Algorithms for Computing the Sparsest Shifts of Polynomials via the Berlekamp/Massey Algorithm*

Mark Giesbrecht[1] Erich Kaltofen[2] Wen-shin Lee[1]

[1]Dept. of Computer Science, University of Waterloo, Waterloo, N2L 3G1, Canada
{mwg,ws2lee}@scg.uwaterloo.ca; http://www.uwaterloo.ca/~mwg http://www.wen-shin.com

[2]Dept. of Mathematics, North Carolina State University, Raleigh, North Carolina 27695-8205, USA
{kaltofen}@math.ncsu.edu; http://www.kaltofen.net

1. INTRODUCTION

Let $f(x_1,\ldots,x_n) \in \mathsf{D}[x_1,\ldots,x_n]$ be a multivariate polynomial whose coefficients are in an integral domain D. A sparsest shift is a vector $(s_1,\ldots,s_n) \in \mathsf{K}$, where K is the field generated by the coefficients of f, such that for

$$f(x_1,\ldots,x_n) = \sum_{i=1}^{t} c_i (x_1+s_1)^{e_{i,1}} \cdots (x_n+s_n)^{e_{i,n}},$$

where $c_i \in \mathsf{K} \setminus \{0\}$, the number of (shifted) terms t is minimized. The problem of computing an absolutely sparsest shift seeks a vector $(s_1,\ldots,s_n) \in \overline{\mathsf{K}}^n$, where $\overline{\mathsf{K}}$ is the algebraic closure of K, such that the number of shifted terms is minimized. Those sparsest shifts need not be unique: consider $f = x^2+x+1$. There are 3 absolutely sparsest shifts with $t=2$, namely $s=-1/2$, $s=\rho_1$ and $s=\rho_2$ where $f=(x-\rho_1)(x-\rho_2)$. However, in [16] it is shown that in the univariate case ($n=1$) over a field of characteristic 0 the absolutely sparsest shift is a unique element in K whenever $t \le (\deg(f)+1)/2$. We note that some generalizations of the uniqueness properties exist to the multivariate case [11, 10]. In section 4 we generalize the problem further by specifying an additional input set S to which the shift vector shall be restricted.

Sparse shifts can dramatically reduce the size of the answer of a symbolic expression. A classical example, due to Joel Moses, is $\int 1+(x+1)^n \mathrm{d}x = x+(x+1)^{n+1}/(n+1)$. Sparse shifts can be useful when interpolating the black box polynomial outputs of the algorithms in [13], say the black box for the irreducible factors of a matrix determinant with symbolic entries. It is possible that a sparse shift can make a factor manageable, while the standard representation, in Knuth's words [15], "would fill the universe." Algorithms for computing a sparse shift could therefore be considered simplification tools.

We give a new class of algorithms for efficiently computing a sparsest or absolutely sparsest shift. Our algorithms are based on the early termination version [12] of the Ben-Or/Tiwari sparse interpolation algorithm [1]. The main idea is that for a *symbolic* set of interpolation points, a shift must be a root of a discrepancy in the Berlekamp/Massey algorithm [18], which is called by the Ben-Or/Tiwari method. A sparsest shift is the first such zero to occur. We note that our approach is similar to [7], who use Wronskians (see [9]) in place of discrepancies. Here we can assume that the input polynomial f is being interpolated and we are given a black box procedure for its evaluation. For coefficient fields of small cardinality we require that the black box allows evaluations on points from an extension field [8], which can be realized in a computer program as the so-called extended domain black box object of [5]. We note that for sake of efficiency it is sometimes useful to compute the coefficients of f via interpolation before employing our methods.

Through randomization we can dramatically improve the efficiency of our algorithms. Our randomization is of the Las Vegas kind—always correct and probably fast—because one may always check a candidate sparsest shift via any of the variants of the Ben-Or/Tiwari sparse interpolation algorithm. First, we may choose random values as interpolation points rather than symbolic ones, and employ the probabilistic analysis of [4, 21, 20]. In the univariate case we replace the polynomial root finder by a GCD procedure. This is possible since the sparsest shifts are the roots of a sequence of discrepancies. We can provide a complete probabilistic analysis when the algorithm is run on two independent trials or when all discrepancies up to $2\deg(f)$ are considered. We propose the use of the GCD of two or three subsequent discrepancies in practice, but so far this must be considered a heuristic. For $\mathsf{K}=\mathbb{Q}$ and again $n=1$, we can further eliminate the indeterminate shift variable in our algorithm by evaluating at random integers such that the shift is determined through a large prime factor. For that case we thus have (heuristically) reduced our method to a single Berlekamp/Massey run on rational numbers; we can provide proof for a method that uses 10 independent trials with the provision that the sparsest shift is unique, for instance, when $t \le (\deg(f)+1)/2$.

*This material is based on work supported in part by Natural Sciences and Engineering Research Council of Canada and the Ontario Research & Devopment Challenge Fund (Giesbrecht, Lee) and by the National Science Foundation under Grant Nos. DMS-9977392, CCR-9988177 and CCR-0113121 (Kaltofen).

ISSAC 2002, July 7-10, 2002, Lille, France
©2000 ACM 1-58113-484-3/ 02/ 0007 $5.00

The running times of our methods compare favorably with the previously best algorithms [7, 11, 16, 10]. Not accounting for the length of the intermediately computed scalars, our method at its best, in the univariate rational case when no symbolic value for the shift is carried along, requires $O(t^2)$ operations and $O(t)$ evaluations of f. The algorithm in [16] uses $O(t^2 \deg(f)+t^5)$ arithmetic operations and $4t+2$ values of f and its derivatives. We note that [7] have established the problem to be in polynomial-time. Both ours and Lakshman & Saunders's estimates are given under the assumption that quadratic time polynomial multiplication is employed. The slower version of our univariate algorithm, which carries a symbolic value for the shift, requires $O(t^3 \deg(f) + t^2 \deg(f)^2 \log t)$ operations and $O(t^2 \deg(f))$ polynomial evaluations, which for small t is worse than [16]. However, only a careful experimental comparison of both algorithms, which also controls the length of the intermediate scalars can fully determine which of these two algorithms is better. We intend to do this in the immediate future.

As a sub-procedure our algorithm executes the Berlekamp/Massey algorithm on a sequence of large integers or polynomials. We give a fraction-free version of the Berlekamp/Massey algorithm, which does not require rational numbers or functions and GCD operations on the arising numerators and denominators. The relationship between the solution of Toeplitz systems, Padé approximations, and the Euclidean algorithm is classical. Fraction-free versions [3] can be obtained from the subresultant PRS algorithm [2]. Dornstetter [6] gives an interpretation of the Berlekamp/Massey algorithm as a partial extended Euclidean algorithm. We map the subresultant PRS algorithm onto Dornstetter's formulation. We note that the Berlekamp/Massey algorithm is more efficient than the classical extended Euclidean algorithm.

2. THE FRACTION-FREE BERLEKAMP/ MASSEY ALGORITHM

The Berlekamp/Massey algorithm [18] processes a sequence of elements $a_0, a_1, \ldots$ from a field K. If the sequence is linearly generated, the algorithm determines its minimal generator $\Lambda(z) = z^t + \lambda_{t-1}z^{t-1} + \cdots + \lambda_0$ after processing $2t$ elements from the sequence. When the sequence is from an integral domain D, such as $\mathbb{Z}$ or $\mathsf{K}[x]$, the Berlekamp/Massey algorithm may introduce fractions, making data representation and modularization more difficult. In this section we present a fraction-free version of the Berlekamp/Massey algorithm, which never introduces an element in the field of fractions of D, and in which all divisions are exact.

