

Efficiently Computing Gröbner Bases of Ideals of Points

Winfried Just¹ and Brandilyn Stigler²

¹Department of Mathematics, Ohio University, Athens, OH 45701

²Mathematical Biosciences Institute, The Ohio State University, Columbus, OH 43210

February 2, 2008

DEDICATED TO AVNER FRIEDMAN ON THE OCCASION OF HIS 75TH BIRTHDAY

Abstract

We present an algorithm for computing Gröbner bases of vanishing ideals of points that is optimized for the case when the number of points in the associated variety is less than the number of indeterminates. The algorithm first identifies a set of essential variables, which reduces the time complexity with respect to the number of indeterminates, and then uses PLU decompositions to reduce the time complexity with respect to the number of points. This gives a theoretical upper bound for its time complexity that is an order of magnitude lower than the known one for the standard Buchberger-Möller algorithm if the number of indeterminates is much larger than the number of points. Comparison of implementations of our algorithm and the standard Buchberger-Möller algorithm in *Macaulay 2* confirm the theoretically predicted speedup. This work is motivated by recent applications of Gröbner bases to the problem of network reconstruction in molecular biology.

Keywords: Gröbner basis, vanishing ideal of points, zero-dimensional radical ideal, standard monomial, biological applications, run-time complexity. MSC: 13P10, 92C40.

1 Introduction

Recently, Gröbner bases have been proposed as a promising selection tool in applications to molecular biology [7, 3]. In these applications, the data consists of m vectors of discretized concentration values in a finite field k for a network of n biochemicals. The data points can be viewed as an affine variety V with points in k^n of multiplicity one and correspond to the vanishing ideal $\mathbf{I}(V)$ of these points in the polynomial ring $k[x_1, \dots, x_n]$. Each variable x_i represents the i -th biochemical which takes on values in k . Typically, the number of data points $m = |V|$ is on the order of tens, while the number of variables n may be in the thousands (for example, see [13]). This requires finding Gröbner bases in situations where $m \ll n$, and the run-time of algorithms for this step constitutes a bottleneck for overall feasibility of these calculations. The primary motivation of this paper is to find an algorithm that optimizes run-time in the case when $m \ll n$.

Several methods have been described and implemented for computing Gröbner bases and the associated standard monomials of vanishing ideals of points. In [10], the authors presented the Buchberger-Möller (BM) algorithm for computing the reduced Gröbner basis of the ideal of a variety V over a field. The BM algorithm performs Gaussian elimination on a generalized Vandermonde matrix and its complexity is quadratic in the number of indeterminates and cubic in the number of points in V [1, 9, 11, 12]. Farr and Gao presented an algorithm based on a generalization of Newton interpolation [4]. While the complexity of their algorithm is exponential in the number n of indeterminates, the algorithm has been designed for the case in which n is small as compared to the number of points. Lederer proposed a method for lexicographic term orders which gives insight into the structure of the Gröbner basis [8]. Cerlienco and Mureddu proposed a combinatorial method that uses Ferrers diagrams to compute the set of standard monomials for the vanishing ideal of a given set of points with respect to an inverse lexicographical order [2].

In [6], the present authors introduced a modification of BM specifically for the case when the number of points m in a given variety is less than the number of indeterminates n . The EssBM (for Essential Buchberger-Möller) algorithm proposed in that paper identifies *essential* variables, that is, those in the support of the standard

monomials associated to the ideal of the points, and computes the relations in the reduced Gröbner basis in terms of these variables using BM. Since the standard monomials are in terms of at most m variables, the computation of a Gröbner basis can be restricted to a proper subring of the underlying ring involving only the essential variables. EssBM was shown to have a worst-case complexity of $O(nm^3 + m^6)$, which is dominated by the first term when $n \gg m$.

Here we present an improvement of the EssBM algorithm in which we eliminate the use of BM altogether. This new algorithm, which we call EssGB (for Essential Gröbner Bases), makes use of PLU decompositions providing an overall improvement in worst-case complexity to $O(nm^2 + m^4)$ for a fixed finite field.

The remainder of our paper is organized as follows. In Section 2 we give a description of the algorithm and in Section 3 we provide the theoretical background for it. In Section 4 we estimate the worst-case time complexity of our algorithm. We conclude with a summary of the performance of an implementation of our algorithm in the computer algebra system *Macaulay 2* on test data. These empirical tests confirm the theoretically predicted speedup relative to implementations of BM and EssBM on the same platform.

2 The EssGB Algorithm

Throughout this paper, let $R = k[x_1, \dots, x_n]$ denote a polynomial ring over a finite field k , and let \prec be a fixed term order on R . For $a = (a_1, \dots, a_n) \in \mathbb{Z}_{\geq 0}^n$, let x^a denote the monomial $x_1^{a_1} \cdots x_n^{a_n}$.

Definition 2.1. *The support of a monomial $x^a \in R$ is $\text{supp}(x^a) = \{x_i : x_i | x^a\}$.*

This is not to be confused with the support of a *polynomial* f , denoted $\text{Supp}(f)$, which is the set of monomials that occur in f .

Let $V \subset k^n$ be a variety of points with multiplicity one and $|V| = m < \infty$. We consider the problem of computing the reduced Gröbner basis of the vanishing ideal $\mathbf{I}(V)$ of the points in V with respect to \prec . We call a Gröbner basis G *reduced* if all generators are monic (leading coefficients are equal to 1) and for all $g, h \in G$, if $g \neq h$, then the leading term of g does not divide any monomial in $\text{Supp}(h)$. We call a polynomial $f \in R$ *reduced with respect to G* if f is the normal form of a polynomial $f' \in R$ with respect to G , that is, f is the remainder of f' upon division by the elements of G . If the context is clear, we simply say that f is reduced.

