Computing Gröbner Bases for Vanishing Ideals of Finite Sets of Points

Jeffrey B. Farr[1] and Shuhong Gao[2]

[1] Simon Fraser University, Burnaby, B.C. V5A 1S6, Canada
jfarr@cecm.sfu.ca
http://www.cecm.sfu.ca/~jfarr/
[2] Clemson University, Clemson, SC 29634-0975, USA
sgao@ces.clemson.edu
http://www.math.clemson.edu/~sgao/

Abstract. We present an algorithm to compute a Gröbner basis for the vanishing ideal of a finite set of points in an affine space. For distinct points the algorithm is a generalization of univariate Newton interpolation. Computational evidence suggests that our method compares favorably with previous algorithms when the number of variables is small relative to the number of points. We also present a preprocessing technique that significantly enhances the performance of all the algorithms considered. For points with multiplicities, we adapt our algorithm to compute the vanishing ideal via Taylor expansions.

1 Introduction

Suppose $P_1, \ldots, P_n$ are distinct points in the m-dimensional vector space over a field $\mathbb{F}$. The set of polynomials in $\mathbb{F}[x_1, \ldots, x_m]$ that evaluate to zero at each P_i form a zero-dimensional ideal called the vanishing ideal of the points. The problem that we consider is how to compute the reduced Gröbner basis for the vanishing ideal of any finite set of points under any given monomial order. This problem arises in several applications; for example, see [16] for statistics, [13] for biology, and [18, 11, 12, 6] for coding theory.

A polynomial time algorithm for this problem was first given by Buchberger and Möller (1982) [2], and significantly improved by Marinari, Möller and Mora (1993) [14], and Abbott, Bigatti, Kreuzer and Robbiano (2000) [1]. These algorithms perform Gauss elimination on a generalized Vandermonde matrix and have a polynomial time complexity in the number of points and in the number of variables. O'Keeffe and Fitzpatrick (2002) [9] studied this problem from a coding theory point of view. They present an algorithm that is exponential in the number of variables, and the Gröbner basis which they compute is not reduced.

We present here a variation of the O'Keeffe-Fitzpatrick method. Our approach does, though, compute the *reduced* Gröbner basis and is essentially a generalization of Newton interpolation for univariate polynomials. Even though the time complexity of our algorithm is still exponential in the number of variables, its practical performance improves upon both the O'Keeffe-Fitzpatrick algorithm

M. Fossorier et al. (Eds.): AAECC 2006, LNCS 3857, pp. 118–127, 2006.

and the linear algebra approach if the number of variables is relatively small compared to the number of points.

The rest of the paper is organized as follows. In Section 2, we present our algorithm for distinct points. We also show how multivariate interpolation is a special case of computing vanishing ideals. Section 3 presents experimental time comparisons along with a sorting heuristic for the points. Finally, Section 4 shows how to handle the case for points with multiplicity. Some of the material presented here is surveyed in our recent paper [7], which gives a broader view on how Gröbner basis theory can be used in coding theory.

2 Distinct Points

Throughout this section we fix an arbitrary monomial order (also called term order by some authors) on the polynomial ring $\mathbb{F}[x_1, \ldots, x_m]$. Then each polynomial $f \in \mathbb{F}[x_1, \ldots, x_m]$ has a leading term, denoted by $\mathrm{LT}(f)$, and each ideal has a unique reduced Gröbner basis. For any subset $G \subset \mathbb{F}[x_1, \ldots, x_m]$, we define

$$\mathcal{B}(G) = \{\mathbf{x}^\alpha : \alpha \in \mathbb{N}^m \text{ and } \mathbf{x}^\alpha \text{ is not divisible by } \mathrm{LT}(g) \text{ for any } g \in G\},$$

where $\mathbb{N} = \{0, 1, 2, \ldots\}$ and $\mathbf{x}^\alpha = x_1^{\alpha_1} \cdots x_m^{\alpha_m}$ for $\alpha = (\alpha_1, \ldots, \alpha_m)$. A basic theorem in Gröbner basis theory tells us that, for each ideal $\mathbf{I} \subset \mathbb{F}[x_1, \ldots, x_m]$, the monomials in $\mathcal{B}(\mathbf{I})$ form a basis for the quotient ring $\mathbb{F}[x_1, \ldots, x_m]/\mathbf{I}$ as a vector space over $\mathbb{F}$ (see Section 3 in [4]). This basis is called a monomial basis, or a standard basis, for $\mathbf{I}$ under the given monomial order. For $V \subseteq \mathbb{F}^m$, let $\mathbf{I}(V)$ denote the vanishing ideal of V; that is,

$$\mathbf{I}(V) = \{f \in \mathbb{F}[x_1, \ldots, x_m] : f(P) = 0, \text{ for all } P \in V\}.$$