Recall the pseudo-division of polynomials [15, pp. 425–426, also pp. 428–429] in the fundamental theorem of subresultants [2]. Based on the equivalence between the Berlekamp/Massey algorithm and the extended Euclidean algorithm on polynomials x^{2t} and $a_0x^{2t-1}+a_1x^{2t-2}+\cdots$, Dornstetter [6] interpreted the discrepancies Δ_i as the coefficients in their polynomial remainder sequence (PRS). By computing the corresponding Berlekamp/Massey quantities of the subresultant GCD algorithm, we obtain our fraction-free Berlekamp/Massey algorithm, which outputs an integral multiple of the minimal generator, $\bar{\Lambda}(z) = r \cdot \Lambda(z)$ with $r \in \mathsf{D} \setminus \{0\}$.

Algorithm: `FractionFreeBerlekampMassey`

For $a_0, a_1, \ldots$ from an integral domain D, compute $\bar{\Lambda}_i$, an integral multiple of the minimal generator of $a_0, a_1, \ldots, a_i$.

(1) $\bar{\Lambda}_0^{(rev)} \leftarrow 1$; $B_0 \leftarrow 0$; $L_0 \leftarrow 0$; $\Delta \leftarrow 1$; $g \leftarrow 1$; $h \leftarrow 1$

For $i = 1, 2, \ldots$ **Do**

(2) $\Delta_i \leftarrow \bar{\lambda}_s a_{i-1} + \bar{\lambda}_{s-1}a_{i-2} + \cdots + \bar{\lambda}_0 a_{i-s-1}$;
If $\Delta_i = 0$ then
$\bar{\Lambda}_i^{(rev)} \leftarrow \bar{\Lambda}_{i-1}^{(rev)}$; $B_i \leftarrow z \cdot B_{i-1}$; $L_i \leftarrow L_{i-1}$;

(3) If $\Delta_i \neq 0$ and $2L_{i-1} < i$ then
$\delta \leftarrow i - L_{i-1}$; $\bar{\Lambda}_i^{(rev)} \leftarrow -\Delta \cdot \Delta_i^{\delta} \cdot \bar{\Lambda}_{i-1}^{(rev)} + \Delta_i^{\delta+1} \cdot z \cdot B_{i-1}$;
$B_i \leftarrow \bar{\Lambda}_{i-1}^{(rev)}$; $L_i \leftarrow i - L_{i-1}$; $\Delta \leftarrow \Delta_i$;

(4) If $\Delta_i \neq 0$ and $2L_{i-1} \geq i$ then
$\bar{\Lambda}_i^{(rev)} \leftarrow \bar{\Lambda}_{i-1}^{(rev)} - (\Delta_i/\Delta) \cdot z \cdot B_{i-1}$; $B_i \leftarrow z \cdot B_{i-1}$;
$L_i \leftarrow L_{i-1}$;

(5) If $2L_{i-1} = i$ then
$\bar{\Lambda}_i^{(rev)} \leftarrow (-1)^{\delta+1}/(g \cdot h^{\delta}) \cdot \bar{\Lambda}_i^{(rev)}$;
$g \leftarrow \Delta$; $h \leftarrow h^{1-\delta} \cdot g^{\delta}$;

End For;

3. SPARSE INTERPOLATION ON SHIFTED BASES WITH EARLY TERMINATION

A given polynomial $f(x_1, \ldots, x_n) \in \mathsf{D}[x_1, \ldots, x_n]$ is represented in the standard power basis as:

$$f(x_1, \ldots, x_n) = \sum_{i=1}^{t} u_i \cdot x_1^{e_{1,i}} \cdots x_n^{e_{n,i}}, \qquad (1)$$

with $u_i \in \mathsf{D} \setminus \{0\}$. For any $s = (s_1, \ldots, s_n) \in \overline{\mathsf{K}}^n$, f can be represented in the s-shifted power basis:

$$f(x_1, \ldots, x_n) = \sum_{i=1}^{t(s)} c_i \cdot (x_1 + s_1)^{e_{1,i}} \cdots (x_n + s_n)^{e_{n,i}}, \quad (2)$$

with $c_i \in \overline{\mathsf{K}} \setminus \{0\}$, and $t(s)$ the number of terms. Here, $t(s)$, $e_{j,i}$, and c_i are all dependent on s; the enumeration in i depends on the term order being used, and c_i can be elements from an algebraic extension that allows the shift. The representation in (1) is a special case of (2) when $s = (0, \ldots, 0)$, and f can be represented in the power basis of $y_j = x_j + s_j$:

$$\begin{aligned} f(x_1, \ldots, x_n) &= f(y_1 - s_1, \ldots, y_n - s_n) \\ &= \sum_{i=1}^{t(s)} c_i \cdot (y_1 - s_1 + s_1)^{e_{1,i}} \cdots (y_n - s_n + s_n)^{e_{n,i}} \\ &= \sum_{i=1}^{t(s)} c_i \cdot y_1^{e_{1,i}} y_2^{e_{2,i}} \cdots y_n^{e_{n,i}}. \qquad (3) \end{aligned}$$

Denote the terms in (3) by $M_i(s, y_1, \ldots, y_n) = y_1^{e_{1,i}} \cdots y_n^{e_{n,i}}$ and their evaluations at p_j as $m_i(s) = M_i(s, p_1, \ldots, p_n)$. Now consider the following instance: $p_1 = 2$, $p_2 = 4$, then $m_i(s) = 4$ could be the value of either y_1^2 or y_2. For a given shift s, we shall choose values p_j such that the exponents $e_{1,i}, \ldots, e_{n,i}$ in each $m_i(s)$ can be uniquely determined, and define an auxiliary polynomial with the leading coefficient $\lambda_{t(s)} = 1$:

$$\Lambda(z) = \prod_{i=1}^{t(s)} (z - m_i(s)) = \lambda_{t(s)} z^{t(s)} + \cdots + \lambda_0. \qquad (4)$$

Theorem 1 shows that when a shift s and an upper bound σ, $\sigma \geq t(s)$, are given, the target polynomial can be interpolated in the s-shifted power basis after 2σ evaluations.

THEOREM 1. *Given are a shift $s = (s_1, \ldots, s_n)$ and a polynomial $f(x_1, \ldots, x_n) \in \mathsf{K}[x_1, \ldots, x_n]$. Let $a_i(s) = f(p_1^i - s_1, \ldots, p_n^i - s_n)$ for $i \geq 0$ and assume that $m_i(s)$ are distinct. The sequence $\{a_i(s)\}_{i\geq 0}$ is linearly generated by $\Lambda(z)$ in (4). Furthermore, $\Lambda(z)$ is the minimal generator.*

PROOF. This is an application of the Ben-Or/Tiwari algorithm [1] in the power basis of $y_j = x_j + s_j$. $\square$

Applying the early termination Ben-Or/Tiwari algorithm [12] to the power basis of $y_j = x_j + s_j$, then when p_j are distinct random values, without σ supplied as an input, the target polynomial can be interpolated in the given s-shifted basis with high probability. It requires $2t(s) + \zeta$ black box evaluations, and the threshold $\zeta \geq 1$ is given as input.

