Fitting Algebraic Curves to Noisy Data

Sanjeev Arora [*] Subhash Khot[†]

December 15, 2002

Abstract

We introduce the following problem which is motivated by applications in vision and pattern detection : We are given pairs of datapoints (x_1, y_1), (x_2, y_2), $\ldots$,$(x_m, y_m) \in [-1, 1] \times [-1, 1]$, a noise parameter $\delta > 0$, a degree bound d, and a threshold $\rho > 0$. We desire an algorithm that enlists every degree d polynomial h such that

$$|h(x_i) - y_i| \leq \delta \qquad \text{for at least } \rho \text{ fraction of the indices } i \tag{1}$$

If $\delta = 0$, this is just the list decoding problem that has been popular in complexity theory and for which Sudan gave a poly(m, d) time algorithm. However, for $\delta > 0$, the problem as stated becomes ill-posed and one needs a careful reformulation (see Introduction).

We prove a few basic results about this (reformulated) problem. We show that the problem has no polynomial time algorithm (our counterexample works for $\rho = 0.5$). This is shown by exhibiting an instance of the problem where the number of solutions is as large as $\exp(d^{0.5-\epsilon})$ and every pair of solutions is far from each other in ℓ_∞ norm. On the algorithmic side, we give a rigorous analysis of a brute force algorithm that runs in exponential time. Also, in surprising contrast to our lowerbound, we give a polynomial time algorithm for learning the polynomials assuming the data is generated using a mixture model in which the mixing weights are "nondegenerate."

[*]Supported by David and Lucile Packard Fellowship, NSF Grant CCR-0098180, and NSF ITR Grant CCR-0205594. Work partially done while visiting the Computer Science Dept at UC Berkeley. 35 Olden Street, Princeton NJ 08540. arora@cs.princeton.edu.

[†]Same funding and address as Arora. khot@cs.princeton.edu

1 Introduction

In this paper, we study the following problem motivated by applications in vision (see Section 5) and also interesting from the viewpoint of learning theory, approximation theory and the list decoding problem in complexity theory.

Definition 1 *(Noisy Polynomial Reconstruction Problem) We are given pairs of datapoints* (x_1, y_1), $(x_2, y_2), \ldots, (x_m, y_m)$, *a noise parameter* $\delta > 0$, *a degree bound* d, *and a threshold* $\rho > 0$. *We have to find every degree* d *polynomial* h *satisfying*

$$|h(x_i) - y_i| \le \delta \qquad \text{for at least } \rho \text{ fraction of the indices } i \tag{2}$$

Such a polynomial will be called a (ρ, δ)*-fitting polynomial.*

In the noise-free case (i.e. $\delta = 0$), this is exactly the list decoding problem for which Sudan gave [17] an algorithm that runs in time poly(m, d) and works even for subconstant values of ρ. However the case we are interested in, i.e. when $\delta > 0$, the problem as stated becomes ill-posed and we need to reformulate it carefully.

An immediate objection to Definition 1 is that there could be uncountably many (ρ, δ)-fitting polynomials. Indeed, if there exists a polynomial that is (ρ, γ)-fitting for some $\gamma < \delta$ then perturbing its coefficients gives uncountably-many polynomials that are (ρ, δ)-fitting. To avoid this technicality, we instead demand a collection $\mathcal{C}$ of polynomials such that every (ρ, δ)-fitting polynomial h is "close" to some polynomial $f \in \mathcal{C}$. The notion of closeness we use in this paper is the closeness in ℓ_∞ norm.[1] Since the data itself is corrupted by noise upto δ, it is reasonable to allow a further margin for approximations. We demand that every (ρ, δ)-fitting polynomial h is 3δ-close in ℓ_∞ norm to some polynomial in the desired collection $\mathcal{C}$.

Another issue of concern is whether one should have some bound on the values x_i, y_i of datapoints. We will assume that $x_i, y_i \in [-1, 1]$. The bound on x-coordinates can be assumed w.l.o.g. by scaling. The bound on the y-coordinates is necessary to make the problem well-defined (see Section 2 for a discussion on this issue). Thus all polynomials in the paper will be required to take values in $[-1, 1]$ over the interval $[-1, 1]$ and we call such polynomials "well-behaved" polynomials.

With this reformulation of the problem, we are ready to describe the main results in the paper. For most of the paper we assume that the data is produced using a *mixture model.* Our algorithms work only in this restricted model. Moreover, the most interesting result in the paper is a negative result (see Theorem 3) that holds in the mixture model and therefore, holds for the general problem as well (i.e. problem posed in Definition (1) along with its reformulation).

Definition 2 *(Mixture Model and the Mixture Learning Problem) In the mixture model for generating datapoints, there is a set of* k *well-behaved polynomials of degree* d *and a noise parameter* δ. *The data is produced by picking a sequence of random points in* $[-1, 1]$ *and for each point, randomly picking the value of one of the* k *polynomials at that point and adding (in an adverserial manner) a noise upto* δ *to this value.*

In the mixture learning problem *the algorithm is given a random sample from the above mixture, and told* k, d, *and* δ. *It has to reconstruct the polynomials in the mixture within* ℓ_∞ *error at most* 3δ.

[1]For two polynomials f and g, their ℓ_∞ distance is defined to be $\|f - g\|_\infty = \sup_{x \in [-1,1]} |f(x) - g(x)|$. Their ℓ_2 distance is defined as $\|f - g\|_2 = \left(\int_{-1}^{1} \frac{|f(x) - g(x)|^2}{\sqrt{(1 - x^2)}} dx \right)^{1/2}$. This definition of ℓ_2 norm is standard in Chebyshev approximation, as explained later.

We will also be interested in the special case of the mixture learning problem when $k = 1$. We state this as the Basic Problem.

DEFINITION 3 *(Basic Problem) There is an unknown well-behaved degree d polynomial $g(x)$. Given m random points $a_1, a_2, \ldots, a_m \in [-1, 1]$ and m values $b_1, b_2, \ldots, b_m$ where $b_i \in [g(x_i) - \delta, g(x_i) + \delta]$, how large must m be before we can efficiently reconstruct $g(x)$ within ℓ_∞ norm 3δ?*

The problems we consider have a natural place in the classical theory of approximations and curve-fitting [12]. The method of least square errors is designed for fitting polynomials to noisy data, but it does not suffice even for the Basic Problem because it uses ℓ_2 noise whereas we assume the noise is adverserial and just that it is bounded in ℓ_∞ norm. In Section 3 we show a procedure for which $O(\frac{d^2}{\delta}\log(d/\delta))$ datapoints suffice. This may be seen as an analogue of least square fit for adverserial ℓ_∞ noise.