If $V = \{P_1, \ldots, P_n\}$, $\mathbf{I}(V)$ is also written as $\mathbf{I}(P_1, \ldots, P_n)$.

Lemma 1. *For $g_1, \ldots, g_s \in \mathbf{I} = \mathbf{I}(P_1, \ldots, P_n)$, $\{g_1, \ldots, g_s\}$ is a Gröbner basis for $\mathbf{I}$ if and only if $|\mathcal{B}(g_1, \ldots, g_s)| = n$.*

Proof. By definition $g_1, \ldots, g_s \in \mathbf{I}$ form a Gröbner basis for $\mathbf{I}$ if and only if $\mathcal{B}(g_1, \ldots, g_s) = \mathcal{B}(\mathbf{I})$. One can show by interpolation that $\dim \mathbb{F}[x_1, \ldots, x_m]/\mathbf{I} = n$. But the monomials in $\mathcal{B}(\mathbf{I})$ form a basis for the quotient ring $\mathbb{F}[x_1, \ldots, x_m]/\mathbf{I}$ viewed as a vector space over $\mathbb{F}$. The lemma follows immediately. □

Lemma 2. *Suppose $G = \{g_1, \ldots, g_s\}$ is a Gröbner basis for $\mathbf{I}(V)$, for a finite set $V \subset \mathbb{F}^m$. For a point $P = (a_1, \ldots, a_m) \notin V$, let g_i denote the polynomial in G with smallest leading term such that $g_i(P) \neq 0$, and define*

$$\tilde{g}_j := g_j - \frac{g_j(P)}{g_i(P)} \cdot g_i, \qquad j \neq i, \text{ and}$$
$$g_{ik} := (x_k - a_k) \cdot g_i, \qquad 1 \leq k \leq m.$$

Then

$$\widetilde{G} = \{\tilde{g}_1, \ldots, \tilde{g}_{i-1}, \tilde{g}_{i+1}, \ldots, \tilde{g}_s, g_{i1}, \ldots, g_{im}\}$$

is a Gröbner basis for $\mathbf{I}(V \cup \{P\})$.

Proof. At least one polynomial in G must be nonzero when evaluated at P since $P \notin V$; hence, a suitable g_i exists.

Certainly, $\widetilde{G} \subseteq \mathbf{I}(V \cup \{P\})$ as the new and modified polynomials evaluate to zero at all points in $V \cup \{P\}$. Denote $\mathrm{LT}(g_i)$ by $\mathbf{x}^\alpha$. We claim that

$$\mathcal{B}(\widetilde{G}) = \mathcal{B}(G) \cup \{\mathbf{x}^\alpha\}. \tag{1}$$

By the choice of i, $\mathrm{LT}(\tilde{g}_j) = \mathrm{LT}(g_j)$, for all $j \neq i$. Also, since g_i was replaced in $\widetilde{G}$ by $g_{i1}, g_{i2}, \ldots, g_{im}$, whose leading terms are $\mathbf{x}^\alpha x_1$, $\mathbf{x}^\alpha x_2$, $\ldots$, $\mathbf{x}^\alpha x_m$, we know that $\mathbf{x}^\alpha$ is the only monomial not in $\mathcal{B}(\mathbf{I}(V))$ that is in $\mathcal{B}(\mathbf{I}(V \cup \{P\}))$. Thus, (1) is satisfied, and $|\mathcal{B}(\widetilde{G})| = |\mathcal{B}(G)| + 1$. Since G is a Gröbner basis for $\mathbf{I}(V)$, we have $|\mathcal{B}(G)| = |V|$, and the conclusion follows from Lemma 1. □

Notice that some of the $\mathrm{LT}(g_{ik})$ may be divisible by the leading term of another polynomial in $\widetilde{G}$. In such a case, g_{ik} may be omitted from $\widetilde{G}$ and $\widetilde{G} \setminus \{g_{ik}\}$ is still a Gröbner basis. In fact, we can check for this property before computing g_{ik} so that we save ourselves needless computation. In so doing, we also guarantee that the resulting $\widetilde{G}$ is a minimal Gröbner basis for $\mathbf{I}(V \cup \{P\})$.