Algorithm: `Early Termination Ben-Or/Tiwari on a shifted basis`

Input:
- $f = f(x_1, \ldots, x_n)$, a black box polynomial.
- $s = (s_1, \ldots, s_n)$, a shift in the power basis.
- $\zeta \geq 1$, the threshold for early termination.

Output:
- c_j and $M_j(s)$: $f = \sum_{j=1}^{t(s)} c_j \cdot M_j(s)$ with high probability, or an error message if the procedure fails to complete.

(1) Pick distinct random elements: $p_1, \ldots, p_n$, $p_j \notin \{0, 1\}$.
For $i = 1, 2, \ldots$
(2) Perform the Berlekamp/Massey algorithm on $\{f(p_1^j - s_1, \ldots, p_n^j - s_n)\}_{1\leq j\leq i}$.
If $\Delta_i = 0$ and $i > 2L$ happens ζ times in a row, then
(3) Break out of the loop;
(4) Set $\Lambda(z)$ to the reverse of $\Lambda_i^{(rev)}(z)$ that was computed inside the algorithm;
(5) Compute all the roots of $\Lambda(z)$ in the domain of p_j.
If $\Lambda(z)$ does not completely factor, or not all the roots are distinct, then the early termination was false.
(6) Otherwise, recover the $M_j(s)$ from $m_j(s)$ based on the values of $p_1, \ldots, p_n$. (For example, in a unique factorization domain, one may choose $p_1, \ldots, p_n$ as distinct primes, then each $m_j(s)$ can be uniquely determined).
Again, the term recovery might fail for unlucky p_j.
(7) Obtain all c_j from the $m_j(s)$'s via solving a transposed Vandermonde system.

4. FINDING SPARSE SHIFTS USING EARLY TERMINATION

Based on the sparse interpolation algorithm for shifted bases from the previous section, we present a new class of algorithms for finding sparse shifts. The steps in a sparse algorithm are sensitive to the sparsity of the target polynomial, and we leave the shifts as variables in the procedures. Then we find the solutions to the shift variables that minimize the algorithm steps.

In the introduction, we have introduced the notions of a sparsest shift in K and an absolutely sparsest shift in $\overline{\mathsf{K}}$. We now generalize the problem formulation by constraining the sparsest shifts to arbitrary sets. A sparsest shift *within the set* S is a vector $(s_1, \ldots, s_n) \in S$, where $\varnothing \neq S \subseteq \overline{\mathsf{K}}^n$, such that with $c_i \in \overline{\mathsf{K}} \setminus \{0\}$,

$$f(x_1, \ldots, x_n) = \sum_{i=1}^{t(s)} c_i (x_1 + s_1)^{e_{i,1}} \cdots (x_n + s_n)^{e_{i,n}},$$

the number of (shifted) terms $t(s)$ is minimized. For the problem of computing a sparsest shift we have $S = \mathsf{K}^n$, and for computing an absolutely sparsest shift we have $S = \overline{\mathsf{K}}^n$. In either case S need not be specified on input. A fourth notion is that of a T*-sparse shift* (within S) which is a point $s = (s_1, \ldots, s_n) \in S$ such that for number of shifted terms we have $t(s) \leq T$. Algorithms for computing all T-sparse shifts take T as an additional input. Our main ideas easily generalize to these variants. An application of restricting to a set S is to leaving a selected variable unshifted by making the corresponding component in S equal to 0. We will require unshifted variables in our generalizations to simultaneously sparsely shifting a set of polynomials (see section 5).

We introduce n indeterminates $z_1, \ldots, z_n$ to serve as shift variables. Let $\alpha_i = f(y_1^i - z_1, \ldots, y_n^i - z_n)$ for $i \geq 1$ and apply the fraction-free Berlekamp/Massey algorithm to $\{\alpha_i\}_{i\geq 1}$, a sequence of polynomials in $\overline{\mathsf{K}}[z_1, \ldots, z_n]\,[y_1, \ldots, y_n]$. The discrepancies Δ_i are polynomials in $y_1, \ldots, y_n$ over $\overline{\mathsf{K}}[z_1, \ldots, z_n]$. The following lemma is based on the early termination proof [12] in the standard power basis of y_i.

LEMMA 1. *When a shift $s = (s_1, \ldots, s_n) \in \overline{\mathsf{K}}^n$ is given, the discrepancies Δ_i evaluated at $z_j = s_j$ are non-zero polynomials in y_i for all $1 \leq i \leq 2t(s)$, and zero polynomials for all $i \geq 2t(s) + 1$.*

Thus, we seek sparsest shifts for f within S by finding $\theta = (\theta_1, \ldots, \theta_n) \in S$ that minimize i such that $\Delta_i(\theta_1, \ldots, \theta_n, y_1, \ldots, y_n) = 0$. All our algorithms manipulate the discrepancies $\Delta_i \in \overline{\mathsf{K}}[z_1, \ldots, z_n][y_1, \ldots, y_n]$ from the fraction-free Berlekamp/Massey algorithm. We present our algorithms in three categories. The *Symbolic Shift Algorithms* of subsection 4.1 treat the Δ_i as polynomials in $\overline{\mathsf{K}}[z_1, \ldots, z_n][y_1, \ldots, y_n]$ and work in deterministic polynomial time for constant n over any field over which algebraic systems can be solved. The *Single Projection Algorithms* in subsection 4.2 evaluate each y_j at a value p_j to increase efficiency. Finally, in subsection 4.3, we present the *Double Projection Algorithm* for polynomials $f \in \mathbb{Q}[x]$, wherein the Δ_i are evaluated at random $y = p \in \mathbb{Z}$ as well as random shifts $z = s \in \mathbb{Z}$. This yields a particularly efficient algorithm for rational polynomials.

For simplicity some algorithms are described as finding sparse shifts within certain algebraic extensions, yet they can all be modified as being restricted to a non-empty subset S.

4.1 Symbolic Shift Algorithms

The deterministic symbolic algorithms treat both the basis elements y_i and the shift variables z_i as indeterminates.

Algorithm: `MultivariateSparsestShifts <symbolic>`
Given a polynomial $f(x_1, \ldots, x_n) \in \mathsf{D}[x_1, \ldots, x_n]$, find all the sparsest shifts $s = (s_1, \ldots, s_n) \in S \subset \overline{\mathsf{K}}^n$ for f.

(1) [Initialize.] $i \leftarrow 1$.

(2) [Compute Δ_i.] Let $\alpha_i = f(y_1^i - z_1, \ldots, y_n^i - z_n)$, and update Δ_i by the fraction-free Berlekamp/Massey algorithm on $\alpha_1, \ldots, \alpha_i$.

(3) [Find sparsest shifts.] All solutions $s = (s_1, \ldots, s_n) \in S$ to $\Delta_i(s_1, \ldots, s_n, y_1, \ldots, y_n) = 0$ are sparsest shifts for f within S. The s can be found by solving a system of polynomial equations in $z_1, \ldots, z_n$. If no such s exists then $i \leftarrow i + 1$ and go to step (2).

By lemma 1, a zero of Δ_k stays a zero of Δ_i for all $i \geq k$; those shifts that make Δ_{2T+1} the zero polynomial (in y_j) are all s such that $t(s) \leq T$. For a given T, we find T-sparse shifts in S by solving all $s \in S$ such that $\Delta_{2T+1}(s_1, \ldots, s_n, y_1, \ldots, y_n) = 0$.

We add that for multivariate polynomials, transcendental shifts are possible, for instance $x_1 + x_2 - 1 = (x_1 + z_1) + (x_2 - z_1 - 1)$. In this case the variety of shift points is of dimension higher than 0.