We do not know of any efficient classical algorithm or even an approach for the general problem in Definition 1 and its reformulation. The problem seems to require exponential search (especially if $\rho < 1/2$, when a majority of the datapoints may be "outliers") because of the need to identify a sufficiently large subset of datapoints which are all described by the same polynomial, in the sense of (2). In fact, a basic tool of machine vision called the *Hough transform* —invented in 1962 to detect particle tracks in bubble chamber images— takes precisely such a brute force approach [5, 10, 11]. However, it is not clear *a priori* whether an exponential search is inherent, especially in view of Sudan's *list decoding* algorithm [17], which solves the problem in polynomial (in $d, 1/\rho$) time in the noise-free case. In fact, extending Sudan's algorithm was the original motivation of our paper.

Remark : Actually, the reconstruction problem in which the data comes from a mixture model was solved even earlier for the noiseless case by Ar et al.[1]. We emphasize that in Sudan's and Ar et al's paper, the parameter ρ represents the noise whereas for us, the parameter δ represents the noise. We consider Sudan's and Ar et al's results as the noise-free case, i.e. $\delta = 0$.

In this paper we prove some positive and negative results that shed some light on the status of the noisy polynomial reconstruction problem. Our positive results hold only for the mixture learning problem. We first show the following positive result for the Basic Problem.

THEOREM 1
There is a polynomial time algorithm that solves the Basic Problem for degree d polynomials and noise parameter δ by sampling a dataset of size $m = O(\frac{d^2}{\delta}\log(d/\delta))$. The algorithm reconstructs the unknown polynomial within ℓ_∞ error 3δ with high probability over the choice of the sample.

We use this result to give an algorithm for the mixture learning problem.

THEOREM 2
There is an algorithm that learns a mixture of $k = O(1)$ polynomials with degree d and noise parameter δ in time $\exp(O(kd^2\log(d/\delta)/\delta))$. The algorithm reconstructs all the k polynomials in the mixture upto an ℓ_∞ error 3δ with high probability over the choice of the sample. The size of the sample used is $O(k\frac{d^2}{\delta}\log(d/\delta))$.

This theorem uses a brute-force approach and it can be viewed as a rigorous upper bound on the running time of the Hough transform. To the best of our knowledge, the Hough transform had not been precisely analyzed before, though recently, independently of our work, Dasgupta, Pavlov and Singer [3] analysed an efficient version of the Hough transform for $d = 1$ (the running time and the sample size needed in their algorithm is polynomial in $1/\delta$).

The running time of the algorithm in Theorem 2 can be improved to $\exp(O(kd/\delta \log(d/\delta)))$ in the special case when the polynomials output by the procedure only describe the datapoints whose x-coordinates lie within $[-1+\eta, 1-\eta]$ for some small but fixed $\eta > 0$. This condition seems to arise from boundary effects since the algorithm has no information about the curves outside the interval $[-1, 1]$. We call this a *weak learning algorithm* and it appears as Theorem 7.

The most interesting result in the paper is a negative result. We show in Section 7 that even if the data is produced by a mixture of polynomials, there is no efficient analogue of Sudan's algorithm. Specifically, we show that

THEOREM 3
For every $\epsilon > 0$, there exists two explicitly given well-behaved degree d polynomials g_1, g_2 with the following property : Given a sample (of arbitrarily large size) from a mixture of g_1 and g_2, there are $\exp(d^{0.5-\epsilon})$ essentially different polynomials of degree d which fit 50% of the samplepoints (in the sense of (2)). These polynomials are also well-behaved and the ℓ_∞ distance between every pair of these polynomials is close to 2. This result holds for a noise parameter $\delta > \frac{1}{d^\epsilon}$.

Thus if our goal is to output every polynomial that fits the data, we cannot hope to do it in time less than $\exp(d^{0.5-\epsilon})$. This "bad" example works as long as $\delta > \frac{1}{d^{0.5-\epsilon}}$. We also consider the case when δ is much smaller, say $\delta = 1/d^c$ for an arbitrary constant c independent of d. In Section 8 we extend our bad example to this case; the data is again produced by a mixture of two polynomials and $\exp(d^\epsilon)$ many polynomials fit 50% of the datapoints.

Finally, we note a couple of reasons why our counterexamples in Sections 7 and 8 need not be cause for undue pessimism. First, our bad examples leave open the possibility of an efficient algorithm that outputs at least one (as opposed to *all*) polynomial that describes the data. This may be useful in practice. Second, our counterexample is very degenerate: in practice, the polynomials may be nondegenerate (it is not yet clear what this means) and efficient reconstruction may still be possible. To illustrate this, we give one polynomial-time algorithm in the paper, a weak analogue of the Ar et al. result [1]: We consider a generalization of the mixture model where the polynomials $(g_i)_{i=1}^k$ in the mixture have weights $(w_i)_{i=1}^k$ associated with them and one picks the polynomial g_i with probability w_i for evaluation at any point. Our algorithm reconstructs a mixture of any $k = O(1)$ polynomials when the mixing weights satisfy a nondegeneracy condition. If $k = 2$ the condition specializes to saying that the mixing weights are somewhat different, say 51% and 49%. For general k, if $\{w_1, w_2, \ldots, w_k\}$ are the (unknown) mixing weights, then for every pair of subsets S_1, S_2 of $\{1, \ldots, k\}$, we require that $\sum_{i \in S_1} w_i$ and $\sum_{i \in S_2} w_i$ differ by at least $1/d^c$ for some known c. For instance, this condition is true with high probability for a random set of mixing weights.