To get a reduced Gröbner basis, we still need to reduce the new polynomials g_{ik}. We order the variables in increasing order, say $x_1 < x_2 < \ldots < x_m$, and reduce the polynomials from g_{i1} up to g_{im} . Thus, in Algorithm 1 the polynomials in G are always stored so that the leading terms of its polynomials are in increasing order. This will make sure that each g_{ik} need only be reduced once. Also, $\mathrm{Reduce}(h, G)$ is the unique remainder of h when reduced by polynomials in G.

Algorithm 1

```
1   Input: P_1, P_2, ..., P_n ∈ 𝔽^m, and a monomial order
      We assume that the variables are labelled so that x_1 < ... < x_m.
2   Output: G, the reduced Gröbner basis for I(P_1, ..., P_n), in increasing order.
3
4   G := {1};          /* the ith polynomial in G is denoted g_i */
5   FOR k from 1 to n DO
6     Find the smallest i so that g_i(P_k) ≠ 0;
7     FOR j from i + 1 to |G| DO    g_j := g_j − (g_j(P_k)/g_i(P_k)) · g_i;    END FOR;
8     G := G \ {g_i};
9     FOR j from 1 to m DO
10      IF x_j · LT(g_i) not divisible by any leading term of G THEN
11        Compute h := Reduce((x_j − a_j) · g_i, G);
12        Insert h (in order) into G;
13      END IF;
14    END FOR;
15  END FOR;
16
17  RETURN G.
```

Lemma 2 and the subsequent remarks imply the following theorem.

Theorem 1. *For a finite set $V \subseteq \mathbb{F}^m$ and a given monomial order, Algorithm 1 returns the reduced Gröbner basis for $\mathbf{I}(V)$.*

A related question is multivariate interpolation, and it can easily be solved using Algorithm 1. The interpolation problem is: given the points $P_1, \ldots, P_n \in \mathbb{F}^m$ and any values $r_1, \ldots, r_n \in \mathbb{F}$, find a "smallest" f so that

$$f(P_i) = r_i, \quad 1 \leq i \leq n. \tag{2}$$

Multivariate polynomial interpolation has been extensively studied in the past 30 years (see the survey [10]). The property of being "smallest" is addressed by introducing an appropriate monomial order on $\mathbb{F}[x_1, \ldots, x_m]$. Then there is a unique polynomial $f \in Span_{\mathbb{F}}(\mathcal{B})$ satisfying (2), and it will be the smallest such polynomial under the given monomial order. One strategy for finding this polynomial f is given in [17] that uses separator polynomials. The following theorem, which follows directly from Lemma 1, tells us that any algorithm for computing vanishing ideal can be easily used to solve the interpolation problem.

Theorem 2. *Let G be the reduced Gröbner basis for $\mathbf{I} = \mathbf{I}(P_1, \ldots, P_n)$ under the fixed monomial order $<$ on $\mathbb{F}[x_1, \ldots, x_m]$, and let $\mathcal{B} = \mathcal{B}(\mathbf{I})$ be the corresponding monomial basis. Introduce a new variable z and an elimination order for z that extends $<$. Then the reduced Gröbner basis for $\mathbf{I}((P_1, r_1), \ldots, (P_n, r_n))$, is of the form $G \cup \{z - f\}$, where f is the unique polynomial in $Span_{\mathbb{F}}(\mathcal{B})$ satisfying (2).*

One can easily generalize Theorem 2 to the case when there are more than one z-coordinate. Also, in the case when $m = 1$ Theorem 2 appears in the literature as the "Shape Lemma" (see Exercise 16 in Section 2.4 in [5]).

