RECONSTRUCTING ALGEBRAIC FUNCTIONS FROM MIXED DATA

SIGAL AR*, RICHARD J. LIPTON†, RONITT RUBINFELD‡, AND MADHU SUDAN§

Abstract. We consider a variant of the traditional task of explicitly reconstructing algebraic functions from black box representations. In the traditional setting for such problems, one is given access to an unknown function f that is represented by a black box, or an oracle, which can be queried for the value of f at any input. Given a guarantee that this unknown function f is some nice algebraic function, say a polynomial in its input of degree bound d, the goal of the reconstruction problem is to explicitly determine the coefficients of the unknown polynomial. All work on polynomial interpolation, especially sparse ones, are or may be presented in such a setting. The work of Kaltofen and Trager [25], for instance, highlights the utility of this setting, by performing numerous manipulations on polynomials presented as black boxes.

The variant considered in this paper differs from the traditional setting in that our black boxes represent several algebraic functions – $f_1, \ldots, f_k$, where at each input x, the box arbitrarily chooses a subset of $f_1(x), \ldots, f_k(x)$ to output and we do not know which subset it outputs. We show how to reconstruct the functions $f_1, \ldots, f_k$ from the black box, provided the black box outputs according to these functions "often". This allows us to *group* the sample points into sets, such that for each set, all outputs to points in the set are from the same algebraic function. Our methods are robust in the presence of a small fraction of arbitrary errors in the black box.

Our model and techniques can be applied in the areas of computer vision, machine learning, curve fitting and polynomial approximation, self-correcting programs and bivariate polynomial factorization.

Key Words

Bezout's Theorem, Error Correcting Codes, Multivariate Polynomials, PAC Learning, Noisy Interpolation, Polynomial Factoring, Polynomial Interpolation.

1. Introduction.

Suppose you are given a large set of points in the plane and you are told that an overwhelming majority of these points lie on one of k different algebraic curves of some specified degree bound D (but you are not told anything else about the curves). Given the parameters k and D, your task is to determine or "reconstruct" these algebraic curves, or alternatively, to *group* the points into sets, each of which is on the same degree D curve. Related versions of this problem may also be of interest, such as extensions to higher dimensions, and a setting where instead of the points being given in advance, one is allowed to make queries of the form "what is the value of one of the curves at point x?" (the answer to such a query will not specify which of the k curves was used to compute the value).

Solutions to this fundamental problem have applications to:

*Princeton University, Supported by Dept. of Navy Grant #N00014-85-C-0456, by NSF PYI grant CCR-9057486 and a grant from MITL.

†Princeton University. Part of this research was done while at Matsushita Information Technology Labs.

‡Cornell University. Part of this research was done while at Hebrew University and Princeton University. This research is supported by DIMACS, NSF-STC88-09648, ONR Young Investigator Award N00014-93-1-0590 and grant No. 92-00226 from the United States - Israel Binational Science Foundation (BSF).

§ MIT Laboratory for Computer Science, Cambridge, MA 02139. Email: madhu@lcs.mit.edu. Part of this work was done when this author was at U.C. Berkeley and IBM's Thomas J. Watson Research Center, supported in part by NSF Grant No. CCR 88-96202.

- the grouping problem in computer vision
- computational learning theory
- curve fitting over discrete domains
- simple algorithms for polynomial factorization
- self-correcting programs.

Computer Vision. Consider a computer vision system for a robot that picks parts out of a bin. The input to the system contains an intensity map of the scene. The robot can distinguish between the parts by extracting edges from the image. Current edge detection algorithms use discretised differential operators to extract edges (e.g. [30][10]). These algorithms produce output consisting of a bit map, where for every image point (x, y), the bit value of the point, $e(x, y)$, is set to 1 if and only if this point lies on an edge. For many vision applications it is then desired to connect between neighboring points to achieve a more compact representation of the edge map. This problem, known as "the grouping problem", is complicated by the fact that the parts are cluttered, they may be nonconvex and they may contain holes. No polynomial time algorithm has been found for this problem.

Under the assumption that the edges of the parts are given by piecewise algebraic curves, and that the edge detection process produces results which are free of precision error, our algorithm transforms edge maps into piecewise polynomial curves in polynomial time. The second assumption is unrealistic in real computer vision applications. However, we feel that it suggests an interesting approach which should be studied further.

Computational Learning Theory. Our mechanism can be used to extend some well-known results on learning boolean functions to the setting of learning real-valued functions. Here is a specific instance of such a situation: In the study of economics, the price-demand curve is often considered to be well-described by an algebraic function (e.g. $f(x) = c/x$ or $f(x) = -a \cdot x + b$). However, it is also the case that this curve may change [23]. In particular, there may be several unknown price-demand curves which apply in various situations – one may correspond to the behavior found when the country is at war, a second may apply after a stock market crash, and a third behavior may be found after a change in the tax structure. Some of the factors that determine which curve applies may be obvious, but others may occur because of more subtle reasons. The task of learning the price-demand relationship may be decomposed into the two subtasks of first determining the unknown curves, and then learning what determines the move from one curve to another. Our algorithm gives a solution for the first task.

We consider the Valiant model of PAC learning [36], in which a concept is learnable if there is an efficient algorithm that is able to find a good approximation to the concept on the basis of sample data. In general, our results imply that any function on input x and boolean attributes $(y_1, \ldots, y_m)$ which uses $(y_1, \ldots, y_m)$ to select f_i from a set of polynomial functions $f_1, \ldots, f_k$ and then computes and outputs $f_i(x)$ can be learned, as long as the selector function can be learned.

For example, a *polynomial-valued decision list* given by a list of terms (conjuncts of literals), $(D_1, \ldots, D_k)$ over boolean variables $y_1, \ldots, y_n$, and a list of univariate polynomials, $(f_1, \ldots, f_{k+1})$ in a real variable x, represents a real valued function f as follows:

$$f(x, y_1, \ldots, y_n) = f_i(x)$$

where i is the least index such that $D_i(y_1, \ldots, y_n)$ is true.

If the terms are restricted to being conjunctions of at most c literals, we call it a *polynomial-valued c-decision list.* This is an extension of the *boolean decision list* model defined by Rivest in [32], where the polynomials f_i are restricted to being the constants 0 or 1.

In [32], Rivest shows that the class of boolean c-decision lists is learnable in polynomial time. Using our techniques, in combination with Rivest's algorithm, we can extend this result to show that the class of polynomial-valued c-decision lists can be learned in polynomial time. The only technical point that needs to be made is as follows: Rivest gives an algorithm for producing a decision list that is consistent with the random examples and then argues using an Occam argument (see Blumer et. al. [8]) that any hypothesis that is consistent with the labels of the random examples is a good hypothesis (i.e. computes a function that is usually equal to the target function). Our techniques in combination with Rivest's algorithm yield a consistent hypothesis, but since our hypothesis is not a boolean function, we must use the work of Haussler [22] to see that a consistent hypothesis is a good hypothesis.

Independent of our work, Blum and Chalasani [6] also consider a model of learning from examples where the examples may be classified according to one of several different concepts. In their model an adversary controls the decision of which concept would be used to classify the next example. Under this model they study the task of learning boolean-valued concepts such as k-term DNF's and probabilistic decision lists.

Curve Fitting Problems over Discrete Domains. A typical curve fitting problem takes the following form: Given a set of points $\{(x_1, y_1), \ldots, (x_m, y_m)\}$ on the plane, give a simple curve that "fits" the given points. Depending on the exact specification of the "fit" the problem takes on different flavors: for instance, if the curve is to pass close to all the points, then this becomes a *uniform approximation* problem (see text by Rivlin [33]), while if the curve is supposed to pass through *most* of the points, then it resembles problems from coding theory. Here, we consider a problem that unifies the above two instances over *discrete domains.* For example, given a set of m points, with integer coordinates, we show in Subsection 4.1 how to find a polynomial with integer coefficients that is Δ-close to all but an ϵ fraction of the points (if such a polynomial exists), where ϵ need only be less than 1/2 (provided is m is larger than $\frac{(4\Delta+1)d}{1-2\epsilon}$).

Reducing Bivariate Factoring to Univariate Factoring. In [4] Berlekamp gave a randomized polynomial time algorithm for factoring univariate polynomials over finite fields. Kaltofen [24] and Grigoriev and Chistov [18] show that the problem of bivariate factoring can also be solved in polynomial time by a reduction to univariate factoring, using somewhat deep methods from algebra. Our techniques in Subsection 4.2 give a simple method to reduce the problem of factoring bivariate polynomials to that of factoring univariate polynomials over finite fields in the special case when the bivariate polynomial splits into factors which are monic and of constant degree in one of the variables. Though the results are not new, nor as strong as existing results, the methods are much simpler than those used to obtain the previously known results.

Self-Correcting Programs . One motivation for this work comes from the area of self-correcting programs introduced independently in [7][28]. For many functions, one can take programs that are known to be correct on *most* inputs and apply a simple transformation to produce a program that is correct with high probability on *each* input. But, how bad can a program be, and still allow for such a transformation? There is previous work addressing this question when the functions in question are polynomials (see for example [28],[13],[14]). When the program is not known to be correct on most inputs, the definition of self-correction needs to be modified, since the program can toggle between two seemingly correct functions. Our methods give self-correctors that work when the error of the program is such that it answers according to one of a small number of other algebraic functions. An algebraic decision tree may contain a small number of branches, in which all subtrees are intended to compute the same function but are computed separately for purposes of efficiency. The algebraic decision tree program might err in some of the subtrees and compute the wrong algebraic function. Our self-correctors output a small number of candidates for the correct function.

One particular situation where this is useful is in the computation of the permanent of a matrix, over a finite field. Results of Cai and Hemachandra ([9]), when used in combination with our results, imply that if there is an efficient program which computes the permanent correctly on a non-negligible fraction of the input and computes one of a small number of other algebraic functions on the rest of the inputs, then the permanent can be computed efficiently everywhere.

1.1. The k-Algebraic Black Box Model. We consider the following black-box reconstruction problem, which is general enough to model all of the aforementioned problems. We think of the black-box as “containing” k functions, $f_1, \ldots, f_k$, where f_i is an “algebraically well-behaved” function. For instance, each f_i could be a polynomial of degree at most d, and on every input x the black box outputs $f_i(x)$ for some $i \in [k]$. (Here and throughout this paper, the notation $[k]$ stands for the set of integers $\{1, \ldots, k\}$.) Relating to the problem discussed in the first paragraph of the introduction, this corresponds to the case where for every value of an x-coordinate there is at least one point that has that value. We now present this definition formally, starting with the standard black box model ($k = 1$).

Definition 1.1. *A* black box B *is an oracle representing a function from a finite domain* $\mathcal{D}$ *to a range* $\mathcal{R}$.

There are two kinds of domains that will be of interest to us. One is a finite subset H of a (potentially infinite) field F. The second is an n-dimensional vector space over a finite field F. In both cases the range will be the field F.

