,J}$. We call this quantity the $\mathcal{A}_1$-*error of g on J.

In the j-th iteration, given the current interval partition $\mathcal{I}_j$, we greedily merge pairs of consecutive intervals to form the new partition $\mathcal{I}_{j+1}$. Let s_j be the number of intervals in $\mathcal{I}_j$. In particular, given $\mathcal{I}_j = \{I_{1,j}, \ldots, I_{s_j,j}\}$, we consider the intervals

$$I'_{\ell,j+1} = I_{2\ell-1,j} \cup I_{2\ell,j}$$

for all $1 \le \ell \le s_j/2$.[2] We first iterate through $1 \le \ell \le s_j/2$ and calculate the quantities

$$e_{\ell,j} = e(\widehat{f}, I'_{\ell,j+1}) \ ,$$

i.e., the $\mathcal{A}_1$-errors of the empirical distribution on the candidate intervals.

To construct $\mathcal{I}_{j+1}$, the algorithm keeps track of the largest $O(t)$ errors $e_{\ell,j}$. For each ℓ with $e_{\ell,j}$ being one of the $O(t)$ largest errors, we do not merge the corresponding intervals $I_{2\ell-1,j}$ and $I_{2\ell,j}$. That is, we include $I_{2\ell-1,j}$ and $I_{2\ell,j}$ in the new partition $\mathcal{I}_{j+1}$. Otherwise, we include their union $I'_{\ell,j+1}$ in $\mathcal{I}_{j+1}$. We perform this procedure $O(\log \frac{1}{\epsilon})$ times and arrive at some final partition $\mathcal{I}$. Our output hypothesis is the flattening of $\widehat{f}$ with respect to $\mathcal{I}$.

For a formal description of our algorithm, see the pseudocode given in Algorithm 1 below. In addition to the parameter t, the algorithm has a parameter $\alpha \ge 1$ that controls the trade-off between the approximation ratio C achieved by the algorithm and the number of pieces in the output histogram.

The following theorem characterizes the performance of Algorithm 1, establishing the special case of Theorem 1 corresponding to $d = 0$.

Theorem 10. *Algorithm* ConstructHistogram$(f, t, \alpha, \epsilon, \delta)$ *draws $n = O((\alpha t + \log(1/\delta))/\epsilon^2)$ samples from f, runs in time $O(n\,(\log(1/\epsilon) + \log\log(1/\delta)))$, and outputs a hypothesis h and a corresponding partition $\mathcal{I}$ of size $|\mathcal{I}| \le 2\alpha \cdot t$ such that with probability at least $1 - \delta$ we have*

$$\|h - f\|_1 \le 2 \cdot \mathrm{OPT}_t + \frac{4 \cdot \mathrm{OPT}_t + 4\epsilon}{\alpha - 1} + \epsilon \ . \tag{2}$$

[2] We assume s_j is even for simplicity.

Algorithm 1 Approximating with histograms by merging.

function CONSTRUCTHISTOGRAM(f, t, α, ϵ, δ)
  Draw $n = \Theta((\alpha t + \log 1/\delta)/\epsilon^2)$ samples $x_1 \le x_2 \le \ldots \le x_n$.
  Form the empirical distribution $\widehat{f}$ from these samples.
  Let $\mathcal{I}_0 \leftarrow \{[a, x_1), [x_1, x_1], (x_1, x_2), \ldots, (x_{n-1}, x_n), [x_n, x_n], (x_n, b]\}$ be the initial partition.
  $j \leftarrow 0$
  while $|\mathcal{I}_j| > 2\alpha \cdot t$ **do**
    Let $\mathcal{I}_j = \{I_{1,j}, I_{2,j}, \ldots, I_{s_j-1,j}, I_{s_j,j}\}$
    for $\ell \in \{1, 2, \ldots, \frac{s_j}{2}\}$ **do**
      $I'_{\ell,j+1} \leftarrow I_{2\ell-1,j} \cup I_{2\ell,j}$
      $e_{\ell,j} \leftarrow e(\widehat{f}, I'_{\ell,j+1})$
    end for
    Let L be the set of $\ell \in \{1, 2, \ldots, \frac{s_j}{2}\}$ with the αt largest errors $e_{\ell,j}$.
    Let M be the complement of L.
    $\mathcal{I}_{j+1} \leftarrow \bigcup_{\ell \in L} \{I_{2\ell-1,j}, I_{2\ell,j}\}$
    $\mathcal{I}_{j+1} \leftarrow \mathcal{I}_{j+1} \cup \{I'_{\ell,j+1} \mid \ell \in M\}$
    $j \leftarrow j + 1$
  end while
  return $\mathcal{I} = \mathcal{I}_j$ and the flattening $\overline{\widehat{f}_{\mathcal{I}}}$
end function

Proof. We start by analyzing the running time. To this end, we show that the number of intervals decreases exponentially with the number of iterations. In the j-th iteration, we merge all but αt intervals. Therefore,

$$s_{j+1} = \alpha t + \frac{s_j - \alpha t}{2} = \frac{3s_j}{4} + \frac{2\alpha t - s_j}{4}.$$

Note that the algorithm enters the while loop when $s_j > 2\alpha t$, implying that

$$s_{j+1} < \frac{3s_j}{4}.$$

By construction, the number of intervals is at least αt when the algorithm exits the while loop. Therefore, the number of iterations of the while loop is at most

$$O\left(\log \frac{n}{\alpha t}\right) = O\left(\log(1/\epsilon) + \log\log(1/\delta)\right),$$

which follows by substituting the value of n from the statement of the theorem. We now show that each iteration takes time $O(n)$. Without loss of generality, assume that we compute the $\mathcal{A}_1$-distance only over intervals ending at a data sample. For an interval $J = [c, d]$ containing m sample points, $x_1, \ldots, x_m$, let $C_j = \frac{(x_j - x_1)}{jn} - \frac{(d-c)}{n}$. The $\mathcal{A}_1$-error of $\widehat{f}$ on J is given by $\max C_j - \min C_j$ and can therefore be computed in time proportional to the number of sample points in the interval. Therefore, the total time of the algorithm is $O(n(\log(1/\epsilon) + \log\log(1/\delta)))$, as claimed.

We now proceed to bound the learning error. Let $\mathcal{I} = \{I_1, \ldots, I_{t'}\}$ be the partition of I returned by CONSTRUCTHISTOGRAM. The desired bound on $|\mathcal{I}|$ follows immediately because the algorithm terminates only when $|\mathcal{I}| \le 2\alpha t$. The rest of the proof is dedicated to Equation (2).

Fix $h^* \in \mathcal{H}_t$ such that $\|h^* - f\|_1 = \mathrm{OPT}_t$. Let $\mathcal{I}^* = \{I_1^*, \ldots, I_t^*\}$ be the partition induced by the discontinuities of h^*. Call a point at a boundary of any I_j^* a *jump* of h^*. For any interval $J \subseteq I$, we define $\Gamma(J)$ to be the number of jumps of h^* in the interior of J. Since we draw $n = \Omega((\alpha t + \log 1/\delta)/\epsilon^2)$ samples, Corollary 4 implies that with probability at least $1 - \delta$, we have

$$\|\widehat{f} - f\|_{\mathcal{A}_{(2\alpha+1)t}} \leq \epsilon \ .$$

We condition on this event throughout the analysis.

We split the total error into three terms based on the final partition $\mathcal{I}$:

Case 1: Let $\mathcal{F}$ be the set of intervals in $\mathcal{I}$ with zero jumps in h^*, i.e., $\mathcal{F} = \{J \in \mathcal{I} \,|\, \Gamma(J) = 0\}$.

Case 2a: Let $\mathcal{J}_0$ be the set of intervals in $\mathcal{I}$ that were created in the initial partitioning step of the algorithm and which contain a jump of h^*, i.e., $\mathcal{J}_0 = \{J \in \mathcal{I} \mid \Gamma(J) > 0 \text{ and } J \in \mathcal{I}_0\}$.

Case 2b: Let $\mathcal{J}_1$ be the set of intervals in $\mathcal{I}$ that contain at least one jump and were created by merging two other intervals, i.e., $\mathcal{J}_1 = \{J \in \mathcal{I} \mid \Gamma(J) > 0 \text{ and } J \notin \mathcal{I}_0\}$.

Notice that $\mathcal{F}$, $\mathcal{J}_0$, and $\mathcal{J}_1$ form a partition of $\mathcal{I}$, and thus

$$\|h - f\|_1 = \|h - f\|_{1,\mathcal{F}} + \|h - f\|_{1,\mathcal{J}_0} + \|h - f\|_{1,\mathcal{J}_1} \ .$$

We will bound these three terms separately. In particular, we will show:

$$\|h - f\|_{1,\mathcal{F}} \ \leq \ 2 \cdot \|f - h^*\|_{1,\mathcal{F}} + \|\widehat{f} - f\|_{\mathcal{A}_{|\mathcal{F}|},\mathcal{F}} \ , \tag{3}$$

$$\|h - f\|_{1,\mathcal{J}_0} \ \leq \ \|\widehat{f} - f\|_{\mathcal{A}_{|\mathcal{J}_0|},\mathcal{J}_0} \ , \tag{4}$$

$$\|h - f\|_{1,\mathcal{J}_1} \ \leq \ \frac{4 \cdot \mathrm{OPT}_t + 4\epsilon}{\alpha - 1} + 2 \cdot \|f - h^*\|_{1,\mathcal{J}_1} + \|\widehat{f} - f\|_{\mathcal{A}_{|\mathcal{J}_1|+t},\mathcal{J}_1} \ . \tag{5}$$

Using these results along with the fact that $\|f - h^*\|_{1,\mathcal{F}} + \|f - h^*\|_{1,\mathcal{J}_1} \leq \mathrm{OPT}_t$, we have

$$\begin{aligned}
\|h - f\|_1 &\leq 2 \cdot \mathrm{OPT}_t + \frac{4 \cdot \mathrm{OPT}_t + 4\epsilon}{\alpha - 1} + \|\widehat{f} - f\|_{\mathcal{A}_{|\mathcal{F}|},\mathcal{F}} + \|\widehat{f} - f\|_{\mathcal{A}_{|\mathcal{J}_0|},\mathcal{J}_0} + \|\widehat{f} - f\|_{\mathcal{A}_{|\mathcal{J}_1|+t},\mathcal{J}_1} \\
&\overset{(a)}{\leq} 2 \cdot \mathrm{OPT}_t + \frac{4 \cdot \mathrm{OPT}_t + 4\epsilon}{\alpha - 1} + \|\widehat{f} - f\|_{\mathcal{A}_{(2\alpha+1)t}} \\
&\overset{(b)}{\leq} 2 \cdot \mathrm{OPT}_t + \frac{4 \cdot \mathrm{OPT}_t + 4\epsilon}{\alpha - 1} + \epsilon \ ,
\end{aligned}$$

where inequality (a) follows from Fact 6(d) and inequality (b) follows from the VC inequality. Thus, it suffices to prove Equations (3)–(5).

Case 1. We first consider the interval $\mathcal{F}$. By the triangle inequality, we have

$$\|h - f\|_{1,\mathcal{F}} \leq \|f - h^*\|_{1,\mathcal{F}} + \|h - h^*\|_{1,\mathcal{F}} \ .$$

Thus to show (3), it suffices to show that

$$\|h - h^*\|_{1,\mathcal{F}} \leq \|f - h^*\|_{1,\mathcal{F}} + \|\widehat{f} - f\|_{\mathcal{A}_{|\mathcal{F}|},\mathcal{F}} \ . \tag{6}$$

We prove a slightly more general version of (6) that holds over all finite sets of intervals not containing any jump of h^*. We will use this general version also later in our proof.

Lemma 11. *Let $\mathcal{J} \in \mathfrak{I}_I^\ell$ so that $\Gamma(J) = 0$ for all $J \in \mathcal{J}$. Let $\overline{h} = \overline{\widehat{f}_{\mathcal{J}}}$ denote the flattening of $\widehat{f}$ on $\mathcal{J}$. Then*

$$\|\overline{h} - h^*\|_{1,\mathcal{J}} \le \|f - h^*\|_{1,\mathcal{J}} + \|\widehat{f} - f\|_{\mathcal{A}_\ell,\mathcal{J}} \ .$$

Note that this is indeed a generalization of (6) since for any point x in any interval of $\mathcal{F}$, we have $h(x) = \overline{\widehat{f}_{\mathcal{F}}}(x)$.

Proof of Lemma 11. In any interval $J \in \mathcal{J}$ with $\Gamma(J) = 0$, we have

$$\|\overline{h} - h^*\|_{1,J} \overset{(a)}{=} |\overline{h}(J) - h^*(J)| \overset{(b)}{=} |\widehat{f}(J) - h^*(J)|,$$

where (a) follows from the fact that $\overline{h}$ and h^* are constant in J, and (b) follows from the definition of $\overline{h}$. Thus, we get

$$\begin{aligned}
\|\overline{h} - h^*\|_{1,\mathcal{J}} &= \sum_{J\in\mathcal{J}} \|\overline{h} - h^*\|_{1,J} \\
&= \sum_{J\in\mathcal{J}} |\widehat{f}(J) - h^*(J)| \\
&\overset{(c)}{\le} \sum_{J\in\mathcal{J}} |\widehat{f}(J) - f(J)| + \sum_{J\in\mathcal{J}} |f(J) - h^*(J)| \\
&\overset{(d)}{\le} \|\widehat{f} - f\|_{\mathcal{A}_{|\mathcal{J}|},\mathcal{J}} + \|f - h^*\|_{1,\mathcal{J}}
\end{aligned}$$

where (c) uses the triangle inequality, and (d) follows from the definition of $\mathcal{A}_k$-distance. □

Case 2a. Next, we analyze the error for the intervals in $\mathcal{J}_0$. The set $\mathcal{I}_0$ contains only singletons and intervals with no sample points. By definition, only the intervals in $\mathcal{I}_0$ that contain no samples may contain a jump of h^*. The singleton intervals containing the sample points do not include jumps and are hence covered by Case 1. Since the intervals in $\mathcal{J}_0$ do not contain any samples, our algorithm assigns

$$h(J) = \widehat{f}(J) = 0$$

for any $J \in \mathcal{J}_0$. Hence,

$$\|h - f\|_{1,\mathcal{J}_0} = \|f\|_{1,\mathcal{J}_0} \ .$$

We thus have the following sequence of (in)equalities:

$$\begin{aligned}
\|h - f\|_{1,\mathcal{J}_0} &= \|f\|_{1,\mathcal{J}_0} \\
&= \sum_{J\in\mathcal{J}_0} |f(J)| \\
&= \sum_{J\in\mathcal{J}_0} |f(J) - \widehat{f}(J)| \\
&\le \|f - \widehat{f}\|_{\mathcal{A}_{|\mathcal{J}_0|},\mathcal{J}_0} \ ,
\end{aligned}$$

where the last step uses the definition of the $\mathcal{A}_k$-norm.

Case 2b. Finally, we bound the error for intervals in $\mathcal{J}_1$, i.e., intervals that were created by merging in some iteration of our algorithm and also contain jumps. As before, our first step is the following triangle inequality:

$$\|h - f\|_{1,\mathcal{J}_1} \leq \|h - h^*\|_{1,\mathcal{J}_1} + \|h^* - f\|_{1,\mathcal{J}_1} \ .$$

Consider an interval $J \in \mathcal{J}_1$. Since h is constant in J and h^* has $\Gamma(J)$ jumps in J, $h - h^*$ has at most $\Gamma(J)$ sign changes in J. Therefore,

$$\begin{aligned}
\|h - h^*\|_{1,J} &\overset{(a)}{=} \|h - h^*\|_{\mathcal{A}_{\Gamma(J)+1},J} \\
&\overset{(b)}{\leq} \|h - \widehat{f}\|_{\mathcal{A}_{\Gamma(J)+1},J} + \|\widehat{f} - f\|_{\mathcal{A}_{\Gamma(J)+1},J} + \|f - h^*\|_{\mathcal{A}_{\Gamma(J)+1},J} \\
&\overset{(c)}{\leq} (\Gamma(J)+1)\|h - \widehat{f}\|_{\mathcal{A}_1,J} + \|\widehat{f} - f\|_{\mathcal{A}_{\Gamma(J)+1},J} + \|f - h^*\|_{1,J} \ , \qquad (7)
\end{aligned}$$

where equality (a) follows from Fact 6(a), inequality (b) is the triangle inequality, and inequality (c) uses Fact 6(c). Finally, we bound the $\mathcal{A}_1$-distance in the first term above.

Lemma 12. *For any $J \in \mathcal{J}_1$, we have*

$$\|h - \widehat{f}\|_{\mathcal{A}_1,J} \leq \frac{2\mathrm{OPT}_t + 2\epsilon}{(\alpha-1)t} \ . \qquad (8)$$

Before proving the lemma, we show how to use it to complete Case 2b. Since h is the flattening of $\widehat{f}$ over J, we have that $\|h - \widehat{f}\|_{\mathcal{A}_1,J} = e(\widehat{f}, J)$. Applying (7) gives:

$$\begin{aligned}
\|h - h^*\|_{1,\mathcal{J}_1} &= \sum_{J\in\mathcal{J}_1} \|h - h^*\|_{1,J} \\
&\leq \sum_{J\in\mathcal{J}_1} \left((\Gamma(J)+1)\|h - \widehat{f}\|_{\mathcal{A}_1,J} + \|\widehat{f} - f\|_{\mathcal{A}_{\Gamma(J)+1},J} + \|f - h^*\|_{1,J}\right) \\
&\leq \frac{2\cdot\mathrm{OPT}_t + 2\epsilon}{(\alpha-1)t} \cdot \left(\sum_{J\in\mathcal{J}_1} (\Gamma(J)+1)\right) + \sum_{J\in\mathcal{J}_1} \|\widehat{f} - f\|_{\mathcal{A}_{\Gamma(J)+1},J} + \|f - h^*\|_{1,\mathcal{J}_1} \\
&\overset{(a)}{\leq} \frac{4\cdot\mathrm{OPT}_t + 4\epsilon}{(\alpha-1)} + \|\widehat{f} - f\|_{\mathcal{A}_{t+|\mathcal{J}_1|},\mathcal{J}_1} + \|f - h^*\|_{1,\mathcal{J}_1}
\end{aligned}$$

where inequality (a) uses the fact that $\Gamma(J) \geq 1$ for these intervals and hence

$$\sum_{J\in\mathcal{J}_1} (\Gamma(J)+1) \leq 2\sum_{J\in\mathcal{J}_1} \Gamma(J) \leq 2t \ .$$

We now complete the final step by proving Lemma 12.

Proof of Lemma 12. Recall that in each iteration of our algorithm, we merge all pairs of intervals except those with the αt largest errors. Therefore, if two intervals were merged, there were at least αt other pairs of intervals with larger error. We will use this fact to bound the error on the intervals in $\mathcal{J}_1$.

Consider any interval $J \in \mathcal{J}_1$, and suppose it was created in the jth iteration of the while loop of our algorithm, i.e., $J = I'_{i,j+1} = I_{2i-1,j} \cup I_{2i,j}$ for some $i \in \{1, \ldots, s_j/2\}$. Note that this interval is not merged again in the remainder of the algorithm. Recall that the intervals $I'_{i,j+1}$, for $i \in \{1, \ldots, s_j/2\}$, are the

possible candidates for merging at iteration j. Let $h' = \overline{\widehat{f}}_{\mathcal{I}'_{j+1}}$ be the distribution obtained by flattening the empirical distribution over these candidate intervals $\mathcal{I}'_{j+1} = \{I'_{1,j+1}, \ldots, I'_{s_j/2,j+1}\}$. Note that $h'(x) = h(x)$ for $x \in J$ because J was created in this iteration.

Let $\mathcal{L}$ be the set of candidate intervals $I'_{i,j+1}$ in the set $\mathcal{I}'_{j+1}$ with the largest $\alpha \cdot t$ errors $e(\widehat{f}, I'_{i,j+1})$. Let $\mathcal{L}_0$ be the intervals in $\mathcal{L}$ that do not contain any jumps of h^*. Since h^* has at most t jumps, $|\mathcal{L}_0| \geq (\alpha - 1)t$. Moreover, for any $I' \in \mathcal{L}_0$, by the triangle inequality

$$\begin{aligned} e(\widehat{f}, I') &= \|h' - \widehat{f}\|_{\mathcal{A}_1, I'} \\ &\leq \|h' - h^*\|_{\mathcal{A}_1, I'} + \|f - h^*\|_{\mathcal{A}_1, I'} + \|f - \widehat{f}\|_{\mathcal{A}_1, I'} \\ &\leq \|h' - h^*\|_{\mathcal{A}_1, I'} + \|f - h^*\|_{1, I'} + \|f - \widehat{f}\|_{\mathcal{A}_1, I'} \ . \end{aligned}$$

Summing over the intervals in $\mathcal{L}_0$,

$$\begin{aligned} \sum_{I' \in \mathcal{L}_0} e(\widehat{f}, I') &\leq \sum_{I' \in \mathcal{L}_0} \left(\|h' - h^*\|_{\mathcal{A}_1, I'} + \|f - h^*\|_{1, I'} + \|f - \widehat{f}\|_{\mathcal{A}_1, I'} \right) \\ &\leq \left(\sum_{I' \in \mathcal{L}_0} \|h' - h^*\|_{\mathcal{A}_1, I'} \right) + \|f - h^*\|_{1, \mathcal{L}_0} + \|f - \widehat{f}\|_{\mathcal{A}_{2\alpha t}, \mathcal{L}_0} \\ &\leq \left(\sum_{I' \in \mathcal{L}_0} \|h' - h^*\|_{\mathcal{A}_1, I'} \right) + \mathrm{OPT}_t + \epsilon \ , \end{aligned} \tag{9}$$

where recall that we had conditioned on the last term being at most ϵ throughout the analysis. Since both h and h^* are flat on each interval $I' \in \mathcal{L}_0$, Lemma 11 gives

$$\sum_{I' \in \mathcal{L}_0} \|h' - h^*\|_{\mathcal{A}_1, I'} \leq \|f - h^*\|_{1, \mathcal{L}_0} + \|\widehat{f} - f\|_{\mathcal{A}_{|\mathcal{L}_0|}, \mathcal{L}_0} \leq \mathrm{OPT}_t + \epsilon \ .$$

Plugging this into (9) gives

$$\sum_{I' \in \mathcal{L}_0} e(\widehat{f}, I') \leq 2 \cdot \mathrm{OPT}_t + 2\epsilon \ .$$

Since J was created by merging in this iteration, we have that $e(\widehat{f}, J)$ is no larger than $e(\widehat{f}, I')$ for any of the intervals $I' \in \mathcal{L}_0$ (see lines 12 - 15 of Algorithm 1), and therefore $e(\widehat{f}, J)$ is not larger than their average. Recalling that $|\mathcal{L}_0| \geq (\alpha - 1)t$, we obtain

$$e(\widehat{f}, J) = \|h' - \widehat{f}\|_{\mathcal{A}_1, J} = \|h - \widehat{f}\|_{\mathcal{A}_1, J} \ \leq \ \frac{\sum_{I' \in \mathcal{L}_0} e(\widehat{f}, I')}{(\alpha - 1)t} \leq \ \frac{2\mathrm{OPT}_t + 2\epsilon}{(\alpha - 1)t} \ ,$$

completing the proof of the lemma. □

4.2 The general merging algorithm

We are now ready to present our general merging algorithm, which is a generalized version of the histogram merging algorithm introduced in Section 4.1. The histogram algorithm only uses three main properties of

histogram hypotheses: (i) The number of intersections between two t-histogram hypotheses is bounded by $O(t)$. (ii) Given an interval J and an empirical distribution $\widehat{f}$, we can efficiently find a good histogram fit to $\widehat{f}$ on this interval. (iii) We can efficiently compute the $\mathcal{A}_1$-errors of candidate intervals.

Note that property (i) bounds the complexity of the hypothesis class and leads to a tight sample complexity bound while properties (ii) and (iii) are algorithmic ingredients. We can generalize these three notions to arbitrary classes of piecewise hypotheses as follows. Let $\mathcal{D}$ be a class of hypotheses. Then the generalized variants of properties (i) to (iii) are: (i) The number of intersections between any two hypotheses in $\mathcal{D}$ is bounded. (ii) Given an interval J and an empirical distribution $\widehat{f}$, we can efficiently find the best fit to $\widehat{f}$ from functions in $\mathcal{D}$ with respect to the $\mathcal{A}_k$-distance. (iii) We can efficiently compute the $\mathcal{A}_k$-distance between the empirical distribution and any hypothesis in $\mathcal{D}$. Using these generalized properties, the histogram merging algorithm naturally extends to agnostically learning piecewise hypotheses where each piece is in the class $\mathcal{D}$.

