A Practical Method for the Sparse Resultant

(Extended Abstract)

Ioannis Emiris
emiris@cs.Berkeley.edu

John Canny
jfc@cs.Berkeley.edu

Computer Science Division
571 Evans Hall
University of California at Berkeley
Berkeley CA 94720

Abstract

We propose an efficient method for computing the sparse resultant of a system of $n+1$ polynomial equations in n unknowns. Our approach improves upon [CE93] and constructs a smaller matrix whose determinant is a non-zero multiple of the resultant and from which the latter is easily recovered. For certain classes of systems, it attains optimality by expressing the resultant as a single determinant. An implementation of the algorithm is described and empirical results presented and compared with those from [CE93] and [SZ94]. In addition, the important subproblem of computing Mixed Volumes is examined and an efficient algorithm is implemented.

1 Introduction

We are interested in computing the *sparse resultant* of a system of polynomial equations; it generalizes the determinant of a linear system and provides a condition for solvability. Its main use lies in eliminating variables, thus it is also called the *eliminant*, as well as in solving the system via the *u-resultant*.

Resultant-based methods offer the most efficient solution to different problems in areas ranging from robotics [Can88] to graphics and modeling [BGW88]. A concrete example is the inverse kinematics problem for a 6R robot [MC92c], which is solved using a customized resultant in the order of milliseconds. This consists of finding the angles by which each of the six links of the robot must be rotated in order to attain a given final position. Homotopy methods take about 10 seconds, which is unacceptable of industrial manipulators.

Some important benchmarks are related to implicitizing parametric surfaces, which is a fundamental problem in geometric and solid modeling. Given the parametric expression of a surface

$$(x, y, z, w) = (X(s,t), Y(s,t), Z(s,t), W(s,t)),$$

we wish to find its implicit description as the zero set of a single homogeneous polynomial in x, y, z, w. This is achieved by eliminating the parameters s, t from the system

$$\begin{aligned} wX(s,t) - xW(s,t) &= 0 \\ wY(s,t) - yW(s,t) &= 0 \\ wZ(s,t) - zW(s,t) &= 0, \end{aligned}$$

which is equivalent to computing the system's resultant by considering these equations as polynomials in s, t. For a bicubic surface, methods based on sparse resultants have been shown to run faster by a factor of at least 10^3, compared to Gröbner bases and the Ritt-Wu method [MC92b].

If the parametrization has base points, taking the Dixon resultant of a perturbed system leads to an algorithm that terminates successfully in about 30 minutes, while all major Gröbner bases methods seem to run out of memory after running for a few days, even when working on a homomorphic image of the problem over a finite field [MC92a]. In general, Gröbner bases algorithms can also exploit sparseness; yet, when a sparse resultant is known, the solution is much faster, as seen in these applications.

There exist some classes of problems, such as the kinematics of mechanisms and the generalized kinemat-

ics problem with constraints, for which sparse methods are expected to be very efficient. So, ideally, we would like to have a sparse resultant for every problem, which calls for a general algorithm to construct sparse resultants. The first efficient algorithm was proposed in [CE93], while here we take a different tack at obtaining more compact formulae.

More precisely, we decrease the number as well as the order of the matrices required for defining the resultant and, for certain classes of systems, obtain a single matrix whose determinant equals the resultant, in other words, a *Sylvester-type* formula. In Section 8 we discuss in detail some of these results; in particular we study the class of multigraded systems for which a determinantal formula always exists [SZ94] and is obtained with our method; our results also support the conjecture in that paper.

The rest of this paper is organized as follows. The next section provides all definitions and Section 3 puts the new approach into perspective by outlining previous work in the area. The algorithm for constructing the relevant matrix appears in Section 4. Section 5 discusses an indispensable part of our approach, namely the problem of computing Mixed Volumes, which is also of independent interest due to Theorem 2.5. The asymptotic complexity analysis is presented in Section 6 and the implementation in the following section. Section 8 discusses our experiments, in particular for multihomogeneous and multigraded systems, and compares some of these results with a greedy implementation of the algorithm from [CE93]. The paper concludes with our directions for future work and some open questions.