Let $I \subset R$ be an ideal and G a Gröbner basis for I with respect to \prec . For any $f \in I$, let $LT(f)$ denote the leading term of f with respect to \prec and $\text{tail}(f)$ the polynomial $f - LT(f)$. The ideal generated by the set $\{LT(g) : g \in G\}$ is denoted by $LT(G)$. Further, let $SM(G)$ be the set of monomials not in $LT(G)$. Note that $SM(G)$ is a k -vector space basis for R/I . We call $SM(G)$ the *set of standard monomials associated to G* . In this paper, we restrict our attention to the case where $I = \mathbf{I}(V)$ for a finite variety, that is, I is a zero-dimensional radical ideal, and $SM(G)$ is a finite basis for R/I .

Definition 2.2. *A variable x_i is essential if $x_i \in SM(G)$.*

Equivalently, x_i is essential if and only if there is a monomial $x^a \in SM(G)$ such that $x_i \in \text{supp}(x^a)$. Let $EV(G)$ denote the union of the supports of the standard monomials $x^a \in SM(G)$. Note that $LT(G)$, $SM(G)$, and $EV(G)$ depend only on the ideal $\mathbf{I}(V)$ and the term order \prec . Thus we can indicate this dependence by the notation chosen here.

Let $P = \{p_1, \dots, p_s\} \subset k^n$ be a set of points. A polynomial $f \in R$ is a *separator of $p_i \in P$* if $f(p_i) = 1$ and $f(p_j) = 0$ for all other $p_j \in P$. Given a variety V of points of multiplicity one and a term order \prec , the EssGB algorithm returns the triple $(G, SM(G), S)$, where G is the reduced Gröbner basis of the ideal $\mathbf{I}(V)$ of points in V with respect to \prec ; $SM(G)$ is the set of standard monomials associated to G ; and S is the set of reduced separators of the points in V .

Initialize each set as follows: $EV_0 = \{\}$ and $SM_0 = \{1_R\}$. Let $[n]$ denote the set $\{1, \dots, n\}$ and for $i \in [n]$, let EV_i and SM_i denote i -th approximations of the corresponding sets.

For each $i \in [n]$, do the following. Find the i -th smallest variable, say x_i . Suppose there are r monomials x^{a_1}, \dots, x^{a_r} in SM_{i-1} . Note that these are k -linearly independent. Try to write x_i as a k -linear combination of these monomials. That is, find (if they exist) $c_1, \dots, c_r \in k$, where

$$\begin{aligned} x_i(p_1) &= \sum_{j=1}^r c_j x^{a_j}(p_1) \\ x_i(p_2) &= \sum_{j=1}^r c_j x^{a_j}(p_2) \\ &\dots \\ x_i(p_m) &= \sum_{j=1}^r c_j x^{a_j}(p_m) \end{aligned} \tag{1}$$

and $x^a(p_t)$ is the evaluation of x^a at the t -th point in V for $t \in [m]$.

For solving the system (1) we will use a PLU decomposition $P_{i-1}L_{i-1}U_{i-1}$ of the matrix $A_{i-1} = (x^{a_j}(p_t))$ of the monomials $x^{a_j} \in SM_{i-1}$ evaluated at the points in V . This will reduce the time complexity at each step at which no new essential variable is added to EV_{i-1} . Note that, in general, A_{i-1} will not be square, but will have dimensions $m \times r$, where $r = |SM_{i-1}| \leq m$. Still, the standard Gaussian elimination procedure can be applied to find matrices $P_{i-1}, L_{i-1}, U_{i-1}$ whose product is A_{i-1} and such that P_{i-1} has dimensions $m \times m$ and undoes all row exchanges of the Gaussian elimination, L_{i-1} is an $m \times m$ lower triangular matrix with ones on the main diagonal, and U_{i-1} has dimensions $m \times r$ and is upper triangular in the sense that $u_{k\ell} = 0$ whenever $k > \ell$. Thus even if A_{i-1} is not square, it has a PLU decomposition in the above sense.

If the system (1) has a solution, then it must be unique (see Lemma 3.5). If $c_j = 0$ whenever $x_i \prec x^{a_j}$, then x_i is inessential and is the leading monomial of a polynomial in $\mathbf{I}(V)$. In this case, $EV_i = EV_{i-1}$, $SM_i = SM_{i-1}$, $A_i = A_{i-1}$, $P_i = P_{i-1}$, $L_i = L_{i-1}$, and $U_i = U_{i-1}$.

If no solution exists or $c_j \neq 0$ for some j with $x_i \prec x^{a_j}$, then x_i is an essential variable and hence is a standard monomial. In this case let $EV_i = EV_{i-1} \cup \{x_i\}$; compute the set SM_i of standard monomials for the ideal $\mathbf{I}(V) \cap k[EV_i]$ of the points projected onto the variables in EV_i (see Lemma 3.1); and compute the PLU decomposition $P_iL_iU_i = A_i$ of the matrix $A_i = (x^{a_j}(p_t))$ of the monomials $x^{a_j} \in SM_i$ evaluated at the points in V .

At the end of the loop, all essential variables and standard monomials have been identified. The minimum set (with respect to inclusion) of generators x^a of the leading term ideal of $\mathbf{I}(V)$ is identified (see Lemma 3.3), and for each of these generators a polynomial $x^a - g$ is computed so that the set of all of these polynomials forms a reduced Gröbner basis for $\mathbf{I}(V)$. Finally, the set S of reduced separators is then computed by solving a system of linear equations.

2.1 EssGB

Let M be an $(m \times n)$ -matrix with rows being the points of V , and \prec a term order. We will assume that $x_1 \prec \dots \prec x_n$.

Input: $M; \prec$

Output: (GB, SM_n, S) where GB is the (reduced) Gröbner basis for $\mathbf{I}(V)$ with respect to \prec , SM_n is the set of standard monomials for GB , and S is the set of reduced separators of the points in V .