The following definitions formally describe the aforementioned framework. We first require a mild condition on the underlying distribution family:

Definition 13. *Let $\mathcal{D}$ be a family of measurable functions defined over subsets of I. $\mathcal{D}$ is said to be* full *if for each $J \subseteq I$, there exists a function g in $\mathcal{D}$ whose domain is J. Let $\mathcal{D}_J$ be the elements of $\mathcal{D}$ whose domain is J.*

Our next definition formalizes the notion of piecewise hypothesis whose components come from $\mathcal{D}$:

Definition 14. *A function $h : I \to \mathbb{R}$ is a* t-piece $\mathcal{D}$-function *if there exists a partition of I into intervals $I_1, \ldots, I_{t'}$ with $t' \leq t$, such that for every i, $1 \leq i \leq t'$, there exists $h_i \in \mathcal{D}_{I_i}$ satisfying that $h = h_i$ on I_i. Let $\mathcal{D}_t$ denote the set of all t-piece $\mathcal{D}$-functions.*

The main property we require from our full function class $\mathcal{D}$ is that any two functions in $\mathcal{D}$ intersect a bounded number of times. This is formalized in the definition below:

Definition 15. *Let $\mathcal{D}$ be a full family over I and $J \subseteq I$. Suppose $h \in \mathcal{D}_J$ and $h' \in \mathcal{D}_k$ for some $k \geq 1$. Let $h' = h'_{I_i}$, $1 \leq i \leq k$, for some interval partition $I_1, \ldots, I_k$ of I and $h'_{I_i} \in \mathcal{D}_{I_i}$. Let s denote the number of endpoints of the I_i's contained in J. We say that $\mathcal{D}$ is* d-sign restricted *if the function $h - h'$ has at most $(s+1)d$ sign changes on J, for any h and h'.*

The following simple examples illustrate that histograms and more generally piecewise polynomial functions fall into this framework.

Example 1. *Let $\mathcal{H}_J$ be the set of constant functions defined on J. Then if $\mathcal{H} = \cup_{J \subseteq I} \mathcal{H}_J$, the set $\mathcal{H}_t$ of t-piece $\mathcal{H}$-functions is the set of piecewise constant functions on I with at most t interval pieces. (Note that this class is the set of* t-histograms*.)*

Example 2. *For $J \subseteq I$, we define $\mathcal{P}_{J,d}$ to be set of degree-d nonnegative polynomials on J, and $\mathcal{P}_d \stackrel{def}{=} \cup_J \mathcal{P}_{J,d}$. Since the degree d will be fixed throughout this paper, we sometimes simply denote this set by $\mathcal{P}$. The set $\mathcal{P}_{t,d}$ of t-piece $\mathcal{P}$-functions is the set of t-piecewise degree-d non-negative polynomials. It is easy to see that this class is full over I. Since any two polynomials of degree d intersect at most d times, it is easy to see that $\mathcal{P}_d$ forms a d-sign restricted class.*

We are now ready to formally define our general learning problem. Fix positive integers t, d and a full d-sign restricted class of functions $\mathcal{D}$. Given sample access to any pdf $f : I \to \mathbb{R}_+$, we want to compute a

good $\mathcal{D}_t$ approximation to f. We define $\mathrm{OPT}_{\mathcal{D},t} \stackrel{\text{def}}{=} \inf_{g\in\mathcal{D}_t} \|g-f\|_1$. Our goal is to find an $O(t)$-piece $\mathcal{D}$-function $h : I \to \mathbb{R}$ such that $\|h-f\|_1 \le C\cdot \mathrm{OPT}_{\mathcal{D},t} + O(\epsilon)$, with high probability over the samples, where C is a universal constant.

Our iterative merging algorithm takes as input samples from an arbitrary distribution, and outputs an $O(t)$-piecewise $\mathcal{D}$ hypothesis satisfying the above agnostic guarantee. Our algorithm assumes the existence of two subroutines, which we call $\mathcal{A}_k$-projection and $\mathcal{A}_k$-computation oracles. The $\mathcal{A}_k$-projection oracle was defined in Definition 7 and is restated below along with the definition of the $\mathcal{A}_k$-computation oracle (Definition 16).

Definition 7. *Fix $\eta > 0$. An algorithm $\mathcal{O}_p(\widehat{f}, J, \eta)$ is an* η-approximate $\mathcal{A}_k$-projection oracle *for $\mathcal{D}$ if it takes as input an interval J and $\widehat{f}$, and returns a hypothesis $h \in \mathcal{D}$ such that*

$$\|h-\widehat{f}\|_{\mathcal{A}_k} \le \inf_{h'\in\mathcal{D}} \|h'-\widehat{f}\|_{\mathcal{A}_k,J} + \eta\ .$$

Definition 16. *Fix $\eta > 0$. An algorithm $\mathcal{O}_c(\widehat{f}, h_J, J, \eta)$ is an* η-approximate $\mathcal{A}_k$-computation oracle *for $\mathcal{D}$ if it takes as input $\widehat{f}$, a subinterval $J \subseteq I$, and a function $h_J \in \mathcal{D}_J$, and returns a value ξ such that*

$$\left| \|h_J-\widehat{f}\|_{\mathcal{A}_k,J} - \xi \right| \le \eta\ .$$

We consider a d-sign restricted full family $\mathcal{D}$, and a fixed $\eta > 0$. Let $R_p(I) = R_p(I, \widehat{f}, \mathcal{O}_p)$ and $R_c(I) = R_c(I, \widehat{f}, \mathcal{O}_c)$ be the time used by the oracle $\mathcal{O}_p$ and $\mathcal{O}_c$, respectively. With a slight abuse of notation, for a collection of at most $2n$ intervals containing n points in the support of the empirical distribution, we also define $R_p(n)$ and $R_c(n)$ to be the maximum time taken by $\mathcal{O}_p$ and $\mathcal{O}_c$, respectively.

We are now ready to state the main theorem of this section:

Theorem 17. *Let $\mathcal{O}_p$ and $\mathcal{O}_c$ be η-approximate $\mathcal{A}_k$-projection and $\mathcal{A}_k$-computation oracles for $\mathcal{D}$. Algorithm* GENERAL-MERGING$(f, t, \alpha, \epsilon, \delta)$ *draws $n = O((\alpha dt + \log(1/\delta))/\epsilon^2)$ samples, has time complexity $O\left((R_p(n) + R_c(n)) \log \frac{n}{\alpha t}\right)$, and outputs a hypothesis h and an interval partition $\mathcal{I}$ such that $|\mathcal{I}| \le 2\alpha\cdot t$ and with probability at least $1-\delta$, we have*

$$\|h-f\|_1 \le 3\cdot \mathrm{OPT}_{\mathcal{D},t} + \frac{\mathrm{OPT}_{\mathcal{D},t}+\epsilon}{\alpha-1} + 2\epsilon + \eta\ . \tag{10}$$

In the remainder of this section, we provide an intuitive explanation of our general merging algorithm followed by a detailed pseudocode.

The algorithm GENERAL-MERGING and its analysis is a generalization of the CONSTRUCTHISTOGRAM algorithm from the previous subsection. More formally, the algorithm proceeds greedily, as before. We take $n = O((\alpha dt + \log 1/\delta)/\epsilon^2)$ samples $x_1 \le \ldots \le x_n$. We construct $\mathcal{I}_0$ as in (1). In the j-th iteration, given the current partition $\mathcal{I}_j = \{I_{1,j}, \ldots, I_{s_j,j}\}$ with s_j intervals, consider the intervals

$$I'_{\ell,j+1} = I_{2\ell-1,j} \cup I_{2\ell,j}$$

for $\ell \le s_j/2$. As for histograms, we want to compute the errors in each of the new intervals created. To do this, we first call the $\mathcal{A}_k$–projection oracle with $k = d+1$ on this interval to find the approximately best fit in $\mathcal{D}$ for $\widehat{f}$ over these new intervals, namely:

$$h'_{\ell,j} = \mathcal{O}_p\left(\widehat{f}, I'_{\ell,j+1}, \frac{\eta}{O(t)}\right).$$

To compute the error, we call the $\mathcal{A}_k$–computation oracle with $k = d + 1$, i.e.:

$$e_{\ell,j} = \mathcal{O}_c\left(\widehat{f}, h'_{\ell,j}, I'_{\ell,j+1}, \frac{\eta}{O(t)}\right) .$$

As in CONSTRUCTHISTOGRAM, we keep the intervals with the largest $O(\alpha t)$ errors intact and merge the remaining pairs of intervals. We perform this procedure $O(\log \frac{n}{\alpha t})$ times and arrive at some final partition $\mathcal{I}$ with $O(\alpha t)$ pieces. Our output hypothesis is the output of $\mathcal{O}_p(\widehat{f}, I)$ over each of the final intervals I.

The formal pseudocode for our algorithm is given in Algorithm 2. We assume that $\mathcal{D}$ and d are known and fixed and are not mentioned explicitly as an input to the algorithm. Note that we run the algorithm with $\eta = \epsilon$ so that Theorem 17 has an additional $O(\epsilon)$ error. The proof of Theorem 17 is very similar to that of the histogram merging algorithm and is deferred to Appendix A.

Algorithm 2 Approximating with general hypotheses by merging.

function GENERAL-MERGING($f, d, t, \alpha, \epsilon, \delta$)
  Draw $n = \Theta((\alpha d t + \log 1/\delta)/\epsilon^2)$ samples $x_1 \le x_2 \le \ldots \le x_n$.
  Form the empirical distribution $\widehat{f}$ from these samples.
  Let $\mathcal{I}_0 \leftarrow \{[a, x_1), [x_1, x_1], (x_1, x_2), \ldots, (x_{n-1}, x_n), [x_n, x_n], (x_n, b]\}$ be the initial partition.
  $j \leftarrow 0$
  while $|\mathcal{I}_j| > 2\alpha \cdot t$ **do**
    Let $\mathcal{I}_j = \{I_{1,j}, I_{2,j}, \ldots, I_{s_j-1,j}, I_{s_j,j}\}$
    for $\ell \in \{1, 2, \ldots, \frac{s_j}{2}\}$ **do**
      $I'_{\ell,j+1} \leftarrow I_{2\ell-1,j} \cup I_{2\ell,j}$
      $h'_{\ell,j} \leftarrow \mathcal{O}_p(\widehat{f}, I'_{\ell,j+1}, \frac{\epsilon}{2\alpha t})$
      $e_{\ell,j} \leftarrow \mathcal{O}_c(\widehat{f}, h'_{\ell,j}, I'_{\ell,j+1}, \frac{\epsilon}{2\alpha t})$
    end for
    Let L be the set of $\ell \in \{1, 2, \ldots, \frac{s_j}{2}\}$ with the αt largest errors $e_{\ell,j}$.
    Let M be the complement of L.
    $\mathcal{I}_{j+1} \leftarrow \bigcup_{\ell \in L} \{I_{2\ell-1,j}, I_{2\ell,j}\}$
    $\mathcal{I}_{j+1} \leftarrow \mathcal{I}_{j+1} \cup \{I'_{\ell,j+1} \mid \ell \in M\}$
    $j \leftarrow j + 1$
  end while
  return $\mathcal{I} = \mathcal{I}_j$ and the functions $\mathcal{O}_p(\widehat{f}, J, \frac{\epsilon}{2\alpha t})$ for $J \in \mathcal{I}$
end function

4.3 Putting everything together

In Sections 5 and 6.3, we present an efficient approximate $\mathcal{A}_k$-projection oracle and an $\mathcal{A}_k$-computation oracle for $\mathcal{P}_d$, respectively. We show that:

Theorem 18. *Fix $J \subseteq [-1, 1]$ and $\eta > 0$. For all $k \le d$, there is an η-approximate $\mathcal{A}_k$-projection oracle for $\mathcal{P}_d$ which runs in time*

$$O\left(\left(d^3 \log\log 1/\eta + sd^2 + d^{\omega+2}\right) \log^2 \frac{1}{\eta}\right).$$

where s is the number of samples in the interval J.

Theorem 19. *There is an η-approximate $\mathcal{A}_k$-computation oracle for $\mathcal{P}_d$ which runs in time $O((s+d)\log^2(s+d))$ where s is the number of samples in the interval J.*

The algorithm GENERALMERGING, when used in conjunction with the oracles $\mathcal{O}_p$ and $\mathcal{O}_c$ given in Theorems 18 and 19 (for $\eta = \epsilon$), yields Theorem 1. For this choice of oracles we have that $R_p(n)+R_c(n) = O(n \cdot d^{\omega+2} \cdot \log^3 \frac{1}{\epsilon})$. This completes the proof.

5 A fast $\mathcal{A}_k$-projection oracle for polynomials

We now turn our attention to the $\mathcal{A}_k$-projection problem, which appears as the main subroutine in the general merging algorithm (see Section 4.2). In this section, we let $E \subset J$ be the set of samples drawn from the unknown distribution. To emphasize the dependence of the empirical distribution on E, we denote the empirical distribution by $\widehat{f}_E$ in this section. Given an interval $J = [a, b]$ and a set of samples $E \subset J$, the goal of the $\mathcal{A}_k$-projection oracle is to find a hypothesis $h \in \mathcal{D}$ such that the $\mathcal{A}_k$-distance between the empirical distribution $\widehat{f}_E$ and the hypothesis h is minimized. In contrast to the merging algorithm, the $\mathcal{A}_k$-projection oracle depends on the underlying hypothesis class $\mathcal{D}$, and here we present an efficient oracle for non-negative polynomials with fixed degree d. In particular, our $\mathcal{A}_k$-projection oracle computes the coefficients $c \in \mathbb{R}^{d+1}$ of a degree-d polynomial p_c that approximately minimizes the $\mathcal{A}_k$-distance to the given empirical distribution $\widehat{f}_E$ in the interval J. Moreover, our oracle ensures that p_c is non-negative for all $x \in J$.

At a high-level, we formulate the $\mathcal{A}_k$-projection as a convex optimization problem. A key insight is that we can construct an efficient, approximate *separation oracle* for the set of polynomials that have an $\mathcal{A}_k$-distance of at most τ to the empirical distribution $\widehat{f}_E$. Combining this separation oracle with existing convex optimization algorithms allows us to solve the feasibility problem of checking whether we can achieve a given $\mathcal{A}_k$-distance τ. We then convert the feasibility problem to the optimization variant via a binary search over τ.

In order to simplify notation, we assume that the interval J is $[-1, 1]$ and that the mass of the empirical distribution $\widehat{f}_E$ is 1. Note that the general $\mathcal{A}_k$-projection problem can easily be converted to this special case by shifting and scaling the sample locations and weights before passing them to the $\mathcal{A}_k$-projection subroutine. Similarly, the resulting polynomial can be transformed to the original interval and mass of the empirical distribution on this interval.[3]

5.1 The set of feasible polynomials

For the feasibility problem, we are interested in the set of degree-d polynomials that have an $\mathcal{A}_k$-distance of at most τ to the empirical distribution $\widehat{f}_E$ on the interval $J = [-1, 1]$ and are also non-negative on J. More formally, we study the following set.

Definition 20 (Feasible polynomials)**.** *Let $E \subset J$ be the samples of an empirical distribution with $\widehat{f}_E(J) = 1$. Then the set of (τ, d, k, E)-feasible polynomials is*

$$\mathcal{C}_{\tau,d,k,E} := \left\{ c \in \mathbb{R}^{d+1} \mid \|p_c - \widehat{f}_E\|_{\mathcal{A}_k,J} \leq \tau \text{ and } p_c(x) \geq 0 \text{ for all } x \in J \right\} .$$

[3]Technically, this step is actually necessary in order to avoid a running time that depends on the shape of the unknown pdf f. Since the pdf f could be supported on a very small interval only, the corresponding polynomial approximation could require arbitrarily large coefficients (the empirical distribution would have all samples in a very small interval). In that case, operations such as root-finding with good precision could take an arbitrary amount of time. In order to circumvent this issue, we make use of the real-RAM model to rescale our samples to $[-1, 1]$ before processing them further. Combined with the assumption of unit probability mass, this allows us to bound the coefficients of candidate polynomials in the current interval.

When d, k, and E are clear from the context, we write only $\mathcal{C}_\tau$ for the set of τ-feasible polynomials.

Considering the original $\mathcal{A}_k$-projection problem, we want to find an element $c^* \in \mathcal{C}_{\tau^*}$, where τ^* is the smallest value for which $\mathcal{C}_{\tau^*}$ is non-empty. We solve a slightly relaxed version of this problem, i.e., we find an element c for which the $\mathcal{A}_k$-constraint and the non-negativity constraint are satisfied up to small additive constants. We then post-process the polynomial p_c to make it truly non-negative while only increasing the $\mathcal{A}_k$-distance by a small amount.

Note that we can "unwrap" the definition of the $\mathcal{A}_k$-distance and write $\mathcal{C}$ as an intersection of sets in which each set enforces the constraint $\sum_{i=1}^{k} |p_c(I_i) - \widehat{f}_E(I_i)| \leq \tau$ for one collection of k disjoint intervals $\{I_1, \ldots, I_k\}$. For a fixed collection of intervals, we can then write each $\mathcal{A}_k$-constraint as the intersection of *linear* constraints in the space of polynomials. Similarly, we can write the non-negativity constraint as an intersection of pointwise non-negativity constraints, which are again linear constraints in the space of polynomials. This leads us to the following key lemma. Note that convexity of $\mathcal{C}_\tau$ could be established more directly[4], but considering $\mathcal{C}_\tau$ as an intersection of halfspaces illustrates the further development of our algorithm (see also the comments after the lemma).

Lemma 21 (Convexity)**.** *The set of τ-feasible polynomials is convex.*

Proof. From the definitions of $\mathcal{C}_\tau$ and the $\mathcal{A}_k$-distance, we have

$$\begin{aligned}
\mathcal{C}_\tau &= \{c \in \mathbb{R}^{d+1} \mid \|p_c - \widehat{f}_E\|_{\mathcal{A}_k, J} \leq \tau \text{ and } p_c(x) \geq 0 \text{ for all } x \in J\} \\
&= \{c \in \mathbb{R}^{d+1} \mid \sup_{\mathcal{I} \in \mathfrak{I}_J^k} \sum_{I \in \mathcal{I}} |p_c(I) - \widehat{f}_E(I)| \leq \tau\} \cap \{c \in \mathbb{R}^{d+1} \mid p_c(x) \geq 0 \text{ for all } x \in J\} \\
&= \bigcap_{\mathcal{I} \in \mathfrak{I}_J^k} \{c \in \mathbb{R}^{d+1} \mid \sum_{I \in \mathcal{I}} |p_c(I) - \widehat{f}_E(I)| \leq \tau\} \cap \bigcap_{x \in J} \{c \in \mathbb{R}^{d+1} \mid p_c(x) \geq 0\} \\
&= \bigcap_{\mathcal{I} \in \mathfrak{I}_J^k} \bigcap_{\xi \in \{-1,1\}^k} \{c \in \mathbb{R}^{d+1} \mid \sum_{i=1}^{k} \xi_i (p_c(I_i) - \widehat{f}_E(I_i)) \leq \tau\} \cap \bigcap_{x \in J} \{c \in \mathbb{R}^{d+1} \mid p_c(x) \geq 0\} \,.
\end{aligned}$$

In the last line, we used the notation $\mathcal{I} = \{I_1, \ldots, I_k\}$. Since the intersection of a family of convex sets is convex, it remains to show that the individual $\mathcal{A}_k$-distance sets and non-negativity sets are convex. Let

$$\begin{aligned}
\mathcal{M} &= \bigcap_{\mathcal{I} \in \mathfrak{I}_J^k} \bigcap_{\xi \in \{-1,1\}^k} \{c \in \mathbb{R}^{d+1} \mid \sum_{i=1}^{k} \xi_i (p_c(I_i) - \widehat{f}_E(I_i)) \leq \tau\} \\
\mathcal{N} &= \bigcap_{x \in J} \{c \in \mathbb{R}^{d+1} \mid p_c(x) \geq 0\} \,.
\end{aligned}$$

We start with the non-negativity constraints encoding the set $\mathcal{N}$. For a fixed $x \in J$, we can expand the constraint $p_c(x) \geq 0$ as

$$\sum_{i=0}^{d} c_i \cdot x^i \geq 0 \,,$$

which is clearly a linear constraint on the c_i. Hence, the set $\{c \in \mathbb{R}^{d+1} \mid p_c(x) \geq 0\}$ is a halfspace for a fixed x and thus also convex.

[4]Norms give rise to convex sets and the set of non-negative polynomials is also convex.

Next, we consider the $\mathcal{A}_k$-constraints $\sum_{i=1}^{k} \xi_i(p_c(I_i) - \widehat{f}_E(I_i)) \leq \tau$ for the set $\mathcal{M}$. Since the intervals $I_1, \ldots, I_k$ are now fixed, so is $\widehat{f}_E(I_i)$. Let α_i and β_i be the endpoints of the interval I_i, i.e., $I_i = [\alpha_i, \beta_i]$. Then we have

$$p_c(I_i) \;=\; \int_{\alpha_i}^{\beta_i} p_c(x)\,\mathrm{d}x \;=\; P_c(\beta_i) - P_c(\alpha_i)\;,$$

where $P_c(x)$ is the indefinite integral of $P_c(x)$, i.e.,

$$P_c(x) \;=\; \sum_{i=0}^{d} c_i \cdot \frac{x^{i+1}}{i+1}\;.$$

So for a fixed x, $P_c(x)$ is a linear combination of the c_i. Consequently $\sum_{i=1}^{k} \xi_i p_c(I_i)$ is also a linear combination of the c_i, and hence each set in the intersection defining $\mathcal{M}$ is a halfspace. This shows that $\mathcal{C}_\tau$ is a convex set. □

It is worth noting that the set $\mathcal{N}$ is a spectrahedron (the feasible set of a semidefinite program) because it encodes non-negativity of a univariate polynomial over a fixed interval. After restricting the set of coefficients to non-negative polynomials, we can simplify the definition of the $\mathcal{A}_k$-distance: it suffices to consider sets of intervals with endpoints at the locations of samples (see Lemma 37). Hence, we can replace the supremum in the definition of $\mathcal{M}$ by a maximum over a finite set, which shows that $\mathcal{C}_\tau$ is also a spectrahedron. This suggests that the $\mathcal{A}_k$-projection problem could be solved by a black-box application of an SDP solver. However, this would lead to a running time that is *exponential* in k because there are more than $\binom{|E|}{2k}$ possible sets of intervals. While the authors of [CDSS14] introduce an encoding of the $\mathcal{A}_k$-constraint with fewer linear inequalities, their approach increases the number of variables in the optimization problem to depend polynomially on $\frac{1}{\epsilon}$, which leads to a super-linear running time.

Instead of using black-box SDP or LP solvers, we construct an algorithm that exploits additional structure in the $\mathcal{A}_k$-projection problem. Most importantly, our algorithm separates the dimension of the desired degree-d polynomial from the number of samples (or equivalently, the error parameter ϵ). This allows us to achieve a running time that is *nearly-linear* for a wide range of distributions. Interestingly, we can solve our SDP significantly faster than the LP which has been proposed in [CDSS14] for the same problem.

5.2 Separation oracles and approximately feasible polynomials

In order to work with the large number of $\mathcal{A}_k$-constraints efficiently, we "hide" this complexity from the convex optimization procedure by providing access to the constraints only through a separation oracle. As we will see in Section 6, we can utilize the structure of the $\mathcal{A}_k$-norm and implement such a separation oracle for the $\mathcal{A}_k$-constraints in nearly-linear time. Before we give the details of our separation oracle, we first show how we can solve the $\mathcal{A}_k$-projection problem assuming that we have such an oracle. We start by formally defining our notions of separation oracles.