For a univariate $f(x)$, a number of special "tricks" can be employed. In a z-shifted basis $y = x + z$, the discrepancies $\Delta_i \in \mathsf{K}[z][y]$ from the fraction-free Berlekamp/Massey algorithm are viewed as polynomials in y with coefficients in $\mathsf{K}[z]$. Every Δ_i can be written as a product of its primitive part $\varphi_i(y) \in \mathsf{K}[z][y]$ and content $g_i \in \mathsf{K}[z]$ as

$$\Delta_i = g_i \cdot \varphi_i(y). \tag{5}$$

Since $f(x)$ is a non-zero polynomial, $\Delta_i \neq 0$. As i is being increased from 1, a sparsest shift θ appears at the first i such that Δ_i becomes a zero polynomial in y, that is, when $g_i = 0$. If a shift can be in $\overline{\mathsf{K}}$, the first time g_i is a non-trivial polynomial in $\mathsf{K}[z]$, the solutions to $g_i = 0$ are the absolutely sparsest shifts for f. Since all zeros of g_i are zeros of g_{i+1}, we find the first non-trivial GCD of g_i and g_{i+1} in $\mathsf{K}[z]$.

Algorithm: `UnivariateSparsestShifts <symbolic>`
Given a polynomial $f(x) \in \mathsf{D}[x]$, find all the absolutely sparsest shifts for f. This algorithm requires a root finder in $\mathsf{K}[z]$.

(1) [Initialize.] $i \leftarrow 1$, $g_0 \leftarrow 1$.

(2) [Compute Δ_i.] Let $\alpha_i = f(y^i - z)$. Update Δ_i by the fraction-free Berlekamp/Massey algorithm on the sequence $\alpha_1, \ldots, \alpha_i$.

(3) [GCD of g_i and g_{i-1}.] Compute the content of $\gcd(\Delta_i, \Delta_{i-1})$, which is $\gcd(g_i, g_{i-1})$. If it is a non-trivial polynomial in $\mathsf{K}[z]$, all zeros of $\gcd(g_i, g_{i-1})$ are absolutely sparsest shifts; if not, $i \leftarrow i + 1$ and go to step (2).

Similarly, for a univariate polynomial, we solve $\gcd(g_{2T+1}, g_{2T+2})$=0 to find all T-sparse shifts.

4.2 Single Projection Algorithms

By projecting variables y_j to values p_j, the efficiency of the above algorithms can be improved substantially. There is a trade-off: the output might include wrong results. In addition, the results become probabilistic when the p_j are random.

The next algorithm is based on the early termination.

Algorithm: `MultivariateSparseShiftsEquation <one projection>`
Given a polynomial $f(x_1, \ldots, x_n) \in \mathsf{D}[x_1, \ldots, x_n]$ and a positive integer T, return a polynomial equation that all T-sparse shifts have to satisfy.

(1) [Initialize.] Choose distinct random values $p_1, \ldots, p_n$.

(2) [Compute Δ_{2T+1}.] Let $\alpha_i = f(p_1^i - z_1, \ldots, p_n^i - z_n)$, compute Δ_{2T+1} by the fraction-free Berlekamp/Massey algorithm on $\alpha_1, \ldots, \alpha_{2T+1}$.

(3) [Return $\Delta_{2T+1} = 0$.] Every T-sparse shift $s = (s_1, \ldots, s_n)$ has to satisfy $\Delta_{2T+1}(s_1, \ldots, s_n) = 0$, where Δ_{2T+1} is a polynomial in $z_1, \ldots, z_n$.

If we restrict the set of shifts within a set S, the single constraint $\Delta_{2T+1} = 0$ may be sufficient to locate all T-sparse shifts within S. Additional equations can be generated by running the algorithm for different random p_j's. We still need to prove that eventually all false solutions, that is zeros that do not yield a T-sparse shift, are eliminated. In the univariate case we can give a completely proven algorithm.

In the univariate case, we can find either the sparsest shifts or T-sparse shifts through solving a polynomial equation. Consider $\Delta_i = g_i \cdot \varphi_i(y)$ in (5) and distinct random values p, q. By the Schwartz-Zippel lemma, with high probability $\gcd(\Delta_i(p), \Delta_i(q)) = \gcd(g_i \cdot \varphi_i(p), g_i \cdot \varphi_i(q)) = g_i$ and our next algorithm follows.

Algorithm: `UnivariateSparsestShifts <one projection, two sequences>`
Given a polynomial $f(x) \in \mathsf{D}[x]$, this algorithm returns all the absolutely sparsest shifts for f with high probability.

(1) [Initialize.] $i \leftarrow 1$.

(2) [Compute Δ_i] Let p, q be distinct random values and $\alpha_i = f(p^i - z)$, $\beta_i = f(q^i - z)$. Update $\Delta_i(p)$ and $\Delta_i(q)$ by the fraction-free Berlekamp/Massey algorithm on the sequences: $\alpha_1, \ldots, \alpha_i$ and $\beta_1, \ldots, \beta_i$. As a reminder, α_i, β_i, $\Delta_i(p)$, $\Delta_i(q)$ are all polynomials in z.

(3) [GCD of $\Delta_i(p)$ and $\Delta_i(q)$.] If $\gcd(\Delta_i(p), \Delta_i(q))$ is non-trivial in $\mathsf{K}[z]$, the common roots of $\Delta_i(p)$ and $\Delta_i(q)$ are absolutely sparsest shifts for f; if $\gcd(\Delta_i(p), \Delta_i(q))$ is trivial in $\mathsf{K}[z]$, $i \leftarrow i + 1$ and go to step (2).

To further increase the probability of correctness, we may project y to more distinct random values $p_1, \ldots, p_k$, form a sequence for each of the values, then look for the first i such that $\gcd(\Delta_i(p_1), \ldots, \Delta_i(p_k))$ is non-trivial in $\mathsf{K}[z]$.

We can reduce the projection to a single sequence by taking GCD's of subsequent elements in the sequence. Recall the primitive part of Δ_i in (5), in single projection, we need to also make sure there is no non-trivial GCD of $\varphi_i(p)$ and $\varphi_{i+1}(p)$ in $\mathsf{K}[z]$ for all p.

THEOREM 2. *Suppose that the absolutely sparsest shift of $f(x)$ has $\tau < \deg(f)$ terms and assume that $\binom{\deg(f)}{j} \neq 0$*

for all $0 < j < \deg(f)$ when computed as an element in D. Then for $\Gamma = \mathrm{GCD}_{2\tau+1 \le i \le 2\deg(f)}(\Delta_i(z,y))$ (over the quotient field of D) we have $\Gamma = g(z)\gamma(y)$ were $g(z) \in \mathsf{D}[z]$ and $\gamma(y) \in \mathsf{D}[y]$.