1. Initialize $EV_0 := \{\}$, $SM_0 := \{1_R\}$, $GB := \{\}$, $A_0 := [1, \dots, 1]^T \in k^m$, $P_0 := Id(m)$, $U_0 := [1, 0, \dots, 0]^T \in k^m$, and let L_0 be the $(m \times m)$ -matrix that has ones in the first column and on the diagonal, and zeros elsewhere.
2. FOR $i \in \{1..n\}$ do
 - (a) Initialize $x_i := i$ -th smallest variable, $r := |SM_{i-1}|$, and $b_i := i$ -th column of M .
 - (b) IF there is no solution $c = [c_1, \dots, c_r]^T$ to the system $P_{i-1}L_{i-1}U_{i-1} \cdot c = b_i$ such that $c_j = 0$ whenever $x_i \prec x^{a_j}$ THEN
 - i. $EV_i := EV_{i-1} \cup \{x_i\}$.
 - ii. Compute SM_i in $k[EV_i]$ using the algorithm SM-A.
 - iii. Compute the matrix $A_i := (x^{a_j}(p_t))$, for $x^{a_j} \in SM_i$ and p_t the point in row t of M .
 - iv. Compute the PLU decomposition $P_iL_iU_i$ of A_i .
3. Compute the set LT of generators of the leading term ideal of $\mathbf{I}(V)$ using the algorithm LT-A.
4. FOR $j \in \{1..|LT|\}$ do
 - (a) Let $b_j = (x^{d_j}(p_t))$ be the $(m \times 1)$ -vector of values of the monomial $x^{d_j} \in LT$ evaluated at the points p_t in M .
 - (b) Find a solution $[c_1, \dots, c_m]^T$ of $P_nL_nU_n \cdot c = b_j$.
 - (c) $GB = GB \cup \{x^{d_j} - \sum c_\ell x^{a_\ell}\}$ where $x^{a_\ell} \in SM_n$.
5. Compute the set S of reduced separators for M using the algorithm SP-A.
6. RETURN GB, SM_n , and S .

2.2 Supporting algorithms

This section contains the subroutines used in the main algorithm EssGB.

2.2.1 SM-A

The algorithm SM-A generates a set SM_i of standard monomials for $\mathbf{I}(V) \cap k[EV_i]$, given a newly identified essential variable x_i and the set SM_{i-1} of standard monomials for $\mathbf{I}(V) \cap k[EV_{i-1}]$. It first constructs a sorted set of candidate monomials by forming all products $x_i^q x^a$ of monomials in SM_{i-1} and powers of x_i , for $0 \leq q < |k|$. Then the monomials which are k -linearly independent can be found by identifying the pivots of the evaluation matrix $A_i := (x^{a_j}(p_t))$, where $x^{a_j} \in C$ and p_t is the point in row t of M .

Input: x_i an essential variable; SM_{i-1} .

Output: SM_i the set of standard monomials for $\mathbf{I}(V) \cap k[EV_i]$.

1. Compute the set $C = \{x_i^q x^a : x^a \in SM_{i-1}, 0 \leq q < |k|\}$ of candidate standard monomials.
2. Sort C so that $C = \{x^{a_1}, \dots, x^{a_s} : s = |C|, x^{a_j} \prec x^{a_{j+1}} \text{ for all } j\}$.
3. Compute the matrix $A := (x^{a_j}(p_t))$, for $x^{a_j} \in C$ and p_t the point in row t of M .
4. Compute the row-echelon form U of A .
5. Identify the columns $\pi(1), \dots, \pi(r)$ corresponding to the $r \leq s$ pivots of U .
6. RETURN $SM_i = \{x^{a_{\pi(1)}}, \dots, x^{a_{\pi(r)}}\}$.

2.2.2 LT-A

This algorithm identifies all minimal leading terms x^a of $\mathbf{I}(V)$. We use the following observation, which will be proved in the next section (Lemma 3.3).

Remark 2.3. *The ideal $LT(G)$ is generated by variables $x_i \notin EV(G)$ and monomials x^a such that $\text{supp}(x^a) \subset EV(G)$, $x^a \notin SM(G)$, and x^a is minimal in the sense that no monomial in $LT(G)$ divides x^a .*

Recall that SM_n and EV_n are the sets of standard monomials and essential variables, respectively, after the execution of Step 2. We will assume that SM_n and EV_n are sorted according to \prec .

Input: $SM_n; EV_n$.

Output: LT the set of generators of the leading term ideal of $\mathbf{I}(V)$.

1. Initialize $C :=$ the $(r \times m)$ -matrix of ones, where $r := |EV_n|, m := |SM_n|; LT := \{\}$.
2. FOR $i \in \{1..r\}$ do
 - (a) FOR $j \in \{1..m\}$ do
 - i. IF $x_i x^{a_j} \in SM_n$ where $x_i \in EV_n$ and $x^{a_j} \in SM_n$
 - ii. THEN $C(i, j) := 0$
 - iii. ELSE FOR $k \in \{1..j-1\}$ do
 - A. IF $C(i, k) == 1$ AND $(x^{a_j}) \% (x^{a_k}) == 0$
 - B. THEN $C(i, j) := 0$.
3. FOR $i \in \{1..r\}$ do
 - (a) FOR $j \in \{1..m\}$ do
 - i. IF $C(i, j) == 1$
 - ii. THEN $LT = LT \cup \{x_i x^{a_j}\}$.
4. Remove repeated elements in LT .
5. $LT = LT \cup \{x_i \notin EV_n\}$.
6. RETURN LT .

2.2.3 SP-A

The algorithm SP-A computes the separators of the points in $V = \{p_1, \dots, p_m\}$ in terms of the standard monomials associated to the ideal of the points. For each point p_t , we wish to find a polynomial $s_t(\mathbf{x}) = \sum_{j=1}^m c_j x^{a_j} \in k[x_1, \dots, x_n]$ that satisfies the following:

$$\sum_{j=1}^m c_j x^{a_j}(p_t) = 1; \quad \sum_{j=1}^m c_j x^{a_j}(p_\ell) = 0, \quad \text{for all } \ell \neq t.$$

We can do so by solving the system $A_n c = e_t$, where A_n is the evaluation matrix $A_n = (x^{a_j}(p_\ell))_{\ell, j \in \{1..m\}}$ constructed during execution of EssGB, $c = [c_1, \dots, c_m]^T \in k^m$ is a vector of unknowns, and e_t is a standard column basis vector.