Definition 22 (Separation oracle)**.** *A separation oracle $\mathcal{O}$ for the convex set $\mathcal{C}_\tau$ is a function that takes as input a coefficient vector $c \in \mathbb{R}^{d+1}$ and returns one of the following two results:*

1. *"yes" if $c \in \mathcal{C}_\tau$.*
2. *a separating hyperplane $y \in \mathbb{R}^{d+1}$. The hyperplane y must satisfy $y^T c' \leq y^T c$ for all $c' \in \mathcal{C}_\tau$.*

For general polynomials, it is not possible to perform basic operations such as root finding *exactly*, and hence we have to resort to approximate methods. This motivates the following definition of an *approximate* separation oracle. While an approximate separation oracle might accept a point c that is not in the set $\mathcal{C}_\tau$, the point c is then guaranteed to be close to C_τ.

Definition 23 (Approximate separation oracle)**.** *A μ-approximate separation oracle $\mathcal{O}$ for the set $\mathcal{C}_\tau = \mathcal{C}_{\tau,d,k,E}$ is a function that takes as input a coefficient vector $c \in \mathbb{R}^{d+1}$ and returns one of the following two results, either "yes" or a hyperplane $y \in \mathbb{R}^{d+1}$.*

1. *If $\mathcal{O}$ returns "yes", then $\|p_c - \widehat{f}_E\|_{\mathcal{A}_k,J} \leq \tau + 2\mu$ and $p_c(x) \geq -\mu$ for all $x \in J$.*
2. *If $\mathcal{O}$ returns a hyperplane, then y is a separating hyperplane; i.e. the hyperplane y must satisfy $y^T c' \leq y^T c$ for all $c' \in \mathcal{C}_\tau$.*

In the first case, we say that p_c is a 2μ-approximately feasible polynomial.

Note that in our definition, separating hyperplanes must still be exact for the set $\mathcal{C}_\tau$. Although our membership test is only approximate, the exact hyperplanes allow us to employ several existing separation oracle methods for convex optimization. We now formally show that many existing methods still provide approximation guarantees when used with our approximate separation oracle.

Definition 24 (Separation Oracle Method)**.** *A separation oracle method (SOM) is an algorithm with the following guarantee: let $\mathcal{C}$ be a convex set that is contained in a ball of radius 2^L. Moreover, let $\mathcal{O}$ be a separation oracle for the set $\mathcal{C}$. Then SOM$(\mathcal{O}, L)$ returns one of the following two results:*

1. *a point $x \in \mathcal{C}$.*
2. *"no" if $\mathcal{C}$ does not contain a ball of radius 2^{-L}.*

We say that an SOM is canonical *if it interacts with the separation oracle in the following way: the first time the separation oracle returns "yes" for the current query point x, the SOM returns the point x as its final answer.*

There are several algorithms satisfying this definition of a separation oracle method, e.g., the classical Ellipsoid method [Kha79] and Vaidya's cutting plane method [Vai89]. Moreover, all of these algorithms also satisfy our notion of a *canonical* separation oracle method. We require this technical condition in order to prove that our approximate separation oracles suffice. In particular, by a straightforward simulation argument, we have the following:

Theorem 25. *Let $\mathcal{M}$ be a canonical separation oracle method, and let $\mathcal{O}$ be a μ-approximate separation oracle for the set $\mathcal{C}_\tau = \mathcal{C}_{\tau,d,k,E}$. Moreover, let L be such that $\mathcal{C}_\tau$ is contained in a ball of radius 2^L. Then $\mathcal{M}(\mathcal{O}, L)$ returns one of the following two results:*

1. *a coefficient vector $c \in \mathbb{R}^{d+1}$ such that $\|p_c - \widehat{f}_E\|_{\mathcal{A}_k,J} \leq \tau + 2\mu$ and $p_c(x) \geq -\mu$ for all $x \in J$.*
2. *"no" if $\mathcal{C}$ does not contain a ball of radius 2^{-L}.*

5.3 Bounds on the radii of enclosing and enclosed balls

In order to bound the running time of the separation oracle method, we establish bounds on the ball radii used in Theorem 25.

Upper bound When we initialize the separation oracle method, we need a ball of radius 2^L that contains the set $\mathcal{C}_\tau$. For this, we require bounds on the coefficients of polynomials which are bounded in L_1 norm. Bounds of this form were first established by Markov [Mar92].

Lemma 26. *Let p_c be a degree-d polynomial with coefficients $c \in \mathbb{R}^{d+1}$ such that $p(x) \geq 0$ for $x \in [-1,1]$ and $\int_{-1}^{1} p(x)\,\mathrm{d}x \leq \alpha$, where $\alpha > 0$. Then we have*

$$|c_i| \leq \alpha \cdot (d+1)^2 \cdot (\sqrt{2}+1)^d \quad \textit{for all } i = 0, \ldots, d\ .$$

This lemma is well-known, but for completeness, we include a proof in Appendix B. Using this lemma, we obtain:

Theorem 27 (Upper radius bound)**.** *Let $\tau \leq 1$ and let A be the $(d+1)$-ball of radius $r = 2^{L_u}$ where*

$$L_u = d\log(\sqrt{2}+1) + \frac{3}{2}\log d + 2\ .$$

Then $\mathcal{C}_{\tau,d,k,E} \subseteq A$.

Proof. Let $c \in \mathcal{C}_{\tau,d,k,E}$. From basic properties of the L_1- and $\mathcal{A}_k$-norms, we have

$$\int_{-1}^{1} p_c\,\mathrm{d}x \ = \ \|p_c\|_{1,J} \ = \ \|p_c\|_{\mathcal{A}_k,J} \ \leq \ \|\widehat{f}_E\|_{\mathcal{A}_k,J} + \|p_c - \widehat{f}_E\|_{\mathcal{A}_k,J} \ \leq \ 1+\tau \ \leq \ 2\ .$$

Since p_c is also non-negative on J, we can apply Lemma 26 and get

$$|c_i| \leq 2 \cdot (d+1) \cdot (\sqrt{2}+1)^d \quad \text{for all } i = 0, \ldots, d\ .$$

Note that the above constraints define a hypercube B with side length $s = 4 \cdot (d+1) \cdot (\sqrt{2}+1)^d$. The ball A contains the hypercube B because $r = \sqrt{d+1} \cdot s$ is the length of the longest diagonal of B. This implies that $C_{\tau,d,k,E} \subseteq B \subseteq A$. □

Lower bound Separation oracle methods typically cannot directly certify that a convex set is empty. Instead, they reduce the volume of a set enclosing the feasible region until it reaches a certain threshold. We now establish a lower bound on volumes of sets $\mathcal{C}_{\tau+\eta}$ that are feasible by at least a margin η in the $\mathcal{A}_k$-distance. If the separation oracle method cannot find a small ball in $\mathcal{C}_{\tau+\eta}$, we can conclude that achieving an $\mathcal{A}_k$-distance of τ is infeasible.

Theorem 28 (Lower radius bound)**.** *Let $\eta > 0$ and let τ be such that $\mathcal{C}_\tau = \mathcal{C}_{\tau,d,k,E}$ is non-empty. Then $\mathcal{C}_{\tau+\eta}$ contains a ball of radius $r = 2^{-L_\ell}$, where*

$$L_\ell \ = \ \log \frac{4(d+1)}{\eta}\ .$$

Proof. Let c^* be the coefficients of a feasible polynomial, i.e., $c^* \in \mathcal{C}_\tau$. Moreover, let c be such that

$$c_i = \begin{cases} c_0^* + \frac{\eta}{4} & \text{if } i = 0 \\ c_i^* & \text{otherwise} \end{cases}\ .$$

Since p_{c^*} is non-negative on J, we also have $p_c(x) \geq \frac{\eta}{4}$ for all $x \in J$. Moreover, it is easy to see that shifting the polynomial p_{c^*} by $\frac{\eta}{4}$ changes the $\mathcal{A}_k$-distance to $\widehat{f}_E$ by at most $\frac{\eta}{2}$ because the interval J has length 2.

Hence, $\|p_c - \widehat{f}_E\|_{\mathcal{A}_k,J} \leq \tau + \frac{\eta}{2}$ and so $c \in \mathcal{C}_{\tau+\eta}$. We now show that we can perturb the coefficients of c slightly and still stay in the set of feasible polynomials $\mathcal{C}_{\tau+\eta}$.

Let $\nu = \frac{\eta}{4(d+1)}$ and consider the hypercube

$$B = \{c' \in \mathbb{R}^{d+1} \mid c'_i \in [c_i - \nu,\, c_i + \nu] \text{ for all } i\} \,.$$

Note that B contains a ball of radius $\nu = 2^{-L_\ell}$. First, we show that $p_{c'}(x) \geq 0$ for all $x \in J$ and $c' \in B$. We have

$$\begin{aligned}
p_{c'}(x) &= \sum_{i=0}^{d} c'_i x_i \\
&= \sum_{i=0}^{d} c_i x^i + \sum_{i=0}^{d} (c'_i - c_i) x^i \\
&\geq p_c(x) - \sum_{i=0}^{d} \nu |x^i| \\
&\geq \frac{\eta}{4} - (d+1) \cdot \nu \\
&\geq 0 \,.
\end{aligned}$$

Next, we turn our attention to the $\mathcal{A}_k$-distance constraint. In order to show that $p_{c'}$ also achieves a good $\mathcal{A}_k$-distance, we bound the L_1-distance to p_c.

$$\begin{aligned}
\|p_c(x) - p_{c'}(x)\|_{1,J} &= \int_{-1}^{1} |p_c(x) - p_{c'}(x)| \,\mathrm{d}x \\
&\leq \int_{-1}^{1} \sum_{i=0}^{d} \nu \cdot |x^i| \,\mathrm{d}x \\
&\leq \int_{-1}^{1} (d+1)\nu \,\mathrm{d}x \\
&= 2(d+1)\nu \\
&\leq \frac{\eta}{2} \,.
\end{aligned}$$

Therefore, we get

$$\begin{aligned}
\|p_{c'} - \widehat{f}_E\|_{\mathcal{A}_k,J} &\leq \|p_{c'} - p_c\|_{\mathcal{A}_k,J} + \|p_c - \widehat{f}_E\|_{\mathcal{A}_k,J} \\
&\leq \|p_{c'} - p_c\|_{1,J} + \tau + \frac{\eta}{2} \\
&\leq \tau + \eta \,.
\end{aligned}$$

This proves that $c' \in \mathcal{C}_{\tau+\eta}$ and hence $B \subseteq \mathcal{C}_{\tau+\eta}$. □

5.4 Finding the best polynomial

We now relate the feasibility problem to our original optimization problem of finding a non-negative polynomial with minimal $\mathcal{A}_k$-distance. For this, we perform a binary search over the $\mathcal{A}_k$-distance and choose

Algorithm 3 Finding polynomials with small $\mathcal{A}_k$-distance.

function FINDPOLYNOMIAL(d, k, E, η)
  ▷ Initial definitions
  Let $\eta' = \frac{\eta}{15}$.
  Let $L_u = d\log(\sqrt{2}+1) + \frac{3}{2}\log d + 2$.
  Let $L_\ell = \log \frac{4(d+1)}{2\eta'}$.
  Let $L = \max(L_u, L_\ell)$.
  Let $\mathcal{M}$ be a canonical separation oracle method.
  Let $\mathcal{O}_\tau$ be an η'-approximate separation oracle
    for the set of (τ, d, k, E)-feasible polynomials.
  $\tau_\ell \leftarrow 0$
  $\tau_u \leftarrow 1$
  while $\tau_u - \tau_\ell \geq \eta'$ **do**
    $\tau_m \leftarrow \frac{\tau_\ell + \tau_u}{2}$
    $\tau'_m \leftarrow \tau_m + 2\eta'$
    if $\mathcal{M}(\mathcal{O}_{\tau'_m}, L)$ returned a point **then**
      $\tau_u \leftarrow \tau_m$
    else
      $\tau_\ell \leftarrow \tau_m$ ▷ $\mathcal{C}_{\tau_{m'}}$ does not contain a ball of radius 2^{-L} and hence $\mathcal{C}_{\tau_m}$ is empty.
    end if
  end while
  $c' \leftarrow \mathcal{M}(\mathcal{O}_{\tau_u + 10\eta'}, L)$ ▷ Find final coefficients.
  $c_0 \leftarrow c'_0 + \eta'$ and $c_i \leftarrow c'_i$ for $i \neq 0$ ▷ Ensure non-negativity.
  return c
end function

our error parameters carefully in order to achieve the desired approximation guarantee. See Algorithm 3 for the corresponding pseudocode.

The main result for our $\mathcal{A}_k$-oracle is the following:

Theorem 29. *Let $\eta > 0$ and let τ^* be the smallest $\mathcal{A}_k$-distance to the empirical distribution $\widehat{f}_E$ achievable with a non-negative degree-d polynomial on the interval J, i.e., $\tau^* = \min_{h \in \mathcal{P}_{J,d}} \|h - \widehat{f}_E\|_{\mathcal{A}_k,J}$. Then* FINDPOLYNOMIAL *returns a coefficient vector $c \in \mathbb{R}^{d+1}$ such that $p_c(x) \geq 0$ for all $x \in J$ and $\|p_c - \widehat{f}_E\|_{\mathcal{A}_k,J} \leq \tau^* + \eta$.*

Proof. We use the definitions in Algorithm 3. Note that τ^* is the smallest value for which $\mathcal{C}_{\tau^*} = \mathcal{C}_{\tau^*,d,k,E}$ is non-empty. First, we show that the binary search maintains the following invariants: $\tau_\ell \leq \tau^*$ and there exists a $4\eta'$-approximately τ_u-feasible polynomial. This is clearly true at the beginning of the algorithm: (i) Trivially, $\tau^* \geq 0 = \tau_\ell$. (ii) For $c = (0, 0, \cdots, 0)^T$, we have $\|p_c - \widehat{f}_E\|_{\mathcal{A}_k,J} \leq 1 = \tau_u$ and $p_c(x) \geq 0$, so p_c is τ_u-feasible (and hence also approximately τ_u-feasible).

Next, we consider the two cases in the while-loop:

1. If the separation oracle method returns a coefficient vector c such that the polynomial p_c is $2\eta'$-approximately τ'_m-feasible, then p_c is also $4\eta'$-approximately τ_m-feasible because $\tau'_m = \tau_m + 2\eta'$. Hence, the update of τ_u preserves the loop invariant.

2. If the separation oracle method returns that $\mathcal{C}_{\tau'_m}$ does not contain a ball of radius 2^{-L}, then τ_m must be empty (by the contrapositive of Theorem 28). Hence, we have $\tau^* \geq \tau_m$ and the update of τ_ℓ preserves the loop invariant.

We now analyze the final stage of FINDPOLYNOMIAL after the while-loop. First, we show that $\mathcal{C}_{\tau_u + 8\eta'}$ is non-empty by identifying a point in the set. From the loop invariant, we know that there is a coefficient vector v' such that $p_{v'}$ is a $4\eta'$-approximately τ_u-feasible polynomial. Consider v with $v_0 := v'_0 + 2\eta'$ and $v_i := v'_i$ for $i \neq 0$. Then we have

$$\|p_v - p_{v'}\|_{1,J} \;=\; \int_{-1}^{1} |p_v(x) - p_{v'}(x)| \,\mathrm{d}x \;=\; \int_{-1}^{1} 2\eta' \,\mathrm{d}x \;=\; 4\eta' \,.$$

Hence, we also get

$$\|p_v - \widehat{f}_E\|_{\mathcal{A}_k,J} \overset{(a)}{\leq} \|p_v - p_{v'}\|_{\mathcal{A}_k,J} + \|p_{v'} - \widehat{f}_E\|_{\mathcal{A}_k,J} \overset{(b)}{\leq} \|p_v - p_{v'}\|_{1,J} + \tau_u + 4\eta' \;\leq\; \tau_u + 8\eta' \,.$$

We used the triangle inequality in (a) and the fact that $p_{v'}$ is $4\eta'$-approximately τ_u-feasible in (b). Moreover, we have $p_{v'}(x) \geq -2\eta'$ for all $x \in J$ and thus $p_v(x) \geq 0$ for all $x \in J$. This shows that $\mathcal{C}_{\tau_u + 8\eta'}$ is non-empty because $v \in \mathcal{C}_{\tau_u + 8\eta'}$.

Finally, consider the last run of the separation oracle method in line 20 of Algorithm 3. Since $\mathcal{C}_{\tau_u + 8\eta'}$ is non-empty, Theorem 28 shows that $\mathcal{C}_{\tau_u + 10\eta'}$ contains a ball of radius 2^{-L}. Hence, the separation oracle method must return a coefficient vector $c' \in \mathbb{R}^{d+1}$ such that $p_{c'}$ is $2\eta'$-approximately $\tau_u + 10\eta'$-feasible. Using a similar argument as for v, we can make $p_{c'}$ non-negative while increasing its $\mathcal{A}_k$-distance to $\widehat{f}_E$ by only $2\eta'$, i.e., we can show that $p_c(x) \geq 0$ for all $x \in J$ and that

$$\|p_c - \widehat{f}_E\|_{\mathcal{A}_k,J} \leq \tau_u + 14\eta' \,.$$

Since $\tau_u - \tau_\ell \leq \eta'$ and $\tau_\ell \leq \tau^*$, we have $\tau_u \leq \tau^* + \eta'$. Therefore, $\tau_u + 14\eta' \leq \tau^* + 15\eta' = \tau^* + \eta$, which gives the desired bound on $\|p_c - \widehat{f}_E\|_{\mathcal{A}_k,J}$. □

In order to state a concrete running time, we instantiate our algorithm FINDPOLYNOMIAL with Vaidya's cutting plane method as the separation oracle method. In particular, Vaidya's algorithm runs in time $O(TdL+d^{\omega+1}L)$ for a feasibility problem in dimension d and ball radii bounds of 2^L and 2^{-L}, respectively. T is the cost of a single call to the separation oracle and ω is the matrix-multiplication constant. Then we get:

Theorem 30. *Let $\mathcal{O}$ be an $\frac{\eta}{14}$-approximate separation oracle that runs in time T. Then* FINDPOLYNOMIAL *has time complexity $O((Td^2+d^{\omega+2})\log^2\frac{1}{\eta})$.*

Proof. The running time of FINDPOLYNOMIAL is dominated by the binary search. It is easy to see that the binary search performs $O(\log\frac{1}{\eta})$ iterations, in which the main operation is the call to the separation oracle method. Our bounds on the ball radii in Theorems 27 and 28 imply $L=O(d+\log\frac{1}{\eta})$. Combining this with the running time bound for Vaidya's algorithm gives the time complexity stated in the theorem. □

In Section 6 we describe a μ-approximate separation oracle that runs in time $\widetilde{O}(dk+d\log\log 1/\mu+s)$, where s is the number of samples in the empirical distribution on the interval J. Plugging this oracle directly into our algorithm FINDPOLYNOMIAL gives an η-approximate $\mathcal{A}_k$-projection oracle which runs in time $O((d^3k+d^3\log\log 1/\eta+sd^2+d^{\omega+2})\log^2\frac{1}{\eta})$. This algorithm is the algorithm promised in Theorem 18.

6 The separation oracle and the $\mathcal{A}_k$-computation oracle

In this section, we construct an efficient approximate separation oracle (see Definition 23) for the set C_τ over the interval $J=[-1,1]$. We denote our algorithm by APPROXSEPORACLE. Let A be the ball defined in Lemma 27. We will show:

Theorem 31. *For all $\mu>0$,* APPROXSEPORACLE(c,μ) *is a μ-approximate separation oracle for C_τ that runs in time $\widetilde{O}(dk+d\log\log\frac{1}{\mu}+s)$, where s the number of samples in J, assuming all queries are contained in the ball A.*

Along the way we also develop an approximate $\mathcal{A}_k$-computation oracle COMPUTEAK.

6.1 Overview of APPROXSEPORACLE

APPROXSEPORACLE consists of two parts, TESTNONNEGBOUNDED and AKSEPARATOR. We show:

Lemma 32. *For any $\tau\le 2$, given a set polynomial coefficients $c\in A\subset\mathbb{R}^{d+1}$, the algorithm* TESTNONNEGBOUNDED(c,μ) *runs in time $O(d\log^2 d(\log^2 d+\log\log 1/\mu))$ and outputs a separating hyperplane for $\mathcal{C}_\tau$ or "yes". Moreover, if there exists a point $x\in[-1,1]$ such that $p_c(x)<-\mu$, the output is always a separating hyperplane.*

We show in the next section that whenever $c\notin\mathcal{C}_\tau$ the output is "yes".

Theorem 33. *Given a set of polynomial coefficients $c\in A\subset\mathbb{R}^{d+1}$ such that $p_c(x)\ge-\mu$ for all $x\in[-1,1]$, there is an algorithm* AKSEPARATOR(c,μ) *that runs in time $O(dk+(s+d)\log^2(s+d))$ and either outputs a separating hyperplane for c from $\mathcal{C}_\tau$ or returns "yes". Moreover, if $\|p_c-\widehat{f}_E\|_{\mathcal{A}_k}>\tau+2\mu$, the output is always a separating hyperplane.*

APPROXSEPORACLE given TESTNONNEGBOUNDED and AKSEPARATOR Given TESTNONNEGBOUNDED and AKSEPARATOR, it is straightforward to design APPROXSEPORACLE.

We first run TESTNONNEGBOUNDED(c, μ). If it outputs a separating hyperplane, we return the hyperplane. Otherwise, we run AKSEPARATOR(c, μ), and again if it outputs a separating hyperplane, we return it. If none of these happen, we return "yes". Lemma 32 and Theorem 33 imply that APPROXSEPORACLE is correct and runs in the claimed time:

$$O(d \log^2 d(\log^2 d + \log\log 1/\mu)) \ + \ O(dk + (s+d)\log^2(s+d)) \ = \ \widetilde{O}(dk + d\log\log 1/\mu + s) \ .$$

In the following sections, we prove Lemma 32 and Theorem 33. In Section 6.2 we describe TESTNONNEGBOUNDED and prove Lemma 32, and in Section 6.3 we describe AKSEPARATOR and prove Theorem 33.

6.2 Testing non-negativity and boundedness

Formally, the problem we solve here is the following testing problem:

Definition 34 (Approximate non-negativity test)**.** *An approximate non-negativity tester is an algorithm satisfying the following guarantee. Given a polynomial $p = \sum_{i=0}^{d} c_i x^i$ with $\max_i |c_i| \le \alpha$ and a parameter $\mu > 0$, return one of two results:*

- *a point $x \in [-1, 1]$ at which $p(x) < -\mu/2$.*
- *"OK".*

Moreover, it must return the first if there exists a point $x' \in [-1, 1]$ so that $p(x') < -\mu$.

Building upon the classical polynomial root-finding results of [Pan01], we show:

Theorem 35. *Consider p and μ from Definition 34. Then there exists an algorithm* TESTNONNEG(p, μ) *that is an approximate non-negativity tester and runs in time*

$$O(d \log^2 d \cdot (\log^2 d + \log\log \alpha + \log\log(1/\mu))) \ ,$$

where α is a bound on the coefficients of p.

This theorem is proved in Section B.2.

We have a bound on the coefficients c since we may assume that $c \in A$, and so we can use this algorithm to efficiently test non-negativity as we require. Our algorithm TESTNONNEGBOUNDED simply runs TESTNONNEG(p_c, μ). If this returns "yes", then TESTNONNEGBOUNDED outputs "yes". Otherwise, TESTNONNEG(p_c, μ) outputs a point $x \in [-1, 1]$ such that $p_c(x) \le -\mu/2$. In that case, TESTNONNEGBOUNDED returns the hyperplane defined by $y = -(1, x, x^2, \ldots, x^d)^T$, i.e., $p_c(x) = -y^T c$. Note that for all $c' \in \mathcal{C}_\tau$ we have $p_{c'}(y) \ge 0$ and hence $y^T c' \le 0$. This shows that

$$y^T c' \ \le \ 0 \ < \ \frac{\mu}{2} \ \le \ -p_c(x) \ = \ y^T c$$

as desired.