Previous research on black box reconstruction focused on the following: Assuming that B is one of a special class of functions (for example, that B is a degree d polynomial), determine an explicit representation of B. In our model, there may be more than one output that is valid for each input. More specifically:

Definition 1.2. *A black box B mapping a finite subset H of a field F to F is a* (k, d)-polynomial black box *if there exist polynomials $f_1, \ldots, f_k : F \to F$ of degree at most d such that for every input $x \in H$, there exists $i \in \{1, \ldots, k\}$, such that $B(x) = f_i(x)$. In such a case, we say that the functions $f_1, \ldots, f_k$ describe the black box B.*

Our first *black box reconstruction problem* is:

> *Given a (k,d)-polynomial black box B, find a set of functions $f_1,\dots,f_k$ that describe B.*

The definition of a (k,d)-polynomial black box can be generalized to situations involving noise as follows:

Definition 1.3. *For $\epsilon \in [0,1]$ and for a finite subset H of a field F, a black box $B : H \to F$ is an* ϵ-noisy (k,d)-polynomial black box *if there exist polynomials $f_1,\dots,f_k : F \to F$ of degree at most d and a set $S \subset H$, with $|S| \geq (1-\epsilon)|H|$, such that for every input $x \in S$, there exists $i \in \{1,\dots,k\}$ such that $B(x) = f_i(x)$. In such a case, the functions $f_1,\dots,f_k$ are said to describe B.*

The notion of the reconstruction problem generalizes to the noisy case in the obvious way. We now attempt to generalize the problem to allow the black box to compute other algebraic functions, such as $B(x) = \sqrt{x}$ etc. This part is somewhat more technical, so we introduce some new terminology:

Definition 1.4. *For positive integers d_x, d_y and indeterminates x, y, the* $\{(d_x,x),(d_y,y)\}$-weighted degree *of a monomial $x^i y^j$ is $id_x + jd_y$. The $\{(d_x,x),(d_y,y)\}$-weighted degree of a polynomial $Q(x,y)$ is the maximum over all monomials in Q (i.e., the monomials with non-zero coefficients in Q) of their $\{(d_x,x),(d_y,y)\}$-weighted degree.*

We now introduce the notion of an algebraic box and show how it relates to the earlier notion of a polynomial black box.

Definition 1.5. *For a finite subset H of a field F, A black box $B : H \to F$ is a* (k,d)-algebraic black box *if there exists a polynomial $Q(x,y)$ of $\{(1,x),(d,y)\}$-weighted degree at most kd, such that for every input $x \in H$, the output y of the black box satisfies $Q(x,y) = 0$. In such a case, we say that the polynomial Q describes B.*

For example, if $B(x) = \sqrt{x}$, the polynomial $Q(x,y) = (y^2 - x)$ satisfies the requirement of the definition and describes B. The $\{(1,x),(d,y)\}$-weighted degree of Q is $2d$.

Proposition 1.6. *If B is a (k,d)-polynomial black box then B is also a (k,d)-algebraic black box.*

Proof. Let B be a (k,d)-polynomial black box and let $f_1,\dots,f_k$ describe it. Then the polynomial $Q(x,y) \stackrel{\text{def}}{=} \prod_{i=1}^{k}(y - f_i(x))$ describes B and has $\{(1,x),(d,y)\}$-weighted degree at most kd. □

The (k,d)-algebraic black box reconstruction problem is:

> Given a (k,d)-algebraic box B, find the polynomial Q of $\{(1,x),(d,y)\}$-weighted degree at most kd which describes it.

The definition and proposition can be extended easily to the ϵ-noisy case.

All the above definitions generalize to a case where the input is an n-dimensional vector over F and the black box is computing n-variate functions. In particular, we have:

Definition 1.7. *For a finite field F, a n-variate black box $B : F^n \to F$ is a* (k,d)-polynomial black box *if there exist polynomials $f_1,\dots,f_k : F^n \to F$ of total degree at most d such that for every input $(x_1,\dots,x_n) \in F^n$ there exists $i \in \{1,\dots,k\}$ such that $B(x_1,\dots,x_n) = f_i(x_1,\dots,x_n)$.*

DEFINITION 1.8. *For a finite field F, An n-variate black box $B : F^n \to F$ is a (k, d)-*algebraic black box *if there exists a polynomial $Q(x_1, \ldots, x_n, y)$ of $\{(1, x_1), \ldots, (1, x_n), (d, y)\}$-weighted degree at most kd such that for every input $(x_1, \ldots, x_n) \in F^n$, the output y of the black box satisfies $Q(x_1, \ldots, x_n, y) = 0$.*

Again the reconstruction problems are defined correspondingly. In this paper we attempt to solve all such reconstruction problems. Notice that this problem is not well-defined if there exist multiple solutions, say Q and $\tilde{Q}$, such that both Q and $\tilde{Q}$ describe the black box. Much of the work is done in establishing conditions under which any $\tilde{Q}$ that describes the black box gives a meaningful answer.

1.2. Previous Work and Our Results. The setting where the black box represents a *single* polynomial or rational function, without noise, is the classic interpolation problem and is well studied. Efficient algorithms for sparse multivariate polynomial interpolation are given by Zippel [40, 41], Grigoriev, Karpinski and Singer [21] and Borodin and Tiwari [3], and for sparse rational functions by Grigoriev, Karpinski and Singer [20].

The case where the black box represents a single function with some noise has also been studied previously. Welch and Berlekamp [39, 5] (see also [14]) show how to reconstruct a univariate polynomial from a $(\frac{1}{2} - \delta)$-noisy $(1, d)$-polynomial black box and Coppersmith [11], Gemmell, Lipton, Rubinfeld, Sudan and Wigderson [13] and Gemmell and Sudan [14] show how to do the same for multivariate polynomials. All the above mentioned results require, however, that the field size be at least polynomially large in $\frac{d}{\delta}$. The conditions are required to ensure that there is a *unique* degree d polynomial describing the black box on $1/2 + \delta$ fraction of the inputs. Reconstructing functions from a black box representing more than one function, or when the function it represents is not guaranteed to be unique, seems to be a relatively unexplored subject. The works of Goldreich and Levin [15] and Kushilevitz and Mansour [26] are the only exceptions we know of. Both papers study the reconstruction of n-variate linear functions (i.e., homogenous polynomials of degree 1) from a $1/2 - \delta$-noisy black box over GF(2). In this case, there can be up to $O(\frac{1}{\delta}^2)$ polynomials representing the black box and they reconstruct all such polynomials.

The main result in this paper is an algorithm for reconstructing algebraic functions describing noisy algebraic black boxes, which works when the black box satisfies certain conditions. To see why the result needs to have some conditions on the black box, consider the following example: Suppose the black box is described by the polynomial $(x^2 + y^2 - 1)(x + y - 1)$. But suppose for every x the black box always outputs a y from the unit circle (and never according to the line $x + y - 1 = 0$. Then clearly the reconstruction algorithm has no information to reconstruct the line $x + y - 1$. The condition imposed on the black box essentially addresses this issue. We describe the result for univariate ϵ-noisy (k, d)-polynomial black boxes. We present a randomized algorithm which takes as input a parameter $p > \epsilon$ and with high probability outputs a list of all polynomials f_i which describe the black box on more than p fraction of the input, provided $(p - \epsilon)|H| > kd$. (This condition amounts to saying that the black box must output according to f_i sufficiently often.) The running time of the algorithm is a polynomial in k, d and $\frac{1}{(p-\epsilon)}$. This result is presented along with generalizations to univariate noisy algebraic black boxes in Section 2.

To reconstruct a univariate polynomial, we sample the black box on a small set of inputs and construct a bivariate polynomial $\tilde{Q}(x, y)$ which is zero at **all** the sample points. Then we use bivariate polynomial factorization to find a factor of the form $(y - f(x))$. If it exists, $f(x)$ then becomes our candidate for output. We show that if the number of points (x, y) such that $y = f(x)$ is large in the sample we chose, then $y - f(x)$ has to be a factor of any $\tilde{Q}$ which all the sample points satisfy.

Our results do not generalize immediately to multivariate polynomials. Among other factors, one problem is that an n-variate polynomial of degree d has $\binom{n+d}{d}$ coefficients, which is exponential in n (or d). This seems to make the problem inherently hard to solve in time polynomial in n and d. However we bypass this, once again using the idea of black boxes. Instead of trying to reconstruct the multivariate polynomial explicitly (i.e., by determining all its coefficients), we allow the reconstruction algorithm to reconstruct the polynomial implicitly, i.e., by constructing a black box which computes the multivariate polynomial. If the polynomial turns out to be sparse then we can now use any sparse interpolation algorithm from [3, 20, 21, 40] to reconstruct an explicit representation of the polynomials in time polynomial in n, d and the number of non-zero coefficients. On the other hand, by using the techniques of [25] we can also continue to manipulate the black boxes as they are for whatever purposes[1].

We now describe our result for reconstructing multivariate polynomials. We present a randomized algorithm which takes as input a parameter p and with high probability reconstructs probabilistic black boxes for all the polynomials $f_1, \ldots, f_k$ describing a noisy (k, d)-black box over a finite field F, provided the noisy (k, d)-black black box satisfies the following conditions: (1) Every polynomial is represented on at least a p-fraction of the inputs (i.e. for every i, $\Pr_{\hat{x} \in F^n}[B(\hat{x}) = f_i(\hat{x})] \geq p$). (2) The finite field F over which the black box works is sufficiently large ($|F|$ should be polynomially large in $k, d, \frac{1}{(p-\epsilon)}$). The running time of the algorithm is polynomial in k, d and $\frac{1}{(p-\epsilon)}$. The main technique employed here is a randomized reduction from the multivariate to the univariate case. We note that the solution obtained here for the multivariate case differs in several aspects from the solution for the univariate case. First, this algorithm does not extend to the case of finite subsets of infinite fields. Second, it needs to make sure that all the polynomials are well-respresented in the black box. The latter aspect is a significant weakness and getting around this is an open question.

Subsequent work. One of the main questions left open by this paper is the problem of reconstructing all degree d polynomials that agree with an arbitrary black box on ϵ fraction of the inputs. Some recent work has addressed this question. Goldreich, Rubinfeld and Sudan [16] give an algorithm to (explicitly) reconstruct all n-variate degree d polynomials agreeing with a black box over F on ϵ fraction of the inputs, provided $\epsilon \geq 2\sqrt{d/|F|}$. Their algorithm runs in time $O((n, \frac{1}{\epsilon})^{\mathrm{poly}(d)})$, which is exponential in d. Their algorithm generalizes the earlier mentioned solution of Goldreich and Levin [15]. For the case of univariate polynomials, Sudan [35], has given a polynomial time algorithm which can find all degree d polynomials agreeing with a black box on ϵ fraction of the domain, provided $\epsilon \geq 2\sqrt{d/|F|}$. The main contribution in [35] is a simple observation which shows that m input/output pairs from any black box can be thought of as the output of a $(1, O(\sqrt{m}))$-algebraic black box. Using this observation, Lemma 2.10 of this paper is applied to reconstruct all polynomials of low

[1]The idea of manipulating multivariate polynomials and rational functions represented by black boxes was proposed by Kaltofen and Trager in [25]. They show that it is possible to factor and compute g.c.d.'s for polynomials given by such a representation and to separate the numerator from the denominator of rational functions given by such a representation.

degree which describe the black box on ϵ-fraction of the inputs. Finding a similar solution for the multivariate cases remains open (some cases are addressed by [35], but the problem is not completely resolved). A second question that is left open is the task of solving the (k, d)-polynomial black box problem over the reals, where the points are not provided to infinite precision. This is the true problem underlying the application to computer vision. While the ideas in this paper do not immediately apply to this question, some variants (in particular, the variant employed in [35]) seem promising and deserve to be investigated further.

In other related work, Rubinfeld and Zippel [34] have employed the black box reconstruction problem and build on the techniques presented in this paper to present a modular approach to the polynomial factorization problem. While the application presented in this paper (in Section 4.2) is to a restricted subclass of the bivariate factorization problem, the work of [34] finds an application to the general multivariate factorization problem.