Input: $SM_n = \{x^{a_1} = 1, \dots, x^{a_m}\}$, the set of standard monomials in increasing \prec -order; $(m \times m)$ -matrix A_n in its PLU form $P_n L_n U_n$.

Output: $S = \{s_1(\mathbf{x}), \dots, s_m(\mathbf{x})\}$ the set of reduced separators of the points in V .

1. Initialize $S = \{\}$.
2. FOR $t \in \{1..m\}$ do
 - (a) Compute $c = [c_1, \dots, c_m]^T$ such that $P_n L_n U_n \cdot c = e_t$.
 - (b) $S = S \cup \{s_t(\mathbf{x}) := \sum_{j=1}^m c_j x^{a_j}\}$.
3. RETURN S .

3 Theoretical Background

3.1 SM-A

Recall that the SM-A algorithm computes a sorted list $C = \{x^{a_1}, \dots, x^{a_s}\}$ of candidate monomials and returns $SM_i = \{x^{a_{\pi(1)}}, \dots, x^{a_{\pi(r)}}\} \subset C$, where $\pi(1), \dots, \pi(r)$ refer to the columns of the row echelon form of A corresponding to pivots and $A := (x^{a_j}(p_t))$ is the evaluation matrix computed in Step 3 of the subroutine.

Lemma 3.1. *Let SM_i be the output returned by the SM-A subroutine, given an essential variable x_i and the set SM_{i-1} of standard monomials for $\mathbf{I}(V) \cap k[EV_{i-1}]$. Then SM_i is the set of standard monomials for $\mathbf{I}(V) \cap k[EV_i]$, where $EV_i = EV_{i-1} \cup \{x_i\}$.*

Proof. The set C consists of all multiples of x_i and $x^a \in SM_{i-1}$ and so generates the k -vector space $R/I \cap k[EV_i]$. The dimension of this space is equal to the number r of nonzero rows of the matrix U as computed in Step (4) of SM-A.

Now consider $x^{a_k} \in C$ that is not in the set SM_i returned by SM-A, and let $U(k)$ consist of the first k columns of U . Then $U(k-1)$ and $U(k)$ have the same rank and it follows that the k -th column of A is a linear combination of the columns of A indexed $j = 1, \dots, k-1$. This means that

$$x^{a_k} - \sum_{j=1}^{k-1} c_j x^{a_j} \in \mathbf{I}(V), \quad (2)$$

for some coefficients c_j . Since the elements of C were listed in increasing order with respect to \prec , the monomial x^{a_k} is the leading monomial in (2) and therefore cannot be a standard monomial. Since there must be r standard monomials for $\mathbf{I}(V) \cap k[EV_i]$, these must by default be the monomials returned in Step (6) of SM-A. \square

Remark 3.2. *At the end of Step 2 of EssGB, the set SM_n is indeed the set of standard monomials for $\mathbf{I}(V) \cap k[EV_n]$ (Corollary 2 in [6]). Furthermore, it is the set of standard monomials for $\mathbf{I}(V)$ with respect to \prec (Theorem 7 in [6]).*

3.2 LT-A

Let LT be the output returned by LT-A and let EV_n and SM_n be the sets of essential variables and the standard monomials SM_n as computed in Step 2 of EssGB. Define B to be the set $B = \{x_i : x_i \notin EV_n\} \cup \{x^a : \text{supp}(x^a) \subset EV_n, x^a \notin SM_n, x^a \text{ minimal}\}$, where *minimal* means no monomial in LT divides x^a .

Lemma 3.3. *Let G be a Gröbner basis for $\mathbf{I}(V)$. Then the leading term ideal $LT(G)$ is generated by B .*

Proof. Since the sets of leading terms and of standard monomials for an ideal are mutually exclusive, by definition $B \subset LT(G)$. Let $x^a \in LT(G)$. If $\text{supp}(x^a) \not\subset EV_n$, then there is $x_i \in B$ that divides x^a . Now suppose $\text{supp}(x^a) \subset EV_n$. Clearly $x^a \notin SM_n$. Since there are a finite number of divisors of x^a , there is $x^b \in B$ that divides x^a . Hence, B generates $LT(G)$. \square

Note that B represents the minimum set (with respect to inclusion) of generators for $LT(G)$. In particular, no monomial $x^a \in B$ divides any other monomial in B . Furthermore, the set LT returned by LT-A is the set B .

3.3 SP-A

We know that separators exist (see Corollary 2.14 in [12]). We also know that separators have a canonical form.

Lemma 3.4. *Let $P \subset k^n$ be a set of points, \prec a term order, and G a Gröbner basis of $\mathbf{I}(P)$ with respect to \prec . The reduced separators of the points in P can be written uniquely in terms of the standard monomials in $SM(G)$.*

Proof. Let f be a separator of a point in P . Since there is $p \in P$ such that $f(p) = 1$, then $f \notin \mathbf{I}(P)$. Hence f is a nonzero element of $R/\mathbf{I}(P)$. As $R/\mathbf{I}(P)$ is generated (as a k -vector space) by $SM(G)$, then f has a unique k -linear representation in terms of the standard monomials which is reduced with respect to G . \square

3.4 EssGB

Lemma 3.5. *For all i , the system $P_{i-1}L_{i-1}U_{i-1} \cdot c = b_i$ obtained during Step 2(b) of the execution of the algorithm EssGB has at most one solution.*

Proof. Recall that $A_{i-1} := P_{i-1}L_{i-1}U_{i-1}$ is the $(m \times r)$ -matrix $(x^{a_j}(p_t))$ where the monomials $x^{a_1}, \dots, x^{a_r} \in SM_{i-1}$ and p_t is the point in row t of M . Since the monomials x^{a_1}, \dots, x^{a_r} are chosen to be linearly independent, the rank of A_{i-1} is r . Hence A_{i-1} has a trivial null space. \square

Recall that the LT-A algorithm returns the minimum set $LT = B$ of generators for the leading term ideal of $\mathbf{I}(V)$.