Proof of Lemma 32. The correctness of this algorithm follows from the correctness of TESTNONNEG. We therefore only bound the running time. The worst-case runtime of this algorithm is exactly the runtime of TESTNONNEG(p_c, μ) for any $c \in A$. Since we run TESTNONNEG(p_c, μ) only when $\max_{i\in[d]} |c_i| \le 2^{L_u} = 2^{O(d)}$ (see Theorem 27) , the runtime of TESTNONNEG(p_c, μ) is

$$O(d \log^2 d(\log^2 d + \log\log 1/\mu)),$$

as claimed. □

6.3 An $\mathcal{A}_k$-computation oracle

We now consider the $\mathcal{A}_k$-distance computation between two functions, one of which is a polynomial and the other an empirical distribution. In this subsection, we describe an algorithm COMPUTEAK, and show:

Theorem 36. *Given a polynomial p such that $p(x) \ge -\mu$ for all $x \in [-1, 1]$ and an empirical distribution $\widehat{f}$ supported on s points, for any $k \le d$,* COMPUTEAK$(p, \widehat{f}, k)$ *runs in time $O((s + d)\log^2(s + d))$, and computes a value $v \in \mathbb{R}_+$ such that $|v - \|p - \widehat{f}\|_{\mathcal{A}_k}| \le 2\mu$ and a set of intervals $I_1, \ldots, I_k$ so that*

$$\sum_{i=1}^{k} \left| p(I_i) - \widehat{f}(I_i) \right| = v \ .$$

Note that this theorem immediately implies Theorem 19.

AKSEPARATOR given COMPUTEAK: Before describing COMPUTEAK, we show how to design AKSEPARATOR satisfying Theorem 33 given such a subroutine COMPUTEAK.

The algorithm AKSEPARATOR is as follows: we run COMPUTEAK$(p_c, \widehat{f}, k)$, let v be its estimate for $\|p_c - \widehat{f}\|_{\mathcal{A}_k}$, and let $I_1, \ldots, I_k$ be the intervals it produces. If $v \le \tau$, we output "yes".

Otherwise, suppose

$$v = \sum_{i=1}^{k} |p_c(I_i) - \widehat{f}(I_i)| > \tau \ .$$

Note that if $\|p_c - \widehat{f}\|_{\mathcal{A}_k} > \tau + 2\mu$, this is guaranteed to happen since v differs from $\|p_c - \widehat{f}\|_{\mathcal{A}_k}$ by at most 2μ. Let $\sigma_i = \text{sign}(p_c(I_i) - \widehat{f}(I_i))$. Let $I_i = [a_i, b_i]$. Then

$$\begin{aligned}\sum_{i=1}^{k} |p_c(I_i) - \widehat{f}(I_i)| &= \sum_{i=1}^{k} \sigma_i \left(\int_{a_i}^{b_i} p_c(x)\,\mathrm{d}x - \widehat{f}(I_i) \right) \\ &= \sum_{i=1}^{k} \sigma_i \left(\sum_{j=0}^{d} \frac{1}{j+1} \left(b_i^{j+1} - a_i^{j+1} \right) c_j - \widehat{f}(I_i) \right),\end{aligned}$$

and therefore,

$$\sum_{i=1}^{k} \sigma_i \sum_{j=0}^{d} \frac{1}{j+1} \left(b_i^{j+1} - a_i^{j+1} \right) c_j > \tau + \sum_{i=1}^{k} \sigma_i \widehat{f}(I_i) \ . \tag{11}$$

Note that the left hand side is linear in c when we fix σ_i, and this is the separating hyperplane AKSEPARATOR returns in this case.

Proof of Theorem 33 given Theorem 36. We first argue about the correctness of the algorithm. If $\|p_c - \widehat{f}\|_{\mathcal{A}_k} \geq \tau + 2\mu$, then COMPUTEAK guarantees that

$$v = \sum_{i=1}^{i} |p_c(I_i) - \widehat{f}(I_i)| > \tau \ .$$

Consider the hyperplane constructed in (11). For any $c' \in \mathcal{C}_\tau$

$$\begin{aligned}\sum_{i=1}^{k} \sigma_i \left(\sum_{j=0}^{d} \frac{1}{j+1} \left(b_i^{j+1} - a_i^{j+1} \right) c'_j - \widehat{f}(I_i) \right) &\leq \sum_{i=1}^{k} \left| \sum_{j=0}^{d} \frac{1}{j+1} \left(b_i^{j+1} - a_i^{j+1} \right) c'_j - \widehat{f}(I_i) \right| \\ &= \sum_{j=0}^{d} \left| p_{c'}(I_j) - \widehat{f}(I_i) \right| \\ &\leq |p_{c'} - \widehat{f}\|_{\mathcal{A}_k} \leq \tau \ ,\end{aligned}$$

where the last inequality is from the definition of $\mathcal{C}_\tau$. Therefore this is indeed a separating hyperplane for c and $\mathcal{C}_\tau$. Moreover, given $I_1, \ldots, I_k$ and v, this separating hyperplane can be computed in time $O(dk)$. Thus the entire algorithm runs in time $O(dk + (s+d)\log^2(s+d)$ as claimed. □

6.3.1 A Reduction from Continuous to Discrete

We first show that our $\mathcal{A}_k$–computation problem reduces to the following discrete problem: For a sequence of real numbers $c_1, \ldots, c_r$ and an interval $I = [a, b]$ in $[r]$, let $w(I) = \sum_{a \leq i \leq b} c_i$. We show that our problem reduces to the problem DISCRETEAK, defined below.

DISCRETEAK: Given a sequence of r real numbers $\{c_i\}_{i=1}^r$ and a number k, find a set of k disjoint intervals $I_1, \ldots, I_k$ that maximizes

$$\sum_{i=1}^{k} |w(I_i)| \ .$$

We will denote the maximum value obtainable $\|\{c_i\}\|_{\mathcal{A}_k}$, i.e.,

$$\|\{c_i\}\|_{\mathcal{A}_k} = \max_{\mathcal{I}} \sum_{I \in \mathcal{I}} |w(I)| \ ,$$

where the $\mathcal{I}$ is taken over all collections of k disjoint intervals.

We will show that it is possible to reduce the continuous problem of approximately computing the $\mathcal{A}_k$ distance between p and $\widehat{f}$ to solving DISCRETEAK for a suitably chosen sequence of length $O(d)$. Suppose the empirical distribution $\widehat{f}$ is supported at s points $a < x_1 \leq \ldots \leq x_s \leq b$ in this interval. Let $\mathcal{X}$ be the support of $\widehat{f}$. Let $p[\alpha, \beta] = \int_\alpha^\beta p(x)dx$. Consider the following sequences of length $2s+1$:

$$E(i) = \begin{cases} 1/n & \text{if } i \text{ is even,} \\ 0 & \text{if } i \text{ is odd.} \end{cases} \ , \ P_{\text{disc}}(i) = \begin{cases} p[x_\ell, x_{\ell+1}] & \text{if } i = 2\ell+1, \\ 0 & \text{if } i \text{ is even.} \end{cases} \ ,$$

where for simplicity we let $s_0 = a$ and $s_{s+1} = b$. The two sequences are displayed in Table 2.

Then we have the following lemma:

i	1	2	3	4	...	$2s$	$2s+1$
$E(i)$	0	$\frac{1}{n}$	0	$\frac{1}{n}$	...	$\frac{1}{n}$	0
$P_{\text{disc}}(i)$	$p[a, x_1]$	0	$p[x_1, x_2]$	0	...	0	$p[x_s, b]$

Table 2: The sequences $E(i)$ and $P_{\text{disc}}(i)$.

Lemma 37. *For any polynomial p so that $p(x) \geq -\mu$ on $[-1, 1]$*

$$\left| \|p - \widehat{f}\|_{\mathcal{A}_k} - \|\{P_{\text{disc}} - E\}\|_{\mathcal{A}_k} \right| < 2\mu \ .$$

Moreover, given k intervals $I_1, \ldots, I_k$ maximizing $\|\{P_{\text{disc}} - E\}\|_{\mathcal{A}_k}$, one can compute k intervals $J_1, \ldots, J_k$ so that

$$\left| \sum_{i=1}^{k} \left| p(J_i) - \widehat{f}(J_i) \right| - \|\{P_{\text{disc}} - E\}\|_{\mathcal{A}_k} \right| < 2\mu$$

in time $O(k)$.

Proof. We first show that

$$\|p - \widehat{f}\|_{\mathcal{A}_k} \geq \|\{P_{\text{disc}} - E\}\|_{\mathcal{A}_k} \ .$$

Let $I_1, \ldots, I_k$ be a set of disjoint intervals in $[2d + 1]$ achieving the maximum on the RHS. Then it suffices to demonstrate a set of k disjoint intervals $J_1, \ldots, J_k$ in I satisfying

$$\sum_{i=1}^{k} \left| p(J_i) - \widehat{f}(J_i) \right| \geq \sum_{i=1}^{k} |P_{\text{disc}}(I_i) - E(I_i)| \ . \tag{12}$$

We construct the J_i as follows. Fix i, and let $I_i = [a_i, b_i]$. Define J_i to be the interval from a_i to b_i. If a_i is even (i.e., if $P_{\text{disc}}(a_i) - E(i)$ has only a contribution from $-E(a_i)$), include the left endpoint of this interval from J_i, otherwise (i.e., if $P_{\text{disc}}(a_i) - E(i)$ has only a contribution from $P_{\text{disc}}(a_i)$), exclude it, and similarly for the right endpoint. Then, by observation, we have $P_{\text{disc}}(I_i) - E(I_i) = p(J_i) - \widehat{f}(J_i)$, and thus this choice of J_i satisfies (12), as claimed.

Now we show the other direction, i.e., that

$$\|p - \widehat{f}\|_{\mathcal{A}_k} \leq \|\{P_{\text{disc}} - E\}\|_{\mathcal{A}_k} + 2\mu \ .$$

Let $I_1, \ldots, I_k$ denote a set of disjoint intervals in I achieving the maximum value on the LHS. It suffices to demonstrate a set of k disjoint intervals $J_1, \ldots, J_k$ in I satisfying

$$\sum_{i=1}^{k} \left| p(I_i) - \widehat{f}(I_i) \right| \leq \sum_{i=1}^{k} |P_{\text{disc}}(J_i) - E(J_i)| + 2\mu \ . \tag{13}$$

We first claim that we may assume that the endpoints of each I_i are at a point in the support of the empirical. Let a_i and b_i be the left and right endpoints of I_i, respectively. Cluster the intervals I_i into groups, as follows: cluster any set of consecutive intervals $I_j, \ldots, I_{j'}$ if it is the case that $p(I_\ell) - \widehat{f}(I_\ell) \geq 0$ for $\ell = j, \ldots, j'$, and $[b_\ell, a_{\ell+1}]$ contains no points of the empirical, for $\ell = j, \ldots, j' - 1$. Put all other

intervals not clustered this way in their own group. That is, cluster a set of consecutive intervals if and only if on all of them the contribution to the LHS is non-negative, and there are no points of the empirical between them. Let the clustering be $\mathcal{I}_1, \ldots, \mathcal{I}_{k'}$, and let J_i be the smallest interval containing all the intervals in $\mathcal{I}_j$. Let c_i and d_i denote the left and right endpoints of J_i, respectively. Associate to each cluster a sign $\sigma_i \in \{-1, 1\}$ which is the (unique) sign of $p(I) - \widehat{f}(I)$ for all $I \in \mathcal{I}_j$. Since $p(i) \geq -\mu$, this clustering has the property that for any cluster $\mathcal{I}_i$, we have

$$\left| \left(\sum_{I \in \mathcal{I}_i} p(I) - \widehat{f}(I) \right) - (p(J_i) - \widehat{f}(J_i)) \right| \leq \mu \cdot |J_i - \bigcup_{I \in \mathcal{I}_j} I| \ .$$

Then, for all i, if $\sigma_i = 1$, take the interval $I_i' = (x_j, x_\ell)$ where x_j is the largest point in $\mathcal{X}$ so that $x_j \leq c_i$, and where x_ℓ is the smallest point in $\mathcal{X}$ so that $x_\ell \geq d_i$. Then since $p \geq \mu$ on $[-1, 1]$ and the new interval contains no points in the support of $\widehat{f}$ which are not in $\cup_{I \in \mathcal{I}_j} I$ or J_i, we have

$$p(I_i') - \widehat{f}(I_i') \geq p(J_i) - \widehat{f}(I_i') - \mu \left| I_i' - J_i \right| \geq \left(\sum_{I \in \mathcal{I}_i} p(I) - \widehat{f}(I) \right) - \mu |J_i - \cup_{I \in \mathcal{I}_j} I| - \mu \left| I_i' - J_i \right| \ .$$

Alternatively, if $\sigma_i < 0$, take the interval $I_i' = [x_j, x_\ell]$ where x_j is the smallest point in $\mathcal{X}$ so that $x_j \geq c_i$ and x_ℓ is the largest point in $\mathcal{X}$ so that $x_\ell \leq d_i$. By the analogous reasoning as before we have that $p(I_i') - \widehat{f}(I_i') \leq p(J_i) - \widehat{f}(J_i) + \mu |J_i|$,[5] and therefore $|p(I_i') - \widehat{f}(I_i')| + \mu |I_i| \geq |p(J_i) - \widehat{f}(J_i)|$. Thus,

$$\begin{aligned} \sum_{i=1}^{k} \left| p(I_i) - \widehat{f}(I_i) \right| &\leq \sum_{i=1}^{k'} \left(|p(I_i') - \widehat{f}(I_i')| + \mu |J_i - \cup_{I \in \mathcal{I}_j} I| + \mu \left| I_i' - J_i \right| \right) \\ &\leq \sum_{i=1}^{k'} \left| p(I_i') - \widehat{f}(I_i') \right| + 2\mu \ . \end{aligned}$$

since $\sum_{i=1}^{k'} \left(|J_i - \cup_{I \in \mathcal{I}_j} I + \mu \left| I_i' - J_i \right| \right) \leq 2$ as the intervals in the sum are disjoint subintervals in $[-1, 1]$.

Now it is straightforward to define the J_i. Namely, for each I_i with endpoints $x_{i_1} \leq x_{i_2}$ so that $x_{i_1}, x_{i_2} \in \mathcal{X}$, define

$$J_i = \begin{cases} [i_1, i_2] & \text{if } x_{i_1}, x_{i_2} \in \mathcal{X}; \\ [i_1 + 1, i_2] & \text{if } x_{i_1} \notin \mathcal{X} \text{ and } x_{i_2} \in \mathcal{X}; \\ [i_1, i_2 - 1] & \text{if } x_{i_1} \in \mathcal{X} \text{ and } x_{i_2} \notin \mathcal{X}; \\ [i_1 + 1, i_2 - 1] & \text{if } x_{i_1}, x_{i_2} \notin \mathcal{X} \ . \end{cases}$$

One can check that with this definition of the J_i, we have $p(I_i) - \widehat{f}(I_i) = P_{\text{disc}}(J_i) - E(J_i)$; moreover, all the J_i are discrete and thus this choice of J_i satisfies (6.3.1).

Moreover, the transformation claimed in the lemma is the transformation provided in the first part of the argument. It is clear that this transformation is computable in a single pass through the intervals $I_1, \ldots, I_k$. This completes the proof. □

[5] Since each cluster with negative sign has exactly a single interval in the original partition, notationally we will not distinguish between J_i and the one interval in the original partition in $\mathcal{I}_i$, when $\sigma_i = -1$.

6.3.2 Description of COMPUTEDISCRETEAK

For the rest of this section we focus on solving DISCRETEAK. A very similar problem was considered in [Csu04] who showed an algorithm for the problem of computing the set of k disjoint intervals $I_1, \ldots, I_k$ maximizing

$$\left| \sum_{i=1}^{k} w(I_i) \right|$$

which runs in time $O(d \cdot \min\{\log d, k\})$ time. We require a modified version of this algorithm which we present and analyze below. We call our variant COMPUTEDISCRETEAK.

Here is an informal description of COMPUTEDISCRETEAK. First, we may assume the original sequence is alternating in sign, as otherwise we may merge two consecutive numbers without consequence. We start with the set of intervals $\mathcal{I}_0 = I_{0,1} \leq \ldots \leq I_{0,r}$, where $I_{0,i} = [c_i, c_i]$ contains only the point c_i. We first compute $\mathcal{J}_0$ and m_0, where $\mathcal{J}_0$ is the set of k intervals I in $\mathcal{I}_0$ with largest $|w(I)|$, and $m_0 = \sum_{I \in \mathcal{J}_0} |w(I)|$. Iteratively, after constructing $\mathcal{I}_i = \{I_{i,1}, \ldots, I_{i,r}\}$, we construct $\mathcal{I}_{i+1}$ by finding the set $I_{i,j}$ with minimal $|w(I_{i,j})|$ amongst all intervals in $\mathcal{I}_i$, and merging it with both of its neighbors (if it is the first or last interval and so only has one neighbor, instead discard it), that is,

$$\mathcal{I}_{i+1} = \{I_{i,1}, \ldots, I_{i,j-2}, I_{i,j-1} \cup I_{i,j} \cup I_{i,j+1}, I_{i,j+2}, \ldots, I_{i,r_i}\} .$$

We then compute $\mathcal{J}_{i+1}$ and m_{i+1} where $\mathcal{J}_{i+1}$ is the set of k intervals I in $\mathcal{I}_{i+1}$ with largest $|w(I)|$, and $m_{i+1} = \sum_{I \in \mathcal{J}_{i+1}} |w(I)|$. To perform these operations efficiently, we store the weights of the intervals we create in priority queues. We repeat this process until the collection of intervals $\mathcal{I}_\ell$ has $\leq k$ intervals. We output $\mathcal{J}_i$ and w_i, where w_i is the largest amongst all $w_{i'}$ computed in any iteration. An example of an iteration of the algorithm is given in Figure 1, and the formal definition of the algorithm is in Algorithm 4.

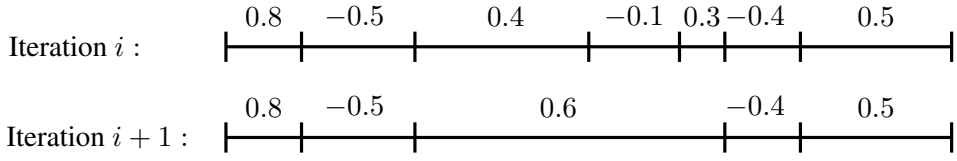

Figure 1: An iteration of COMPUTEDISCRETEAK. The numbers denote the weight of each interval. The interval with smallest weight (in absolute value) is chosen and merged with adjacent intervals. Note that if weights are of alternating signs at the start, then they are of alternating signs at each iteration.

The following runtime bound can be easily verified:

Theorem 38. *Given* $\{c_i\}_{i=1}^r$, COMPUTEDISCRETEAK$(\{c_i\}_{i=1}^r, k)$ *runs in time* $O(r \cdot \min\{\log r, k\})$.

The nontrivial part of the analysis is correctness.

Theorem 39. *Given* $\{c_i\}_{i=1}^r$ *and* k*, the set of intervals returned by* COMPUTEDISCRETEAK$(\{c_i\}_{i=1}^r, k)$ *solves the problem* DISCRETEAK.

Proof. Our analysis follows the analysis in [Csu04]. We call any $\mathcal{I}^*$ which attains the maximum for the DISCRETEAK problem a *maximal subset*, or *maximal* for short. For any two collections of disjoint intervals $\mathcal{I}', \mathcal{I}''$ in $[r]$, we say that $\mathcal{I}'$ *is contained in* $\mathcal{I}''$ if all the boundary points of intervals in $\mathcal{I}'$ are also boundary

Algorithm 4 Computing the discrete $\mathcal{A}_k$ norm of a sequence.

function COMPUTEDISCRETEAK($\{c_i\}_{i=1}^r, k$)
  Let $\mathcal{I}_0 \leftarrow \{[c_1, c_1], [c_2, c_2], \ldots, [c_r, c_r]\}$ be the initial set of intervals.
  Let Q be an empty priority queue.
  for $I \in \mathcal{I}_0$ **do**
    $Q.push(I, w(I))$
  end for
  $i \leftarrow 0$
  while $|\mathcal{I}_i| > k$ **do**
    Let $I \leftarrow Q.deleteMin()$.
    if I is not the leftmost or rightmost interval **then**
      Let I_{left} and I_{right} be its left and right neighbors, respectively.
      $Q.remove(I_{left})$
      $Q.remove(I_{right})$
      Let $I' = I_{left} \cup I \cup I_{right}$
      $Q.push(I', w(I'))$
    end if
    $i \leftarrow i + 1$
    Let $\mathcal{I}_i$ be the elements of Q
    Let $\mathcal{J}_i$ be the k intervals in $\mathcal{I}_i$ with maximum weight
    Let $w_i = \sum_{I \in \mathcal{J}_i} w(I)$
  end while
  return w_j and $\mathcal{J}_j$ where w_j satisfies $w_j \geq w_i$ for all i.
end function

points of intervals in $\mathcal{I}''$. Figure 2 shows an example of two collections of intervals, one contained in the other. If there is a maximal $\mathcal{I}^*$ that is contained in $\mathcal{I}$ we say that $\mathcal{I}$ contains a maximal subset. We say that $\mathcal{I}'$ is *atomic* with respect to $\mathcal{I}''$ if every interval in $\mathcal{I}'$ is also in $\mathcal{I}''$. Figure 3 gives an example of two collections of intervals, one atomic with respect to the other. If there is a maximal $\mathcal{I}^*$ that is atomic with respect to $\mathcal{I}$ then we say that the maximum is atomic with respect to $\mathcal{I}$.

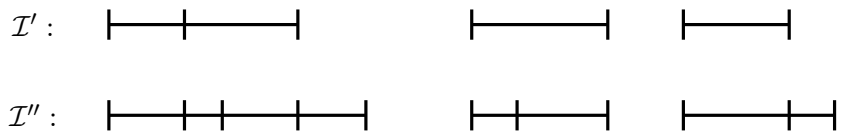

Figure 2: $\mathcal{I}'$ is contained in $\mathcal{I}''$ since each boundary point of all intervals in $\mathcal{I}'$ are boundary points of some interval in $\mathcal{I}''$.

We will prove the following invariant of our algorithm:

Lemma 40. *For any $i \geq 0$, if $\mathcal{I}_i$ contains a maximal subset, then either the maximum is atomic with respect to $\mathcal{I}_i$ or $\mathcal{I}_{i+1}$ contains a maximal subset.*

Before we prove this lemma, let us see how it suffices to prove Theorem 39. Now the set $\mathcal{I}_0$ contains a maximal subset. By induction and Lemma 40, for all i, as long as the maximum is not atomic with respect

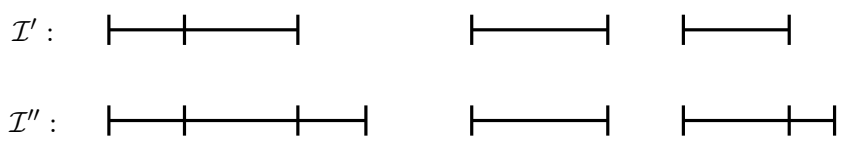

Figure 3: $\mathcal{I}'$ is atomic with respect to $\mathcal{I}''$, since each interval in $\mathcal{I}'$ is also an interval in $\mathcal{I}''$.

to $\mathcal{I}_i$, $\mathcal{I}_{i+1}$ contains a maximal subset. COMPUTEDISCRETEAK stops iterating at iteration i_f if $\mathcal{I}_{i_f}$ has at most k intervals. At this point either the maximum was atomic with respect to some $\mathcal{I}_i$, or $\mathcal{I}_{i_f}$ contains a maximal subset. Let $\mathcal{I}^*$ be any maximal subset it contains. We observe that

$$\sum_{I \in \mathcal{I}^*} |w(I)| \leq \sum_{I \in \mathcal{I}_{i_f}} |w(I)| \ ,$$

and moreover, $\mathcal{I}_{i_f}$ has k pieces, so $\mathcal{I}_{i_f}$ is itself maximal, and is atomic with respect to itself.

Thus, there is some i so that $\mathcal{I}_i$ contains a maximal subset that is atomic with respect to $\mathcal{I}_i$. Call this maximal subset $\mathcal{I}^*$. But then since it is atomic with respect to $\mathcal{I}_i$, we have that

$$\sum_{I \in \mathcal{I}^*} |w(I)| \leq \sum_{I \in \mathcal{J}_i} |w(I)| = m_i \ ,$$

since $\mathcal{J}_i$ is chosen to maximize the sum over all sets of k intervals which are atomic with respect to $\mathcal{I}_i$. Since $\mathcal{I}^*$ achieves the maximum for DISCRETEAK, we conclude that m_i is indeed the maximum. Thus whatever $m_{i'}$ we output is also the maximum, and its $\mathcal{J}_{i'}$ attains the maximum. This completes the proof of Theorem 39 assuming Lemma 40. □

We now prove Lemma 40.