PROOF. As stated above, if $\Gamma(\theta, y) = 0$ for some θ in the algebraic closure of the quotient field of D, denoted by $\overline{\mathsf{D}}$, then $f(y-\theta)$ is τ-sparse in y. By assumption, there exists such a shift, and therefore $z-\theta$ divides Γ. As in (5) we factor $\Gamma(z,y) = g(z)\gamma(z,y)$, where $g \in \mathsf{D}[z]$ and $\gamma(z,y) \in \mathsf{D}[z,y]$ whose content in $\mathsf{D}[z]$ is 1. We claim that $\gamma(z,y) \in \mathsf{D}[y]$. Let us suppose the contrary. Then there exists an element σ in the algebraic closure of $\mathsf{D}(z)$ and transcendental over $\overline{\mathsf{D}}$ such that $\gamma(z,\sigma) = 0$. We thus have that $\Delta_i(z,\sigma) = 0$ for all $2\tau+1 \le i \le 2\deg(f)$. Since the terms σ^i are all distinct, we then get from the Ben-Or/Tiwari algorithm, using $p = \sigma$ and re-interpreting the coefficient field of f to be the algebraic closure of $\mathsf{D}(z)$, that $f(y-z)$ is τ-sparse. Let $d = \deg(f)$ and c_d be the leading coefficient of f. However, the term $c_d\binom{d}{j}z^{d-j}$ is unique in the coefficient of y^j of $f(y-z)$, so $f(y-z)$ is actually d-sparse over $\mathsf{D}[z]$. $\square$

The algorithm using a single projected sequence is as follows:

Algorithm: `UnivariateSparsestShifts`
`<one projection, one sequence>`
Given a polynomial $f(x) \in \mathsf{D}[x]$ and δ an upper bound on $\deg(f)$, this algorithm finds all absolutely sparsest shifts for f with high probability.

(1) [Initialize.] Choose a random.

(2) [Compute $\Delta_1, \ldots, \Delta_{2\delta}$] Let $\alpha_i = f(p^i - z)$, compute $\Delta_1, \ldots, \Delta_{2\delta}$ by the fraction-free Berlekamp/Massey algorithm on polynomial sequence $\alpha_1, \ldots, \alpha_{2\delta+1}$.

(3) [Minimize i so that $\gcd(\Delta_i, \ldots, \Delta_{2\delta})$ is non-trivial.] Whenever $\gcd(\Delta_{2\delta}, \ldots, \Delta_{i-1})$ becomes trivial in $\mathsf{K}[z]$, $g = \gcd(\Delta_{2\delta}, \ldots, \Delta_i)$.

(4) [Solve $g = 0$.] With high probability, all solutions to $g = 0$ are absolutely sparsest shifts for f.

In fact, we conjecture that instead of taking the GCD of all discrepancies up to $2\deg(f)$, we need only look for the GCD of a constant number of discrepancies after the absolutely sparse case $t(s) = \tau$ is reached, that is, we compute $\gcd(\Delta_{2\tau+1}, \ldots, \Delta_{2\tau+\zeta})$ for some constant ζ. For all examples we have tried, $\zeta = 1$ is sufficient.

4.3 Two Projections: Finding the sparsest shifts of a rational polynomial

When $f \in \mathbb{Q}[x]$, we can project the sequence $\{f(y^i + z)\}_{i \ge 1}$ both on a random y and random z from $\mathbb{Z}$, and use the multiplicative structure of the integers to recover the sparsest shift. Thus, finding the sparsest shift will be reduced to running the Berlekamp/Massey algorithm on a small number of integer sequences (conjecturally only one). The existence of a large prime factor in the GCD's of two discrepancies will reveal the sparsest shift. This dramatically improves the efficiency. It also allows us to work completely with a black-box representation for f, requiring only the value of f at points in $\mathbb{Z}$.

Finding factors of a black-box polynomials

We begin by demonstrating a general algorithm for finding a linear factor in one variable of a black-box bivariate polynomial. This will be applied to the discrepancy polynomials

Let $\Phi \in \mathbb{Q}[z,y]$ be a black-box polynomial of degree t in y and degree d in z. Suppose that

$$\Phi(z,y) = (az-b)^e \Psi(z,y),$$

where $a, b \in \mathbb{Z}$ are relatively prime, $e \ge 1$, and $\Psi(z,y) \in \mathbb{Q}[x,z]$ has no non-trivial factor in $\mathbb{Z}[z]$. In this section we give a Monte Carlo algorithm to find a and b with a small constant number of evaluations of Φ.

A number $m \in \mathbb{Q}$ is said to be μ-smooth, for some $\mu > 0$, if all prime factors of both the numerator and denominator of m are less than μ. We say that a polynomial is μ-primitive if it is a μ-smooth number times a primitive, integer polynomial. For any $\Psi = \sum_{ij} \Psi_{ij} y^i z^j \in \mathbb{Z}[z,y]$, let $\|\Psi\| = \max|\Psi_{ij}|$. The *height* of a rational number $\alpha/\beta \in \mathbb{Q}$ (where $\gcd(\alpha,\beta) = 1$) is $\mathcal{H}(\alpha/\beta) = \max\{|\alpha|, |\beta|\}$. Define the *denominator* $\mathrm{denom}(\Phi)$ of $\Phi \in \mathbb{Q}[z,y]$ as the LCM of the denominators of its coefficients. The *content* of Φ is then defined as the usual content of the integer polynomial $\mathrm{denom}(\Psi) \cdot \Psi$. The height of $\Phi \in \mathbb{Q}[z,y]$ is $\mathcal{H}(\Phi) = \max\{|\mathrm{denom}(\Phi)|, \|\mathrm{denom}(\Phi) \cdot \Phi\|\}$. Note that this is the height of Φ in the standard, unshifted, power basis.

To begin with we will insist that Φ is *μ-primitive*, and treat the general case separately below.

Algorithm: `FindLinearFactor`

Input:
- Black box for $\Phi \in \mathbb{Q}[z,y]$;
- Degree bound $C \ge \deg_x \Phi$ and $D \ge \deg_z \Phi$;
- $H > \mathcal{H}(\Phi)$;
- $S >$ height of the sought linear factor;

$\boxed{\Phi \text{ is assumed to be } \mu\text{-primitive, where } \mu = 30CD^2(3+\log(CD)) + 20D\log\|\Phi\|;}$

Output:
- A candidate factor $az - b$ of Φ, where $a, b \in \mathbb{Z}$ are relatively prime;
  or a report "No linear factor in z exists";

(1) $\mathcal{L}_0 = \{0, \ldots, 20DC\}$; $\mathcal{L}_1 = \{0, \ldots, \max\{27, S^4, \mu\}\}$;
(2) Choose random $\gamma_1, \gamma_2 \in \mathcal{L}_0$, $\sigma \in \mathcal{L}_1$;
(3) Let $\bar{q} = \gcd(\mathrm{numer}(\Phi(\gamma_1, \sigma)), \mathrm{numer}(\Phi(\gamma_2, \sigma))$;
(4) Let $q = \bar{q}/m$, with m the largest μ-smooth factor of $\bar{q}$;
(5) If $q = 1$
(6) Then Return "No linear factor in z exists";
(7) Else
(8) Find w and largest $e \ge 1$ such that $q = w^e$
(9) If $w < 2\|\Phi\|^2$
(10) Then Return "Failure";
(11) Else Return $a, b \in \mathbb{Z}$ such that $\gcd(a,b) = 1$, $|a|, |b| \le S$, and $-b/a \equiv \sigma \bmod w$;

THEOREM 3. *For any black-box $\Phi \in \mathbb{Z}[z,y]$ meeting the input criteria,* `FindLinearFactor` *works correctly as stated with probability at least $1/5$ on any invocation.*

Comments

- Obviously the algorithm can be run repeatedly until a factor is found, or the user is satisfied that with sufficiently high probability no linear factor exists.
- The probability of success is undoubtedly much higher than is proven here.
- If w is too small and the algorithm reports "Failure" in step (10), we get a useful modular relation between a and b. Collecting these may allow us to construct a, b without ever getting a really large w;

To prove Theorem 3, we require a number of lemmas. The first simply says that the GCD of an evaluation of two relatively prime integer polynomials is generally smooth.