1.1 The techniques used

Algorithms for the noise-free case such as those of Berlekamp-Welch [19] ($\delta = 0$, $\rho > 1/2$) and Sudan's list decoding [17] ($\delta = 0$ and ρ can be less than $1/2$) use notions such as "roots of the polynomial", "divisibility of one polynomial by another," etc. that do not hold up well in the noisy case. For instance, the algorithms use the fact that a nonzero degree d polynomial has at most d roots. An analogue of this fact in the noisy case would be "a nonzero degree d polynomial is far from zero most of the time." Unfortunately, this is not true: x^d, though it only has 1 root, is very close to 0 when $|x| < 1 - \eta$ for any fixed η and nevertheless takes the value 1 at $x = 1$. Luckily, representing polynomials in the Chebyshev basis removes a lot of these difficulties, as we will see in Section 3. We will also need basic ideas of approximation theory. The algorithms will use Bernstein-Markov upperbound on derivatives of polynomials (Theorem 5) and the lowerbounds/counterexamples will use results concerning conditions under which "smooth" functions can be well-approximated by degree d polynomials.

2 Basic definitions

For a polynomial $p(x) : [-1, 1] \to \Re$ let $\|p(x)\|_\infty = \sup_{x\in[-1,1]} \{|p(x)|\}$. A polynomial $p(x)$ is called "well-behaved" if $\|p(x)\|_\infty \leq 1$.

DEFINITION 4 *A polynomial $q(x)$ is a* γ-approximation *to p if $\|p - q\|_\infty \leq \gamma$.*

We will assume that the data is generated by a mixture model as in Definition 2. The polynomials $g_1, g_2, \ldots, g_k$ in the mixture are assumed to be "well-behaved" degree d polynomials. This assumption is necessary to make the mixture learning problem well-defined. One cannot expect an algorithm to solve the problem for a fixed value of δ without any restriction on the ℓ_∞ norm of the polynomials. An alternative would be, if the ℓ_∞ norm is a then assume the noise parameter to be $\delta \cdot a$.

Note that the mixture model defined in Definition 2 assumes equal mixing weights for the polynomials in the mixture. In general we could allow weights $w_1, w_2, \ldots, w_k$ where $\sum_i w_i = 1$ (i.e., when picking which polynomial to use at a particular point, the ith polynomial g_i is picked with probability w_i). All our algorithms work with any general mixing weights, and our lower-bounds/counterexamples use equal mixing weights.

As mentioned in Definition 2, The goal will be to recover a set of polynomials containing, for each g_i, a polynomial f_i that is a 3δ-approximation to g_i.

3 The basic problem

Before considering how to reconstruct the mixture of polynomials, let us consider the basic problem defined in Definition 3, which seeks a noisy analogue of the standard fact that a degree d polynomial can be reconstructed from its value at any $d + 1$ points.

We have not yet found a satisfactory analysis of this problem in the literature. Note that we are not restricting attention to any specific reconstruction algorithm (say, Lagrange interpolation), so this is not a standard problem of sensitivity analysis.

The following approach suggests itself for reconstructing the polynomial $g(x)$ in the Basic Problem : Given the datapoints $(a_j, b_j)_{j=1}^m$, denote the unknown coefficients of $g(x)$ by $c_0, c_1, \ldots, c_d$ and solve the following linear program.

$$b_j - \delta \leq \sum_{i=0}^{d} c_i a_j^i \leq b_j + \delta \qquad j = 1, 2, \ldots, m \tag{3}$$

Note that the LP is feasible since the coefficients of g satisfy it. (Remark: Note also that if we didn't know δ explicitly we could just make δ a variable and find the smallest δ for which the LP is feasible.)

Unfortunately, feasible solutions to this LP need not give 3δ-approximations to g even if m is large. The LP only approximates the unknown polynomial on the given points and not everywhere in the interval [-1,1]. We need to somehow incorporate the constraint that the unknown polynomial must take values in $[-1, 1]$. We do not yet know how to do this directly but we can do it indirectly once we move to the Chebyshev representation of the polynomial. A brief introduction to Chebyshev representation is as follows (see [16, Chapter 2]): For each degree n, the Chebyshev polynomial $T_n(x)$ is defined as $T_n(x) = \cos(n\ \cos^{-1}(x))$ where $T_0 = 1/\sqrt{2}$. The Chebyshev polynomials are orthonormal functions with respect to the following inner product

$$\langle f, g \rangle = \frac{2}{\pi} \int_{-1}^{1} \frac{f(x)g(x)}{\sqrt{1-x^2}} dx \tag{4}$$

(The integral exists when f, g are polynomials.) Thus every degree d polynomial has a unique representation as $\sum_{i=0}^{d} c_i T_i(x)$. The ℓ_2 norm of a function is $\|f\|_2 = \sqrt{\langle f, f \rangle}$; the ℓ_2 norm of the polynomial $\sum_{i=0}^{d} c_i T_i(x)$ turns out to be $(\sum_{i=0}^{d} c_i^2)^{1/2}$. (Thus in particular, the constant polynomial $f(x) = \alpha$ has ℓ_2 norm $\sqrt{2}\,|\alpha|$.)

Intuitively speaking, the advantage of the Chebyshev representation is that a polynomial has a "small" coefficient vector iff $\|f\|_2$ is small. Note that the unknown polynomial takes values in $[-1, 1]$ and hence its ℓ_2-norm is at most $\sqrt{2}$. In particular, each of its Chebyshev coefficient is at most $\sqrt{2}$.

Let I be a set of d^6 equally spaced points that cover the interval $[-1, 1]$. We construct a new LP as follows : The variables are the Chebyshev coefficients $c_0, c_1, \dots, c_d$ of the unknown polynomial. We have constraints as in (3) and we add extra constraints that bound each of the Chebyshev coefficient c_i and enforce the condition that the polynomial takes values in [-1,1] for all points in the set I. Thus the LP is :

$$\begin{array}{rll} b_j - \delta \leq & \sum_{i=0}^{d} c_i T_i(a_j) \leq b_j + \delta & j = 1, 2, \dots, m \\ & |c_i| \leq \sqrt{2} & i = 0, 1, \dots, d \\ & |\sum_{i=0}^{d} c_i T_i(x)| \leq 1 & \forall\, x \in I \end{array}$$

The following Theorem is sufficient to prove Theorem 1.

THEOREM 4
Let $m = \Omega(\frac{d^2}{\delta} \log(d/\delta))$ and let h be the polynomial obtained from taking any feasible solution to the above LP. With probability at least $1/2$ (over the choice of the random sample), h is a 3δ-approximation to the unknown polynomial.

We need the Bernstein-Markov inequality (see [6]) which we state below.