2 Preliminaries

This section defines the problem and the necessary concepts in order to exploit sparseness in polynomial systems. Suppose that we are given $n+1$ polynomials $f_1, \ldots, f_{n+1} \in \mathbf{C}[x_1, \ldots, x_n]$ and that we seek a condition on the coefficients of the f_i that indicates when the system has a solution. Such systems may have *trivial* solutions with some $x_i = 0$ for all coefficient specializations, so we concentrate on solutions $x = \xi$ with $\xi \in (\mathbf{C}^*)^n$, where $\mathbf{C}^* = \mathbf{C} - \{0\}$. Under this assumption, we can deal with the more general case of *Laurent* polynomials $f_i \in \mathbf{C}[x_1, x_1^{-1}, \ldots, x_n, x_n^{-1}] = \mathbf{C}[x, x^{-1}]$.

We use x^e to denote the monomial $x_1^{e_1} \cdots x_n^{e_n}$, where $e = (e_1, \ldots, e_n) \in \mathbf{Z}^n$ is an exponent vector. Let $\mathcal{A}_i = \{a_{i1}, \ldots, a_{im_i}\} \subset \mathbf{Z}^n$ denote the set of exponent vectors corresponding to monomials in f_i with nonzero coefficients, called the *support* of f_i. Then

$$f_i = \sum_{j=1}^{m_i} c_{ij} x^{a_{ij}}, \qquad \text{for } i = 1, \ldots, n+1 \qquad (1)$$

and we suppose $c_{ij} \neq 0$ so that $\mathcal{A}_i$ is uniquely defined given f_i. A polynomial system is *unmixed* if all the exponent sets $\mathcal{A}_i$ are the same for $i = 1, \ldots, n+1$, otherwise it is *mixed*.

To exploit sparseness, certain geometric concepts are needed.

Definition 2.1 The *Newton Polytope* of f_i is the convex hull of $\mathcal{A}_i$, denoted $Q_i = \mathrm{Conv}(\mathcal{A}_i) \subset \mathbf{R}^n$.

For convex sets there is a natural commutative addition operation called Minkowski Addition.

Definition 2.2 The *Minkowski Sum* $A + B$ of convex polytopes A and B in $\mathbf{R}^n$ is the set

$$A + B = \{a + b \,|\, a \in A, b \in B\}.$$

$A + B$ is a convex set, and can be computed as the convex hull of all sums of pairs $(a + b)$ of *vertices* of A and B respectively, where the vertex sum is understood to be componentwise.

Definition 2.3 The *Minkowski Difference* $A - B$ of convex polytopes A and B in $\mathbf{R}^n$ is the set

$$A - B = \{a' \in A \,|\, a' + B \subset A\}.$$

$A - B$ is a convex subset of A but does not define an inverse of the addition operation, since it does not equal $A + (-B)$ and, in general $B + (A - B) \subsetneq A$. However, when A is itself a Minkowski Sum $C + D$, then $(C + D) - D = C$, for convex polytopes C, D.

Let $\mathrm{Vol}(A)$ denote the usual n-dimensional volume of A. We can define a function called Mixed Volume as follows.

Definition 2.4 Given convex polytopes $A_1, \ldots, A_n \subseteq \mathbf{R}^n$, there is a unique, up to multiplication by a scalar, real-valued function $MV(A_1, \ldots, A_n)$ called the *Mixed Volume* which is multilinear with respect to Minkowski Sum. Moreover, if $A_1 = \cdots = A_n$ then $MV(A_1, \ldots, A_n) = n!\mathrm{Vol}(A_1)$.

Equivalently, if $\lambda_1, \ldots, \lambda_n$ are scalars, then $MV(A_1, \ldots, A_n)$ is precisely defined as the coefficient of $\lambda_1 \lambda_2 \cdots \lambda_n$ in $\mathrm{Vol}(\lambda_1 A_1 + \cdots + \lambda_n A_n)$ expanded as a polynomial in $\lambda_1, \ldots, \lambda_n$.