This same strategy can be modified for multivariate rational function interpolation. In this case the Gröbner basis computation is performed for a submodule of rank two rather than for an ideal. The major hurdle that has to be overcome before applying a modified Algorithm 1 is the selection of an appropriate term order. We refer the reader to [6] for more details.

3 Time Complexity

3.1 The Cost of Reduction

All the steps in Algorithm 1 are straightforward to analyze except the reduction step in line 11. We use standard Buchberger reduction (i.e., repeated division). This reduction has a worst-case time complexity that may be exponential in the number m of variables. It is possible to make this step polynomial time by using the border-basis reduction technique introduced in [8]. The border Gröbner basis computed, however, is quite large in general. For example, the reduced Gröbner basis for the vanishing ideal of a random set of 500 points from $\mathbb{F}_2^{10}$ under *lex* order usually contains around 100 polynomials, while the border basis typically contains over 2000. So the running time and memory usage of Algorithm 1 using border-basis reduction are much worse than the original. For these reasons we ignore the theoretical "improvements" that border-basis reduction provides.

3.2 Running Time Comparison

As we mentioned earlier, the methods in [1, 2, 14] are based on Gauss elimination and have a polynomial time complexity $O(n^3m^2)$. We compare our Algorithm 1 particularly with the algorithm (MMM) of Marinari, Möller and Mora [14]. Although the algorithm of [1] has an excellent implementation in the computer algebra system CoCoA, it is not appropriate for us to compare this compiled code with our interpreted code (see specs below).

The Gröbner basis found via the algorithm (O'K-F) of O'Keeffe and Fitzpatrick [9] is minimal in the sense that the number of polynomials in the basis is the smallest, but the length of the polynomials computed may grow exponentially in the number m of variables. So, most of the computing time in O'K-F is taken up with dealing with large polynomials, and most of the time in Algorithm 1 involves the reduction step, *i.e.*, computing $\text{Reduce}(g_{ij}, G)$.

Table 1 presents running times for the algorithms for various point sets. We have chosen three high-dimensional vector spaces and three low-dimensional vector spaces to highlight the significance of the dimension. The times are the average running times in seconds for randomly chosen point sets from the specified vector space (based on 100 experiments for $n = 250$, 10 experiments for

Table 1. Average running times for 250, 500 and 1000 random points from $\mathbb{F}_q^m$

q	m	MMM		Algorithm 1		O'K-F	
		grlex	*lex*	*grlex*	*lex*	*grlex*	*lex*
11	3	2.440	1.404	1.182	0.356	1.006	0.705
31	3	2.762	1.481	1.418	0.312	1.213	0.395
101	3	2.867	1.423	1.512	0.220	1.289	0.218
2	10	2.542	1.321	2.201	1.142	3.015	9.832
2	12	4.954	1.894	4.207	2.381	4.037	14.43
2	15	7.568	2.875	7.592	7.565	8.066	22.54

q	m	MMM		Algorithm 1		O'K-F	
		grlex	*lex*	*grlex*	*lex*	*grlex*	*lex*
11	3	14.72	10.50	5.726	1.789	5.126	9.547
31	3	19.33	10.53	9.357	1.688	8.236	3.043
101	3	20.08	10.66	11.16	1.141	9.467	1.371
2	10	12.10	7.773	8.849	4.314	18.81	332
2	12	24.43	11.62	21.68	13.49	31.90	367
2	15	64.69	16.79	63.74	33.42	53.38	522

q	m	MMM		Algorithm 1		O'K-F	
		grlex	*lex*	*grlex*	*lex*	*grlex*	*lex*
31	3	143	80.98	71.04	10.80	68.05	39.59
101	3	149	86.05	107	6.852	92.92	11.52
2	12	173	75.62	146	74.36	307	10321
2	15	315	117	312	218	471	16609

$n = 500, 1000$). The algorithms were implemented in Magma version 2.11 and run on an Apple Power Macintosh G5 computer, 2.5 GHz CPU, 2 GB RAM.

The timings indicate that Algorithm 1 is faster than MMM—by a factor of two for *grlex*, a factor approaching ten for *lex*—if the dimension m (the number of variables) is small relative to the number n of points. In comparison to O'K-F, Algorithm 1 takes roughly the same time for small m and n, but O'K-F slows down somewhat when n increases and slows down quickly when m increases.