THEOREM 5 (BERNSTEIN-MARKOV INEQUALITY)
For every degree d polynomial $f : [-1, 1] \to \Re$

$$\|f'\|_\infty \leq \|f\|_\infty \cdot d^2$$

$$|f'(x)| \leq \frac{d}{\sqrt{1-x^2}} \cdot \|f\|_\infty \quad \forall x \in (-1, 1)$$

Now we prove Theorem 4.

PROOF:
First we claim that $\|h\|_\infty \leq 1 + d^3/|I| \leq 2$. Since $h(x) = \sum_{i=0}^{d} c_i T_i(x)$, we have

$$|h'(x)| \leq \sum_{i=0}^{d} |c_i||T_i'(x)| \leq \sum_{i=0}^{d} \sqrt{2} i^2 \leq d^3$$

where we used the fact that $|T_i'(x)| \leq i^2$ for $x \in [-1,1]$ (this follows from the Bernstein-Markov inequality since $\|T_i(x)\|_\infty = 1$). By construction, h takes values in $[-1,1]$ for all points in I, and the distance between successive points of I is $2/|I|$. Now the claim follows from the fact that the derivative h' by definition gives the rate of change in h.

We showed that $\|h'\|_\infty \leq d^3$, however using Bernstein-Markov inequality again and the bound $\|h\|_\infty \leq 2$, we get a stronger bound of $\|h'\|_\infty \leq 2d^2$.

Now we are ready to finish the proof of the Theorem. Let ϵ denote the largest distance between two successive points out of $a_1, a_2, \ldots, a_m$. That is, every interval of size ϵ contains at least one of the points. (The points form an ϵ-net.) A simple calculation shows that the probability is high that $\epsilon = O(\log m/m) = O(\delta/d^2)$.

Now h and g are functions satisfying $\|g'\|_\infty, \|h'\|_\infty \leq 2d^2$, and hence $\|(h-g)'\|_\infty \leq 4d^2$. But h, g differ by at most 2δ on the points in the ϵ-net, so $\|(h-g)\|_\infty \leq 2\delta + 4\epsilon d^2$, which is at most 3δ when the constants are appropriately chosen.

□

We state the following corollary of the proof of Theorem 4.

Corollary 6

In the setting of Theorem 4, if the unknown polynomial g satisfies $\|g'\|_\infty = C$ then $m = \Omega(C/\delta \log(C/\delta))$ random samples suffice.

Proof: Since $\|g'\|_\infty = C$, we can add the following constraints to the LP :

$$|\sum_{i=0}^{d} c_i T_i'(x)| \leq C \quad \forall\ x\ \in I$$

The bound $\|h'\|_\infty \leq 2d^2$ still holds. We then apply Bernstein-Markov inequality to h' and get the bound $\|h''\|_\infty \leq 2d^4$. This bound and the fact that $|h'(x)| \leq C \quad \forall\ x \ \in I$ together imply the stronger bound $\|h'\|_\infty \leq C + 2d^4/|I| \leq 2C$. A random sample of $m = \Omega(C/\delta \log(C/\delta))$ points forms a $\frac{\delta}{5C}$-net and h differs by at most δ from the unknown polynomial g on these sample points. Hence $|h(x) - g(x)| \leq 3\delta$ everywhere in $[-1,1]$. □

Other classes of polynomials. We note that the Bernstein-Markov inequality is tight in general, but many interesting classes of polynomials are known for which the derivative has a tighter upperbound. For these polynomials, Corollary 6 gives better upperbound on m. Erdös [7] shows that if g has all its zeroes in $\Re \setminus (-1,1)$ then $\|g'\|_\infty = O(d)$. Erdelyi [6] has shown that the same bound holds if the polynomial has at most $O(1)$ zeroes in the unit disk in the complex plane. If the polynomial has form $\sum_i a_i x^i$ where $|a_i| \leq 1$ then $\|g'\|_\infty \leq O(d^{3/2})$.

We also note that we can generalize our algorithm to learn curves of the type $y^c = g(x)$, where c is some constant. (For instance, a circle in the plane is described by $y^2 = a^2 - x^2$ for some a.) For each datapoint (a_i, b_i) we just change it to (a_i, b_i^c) and then run our algorithm. This gives, for instance, a rigorous analysis of the Hough transform for such curves.

4 Learning mixtures of polynomials

The algorithms to learn the mixture of polynomials are easily derived from the results of the previous section. We prove Theorem 2 in this section.

In order to reconstruct all k polynomials in the mixture, we take a random sample of $M \geq \Omega((kd^2 \log(d/\delta))/\delta)$ points in $[-1,1]$. With high probability, there are $m \geq \Omega((d^2 \log(d/\delta))/\delta)$ samples from each polynomial. Now exhaustively enumerating all subsets of size $\binom{M}{m}$ samplepoints and applying the reconstruction procedure of the previous section to each subset, we are guaranteed to find 3δ-approximations to all polynomials in the mixture.

Note that this is basically an implementation of the Hough transform, except it is backed by our formal analysis of the reconstruction procedure of the previous section.

Remark 1: If the mixing weights of the polynomials are not equal but are $w_1, w_2, \dots, w_k$ then the number of samples needed is $M = \Omega((kd^2 \log(d/\delta))/w_{\min}\delta)$ samples where $w_{\min} = \min_i w_i$. Taking this many samples ensures that with high probability there are at least $\Omega((d^2 \log(d/\delta))/\delta)$ random samples from each polynomial.

Remark 2: There will be "false" positives in general: the algorithm finds polynomials that fit the data even though they were not in the mixture. This is unavoidable, as we show in Section 7.

4.1 Better algorithm for weak learning

Now we consider the weakening of the mixture learning problem in Definition 2 in which the polynomials have to fit the unknown polynomials only in the interval $[-1+\eta, 1+\eta]$ for some small constant $\eta > 0$. Let us call this *weak learning.*

THEOREM 7
Weak learning is possible in $\exp(O(\frac{kd}{\delta}\log(d/\delta)))$ *time when the data is produced using a mixture of k polynomials.*

PROOF: We proceed in a similar manner as in the proof of Theorem 4. However, when restricted to the interval $[-1+\eta, 1-\eta]$, the Bernstein-Markov inequality gives a better bound on the derivatives and we get $|g'(x)|, |h'(x)| \leq \frac{2d}{\sqrt{\eta}}$ when $x \in [-1+\eta, 1-\eta]$. So it suffices to take $m = O((\frac{d}{\delta\sqrt{\eta}}\log(d/\delta)))$ and $M = O((k\frac{d}{\delta\sqrt{\eta}}\log(d/\delta)))$ in the proof of Theorem 2. □

5 Applications to machine vision and pattern detection.

Possible applications to machine vision and feature detection motivated our study. Much work in vision is influenced by the so-called *Marr paradigm* [13], also called *inverse optics.* The idea is to roughly mimic the human vision system, which appears to understand scenes by first detecting low-level features (shape, texture) and then progressing to high-level recognition. Note that everyday scenes often have simple descriptions —for instance, artists have no problems depicting a scene with a few pencil strokes. But such simple descriptions seem hard for computers to come by.