Proof of Lemma 40. It suffices to show that if $\mathcal{I}_i$ contains a maximal subset, but the maximum is not atomic with respect to $\mathcal{I}_i$, then $\mathcal{I}_{i+1}$ also contains a maximal subset. Thus, let $\mathcal{I}^*$ be such that

1. $\mathcal{I}^*$ is maximal
2. $\mathcal{I}^*$ is contained in $\mathcal{I}_i$, and
3. there is no $\mathcal{I}_1^* \neq \mathcal{I}^*$ satisfying conditions (1) and (2) so that every interval in $\mathcal{I}_1^*$ is contained in some interval in $\mathcal{I}^*$.

Such an $\mathcal{I}^*$ clearly exists by the assumption on $\mathcal{I}_i$. Note that $\mathcal{I}^*$ cannot be atomic with respect to $\mathcal{I}_i$. By observation we may assume that no interval I' in a maximal subset will ever end on a point a so that $w(I')$ and c_a have different signs, since otherwise we can easily modify the partition to have this property while still maintaining properties (1)-(3). More generally, we may assume there does not exist an interval I'' contained in I' with right endpoint equal to I''s right endpoint (resp. left endpoint equal to I''s left endpoint) so that $w(I')$ and $w(I'')$ have different signs.

Let $I_j = [\beta_2, \beta_3]$ be the interval in $\mathcal{I}_i$ with minimal $|w(I)|$ amongst all $I \in \mathcal{I}_i$. WLOG assume that it is not the leftmost or rightmost interval (the analysis for these cases is almost identical and so we omit it). Let β_1 be the left endpoint of I_{j-1} and β_4 be the right endpoint of I_{j+1}. WLOG assume that $w(I_j) < 0$.

The initial partition $\mathcal{I}_0$ had the property that the signs of the values $w(I)$ for $I \in \mathcal{I}_0$ alternated, and through a simple inductive argument it follows that for all $\mathcal{I}_i$, the signs of the values $w(I)$ for $I \in \mathcal{I}_i$ still alternate. Thus, we have $w(I_{j-1}), w(I_{j+1}) \geq 0$. Since $\mathcal{I}^*$ is not atomic, there is some $I_a \in \mathcal{I}^*$ which contains at least two intervals I_1, I_2 of $\mathcal{I}_i$. Moreover, since the signs of the $w(I)$ of the intervals in $\mathcal{I}_i$ alternate, we may assume that $w(I_1)$ and $w(I_2)$ have different signs. Thus, by observation, we may in fact assume that I_a contains three consecutive intervals $I_1 < I_2 < I_3$, and that $w(I_a), w(I_1), w(I_3)$ have the same sign, and $w(I_2)$ has a different sign. Moreover, $|w(I_j)| \leq \min\{w(I_1), w(I_2), w(I_3)\}$. Moreover, define I_a^1 to be the interval which shares a left endpoint with I_a, and which has right endpoint the right endpoint of I_1, and I_a^2 to be the interval which shares a right endpoint with I_a, and which has left endpoint the left endpoint of I_3 (See Figure 4).

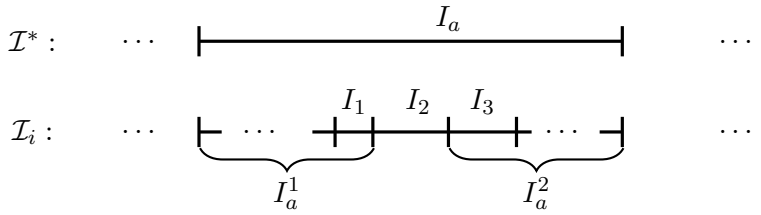

Figure 4: The interval I_a, and the intervals I_1, I_2, and I_3.

We must have that $w(I_a^1), w(I_a^2)$ are the same sign as $w(I_a)$ as otherwise, say if $w(I_a^1)$'s sign was different from $w(I_a)$'s sign, we would have $|w(I_a)| \leq |w(I_a^2)|$ and so the existence of the collection of intervals $\mathcal{I}' = (\mathcal{I}^* \setminus I_a) \cup I_a^2$ violates condition (3), since it is contained in $\mathcal{I}_i$, and

$$\sum_{I \in \mathcal{I}'} |w(I)| = \sum_{I \in \mathcal{I}^*} |w(I)| - |w(I_a)| + |w(I_a^2)| \geq \sum_{I \in \mathcal{I}^*} |w(I)| \,,$$

so it is maximal.

Since $\mathcal{I}^*$ is contained in $\mathcal{I}_i$, the only boundary points that intervals in $\mathcal{I}_i$ can have in the interval $[\beta_1, \beta_4]$ are at the points β_i for $i \in \{1, 2, 3, 4\}$. There are a few cases.

Case 1 If no interval in $\mathcal{I}^*$ has any boundary point at β_2 or β_3, then it is still contained in $\mathcal{I}_{i+1}$, by the definition of $\mathcal{I}_{i+1}$.

Case 2 If $[\beta_2, \beta_3] \in \mathcal{I}^*$, define $\mathcal{I}' = (\mathcal{I}^* \setminus \{[\beta_2, \beta_3], I_a\}) \cup \{I_a^1, I_a^2\}$. Then

$$\sum_{I \in \mathcal{I}'} |w(I)| = \sum_{I \in \mathcal{I}^*} |w(I)| - |w(I_j)| + |w(I_2)| \geq \sum_{I \in \mathcal{I}^*} |w(I)|$$

by the choice of I_j, so $\mathcal{I}'$ is maximal, contained in $\mathcal{I}_i$, and thus its existence violates condition (3), so this case is impossible. This is illustrated in Figure 5, where for simplicity I_a contains precisely three intervals.

Case 3 If β_3 is the right endpoint of some interval $I \in \mathcal{I}^*$, then by the same reasoning as before, we may assume that $w(I) < 0$. Then, let I' be the interval with the same left endpoint as I but with right endpoint β_1. Since then $w(I') = w(I) - w(I_{j-1}) - w(I_j) \leq w(I)$, the partition $\mathcal{I}' = \mathcal{I}^* \setminus I \cup I'$ is maximal, contained in $\mathcal{I}_i$, and its existence again violates condition (3), so this case is impossible. An illustration is given in Figure 6.

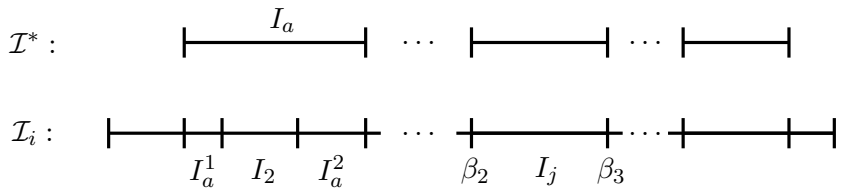

Figure 5: When I_j is an element of $\mathcal{I}^*$, we can drop it and add the intervals I_a^1 and I_a^2 achieving a larger weight.

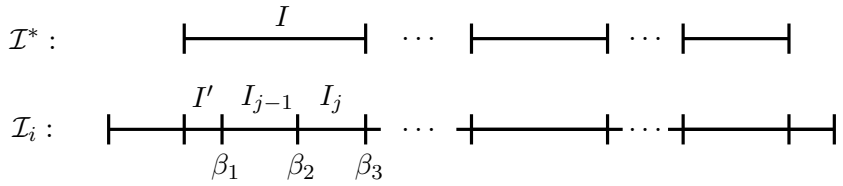

Figure 6: $\mathcal{I}^*$ can drop I and instead take I' to get a larger weight.

Case 4 If β_2 is the left endpoint of some interval $I \in \mathcal{I}^*$, then analogous reasoning to that in Case 3 results in a contradiction, so this case is also impossible.

Case 5a If β_2 is the right endpoint of some interval $I \in \mathcal{I}^*$, and no interval in $\mathcal{I}^*$ contains I_{j+1}, then we know that $w(I) \geq 0$. Let I' be the interval $I \cup I_j \cup I_{j+1}$. Then, the partition $\mathcal{I}' = \mathcal{I}^* \setminus I \cup I'$ is maximal by the same kind of reasoning as before, and $\mathcal{I}'$ is contained in $\mathcal{I}_{j+1}$. Thus, this case is possible and consistent with the Lemma.

Case 5b If β_2 is the right endpoint of some interval $I \in \mathcal{I}^*$ and β_3 is the left endpoint of some interval $I' \in \mathcal{I}^*$, then we know that $w(I), w(I') \geq 0$. Let $I'' = I \cup I_j \cup I'$. Define $\mathcal{I}' = (\mathcal{I}^* \setminus \{I, I', I_a\}) \cup \{I'', I_a^1, I_a^2\}$. Then,

$$\sum_{I \in \mathcal{I}'} |w(I)| = \sum_{I \in \mathcal{I}^*} |w(I)| - |w(I_j)| + |w(I_2)| \geq \sum_{I \in \mathcal{I}^*} |w(I)| \,,$$

so again this is a maximal subset which is now contained in $\mathcal{I}_{j+1}$.

Case 6 If β_3 is the left endpoint of some interval in $\mathcal{I}^*$, by analogous reasoning to that in Cases 5a and 5b, we may conclude that in this case, the Lemma holds.

These cases encompass all possible cases, and thus we conclude that the Lemma holds, as claimed. □

6.3.3 Description of COMPUTEAK

Our algorithm COMPUTEAK uses Fact 41 below, produces the sequence $P_{\text{disc}}(i) - E(i)$, and computes $\|\{P_{\text{disc}} - E\}\|_{\mathcal{A}_k}$ using COMPUTEDISCRETEAK.

It thus suffices to show that we can construct this sequence $P_{\text{disc}}(i) - E(i)$ efficiently when we are given the empirical distribution and the polynomial p. The only difficulty lies in efficiently computing the $p[x_i, x_{i+1}]$. This problem is equivalent to efficiently evaluating the integral of p at all the points in $\mathcal{X}$, which

is in turn equivalent to efficiently evaluating a degree $d+1$ polynomial at s points. To do so, we use the following well-known fact:

Fact 41 ([VZGG13], p. 299 and p. 245)**.** *Let $x_1, \ldots, x_s$ be a set of s real numbers and let p be a polynomial of degree at most s. Then there is an algorithm that computes $p(x_1), \ldots, p(x_s)$ in time $O(s \log^2 s)$.*

After solving the discretized version of the problem, the algorithm outputs the estimate that COMPUTEDISCRETEAK computed and the processed version of the intervals, where the processing is the one described in Lemma 37. Thus, we have:

Proof of Theorem 36. The correctness of the algorithm follows from Lemma 37 and the arguments given above. Thus it suffices to bound the running time. The time required to produce the sequence $P_{\mathrm{disc}}(i) - E(i)$ is bounded by computing the $p[x_i, x_{i+1}]$, which can be done in time $O((s+d)\log^2(s+d))$ by Fact 41. Moreover, the running time of COMPUTEDISCRETEAK on the sequence $P_{\mathrm{disc}}(i) - E(i)$ is $O(s \log s)$. Hence, the running time of the overall algorithm is $O((s+d)\log^2(s+d))$, as claimed. □

7 Applications

In this section, we apply our main result to obtain near optimal estimators for various classes of structured distributions. As described in Table 1, we consider arbitrary mixtures of well-studied distribution families, including log-concave distributions, normal distributions, densities with bounded number of modes, and density functions in Besov spaces. We also consider mixtures of discrete structured distributions over an ordered domain, such as multi-modal distributions, monotone hazard rate distributions, Poisson, and Binomial distributions. For all these classes, our sample complexity and running time match the information-theoretic optimum, up to at most logarithmic factors.

We note that even though our algorithm is stated for distributions over a known finite interval, they are also applicable to distributions over the entire real line, such as (mixtures of) Gaussians or Poisson distributions. This follows from the following fact: let $x_{\min}$ and $x_{\max}$ be the smallest and largest elements among $\frac{\log(1/\delta)}{\epsilon^2}$ draws from any distribution. Then with probability at least $1-\delta$, the distribution assigns probability mass at least $1-\epsilon$ to the interval $[x_{\min}, x_{\max}]$. Thus, at a cost of $\frac{\log(1/\delta)}{\epsilon^2}$ samples, we may truncate the distribution and thereafter only consider this finite interval.

7.1 Mixture of log-concave distributions

For an interval $I \subseteq \mathbb{R}$, a function $g : I \to \mathbb{R}$ is called *concave* if for any $x, y \in I$ and $\lambda \in [0,1]$ it holds $g(\lambda x + (1-\lambda)y) \geq \lambda g(x) + (1-\lambda)g(y)$. A function $h : I \to \mathbb{R}_+$ is called *log-concave* if $h(x) = \exp(g(x))$, where $g : I \to \mathbb{R}$ is concave. A density f is a k-mixture of log-concave density functions if there exist $w_1, \ldots, w_k \geq 0$, $\sum_i w_i = 1$ and log-concave density functions $f_1, \ldots, f_k$ such that $f = \sum w_i f_i$. The class of log concave distributions is very broad and contains the class of Gaussians, uniform, exponential, Gamma, Beta, and Weibull distributions. Log-concave distributions have received significant interest in economics and statistics [BB05, CSS10, DR09, DW13, CS13, KS14, BD14, HW15].

It was shown in [CDSS14a] that a k-mixture of log-concave density functions can be ϵ-approximated in L_1-norm by a t-piecewise linear density, for $t = \widetilde{O}(k/\sqrt{\epsilon})$. Using this structural result, [CDSS14a] gave a polynomial time algorithm with sample complexity $\widetilde{O}(t/\epsilon^2) = \widetilde{O}(k/\epsilon^{5/2})$ to agnostically learn a k-mixture of log-concave distributions. This sample bound is nearly optimal, as $\Omega(k/\epsilon^{5/2})$ samples are necessary for this learning problem.

Our main result yields a sample optimal and nearly-linear time algorithm for this problem. In particular, this follows from a combination of Theorem 1 and a recently obtained tight structural result that removes the logarithmic factors from the previous construction of [CDSS14a]. In particular, it is shown in [DK15] that a k-mixture of log-concave density functions can be ϵ-approximated in L_1-norm by a t-piecewise linear density, for $t = O(k/\sqrt{\epsilon})$. As a corollary, we obtain the following:

Theorem 42. *There is an agnostic learning algorithm for the class of k-mixtures of log-concave distributions over the real line that uses $O(k/\epsilon^{5/2})$ samples and runs in time $\widetilde{O}(k/\epsilon^{5/2})$.*

7.2 Mixture of Gaussians

Let $N(\mu, \sigma^2)$ denote the normal distribution with mean μ and variance σ^2. A density $f : \mathbb{R} \to \mathbb{R}_+$ is a k-mixture of Gaussians if there exist $w_1, \ldots, w_k \geq 0$, $\sum_i w_i = 1$, $\mu_1, \ldots, \mu_k \in \mathbb{R}$, and $\sigma_1, \ldots, \sigma_k \in \mathbb{R}_+$ such that $f = \sum_{i=1}^k w_i N(\mu_i, \sigma_i^2)$.

In the theoretical computer science community, the problem of parameter estimation for Gaussian mixtures was initiated by [Das99]. Recent work has obtained polynomial sample and time algorithms for this problem under the conditions of identifiability [MV10, BS10]. We remark that learning the parameters of a mixture of k univariate Gaussians to accuracy ϵ requires $\Omega((1/\epsilon)^{6k-2})$ samples [HP15] .

The problem of proper learning for Gaussian mixtures has also been recently studied in [DK14, SOAJ14] who obtain algorithms that draw $\widetilde{O}(k/\epsilon^2)$ samples and run in time $O((1/\epsilon)^{3k-1})$. Another approach, due to [BSZ15], outputs a mixture of $O(k/\epsilon^3)$ Gaussians in time and sample complexity of $O(k/\epsilon^6)$.

It is well-known (see, e.g., [Tim63, Section 7.21] or [CDSS14a]) that a normal distribution is ϵ-close to a 3-piecewise polynomial of degree $O(\log(1/\epsilon))$. Using this structural result, [CDSS14a] obtain a nearly sample optimal and polynomial time agnostic learning algorithm for this problem.

As a corollary of Theorem 1, we obtain a nearly sample optimal and nearly-linear time algorithm. (The sample complexity of our algorithm is better than that of [CDSS14a] by logarithmic factors.) In particular:

Theorem 43. *There is an agnostic learning algorithm for k-mixtures of univariate Gaussians that draws $O((k/\epsilon^2)\log(1/\epsilon))$ samples and runs in time $\widetilde{O}(k/\epsilon^2)$.*

7.3 Densities in Besov spaces

Densities in Besov spaces constitute a broad family of distributions, including piecewise polynomials and the exponential family. Density estimation for functions in Besov spaces has received considerable attention in the statistics and information theory literature. A lot of the early work on the topic relied on wavelet techniques, based on the fact that functions in Besov spaces are amenable to multiscale decompositions [DeV98, DJKP96, DJ98].

A piecewise smooth density function f has the following decomposition,

$$f(x) = \sum_k c_{j_0,k}\phi_{j_0,k}(x) + \sum_{j=j_0}^{\infty}\sum_k d_{j_0,k}\psi_{j_0,k}(x)$$

where the ϕ's are scaling functions and the ψ's are wavelet functions. The Besov space $B_q^\alpha(L_p([0,1]))$ is the following subset of such density functions

$$B_q^\alpha(L_p([0,1])) \stackrel{\text{def}}{=} \left\{ f : \|c_{j_0,k}\|_{\ell_p} + \left(\sum_{j=j_0}^{\infty} \left(2^{\alpha j p} \sum_k |d_{j,k}|^p \right)^{q/p} \right)^{1/q} < \infty \right\},$$

for parameters $\alpha > \frac{1}{p} > 0$ and $q > 0$, where $\{c_{j_0,k}\}$ and $\{d_{j,k}\}$ are the scaling and wavelet coefficients in the wavelet expansion of f.

Nowak and Willett [WN07] showed that any density f in $B^\alpha_q(L_p([0,1]))$ for $0 < q \leq p$, with $\frac{1}{p} = \alpha + \frac{1}{2}$, can be approximated up to L_1 error ϵ with $n = O_\alpha\left(\frac{\log^2(1/\epsilon)}{\epsilon^{\alpha+1/2}}\right)$ samples. They also propose an algorithm for this problem with running time $\Omega(n^3)$.

As a corollary of our main result, we obtain a sample optimal and nearly-linear time agnostic algorithm for this problem. A result in [DeV98] implies that under the above assumptions on α, p, q, any function in $B^\alpha_q(L_p([0,1]))$ can be ϵ-approximated in L_1-norm by an $O_\alpha(\epsilon^{-1/\alpha})$-piece degree-$O(\lceil\alpha\rceil)$ polynomial. Combined with our main result, we obtain an algorithm with sample complexity $O_\alpha\left(\frac{1}{\epsilon^{2+1/\alpha}}\right)$, which is optimal up to constant factors [WN07]. Moreover, the running time of our algorithm is nearly-linear in the number of samples. In particular:

Theorem 44. *There is an agnostic learning algorithm for $B^\alpha_q(L_p([0,1]))$, with $0 < q < p$, $1/p = \alpha + 1/2$ with sample complexity $O_\alpha\left(\frac{1}{\epsilon^{2+1/\alpha}}\right)$ and running time $\widetilde{O}_\alpha\left(\frac{1}{\epsilon^{2+1/\alpha}}\right)$.*

7.4 Mixtures of t-monotone distributions

A density $f : \mathbb{R} \to \mathbb{R}_+$ is 1-monotone if it is non-increasing. It is 2-monotone if it is non-increasing and convex, and t-monotone for $t \geq 3$ if $(-1)^j f^{(j)}$ is non-negative, non-increasing, and convex for $j = 0, \ldots, t-2$. A number of recent works in statistics studied the problem of estimating t-monotone density functions in the context of the MLE [BW07, GW09, BW10].

Implicit in [KL04, KL07] is the fact that any t-monotone bounded density function over $[0,1]$ can be approximated with an $O(1/\epsilon^{1/t})$ piecewise degree $t-1$ polynomial. Using this along with our main result yields the following guarantee on learning t-monotone distributions.

Theorem 45. *There exists an agnostic learning algorithm for k-mixtures of t-monotone distributions that uses $O(tk/\epsilon^{2+1/t})$ samples and runs in time $\widetilde{O}(kt^{2+\omega}/\epsilon^{2+1/t})$.*

The above is a significant improvement in the running time compared to [CDSS14a]. Note that for $t = 1, 2$, the sample complexity of our algorithm is optimal. This follows from known lower bounds of $\Omega(1/\epsilon^3)$ for $t = 1$ [Bir87a] and of $\Omega(1/\epsilon^{5/2})$ for $t = 2$ [DL01].

7.5 Mixtures of discrete distributions

Our main result applies to the discrete setting as well, leading to fast algorithms for learning mixtures of discrete distributions that can be well-approximated by piecewise polynomials.

Mixtures of t-modal discrete distributions and MHR distributions. A distribution over $[N]$ is unimodal if there is a $j \in [N]$ such that the pmf is non-decreasing up to j, and non-increasing after j. A distribution is t-modal if there is a partition of $[N]$ into at most t intervals over which the conditional pmf is unimodal. It follows from [Bir87b, CDSS13] that any mixture of k t-modal distributions is ϵ-close to a $(kt/\epsilon)\log(N/kt)$-histogram. [CDSS14b] implies an algorithm for this problem that uses $n = \widetilde{O}(kt\log(N)/\epsilon^3)$ samples and runs in time $\widetilde{O}(n)$. As a corollary of our main result, we obtain the first sample optimal (up to constant factors) and nearly-linear time algorithm:

Theorem 46. *There is an agnostic learning algorithm for k-mixtures of t-modal distributions over $[N]$ that draws $O(\frac{kt\log(N/kt)}{\epsilon^3})$ samples and runs in time $O(\frac{kt\log(N/kt)}{\epsilon^3}\log(1/\epsilon))$.*

We similarly obtain a sample optimal and near-linear time algorithm for learning mixtures of MHR distributions.

For a distribution p on $[N]$, the function $H(i) \stackrel{\text{def}}{=} \frac{p(i)}{\sum_{j \geq i} p(j)}$ is called the hazard rate function of p. The distribution p is a monotone hazard distribution (MHR) if $H(i)$ is non-decreasing. [CDSS13] shows that a mixture of k MHR distributions over $[N]$ can be approximated up to distance ϵ using an $O(k \log(N/\epsilon)/\epsilon)$-histogram. Using this, [CDSS14b] yields a $\widetilde{O}(k \log(N/\epsilon)/\epsilon^3)$ sample, $\widetilde{O}(k \log(N/\epsilon)/\epsilon^3)$ time algorithm to estimate mixtures of MHR distributions. We obtain

Theorem 47. *There is an agnostic learning algorithm for k-mixtures of MHR distributions over $[N]$ that draws $O(k \log(N/\epsilon)/\epsilon^3)$ samples and runs in time $O(\frac{k \log(N/\epsilon)}{\epsilon^3} \log(1/\epsilon))$.*

Mixtures of Binomial and Poisson distributions. We consider mixtures of k Binomial and Poisson distributions. For these distribution families, the best sample complexity attainable using the techniques of [CDSS14a, CDSS14b] is $\widetilde{O}(k/\epsilon^3)$. This follows from the fact that approximating a k-mixture of Binomial or Poisson distributions by piecewise constant distributions requires $\Theta(k/\epsilon)$ pieces.

A recent result of [DDS15] shows that any Binomial or Poisson distribution can be approximated to L_1 distance ϵ using t-piecewise degree-d polynomials for $t = O(1)$ and $d = O(\log(1/\epsilon))$. Therefore, a Binomial or Poisson k-mixture can be approximated with $O(k)$-piecewise, degree-$O(\log(1/\epsilon))$ polynomials. Since our main result applies to discrete piecewise polynomials as well, we obtain the following:

Theorem 48. *There is an agnostic learning algorithm for k-mixtures of Binomial or Poisson distributions that uses $O(\frac{k}{\epsilon^2} \log(1/\epsilon))$ samples and runs in time $\widetilde{O}(k/\epsilon^2)$.*

8 Experimental Evaluation

In addition to the strong theoretical guarantees proved in the previous sections, our algorithm also demonstrates very good empirical performance. In order to evaluate our algorithm, we conduct several experiments on synthetic data. We remark that the evaluation here is preliminary, and we postpone a more detailed experimental study, including a comparison with related algorithms, to future work. Nevertheless, our results here show that both the empirical sample and time complexity are nearly optimal in a strong sense. For example, no histogram learning algorithm that requires sorted samples can outperform the running time of our method by more than 30%. Similarly, our learning algorithm for piecewise linear hypotheses only adds a factor of $2 - 3\times$ overhead to the time needed to sort the samples. Moreover, the sample complexity of our algorithm matches the quantity $t \cdot (d+1)/\epsilon^2$ up to a small constant between 1 and 2.