3.3 Sorting the Points

A clever ordering of the points can improve the running time of Algorithm 1, O'K-F and, somewhat surprisingly, MMM. The significance of improvement depends on both the chosen monomial order and the geometric structure of the points.

The details of this ordering are quite simple. If $x_1 < \ldots < x_m$, then group the points first according to the x_1-coordinate; these groups are ordered by the number of elements, largest to smallest (specifically, nonincreasingly). Within each

Table 2. Running times for 250, 500 and 1000 random points (sorted) from $\mathbb{F}_q^m$

q	m	MMM		Algorithm 1		O'K-F	
		grlex	*lex*	*grlex*	*lex*	*grlex*	*lex*
11	3	2.004	0.714	1.066	0.277	0.890	0.286
31	3	2.735	0.956	1.418	0.273	1.189	0.264
101	3	2.867	1.076	1.516	0.206	1.277	0.193
2	10	1.557	0.746	1.363	0.575	1.534	0.646
2	12	3.342	1.131	3.257	1.091	2.544	1.158
2	15	5.061	1.802	5.360	3.689	5.283	2.118

q	m	MMM		Algorithm 1		O'K-F	
		grlex	*lex*	*grlex*	*lex*	*grlex*	*lex*
11	3	10.31	3.558	4.978	1.277	4.426	1.421
31	3	19.017	5.693	9.298	1.418	8.184	1.406
101	3	20.11	6.996	11.158	1.049	9.394	0.962
2	10	5.461	2.498	4.534	1.681	6.308	2.426
2	12	13.70	5.250	13.34	6.022	16.19	7.850
2	15	46.06	8.058	53.42	11.16	34.39	12.39

q	m	MMM		Algorithm 1		O'K-F	
		grlex	*lex*	*grlex*	*lex*	*grlex*	*lex*
31	3	139	34.41	70.66	8.754	70.35	9.041
101	3	149	47.38	107	6.037	91.97	5.471
2	12	90.44	22.45	88.74	23.63	121	45.02
2	15	169	40.91	182	71.13	217	114

of the groups, repeat the process, but according to the x_2-coordinate. Continue for $x_3, \ldots, x_m$.

A comparison of Table 2 with Table 1 indicates the sizable impact of reordering. Essentially, this sorting decreases the amount of reduction that Algorithm 1 needs to do. Further, although O'K-F does not involve reduction, it is also helped since the Gröbner basis remains comparatively small. In MMM, reordering the points corresponds to a favorable reordering of the columns in an implicit matrix to which Gauss elimination is applied.

Gröbner bases under *lex* order experience the greatest benefit since they typically require the most reduction and are prone to exponential growth without reduction. Gröbner bases under *grlex* order with points from a low-dimensional vector space experience little or no speedup.

4 Points with Multiplicities

We now consider the case in which some points in the vanishing set have multiplicity. A general notion of *algebraic multiplicity* is described in [14] and [15]. We will adopt a special form used by Cerlienco and Mureddu [3] that is general enough for most applications.

Let $\mathbf{v} = (v_1, \ldots, v_m) \in \mathbb{Z}^m$. We define a differential operator $D^{\mathbf{v}}$ by

$$D^{\mathbf{v}} = \frac{1}{v_1! \cdots v_m!} \cdot \frac{\partial^{v_1+\ldots+v_m}}{\partial {x_1}^{v_1} \cdots \partial {x_m}^{v_m}}.$$

We note that $D^{\mathbf{v}}$ is a linear map on functions with the m variables $x_1, \ldots, x_m$. Let $P \in \mathbb{F}^m$ and f be any function on $x_1, \ldots, x_m$. We employ the notation

$$[D^{\mathbf{v}} f](P) = D^{\mathbf{v}} f|_{\mathbf{x}=P}, \tag{3}$$

where $P = (a_1, \ldots, a_m) \in \mathbb{F}^m$. Then, under reasonable conditions (analytic or algebraic) on f, we have

$$f(\mathbf{x} + P) = \sum_{\mathbf{v} \in \mathbb{N}^m} [D^{\mathbf{v}} f](P) \cdot \mathbf{x}^{\mathbf{v}}. \tag{4}$$