It can be shown that an inclusion/exclusion formula also holds for the Mixed Volume:

$$MV(A_1, \ldots, A_n) = \sum_{I \subset \{1, \ldots, n\}} (-1)^{n-|I|} \mathrm{Vol}(\sum_{i \in I} A_i) \qquad (2)$$

where I ranges over all subsets of $\{1,\ldots,n\}$, $|I|$ denotes its cardinality and the second summation indicates Minkowski Addition.

Before formally introducing the sparse resultant, we mention the BKK bound on the number of common roots of a system, which is the cornerstone of sparse elimination theory, due to work in [Ber75], [Kus75] and [Kho78].

Theorem 2.5 [Ber75] For polynomials $f_1,\ldots,f_n \in \mathbf{C}[x,x^{-1}]$ the number of solutions of the system in $(\mathbf{C}^*)^n$ is either infinite, or does not exceed the Mixed Volume $MV(Q_1,\ldots,Q_n)$. For almost all specializations of the coefficients c_{ij} the number of solutions is exactly the Mixed Volume.

In general, the resultant is a necessary and generically sufficient condition for such a system to have a solution in $(\mathbf{C}^*)^n$. We adopt the following definition for the sparse resultant from [PS93]. Regard a polynomial f_i as a generic point $(c_{i1},\ldots,c_{im_i}) \in \mathbf{C}^{m_i}$ in the space of all possible polynomials with the given supports $\mathcal{A}_i$. It is natural to identify scalar multiples, so the space of all such polynomials contracts to the projective space $\mathbf{P}^{m_i-1}$. Then the input system (1) can be thought of as a point

$$c = (c_{11},\ldots,c_{1m_1},\ldots,c_{(n+1)1},\ldots,c_{(n+1)m_{n+1}}) \in \mathbf{P}^{m_1-1}\times\cdots\times\mathbf{P}^{m_{n+1}-1}.$$

Let $Z_0 = Z_0(\mathcal{A}_1,\ldots,\mathcal{A}_{n+1})$ be the set of all points c such that the system has a solution in $(\mathbf{C}^*)^n$. Finally, let $Z = Z(\mathcal{A}_1,\ldots,\mathcal{A}_{n+1})$ denote the Zariski closure of Z_0 in the product of projective spaces. Z is an irreducible algebraic set.

Definition 2.6 The *sparse resultant* of system (1) is $R = R(\mathcal{A}_1,\ldots,\mathcal{A}_{n+1})$, which is an irreducible polynomial in $\mathbf{Z}[c]$. If $\operatorname{codim}(Z) = 1$ then $R(\mathcal{A}_1,\ldots,\mathcal{A}_{n+1})$ is the defining polynomial of the hypersurface Z. If $\operatorname{codim}(Z) > 1$ then $R(\mathcal{A}_1,\ldots,\mathcal{A}_{n+1}) = 1$.

Let $\deg_{f_i} R$ denote the degree of the resultant R in the coefficients of polynomial f_i of the sparse resultant $R = R(\mathcal{A}_1,\ldots,\mathcal{A}_{n+1})$. In addition, define

$$MV(i) = MV(Q_1,\ldots,Q_{i-1},Q_{i+1},\ldots,Q_{n+1})$$

to be the n-fold Mixed Volume corresponding to $i \in \{1,\ldots,n+1\}$. A consequence of Bernstein's theorem is

Proposition 2.7 [PS93] The sparse resultant is separately homogeneous in the coefficients $(c_{i1},\ldots,c_{im_i})$ of each f_i and its degree in these coefficients equals the mixed volume of the other n Newton polytopes, i.e. $\deg_{f_i} R = MV(i)$.