All experiments in this section were conducted on a laptop computer from 2010, using an Intel Core i7 CPU with 2.66 GHz clock frequency, 4 MB of cache, and 8 GB of RAM. We used Mac OS X 10.9 as operating system and g++ 4.8 as compiler with the -O3 flag (we implemented our algorithms in C++). All reported running times and learning errors are averaged over 100 independent trials. As an illustrative baseline, sorting 10^6 double-precision floating point numbers with the std::sort algorithm from the C++ STL takes about 100 ms on the above machine.

Figure 7 shows the three distributions we used in our experiments: a mixture of two Gaussians, a mixture of two Beta distributions, and a mixture of two Gamma distributions. The three distributions have different shapes (e.g., different numbers of modes), and the support size considered for these distributions differs.

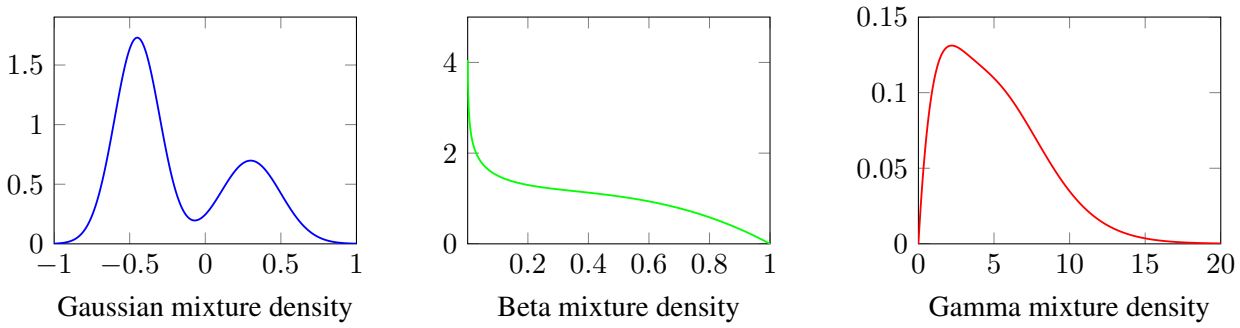

Figure 7: The three test distributions.

8.1 Histogram hypotheses

In order to evaluate our histogram learning algorithm (see Section 4.1), we use the following test setup. For a given unknown distribution with pdf f, we draw n i.i.d. samples from the unknown distribution. We then give the sorted samples as input to our algorithm, which produces a histogram hypothesis h. We set the parameters of our algorithm so that the resulting histogram contains 80 constant pieces. As performance measures, we record the running time of our algorithm (excluding sorting) and the L_1-learning error achieved, i.e., $\|f - h\|_1$.

Figure 8 contains the running time results, both on a linear scale and on a logarithmic scale. The results indicate three important points: (i) The running time of our algorithm scales nearly-linearly with the input size, i.e., the number of samples n. (ii) The constant hidden in the big-O notation of our analysis is very small. In particular, the algorithm runs in less than 35 milliseconds for 10^6 samples. Note that this is three times faster than sorting the samples. (iii) The running time of our algorithm essentially does not depend on the unknown distribution. Such robustness guarantees are very desirable for reliable performance in practice.

The L_1-learning error results are displayed in Figure 9. The results show that the best learning error achievable with 80-piece histograms depends on the shape of the underlying distribution: 2-GMMs are harder to approximate than the Beta and Gamma mixtures. This shows that for large number of samples, it is beneficial to use richer hypotheses classes such as piecewise linear functions (see the next subsection). Nevertheless, our algorithm exhibits a good decay of the learning error before the regime where OPT_{80} dominates.

8.2 Piecewise linear hypotheses

Next, we turn our attention to the more challenging case of agnostically learning piecewise linear densities. This is an interesting case because, in contrast to the histogram algorithm, the piecewise linear algorithm requires our full set of tools developed in Sections 3 – 6. For the case of piecewise linear functions, the structure of the feasible set is still somewhat simpler than for general degree-d polynomials because the non-negativity constraint on a given interval can be encoded with two linear inequalities, i.e., the feasible set is a polytope instead of a spectrahedron. We use this additional structure in our piecewise linear algorithm. However, we did not implement further potential optimizations and resorted to an off-the-shelf linear program (LP) solver (GLPK, the GNU Linear Programming Kit) instead of a customized LP solver. We believe that the running time of our algorithm can be improved further by implementing a custom LP solver that better utilizes the structure and small size of our LPs (and also takes into account that we solve many

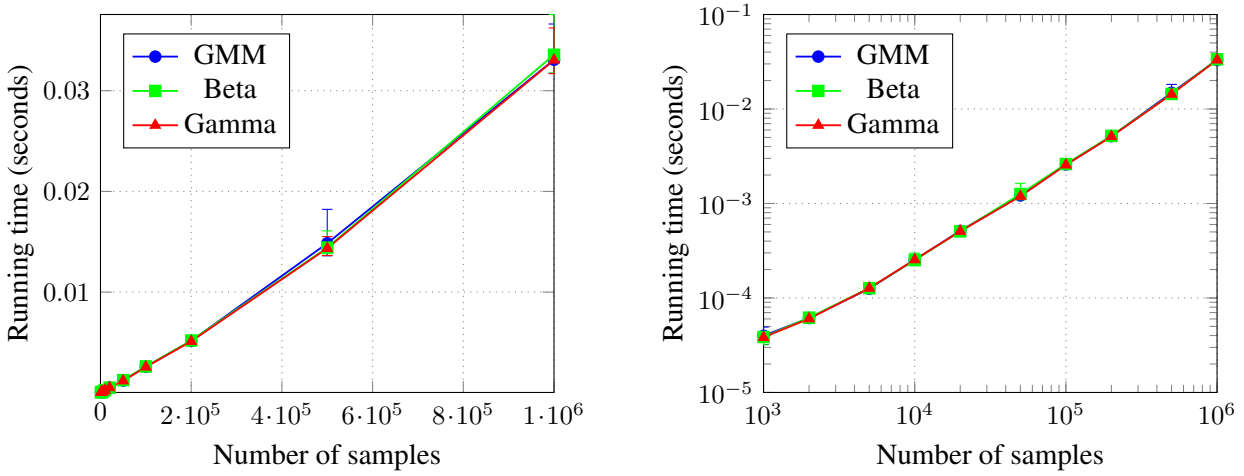

Figure 8: Running times for density estimation with histogram hypotheses. The left plot shows the results on a linear scale, the right plot on a logarithmic scale. As predicted by our analysis, the running time of our algorithm scales nearly-linearly with the input size n. Moreover, the constant in the big-O is very small: for $n = 10^6$, our algorithm takes less than 35 milliseconds, which is about three times faster than sorting the samples. The running time performance of our algorithm is also essentially independent of the unknown distribution.

such small LPs).

We repeat the same experimental procedure as for piecewise histogram hypotheses, but use 40 linear pieces this time. Figure 10 contains the running time results of our algorithm. Again, the results show three important points: (i) As predicted, the running time scales nearly-linearly with n. (ii) In spite of using an off-the-shelf LP solver, the constant factor in our running time is still good. In particular, our algorithm requires less than 0.3 seconds for 10^6 samples. This is only three times slower than the time required for sorting the samples. We believe that with a customized LP solver, we can bring this overhead down to a factor closer to two. (iii) Again, the running time of our algorithm is very robust and does not depend on the shape of the unknown distribution.

Next, we consider the learning error achieved by our piecewise-linear algorithm, which is displayed in Figure 11. Compared with the plots for piecewise constant hypotheses above, the results show that piecewise linear hypotheses can approximate the unknown densities significantly better, especially for the case of the 2-GMM. Three points are worth noting: (i) The slope of the curve in the log-scale plot is about -0.477. Note that this matches the $\frac{1}{\epsilon^2}$ term in our learning error guarantee $O(\frac{t \cdot (d+1)}{\epsilon^2})$ almost perfectly. (ii) Moreover, the constant factor achieved by our algorithm is close to 1. In particular, the learning error for the 2-GMM and $n = 10^6$ samples is roughly 0.00983. Using this as $\epsilon = 0.00983$ together with $t = 40$ and $d = 1$ in $\frac{t \cdot (d+1)}{\epsilon^2}$ gives about 830,000, which almost matches the $n = 10^6$ samples for which this error was obtained. (iii) The learning error of our algorithm is robust and essentially independent of the underlying distribution.

8.3 Comparison with prior work

The paper closest to ours is [CDSS14a]. We did not implement their method because the algorithmic approach of [CDSS14a] is rather complicated and we believe that our algorithm has a significantly better running time for any non-trivially small number of samples. We now substantiate this claim with a simple

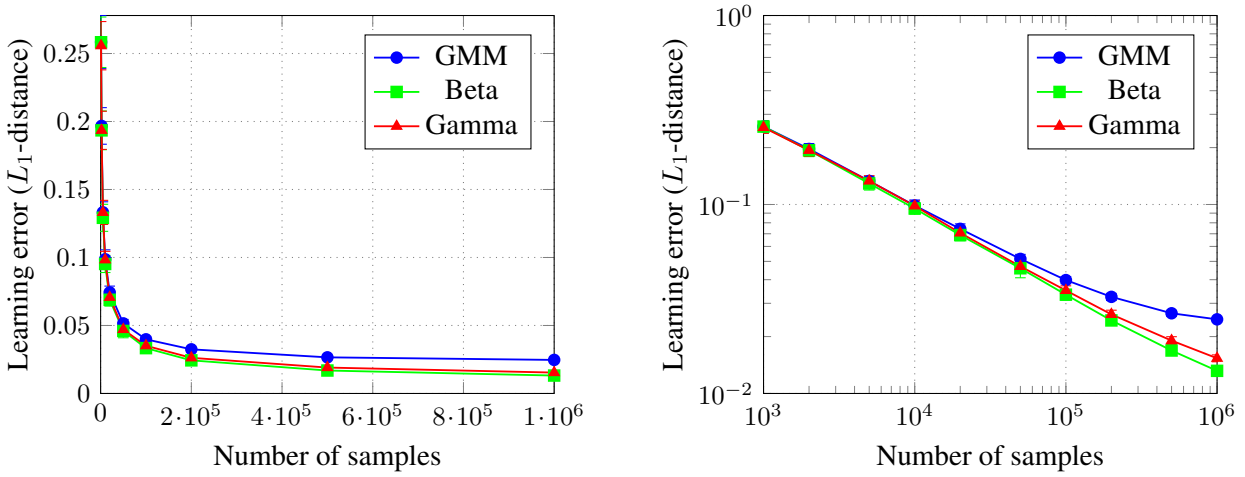

Figure 9: Learning error for density estimation with histogram hypotheses. The left plot shows the results on a linear scale, the right plot on a logarithmic scale. The results clearly show that some distributions such as 2-GMMs are harder to approximate with 80-piecewise constant hypotheses than others. Before the optimal learning error OPT_{80} dominates, our algorithm nevertheless demonstrates a quickly diminishing learning error.

running time estimate.

Recall that the algorithm of [CDSS14a] has a running time of $\widetilde{O}(\frac{t^2 d^{3.5}}{\epsilon^{4.5}}(1/\epsilon + d^3))$ (see Section 1.4).[6] We now instantiate this bound for our piecewise-linear (i.e., $d = 1$) density estimation example with $n = 10^6$ samples above. Using $t = 40$ pieces, our algorithm achieves an approximation error of less than $\epsilon = 0.01$. Substituting these numbers into the time complexity of [CDSS14a] and ignoring logarithmic and constant factors, we get

$$\frac{t^2 \cdot (d+1)^{3.5}}{\epsilon^{4.5}}(1/\epsilon + (d+1)^3)) \;\approx\; 2.0 \cdot 10^{15} \;.$$

We use this expression as the number of simple instructions required by their algorithm. Assuming a modern CPU that executes $3 \cdot 10^9$ instructions per second, their algorithm would run for about $6.5 \cdot 10^5$ seconds, which is roughly 7.5 days. Clearly, such a running time is infeasible in practice. As mentioned in the previous subsection, our algorithm requires 0.3 seconds for this example, which leads to a speed-up of more than 10^6.

We remark that this back-of-the-envelope calculation is in favor of [CDSS14a]. Using the same calculation approach for our time complexity would give

$$\frac{t \cdot (d+1)^6}{\epsilon^2} \;\approx\; 2.6 \cdot 10^7 \;.$$

Assuming the same CPU, the corresponding running time estimate is roughly 0.009 seconds. This underestimates our running time by a factor of more than 30. Considering that the time complexity of [CDSS14a] contains more logarithmic factors than our time complexity and most likely also a larger constant factor, the

[6] In Section 1.4, we state the running time with d in place of $d+1$ to simplify the expression. Note that as for our algorithm, the correct dependence is in terms of $d+1$ because a degree-d polynomial has $d+1$ parameters. For instance, the running time does not vanish for the histogram ($d = 0$) case.

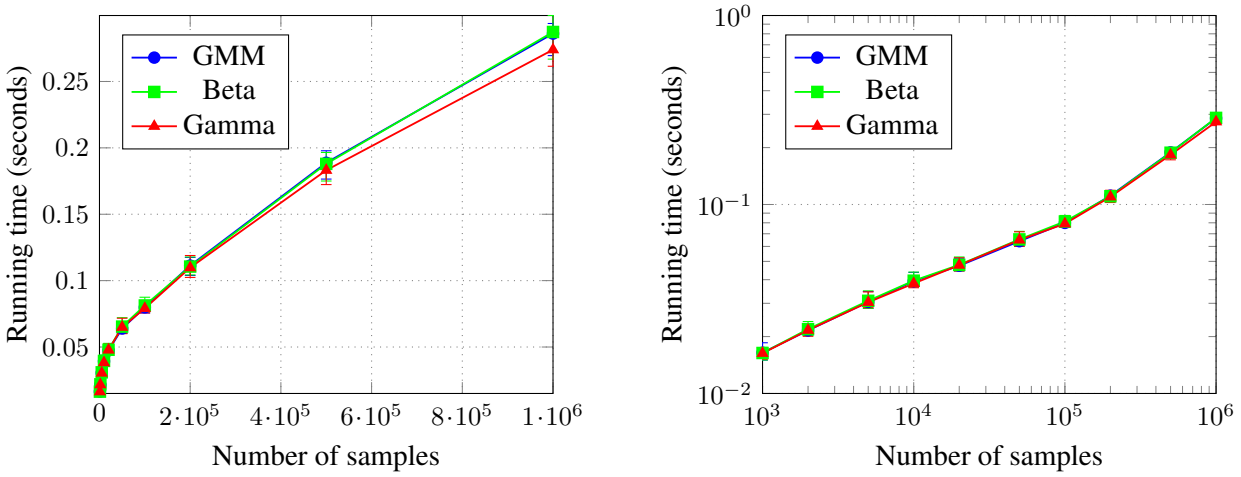

Figure 10: Running times for density estimation with piecewise-linear hypotheses. The left plot shows the results on a linear scale, the right plot on a logarithmic scale. As predicted by our analysis, the running time of our algorithm scales nearly-linearly with the input size n. Moreover, the constant in the big-O is quite small: for $n = 10^6$, our algorithm takes less than 0.3 seconds, which is only three times slower than sorting the samples. Note that this means that no algorithm that relies on sorting the samples can be more than 4 times faster than our algorithm when the total running time with sorting is taken into account. As before, the running time of our algorithm is also essentially independent of the unknown distribution.

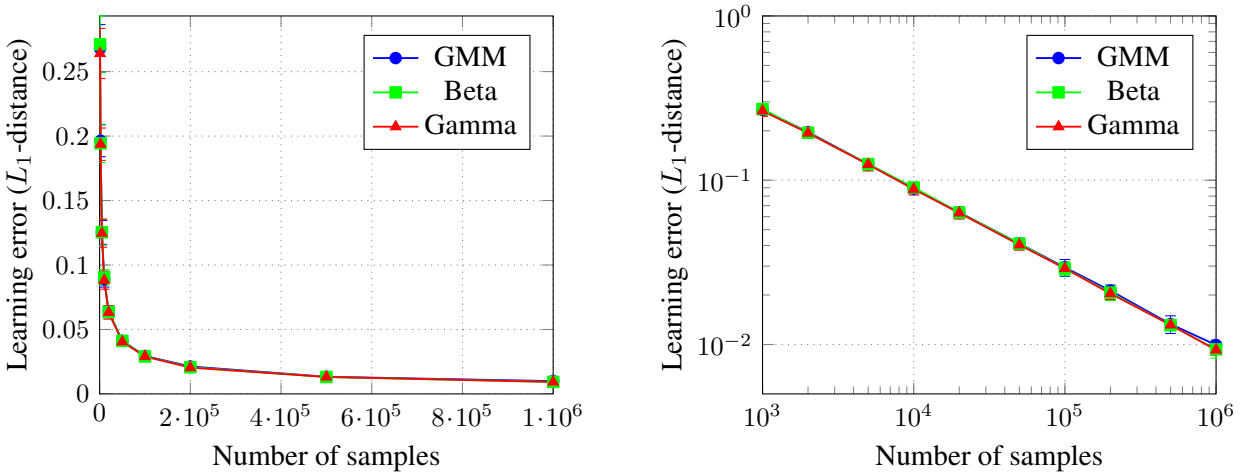

Figure 11: Learning error for density estimation with piecewise-linear hypotheses. The left plot shows the results on a linear scale, the right plot on a logarithmic scale. The slope of the curve in the log-scale plot is roughly $-.477$, which almost exactly matches the asymptotic guarantee for our algorithm. Moreover, the average learning error for 2-GMMs with $n = 10^6$ samples is about 0.00983. Substituting this into the theoretical guarantee $\frac{t\cdot(d+1)}{\epsilon^2}$ gives a sample requirement of roughly 830,000, i.e., very close to the 10^6 samples our algorithm required to achieve this error. Similar to the running time, the learning error is also robust and essentially independent of the underlying distribution.

true gap in the running times of [CDSS14a] and our algorithm might be even larger than the $10^6\times$ speed-up mentioned above.

Acknowledgements

We thank Chinmay Hegde for his contributions to the early stages of this work. We would like to thank Yin Tat Lee and Aaron Sidford for useful discussions, and Richard Samworth for his help with the statistics literature.

References

[ADH+15] J. Acharya, I. Diakonikolas, C. Hegde, J. Li, and L. Schmidt. Fast and Near-Optimal Algorithms for Approximating Distributions by Histograms. In *PODS*, 2015.

[AK01] S. Arora and R. Kannan. Learning mixtures of arbitrary Gaussians. In *STOC*, pages 247–257, 2001.

[AM05] D. Achlioptas and F. McSherry. On spectral learning of mixtures of distributions. In *COLT*, pages 458–469, 2005.

[BB05] M. Bagnoli and T. Bergstrom. Log-concave probability and its applications. *Economic theory*, 26(2):445–469, 2005.

[BBBB72] R. E. Barlow, D. J. Bartholomew, J. M. Bremner, and H. D. Brunk. *Statistical Inference under Order Restrictions*. Wiley, New York, 1972.

[BD14] F. Balabdaoui and C. R. Doss. Inference for a Mixture of Symmetric Distributions under Log-Concavity. Available at http://arxiv.org/abs/1411.4708, 2014.

[Bir87a] L. Birgé. Estimating a density under order restrictions: Nonasymptotic minimax risk. *Annals of Statistics*, 15(3):995–1012, 1987.

[Bir87b] L. Birgé. On the risk of histograms for estimating decreasing densities. *Annals of Statistics*, 15(3):1013–1022, 1987.

[BNSV15] M. Bun, K. Nissim, U. Stemmer, and S. P. Vadhan. Differentially private release and learning of threshold functions. *CoRR*, abs/1504.07553, 2015.

[Bru58] H. D. Brunk. On the estimation of parameters restricted by inequalities. *The Annals of Mathematical Statistics*, 29(2):pp. 437–454, 1958.

[BRW09] F. Balabdaoui, K. Rufibach, and J. A. Wellner. Limit distribution theory for maximum likelihood estimation of a log-concave density. *The Annals of Statistics*, 37(3):pp. 1299–1331, 2009.

[BS10] M. Belkin and K. Sinha. Polynomial learning of distribution families. In *FOCS*, pages 103–112, 2010.

[BSZ15] A. Bhaskara, A. T. Suresh, and M. Zaghimoghaddam. Sparse Solutions to Nonegative Linear Systems and Applications. In *AISTATS*, 2015.

[BW07] F. Balabdaoui and J. A. Wellner. Estimation of a k-monotone density: Limit distribution theory and the spline connection. *The Annals of Statistics*, 35(6):pp. 2536–2564, 2007.

[BW10] F. Balabdaoui and J. A. Wellner. Estimation of a k-monotone density: characterizations, consistency and minimax lower bounds. *Statistica Neerlandica*, 64(1):45–70, 2010.

[CDSS13] S. Chan, I. Diakonikolas, R. Servedio, and X. Sun. Learning mixtures of structured distributions over discrete domains. In *SODA*, pages 1380–1394, 2013.

[CDSS14a] S. Chan, I. Diakonikolas, R. Servedio, and X. Sun. Efficient density estimation via piecewise polynomial approximation. In *STOC*, pages 604–613, 2014.

[CDSS14b] S. Chan, I. Diakonikolas, R. Servedio, and X. Sun. Near-optimal density estimation in near-linear time using variable-width histograms. In *NIPS*, pages 1844–1852, 2014.

[Che82] E. W. Cheney. *Introduction to Approximation Theory: Second Edition*. AMS Chelsea Publishing, 1982.

[CS13] Y. Chen and R. J. Samworth. Smoothed log-concave maximum likelihood estimation with applications. *Statist. Sinica*, 23:1373–1398, 2013.

[CSS10] M. Cule, R. Samworth, and M Stewart. Maximum likelihood estimation of a multi-dimensional log-concave density. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 72:545–607, 2010.

[Csu04] M. Csuros. Maximum-scoring segment sets. *IEEE/ACM Trans. Comput. Biol. Bioinformatics*, 1(4):139–150, October 2004.

[CT04] K. S. Chan and H. Tong. Testing for multimodality with dependent data. *Biometrika*, 91(1):113–123, 2004.

[Das99] S. Dasgupta. Learning mixtures of Gaussians. In *FOCS*, pages 634–644, 1999.

[DDO+13] C. Daskalakis, I. Diakonikolas, R. O'Donnell, R.A. Servedio, and L. Tan. Learning Sums of Independent Integer Random Variables. In *FOCS*, pages 217–226, 2013.

[DDS12a] C. Daskalakis, I. Diakonikolas, and R.A. Servedio. Learning k-modal distributions via testing. In *SODA*, pages 1371–1385, 2012.

[DDS12b] C. Daskalakis, I. Diakonikolas, and R.A. Servedio. Learning Poisson Binomial Distributions. In *STOC*, pages 709–728, 2012.

[DDS15] C. Daskalakis, I. Diakonikolas, and A. Stewart. Personal communication, 2015.

[DeV98] R. A. DeVore. Nonlinear approximation. *ACTA NUMERICA*, 7:51–150, 1998.

[DG85] L. Devroye and L. Györfi. *Nonparametric Density Estimation: The L_1 View*. John Wiley & Sons, 1985.

[DHS15] I. Diakonikolas, M. Hardt, and L. Schmidt. Differentially private learning of structured discrete distributions. In *NIPS*, 2015.

[DJ98] D. L. Donoho and I. M. Johnstone. Minimax estimation via wavelet shrinkage. *Annals of Statistics*, 26(3):879–921, 1998.

[DJKP95] D. L. Donoho, I. M. Johnstone, G. Kerkyacharian, and D. Picard. Wavelet shrinkage: asymptopia. *Journal of the Royal Statistical Society, Ser. B*, pages 371–394, 1995.

[DJKP96] D. L. Donoho, I. M. Johnstone, G. Kerkyacharian, and D. Picard. Density estimation by wavelet thresholding. *Annals of Statistics*, 24(2):508–539, 1996.

[DK14] C. Daskalakis and G. Kamath. Faster and sample near-optimal algorithms for proper learning mixtures of gaussians. In *COLT*, pages 1183–1213, 2014.

[DK15] I. Diakonikolas and D. Kane. Personal communication, 2015.