1.3. Organization . The rest of this paper is organized as follows. In Section 2, we describe our results for univariate polynomials, rational functions and other algebraic functions. In Section 3, we consider extensions of the reconstruction problem to the case of multivariate polynomials. Finally, in Section 4, we describe several applications of our work.

2. Univariate Black Boxes. In this section we consider the univariate reconstruction problem for (noisy) (k, d)-polynomial and algebraic black boxes. We describe our general format of our results with the example of a (k, d)-polynomial black box described by $f_1, \ldots, f_k$. We present a solution in the form of an algorithm which takes m input/output pairs $\{(x_1, y_1), \ldots, (x_m, y_m)\}$ of the black box and attempts to reconstruct the polynomials $f_1, \ldots, f_k$ from this set of input/output pairs. In order to reconstruct a small set of polynomials which includes a specific polynomial f_i, the algorithm (obviously) needs to find sufficiently many points (x_j, y_j) such that $y_j = f_i(x_j)$ ($d + 1$ such points are needed). We present complementary bounds, showing that if the number of points on f_i is sufficiently large, then the output is guaranteed to include f_i. We then show how some simple sampling of the black box (either by exhaustively sampling all points from the domain H or by picking a random sample of x_j's chosen independently and uniformly at random from H) yields a collection of input/output pairs which satisfies the required condition, provided H is large enough and the fraction of inputs on which B's output is described by f_i is large enough.

2.1. An Intermediate Model. As a first step towards solving the algebraic reconstruction problem, we consider the case where the black box outputs *all* of $f_1(x), \ldots, f_k(x)$ on any input x. We refer to this as a (k, d)-*total polynomial black box*. These are output in arbitrary order, which is not necessarily the same for each x. We further assume that there are no errors in the output. Thus the reconstruction problem we wish to solve may be stated formally as:

Given: *Positive integers k and d, a field F and a black box $B = (B_1, \ldots, B_k)$ where $B_i : F \to F$ with the property that there exist polynomials $f_1, \ldots f_k$ of degree at most d over F, such that for every $x \in F$, the multisets $\{B_1(x), \ldots, B_k(x)\}$ and $\{f_1(x), \ldots, f_k(x)\}$ are identical.*

Problem: *Find $f_1, \ldots, f_k$.*

We reduce the problem of extracting the polynomials to that of bivariate polynomial factorization. The main idea underlying this reduction is the following: On input x if the (k,d)-total polynomial black box outputs $\{y_1, \ldots, y_k\}$, we know that $\forall j \in [k]$, $\exists i \in [k]$ such that $y_j = f_i(x)$. Therefore, each input/output pair $(x; y_1, \ldots, y_k)$ of the black box satisfies the relation:

$$\sum_j \prod_i \; (y_j - f_i(x)) = 0.$$

Our aim will be to construct a related polynomial which will enable us to recover the f_i's.

Consider the functions $\sigma_j : F \mapsto F$, $j \in [k]$ defined as

$$\sigma_j(x) \stackrel{\text{def}}{=} \sum_{S \subset [k], |S| = j} \; \prod_{i \in S} f_i(x).$$

(these are the *primitive symmetric* functions of $f_1, \ldots, f_k$).

Observe that $\sigma_j(x)$ can be evaluated at any input x using the given (k,d)-total polynomial black box, using the identity

$$\sigma_j(x) = \sum_{S \subset [k], |S| = j} \; \prod_{i \in S} B_i(x).$$

Furthermore this computation can be performed in time in $O(k \log k \log \log k)$ using a fast Fourier transform (see survey article by von zur Gathen [12], pages 320–321). Observe further that σ_j is a polynomial of degree at most jd. Hence evaluating it at $jd + 1$ points suffices to find all the coefficients of this polynomial (if the black box outputs every $f_i(x)$ for every x).

Now consider the following bivariate polynomial, in x and a new indeterminate y:

$$Q(x,y) \stackrel{\text{def}}{=} y^k - \sigma_1(x) y^{k-1} + \cdots + (-1)^k \sigma_k(x).$$

From the explicit representation of the σ_i's, we can also compute an explicit representation of Q. But now notice that Q can equivalently be written as:

$$Q(x,y) = \prod_{i=1}^{k} (y - f_i(x)).$$

(The equivalence follows from the definition of the σ_j's.) Therefore, to recover the f_i's, all we have to do is find the factors of the bivariate polynomial Q. Bivariate factorization can be done efficiently over the rationals [18, 24, 29] and can be done efficiently (probabilistically) over finite fields [17, 24].

We now summarize our algorithm. The input to the algorithm is $kd + 1$ distinct elements $\{x_1, \ldots, x_{kd+1}\}$ from the the field F, and a set $\{y_{1,j}, \ldots, y_{k,j}\}$ for every $j \in [kd+1]$ representing the output of the black box B on input x_j.

1. Evaluate $\sigma_j(x_i)$ for every $j \in [k]$ and every $i \in [kd+1]$.
2. Interpolate for the coefficients of $\sigma_j(x)$ and let $\sigma_{j,l}$ be the coefficient of x^l in σ_j.

3. Let $Q(x,y)$ be the polynomial $\sum_{j=0}^{k}\sum_{l=0}^{jd}(-1)^j\sigma_{j,l}x^l y^{k-j}$.
4. Factor Q into its irreducible factors. This will yield $Q(x,y)=\prod_{i=1}^{k}(y-g_i(x))$.
5. Output the polynomials $g_1,\dots,g_k$.

The arguments leading to this algorithm prove its correctness and we have the following lemma.

LEMMA 2.1. *Let $\{(x_j;(y_{j,1},\dots,y_{j,k})\}_{j=1}^{kd+1}$ be the input/output pairs of a (k,d) total polynomial black box B over a field F on $kd+1$ distinct inputs. Then there exists an randomized algorithm whose running time is polynomial in k,d which explicitly reconstructs the set of polynomials $\{f_1,\dots,f_k\}$ which describe B.*

Since the only condition on the x_j's is that they be distinct, it is easy to get a total polynomial reconstruction algorithm from the above lemma and thus we get the following theorem.

THEOREM 2.2. *Let $f_1,\dots,f_k$ be degree d polynomials over $\mathcal{Q}$ (the rationals) or a finite field F of cardinality at least $kd+1$. Given a black box B which on input x outputs the multiset $\{f_1(x),\dots,f_k(x)\}$ (in arbitrary order), there exists an algorithm which queries the black box on $kd+1$ distinct inputs and reconstructs the polynomials that describe the black box. The algorithm is deterministic when the polynomials are over $\mathcal{Q}$ and probabilistic when the polynomials are over some finite field.*

2.2. (k,d)-polynomial black boxes. We now build on the methods of the previous section to reconstruct information from a (k,d)-polynomial black box which outputs the value of *one* of k univariate polynomials $f_1,\dots,f_k$, on every input. Our method extends immediately to two more general cases:

1. (k,d)-algebraic black boxes.
2. Noisy (polynomial and algebraic) black boxes.

The generalizations are dealt with in the next section.

The problem we wish to solve is formally stated as:

Given: *Positive integers k and d, a field F, a finite set $H\subseteq F$ and a black box $B:H\to F$ with the property that there exist polynomials $f_1,\dots f_k$ of degree at most d over F, such that for every $x\in H$, $B(x)\in\{f_1(x),\dots,f_k(x)\}$.*

Problem: *Find $f_1,\dots,f_k$.*

Our solution for this problem is based on the solution of the previous subsection. The critical observation is that the polynomial Q produced by the algorithm of the previous section always satisfied the property $Q(x,y)=0$ for any input x to the black box and where y is any element of the output set of the black box on input x. We will try to construct a polynomial Q in two variables as in the previous section, satisfying the property that if $y=B(x)$ is the output of the black box on input x, then $Q(x,y)=0$. However, we will not be able to construct the polynomials $\sigma_i(x)$ as in the previous case. Hence, we will abandon that part of the algorithm and directly try to find any polynomial $\tilde{Q}$ such that $\tilde{Q}(x,y)=0$ on all the sampled points. We will then use the factors of this polynomial to determine the f_i's as in the previous section. Thus our algorithm is summarized as follows:

The input to the algorithm is m distinct pairs of elements $\{(x_1,y_1),\dots,(x_m,y_m)\}$.

1. Interpolate to find a set of coefficients $\tilde{q}_{lj}$ of the polynomial

$$\tilde{Q}(x,y) = \sum_{l=0}^{k} \sum_{j=0}^{dl} \tilde{q}_{lj} y^{k-l} x^j$$

   that satisfies $\tilde{Q} \not\equiv 0$ and $\tilde{Q}(x_i, y_i) = 0$ for $i \in [m]$.
2. Factor the polynomial $\tilde{Q}$ and if it has any factors of the form $(y - g(x))$, output g as a candidate polynomial.

Notes: The important step above is Step 1 which involves finding a non-trivial solution to a homogenous linear system. First we need to make sure this system has at least *one* solution. This is easy since $Q(x,y) = \prod_i (y - f_i(x))$ is such a solution. However the solution in such a step need not necessarily be unique and we will simply find *any* solution to this system and show that it suffices, under certain conditions, for Step 2. In what follows, we shall examine the conditions under which the output will include a certain polynomial f_i.

LEMMA 2.3. *For a set $\{(x_j, y_j) | j \in [m]\}$ of m distinct pairs from $F \times F$, if $\tilde{Q}$ is a bivariate polynomial of $\{(1,x),(d,y)\}$-weighted degree kd satisfying*

$$\forall\ j \in [m],\ \tilde{Q}(x_j, y_j) = 0$$

and f is a univariate polynomial of degree d satisfying

$$|\{j | f(x_j) = y_j\}| > kd,$$

then the polynomial $(y - f(x))$ divides the polynomial $\tilde{Q}(x,y)$.

Proof. Let $S \overset{\text{def}}{=} \{j | f(x_j) = y_j\}$. Notice that for distinct $j_1, j_2 \in S$, $x_{j_1} \neq x_{j_2}$, or else the pairs $(x_{j_1}, f(x_{j_1}))$ and $(x_{j_2}, f(x_{j_2}))$ are not distinct.

Consider the univariate polynomial $\tilde{Q}_f(x) \equiv \tilde{Q}(x, f(x))$. For all indices $j \in S$ we have that $\tilde{Q}(x_j) = 0$. Furthermore $\tilde{Q}_f(x)$ is a polynomial of degree at most kd in x. Hence if $\tilde{Q}_f$ is zero at $|S| > kd$ places, then it must be identically zero, implying that $(y - f(x)) | \tilde{Q}(x,y)$. □

The lemma above guarantees that under certain circumstances, the factors of $\tilde{Q}(x,y)$ do give useful information about the f_i's. The effect is summarized in the following lemma.

LEMMA 2.4. *Let $\{(x_1,y_1), \ldots, (x_m,y_m)\}$ be m distinct pairs of elements which are the input/output pairs of a (k,d)-polynomial black box B described by polynomials $f_1, \ldots, f_k$. If there exists an $i \in [k]$ such that $|\{j | y_j = f_i(x_j)\}| > kd$, then a set of at most k polynomials $\{g_1, \ldots g_k\}$ that includes f_i can be found in time polynomial in m, k and d.*

Remark: Notice that Lemma 2.4 is a strict strengthening of Lemma 2.1.