Lemma 3.6. *A finite set $G \subset \mathbf{I}(V)$ is the reduced Gröbner basis of $\mathbf{I}(V)$ with respect to \prec if and only if*

1. G is monic.
2. $\{LT(g) : g \in G\} = B$ and
3. $\text{Supp}(\text{tail}(g)) \subset SM_n$ for every $g \in G$.

Proof. Let $I = \mathbf{I}(V)$. If G is the reduced Gröbner basis for I , then (1) holds by definition. By Lemma 3.3, $B \subseteq \{LT(g) : g \in G\}$. On the other hand, we cannot have different $g, h \in G$ with the leading term of g dividing the leading term of h . Therefore $\{LT(g) : g \in G\}$ must be equal to the minimum set B of its generators, and (2) holds. Moreover, if $x^a \notin SM_n$, then there must be some $x^b \in B$ that divides x^a , and hence x^a cannot be in $\text{Supp}(\text{tail}(g))$ for any $g \in G$, which is equivalent to condition (3).

Now let $G \subset I$ be a finite set that satisfies (1)–(3). Let H be any Gröbner basis, and let $f \in \mathbf{I}(V)$. Then the leading monomial of some $h \in H$ divides $LT(f)$, and by Lemma 3.3, some $x^a \in B$ divides $LT(f)$. Now (2) implies that $LT(g)$ divides $LT(f)$ for some $g \in G$. Thus G is a Gröbner basis and is monic by (1).

Finally, let g, h be different elements of G . Then $LT(g)$ does not divide $LT(h)$ by minimality of B . Moreover, $LT(g)$ cannot divide any monomial in $\text{Supp}(\text{tail}(h))$, since by (3) the latter monomials are standard monomials, while $LT(g)$ is not in SM_n . \square

Theorem 3.7. *Let (G, SM_n, S) be the output returned by the EssGB algorithm, given a variety V and a term order \prec . Then G is the reduced Gröbner basis of $\mathbf{I}(V)$ with respect to \prec , SM_n is the set of standard monomials associated to G , and S is the set of reduced separators of the points in V .*

Proof. This follows from Remark 3.2 and Lemmas 3.4 and 3.6. \square

The algorithm EssGB can be simplified for lexicographical orders. Specifically once a monomial has been identified as a standard monomial in Step i of EssGB, then it continues to be a standard monomial in subsequent iterations. This property can be used to simplify the algorithm SM-A for the case of lexicographical orders. However, the simplification would not reduce the order of magnitude of our worst-case run-time estimate, and we did not implement it.

4 Complexity of the Algorithms

4.1 Complexity of SM-A

Let $p = |k|$. There are $O(pm)$ candidate monomials, which require $O(pm^2 \log(pm))$ steps to sort, assuming that comparison of two exponents is an operation of cost $O(m)$. The matrix A_i has $O(m \cdot pm)$ entries. Computing the row-echelon form of A has time complexity $O(pm \cdot m^2)$. Identification of the columns with pivots is an $O(m \cdot pm)$ operation. Hence the worst-case complexity of SM-A is

$$O(pm + pm^2 \log(pm) + pm^3 + pm^2) = O(pm^2 \log(pm) + pm^3).$$

4.2 Complexity of LT-A

As there are at most m^2 candidate monomials, initialization of the matrix C requires $O(m^2)$ operations. The FOR loop in Step 2 is executed $O(m)$ times, similarly for the FOR loop in Step 2(a). Checking for membership of SM_n in the IF clause of Step 2(a)(i) requires $O(m)$ operations. Checking for divisibility in Step 2(a)(iii) can be implemented by using a look-up table, and can be presumed to have a constant cost in each iteration of 2(a)(iii), while creating the look-up table requires a one-time cost of $O(m^3)$. In all, the cost associated to Step 2 is $O(m^3)$. Step 3 requires $O(m^2)$ computations, while Step 4 requires $O(m^2 \log m)$ computations. The last step requires $O(n)$ computations as there are at most $n - m$ inessential variables. Overall the complexity of LT-A is

$$O(m^2 + m^3 + m^2 + m^2 \log m + n) = O(n + m^3).$$

4.3 Complexity of SP-A

Initialization of the set S is a constant operation. For the FOR loop, since we are using the PLU decomposition of the matrix A , solving each of the m systems in 2(a) requires $O(m^2)$ steps for forward and backward substitution. Maintenance of the set S in 2(b) requires m scalar multiplications. Hence, the complexity of the SP-A algorithm is $O(m)O(m^2 + m) = O(m^3)$.

4.4 Complexity of EssGB

Initialization has cost $O(m)$. In the main FOR loop (Step 2), the IF statement assumes that we have a linear system in PLU form and so requires $O(m^2)$ operations for solving the system using forward and backward substitutions. Given no solution (entering the THEN clause), to compute the new set of standard monomials is $O(pm^2 \log(pm) + pm^3)$. Construction of the matrix A_i requires $O(m^2)$ operations since the numbers of its rows and columns are both bounded above by m and another $O(m^3)$ to compute its PLU decomposition. Since there are at most m essential variables, the THEN clause will only be executed $O(m)$ times, resulting in

$$O(n)O(m^2) + O(m)O(m^2 + pm^2 \log(pm) + pm^3 + m^2 + m^3) = O(nm^2 + pm^3 \log(pm) + pm^4)$$

as the total cost for Step 2.