Clearly, improving machine vision will entail in improvement in algorithms at all levels. The problems studied in this paper are related to low-level vision: how to detect shapes. Current approaches to this problem start by detecting *edges*, defined roughly as regions of local discontinuity (in light intensity, color, etc). Hueckel [9], Marr and Hildreth [14], and Canny [2] are some landmarks in the design of edge detection algorithms.

After detecting edges, one may try to piece together such local data into a more global (and hopefully, more succinct) description. One promising idea is to identify *low degree algebraic curves* in the data. These curves (boundaries of roads, doors, houses, cars) often occur in scenes [2]. Of

[2] Of course, the true curve may not be a polynomial or even algebraic, but one could consider its Chebyshev approximation.

course, a description in terms of polynomial curves would necessarily need to be approximate to allow for noise and also piecewise (e.g., the curve of the road may be occluded by k foreground objects, and so appear as a union of $k+1$ segments). Note that finding such curves may be useful not just in understanding the scene, but also in drawing a quick sketch of the pixel data, which may be useful for compression, organizing a graphics library for quick searching, etc. Such global curve-fitting in the presence of noise seems difficult because of the exponential nature of the Hough transform, and the research in this paper stemmed from trying to understand if exponential search is inherent. Unfortunately, this necessarily involves considering the degree d to be "large," whereas even the case $d = 3$ or 4 is very interesting in practice.

However, we do point out that our brute-force algorithms have a property desirable for vision applications, namely, they continue to work even when part of the curve is hidden due to occlusions. Specifically, data might be missing for one of the polynomials in some interval $[u, v]$. (Thus far we had assumed that we are given values of the polynomials at random points in $[-1, 1]$, and hence every reasonably large subinterval contains at least one of the points.)

We remark that it is not possible in general to reconstruct the polynomial in the interval for which data is unavailable; see Section 7.1. Thus we only try to reconstruct the polynomial in intervals where data is available.

In the setting of Theorem 4, this corresponds to the situation where the a_i's lie in some sequence of nonoverlapping intervals $[l_1, j_1], [l_2, j_2], .., [l_s, j_s]$ (think of s as a constant, and the length of each interval as at least $1/\sqrt{d}$ or so, so that a random sample of points is an ϵ-net for them). The goal is to find a polynomial which describes g only in these subintervals. Note that we have to assume that g is well-behaved in the entire interval $[-1, 1]$.

The proof of the theorem clearly generalizes to this setting and it shows that h and g agree within 3δ in the intervals for which information is available.

6 Mixture learning when mixing weights are nondegenerate

In this section, we first describe a polynomial time algorithm to reconstruct a mixture of two well-behaved polynomials (g_1, g_2) where the mixing weights are somewhat different, i.e. bounded away from $1/2$.

THEOREM 8
There exists a polynomial time algorithm to learn a mixture of two polynomials when their mixing weights are bounded away from $1/2$.

PROOF: Assume that the mixing weights of the two polynomials $g_1(x)$ and $g_2(x)$ are 60% and 40%, the general case would follow similarly.

The algorithm is very simple given our analysis in Section 3. For any point $x \in [-1, 1]$ consider a small interval $[x - \frac{\delta}{d^3}, x + \frac{\delta}{d^3}]$. For a sample of $d^4 \log(1/\delta)/\delta$ points, with high probability, $\Omega(\log d)$ of the samples lie in this interval. We are given that 60% of these sample points give an approximate value of $g_1(x)$, i.e. the value lies in the interval $[g_1(x) - \delta, g_1(x) + \delta]$ and 40% of the samples give a value in the interval $[g_2(x) - \delta, g_2(x) + \delta]$. We note here that since the derivatives are bounded by $O(d^2)$, the value of the two polynomials is essentially constant over the interval $[x - \frac{\delta}{d^3}, x + \frac{\delta}{d^3}]$. Hence 60% of the values seen in this interval will lie in $[g_1(x) - \delta, g_1(x) + \delta]$ and 40% of them lie in $[g_2(x) - \delta, g_2(x) + \delta]$. If the two intervals overlap, we assume $g_1(x)$ and $g_2(x)$ are the same. If the two do not overlap, we call the one containing 60% of the values as $g_1(x)$ and the one containing 40% of the points as $g_2(x)$. (This is where we use the assumption that the mixing weights are not too similar; it breaks the symmetry between the two polynomials.) Thus at every point x we can reconstruct both $g_1(x)$ and $g_2(x)$. The sample is large enough so that we can reconstruct the values of the two polynomials at say, $\Omega(d^2/\delta)$ equally spaced points. Now applying the techniques from Section 3 enables us to recover the polynomials g_1, g_2. $\square$

We can generalize this theorem to a mixture of any $k = O(1)$ polynomials provided the mixing weights satisfy the following non-degeneracy condition.

DEFINITION 5 *A set of mixing weights $\{w_1, w_2, \ldots, w_k\}$ is called non-degenerate if each weight is at least $1/d^c$ and for every two distinct subsets S_1, S_2 of $\{1, \ldots, k\}$, the sums $\sum_{i \in S_1} w_i$ and $\sum_{i \in S_2} w_i$ differ by at least $1/d^c$. Here $c > 1$ is known.*

THEOREM 9
There exists a poly$(d, 1/\delta)$ time algorithm to learn a mixture of $k = O(1)$ polynomials when their mixing weights are non-degenerate.

PROOF: First we describe a polynomial-time algorithm that solves the reconstruction problem assuming the weights are nondegenerate and known to the algorithm. From this the result will follow since the algorithm can in poly(kd) time try all choices of mixing weights which are multiples of $1/kd^c$ and proceed assuming that those are the true mixing weights. When it hits upon the correct mixing weights, it will succeed.