To finish the analysis of the algorithm we need to determine how to sample the black box B so as to get enough points according to f_i. Let $p_i \overset{\text{def}}{=} \Pr_{x \in H}[B(x) = f_i(x)]$ and $\delta > 0$ be the confidence parameter. Let $M = \frac{4}{p_i}(kd + \ln \frac{2}{\delta})$. The strategy for picking the points $\{(x_1,y_1), \ldots, (x_m,y_m)\}$ depends on $|H|$. If $|H| \geq \frac{2}{\delta}\binom{M}{2}$, then we let $m = M$ and pick m elements $x_1, \ldots, x_m$ independently and uniformly at random from H. Lemma A.1 in the appendix (shown using a simple combination of Chernoff bounds and the "birthday problem analysis") shows that the sampled points are all distinct and satisfy $|\{j : B(x_j) = f_i(x_j)\}| > kd$ with probability at least $1 - \delta$. Thus in this case we will use $\{(x_1, B(x_1), \ldots, (x_m, B(x_m)\}$ as the input to the algorithm described above. If on the other hand $|H|$ is not large enough, then we will simply sample every point in H (i.e., the input set will be $\{(x, B(x))|x \in H\}$) implying in particular that $m = |H|$, and in this case the algorithm described above will include f_i as part of its output provided $p_i|H| > kd$. Notice that in both cases the running time of the algorithm is polynomial in M, which is in turn bounded by some polynomial in $k, d, \frac{1}{p_i}, \frac{1}{\delta}$. Furthermore, by choosing $\delta' = \delta/k$ and a threshold parameter p and running the algorithm above with confidence parameter δ', we find that the algorithm above recovers, with confidence $1 - \delta$, every polynomial f_i, such that $p_i \geq p$. The running time is still a polynomial in $k, d, \frac{1}{p}, \frac{1}{\delta}$. This yields the following theorem.

THEOREM 2.5. *Let B be a (k, d)-polynomial black box, mapping a finite domain H to a field F, described by polynomials $f_1, \ldots, f_k$. For $i \in [k]$, let $p_i \overset{\text{def}}{=} \Pr_{x \in H}[B(x) = f_i(x)]$. There exists an algorithm which takes as input a confidence parameter $\delta > 0$ and a threshold $p > 0$, runs in time $\text{poly}(k, d, \frac{1}{p}, \frac{1}{\delta})$ and makes calls to the black box B and with probability at least $1 - \delta$ reconstructs a list of at most k polynomials which includes all polynomials f_i such that $p_i \geq p$, provided $p > \frac{kd}{|H|}$.*

2.3. (k, d)-algebraic black boxes. The algorithm of Section 2.2 extends immediately to the case of algebraic black boxes. Here, by definition, the input/output pair of the black box, (x, y), satisfies an algebraic relation of the form $Q(x, y) = 0$. We can attempt to find a polynomial $\tilde{Q}$ which satisfies $\tilde{Q}(x, y) = 0$ for all the sampled points by interpolation (Step 3 in the algorithm of Section 2.2).

As in the previous section, it will not be possible to guarantee that the output we produce will be exactly Q. For instance, if $Q(x, y) = (x^2+y^2-1)(x+y-1)$, but all the points actually come from the unit circle, then the algorithm has no information to point to the line $x+y-1 = 0$. Thus, as in the previous section, we will only attempt to find those parts of the curve that describe significant portions of output of the black box. More precisely, if $Q(x, y)$ factors into irreducible factors $Q_1(x, y), \ldots . Q_l(x, y)$ and we know that many points satisfy, say, $Q_1(x_j, y_j) = 0$, then we would like Q_1 to be one of the outputs of the algorithm.

The proof that this is indeed the case is slightly more complicated than in the previous subsection. We will use a version of Bezout's theorem ([38], Theorem 3.1). Essentially, Bezout's theorem states that two algebraic curves in the plane cannot intersect in infinitely many points, unless they are identical. The theorem gives an explicit bound on the number of points where two curves of degree d_1 and d_2 may meet. Bezout's bound is slightly weaker than the one we wish to prove for the case of (k, d)-algebraic black boxes, so we prove our lemma from first principles.

Before going on to the next lemma we review a couple of standard definitions from algebra (cf. [38]).

DEFINITION 2.6. *Given univariate polynomials $P(y) = \sum_{i=0}^{d_1} \alpha_i y^i$, and $Q(y) = \sum_{j=0}^{d_2} \beta_j y^j$ over some domain F, let $M(P, Q)$ be the $(d_1 + d_2) \times (d_1 + d_2)$ matrix* *given as follows:*

$$M(P,Q) = \begin{bmatrix} \alpha_0 & \alpha_1 & \cdots & \alpha_{d_1-1} & \alpha_{d_1} & 0 & \cdots & 0 & 0 & 0 & \cdots & 0 \\ 0 & \alpha_0 & \cdots & \alpha_{d_1-2} & \alpha_{d_1-1} & \alpha_{d_1} & \cdots & 0 & 0 & 0 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 0 & 0 & 0 & \cdots & \alpha_0 & \alpha_1 & \alpha_2 & \cdots & \alpha_{d_1} \\ \beta_0 & \beta_1 & \cdots & \beta_{d_1-1} & \beta_{d_1} & \beta_{d_1+1} & \cdots & \beta_{d_2-1} & \beta_{d_2} & 0 & \cdots & 0 \\ 0 & \beta_0 & \cdots & \beta_{d_1-2} & \beta_{d_1-1} & \beta_{d_1} & \cdots & \beta_{d_2-2} & \beta_{d_2-1} & \beta_{d_2} & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \beta_0 & \beta_1 & \beta_2 & \cdots & \beta_{d_2-d_1} & \beta_{d_2-d_1+1} & \beta_{d_2-d_1+2} & \cdots & \beta_{d_2} \end{bmatrix}$$

The resultant *of the polynomials P and Q, denoted Res(P,Q) is the determinant of $M(P,Q)$. For multivariate polynomials $P(x_1,\ldots,x_n;y)$ and $Q(x_1,\ldots,x_n,y)$ their resultant with respect to y is defined similarly by viewing P,Q as polynomials in y with coefficients from the ring of polynomials in $x_1,\ldots,x_n$. We define the matrix $M_y(P,Q)$ as above and its determinant is the resultant Res$_y(P,Q)$.*

LEMMA 2.7. *For a set of points $\{(x_1,y_1),\ldots,(x_m,y_m)\}$, with the x_j's being distinct, if $\tilde{Q}(x,y)$ and $Q_1(x,y)$ are polynomials of $\{(1,x),(d,y)\}$-weighted degree at most kd and k_1d respectively, satisfying the properties: (1) $\forall\ j \in [m]$, $\tilde{Q}(x_j,y_j) = 0$ and (2) $|\{j|Q_1(x_j,y_j) = 0\}| > kk_1d$, then the polynomials $Q_1(x,y)$ and $\tilde{Q}(x,y)$ share a non-constant common factor.*

Proof. Consider the resultant $R_y(x)$ of the polynomials $\tilde{Q}(x,y)$ and $Q_1(x,y)$ with respect to y. Observe that the resultant is a polynomial in x. The following claim bounds the degree of this polynomial.

CLAIM 2.8. *$R_y(x)$ is a polynomial of degree at most k_1kd.*
Proof. The determinant of the matrix $M_y(\tilde{Q},Q_1)$ is given by

$$\sum_{\pi}\prod_{i=1}^{k+k_1} \text{sign}(\pi)(M_y(\tilde{Q},Q_1))_{i\pi(i)},$$

where π ranges over all permutations from $[k+k_1]$ to $[k+k_1]$ and sign(π) denotes the sign of the permutation. We will examine every permutation $\pi : [k+k_1] \to [k+k_1]$ and show that the degree of the term $\prod_{i=1}^{k+k_1}(M_y(\tilde{Q},Q_1))_{i\pi(i)}$ (viewed as a polynomial in x) is at most kk_1d. This will suffice to show that the determinant is a polynomial of degree at most kk_1d.
Let d_{ij} denote the degree of the entry $(M_y(\tilde{Q},Q_1))_{ij}$. Observe that, by the definition of the resultant, $d_{ij} \le (i+k-j)d$ for $i \le k_1$ and $d_{ij} \le (i-j)d$ for $i \ge k_1$. (Here we consider the polynomial 0 as having degree $-\infty$.) Thus the degree of the term $\prod_{i=1}^{k+k_1}(M_y(\tilde{Q},Q_1))_{i\pi(i)}$ is given by

$$\begin{aligned}\sum_{i=1}^{k+k_1} d_{i\pi(i)} &= \sum_{i=1}^{k_1} d_{i\pi(i)} + \sum_{i=k_1+1}^{k+k_1} d_{i\pi(i)} \\ &\le \sum_{i=1}^{k_1}(i+k-\pi(i))d + \sum_{i=k_1+1}^{k+k_1}(i-\pi(i))d\end{aligned}$$

$$= \sum_{i=1}^{k+k_1} id - \sum_{i=1}^{k+k_1} \pi(i)d \ + \ kk_1 d$$
$$= kk_1 d.$$

This concludes the proof. □

It is well-known that the resultant of two polynomials is zero iff the polynomials share a common factor (cf. [38], Chapter 1, Theorem 9.3). We will show that $R_y(x)$ is identically zero and this will suffice to prove the lemma. We show this part in the next claim by showing that $R_y(x)$ has more zeroes than the upper bound on its degree.

Claim 2.9. *For every j such that $\tilde{Q}(x_j, y_j) = Q_1(x_j, y_j) = 0$, $R_y(x_j) = 0$.*
Proof. Fix x_j and consider the polynomials $\tilde{q}(y) = \tilde{Q}(x_j, y)$ and $q_1(y) = Q_1(x_j, y)$. Now $R_y(x_j)$ gives the resultant of the polynomials $\tilde{q}(y)$ and $q_1(y)$. Now we know that $\tilde{q}(y_j) = q_1(y_j) = 0$ implying that $(y - y_j)$ is a common factor of $\tilde{q}$ and q_1. Therefore the resultant of $\tilde{q}$ and q_1 must be zero, implying $R_y(x_j) = 0$. □

□

Since the above holds for any factor Q_i of Q, we have:

Lemma 2.10. *Let B be a (k, d)-algebraic black box described by a bivariate polynomial Q with no repeated non-constant factors. Let $Q_1, \ldots, Q_l$ be the irreducible factors of Q of $\{(1, x), (d, y)\}$-weighted degree $k_1 d, \ldots, k_l d$ respectively. Given m pairs of elements $\{(x_1, y_1), \ldots, (x_m, y_m)\}$ which are the input/output pairs of B on m distinct inputs, if there exists an $i \in [k]$ such that $|\{j | Q_i(x_j, y_j) = 0\}| > k_i kd$, then a set of at most k polynomials $\{\tilde{Q}_1, \ldots, \tilde{Q}_k\}$ that includes Q_i can be found in time polynomial in m, k and d.*

Remark: For a set of pairs $\{(x_1, y_1), \ldots, (x_m, y_m)\}$, with distinct x_j's, Lemma 2.10 is a strengthening of Lemma 2.4. Unfortunately, the proof as shown above does not extend to the case of where the pairs are distinct, but the x_j are not. Due to this limitation, Lemma 2.10 does not even cover the case of Lemma 2.1.

Once again, using a sampling method similar to that used for Theorem 2.5, we get the following theorem.