We call the right-hand side of (4) the Taylor expansion of f at P, denoted by $T(f, P)$. Note that (4) is equivalent to

$$f(\mathbf{x}) = \sum_{\mathbf{v} \in \mathbb{N}^m} [D^{\mathbf{v}} f](P) \cdot (\mathbf{x} - P)^v = \sum_{\mathbf{v} \in \mathbb{N}^m} [D^{\mathbf{v}} f](P) \cdot (x_1 - a_1)^{v_1} \cdots (x_m - a_m)^{v_m},$$

which is the more typically referred to form of Taylor expansion.

A subset $\Delta \subseteq \mathbb{N}^m$ is called a *delta set* (or a *Ferrers diagram*, or an *order ideal*), if it closed under the division order; that is, if $\mathbf{u} \in \Delta$ then $\mathbf{v} \in \Delta$ for all $\mathbf{v} = (v_1, \ldots, v_m) < \mathbf{u} = (u_1, \ldots, u_m)$ componentwise. Define

$$T(f, P, \Delta) = \sum_{\mathbf{v} \in \Delta} [D^{\mathbf{v}} f](P) \cdot \mathbf{x}^{\mathbf{v}}. \tag{5}$$

$T(f, P)$ denotes the full (possibly infinite if f is not a polynomial) Taylor expansion of f, while $T(f, P, \Delta)$ is truncated to consider only those coefficients corresponding to monomials with exponents in Δ. For any nonzero polynomial $f \in \mathbb{F}[x_1, \ldots, x_m] = \mathbb{F}[\mathbf{x}]$, a point $P \in \mathbb{F}^m$ and a delta set $\Delta \subset \mathbb{N}^m$, f is said to vanish at P with multiplicity Δ if $T(f, P, \Delta) = 0$.

On the other hand, f is said to have *arithmetic multiplicity* m_0 at P if $[D^{\mathbf{v}} f](P) = 0$ for all v with $|v| = v_1 + v_2 + \ldots + v_m < m_0$. In terms of the algebraic definition, this implies that the multiplicity set Δ is restricted to a triangular shape. The algebraic definition clearly subsumes the arithmetic one.

With the algebraic definition of multiplicity in mind, we generalize Algorithm 1 to compute the vanishing ideal of a set of points $\{P_1, \ldots, P_n\}$, each point having multiplicity defined by the sets $\Delta_1, \ldots, \Delta_n$. Denote this ideal by

$$\mathbf{I}((P_1, \Delta_1), \ldots, (P_n, \Delta_n)) = \{f \in \mathbb{F}[x_1, \ldots, x_m] : T(f, P_i, \Delta_i) = 0, \quad 1 \le i \le n\}.$$

Since the Δ_i's are delta sets, one can show that this set is indeed an ideal in $\mathbb{F}[x_1, \ldots, x_m]$. (This is not true if the Δ_i's are not all delta sets.)

Algorithm 2 varies from Algorithm 1 in the following way. Instead of evaluating each $f \in G$ at P_i, we need to compute the truncated Taylor expansion $T(f, P_i, \Delta_i)$; we denote the set of these expansions by $\mathcal{T}$. The points in each Δ_i must be ordered in nondecreasing order to ensure that these Taylor expansions may be computed efficiently and to ensure that G at each iteration (*i.e.*, at the start of line 8) is a Gröbner basis for the vanishing ideal of the points $P_1, \ldots, P_k$ with multiplicities $\Delta_1, \ldots, \Delta_{k-1}$ and the subset of points in Δ_k up to $\mathbf{v}$.

Like Algorithm 1, Algorithm 2 is an iterative method; in fact, not only does the algorithm build the Gröbner basis for the vanishing ideal "one point at a time" but it also builds it "one multiplicity at a time." That is, when a new point is introduced, the algorithm updates the Gröbner basis by stepping through the corresponding multiplicity set element by element. Of course if each multiplicity set is trivial ($|\Delta_i| = 1$), then Algorithm 2 is equivalent to Algorithm 1.