The main use of the nondegeneracy condition will be that knowing the sum of weights in an unknown subset of $\{1, \ldots, k\}$ allows us to find the subset itself. Thus this sum of weights is a unique "signature" for the subset. Formally, for each subset $S \subseteq \{1, \ldots, k\}$ we call $w_S = \sum_{i \in S} w_i$ the signature of S.

The following is the algorithm assuming the mixing weights are known and nondegenerate. As in the algorithm for $k = 2$, we generate for each point x a set of $s \leq k$ values that are observed in the data whose x-coordinate is in some small interval around x. Each value has a certain frequency. In the above $60 - 40$ example, the possible frequencies could only be 40%, 60%, or 100%. More generally the frequencies correspond to subsets of polynomials that take approximately the same value at that point. The nondegeneracy condition allows us to associate a subset with each frequency; this is just the signature of the subset.

Thus for each polynomial in the mixture we know its value (upto δ) at *poly*(d) points and we can apply our algorithm for the Basic Problem. $\square$

7 A counter-example

In this section we prove Theorem 3. We show that given a noisy data which is known to come from a mixture of two degree d polynomials, there could be $2^{d^{0.5-\epsilon}}$ pairs of polynomials (f, g) that can explain the data within noise $\delta = 1/d^\epsilon$. Thus the number of "correct answers" for the polynomial reconstruction problem is superpolynomial and hence a polynomial-time algorithm cannot enumerate them all. For concreteness, the counterexample below shows this for $\epsilon = 1/6$ but it generalizes to any fixed $0 < \epsilon < 1/2$.

We define piecewise continuous functions $f_i, 0 \leq i \leq \frac{1}{3}d^{1/3} - 1$ as follows. The support of the function f_i is $[\frac{3i+1}{d^{1/3}}, \frac{3i+2}{d^{1/3}}]$. So the functions f_i have disjoint supports. We define

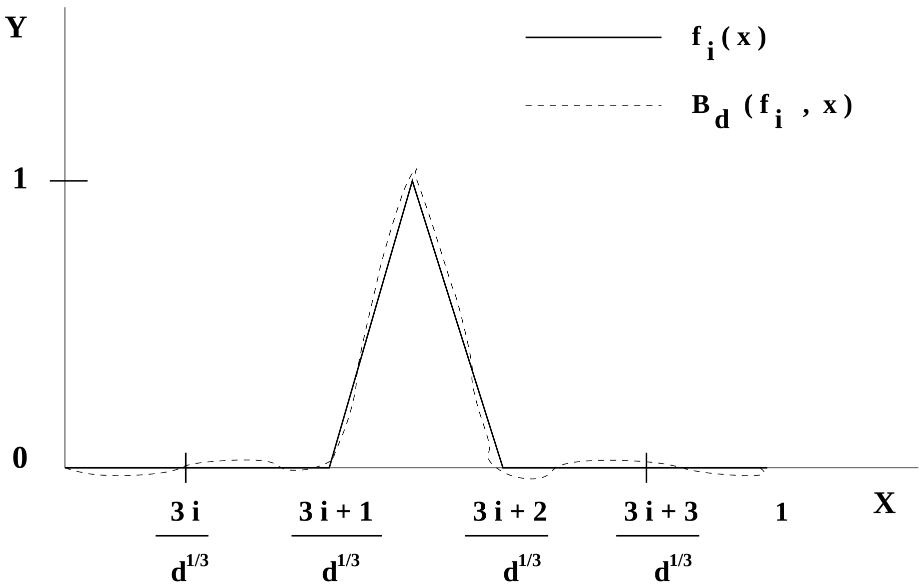

Figure 1: The functions f_i and their approximations $B_d(f_i, x)$

$$f_i(x) = \begin{cases} 0 & \text{if } x \leq \frac{3i+1}{d^{1/3}} \\ 2d^{1/3}(x - \frac{3i+1}{d^{1/3}}) & \text{if } \frac{3i+1}{d^{1/3}} \leq x \leq \frac{3i+\frac{3}{2}}{d^{1/3}} \\ 1 - 2d^{1/3}(x - \frac{3i+\frac{3}{2}}{d^{1/3}}) & \text{if } \frac{3i+\frac{3}{2}}{d^{1/3}} \leq x \leq \frac{3i+2}{d^{1/3}} \\ 0 & \text{if } x \geq \frac{3i+2}{d^{1/3}} \end{cases}$$

The data is generated as follows: pick a random point x, and check if there is an i such that x lies in the support of f_i. If so, output $f_i(x)$ and $-f_i(x)$. Otherwise output 0. We claim that this data can be fitted (in the sense of (2) and $\rho = 50\%$) with a degree d polynomial within a noise parameter $1/d^{1/6}$. In fact, we show that $2^{\Omega(d^{1/3})}$ such polynomials exist.

We use the proof of the classical Weirstrass approximation theorem using Bernstein polynomials.

Definition 6 *For a continuous function $f : [0,1] \to \Re$, and integer d, the Bernstein polynomial $B_d(f,x)$ is defined as*

$$B_d(f,x) = \sum_{k=0}^{d} f(\frac{k}{d}) \binom{d}{k} x^k (1-x)^{d-k}$$

We use the following approximation theorem ([16, page 30]).

Theorem 10
If $f : [0,1] \to \Re$ satisfies the Lipschitz condition $|f(x) - f(y)| \leq L|x-y|$ then the Bernstein polynomials uniformly converge to f. Specifically