Lemma 2. *Let $g, h \in \mathbb{Z}[y]$ be relatively prime, primitive polynomials of degree at most d and resultant $r \in \mathbb{Z}$. For a randomly, chosen $\gamma \in \{0, ..., 10d \log r\}$, $\gcd(g(\gamma), h(\gamma))$ is $(10d \log r)$-smooth with probability at least $9/10$.*

Proof. Since g, h are relatively prime, there exist $u, v \in \mathbb{Z}[y]$ such that $u(y)g(y) + v(y)h(y) = r$. Thus, if any prime divides $g(\gamma)$ and $h(\gamma)$, that prime divides r as well. Suppose then that p is a prime dividing r. Then there exists $u_p, v_p, w_p \in \mathbb{Z}[y]$ such that w_p is the GCD of g, h modulo p, and $0 < \deg w_p < d$, and

$$u_p(y)g(y) + v_p(y)h(y) = w_p(y) + pQ_p(y)$$

for some $Q_p \in \mathbb{Z}[y]$. If p divides $g(\gamma)$ and $h(\gamma)$, we have $w_p(\gamma) \equiv 0 \bmod p$. For $p > 10d \log r$ the number of γ such that $w_p(\gamma) \equiv 0 \bmod p$ is less than d. We know r has at most $\log r$ prime factors, so the probability that $w_p(\gamma) \equiv 0 \bmod p$ for *any* prime $p > 10d \log r$ is at most $1/10$. $\square$

We look now at the probability that a number in an arithmetic progression is *rough*, i.e., has a large prime factor. This theorem is an extension of an exercise of Knuth [14]. Let $a, b \in \mathbb{Z}$ be relatively prime. We say that an integer x $(0 \le x < M)$ is $(\sqrt{M}; a, b)$-rough if the largest prime factor of $ax + b$ is greater than $\sqrt{M}$.

Lemma 3. *Let $a, b \in \mathbb{Z}$ be relatively prime and $M \ge \max\{a^3, b^2, 923\}$. The number of $(\sqrt{M}; a, b)$-rough integers x with $0 \le x < M$ is at least $M/4$.*

Proof. We assume $a > 0$. For a prime $p > \sqrt{aM}$, there is a unique x_0 such that $0 \le x_0 < p$ and $ax_0 + b \equiv 0 \bmod p$. Thus, the set of all x such that $ax+b \equiv 0 \mod p$ is $x_0, x_0 + p, \ldots, x_0 + kp$, where $x_0 + kp < M$ and $x_0 + (k+1)p \ge M$. For any p there are at least $M/p - 1$ such numbers. Moreover, it is easily shown that any number can appear in the sequence for at most one prime. Summing all primes p such that $\sqrt{aM} < p < M$, we count

$$\sum_{\sqrt{aM}<p<M} \frac{M}{p} - 1 = M \sum_{\sqrt{aM}<p<M} \frac{1}{p} - \sum_{\sqrt{aM}<p<M} 1$$
$$\ge M\left(\log\log M - \log\log\sqrt{aM} - \frac{1}{2\log^2\sqrt{M}} - \frac{1}{\log^2\sqrt{aM}}\right)$$
$$-\pi(M)$$
$$= M\left(\log\frac{3}{2} - \frac{1}{2\log^2 M} - \frac{1}{\log^2 M^{2/3}}\right) - \frac{M}{-1.5 + \log M}$$

which is $\ge M/4$ for $M \ge 13364$. Here $\pi(m)$ is the number of primes less than or equal to m, and Theorem 2 of [19], shows $\pi(m) < m/(-1.5 + \log(m))$ for $m > 5$. We also use Theorems 5 and 6 from [19] which show that

$$\log\log m + B - \frac{1}{2\log^2 m} < \sum_{p \le m} \frac{1}{p} < \log\log m + B + \frac{1}{\log^2 m}$$

for $m \ge 286$. We verify the theorem for all $M \ge 923$. $\square$

Proof of Theorem 3. First, consider a primitive polynomial $\Psi \in \mathbb{Z}[z, y]$ of degree $c \le C$ in y and $d \le D$ in z, that has no non-trivial factor in z only. For any $\gamma \in \mathcal{L}_0$, note that $\Psi(\gamma, z)$ is also primitive of degree d.

We now show that for randomly chosen $\gamma_1, \gamma_2 \in \mathcal{L}_0$, $\gcd(\Psi(\gamma_1, z), \Psi(\gamma_2, z)) = 1$ with probability at least $9/10$. Let y_1, y_2 be two new indeterminates and consider the resultant $R(y_1, y_2)$ of $\Psi(y_1, z)$ and $\Psi(y_2, z)$ as polynomials in $\mathbb{Q}(y_1, y_2)[z]$. R is a primitive polynomial of degree $2dc \le 2DC$. $R(\gamma_1, \gamma_2) \ne 0$ with probability at least $9/10$ by the Schwartz-Zippel Lemma, and hence $\Psi(\gamma_1, z)$ and $\Psi(\gamma_2, z))$ are relatively prime with probability $\ge 9/10$.

Assume now that $\Psi(\gamma_1, z)$ and $\Psi(\gamma_2, z)$ are in fact relatively prime. It is easily derived that $\|\Psi(\gamma_i, z)\| \le (20DC)^C \cdot \|\Psi\|$. Thus, the resultant r of $\Psi(\gamma_1, z)$ and $\Psi(\gamma_2, z)$ is at most $(2D)^{2D} \cdot (20DC)^{2DC} \cdot \|\Psi\|^{2D}$. Simplifying this, we note that $\log r \le 3DC(3 + \log(DC)) + 2D \log \|\Phi\|$. By Lemma 2, for a randomly chosen $\sigma \in \mathcal{L}_1$, $\gcd(\Psi(\gamma_1, \sigma), \Psi(\gamma_2, \sigma))$ is μ-smooth with probability at least $9/10$.

Now consider the full case when $\Phi(z, y) = m \cdot (az + b)^e \cdot \Psi(z, y)$, where $m \in \mathbb{Q}$ is μ-smooth, $a, b \in \mathbb{Z}$ are relatively prime, and Ψ is primitive and has no factor purely in $\mathbb{Z}[z]$. Then $\bar{q} = m \cdot (a\sigma - b) \cdot \gcd(\Psi(a_1, \sigma), \Psi(a_2, \sigma)$. From above we see $\gcd(\Psi(a_1, \sigma), \Psi(a_2, \sigma))$ is μ-smooth with probability at least $81/100$. Thus w is equal to the factor $a\sigma - b$ which is hopefully not μ-smooth. Both $|\sigma|$ and $|b|$ are less than $\|\Phi\|$. By Lemma 3, $(a\sigma + b)$ has a prime factor of size greater than $2 \cdot |a| \cdot |b| \ge \|\Phi\|^2$, with probability at least $1/4$, and in this case we recover a, b as described in step (11). To conclude, for any input, on any invocation the algorithm succeeds with probability at least $(81/100) \cdot (1/4) \ge 1/5$. $\square$

Approximating the denominator and content

To complete the general algorithm, we must identify the μ-primitive part of a black-box polynomial. The following algorithm does this with 2 evaluations of the black-box.