The following analogue to Lemma 1 is necessary to establish the correctness of Algorithm 2. We omit the proof, noting only that the key step that established Algorithm 1 is the same for this algorithm. Namely, at each step in the algorithm, we add exactly one element from a multiplicity set and exactly one element to the monomial basis. This ensures that our basis G is always Gröbner.

Lemma 3. *Fix a monomial order on* $\mathbb{F}[\mathbf{x}]$*, and let* $V = \{(P_1, \Delta_1), \ldots, (P_n, \Delta_n)\}$*, where* $P_i \in \mathbb{F}^m$ *are distinct and* $\Delta_i \subset \mathbb{N}^m$ *are delta sets. Then* $\{g_1, \ldots, g_s\} \subset \mathbf{I}(V)$ *is a Gröbner basis for* $\mathbf{I}$ *if and only if* $|\mathcal{B}(g_1, \ldots, g_s)| = \sum_{j=1}^{n} |\Delta_j|$.

Proof. We know (Lemma 3.8 in [14]) that $g_1, \ldots, g_s \in \mathbf{I}(V)$ form a Gröbner basis if and only if $|\mathcal{B}(g_1, \ldots, g_s)| = \dim \mathbb{F}[x_1, \ldots, x_m]/\mathbf{I}(V)$. We just need to show that the latter has dimension equal to $\sum_{j=1}^{n} |\Delta_j|$. To see this, let $I_j = \mathbf{I}(P_j, \Delta_j)$, the vanishing ideal of P_j with multiplicity Δ_j. Then $\mathbf{I}(V) = I_1 \cap \cdots \cap I_n$ and

$$\mathbb{F}[x_1, \ldots, x_m]/\mathbf{I}(V) \cong \bigoplus_{j=1}^{n} \mathbb{F}[x_1, \ldots, x_m]/I_j,$$

Table 3. Algorithm for computing the reduced Gröbner basis for the vanishing ideal of a set of points with multiplicities

Algorithm 2

Input: $P_1, \ldots, P_n \in \mathbb{F}^m$; $\Delta_1, \ldots, \Delta_n \subset \mathbb{N}^m$; and a monomial order.
Output: G, the reduced Gröbner basis for $\mathbf{I}((P_1, \Delta_1), \ldots, (P_n, \Delta_n))$, in increasing order.

$G := \{1\}$; /* g_i *is the* i*th polynomial in* G*, in increasing order* */
Order the variables so that $x_1 < x_2 < \ldots < x_m$;
Order the elements in each Δ_k in nondecreasing order under the division order;
FOR k from 1 to n DO
  Compute $\mathcal{T} = \{T_j = T(g_j, P_k, \Delta_k) : g_j \in G\}$, the set of (truncated) Taylor expansions;
  FOR $\mathbf{v}$ in Δ_k DO
    Find the smallest i so that $\text{coeff}(T_i, \mathbf{x}^{\mathbf{v}}) \neq 0$;
    FOR j from $i+1$ to $|G|$ DO
      $\delta := \text{coeff}(T_j, \mathbf{x}^{\mathbf{v}})$ / $\text{coeff}(T_i, \mathbf{x}^{\mathbf{v}})$;
      $g_j = g_j - \delta \cdot g_i$;
      $T_j = T_j - \delta \cdot T_i$;
    END FOR;
    $G := G \setminus \{g_i\}$ and $\mathcal{T} := \mathcal{T} \setminus \{T_i\}$;
    FOR j from 1 to m DO
      IF $x_j \cdot \text{LT}(g_i)$ not divisible by any LT of G THEN
        Compute $h := \text{Reduce}((x_j - a_j) \cdot g_i, G)$;
        $T_h := x_j \cdot T_i$ (truncated);
        Insert (in order) h into G and T_h into $\mathcal{T}$;
      END IF;
    END FOR;
  END FOR;
END FOR;

RETURN G.

as rings over $\mathbb{F}$. Note that $\{\mathbf{x}^\alpha : \alpha \in \Delta_j\}$ forms a basis for $\mathbb{F}[x_1, \ldots, x_m]/I_j$ as a vector space over $\mathbb{F}$, so its dimension is $|\Delta_j|$. The lemma follows immediately. □

5 Final Remarks

Algorithm 1 is included in MAPLE10 under the command `VanishingIdeal` in the `PolynomialIdeals` package. MAPLE code for the application of this algorithm to multivariate polynomial and rational function interpolation may be downloaded from [19]. A GAP implementation of Algorithm 1 by Joyner [20] is also available.