Executing Step 3 is $O(n + m^3)$, as derived above. Construction of b_j in Step 4(a) requires $O(m)$ operations, while solving the system in 4(b) requires $O(m^2)$ operations each for forward and backward substitution. Appending to the list GB is an $O(m)$ operation. Since there are at most $n + m^2$ leading terms, the total cost of Step 4 is $O(n + m^2)O(m^2) = O(nm^2 + m^4)$.

Executing Step 5 is $O(m^3)$, as derived above. Thus the worst-case complexity of the EssGB algorithm is

$$\begin{aligned} O(m) + O(nm^2 + pm^3 \log(pm) + pm^4) + O(n + m^3) + O(nm^2 + m^4) + O(m^3) \\ = O(pm^3 \log(pm) + pm^4 + nm^2). \end{aligned}$$

If we assume p to be fixed, then the complexity can be reduced to $O(nm^2 + m^4)$. For the applications to biological data where $n \gg m$, the complexity is dominated by $O(nm^2)$.

5 Performance of the EssGB Algorithm

We compared the run-times of the algorithms EssGB, EssBM, and BM on randomly generated varieties containing m points in k^n , where k is a finite field of the form $\mathbb{Z}/p\mathbb{Z}$. We performed this comparison in *Macaulay 2*, version 0.9.97, where each algorithm has been implemented.

We generated $r = 10$ affine varieties for changing values of p , n , and m . Since the algorithms require specification of a term order, we consider this to be parameter as well. The table below lists the values we used for this comparative study.

Parameters	Values
$p =$ cardinality of k	{5, 101}
$n =$ number of variables	{100, 200, 300}
$m =$ number of points	{5, 10, 15}
$\prec =$ term order	{Lex, GRevLex} with default variable order

For $1 \leq i \leq r$, the i -th variety consists of $nr(i)$ randomly generated points, where

$$nr(i) = \frac{m}{5} \left\lceil \frac{r-i+1}{2} \right\rceil.$$

The remaining $m - nr(i)$ points were generated using random homogenous linear polynomials $g_1, \dots, g_{m-nr(i)}$, where $g_j \in k[y_1, \dots, y_{nr(i)}]$. To generate the j -th new point p_j , the coordinates of p_j are computed individually; that is, for $1 \leq \ell \leq n$

$$p_{j\ell} := g_j(p_{1\ell}, \dots, p_{nr(i)\ell}).$$

Note that for $i = 1, 2$, all points are randomly generated. This will result, with probability very close to one, in a variety where the points are in *general position*, that is, there are no linear dependencies among the points ([5], pg. 7). In the runs for $i = 3, \dots, 10$, the enforced randomly chosen linear dependencies ensure that the linear span of the generated variety will have dimension $\leq nr(i)$ (and equal to $nr(i)$ with probability close to one). This choice of test data allowed us to compare run-times of the three algorithms on ideals of varieties with different geometric properties.

We applied the three algorithms to each of the generated varieties. The run-time results are displayed in Figures 1 and 2.

6 Discussion

Recently, Gröbner bases have been used as a selection tool in applications to molecular biology [7, 3]. In these applications, the number of data points m tends to be significantly smaller than the number of variables n . The computation of Gröbner bases constitutes a bottleneck for overall feasibility of these calculations. The primary motivation for our paper was to find an algorithm that optimizes run-time in the case when $m \ll n$.

The time complexity of the standard BM algorithm has been reported in the literature as quadratic in the number of variables n and cubic in the number of points m [10]. This makes it too slow for the applications mentioned in the preceding paragraph. In [6], we developed an algorithm EssBM that has a provable worst-case time complexity of $O(nm^3 + m^6)$ for a fixed finite field k . For the algorithm EssGB presented here, we can improve this worst-case estimate to $O(nm^2 + m^4)$ for a fixed finite field k . The reduction from quadratic to linear scaling in the run-time was achieved in both EssBM and EssGB by first identifying the set of essential variables in a single loop of length n , and performing the most expensive steps of the computation only for these essential variables. While EssBM still uses BM as a subroutine on the reduced set of variables, EssGB eliminates calls to BM altogether and computes all relevant objects by solving systems of k -linear equations. The coefficient matrices used in these equations change only when a new essential variable is encountered. This allows us to use PLU decompositions to reduce the cost to $O(m^2)$ in all but m of the n steps of the main loop, and our overall worst-case estimate follows.

Based on this estimate, one would expect our algorithm to be significantly faster than both BM and EssGB when $m \ll n$. We tested this prediction for randomly generated varieties, with $|V| = m \in \{5, 10, 15\}$. We tested the algorithm on varieties that were generated totally randomly, which should ensure that the points will almost certainly be in general position, and on varieties where an increasing number of the points were expressed as linear combinations of previously defined random points. In order to ensure that we have enough different linear combinations of two points, the smallest field for which we tested our algorithm was $\mathbb{Z}/5\mathbb{Z}$. We also run tests for the rather large field $\mathbb{Z}/101\mathbb{Z}$.

Our test runs neatly confirm that our algorithm EssGB has comparable performance with BM when $n = 100$, and significantly outperforms the latter when $n = 300$. The single exception are the simulations where $m = 15$, $p = 101$, and a GrevLex term order \prec was used. In these simulations the performance of our algorithm becomes only comparable to that of BM when $n = 300$. However, the general pattern still holds: The more variables, the better EssGB performs relative to BM.

We also observed that in general the run-times of EssGB are more consistent for different varieties under the same parameter settings than those for BM or EssBM. The only exception here are the experiments with $m = 10, 15$, $p = 101$, and a Lex term order \prec , where similar magnitudes of run-time fluctuations were observed for all three algorithms. The experiments with GrevLex term orders and $p = 101$ also show a significant decrease of the run-time of EssGB when the number of linear dependencies among the points in the variety increases.

Our simulations do not in general show an advantage of our previous algorithm EssBM over BM, although EssBM clearly does become more competitive with BM as the number n of variables increases. Previous experiments reported in [6] had shown that EssBM outperforms BM when the number of variables starts exceeding 200. However, these experiments were run in implementation 0.9.8 of *Macaulay 2*, while the simulations presented here were run on version 0.9.97. We noticed a significant speedup of the run-times for both BM and EssBM between both versions; it was relatively larger for BM.