[DL01] L. Devroye and G. Lugosi. *Combinatorial methods in density estimation*. Springer Series in Statistics, Springer, 2001.

[DR09] L. Dumbgen and K. Rufibach. Maximum likelihood estimation of a log-concave density and its distribution function: Basic properties and uniform consistency. *Bernoulli*, 15(1):40–68, 2009.

[DS00] S. Dasgupta and L. Schulman. A two-round variant of EM for Gaussian mixtures. In *UAI*, pages 143–151, 2000.

[DW13] C. R. Doss and J. A. Wellner. Global Rates of Convergence of the MLEs of Log-concave and s-concave Densities. Available at http://arxiv.org/abs/1306.1438, 2013.

[FM99] Y. Freund and Y. Mansour. Estimating a mixture of two product distributions. In *COLT*, pages 183–192, 1999.

[FOS05] J. Feldman, R. O'Donnell, and R. Servedio. Learning mixtures of product distributions over discrete domains. In *FOCS*, pages 501–510, 2005.

[Fou97] A.-L. Fougères. Estimation de densités unimodales. *Canadian Journal of Statistics*, 25:375–387, 1997.

[GJ14] P. Groeneboom and G. Jongbloed. *Nonparametric Estimation under Shape Constraints: Estimators, Algorithms and Asymptotics*. Cambridge University Press, 2014.

[Gre56] U. Grenander. On the theory of mortality measurement. *Skandinavisk Aktuarietidskrift*, 39:125–153, 1956.

[Gro85] P. Groeneboom. Estimating a monotone density. In *Proc. of the Berkeley Conference in Honor of Jerzy Neyman and Jack Kiefer*, pages 539–555, 1985.

[GW09] F. Gao and J. A. Wellner. On the rate of convergence of the maximum likelihood estimator of a k-monotone density. *Science in China Series A: Mathematics*, 52:1525–1538, 2009.

[Hen88] P. Henrici. *Applied and Computational Complex Analysis, Power Series Integration Conformal Mapping Location of Zero*. Applied and Computational Complex Analysis. Wiley, 1988.

[HP76] D. L. Hanson and G. Pledger. Consistency in concave regression. *The Annals of Statistics*, 4(6):pp. 1038–1050, 1976.

[HP15] M. Hardt and E. Price. Sharp bounds for learning a mixture of two gaussians. In *STOC*, 2015.

[HW15] Q. Han and J. A. Wellner. Approximation and Estimation of s-Concave Densities via Renyi Divergences. Available at http://arxiv.org/abs/1505.00379, 2015.

[Ize91] A. J. Izenman. Recent developments in nonparametric density estimation. *Journal of the American Statistical Association*, 86(413):205–224, 1991.

[JW09] H. K. Jankowski and J. A. Wellner. Estimation of a discrete monotone density. *Electronic Journal of Statistics*, 3:1567–1605, 2009.

[Kha79] L. Khachiyan. A polynomial algorithm in linear programming. *Soviet Math. Dokl*, 20:1093–1096, 1979.

[KL04] V. N. Konovalov and D. Leviatan. Free-knot splines approximation of s-monotone functions. *Advances in Computational Mathematics*, 20(4):347–366, 2004.

[KL07] V. N. Konovalov and D. Leviatan. Freeknot splines approximation of sobolev-type classes of s -monotone functions. *Adv. Comput. Math.*, 27(2):211–236, 2007.

[KM10] R. Koenker and I. Mizera. Quasi-concave density estimation. *Annals of Statistics*, 38(5):2998–3027, 2010.

[KMR$^+$94] M. Kearns, Y. Mansour, D. Ron, R. Rubinfeld, R. Schapire, and L. Sellie. On the learnability of discrete distributions. In *STOC*, pages 273–282, 1994.

[KMV10] A. T. Kalai, A. Moitra, and G. Valiant. Efficiently learning mixtures of two Gaussians. In *STOC*, pages 553–562, 2010.

[KP92] G. Kerkyacharian and D. Picard. Density estimation in Besov spaces. *Statistics & Probability Letters*, 13(1):15–24, 1992.

[KPT96] G. Kerkyacharian, D. Picard, and K. Tribouley. Lp adaptive density estimation. *Bernoulli*, 2(3):pp. 229–247, 1996.

[KS14] A. K. H. Kim and R. J. Samworth. Global rates of convergence in log-concave density estimation. Available at http://arxiv.org/abs/1404.2298, 2014.

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

[Las01] J. B. Lasserre. Global optimization with polynomials and the problem of moments. *SIAM Journal on Optimization*, 11(3):796–817, 2001.

[LB99] J. Q. Li and A. R. Barron. Mixture density estimation. In *NIPS*, pages 279–285, 1999.

[Lin95] B. Lindsay. *Mixture models: theory, geometry and applications*. Institute for Mathematical Statistics, 1995.

[LS14] Y. T. Lee and A. Sidford. Path finding methods for linear programming: Solving linear programs in $\widetilde{O}\sqrt{\text{rank}}$ iterations and faster algorithms for maximum flow. In *FOCS*, pages 424–433, 2014.

[LS15] J. Li and L. Schmidt. A Nearly Optimal and Agnostic Algorithm for Properly Learning a Mixture of k Gaussians, for any Constant k. Manuscript, available on arxiv., 2015.

[Mar92] VA Markov. On functions deviating least from zero in a given interval. *Izdat. Imp. Akad. Nauk, St. Petersburg*, pages 218–258, 1892.

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

[Pan01] V. Y. Pan. Univariate polynomials: nearly optimal algorithms for factorization and rootfinding. In *ISSAC*, pages 253–267. ACM, 2001.

[Par03] P. A. Parrilo. Semidefinite programming relaxations for semialgebraic problems. *Mathematical programming*, 96(2):293–320, 2003.

[Pea95] K. Pearson. Contributions to the mathematical theory of evolution. ii. skew variation in homogeneous material. *Philosophical Trans. of the Royal Society of London*, 186:343–414, 1895.

[Rao69] B. L. S. Prakasa Rao. Estimation of a unimodal density. *Sankhya A*, 31:23–36, 1969.

[RW84] R. A. Redner and H. F. Walker. Mixture densities, maximum likelihood and the EM algorithm. *SIAM Review*, 26:195–202, 1984.

[Sco92] D. W. Scott. *Multivariate Density Estimation: Theory, Practice and Visualization*. Wiley, New York, 1992.

[SHKT97] C. J. Stone, M. H. Hansen, C. Kooperberg, and Y. K. Truong. Polynomial splines and their tensor products in extended linear modeling: 1994 wald memorial lecture. *Annals of Statistics*, 25(4):1371–1470, 1997.

[Sho87] N. Z. Shor. Class of global minimum bounds of polynomial functions. *Cybernetics and Systems Analysis*, 23(6):731–734, 1987.

[Sil86] B. W. Silverman. *Density Estimation*. Chapman and Hall, London, 1986.

[SOAJ14] A. T. Suresh, A. Orlitsky, J. Acharya, and A. Jafarpour. Near-optimal-sample estimators for spherical gaussian mixtures. In *NIPS*, pages 1395–1403, 2014.

[Sto94] C. J. Stone. The use of polynomial splines and their tensor products in multivariate function estimation. *Annals of Statistics*, 22(1):pp. 118–171, 1994.

[SW09] G.R. Shorack and J.A. Wellner. *Empirical Processes with Applications to Statistics*. Classics in Applied Mathematics. Society for Industrial and Applied Mathematics (SIAM, 3600 Market Street, Floor 6, Philadelphia, PA 19104), 2009.

[Tim63] A. F. Timan. *Theory of approximation of functions of a real variable*, volume 34. Courier Corporation, 1963.

[TSM85] D. M. Titterington, A. F. M. Smith, and U. E. Makov. *Statistical analysis of finite mixture distributions*. Wiley & Sons, 1985.

[Vai89] P. Vaidya. A new algorithm for minimizing convex functions over convex sets. In *FOCS*, pages 338–343, 1989.

[Vai96] P. M. Vaidya. A new algorithm for minimizing convex functions over convex sets. *Mathematical Programming*, 73(3):291–341, 1996.

[Val84] L. G. Valiant. A theory of the learnable. In *STOC*, pages 436–445. ACM Press, 1984.

[VC71] V. Vapnik and A. Chervonenkis. On the uniform convergence of relative frequencies of events to their probabilities. *Theory of Probability and Its Applications*, 16:264–280, 1971.

[VW02] S. Vempala and G. Wang. A spectral algorithm for learning mixtures of distributions. In *FOCS*, pages 113–122, 2002.

[VZGG13] J. Von Zur Gathen and J. Gerhard. *Modern computer algebra*. Cambridge university press, 2013.

[Wal09] G. Walther. Inference and modeling with log-concave distributions. *Statistical Science*, 24(3):319–327, 2009.

[Weg70] E. J. Wegman. Maximum likelihood estimation of a unimodal density. I. and II. *Annals of Mathematical Statistics*, 41:457–471, 2169–2174, 1970.

[WN07] R. Willett and R. D. Nowak. Multiscale poisson intensity and density estimation. *IEEE Transactions on Information Theory*, 53(9):3171–3187, 2007.

[WW83] E. J. Wegman and I. W. Wright. Splines in statistics. *Journal of the American Statistical Association*, 78(382):pp. 351–365, 1983.

Appendix

A Analysis of the General Merging Algorithm: Proof of Theorem 17

This section is dedicated to the proof of Theorem 17. The proof is a generalization of that of Theorem 10. Recall the statement of Theorem 17:

Theorem 17. *Let $\mathcal{O}_p$ and $\mathcal{O}_c$ be η-approximate $\mathcal{A}_k$-projection and $\mathcal{A}_k$-computation oracles for $\mathcal{D}$. Algorithm* GENERAL-MERGING$(f, t, \alpha, \epsilon, \delta)$ *draws $n = O((\alpha dt + \log(1/\delta))/\epsilon^2)$ samples, has time complexity $O\left((R_p(n) + R_c(n)) \log \frac{n}{\alpha t}\right)$, and outputs a hypothesis h and an interval partition $\mathcal{I}$ such that $|\mathcal{I}| \leq 2\alpha \cdot t$ and with probability at least $1 - \delta$, we have*

$$\|h - f\|_1 \leq 3 \cdot \mathrm{OPT}_{\mathcal{D},t} + \frac{\mathrm{OPT}_{\mathcal{D},t} + \epsilon}{\alpha - 1} + 2\epsilon + \eta \,. \tag{10}$$

Proof. We first bound the running time. The number of iterations of the algorithm is $O(\log(n/\alpha t))$ by the same argument as for histograms, since the number of intervals reduces by a factor of $3/4$ in each iteration.

In each iteration, we compute the closest function in $\mathcal{D}$ and the corresponding $\mathcal{A}_{d+1}$ distance, hence the runtime per iteration is bounded by $R_p(n) + R_c(n)$, by definition.

We now prove the error guarantee. Let $\mathcal{I} = \{I_1, \ldots, I_{t'}\}$ be the partition of I returned by GENERAL-MERGING, and let h be the function returned. The desired bound on t' is immediate since the algorithm terminates only when $t' \leq 2\alpha t$. We now prove (10).

Let $h^* \in \mathcal{D}_t$ be such that $\|h^* - f\|_1 = \mathrm{OPT}_{\mathcal{D},t}$. Let $\mathcal{I}^* = \{I_1^*, \ldots, I_t^*\}$ be a partition with at most t pieces such that $h^* \in \mathcal{D}_{I_i^*}$ for all i. Call the end-points of I_j^*'s as *jumps* of h^*. For any interval $J \subseteq I$ let $\Gamma(J)$ be the number of jumps of h^* in the interior of J. Since we draw $n = \Omega((\alpha dt + \log 1/\delta)/\epsilon^2)$ samples, Corollary 4 implies that with probability at least $1 - \delta$,

$$\|\widehat{f} - f\|_{\mathcal{A}_{(2\alpha+1)(d+1)t}} \leq \epsilon \ .$$

We condition on this event throughout the analysis.
We split the total error into three terms based on the final partition $\mathcal{I}$:

Case 1: Let $\mathcal{F}$ be the set of intervals in $\mathcal{I}$ with no jumps in h^*, i.e., $\mathcal{F} = \{J \in \mathcal{I} \,|\, \Gamma(J) = 0\}$.

Case 2a: Let $\mathcal{J}_0$ be the set of intervals in $\mathcal{I}$ that were created in the initial partitioning step of the algorithm and contain a jump of h^*, i.e., $\mathcal{J}_0 = \{J \in \mathcal{I} \mid \Gamma(J) > 0 \text{ and } J \in \mathcal{I}_0\}$.

Case 2b: Let $\mathcal{J}_1$ be the set of intervals in $\mathcal{I}$ that contain at least one jump, and were created by merging two other intervals, i.e., $\mathcal{J}_1 = \{J \in \mathcal{I} \mid \Gamma(J) > 0 \text{ and } J \notin \mathcal{I}_0\}$.

Notice that $\mathcal{F}, \mathcal{J}_0, \mathcal{J}_1$ form a partition of I, and thus

$$\|h - f\|_1 = \|h - f\|_{1,\mathcal{F}} + \|h - f\|_{1,\mathcal{J}_0} + \|h - f\|_{1,\mathcal{J}_1} \ .$$

We bound the error from above in the three cases separately. In particular, we will show:

$$\|h - f\|_{1,\mathcal{F}} \leq 3 \cdot \|f - h^*\|_{1,\mathcal{F}} + 2 \cdot \|\widehat{f} - f\|_{\mathcal{A}_{|\mathcal{F}|\cdot(d+1)},\mathcal{F}} + \frac{\eta}{2\alpha t}|\mathcal{F}| \ , \tag{14}$$

$$\|h - f\|_{1,\mathcal{J}_0} \leq \|\widehat{f} - f\|_{\mathcal{A}_{|\mathcal{J}_0|\cdot(m+1)},\mathcal{J}_0} \ , \tag{15}$$

$$\|h - f\|_{1,\mathcal{J}_1} \leq \frac{\mathrm{OPT}_{\mathcal{D},t} + \epsilon}{(\alpha - 1)} + \|\widehat{f} - f\|_{\mathcal{A}_{d\cdot t + |\mathcal{J}_1|},\mathcal{J}_1} + \|f - h^*\|_{1,\mathcal{J}_1} + \frac{\eta}{2(\alpha - 1)} \ . \tag{16}$$

Using these results along with the fact that $\|f - h^*\|_{1,\mathcal{F}} + \|f - h^*\|_{1,\mathcal{J}_1} \leq \mathrm{OPT}_{\mathcal{D},t}$ and $\alpha > 2$, we have

$$\begin{aligned}
\|h - f\|_1 &\leq 3 \cdot \mathrm{OPT}_{\mathcal{D},t} + \frac{\mathrm{OPT}_{\mathcal{D},t} + \epsilon}{\alpha - 1} + 2\|\widehat{f} - f\|_{\mathcal{A}_{|\mathcal{F}|(d+1)}} + \|\widehat{f} - f\|_{\mathcal{A}_{|\mathcal{J}_0|d}} \\
&\quad + \|\widehat{f} - f\|_{\mathcal{A}_{(|\mathcal{J}_1|+t)d}} + \frac{\eta}{2\alpha t}(|\mathcal{F}| + \mathcal{J}_1) \\
&\overset{(a)}{\leq} 3 \cdot \mathrm{OPT}_{\mathcal{D},t} + \frac{\mathrm{OPT}_{\mathcal{D},t} + \epsilon}{\alpha - 1} + 2\|\widehat{f} - f\|_{\mathcal{A}_{2\alpha t(d+1)}} + \eta \\
&\overset{(b)}{\leq} 3 \cdot \mathrm{OPT}_{\mathcal{D},t} + \frac{\mathrm{OPT}_{\mathcal{D},t} + \epsilon}{\alpha - 1} + 2\epsilon + \eta \ ,
\end{aligned}$$

where (a) follows from Fact 6(d) and since $(|\mathcal{F}| + |\mathcal{J}_1| + |\mathcal{J}_0|) \leq 2\alpha t$, and (b) follows from the VC inequality. Thus, it suffices to prove Equations (14)–(16).

Case 1. We first consider the set of intervals in $\mathcal{F}$. By the triangle inequality we have

$$\|h - f\|_{1,\mathcal{F}} \leq \|f - h^*\|_{1,\mathcal{F}} + \|h - h^*\|_{1,\mathcal{F}} \ .$$

For any interval $J \in \mathcal{F}$, since h and h^* are both in $\mathcal{D}$, they have at most d sign changes, and

$$\|h - h^*\|_{1,J} = \|h - h^*\|_{\mathcal{A}_{d+1},J} \leq \|h - \widehat{f}\|_{\mathcal{A}_{d+1},J} + \|\widehat{f} - h^*\|_{\mathcal{A}_{d+1},J}.$$

By the definition of h and the projection oracle,

$$\|h - \widehat{f}\|_{\mathcal{A}_{d+1},J} \leq \min_{h' \in \mathcal{D}_J} \|h' - \widehat{f}\|_{\mathcal{A}_{d+1},J} + \frac{\eta}{2\alpha t} \leq \|h^* - \widehat{f}\|_{\mathcal{A}_{d+1},J} + \frac{\eta}{2\alpha t}.$$

Therefore,

$$\|h - h^*\|_{1,J} \leq 2 \cdot \|h^* - \widehat{f}\|_{\mathcal{A}_{d+1},J} + \frac{\eta}{2\alpha t}.$$

Again by the triangle inequality,

$$\|h^* - \widehat{f}\|_{\mathcal{A}_{d+1},J} \leq \|h^* - f\|_{\mathcal{A}_{d+1},J} + \|f - \widehat{f}\|_{\mathcal{A}_{d+1},J}.$$

Summing over the intervals in $\mathcal{F}$,

$$\begin{aligned}\sum_{J \in \mathcal{F}} \|h^* - \widehat{f}\|_{\mathcal{A}_{d+1},J} &\leq \sum_{J \in \mathcal{F}} \|h^* - f\|_{\mathcal{A}_{d+1},J} + \sum_{J \in \mathcal{F}} \|f - \widehat{f}\|_{\mathcal{A}_{d+1},J} \\ &\leq \|h^* - f\|_{1,\mathcal{F}} + \|f - \widehat{f}\|_{\mathcal{A}_{|\mathcal{F}|(d+1)},\mathcal{F}}\end{aligned}$$

Combining these, we obtain,

$$\|h - f\|_{1,\mathcal{F}} \leq 3 \cdot \|f - h^*\|_{1,\mathcal{F}} + 2 \cdot \|f - \widehat{f}\|_{\mathcal{A}_{|\mathcal{F}|(d+1)},\mathcal{F}} + \frac{\eta}{2\alpha t}|\mathcal{F}| \ ,$$

which is precisely (14).

Case 2a. We now analyze the error for the intervals $\mathcal{J}_0$. The set $\mathcal{I}_0$ contains only singletons and intervals with no sample points. By definition, with probability 1, only the intervals in $\mathcal{I}_0$ that contain no samples may contain a jump of h^*. The singleton intervals containing the sample points do not include jumps, and are hence covered by Case 1. Since $\mathcal{J}_0$ does not contain any samples, our algorithm assigns

$$h(J) = \widehat{f}(J) = 0$$

for any $J \in \mathcal{J}_0$. Hence,

$$\|h - f\|_{1,\mathcal{J}_0} = \|f\|_{1,\mathcal{J}_0} \ ,$$

and

$$\begin{aligned}\|h - f\|_{1,\mathcal{J}_0} &= \|f\|_{1,\mathcal{J}_0} \\ &= \sum_{J \in \mathcal{J}_0} |f(J)| \\ &= \sum_{J \in \mathcal{J}_0} |f(J) - \widehat{f}(J)| \\ &\leq \|f - \widehat{f}\|_{\mathcal{A}_{|\mathcal{J}_0|(d+1)},\mathcal{J}_0} \ ,\end{aligned}$$

where the last step simply follows from non-negativity of $f - \widehat{f}$ over $\mathcal{J}_0$.

Case 2b. We finally consider $\mathcal{J}_1$, the set of intervals created by merging in some iteration of our algorithm that also contain jumps. As before, our first step is the following triangle inequality:

$$\|h - f\|_{1,\mathcal{J}_1} \leq \|h - h^*\|_{1,\mathcal{J}_1} + \|h^* - f\|_{1,\mathcal{J}_1} \ .$$

Consider an interval $J \in \mathcal{J}_1$ with $\Gamma(J) \geq 1$ jumps of h^*. Since $h \in \mathcal{D}_J$, $h - h^*$ has at most $d \cdot \Gamma(J)$ sign changes in J. Therefore,

$$\begin{aligned}
\|h - h^*\|_{1,J} &\overset{(a)}{=} \|h - h^*\|_{\mathcal{A}_{d\cdot\Gamma(J)+1},J} \\
&\overset{(b)}{\leq} \|h - \widehat{f}\|_{\mathcal{A}_{d\cdot\Gamma(J)+1},J} + \|\widehat{f} - f\|_{\mathcal{A}_{d\cdot\Gamma(J)+1},J} + \|f - h^*\|_{\mathcal{A}_{d\cdot\Gamma(J)+1},J} \\
&\overset{(c)}{\leq} \Gamma(J)\|h - \widehat{f}\|_{\mathcal{A}_{d+1},J} + \|\widehat{f} - f\|_{\mathcal{A}_{d\cdot\Gamma(J)+1},J} + \|f - h^*\|_{1,J} \ , \qquad (17)
\end{aligned}$$

where (a) follows from Fact 6(a), (b) is the triangle inequality, and inequality (c) uses Fact 6(c) along with the fact that $\Gamma(J) \geq 1$ and $d \geq 1$. We start by bounding the $\mathcal{A}_{d+1}$ distance in the first term above.

Lemma 49. *For any $J \in \mathcal{J}_1$, we have*

$$\|h - \widehat{f}\|_{\mathcal{A}_{d+1},J} \leq \frac{\mathrm{OPT}_{\mathcal{D},t} + \epsilon}{(\alpha-1)t} + \frac{\eta}{2(\alpha-1)t} \ . \qquad (18)$$

Before proving this lemma, we use it to complete Case 2b. Summing (7) over $J \in \mathcal{J}_1$ and plugging in the lemma,

$$\begin{aligned}
\|h - h^*\|_{1,\mathcal{J}_1} &\leq \left(\sum_{J\in\mathcal{J}_1}(\Gamma(J)\right) \cdot \left(\frac{\mathrm{OPT}_{\mathcal{D},t} + \epsilon}{(\alpha-1)t} + \frac{\eta}{2(\alpha-1)t}\right) + \sum_{J\in\mathcal{J}_1} \|\widehat{f} - f\|_{\mathcal{A}_{d\cdot\Gamma(J)+1},J} + \|f - h^*\|_{1,\mathcal{J}_1} \\
&\overset{(a)}{\leq} \frac{\mathrm{OPT}_{\mathcal{D},t} + \epsilon}{(\alpha-1)} + \frac{\eta}{2(\alpha-1)} + \|\widehat{f} - f\|_{\mathcal{A}_{d\cdot t+|\mathcal{J}_1|},\mathcal{J}_1} + \|f - h^*\|_1, \mathcal{J}_1
\end{aligned}$$

where the first term in (a) uses the fact that $\sum_{J\in\mathcal{J}_1}\Gamma(J) \leq t$ and the second term uses this in conjunction with Fact 6(d).

We now prove Lemma 49.

Proof of Lemma 49. Each iteration of our algorithm merges pairs of intervals except those with the αt largest errors. Therefore, if two intervals were merged, there were at least αt other interval pairs with larger error. We will use this fact to bound the error on the intervals in $\mathcal{J}_1$.

Suppose an interval $J \in \mathcal{J}_1$ was created in the jth iteration of the while loop of our algorithm, i.e., $J = I'_{i,j+1} = I_{2i-1,j} \cup I_{2i,j}$ for some $i \in \{1, \ldots, s_j/2\}$. Recall that the intervals $I'_{i,j+1}$, for $i \in \{1, \ldots, s_j/2\}$, are the candidates for merging at iteration j. Let h' be the distribution given by applying the projection oracle to the empirical distribution over each candidate interval $\mathcal{I}'_{j+1} = \{I'_{1,j+1}, \ldots, I'_{s_j/2,j+1}\}$. Note that $h'(x) = h(x)$ for $x \in J$ since J remains intact through the remainder of the algorithm.