$$|f(x) - B_d(f,x)| \leq \frac{L}{\sqrt{d}} \quad \forall\, x \in [0,1]$$

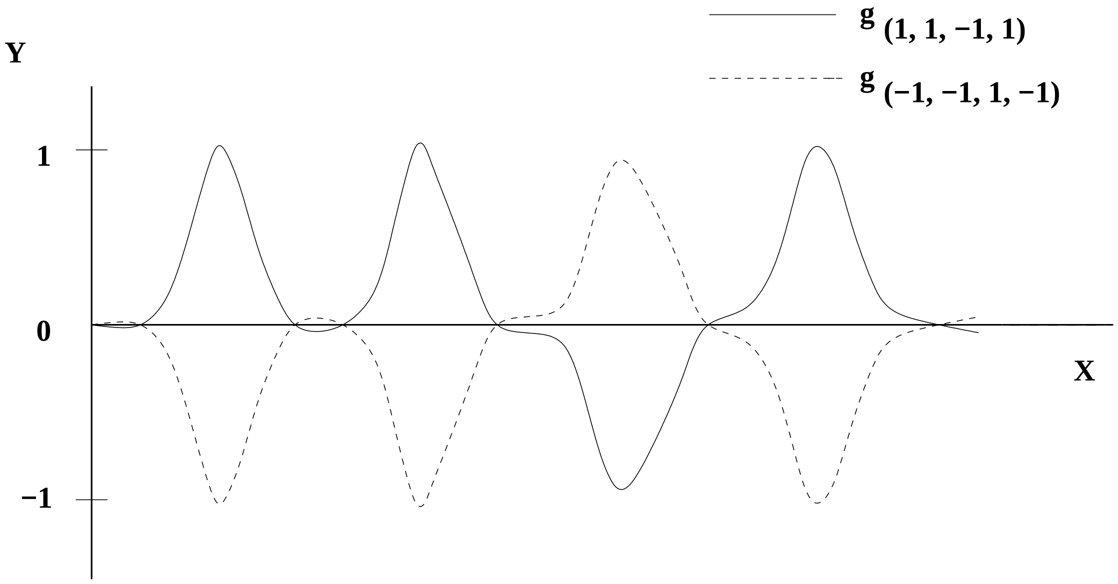

Figure 2: The polynomials $g_{\mathbf{v}}$ and $g_{-\mathbf{v}}$ for $v = (1, 1, -1, 1)$

Remark: The $\sqrt{d}$ appearing here is the reason why this particular counterexample works only for $\epsilon < 1/2$.

Clearly the f_i's defined above satisfy the Lipschitz condition with $L = O(d^{1/3})$. Hence the polynomial $B_d(f_i, x)$ approximates f_i everywhere within $O(\frac{1}{d^{1/6}})$. We will show that $B_d(f_i, x)$ approximates f_i much better outside the interval $[\frac{3i}{d^{1/3}}, \frac{3i+3}{d^{1/3}}]$. Note that f_i is zero outside this interval. We show that $|B_d(f_i, x)| \leq \exp(-\Omega(d^{1/6}))$ outside this interval. This is a simple application of Chernoff bound. Note that

$$f_i(x) - B_d(f_i, x) = \sum_{k=0}^{d} \left[f_i(x) - f_i(\frac{k}{d}) \right] \binom{d}{k} x^k (1-x)^{d-k}$$

For $x \leq \frac{3i}{d^{1/3}}$ we get (the case $x \geq \frac{3i+3}{d^{1/3}}$ follows similarly)

$$|B_d(f_i, x)| \leq \sum_{k=d^{2/3}(3i+1)}^{d} 2\binom{d}{k} x^k (1-x)^{d-k}$$

The right side expression is twice the probability that there are more than $d^{2/3}(3i+1)$ heads in d tosses of a coin which has probability $x \leq \frac{3i}{d^{1/3}}$ of turning up head. A standard application of Chernoff bound gives the result.

Thus the polynomials $B_d(f_i, x)$ are essentially zero outside the interval $[\frac{3i}{d^{1/3}}, \frac{3i+3}{d^{1/3}}]$. Within the interval they have a dome-like shape with height 1. The polynomials $B_d(f_i, x)$ for different i are essentially non-overlapping.

Let $r = \frac{1}{3}d^{1/3} - 1$. Now for a vector $\mathbf{v} \in \{-1, 1\}^r, \mathbf{v} = (v_0, v_1, \ldots v_{r-1})$, define the polynomial

$$g_{\mathbf{v}} = \sum_{j=0}^{r-1} v_j \cdot B_d(f_j, x)$$

Clearly, the polynomials $g_{\mathbf{v}}$ are degree d polynomials. Now consider the mixture of polynomials $g_{\mathbf{v}}$ and $g_{-\mathbf{v}}$. Together, they describe the data within an error $O(1/d^{1/6})$. The same is true for all $\mathbf{v}$. Thus there are $2^r = 2^{\Omega(d^{1/3})}$ correct pairs of polynomials (g, g') which can correctly "explain" the mixed data.

7.1 Counterexample for occluded curves

Consider the following twist to the Basic Problem in Definition 3: there is an interval $[a, b] \subseteq [-1, 1]$ of size at least $O(\frac{1}{\sqrt{d}})$ such that g does not fit any of the points in this interval. (In other words, the noise in this interval is not bounded by δ.) Then we show that solution is not uniquely defined inside $[a, b]$.

The proof consists of noticing that there are degree d polynomials that are very far from zero inside $[a, b]$ but are exponentially close to 0 outside $[a, b]$. Such polynomials can be constructed in a similar manner as the polynomials $B_d(f_i, x)$. We can add any such polynomial to a solution to get another solution that fits the data just as well (since the data excludes points in $[a, b]$).

This justifies the assumption in the mixture model that the set of points that correctly describe the unknown polynomial is an ϵ-net for some small ϵ. It also justifies the assumption in Section 5 that the polynomial is required to fit the data only in the intervals in which correct data is available.

8 A stronger counter-example

The counter-example presented in the last section cannot be pushed through when the additive noise δ is smaller than $\frac{1}{\sqrt{d}}$. In this section, we present a stronger counter-example where δ is exponentially small in d. We show that if the data is generated from a single degree d polynomial and an additive noise of δ is added to it where $\delta = \exp(-d^\epsilon)$ then there could be $\exp(d^\epsilon)$ polynomials that fit 50% of the data for some small constant $\epsilon > 0$.

Let $r = d^\epsilon$ and consider the polynomial $f(x) = T_r(x)^{r+1}$ where $T_r(x)$ is the Chebyshev polynomial of order r. It is well-known that the polynomial $T_r(x)$ has r distinct zeroes $a_1, a_2, \ldots, a_r$ in $(-1, 1)$. These are precisely the zeros of $f(x)$ and at these points the first r derivatives of $f(x)$ vanish as well. Lets define $a_0 = -1$ and $a_{r+1} = 1$. For a vector $v \in \{-1, 1\}^r$ define a function $f_{\mathbf{v}}(x)$ as

$$f_{\mathbf{v}}(x) = (-1)^{v_i} f(x) \text{ for } x \in [a_i, a_{i+1}] \tag{5}$$

Since the first r derivatives of $f(x)$ vanish at its zeroes, the function $f_{\mathbf{v}}(x)$ is r-times differentiable. We show that each of the 2^{r+1} curves $\{f_{\mathbf{v}} : v \in \{-1, 1\}^{r+1}\}$ can be approximated very well with a degree d polynomial. This is shown using the following theorem from [4, Chapter 7, Theorem 6.2].