Algorithm: `DenominatorAndContent`

Input:
- ▸ Black box for $f \in \mathbb{Q}[y]$;
- ▸ a bound D for the degree of f;
- ▸ a bound H for the height of f;

Output:
- ▸ a candidate $\omega \in \mathbb{Q}$ such that content of ωf is μ-smooth, where $\mu = 4D(D+1) + 4D \log H$;

(1) Let $\mathcal{L}_0 = \{0, 2D\}$;
(2) Choose a random $\alpha_0 \in \mathcal{L}_0$ and compute $\nu_0 = f(\alpha_0) \in \mathbb{Q}$; If $\nu_0 = 0$ the goto (2);
(3) Let $\mathcal{L}_1 = \{0, \ldots, \mu\}$;
(4) Choose random $\alpha_1 \in \mathcal{L}_1$; compute $\nu_1 = f(\alpha_1)$;
(5) Let $\tilde{\delta} = \mathrm{lcm}(\mathrm{denom}(\nu_0), \mathrm{denom}(\nu_1))$;

(6) Let $\tilde{\kappa} = \gcd(\tilde{\delta}\nu_0, \tilde{\delta}\nu_1)$;

(7) Return $\omega = \tilde{\delta}/\tilde{\kappa}$

THEOREM 4. *With probability at least 1/2 the output ω of* `DenominatorAndContent`(f, μ) *is such that the content of ωf is μ-smooth.*

PROOF. In Step (2) we simply find a small non-zero evaluation point for f. We expect that at at most 2 evaluations of f are required.

In Step (5) we approximate the denominator δ of f. Suppose $\bar{f} = \delta f$. For any prime $p \mid \delta$ we know that $\bar{f} \not\equiv 0 \bmod p$ (since δ is relatively prime to the content κ of f). Let $d = \deg f$. For $p > \mu$, the number of $\alpha_1 \in \mathcal{L}_1$ for which $\bar{f}(\alpha_1) \equiv 0 \bmod p$ is at most d. Since d has at most $\log|d| < \log \mathcal{H}(f) < \log H$ prime factors, with probability at least 3/4 we choose an $\alpha_1 \in \mathcal{L}_1$ such that $\bar{f}(\alpha_1) \not\equiv 0 \bmod p$ for *all* primes $p \mid d$ and $p > \mu$. In this case the denominator $\tilde{\delta}$ of $f(\alpha_1)$ is the denominator of f times a μ-smooth number.

In Step (6) we approximate the content κ of δf. Suppose that $\tilde{\delta} f$ has content $\tilde{\kappa}$. We know $\tilde{\kappa}$ is κ times some μ-smooth number, and $\tilde{\delta} f = \tilde{\delta}\tilde{\kappa} f_0$, where f_0 is primitive. Clearly $\kappa \mid \nu_1$. For any prime $p > \mu$, the number of $\alpha_1 \in \mathcal{L}_1$ for which $f_0(\alpha_1) \equiv 0 \bmod p$ is at most d. Since numer(ν_0) has at most $\log|d^{d+1}\mathcal{H}(f)^d| < \log|D^{D+1}H^D| < \mu/4$ prime factors greater than μ, with probability $\geq 3/4$ we choose $\alpha_1 \in \mathcal{L}_1$ with $f_0(\alpha_1) \not\equiv 0 \bmod p$ for *all* primes $p \mid \nu_0$, $p > \mu$. $\square$

Once we have the $\omega =$ `DenominatorAndContent`(f), it is easy to construct a black box for the μ-primitive part by multiplying the result of an evaluation of f by ω.

Finding sparsest shifts of integer polynomials

Suppose we have a black box for a rational polynomial $f \in \mathbb{Q}[x]$, and a bound $D \geq d = \deg f$. We now describe the complete algorithm for finding a sparsest shift of f.

We first approximate the content to within a μ-smooth multiple using `DenominatorAndContent`. We then build a new black-box for the μ-primitive part of f (by dividing out the content and denominator) and so assume from now on that f is μ-primitive.

As discussed earlier, when we run the Berlekamp/Massey algorithm on the sequence of polynomials $\{f(y^i + z)\}_{i\geq 1}$, we are really just constructing the discrepancy polynomials $\Delta_i(z, y)$ for $i = 1, 2, \ldots, t$. When we choose a random p and s and run Berlekamp/Massey on $\{f(p^i + s)\}_{i\geq 1}$ we are evaluating the discrepancy polynomials at (s, p). I.e., the Berlekamp/Massey algorithm gives us a black box for the discrepancy polynomials. `FindLinearFactor` will be just what we need to find the smallest t such that $\Delta_{2t-1}(z, y)$ has a factor in z alone (at least in the case when $t \leq (d+1)/2$).

Examining the discrepancy polynomials more closely, for $1 \leq i \leq t$ let $\alpha_i(z, y) = f(y^i - z)$ and

$$\bar{A}_i = \begin{pmatrix} \alpha_1 & \alpha_2 & \cdots & \alpha_i \\ \alpha_2 & \alpha_3 & \iddots & \alpha_{i+1} \\ \vdots & \iddots & \iddots & \vdots \\ \alpha_i & \cdots & \cdots & \alpha_{2i-1} \end{pmatrix} \in \mathbb{Q}[z, y]^{i\times i}, \quad \nabla_i = \det \bar{A}_i$$

The $(2i-1)$st discrepancy of the the sequence $\{\alpha_i\}_{i\geq 1}$ is $\Delta_{2i-1} = \nabla_i/\nabla_{i-1}$ for $i \geq 1$ (taking $\nabla_0 = 1$). The sparsest shift of f occurs when there exists an $s \in \mathbb{Q}$ (or perhaps an algebraic extension of $\mathbb{Q}$) such that $\Delta_{2t-1}(y, s) = 0$, i.e., when Δ_{2t-1} has a factor in z alone.

When $t \leq (d+1)/2$, the sparsest shift is rational and unique, so we can apply the algorithm `FindLinearFactor` to the numerators in the Berlekamp/Massey algorithm to find the sparsest shift.

THEOREM 5. *Given a black-box for a μ-primitive polynomial $f \in \mathbb{Q}[x]$ of degree d, which we assume has a t-sparse shift $s \in \mathbb{Q}$, where $t \leq (d+1)/2$, we can find $s \in \mathbb{Q}$ with an expected $10t$ evaluations of the black-box.*

PROOF. It is straightforward to show the bounds

$$\|\nabla_i\| \leq i^i \cdot 2^{id}(1+d)^i(1+di)^i \cdot \|f\|^d,$$
$$|b| \leq 2^t \cdot t^{2t} \cdot \|f\|^t, \quad |a| \leq 2^t \cdot t^{2t} \cdot \|f\|^t \cdot (2d)^{dt}$$

Now use the algorithm `FindLinearFactor` on each discrepancy in turn. By Theorem 3, at the $(2t+1)$st discrepancy we will find a, b such that $az - b$ divides $\Delta_t(z, y)$ with probability 1/5 on any invocation. The sparsest shift is then $a/-b$. By running the algorithm repeatedly, we expect to find t and s with $5t$ invocations of `FindLinearFactor`, i.e., using $10t$ sequences. $\square$

All the notes following Theorem 3 apply here. In fact we heuristically expect that only one invocation of the algorithm will be needed to achieve success.

Once we find a sparsest shift, the polynomial can be recovered by completing the Ben-Or/Tiwari algorithm steps with the evaluations and generator already computed. Therefore, we regard this algorithm as an improved sparse interpolation algorithm: it *discovers and interpolates* with respect to a possible sparsest basis during the interpolation procedure.