3 Background and the Present Approach

The simplest system is that of two homogeneous polynomials in two unknowns, which was studied by Sylvester who defined the resultant as the determinant of a matrix in the polynomial coefficients [Sal85]. The multivariate resultant for a system of $n+1$ homogeneous polynomials can be defined in several alternative ways. Hurwitz expressed it as the Greatest Common Divisor (GCD) of $n+1$ inertia forms [vdW50], Cayley [Sal85] via a series of n divisions of determinants, while Macaulay [Mac02] defined it as the quotient of a determinant divided by one of its minors. Another approach, related to Cayley's and also to the algorithm in this article, is to define the homogeneous resultant via a Koszul complex, see for instance [Cha92]. Various modern algorithms exist to construct the multivariate homogeneous resultant including, for instance, Lazard's [Laz81] and Canny's [Can88].

The unmixed sparse resultant was defined as the Chow form of a projective toric variety in [Kus75] and [KSZ92]. Algorithms for its computation and evaluation were proposed in [Stu93], the fastest having complexity super-polynomial in the degree of the resultant and exponential in n with a quadratic exponent, which makes it very expensive for large systems.

For the case of *multigraded* systems Sylvester-type formulations of the resultant exist [SZ94]. These are unmixed systems of polynomials that are homogeneous of degree d_j in each group of l_j+1 variables, where there are r groups of variables and $j \in \{1,\ldots r\}$. In addition, for each j, $l_j = 1$ or $d_j = 1$. The same paper presented the following

Conjecture 3.1 [SZ94] If $l_j \geq 2$ and $d_j \geq 2$ for some j, then the multigraded resultant has no Sylvester-type formula.

Examples of systems with a Sylvester-type formula were examined with our implementation and several such formulae were indeed obtained; in addition, instances of systems satisfying the conjecture's conditions were tested and the results support the conjecture.

An explicit formula for the sparse mixed resultant was given in [PS93] as a *Poisson product*, but it does not amount to an efficient algorithm.

The first efficient method that applies to the general mixed case was proposed in [CE93]. Its complexity for unmixed systems as well as "well-behaved" mixed systems is polynomial in the degree of the resultant and exponential in n with a linear exponent. The sparse resultant is defined as the GCD of $n+1$ determinants of matrices with each entry being either some polynomial coefficient or zero. The resultant is computed for a particular coefficient specialization through a series of n

determinant divisions. To give the definition as a GCD, we generalize Hurwitz's inertia forms; the algorithm reduces to Macaulay's for simplex Newton polytopes.

The algorithm constructs the Minkowski Sum of the $n+1$ Newton polytopes and associates a monomial and an input polynomial to each lattice point p in it. A more general way of associating each point to a monomial and a polynomial is presented by Sturmfels [Stu92], thus removing one of the assumptions in [CE93]. The lattice points index the rows and columns of the matrix; the contents of row p correspond to the product of the monomial with the polynomial associated with p. We proved that this gives a well-defined square matrix whose determinant is not identically zero and is a multiple of the resultant. A greedy implementation of the algorithm has been written on MAPLE by the second author and P. Pedersen of Cornell University and produces a matrix whose order does not exceed that of the original algorithm.

4 Matrix Construction

In this section we describe the central part of the algorithm, namely how to obtain a matrix such that its determinant or some maximal minor is a non-zero multiple of the resultant. The matrix is defined via a linear transformation.

To exploit sparseness and achieve the degree bounds of proposition 2.7 we must work on the sublattice $L \subset \mathbf{Z}^n$ generated by the union of all input supports $\cup \mathcal{A}_i$ [PS93], i.e. on the coarsest common refinement of the sublattices generated by each $\mathcal{A}_i$. It is possible to identify L with $\mathbf{Z}^n$ [Stu92]; in what follows, it is assumed that this has already been done.

4.1 The Linear Transformation

The entries of the matrix have to be among the coefficients of the original polynomials which are thought of as indeterminates. In the very last step, they will be specialized to their values.