In summary, both our theoretical run-time estimates and the test runs reported here indicate that EssGB would be the algorithm of choice if Gröbner bases are to be found for a variety V in $k[x_1, \dots, x_n]$ such that $|V| = m \ll n$.

7 Acknowledgements

The authors would like to thank Xiaoping A. Shen and Luis García for their helpful suggestions and discussions. This material is based upon work supported by the National Science Foundation under Agreement No. 0112050.

References

- [1] J. Abbott, A. Bigatti, M. Kreuzer, and L. Robbiano, *Computing ideals of points*, Journal of Symbolic Computation **30** (2000), no. 4, 341–356.
- [2] L. Cerlienco and M. Mureddu, *From algebraic sets to monomial linear bases by means of combinatorial algorithms*, Discrete Mathematics **139** (1995), 73–87.
- [3] E. Dimitrova, A. Jarrah, R. Laubenbacher, and B. Stigler, *A Gröbner fan method for biochemical network modeling*, ISSAC '07: Proceedings of the 2007 international symposium on Symbolic and algebraic computation (New York, NY, USA), ACM, 2007, pp. 122–126.
- [4] J. Farr and S. Gao, *Computing Gröbner bases for vanishing ideals of finite sets of points*, Applied Algebra, Algebraic Algorithms and Error-Correcting Codes: 16th International Symposium, AAEECC-16 (M. Fossorier, H. Imai, S. Lin, and A. Poli, eds.), Lecture Notes in Computer Science, vol. 3857, Springer Berlin, 2006, pp. 118–127.
- [5] J. Harris, *Algebraic geometry: A first course*, 1st ed., Graduate Texts in Mathematics, vol. 133, Springer-Verlag, New York, 1992.
- [6] Winfried Just and Brandilyn Stigler, *Computing Gröbner bases of ideals of few points in high dimensions*, Communications in Computer Algebra **40** (2006), no. 3, 65–96.
- [7] R. Laubenbacher and B. Stigler, *A computational algebra approach to the reverse engineering of gene regulatory networks*, Journal of Theoretical Biology **229** (2004), 523–537.
- [8] M. Lederer, *The vanishing ideal of a finite set of closed points in affine space*, Available at <http://arxiv.org/abs/math/0604133>, 2006.
- [9] M. Marinari, H. M. Möller, and T. Mora, *Gröbner bases of ideals defined by functionals with an application to ideals of projective points*, Applicable Algebra in Engineering, Communication and Computing **4** (1993), 103–145.
- [10] H. M. Möller and B. Buchberger, *The construction of multivariate polynomials with preassigned zeroes*, Computer Algebra: EUROCAM '82 (J. Calmet, ed.), Lecture Notes in Computer Science, vol. 144, Springer Berlin, 1982, pp. 24–31.
- [11] T. Mora and L. Robbiano, *Points in affine and projective spaces*, Computational Algebraic Geometry and Commutative Algebra, Cortona-91 (D. Eisenbud and L. Robbiano, eds.), Symposia Mathematica, vol. 34, Cambridge University Press, 1993, pp. 106–150.

- [12] L. Robbiano, *Gröbner bases and statistics*, Gröbner Bases and Applications (New York) (B. Buchberger and F. Winkler, eds.), London Mathematical Society Lecture Notes Series, vol. 251, Cambridge University Press, 1998, pp. 179–204.
- [13] M. K. S. Yeung, J. Tegnér, and J. Collins, *Reverse engineering gene networks using singular value decomposition and robust regression*, PNAS **99** (2002), no. 9, 6163–6168.