The "one projection, one sequence" algorithm for univariate polynomials of Subsection 4.2 holds even more promise when a second "shift" projection is used. That is, we proceed as in `FindLinearFactor`, but instead of taking the GCD of the discrepancies of two different sequences, we take the GCD's of the $(i-1)$st and ith discrepancies. As noted in Subsection 4.2, we conjecture this reveals the linear factor symbolically, and if this is indeed the case, we might hope that only *one* randomly shifted integer sequence is needed.

When the sparsest shift has $t > (d-1)/2$, the factor of the discrepancy polynomial containing the sparsest shifts as roots may no longer be linear. In particular, in the algorithm `FindLinearFactor`, $\Phi(z, y) = g(z)\Psi(z, y)$ for a nonlinear $g \in \mathbb{Z}[z]$, and when we compute $\gcd(\Phi(\gamma_1, \sigma), \Phi(\gamma_2, \sigma))$, we see a factor of $g(\sigma)$. While it is conjectured that $g(\gamma)$ will have large prime factors with reasonable probability, this appears difficult to prove. Heuristically however, we can interpolate a non-linear g much as we did a linear one. We then factor g to obtain the sparsest shifts for f.

5. FUTURE DIRECTIONS

Our methods are generalizable in several ways. First, the multivariate case can again be handled via randomization and by computing several Berlekamp/Massey discrepancy sequences. For that, we need a lemma that n distinct linear combinations of the algebraic equations in the unknown shifts do not enlarge the solution variety. Multivariate shifts enable us to compute simultaneously sparsest shifts of a set of polynomials. For example, the sparsest shift of

$$f_1(x) + yf_2(x) + y^2 f_3(x) + \cdots + y^{m-1} f_m(x) \in \mathsf{D}[x, y]$$

within $S = \mathsf{K} \times \{0\}$, where $f_i \in \mathsf{D}[x]$ and y is a fresh variable, minimizes the sum of the number of terms in the shifted polynomials $f_i(x + s)$. For a random value of $y \in \overline{\mathsf{K}}$ the algorithm minimizes the combined number of terms. A full discussion will be provided in a future paper.

We may also consider different bases, such as Chebyshev and Pochhammer bases. The determinants considered for early termination and thus for symbolic evaluation are described in [17]. Finally, one may consider so-called sparsifying linear transforms. In the univariate case one searches for domain elements a_1, a_2 such that the substitution $x = a_1 y + a_2$ yields a sparse polynomial. In the multivariate case one uses a linear transform $x = Ay$, where x and y are vectors of variables and A is an invertible matrix with scalar entries [7]. Again the problem reduces to finding roots of certain algebraic equations whose efficient solution we plan to study.

Acknowledgements: We thank George Labahn and three anonymous referees for their comments.

Note: many of the authors' publications cited below are accessible through links in their Internet homepages.

6. REFERENCES

[1] Ben-Or, M., and Tiwari, P. A deterministic algorithm for sparse multivariate polynomial interpolation. In *Proc. Twentieth Annual ACM Symp. Theory Comput.* (New York, N.Y., 1988), ACM Press, pp. 301–309.

[2] Brown, W. S., and Traub, J. F. On Euclid's algorithm and the theory of subresultants. *J. ACM 18* (1971), 505–514.

[3] Cabay, S., and Kossowski, P. Power series remainder sequences and pade fractions over an integral domain. *J. Symbolic Comput. 10* (1990), 139–163.

[4] DeMillo, R. A., and Lipton, R. J. A probabilistic remark on algebraic program testing. *Information Process. Letters 7*, 4 (1978), 193–195.

[5] Díaz, A., and Kaltofen, E. FoxBox a system for manipulating symbolic objects in black box representation. In *Proc. 1998 Internat. Symp. Symbolic Algebraic Comput. (ISSAC'98)* (New York, N. Y., 1998), O. Gloor, Ed., ACM Press, pp. 30–37.

[6] Dornstetter, J. L. On the equivalence between Berlekamp's and Euclid's algorithms. *IEEE Trans. Inf. Theory* IT-*33*, 3 (1987), 428–431.

[7] Grigoriev, D. Y., and Karpinski, M. A zero-test and an interpolation algorithm for the shifted sparse polynomials. In *Proc. AAECC-10* (Heidelberg, Germany, 1993), vol. 673 of *Lect. Notes Comput. Sci.*, Springer Verlag, pp. 162–169.

[8] Grigoriev, D. Y., Karpinski, M., and Singer, M. F. Fast parallel algorithms for sparse multivariate polynomial interpolation over finite fields. *SIAM J. Comput. 19*, 6 (1990), 1059–1063.

[9] Grigoriev, D. Y., Karpinski, M., and Singer, M. F. Computational complexity of sparse rational function interpolation. *SIAM J. Comput. 23* (1994), 1–11.

[10] Grigoriev, D. Y., and Lakshman, Y. N. Algorithms for computing sparse shifts for multivariate polynomials. *Applic. Algebra Engin. Commun. Comput. 11*, 1 (2000), 43–67.

[11] Grigoriev, D. Y., and Lakshman Y. N. Algorithms for computing sparse shifts for multivariate polynomials. In *Proc. 1995 Internat. Symp. Symbolic Algebraic Comput. ISSAC'95* (New York, N. Y., 1995), A. H. M. Levelt, Ed., ACM Press, pp. 96–103.

[12] Kaltofen, E., Lee, W.-s., and Lobo, A. A. Early termination in Ben-Or/Tiwari sparse interpolation and a hybrid of Zippel's algorithm. In *Proc. 2000 Internat. Symp. Symbolic Algebraic Comput. (ISSAC'00)* (New York, N. Y., 2000), C. Traverso, Ed., ACM Press, pp. 192–201.

[13] Kaltofen, E., and Trager, B. Computing with polynomials given by black boxes for their evaluations: Greatest common divisors, factorization, separation of numerators and denominators. *J. Symbolic Comput. 9*, 3 (1990), 301–320.

[14] Knuth, D. E. *Mathematics for the Analysis of Algorithms*, 3rd ed. Birkhäuser, 1983.

[15] Knuth, D. E. *Seminumerical Algorithms*, Third ed., vol. 2 of *The Art of Computer Programming*. Addison Wesley, Reading, Massachusetts, USA, 1997.

[16] Lakshman Y. N., and Saunders, B. D. Sparse shifts for univariate polynomials. *Applic. Algebra Engin. Commun. Comput. 7*, 5 (1996), 351–364.

[17] Lee, W.-s. *Early termination strategies in sparse interpolation algorithms*. PhD thesis, North Carolina State Univ., Raleigh, North Carolina, Dec. 2001. 107 pages.

[18] Massey, J. L. Shift-register synthesis and BCH decoding. *IEEE Trans. Inf. Theory* IT-*15* (1969), 122–127.

[19] Rosser, J. B., and Schoenfeld, L. Approximate formulas for some functions of prime numbers. *Ill. J. Math. 6* (1962), 64–94.

[20] Schwartz, J. T. Fast probabilistic algorithms for verification of polynomial identities. *J. ACM 27* (1980), 701–717.

[21] Zippel, R. Probabilistic algorithms for sparse polynomials. In *Proc. EUROSAM '79* (Heidelberg, Germany, 1979), vol. 72 of *Lect. Notes Comput. Sci.*, Springer Verlag, pp. 216–226.