Consider the following linear map and its matrix M,

$$M : \mathbf{C}^r \to \mathbf{C}^c : (g_1, \ldots, g_{n+1}) \mapsto g = \sum_{i=1}^{n+1} g_i f_i, \qquad (3)$$

where r is the total number of coefficients of the $n+1$ polynomials $g_i \in \mathbf{C}[x, x^{-1}]$ and c is the number of coefficients of g. Let $\mathcal{B}_i = \mathrm{supp}(g_i)$ for $1 \le i \le n+1$; then r is the sum of the cardinalities of all $\mathcal{B}_i$ and c is the cardinality of

$$\mathrm{supp}(g) = \bigcup_{i=1}^{n+1} \{b + \mathcal{A}_i | b \in \mathcal{B}_i\}.$$

More precisely, every row of M is indexed by a monomial of some g_i and every column by a monomial in g. The row corresponding to monomial x^b of g_i contains the coefficients of polynomial $x^b f_i$. In what follows we restrict ourselves to the case that $r \ge c$.

Lemma 4.1 If $f_1, f_2, \ldots, f_{n+1}$ have a common solution $\xi \in (\mathbf{C}^*)^n$, then M is singular.

Proof For generic coefficients, the sparse resultant vanishes exactly when the polynomials have a common solution $\xi \in (\mathbf{C}^*)^n$. If M is not singular, then the map is surjective and there exists polynomials g_i whose image is a monomial. The value of this monomial at every solution ξ of the system is zero. On the other hand, no monomial can vanish at an element of $(\mathbf{C}^*)^n$, which leads to a contradiction. Hence M must be singular. □

Proposition 4.2 Every maximal minor of M is a multiple of the resultant.

Proof Any maximal minor is the determinant of an $r \times r$ submatrix of M and, from the lemma, it vanishes on the set Z_0 of coefficient specializations such that the system (1) has a common solution. Thus the minor is zero on the Zariski closure Z which is the zero set of $R(\mathcal{A}_i, \ldots, \mathcal{A}_{n+1})$. Since the latter is irreducible, it divides the minor. □

Let $\deg_{f_i} D$ denote the degree in the coefficients of polynomial f_i of a maximal minor D of M. It is clear from Proposition 2.7 that if R divides D then $\deg_{f_i} D \ge MV(i)$ for all i.

4.2 Enumerating Lattice Points

We address the question of enumerating all integer lattice points in a convex polytope $P \subset \mathbf{R}^n$ with integer vertices. A simple approach is to start at one of the vertices and examine all $2n$ points at unit distance from it. These are the neighbors of the starting point on the *lattice graph*. The lattice graph includes all points in $\mathbf{Z}^n$ as vertices and edges between every two points that differ in exactly one coordinate by 1. Every new point is tested against P and rejected if it lies on its exterior, otherwise all of its own neighbors are enumerated.

Unfortunately this method does not work in general, because the lattice points in a polytope may fail to form a connected subgraph of the lattice graph. Consider for instance the triangle with vertex set $\{(0,0),(2,2),(3,2)\}$ in 2 dimensions; neither the interior point $(1,1)$ nor the last two vertices can be reached if the recursion starts at the first vertex $(0,0)$.

In the case that the lattice subgraph $P \cap \mathbf{Z}^n$ is *disconnected*, there exists a vertex of the polytope which is

not reached from the start vertex; this gives a correctness condition for this method. When this condition fails, we can resort to looking at a hyperrectangle that contains the polytope. If μ_j and ν_j are respectively the maximum and minimum among the j-th coordinate of all vertices on P, then this rectangle can be chosen to be the lattice zonotope

$$\mathcal{Z}(P) = [\nu_1, \mu_1] \times \cdots \times [\nu_n, \mu_n] \subset \mathbf{R}^n. \qquad (4)$$

Checking all lattice points in $\mathcal{Z}(P)$ guarantees that no points in P are missed.