References

1. Abbott, J., Bigatti, A., Kreuzer, M., Robbiano, L.: Computing ideals of points. J. Symbolic Comput. **30** (2000), 341-356
2. Buchberger, B., Möller, H. M.: The construction of multivariate polynomials with preassigned zeros. Computer algebra, EUROCAM '82, pp. 24-31, Lecture Notes in Comput. Sci., vol. 144, Springer, Berlin-New York, 1982
3. Cerlienco, L. and Mureddu, M.: From algebraic sets to monomial linear bases by means of combinatorial algorithms. Formal power series and algebraic combinatorics (Montreal, PQ, 1992). Discrete Math. **139** (1995), no. 1-3, 73-87
4. Cox, D., Little, J., O'Shea, D.: Ideals, varieties, and algorithms, 2nd ed. Undergraduate Texts in Mathematics, Springer-Verlag, New York, 1997
5. Cox, D., Little, J., O'Shea, D.: Using algebraic geometry. Graduate Texts in Mathematics, 185, Springer-Verlag, New York, 1998
6. Farr, Jeffrey B., Gao, Shuhong: Gröbner bases and generalized Padé approximation. Math. Comp. (to appear)
7. Farr, Jeffrey B., Gao, Shuhong: Gröbner bases, Padé approximation, and decoding of linear codes. Coding Theory and Quantum Computing (Eds. D. Evans et al.), 3–18, Contemp. Math., 381, Amer. Math. Soc., Providence, RI, 2005
8. Faugere, J., Gianni, P., Lazard, D., Mora, T.: Efficient computation of zero-dimensional Gröbner bases by change of ordering. J. Symbolic Comput. **16** (1993), 329-344
9. Fitzpatrick, P., O'Keeffe, H.: Gröbner basis solutions of constrained interpolation problems. Fourth special issue on linear systems and control. Linear Algebra Appl. **351/352** (2002), 533-551
10. Gasca, M., Sauer, T.: Polynomial interpolation in several variables, in Multivariate polynomial interpolation. Adv. Comput. Math. **12** (2000), no. 4, 377-410
11. Guruswami, V., Sudan, M.: Improved decoding of Reed-Solomon and algebraic-geometric codes. IEEE Transactions on Information Theory **46** (1999), no. 6, 1757–1767
12. Koetter, R., Vardy, A.: Algebraic Soft-Decision Decoding of Reed-Solomon Codes. IEEE Transactions on Information Theory **49**(2003), 2809–2825
13. Laubenbacher, R., Stigler, B.: A computational algebra approach to the reverse engineering of gene regulatory networks. Journal of Theoretical Biology **229** (2004), 523-537
14. Marinari, M. G., Möller, H. M., Mora, T.: Gröbner bases of ideals defined by functionals with an application to ideals of projective points. Appl. Algebra Engrg. Comm. Comput. **4** (1993), no. 2, 103-145
15. Marinari, M. G., Möller, H. M., Mora, T.: On multiplicities in polynomial system solving. Trans. Amer. Math. Soc. **348** (1996), no. 8, 3283-3321
16. Pistone, G., Riccomagno, E., Wynn, H. P.: Algebraic Statistics: Computational Commutative Algebra in Statistics. Monographs on Statistics & Applied Probability 89, Chapman & Hall/CRC, 2001
17. Robbiano, L.: Gröbner bases and statistics. *Gröbner bases and applications* (Linz, 1998), 179-204, London Math. Soc. Lecture Note Ser., vol. 251, Cambridge Univ. Press, Cambridge, 1998
18. Sudan, M.: Decoding of Reed Solomon codes beyond the error-correction bound. J. Complexity **13** (1997), no. 1, 180–193
19. Shuhong Gao's webpage. http://www.math.clemson.edu/~sgao/
20. David Joyner's webpage. http://cadigweb.ew.usna.edu/~wdj/gap/curves/