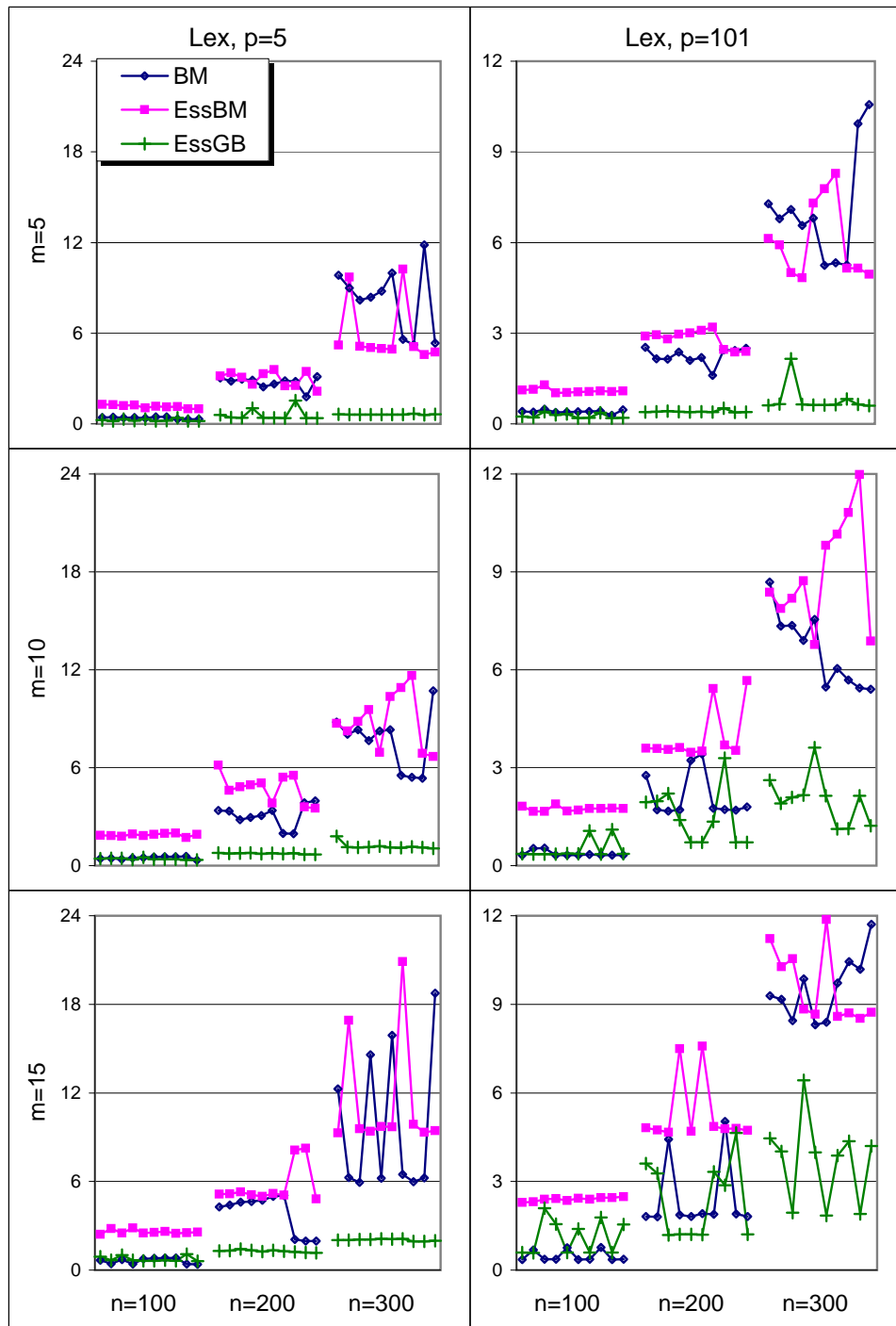


Figure 1: Run-times for the algorithms BM, EssBM, and EssGB for $p \in \{5, 101\}$, $m \in \{5, 10, 15\}$, and $n \in \{100, 200, 300\}$ with a default Lex order.

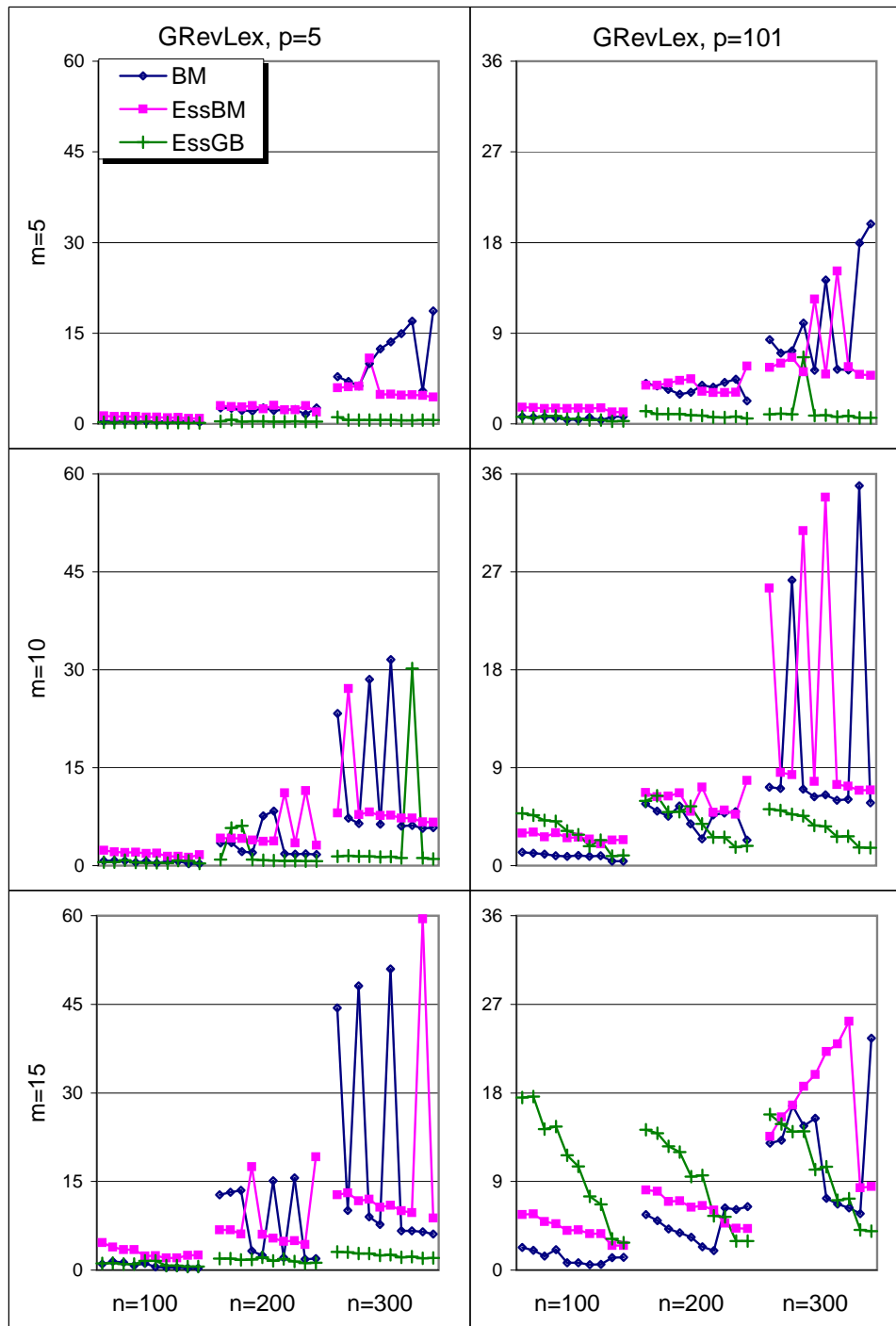


Figure 2: Run-times for the algorithms BM, EssBM, and EssGB for $p \in \{5, 101\}$, $m \in \{5, 10, 15\}$, and $n \in \{100, 200, 300\}$ with a default GRevLex order.