Theorem 2.11. . *Let B be a (k, d)-algebraic black box described by a polynomial Q with distinct irreducible factors $Q_1, \ldots, Q_l$ such that the $\{(1, x), (d, y)\}$-weighted degree of Q is at most kd and of Q_i is at most $k_i d$. Further, let $p_i = \Pr_{x \in H}[Q_i(x, B(x)) = 0]$. There exists a randomized algorithm which takes as input a confidence parameter $\delta > 0$ and a threshold $p > 0$, runs in time $\mathrm{poly}(k, d, \frac{1}{p}, \frac{1}{\delta})$, makes calls to the black box B and with probability at least $1 - \delta$ reconstructs a list of at most k bivariate polynomials which includes every polynomial Q_i such that $p_i / k_i \geq p$, provided $p|H| > kd$.*

2.4. ϵ-noisy black boxes. Finally we extend the reconstruction algorithms of the previous section to the case when the black boxes are allowed to output noise on an ϵ fraction of the inputs from H. As usual the basic algorithm will be to find a polynomial $\tilde{Q}(x, y)$ which is zero on all the input-output pairs of the black box. However we will have to do something about the noisy points which do not lie on any nice algebraic curve. We adapt an algorithm of Welch and Berlekamp [39, 5] (see also [14]) to handle this situation.

Say we sample the black box B in m points $x_1, \ldots, x_m$ and the black box outputs $y_1, \ldots, y_m$ according to some (unknown) polynomial Q in all but m' locations. Say that these locations are given by $E = \{j | Q(x_j, B(x_j)) \neq 0\}$. We use the fact that there exists a non-zero polynomial $W(x)$ of degree at most m' which is zero when $x = x_j$ for $j \in E$. Indeed $W(x) = \prod_{j \in E}(x - x_j)$ is such a polynomial. Let $Q^*(x, y) = Q(x, y) \cdot W(x)$. Then $Q^*(x_j, y_j)$ is zero for all $j \in [m]$. Thus we can modify the algorithm of the previous section to try to find Q^*. This algorithm is summarized as follows:

The input to the algorithms is m pairs of elements $\{(x_1, y_1), \ldots, (x_m, y_m)\}$ with distinct x_j's.

1. Interpolate to find a set of coefficients q_{lj} of the polynomial
$$\tilde{Q}(x, y) = \sum_{l=0}^{k} \sum_{j=0}^{dl+m'} q_{lj} y^{k-l} x^j$$
that satisfies $\tilde{Q}(x_i, y_i) = 0$ for $i \in [m]$. /* The parameter m' will be specified later. */
2. Factor the polynomial $\tilde{Q}$ and output all its irreducible factors.

Let $Q_1(x, y), \ldots, Q_l(x, y)$ be the irreducible factors of the unknown polynomial $Q^*(x, y)$ describing the black box B. We focus on the factor Q_1. Let the $\{(1, x), (d, y)\}$-weighted degree of Q_1 be $k_1 d$. The following two lemmas essentially show that if the fraction of points (x_i, y_i) for which $Q_1(x_i, y_i) = 0$ is sufficiently larger than k_1 times the fraction of noise, then we can reconstruct the polynomial Q_1.

Lemma 2.12. *For a set of points $\{(x_j, y_j) | j \in [m]\}$, if $\tilde{Q}(x, y)$ and $Q_1(x, y)$ are polynomials of $\{(1, x), (d, y)\}$-weighted degree at most $kd + m'$ and $k_1 d$ respectively, satisfying the properties: (1) $\forall\ j \in [m]$, $\tilde{Q}(x_j, y_j) = 0$ and (2) $|\{j | Q_1(x_j, y_j) = 0\}| > k_1(kd + m')$, then the polynomials $Q_1(x, y)$ and $\tilde{Q}(x, y)$ share a non-constant common factor.*

Proof. The proof is a straightforward modification of the proof of Lemma 2.7. The only change is in Claim 2.8, where the bound on the degree of the resultant $\mathrm{Res}_y(\tilde{Q}, Q_1)$ goes up to $k_1(kd + m')$, because the degree of the non-zero entries in the first k_1 columns goes up by m'. □

Lemma 2.13. *Let B be a (k, d)-algebraic black box described by a bivariate polynomial Q with no repeated non-constant factors. Let $Q_1, \ldots, Q_l$ be the irreducible factors of Q of at most $\{(1, x), (d, y)\}$-weighted degree $k_1 d, \ldots, k_l d$ respectively. Given $m' \leq m$ and m pairs $\{(x_1, y_1), \ldots, (x_m, y_m)\}$, which are the input/output pairs of B on distinct inputs, if there exists an $i \in [k]$ such that $|\{j | Q_i(x_j, y_j) = 0\}| > k_i(m' + kd)$ and $|\{j | Q(x_j, y_j) \neq 0\}| \leq m'$, then a set of at most k bivariate polynomials that includes Q_i can be found in time polynomial in m, k and d.*

Lemma A.2 of the appendix ensures that if $M = \frac{k_i k d}{p - k_i \epsilon} + \frac{16}{(p - k_i \epsilon)^2} \ln \frac{3}{\delta}$ and $|H| \geq \frac{3}{\delta}\binom{M}{2}$, then a sample of M elements $\{x_1, \ldots, x_M\}$ chosen independently and uniformly at random from F satisfies the following three properties:

1. The x_j's have no repeated elements.

2. There are (strictly) less than $((\epsilon + p_i/k_i)/2)M - kd/2$ values of j such that $Q(x_j, B(x_j)) \neq 0$.
3. There are at least $((p_i+k_i\epsilon)/2)M+k_ikd/2$ values of j such that $Q_i(x_j, B(x_j)) = 0$.

Thus if $|H| \geq \frac{3}{\delta}\binom{M}{2}$, then we choose $m = M$ randomly points from H and use $\{(x_1, B(x_1)), \ldots, (x_m, B(x_m))\}$ and $m' = ((\epsilon + p_i/k_i)/2)M - kd/2 - 1$ as input to the algorithm described above. If, on the other hand H is small, then we use all $\{(x, B(x))|x \in H\}$ as the input set and use $m' = \epsilon|H|$ as input to our algorithm. In the latter case, Q_i is guaranteed to be part of the output if $(p_i - k_i\epsilon)|H| > kk_id$. This yields the following theorem.

Theorem 2.14. *Let B be an ϵ-noisy (k, d)-algebraic black box described by a polynomial Q with no repeated non-constant factors. Further, let $Q_1, \ldots, Q_l$ be the (distinct) irreducible factors of Q and let $p_i \overset{\text{def}}{=} \Pr_{x\in H}[Q_i(x, B(x)) = 0]$. There exists an algorithm which takes as input $\delta, p > 0$, runs in time $\text{poly}(k, d, \frac{1}{p-\epsilon}, \frac{1}{\delta})$ and with probability at least $1-\delta$ reconstructs a list of at most k bivariate polynomials which includes every Q_i such that $p_i/k_i \geq p$, provided $(p - \epsilon)|H| > kd$.*

3. Multivariate Black Boxes.

In this section, we extend Theorem 2.14 to multivariate polynomial black boxes over finite fields. The methods of Section 2, i.e., those based on trying to find the coefficients of polynomials simultaneously, do not seem to extend directly to the general multivariate case. This is due to the possibly large *explicit* representation of the function extracted from the black box, which makes it inefficient to work with. Instead, we use techniques of pairwise independent sampling to reduce the problem to a univariate situation and then apply Theorem 2.14 to the new univariate problem. We start by summarizing the problem.

Given: *An n-variate ϵ-noisy (k, d)-polynomial black box $B : F^n \to F$. I.e., there exist n-variate polynomials $f_1, \ldots, f_k$ of total degree at most d such that*

$$\Pr_{\hat{x}\in F^n}[\exists i \in [k] \text{ s.t. } B(\hat{x}) = f_i(\hat{x})] \geq 1 - \epsilon$$

and furthermore, each f_i is well represented in B, i.e.,

$$\forall\ i \in [k] \Pr_{\hat{x}\in F^n}[B(\hat{x}) = f_i(\hat{x})] \geq p > \epsilon.$$

Problem: *Construct k black boxes computing the functions $f_1, \ldots, f_k$.*

Notice that we have changed the problem from that of the previous section in several ways. First, we no longer ask for an explicit representation of f_i, but allow for implicit representations. This is a strengthening of the problem, since explicit representations may be much longer than implicit ones and thus allow a reconstruction algorithm much more time than we do. For instance, if the reconstructed function is a sparse multivariate polynomial, then we can use any of the sparse multivariate polynomial interpolation algorithms given in [3, 20, 21, 40] to recover explicit representations of the reconstructed functions, in running time which is polynomial in the number of non-zero coefficients rather than total number of possible coefficients. A second change from the problem of the previous section is that we expect all the polynomials $f_1, \ldots, f_k$ to be well represented in the black box B. This is a weakening of the problem, and we do not know how to get around it.

The outline of the method we use to solve the above problem is as follows. Consider first the slightly simpler problem: Given B and an input $\hat{b} \in F^n$, find the multiset $\{f_1(\hat{b}), \ldots, f_k(\hat{b})\}$. This we solve by a reduction to a univariate version of the reconstruction problem. Now a solution to this problem does not immediately suffice to yield a solution to the n-variate reconstruction problem as described above. This is because the solution produces a multiset of values $\{y_1, \ldots, y_k\}$ for which we do not know which y_j corresponds to f_i. We want the black boxes to always output according to the same polynomial consistently.

In order to solve this problem, we introduce the notion of a *reference point* $\hat{r} \in F^n$, which will have the property that the value of the k different polynomials will be all distinct on this point. We will then use a more general reduction to the univariate problem which will allow us to reconstruct a set of pairs $\{(y_1, z_1), \ldots, (y_k, z_k)\} = \{(f_1(\hat{b}), f_1(\hat{r})), \ldots, (f_k(\hat{b}), f_k(\hat{r}))\}$ This, along with the property of the reference point, allows us to order the points consistently for all inputs $\hat{b}$. We now go into the details.

3.1. Reference Points. The following definition of a reference point is motivated by the above discussion. We wish to consider the polynomial $Q(x_1, \ldots, x_n; y) = \prod_{i=1}^{k}(y - f_i(x_1, \ldots, x_n))$, and want to ensure that at the reference point $\hat{r}$, $f_i(r_1, \ldots, r_n) \neq f_j(r_1, \ldots, r_n)$, whenever $i \neq j$. One way to test for this is to see if the polynomial $p(y) \stackrel{\text{def}}{=} Q(r_1, \ldots, r_n; y)$ has any repeated non-constant factors. This will be our definition of a reference point. Note that this definition is general enough to apply also to polynomials Q which do not factor linearly in y.

Definition 3.1. *For a multivariate polynomial $Q(x_1, \ldots, x_n; y)$ that has no non-constant repeated factors, a* reference point *is an element $\hat{r} = (r_1, \ldots, r_n)$ of F^n such that the univariate polynomial $p(y) \stackrel{\text{def}}{=} Q(r_1, \ldots, r_n; y)$ has no repeated non-constant factors.*

The next lemma will show that a random point is likely to be a reference point for any given polynomial Q, provided the field size is large compared to the degree of the polynomial Q. We will need one more notion which is standard in algebra.

Definition 3.2. *The discriminant of a univariate polynomial $Q(y)$, denoted Δ, is $Res(Q, Q')$ where Q' is the derivative of Q with respect to y. The discriminant of a multivariate polynomial $Q(x_1, \ldots, x_n; y)$ with respect to y, denoted $\Delta(x_1, \ldots, x_n)$, is defined to be $Res_y(Q, Q')$ where Q' is the derivative of Q with respect to y. (Formally the derivative of a monomial $q_i y^i$ is $(q_i + \cdots + q_i)y^{i-1}$, where the summation is of i q_i's. The derivative of a polynomial is simply the sum of the derivatives of the monomials in it.)*

The above definition is motivated by the following well-known fact: A polynomial p (over any unique factorization domain) has repeated non-constant factors if and only if it shares a common factor with its derivative (cf. [27], Theorem 1.68). From the well-known fact about resultants, this extends to saying that a polynomial has repeated non-constant factors if and only if its discriminant is zero.