DEFINITION 7 *For any function $f : [-1, 1] \to \Re$, define its modulus of continuity as*

$$\omega(f, t) = \sup_{|x-y| \leq t} |f(x) - f(y)|$$

THEOREM 11
If a function f is r-times differentiable everywhere in $[-1, 1]$ then there is a constant C_r depending upon r such that error of the best degree d approximation to it satisfies

$$E_d(f) \leq C_r d^{-r} \omega(f^{(r)}, 1/d)$$

where $f^{(r)}$ denotes the r^{th} derivative of f. We have $C_r \leq 2^{O(r)}$.

We apply this theorem to the function $f_{\mathbf{v}}$. Using Theorem 5 and a simple induction, one can show that the r^{th} derivative of $f(x) = T_r(x)^{r+1}$ is at most $r^{O(r)}$ which is also a bound on the r^{th} derivative of $f_{\mathbf{v}}$. This gives

$$\omega(f_{\mathbf{v}}^{(r)}, 1/d) \leq \frac{1}{d}\, r^{O(r)}$$

Recall that $r = d^{\epsilon}$. For a sufficiently small $\epsilon > 0$ we have

$$\begin{aligned} E_d(f_{\mathbf{v}}) &\leq C_r d^{-r} \omega(f_{\mathbf{v}}^{(r)}, 1/d) \\ &\leq 2^{O(r)} d^{-r} \frac{1}{d}\, r^{O(r)} \\ &\leq \exp(-d^{\epsilon}) \end{aligned}$$

All the $2^{d^{\epsilon}}$ pairs $(f_{\mathbf{v}}, f_{-\mathbf{v}})$ describe the same data, finishing the proof.

9 Conclusions

We have tried to present an interesting new problem which seems to be of interest in many applied areas, in addition to being an interesting new problem for the classical theory of curve fitting.

Our counterexamples of Sections 7 and 8 suggest that the problem as stated has no polynomial time algorithm, even when the data is generated by a mixture of polynomials. However, we do not see a reason for pessimism yet. For one thing, the counterexamples really rely on mixing weights being equal, an unrealistic assumption in practice (indeed, when the mixing weights are "nondegenerate", we can do the reconstruction efficiently using the algorithm of Section 6).

All this assumes that the data is generated from a mixture of polynomials. If the data is unstructured, then our counterexample really gives reason for pause, and it appears that no analogue of Sudan's algorithm —which does work for unstructured data—can exist. However, even here, it is conceivable that a polynomial-time algorithm exists which generates at least one curve (as opposed to all curves) that fits ρ fraction of the data. This may be useful in practice.

Acknowledgments

Thanks to Ravi Kannan, Madhu Sudan and Alex Samorodnitsky for useful conversations in various stages of this work and to Dasgupta et al. for sending us their manuscript. We thank Bjorn Poonen for his help with Section 8 and Tamas Erdelyi for useful pointers. Many thanks to annonymous referees for their excellent comments.

References

[1] S. Ar, R. Lipton, R. Rubinfeld, and M. Sudan. Reconstructing algebraic functions from mixed data. SIAM Journal on Computing, 28(2): 487–510, April 1999. *Prelim version in IEEE FOCS 1992.*

[2] J. Canny. A computational approach to edge detection. *IEEE Trans. Pattern Analysis and Machine Intelligence*, Vol PAMI-8, No 6, Nov 1986, 255–274.

[3] S. Dasgupta, E. Pavlov, and Y. Singer. An efficient PAC algorithm for reconstructing a mixture of lines. *13th Int'l Conf. Algorithmic Learning Theory*, Nov 2002.

[4] R. A. DeVore and G. G. Lorentz. Constructive Approximation. Springer Verlag, Berlin Heidelberg 1993.

[5] R.O. Duda and P. E. Hart. The use of the Hough transformation to detect lines and curves in pictures. *Comm. ACM*, **15**:11-15, 1972.

[6] T. Erdelyi. Markov-Bernstein type inequalities for polynomials under Erdos-type constraints. *Paul Erdos and his Mathematics* , July 4-11, 1999, Budapest, Hungary , to appear.

[7] P. Erdös. On extremal properties of the derivatives of polynomials. *Ann. of Math.*, **2**:310–313, 1940.

[8] M. Grotschel, L. Lovász, and A. Schrijver. *Geometric algorithms and combinatorial optimization.* Springer-Verlag, 1993.

[9] A. Hueckel. An operator which locates edges in digital pictures. Journal of the ACM, **20**:113-125, 1971.

[10] J. Illingworth and J. Kittler. A survey of the Hough transform, *Computer Vision, Graphics, and Image Processing*, **44**:87–116, 1988.

[11] V. Leavers. Which Hough transform? A survey of Hough transform methods. *CVGIP - Image Understanding* **58**(2):250-264, 1993.

[12] G. G. Lorentz. Approximation of functions, 2nd Ed., Chelsea, New York, NY, 1986.

[13] D. Marr. *Vision*, W.H. Freeman, 1982.

[14] D. Marr and E. Hildreth. Theory of edge detection. *Proc. Royal Soc. of London*, **207**:187–217, 1980.

[15] J.S. Mill. An examination of Sir William Hamilton's philosophy, Longman Green, London 1889. Discusses in *Selective History of Theories of Visual Perception.*

[16] T. J. Rivlin. Introduction to the Approximation of Functions. Dover Publications, 1981.

[17] M. Sudan. Decoding of Reed Solomon codes beyond the error-correction bound. *Journal of Complexity*, 13(1): 180–193, March 1997. (*Prelim. version in IEEE FOCS 1996.*)

[18] L.G. Valiant. Deductive learning. *Phil. Trans. Royal Soc. Lond.*, A 312:441–446, 1984.

[19] L. Welch and E. R. Berlekamp. Error correction of algebraic block codes, US Patent Number 4,633,470, December 1986.