As with the histogram estimation, for a class $\mathcal{D}$ with at most d sign changes, let $e_d(g, J) = \min_{g'\in\mathcal{D}_J} \|g - g'\|_{A_{d+1}}$. Let $\mathcal{L}$ be the set of candidate intervals $I'_{i,j+1}$ in the set $\mathcal{I}'_{j+1}$ with the largest $\alpha\cdot t$ errors $\|h' - \widehat{f}\|_{\mathcal{A}_{d+1}}$. By the guarantee of projection oracle,

$$\|h' - \widehat{f}\|_{\mathcal{A}_{d+1},I'_{i,j+1}} \leq e_d(\widehat{f}, I'_{i,j+1}) + \frac{\eta}{2\alpha t}.$$

Let $\mathcal{L}_0$ be the intervals in $\mathcal{L}$ that do not contain any jumps of h^*. Since h^* has at most t jumps, $|\mathcal{L}_0| \geq (\alpha - 1)t$.

Therefore,

$$\begin{aligned}\sum_{I' \in \mathcal{L}_0} \|h' - \widehat{f}\|_{\mathcal{A}_{d+1}, I'} &\leq \sum_{I' \in \mathcal{L}_0} \left(e_d(\widehat{f}, I') + \frac{\eta}{2\alpha t} \right) \\ &\leq \sum_{I' \in \mathcal{L}_0} \left(\|h^* - \widehat{f}\|_{\mathcal{A}_{d+1}, I'} + \frac{\eta}{2\alpha t} \right) \\ &\leq \|f - h^*\|_{1, \mathcal{L}_0} + \|f - \widehat{f}\|_{\mathcal{A}_{(d+1)\alpha t}, \mathcal{L}_0} + \eta/2 \\ &\leq \mathrm{OPT}_{\mathcal{D}, t} + \epsilon + \eta/2.\end{aligned}$$

Since h' is h on the interval J, combining with $|\mathcal{L}_0| \geq (\alpha - 1)t$, we obtain

$$\|h' - \widehat{f}\|_{\mathcal{A}_{d+1}, J} = \|h - \widehat{f}\|_{\mathcal{A}_{d+1}, J} \leq \frac{\mathrm{OPT}_{\mathcal{D}, t} + 2\epsilon}{(\alpha - 1)t} + \frac{\eta}{2(\alpha - 1)t} ,$$

completing the proof of the lemma. □

B Additional Omitted Proofs

B.1 Proof of Fact 26

We first require the following classical lemma, first proved by Markov [Mar92]. For completeness, we include an elegant proof by the mathoverflow user fedja[7]. We remark that the bounds in the following fact are essentially tight.

Fact 50 ([Mar92])**.** *Let $p(x) = \sum_{j=0}^{d} c_j x^j$ be a degree-d polynomial so that $|p(x)| \leq 1$ for all $x \in [-1, 1]$. Then $\max_j |c_j| \leq (\sqrt{2} + 1)^d$ for all $j = 0, \ldots, d$.*

Proof. We first claim that $|c_j| \leq \max_{z \in \mathbb{D}} |p(z)|$ where $\mathbb{D}$ is the unit complex disc. To see this, we notice that by Cauchy's integral formula,

$$c_j = \frac{1}{j!} p^{(j)}(0) = \frac{1}{2\pi i} \int_{|\zeta| = 1} \frac{p(\zeta)}{\zeta^{j+1}} d\zeta ,$$

where we also changed the order of differentiation and integration and used

$$\frac{\mathrm{d}}{\mathrm{d}x^j} \frac{p(\zeta)}{\zeta - x} = \frac{j! \cdot p(\zeta)}{(\zeta - x)^{j+1}} .$$

[7] See http://mathoverflow.net/questions/97769/approximation-theory-reference-for-a-bounded-polynomial-having-bounded-coefficie

Therefore, we get

$$\begin{aligned}|c_j| &= \frac{1}{2\pi}\left|\int_{|\zeta|=1}\frac{p(\zeta)}{\zeta^{j+1}}d\zeta\right| \\ &\leq \frac{1}{2\pi}\int_{|\zeta|=1}\left|\frac{p(\zeta)}{\zeta^{j+1}}\right|d\zeta \\ &\leq \max_{|\zeta|=1}|p(z)|\ .\end{aligned}$$

Consider the function

$$F(z) = z^{-m}p\left(\frac{z+z^{-1}}{2}\right)\ .$$

On the domain $\{z : |z| \geq 1\}$, this function is analytic. So by the maximum modulus principle, it is bounded by its value on the unit circle. Since for all $z \in \mathbb{D}$, $(z+z^{-1})/2 = \Re(z)$, we conclude that $|F(z)| \leq \max_{x\in[-1,1]} p(x) \leq 1$ by assumption. Thus we have that

$$p\left(\frac{z+z^{-1}}{2}\right) \leq z^d$$

for all $|z| > 1$. Fix any $w \in \mathbb{D}$. It is straightforward to see that $w = (z+z^{-1})/2$ for some $z \in \mathbb{C}\setminus\{0\}$; by symmetry of z and z^{-1} we conclude that this also holds for some z with $|z| \geq 1$. For each w, arbitrarily choose such a z and denote it z_w. Moreover, for all $|z| > (\sqrt{2}+1)$, we have

$$\begin{aligned}\left|\frac{z+z^{-1}}{2}\right| &\geq \frac{|z|-|z^{-1}|}{2} \\ &> \frac{\sqrt{2}+1-\frac{1}{\sqrt{2}+1}}{2} \geq 1\end{aligned}$$

and thus we conclude that for all $w \in \mathbb{D}$ we have that its corresponding z_w satisfies $|z_w| \leq \sqrt{2}+1$ and therefore $|p(w)| = |p((z_w + z_w^{-1})/2)| \leq z_w^d \leq (\sqrt{2}+1)^d$, as claimed. □

The above statement is for polynomials that are uniformly bounded on $[-1,1]$. We will be interested in bounds for polynomials that integrate to a fixed constant. In order to relate these bounds, we use the following classical result.

Fact 51 (Bernstein's Inequality [Che82])**.** *Let p be a degree-d polynomial and let p' be its derivative. Then*

$$\max_{x\in[-1,1]}|p'(x)| \ \leq\ d^2 \cdot \max_{x\in[-1,1]}|p(x)|\ .$$

With these results, we are now ready to prove Lemma 26.

Proof of Lemma 26. Consider the degree-$(d+1)$ polynomial P such that $P(-1) = 0$ and $P' = p$. This implies that $P(x) = \int_{-1}^{x} p(y)\,\mathrm{d}y$. Since p is non-negative on $[-1,1]$, the bound on $\int_{-1}^{1} p(y)\,\mathrm{d}y$ then gives

$$\max_{x\in[-1,1]}|P(x)| \ \leq\ \alpha\cdot(\sqrt{2}+1)^d\ .$$

Using Bernstein's Inequality (Fact 51), we can convert this bound into a bound on $P' = p$, i.e., we get that $|p(x)| \leq t\cdot(d+1)$ for all $x \in [-1,1]$. Combining this uniform bound on p with Fact 50 gives the desired bound on the coefficients of p. □

B.2 Proof of Lemma 34

Our approach to proving Lemma 34 is relatively straightforward. Assume we had an algorithm A that finds the roots of p exactly. Then one could perform a non-negativity test by running A to find the roots of p', which correspond to the extrema of p. Given the extrema of p, it suffices to check whether p is non-negative at those points and the endpoints of the interval.

However, such an exact root-finding algorithm A does not exist in general. Nevertheless, there are efficient algorithms for finding the approximate roots of p in certain regimes. We leverage these results to construct an efficient non-negativity test. Before we proceed, we remark briefly that we could also utilize the univariate SOS algorithm [Sho87, Las01, Par03], which is arguably more elementary than our approach here, but slower.

Formally, we build on the following result.

Fact 52 ([Pan01], Part II, Theorem 1.1)**.** *Let $\mathbb{D}$ denote the complex unit disc. For all $\nu > 0$, there exists an algorithm* FINDROOTS(q, β) *satisfying the following guarantee: given any degree-d polynomial $q(z) : \mathbb{C} \to \mathbb{C}$ with roots $z_1 \ldots, z_d$ such that $z_i \in \mathbb{D}$ for all i and $\beta \geq d \log d$, returns $z_1^*, \ldots, z_d^*$ so that $|z_j^* - z_j| \leq 2^{2-\beta/d}$ for all j. Moreover,* FINDROOTS *runs in time $O(d \log^2 d \cdot (\log^2 d + \log \beta))$.*

Our polynomials do not necessarily have all roots within the complex unit disc. Moreover, we are only interested in real roots. However, it is not too hard to solve our problems with the algorithm from Fact 52. We require the following structural result:

Fact 53 ([Hen88], Sect. 6.4)**.** *Let $q(x) = x^d + c_{d-1}x^{d-1} + \ldots + c_1 x + c_0$ be a monic polynomial of degree d (i.e., the leading coefficient is 1). Let $\rho(q)$ denote the norm of the largest zero of q. Then*

$$\rho(q) \leq 2 \max_{1 \leq i \leq d} |c_{d-i}|^{1/i} .$$

In order to use the result above, we process our polynomial p so that it becomes monic and still has bounded coefficients. We achieve this by removing the leading terms of p with small coefficients. This then allows us to divide by the leading coefficient while increasing the other coefficients by a controlled amount only. Formally, we require the following definitions.

Definition 54 (Truncated polynomials)**.** *For any degree-d polynomial $p = \sum_{i=0}^{d} c_i x^i$ and $\nu > 0$ let*

$$\Delta = \Delta(p, \nu) = \max \left\{ i : |c_i| \geq \frac{\nu}{2d} \right\} ,$$

and let $\Pi = \Pi_\nu$ be the operator defined by

$$(\Pi p)(x) = \sum_{i=0}^{\Delta(p,\nu)} c_i x^i.$$

Formally, Π acts on the formal coefficient representation of p as $q = \sum c_i x^i$. It then returns a formal representation $\sum_{i=0}^{\Delta(p,\nu)} c_i x^i$. In a slight abuse of notation, we do not distinguish between the formal coefficient representation of p and the polynomial itself. Then Facts 52 and 53 give us the following:

Lemma 55. *There exists an algorithm* FASTAPPROXROOTS(p, ν, μ) *with the following guarantee. Let p be a polynomial as in Definition 34, and let $\nu, \mu > 0$ such that $\nu \leq \frac{1}{2\alpha d}$ (where α and d are as in Def. 34). Then* FASTAPPROXROOTS *returns approximate roots $x_1^*, \ldots, x_{\Delta(p,\nu)}^* \in \mathbb{R}$ so that for all real roots y of $\Pi_\nu p$, there is some j so that $|y - x_j^*| \leq \mu$. Moreover,* FASTAPPROXROOTS *runs in time $O(d \log^2 d \cdot (\log^2 d + \log\log\alpha + \log\log(1/\nu) + \log\log(1/\mu)))$.*

Proof. FASTAPPROXROOTS(p, ν, μ) proceeds as follows. We find $\Delta = \Delta(p, \nu)$ and $\Pi p = \Pi_\nu p$ in time $O(d)$ by a single scan through the coefficients c_i of p. Let $q_1(x) = \frac{1}{c_\Delta}(\Pi p)(x)$. Note that the roots of q_1 are exactly the roots of Πp. Then, by Theorem 53, we have that

$$A \stackrel{\text{def}}{=} 2 \max_{1 \le i \le \Delta} \left| \frac{c_{\Delta - i}}{c_\Delta} \right|^{1/i} \ge \rho(q_1) \; .$$

The quantity A is also simple to compute in a single scan of the c_i. Notice that we have

$$A \le \max \left(2 \max_{1 \le i \le \Delta} \left| \frac{c_{\Delta - i}}{c_\Delta} \right|, 1 \right) \le \underbrace{\frac{2\alpha d}{\nu}}_{B}$$

by the definition of Δ and the assumption that the c_i are bounded by α (Definition 34). Let B denote the right hand side of the expression above. If we let $q(x) = q_1(Ax)$, we have that the roots of q all lie within the complex unit disc. Let $z_1, \ldots, z_\Delta$ be the roots of Πp. Then the roots of q are exactly $z_1/A, \ldots, z_\Delta/A$. Run FINDROOTS$(q, 2d + d \log B + d \log(1/\mu))$, which gives us $z_1^*, \ldots, z_\Delta^*$ so that for all i, we have $|z_i^* - z_i/A| < \mu/B$. Thus, for all i, we have

$$|Az_i^* - z| \le A \frac{\mu}{B} \le \mu \; .$$

FASTAPPROXROOTS(p, ν, μ) returns the numbers $x_i^* = \Re(Az_i^*)$. For any real root x of Πp, there is some z_i^* so that $|Az_i^* - x| < \mu$, and thus $|x_i^* - x| < \mu$ as well. Thus, we output numbers which satisfy the conditions of the Lemma. Moreover, the runtime of the algorithm is dominated by the runtime of FINDROOTS$(q, 2d + d \log B + d \log(1/\mu))$, which runs in time

$$O(d \log^2 d \cdot (\log^2 d + \log(d \log B + d \log(1/\mu)))) =$$
$$O(d \log^2 d \cdot (\log^2 d + \log \log \alpha + \log \log(1/\nu) + \log \log(1/\mu)))$$

This completes the proof. □

Proof of Lemma 34. Let $\nu = \frac{\mu}{2}$, and let $\nu' = \frac{\mu}{4\alpha d(d+1)}$. Set

$$r = (\Pi_\nu p)(x) = \sum_{i=1}^{\Delta(p,\nu)} c_i x^i \; .$$

We can compute the coefficients of r in time $O(d)$. Moreover, $\Pi(r'(x)) = r'(x)$. Let $x_1, \ldots, x_{d'}$, where $d' \le \Delta$, be the roots of $r'(x)$ in $[-1, 1]$. These points are exactly the local extrema of r on $[-1, 1]$. Our algorithm TESTNONNEG(p, μ) then is simple:

1. Run FASTAPPROXROOTS(r, ν', μ) and let $x_1^*, \ldots, x_\Delta^*$ be its output.
2. Let $J = \{i : x_i^* \in [-1, 1]\}$ and construct the set $S = \{-1, 1\} \cup \{x_i : i \in J\}$.
3. Denote the points in S by $x_0 = -1 \le x_1 \le \ldots \le x_{d'-1} \le x_{d'} = 1$, where $d' \le \Delta + 1$.
4. Evaluate the polynomial p at the points in S using the fast algorithm from Fact 41.
5. If at any of these points the polynomial evaluates to a negative number, return that point. Otherwise, return "OK".

The running time is dominated by the call to FASTAPPROXROOTS. By Lemma 55, this algorithm runs in time $O(d \log^2 d \cdot (\log^2 d + \log\log\alpha + \log\log(1/\mu)))$ as claimed.

It suffices to prove the correctness of our algorithm. Clearly, if p is nonnegative on $[-1,1]$, it will always return "OK". Suppose there exists a point $y \in I$ so that $p(y) < -\mu$.

For a function f, and an interval $I = [a,b]$, let $|f|_{\infty,I} = \sup_{x\in I} \|f(x)\|$. Then,

$$\|p - r\|_{\infty,[-1,1]} \;\leq\; \sup_{x\in[-1,1]} \left| \sum_{i=\Delta+1}^{d} c_i x^i \right| \;\overset{(a)}{\leq}\; (d-\Delta)\cdot\frac{\mu}{4d} \;\leq\; \mu/4\,, \tag{19}$$

where the inequality (a) follows from the choice of Δ. Thus $r(y) < -3\mu/4$. Since the points $x_0, x_1, \ldots, x_{d'}$ are extremal for r on I, there exists a $0 \leq j \leq d'$ so that $r(x_j) < -3\nu/4$. If $j = 0$ (resp. $j = m'$), so if $r(-1) < -3\mu/4$ (resp. $r(1) < -3\mu/4$), then by Equation (19), we have $p(-1) < \mu/2$ (resp. $p(1) < -\mu/2$). Thus our algorithm correctly detects this, and the polynomial fails the non-negativity test as intended.

Thus assume $j \in \{1,\ldots,\Delta\}$. By Lemma 55, we know that there is a x^*_ℓ so that $|x^*_\ell - x_j| < \nu'$. Since $x_j \in I$, either $\ell \in J$ or $|x_j + 1| < \nu'$ or $|x_j - 1| < \nu'$, so in particular, there is a point $s \in S$ so that $|x_j - s| < \nu'$. Since for all $x \in [-1,1]$, we have

$$|p'(x)| \leq \sum_{i=1}^{d} \left|ic_i x^i\right| \leq \alpha d(d+1)$$

by the bound on the coefficients of p (see Definition 34). By a first order approximation, we have that

$$|p(x_j) - p(s)| \leq \alpha d(d+1)|x_j - s| \leq \mu/4$$

where the last inequality follows by the definition of ν'. Thus, we have that $p(s) < -\mu/2$, and we will either return s or some other point in $s' \in S$ with $p(s') \leq p(s)$. Thus our algorithm satisfies the conditions on the theorem. □

C Learning discrete piecewise polynomials

Throughout this paper we focused on the case that the unknown distribution has a density f supported on $[-1,1]$, and that the error metric is the L_1-distance with respect to the Lebesgue measure on the real line. We now show that our algorithm and analysis naturally generalize to the case of discrete distributions.

In the discrete setting, the unknown distribution is supported on the set $[N] \stackrel{\text{def}}{=} \{1,\ldots,N\}$, and the goal is to minimize the ℓ_1-distance between the corresponding probability mass functions. The ℓ_1-norm of a function $f : [N] \to \mathbb{R}$ is defined to be $\|f\|_1 = \sum_{i=1}^{N} |f(i)|$ and the ℓ_1-distance between $f, g : [N] \to \mathbb{R}$ is $\|f - g\|_1$.

In the following subsections, we argue that our algorithm also applies to the discrete setting with only minor adaptations. That is, we can agnostically learn discrete piecewise polynomial distributions with the same sample complexity and running time as in the continuous setting. 1

C.1 Problem statement in the discrete setting

Fix an interval $I \subseteq [N]$. We say that a function $p : I \to \mathbb{R}$ is a degree-d polynomial if there is a degree-d real polynomial $q : \mathbb{R} \to \mathbb{R}$ such that $p(i) = q(i)$ for all $i \in I$. We say that $h : [N] \to \mathbb{R}$ is a t-piecewise

degree-d polynomial if there exists a partition of $[N]$ into t intervals so that on each interval, h is a degree-d polynomial. Let $\mathcal{P}^{\text{disc}}_{t,d}$ be the set of t-piecewise degree-d polynomials on $[N]$ which are nonnegative at every point in $[N]$. Fix a distribution (with probability mass function) $f : [N] \to \mathbb{R}$. As in the continuous setting, define $\text{OPT}^{\text{disc}}_{t,d} \stackrel{\text{def}}{=} \min_{g \in \mathcal{P}^{\text{disc}}_{t,d}} \|g - f\|_1$. As before, our goal is the following: given access to n i.i.d. samples from f, to compute a hypothesis h so that probability at least $9/10$ over the samples, we have $\|h - f\|_1 \leq C \cdot \text{OPT}^{\text{disc}}_{t,d} + \epsilon$, for some universal constant C. As before, we let $\widehat{f}$ denote the empirical after taking n samples.

Our algorithms for the continuous setting also work for discrete distributions, albeit with slight modifications. For the case of histogram approximation, the algorithm and its analysis hold verbatim for the discrete setting. The only difference is in the definition of flattening; Definition 8 applies to continuous functions. For a function $f : [N] \to \mathbb{R}$ and an interval $J \subseteq [n]$ the flattening of f on J is now defined to be the constant function on J which divides the total ℓ_1 mass of the function within J uniformly among all the points in J. Formally, if $J = \{a, \ldots, b\}$, we define the flattening of f on J to be the constant function $\bar{f}_J(x) = \frac{\sum_{i \in I} f(i)}{b-a+1}$.

C.2 The algorithm in the discrete setting

Our algorithm in the discrete setting is nearly identical to the algorithm in the continuous setting, and the analysis is very similar as well. Here, we only present the high-level ideas of the discrete algorithm and highlight the modifications necessary to move from a continuous to a discrete distribution.

C.2.1 The $\mathcal{A}_k$-norm and general merging in the discrete setting

We start by noting that the notion of the $\mathcal{A}_k$-norm and the VC inequality also hold in the discrete setting. In particular, the $\mathcal{A}_k$-norm of a function $f : [N] \to \mathbb{R}$ is defined as

$$\|f\|_{\mathcal{A}_k} = \max_{I_1, \ldots, I_k} \sum_{i=1}^{k} |f(I_i)| \ ,$$

where the maximum ranges over all $I_1, \ldots, I_k$ which are disjoint sub-intervals of $[N]$.

The basic properties of the $\mathcal{A}_k$-norm (i.e., those in Lemma 6) still hold true. Moreover, it is well-known that the VC inequality (Theorem 2) still holds in this setting. These properties of the $\mathcal{A}_k$-norm are the only ones that we use in the analysis of GENERALMERGING. Therefore, it is readily verified that the same algorithm is still correct, and has the same guarantees in the discrete setting, assuming appropriate approximate $\mathcal{A}_k$-projection and $\mathcal{A}_k$-computation oracles for polynomials on a fixed subinterval of $[N]$.

C.2.2 Efficient $\mathcal{A}_k$-projection and computation oracles for polynomials

We argue that, as in the continuous setting, we can give efficient $\mathcal{A}_k$-projection and computation oracles for non-negative polynomials of degree d on a discrete interval I, using an $O(d)$-dimensional convex program. By appropriately shifting the interval, we may assume without loss of generality that the interval is of the form $[m] = \{1, \ldots, m\}$ for some $m \leq N$.

The Convex Program As in the continuous case, it can be shown that the set of non-negative polynomials p on $[m]$ satisfying $\|p - \widehat{f}\|_{\mathcal{A}_k} \leq \tau$ is convex (as in Lemma 21), for any fixed $\tau > 0$ (since $\|\cdot\|_{\mathcal{A}_k}$ is a norm). Moreover, using explicit interpolation formulas for polynomials on $[m]$, it is easy to show that every

polynomial in this feasible region has a representation with bounded coefficients (the analogue of Theorem 27), and that the feasible region is robust to small perturbations in the coefficients (the analogue of Theorem 28). Thus, it suffices to give an efficient separation oracle for the feasible set.

The Separation Oracle Recall that the separation oracle in the continuous case consisted of two components: (i) a non-negativity checker (Subsection 6.2), and (ii) a fast $\mathcal{A}_k$-computation oracle (Subsection 6.3). We still use the same approach for the discrete setting.

To check that a polynomial $p : I \to \mathbb{R}$ with bounded coefficients is non-negative on the points in I, we proceed as follows: we use FAST-APPROX-ROOTS to find all the real roots of p up to precision $1/4$, then evaluate p on all the points in I which have constant distance to any approximate root of p. Since p cannot change sign in an interval without roots, this is guaranteed to find a point in I at which p is negative, if one exists. Moreover, since p has at most d roots, we evaluate p at $O(d)$ points; using Fact 41, this can be done in time $O(d \log d \log \log d)$.

Finally, to compute the $\mathcal{A}_k$-distance between $p = \sum_{j=0}^{d} c_j x^j$ and $\widehat{f}$ on an interval I, we use the same reduction as in Section 6.3 with minor modifications. The main difference is that between two points x_i, x_{i+1} in the support of the empirical distribution, the quantity $p[x_i, x_{i+1}]$ (see section 6.3) is now defined to be

$$\begin{aligned} p[x_i, x_{i+1}] &= \sum_{\ell=x_i+1}^{x_{i+1}-1} p(\ell) \\ &= \sum_{\ell=x_i+1}^{x_{i+1}-1} \sum_{j=0}^{d} c_j \ell^j \\ &= \sum_{j=0}^{d} c_j \left(\sum_{\ell=x_i+1}^{x_{i+1}-1} \ell^j \right) . \end{aligned}$$

Notice that the above is still a linear expression in the c_j, and there are simple closed-form expressions for $\left(\sum_{\ell=\alpha}^{\beta} \ell^j\right)$ for all integers α, β and for all $0 \le j \le d$. Following the arguments in Section 6.3 with this substituted quantity, one can show that the quantity returned by APPROXSEPORACLE in the discrete setting is still a separating hyperplane for p and the current feasible set. Moreover, APPROXSEPORACLE still runs in time $\widetilde{O}(d)$.