Lemma 3.3. *For a polynomial $Q(x_1, \ldots, x_n; y)$ of $\{(1, x_1), \ldots, (1, x_n), (d, y)\}$-weighted degree at most kd with no repeated non-constant factors a random point $\hat{r} \in F^n$ is a reference point with probability at least $1 - \frac{k(k-1)d}{|F|}$.*

Proof. Let $\Delta(x_1, \ldots, x_n)$ be the discriminant of Q with respect to y. Notice that Q' is a polynomial of $\{(1, x_1), \ldots, (1, x_n), (d, y)\}$-weighted degree at most $(k-1)d$.

Thus, as in Claim 2.8 we can show that $\Delta(x_1, \ldots, x_n)$ is a polynomial in $x_1, \ldots, x_n$ of degree at most $k(k-1)d$. Since Q has no repeated factors, $\Delta(x_1, \ldots, x_n)$ is not identically zero. Thus for a random point $\hat{r} \in F^n$, the probability that $\Delta(\hat{r}) = 0$ is at most $k(k-1)d/|F|$. But observe that $\Delta(\hat{r})$ is the discriminant of the univariate polynomial $p(y) \overset{\text{def}}{=} Q(r_1, \ldots, r_n, y)$, and if $\Delta(\hat{r}) \neq 0$, then $\hat{r}$ is a reference point. □

3.2. Reduction to the univariate case. We now consider the case where we are given a black box B, described by polynomials $f_1, \ldots, f_k$, and two points $\hat{a}, \hat{b} \in F^n$, and we wish to find a set of k pairs $\{(y_1, z_1), \ldots, (y_k, z_k)\}$ such that for every $i \in [k]$, there exists some $j \in [k]$ such that $(y_j, z_j) = (f_i(\hat{a}), f_i(\hat{b}))$. We solve this problem by creating a univariate reconstruction problem and then using Theorem 2.14 to solve this problem. This reduction builds upon a method of [14], which in turn builds upon earlier work of [1, 13].

We create a univariate "subdomain", more precisely a function $D : F \to F^n$, such that the image of the domain, $\mathrm{Im}(D) \overset{\text{def}}{=} \{D(t) | t \in F\}$, satisfies the following properties:,

1. $\hat{a}$ and $\hat{b}$ are contained in $\mathrm{Im}(D)$..
2. The restriction of a polynomial $\tilde{Q}(x_1, \ldots, x_n, y)$ of $\{(1, x_1), \ldots, (1, x_n), (d, y)\}$-weighted degree kd to $\mathrm{Im}(D)$, i.e., the function $\tilde{Q}^D(t, y) \overset{\text{def}}{=} \tilde{Q}(D(t), y)$, is a bivariate polynomial of $\{(1, t), (3d, y)\}$-weighted degree $3kd$.
3. $\mathrm{Im}(D)$ resembles a randomly and independently chosen sample of F^n of size $|F|$. In particular, with high probability, the fraction of points from $\mathrm{Im}(D)$ where the black box responds with $f(x_1, \ldots, x_n)$ is very close to the fraction of points from F^n where the black box responds with f.

For a finite field F, with $|F| > 3$, $D = (D_1, \ldots, D_n)$, where $D_i : F \to F$ is constructed by picking vectors $\hat{c} = (c_1, \ldots, c_n)$ and $\hat{d} = (d_1, \ldots, d_n)$ at random from F^n and setting $D_i(t) = a_i + c_i t + d_i t^2 + (b_i - c_i - d_i - a_i)t^3$, for $i \in [n]$. By construction it is immediately clear that the "subdomain" D satisfies properties (1) and (2) listed above. The following lemma shows that it also satisfies property (3) above.

Lemma 3.4. *For sets $S_1, \ldots, S_k, E \subset F^n$, let $p_i \overset{\text{def}}{=} |S_i|/|F^n|$ and $\epsilon \overset{\text{def}}{=} |E|/|F^n|$ and let $\gamma > 0$. Then*

$$\Pr_{\hat{c}, \hat{d}} \left[\exists i \in [k] \text{ s.t. } |\mathrm{Im}(D) \cap S_i|/|F| \leq p_i - \gamma/2 \text{ or } |\mathrm{Im}(D) \cap E|/|F| \geq \epsilon + \gamma/2\right] \leq \frac{k+1}{\gamma^2(|F| - 2)}.$$

Proof. Observe that the set of points $\{D(t) | t \in F \setminus \{0, 1\}\}$, constitute a pairwise independent sample of points chosen uniformly at random from F^n. The lemma now follows from a standard application of Chebyshev bounds. □

Thus we obtain the following algorithm (tuned for confidence parameter $\delta = 1/3$).

The algorithm is given a threshold p.

1. Pick $\hat{c}, \hat{d}$ at random from F^n.
2. Let $D(t) = (D_1(t), \ldots, D_n(t))$ be given by $D_i(t) = a_i + c_i t + d_i t^2 + (b_i - a_i - c_i - d_i)t^3$, and let $B' : H \to F$ be the black box given by $B'(t) = B(D(t))$, where $H = F - \{0, 1\}$.
3. Reconstruct all univariate polynomials $g_1, \ldots, g_k$ of degree at most $3d$ describing B', for threshold p and confidence $1 - 1/6$.

4. Output $\{(g_1(0), g_1(1)), \ldots, (g_k(0), g_k(1))\}$.

Let $\gamma = \sqrt{\frac{6(k+1)}{(|F|-2)}}$. By Lemma 3.4 we know that the above algorithm finds a univariate domain D, s.t. at most $\epsilon + \gamma/2$ fraction of the points on the domain are "noisy" and every polynomial is represented on at least $p_i - \gamma/2$ fraction of the domain, with probability at least $1 - 1/6$. Thus if $p_i \geq p$ for every i, and $(p - \epsilon - \gamma)(|F| - 2) > 3kd$, then the univariate reconstruction algorithm is guaranteed to find all the f_i's, with probability at least $1 - 1/6$. The condition on $|F|$ above can be simplified (somewhat) to $(p - \epsilon)|F| > 3kd + \sqrt{6(k+1)|F|} + 2$ and under this condition the algorithm above returns $\{(f_1(0), f_1(1)), \ldots, (f_k(0), f_k(1))\}$ correctly with probability at least $2/3$. Notice that by repeating $\log \frac{1}{\delta}$ times and outputting the majority answer (i.e., the set that is output most often), we can boost the confidence up to $1 - \delta$, for any $\delta > 0$. This yields the following lemma:

Lemma 3.5. *Given an ϵ-noisy n-variate polynomial black box B described by polynomials $f_1, \ldots, f_k$ of degree d, s.t.*

$$\forall i \in [k], \Pr_{\hat{x} \in F^n} [B(\hat{x}) = f_i(\hat{x})] \geq p > \epsilon,$$

there exists a randomized algorithm that takes as input $\delta, p > 0$ and $\hat{a}, \hat{b} \in F^n$, runs in time $\text{poly}(n, k, d, \frac{1}{(p-\epsilon)}, \log \frac{1}{\delta})$ *and outputs the set of k ordered pairs $\{(f_1(\hat{a}), f_1(\hat{b})), \ldots, (f_k(\hat{a}), f_k(\hat{b}))\}$ with probability at least $1 - \delta$ provided $(p - \epsilon)|F| > 3kd + 2 + \sqrt{6(k+1)|F|}$.*

3.3. Putting it together. We are now ready to describe the algorithm for solving the multivariate reconstruction problem. The algorithm has a preprocessing stage where it sets up k black boxes, and a query processing stage where it is given a query point $\hat{a} \in F^n$ and the black boxes compute $f_i(\hat{a})$.

Preprocessing Stage: Given: Oracle access to a black box B described by polynomials $f_1, \ldots, f_k$. Parameters k, ϵ, p and δ.
- Step 1: Pick $\hat{r}$ at random and $\hat{b}$ at random.
- Step 2: Reconstruct, with confidence $1-\delta$, the set $\{(f_1(\hat{r}), f_1(\hat{b})), \ldots, (f_k(\hat{r}), f_k(\hat{b}))\}$ using the algorithm of Section 3.2.
- Step 3: If the multiset $\{f_1(\hat{r}), \ldots, f_k(\hat{r})\}$ has two identical values, then output "failure". Else pass the reference point $\hat{r}$ and the values $f_1(\hat{r}), \ldots, f_k(\hat{r})$ to the Query Processing Stage.

Query Processing Stage: Given: Oracle access to a black box B, $\hat{b} \in F^n$ and parameters k, d, p and δ. Additionally reference point $\hat{r}$ and values $v_1, \ldots, v_k$ passed on by the Preprocessing Stage.
- Step 1: Reconstruct with confidence δ the set $\{(f_1(\hat{r}), f_1(\hat{b})), \ldots, (f_k(\hat{r}), f_k(\hat{b}))\}$ using the algorithm of Section 3.2.
- Step 2: If the set $\{f_1(\hat{r}), \ldots, f_k(\hat{r})\}$ equals the set $\{v_1, \ldots, v_k\}$ then reorder the indices so that $f_i(\hat{r}) = v_i$ for every $i \in [k]$. If the sets are not identical then report "failure".
- Step 3: For every $i \in [k]$, the black box B_i outputs $f_i(b)$.

This yields the following theorem.

Theorem 3.6. *Let B be an ϵ-noisy n-variate (k, d)-polynomial black box s.t.*

$$\Pr_{\hat{x} \in F^n} [\exists i \in [k], \text{ s.t. } B(\hat{x}) = f_i(\hat{x})] \geq 1 - \epsilon$$

$$\textit{and } \forall i \in [k] \Pr_{\hat{x}\in F^n}[B(\hat{x}) = f_i(\hat{x})] \geq p > \epsilon.$$

Then, if $(p-\epsilon)|F| > 3kd+2+\sqrt{6(k+1)|F|}$, there exists a randomized algorithm that takes as input a confidence parameter δ and with probability $1-\delta$ produces k black boxes B_j such that for every $i \in [k]$ there exists $j \in [k]$ s.t. for every input $\hat{b} \in F^n$, the black box B_j computes $f_i(\hat{b})$ with probability $1-\delta$.

4. Applications.

In this section, we describe the application of our techniques to curve fitting and bivariate polynomial factorization.

4.1. Curve Fitting Problems over Discrete Domains.

In this subsection, we study the curve fitting problem over discret domains. Given a set of m points, with integer coordinates, we show how to find a polynomial with integer coefficients that is Δ-close to all but an ϵ fraction of the points (if such a polynomial exists), where ϵ need only be less than 1/2 (provided is m is larger than $\frac{(4\Delta+1)d}{1-2\epsilon}$). Over Z_p (or over the integers) the problem can be formulated as:

Given: *m pairs of points, $\{(x_1, y_1), \ldots, (x_m, y_m)\}$ and ϵ, such that there exists a polynomial f, of degree at most d, such that for all but ϵm values of j in $[m]$*

$$\exists i \in [-\Delta, \Delta] \textit{ s.t. } y_j = f(x_j) + i$$

Problem: *Find such an f.*

Consider $f_i(x) \stackrel{\text{def}}{=} f(x) + i$, where $i \in [-\Delta, \Delta]$. Notice that all but ϵ fraction of the points are described by the polynomial f_i's. Thus the above problem could be thought of a reconstruction problem for an ϵ-noisy $(2\Delta+1, d)$-polynomial black box reconstruction problem. Lemma 2.13 can now be applied to this set of points to get the following result.

Claim 4.1. *If there exists an i such that the number of points for which $y = f_i(x)$ is strictly more than $\epsilon m + kd$, then we can find a small set of polynomials which includes f_i.*

The weakness of the above procedure is that it can only be guaranteed to succeed if ϵ is smaller than $\frac{1}{2\Delta+2}$, since only then can we guarantee the existence of an i such that the polynomial $f_i(x)$ is represented more often than the noise in the input set. We now present a variation of the above method which gets around this weakness and solves the curve fitting problem for strictly positive values of ϵ (independent of Δ) and in fact works for ϵ arbitrarily close to 1/2.

The idea is that we can artificially decrease the influence of the *bad* points. To do this, we look at the following set of points: $\{(x_1^i, y_1^i), \ldots, (x_m^i, y_m^i)\}_{i=-\Delta}^{\Delta}$, where $x_j^i = x_j$ and $y_j^i = y_j - \Delta + i$. (From each point in the original sample, we generate $2\Delta+1$ points, by adding and subtracting up to Δ to the y coordinate of each point.) We show that these points represent the output of a (k, d)-algebraic black box for $k = \epsilon m + (4\Delta+1)d$.

Observe that the following conditions hold for the $(2\Delta+1)m$ points constructed above:

- There exists a polynomial $Q(x, y)$ of $\{(1, x), (d, y)\}$-weighted degree at most $\epsilon m + (4\Delta+1)d$ such that $Q(x, y) = 0$ for all the points. This is the polynomial: $Q(x, y) = W(x) \cdot \Pi_{i=-2\Delta}^{2\Delta}(y - f_i(x))$, where $f_i(x) = f(x) + i$ and $W(x)$ is the polynomial satisfying $W(x_j) = 0$ if $f_i(x_j) \neq y_j$ for any $i \in [-\Delta, \Delta]$. (Notice that the degree of W is ϵm.)

- At least $(1-\epsilon)m$ of the points satisfy $y = f(x)$. This is because for every point in the original (x_j, y_j) such that y_j is within Δ of $f(x_j)$ (and there were $(1-\epsilon)m$ such points), one of the new (x^i_j, y^i_j) pairs satisfies $y^i_j = f(x^i_j)$.

LEMMA 4.2. *Given m points $\{(x_1, y_1), \ldots, (x_m, y_m)\}$, integer d and $\epsilon < \frac{1}{2}$ there exists a polynomial time algorithm that can find all polynomials f of degree d such that f is Δ-close to all but an ϵ fraction of the points (x_j, y_j), provided $m > \frac{(4\Delta+1)d}{1-2\epsilon}$.*

Proof. Find a polynomial $\tilde{Q}(x,y)$ such that $\tilde{Q}(x,y) = 0$ for all the points $\{(x^i_j, y^i_j)\}^{m,\ \Delta}_{j=1, i=-\Delta}$ such that the degree of $\tilde{Q}$ is at most $\epsilon m + (4\Delta+1)d$. If $\epsilon m + (4\Delta+1)d < (1-\epsilon)m$ then by Lemma 2.3 we know that for every candidate function f which forms an (ϵ, Δ) fit on the given points, $(y - f(x))$ divides $\tilde{Q}$. Thus factoring $\tilde{Q}$ will give us all the candidates. ☐

4.2. Reducing Bivariate Factoring to Univariate Factoring. In Section 2.1, we saw how to reduce the problem of reconstructing total polynomial black boxes to the problem of factoring bivariate polynomials. In the specific case of univariate polynomial black boxes over finite fields, we will also reduce the reconstruction problem to that of factoring univariate polynomials into their irreducible factors. As an interesting consequence, we describe a simple way of reducing the problem of factoring special bivariate polynomials over finite fields to the problem of factoring univariate polynomials.

We first show how to reduce the reconstruction problem to that of factoring univariate polynomials: Suppose we have a black box which on input x outputs the (unordered) set $\{f_1(x), \ldots, f_k(x)\}$, where the f_i's are univariate polynomials, each of degree at most d. Sampling from the black box and interpolating, we can find the polynomial $f(x) = \prod_{i=1}^k f_i(x)$ explicitly (in terms of its coefficients). If somehow we could guarantee that at least one of the f_i's is irreducible, we could factor t to find f_i. Such a guarantee is not available, but we simulate it via randomization.

Let $\alpha(x) \in F[x]$ be a random degree d polynomial. We can convert the given set of sample points so that on each input x we have the (still unordered) set $\{g_1(x), \ldots, g_k(x) : g_i(x) = f_i(x) + \alpha(x)\}$. Each of the polynomials g_i is a random degree d polynomial (but they are not necessarily independent). We then use the fact that random polynomials over finite fields have a reasonable chance of being irreducible.

LEMMA 4.3 ([27], p.84). *The probability $P_q(d)$ that a random polynomial of degree d is irreducible over F_q, is at least $\frac{1}{d}(1 - \frac{1}{q-1})$.*

We can thus interpolate (after sampling at $kd+1$ points) and explicitly compute $g(x) = \prod_{i=1}^k g_i(x)$. We factor g into irreducible factors $r_1 \cdot \cdots \cdot r_l$. For each factor r_j of g, we verify whether or not $r_j - \alpha$ is a candidate for one of the f_i's by checking that it evaluates to one of the outputs of the black box B on all the sampled points. By Lemma 4.3 we know that with non-negligible probability g_i is irreducible and if this happens, we find g_i as one of the factors of g (i.e., as one of the r_j's). Subtracting α from g_i gives us f_i, which will pass the candidacy verification.

LEMMA 4.4. *If a degree d polynomial p agrees with one of the outputs of the black box on $kd+1$ different x's, then p agrees with one of the outputs of the black box on all x's.*

Proof. If p agrees with one of the outputs of the black box on $kd+1$ different x's, then by the pigeonhole principle there is a polynomial f_i which agrees with p on at least $d+1$ different x's. Thus $p \equiv f_i$. □

Thus, no r_j which is not equal to one of the g_i's will pass the candidacy verification. By repeating this procedure enough times and outputting all the candidates, we can reconstruct all the polynomials $\{f_1, \dots, f_k\}$. Straightforward analysis shows that the expected number of times that we need to repeat the process (choose random α) is $O(k/P_q(d))$. Refining the analysis, we can show that $O(\ln k/P_q(d))$ times suffice.

From the above, we get the following algorithm for finding the monic linear factors of a bivariate polynomial $Q(x,y)$.

```
program Simple Factor
repeat $O(\ \ln k/P_q(d)\ )$ times
   pick a random degree $d$ polynomial
         $\alpha(x)$ over $F$
   factor $Q(x,\alpha(x))$
   for every factor $g(x)$ of $Q(x,\alpha(x))$
      if $(y+g(x)-\alpha(x))$ divides $Q(x,y)$
         output $(y+g(x)-\alpha(x))$
end
```

CLAIM 4.5. *Given a bivariate polynomial $Q(x,y)$, over a finite field F, of total degree at most kd, the algorithm* **Simple Factor** *finds all the linear and monic factors of $Q(x,y)$.*

We next extend this mechanism, and apply the reconstruction mechanism of Section 2.2 to the problem of finding the factors of $Q(x,y)$ which are monic and of constant degree in y. Our mechanism tries to isolate some factor $A(x,y)$ of $Q(x,y)$ of the form

$$A(x,y) = y^c + a_{c-1}(x)y^{c-1} + \cdots + a_0(x)$$

where the a_i's are polynomials in x of degree at most d (and c is a constant).

Let $Q(x,y)$ be a polynomial of $\{(1,x),(d,y)\}$-weighted degree kd. For each $i \in [c]$ we construct a program P_i which is supposed to be a (K,d)-algebraic box for a_i, for some $K \le 2ikd\binom{k}{c}$. We then use our reconstruction procedure (Theorem 2.14) to produce, for each $i \in [c]$, a list of at most K polynomials which contains a_i. This, in turn, gives a set of at most K^c polynomials in x and y which contains $A(x,y)$. $A(x,y)$ can be isolated from this set by exhaustive search. The running time of this algorithm is thus some polynomial in $(kd)^{c^2}$.

The program P_i for a_i works as follows on input x_1:

- P_i constructs the polynomial $Q_{x_1}(y) \equiv Q(x_1,y)$ (which is a polynomial in y) and factors Q_{x_1}.
- Let S be the set of factors of Q_{x_1}. (S contains polynomials in y.)
- Let S^c be the set of polynomials of degree c obtained by taking products of polynomials in S.
- P_i picks a random polynomial f in S^c and outputs the coefficient of y^i in f.

We now show that P_i is a $(2ikd\binom{k}{c}, d)$-algebraic black box described by some polynomial $Q^*(x,y)$ such that $y - a_i(x)$ divides $Q^*(x,y)$. Let $Q(x,y) = q_k(x)y^k + q_{k-1}y^{k-1} +$

$\cdots + q_0(x)$. Over the algebraic closure of the quotient ring of polynomials in x, $Q(x,y)$ factors into linear factors in y; let this factorization be

$$Q(x,y) = q_k(x)(y - b_1(x))(y - b_2(x)) \cdots (y - b_k(x)).$$

(The $b_i(x)$ are some functions of x, but not necessarily polynomials.) For $T \subset [k]$, $|T| = c$, let $\sigma_{T,i}(x) \overset{\text{def}}{=} \sum_{S \subset T, |S|=i} \prod_{l \in S} b_l(x)$. Notice that the function $a_i(x)$ that we are interested in is actually $\sigma_{T,i}(x)$ for some T. Notice further that the output of the program P_i is always $\sigma_{T,i}(x)$ for some T (though this T is some arbitrary subset of $[k]$). Thus the input/output pairs (x,y) of the program P_i always satisfy $\prod_{T \subset [k], |T|=c}(y - \sigma_{T,i}(x)) = 0$. Unfortunately, $\sigma_{T,i}(x)$ need not be a polynomial in x. So we are not done yet. We will show that $Q^*(x,y) \overset{\text{def}}{=} (q_k(x,y)^N) \prod_{T \subset [k], |T|=c}(y - \sigma_{T,i}(x))$ is actually a polynomial in x and y of $\{(1,x),(d,y)\}$-weighted degree at most Ki, where $N = i\binom{k}{c}$. To see this, consider the coefficient of y^j in $Q^*(x,y)$. This is $q_k(x)^N$ times some polynomial in $b_1(x), \ldots, b_k(x)$, denoted $g_j(b_1(x), \ldots, b_k(x))$. By definition of Q^*, we notice that g_j is a symmetric polynomial in $b_1(x), \ldots, b_k(x)$ of degree at most $i\binom{k}{c}$. We now invoke the "fundamental theorem of symmetric polynomials" ([27], pages 29–30) which states that a symmetric polynomial of degree D in variables $z_1, \ldots, z_k$ can be expressed as a polynomial of degree at most D in the *primitive* symmetric functions in $z_1, \ldots, z_k$. In our case this translates into saying that, there exists some polynomial $\tilde{g}_j$ of degree at most N s.t. $g_j(b_1(x), \ldots, b_k(x)) = \tilde{g}_j(\frac{q_{k-1}(x)}{q_k(x)}, \ldots, \frac{q_0(x)}{q_k(x)})$ (since the primitive symmetric functions in $b_1(x), \ldots, b_k(x)$ are actually $\frac{q_{k-1}(x)}{q_k(x)}, \ldots, \frac{q_0(x)}{q_k(x)}$. Thus we find that the coefficient of y^j in $Q^*(x,y)$ is a polynomial in x of degree at most $ikd\binom{k}{c}$. The claimed bound on the degree of Q^* now follows easily.

Thus we get the following lemma.

Lemma 4.6. *Given a polynomial $Q(x,y)$ of $\{(1,x),(d,y)\}$-weighted degree kd, there is an algorithm that runs in time polynomial in $(kd)^{c^2}$ which finds all factors of Q that are monic and of degree c in y.*

Acknowledgments. We are very grateful to Avi Wigderson for asking a question that started us down this line of research and for helpful discussions. We are very grateful to Umesh Vazirani for his enthusiasm, his valuable opinions and suggestions, and the time that he spent with us discussing this work. We thank Ronen Basri, Oded Goldreich and Mike Kearns for their comments on the writeup of this paper. We thank Joel Friedman for technical discussions about questions related to the topics of this paper. We also thank the anonymous referees for their extensive reports and catching many blatant as well as subtle errors from an earlier version of this paper.

REFERENCES

[1] D. Beaver and J. Feigenbaum. Hiding Instance in Multioracle Queries. In *Symposium on Theoretical Aspects of Computer Science*, 1990.

[2] M. Ben-Or. Probabilistic Algorithms in Finite Fields In *Proc. 22nd IEEE Symposium on Foundations of Computer Science*, pp. 394–398, 1981.

[3] M. Ben-Or and P. Tiwari. A Deterministic Algorithm for Sparse Multivariate Polynomial Interpolation. In *Proc. 20th Symposium on Theory of Computing*, pp. 301-309, 1988.

[4] E. Berlekamp. Factoring Polynomials over Large Finite Fields. *Mathematics of Computation*, page 713, vol. 24, no. 111, 1970.

[5] E. Berlekamp. Bounded Distance +1 Soft-Decision Reed-Solomon Decoding. In *IEEE Transactions on Information Theory*, pp. 704-720, vol. 42, no. 3, May 1996.

[6] A. Blum and P. Chalasani. Learning Switching Concepts. *Proc. 5th Annual Workshop on Computational Learning Theory*, pp. 231–242, 1992.

[7] M. Blum, M. Luby, R. Rubinfeld. Self-Testing/Correcting with Applications to Numerical Problems. *Journal of Computer and System Sciences*, Vol. 47, No. 3, pp. 549–595, December 1993.

[8] A. Blumer, A. Ehrenfeucht, D. Haussler and M. K. Warmuth. Occam's Razor. *Information Processing Letters* 24 (1987) pp. 377-380.

[9] J.Y. Cai and L. Hemachandra. A note on enumerative counting. *Information Processing Letters*, 38, pp. 215-219, 1991.

[10] Canny J.. Finding edges and lines in images. *M.I.T., Artificial Intelligence Laboratory Report*, AI-TR-720, 1983.

[11] D. Coppersmith, Personal communication to Ronitt Rubinfeld, Fall 1990.

[12] J. von zur Gathen. Algebraic Complexity Theory. *Annual Reviews of Computer Science*, 3:317–347, 1988.

[13] P. Gemmell, R. Lipton, R. Rubinfeld, M. Sudan and A. Wigderson. Self-Testing/Correcting for Polynomials and for Approximate Functions. *Proc. 23rd ACM Symposium on Theory of Computing*, pp.32–42, 1991.

[14] P. Gemmell and M. Sudan. Highly resilient correctors for polynomials. *Information Processing Letters*, 43, pp. 169-174, 1992.

[15] O. Goldreich and L. Levin. A hard-core predicate for any one-way function. *Proceedings of the 21st ACM Symposium on Theory of Computing*, 1989.

[16] O. Goldreich, R. Rubinfeld and M. Sudan. Learning polynomials with queries: The highly noisy case. *Proceedings of the 36th IEEE Symposium on Foundations of Computer Science*, pp. 294–303, 1995.

[17] D. Grigoriev. Factorization of Polynomials over a Finite Field and the Solution of Systems of Algebraic Equations. Translated from *Zapiski Nauchnykh Seminarov Lenningradskogo Otdeleniya Matematicheskogo Instituta im. V. A. Steklova AN SSSR*, Vol. 137, pp. 20-79, 1984.

[18] D. Grigoriev and A. Chistov. Fast decomposition of polynomials into irreducible ones and the solution of systems of algebraic equations. *Soviet Math. Doklady*, 29, pp. 380–383, 1984.

[19] D. Grigoriev and M. Karpinski. Algorithms for Sparse Rational Interpolation. ICSI Tech. Report, TR-91-011, January 1991.

[20] D. Grigoriev, M. Karpinski, M.F. Singer. Interpolation of Sparse Rational Functions Without Knowing Bounds on Exponents. Institut für Informatik der Universität Bonn, Report No. 8539-CS, Nov. 1989.

[21] D. Grigoriev, M. Karpinski, M.F. Singer. Fast Parallel Algorithms for Sparse Multivariate Polynomial Interpolation over Finite Fields. *SIAM Journal on Computing*, Vol. 19, No. 6, pp 1059-1063, Dec. 1990.

[22] D. Haussler. Decision Theoretic Generalizations of the PAC Model for Neural Net and Other Learning Applications. *Information and Computation*, 100, pp. 78-150, 1992.

[23] J. Henderson and R. Quandt. **Microeconomic Theory**, McGraw Hill Book Company, 1958, 1971.

[24] E. Kaltofen. A Polynomial-Time Reduction from Bivariate to Univariate Integral Polynomial Factorization. In *23rd Annual Symposium on Foundations of Computer Science*, pp. 57-64, 1982.

[25] E. Kaltofen and B. Trager. Computing with Polynomials Given by Black Boxes for Their Evaluations: Greatest Common Divisors, Factorization, Separation of Numerators and Denominators. In *29th Annual Symposium on Foundations of Computer Science*, pp. 296–305, 1988.

[26] E. Kushilevitz and Y. Mansour. Learning decision trees using the Fourier spectrum. *Proc. 23rd ACM Symposium on Theory of Computing*, 1991.

[27] R. Lidl and H. Niederreiter. **Introduction to finite fields and their applications**. Cambridge University Press, 1986.

[28] R. Lipton. New directions in testing. In *Distributed Computing and Cryptography*, DIMACS Series on Discrete Mathematics and Theoretical Computer Science, v. 2, pp. 191–202, 1991.

[29] A. K. Lenstra, H. W. Lenstra and L. Lovasz. Factoring polynomials with rational coefficients. Math. Ann., 261, pp. 515–534, 1982.

[30] D. Marr and E. Hildreth. Theory of edge detection. *Proceeding of the Royal Society London, B207*, pp. 187-217, 1980.

[31] R. Motwani and P. Raghavan. **Randomized Algorithms**. Cambridge University Press, 1995.

[32] R. Rivest. Learning Decision Lists. *Machine Learning*, 2(3), pp. 229-246, 1987.

[33] T.J. Rivlin. **An Introduction of the Approximation of Functions**. Dover Publications, 1969.

[34] R. Rubinfeld and R. Zippel. A new modular interpolation algorithm for factoring multivariate polynomials. *Proc. Algorithmic Number Theory Symposium*, 1994.

[35] M. Sudan. Decoding of Reed Solomon codes beyond the error-correction bound. *Journal of Complexity*, 13(1):180-193, March 1997.

[36] L. Valiant, A Theory of the Learnable, *Communications of the ACM*, Vol. 27, No. 11, pp. 1134-1142, November 1984.

[37] Van der Waerden. **Algebra**, Volume 1. Frederick Ungar Publishing Co., Inc., page 82.

[38] R. J. Walker. **Algebraic Curves**. Princeton University Press, 1950.

[39] L. Welch and E. Berlekamp. Error Correction of Algebraic Block Codes. *US Patent Number 4,633,470*, issued December 1986.

[40] R.E. Zippel. Probabilistic Algorithms for Sparse Polynomials. In *Proceedings of the European Conference on Symbolic and Algebraic Manipulation '79*, Springer Verlag LNCS, v. 72, pp. 216-226, 1979.

[41] R.E. Zippel. Interpolating Polynomials from their Values. *J. Symbolic Computation* 9, pp. 375-403, 1990.

Appendix A. Appendix.

LEMMA A.1. *Given $S \subset H$, with $|S|/|H| = p$, if $|H| \geq \frac{2}{\delta}\binom{M}{2}$, and $M \geq \frac{2}{p}(kd + \ln\frac{2}{\delta})$, then the probability that M points $x_1, \ldots, x_M$ chosen uniformly at random from H turn out to be distinct and satisfy $|\{j|x_j \in S\}| \geq kd$ is at least $1-\delta$.*

Proof. First observe that for a given $i \neq j$, the probability that $x_i = x_j$ is exactly $1/|H|$. Thus the probability that there exists i, j s.t. $x_i = x_j$ is at most $\binom{M}{2}/|H| \leq \delta/2$.

Now for the second part we use Chernoff bounds (in particular we use a bound from [31], Theorem 4.2). Let X denote the number of elements j s.t. $x_j \in S$. Let $\mu = Mp$ denote the expected value of X. Then $\Pr[X \leq K] \leq \exp(-(\mu - K)^2/2\mu)$. Plugging in the values of $\mu = 2(kd + \ln\frac{2}{\delta})$ and $K = kd$ and simplifying we find that $\exp(-(\mu - K)^2/2\mu) \leq \delta/2$.

Thus the probability that either of the above two events happen is bounded by at most δ as desired. □

Lemma A.2. *Given $S, E \subset H$, with $|S|/|H| = p$ and $|E|/|H| = \epsilon$ if $|H| \geq \frac{3}{\delta}\binom{M}{2}$, and $M \geq \frac{k_1 kd}{p-k_1\epsilon} + \frac{16}{(p-k_1\epsilon)^2}\ln\frac{3}{\delta}$, then the probability that M points $x_1, \ldots, x_M$ chosen uniformly at random from H turn out to be distinct and satisfy $|\{j|x_j \in S\}| > M(p+k_1\epsilon)/2 + k_1kd/2$ and $|\{j|x_j \in E\}| < M(p/k_1 + \epsilon)/2 - kd/2$ is at least $1-\delta$.*

Proof. First we argue as above that the probability that the x_j's are not distinct is at most $\delta/3$.

Now for the second part we again use Chernoff bounds (this time we use the bounds from [31], Theorem 4.2 and Theorem 4.3). Let X denote the number of elements j s.t. $x_j \in S$. The above mentioned bounds translate to show that the probability that the number of points from S is less than $Mp - \lambda\sqrt{Mp}$ is at most $\exp(-\lambda^2/2)$. Similarly the probability that the number of points from E turns out to be more than $M\epsilon + \lambda\sqrt{M\epsilon}$ is bounded by at most $\exp(-\lambda^2/4)$. Each of these probabilities is at most $\delta/3$ if we choose $\lambda = 2\sqrt{\ln\frac{3}{\delta}}$. The lemma now follows from the fact that for the chosen value of M, we have that $M\epsilon + \lambda\sqrt{M} < M(p/k_1 + \epsilon)/2 - kd/2$ and $Mp + \lambda\sqrt{M} > M(p + k_1\epsilon)/2 + k_1kd/2$. □