4.3 The Algorithm

It is possible for every maximal minor of M defined in Section 4.1 to be identically zero, so once we have defined M we must perform an explicit test on its rank. Below we specify how to construct M in order to increase the probability that it is not identically singular. Let

$$Q = Q_1 + \cdots + Q_{n+1} \subset \mathbf{R}^n$$

be the Minkowski Sum of the input Newton polytopes; consider all n-fold partial Minkowski Sums

$$Q^i = Q - Q_i = \sum_{j \neq i} Q_j \subset \mathbf{R}^n$$

and let

$$T_i = Q^i \cap \mathbf{Z}^n = (Q - Q_i) \cap \mathbf{Z}^n.$$

We shall restrict our choice of $\mathcal{B}_i$ by requiring that it is a subset of T_i; notice that this is also the case in [CE93], although described in different terms. One consequence is that the supports of all products $g_i f_i$ lie within the Minkowski Sum Q, i.e. $\mathrm{supp}(g_i f_i) \subset Q$, therefore $\mathrm{supp}(g) \subset Q$.

In specifying the supports $\mathcal{B}_i$ as subsets of T_i we shall make use of vectors $u_i \in \mathbf{Q}^n$, $1 \leq i \leq n+1$ on a fixed direction $\hat{u} \in \mathbf{Q}^n$. Every u_i may vary in length, but for a given length it corresponds to a unique, one-dimensional polytope $U_i \in \mathbf{R}^n$, defined as the convex hull of the origin and the head of u_i. Then the support of g_i is

$$\mathcal{B}_i = (Q^i - U_i) \cap \mathbf{Z}^n = T_i - U_i.$$

For any i, when u_i decreases in length, the one-dimensional volume of U_i decreases and the cardinality of $\mathcal{B}_i$ tends to that of T_i.

Given polytope Q^i and vector u_i it is possible to directly enumerate $\mathcal{B}_i$ based on the method of Section 4.2. Pick a starting point among the vertices of Q^i which also lies in $Q^i - U_i$. Then visit all of its lattice neighbors and then their own neighbors and so on, as long as every point lies in $Q^i - U_i$. This condition is tested, for point $p \in Q^i$, by taking point $q = p + u_i$ and testing whether it lies in Q^i or not. It is clear that the method works even if u_i has rational coordinates.

This enumeration is not complete unless it reaches all vertices v of Q^i for which $v - u_i$ lies in Q^i. If this condition fails, the algorithm checks the lattice points in zonotope $\mathcal{Z}(Q^i) - U_i$. It is clear that this Minkowski Difference is indeed a zonotope and can be computed as the intersection of $\mathcal{Z}(Q^i)$ and a copy of this polytope shifted by $-u_i$.

The degree of the determinant or minor D that we wish to obtain, in the coefficients of f_i, must be at least $MV(i)$. Hence we pick the initial $\mathcal{B}_i$'s to be of cardinality exactly equal to $MV(i)$; if the produced matrix has a maximal minor that is not identically zero, then this minor equals the resultant and we have obtained a Sylvester-type formula.

Otherwise, the algorithm has to increment all sets $\mathcal{B}_i$ with $i > 1$; for this, it decreases the length of the respective vectors u_i. The new points in $\mathcal{B}_i$ will correspond to additional rows which will be appended to the existing matrix in an incremental way. There will also be new columns that are added to the matrix. Since the cardinality of $\mathcal{B}_1$ is fixed to $MV(1)$, the maximal minor D has, at any time, $\deg_{f_1} D = \deg_{f_1} R$. This will enable us to define R as the GCD of at most $n+1$ of these minors.

Now we address the issue of incrementing the $\mathcal{B}_i$. Let the *$\hat{u}$-distance* of a point in T_i be its distance from the boundary of Q^i on the direction of $\hat{u}$. Then $\mathcal{B}_i$ contains exactly those points with $\hat{u}$-distance at least as large as the one-dimensional volume of U_i or, equivalently, the length of u_i. In addition, the minimum $\hat{u}$-distance in every $\mathcal{B}_i$ is well-defined and denoted by δ_i. There are two major schemes for incrementing $\mathcal{B}_i$, for $i > 1$:

1. Either try to equalize the minimum $\hat{u}$-distance in every $\mathcal{B}_i$, by incrementing only those $\mathcal{B}_i$ with maximum δ_i, which intuitively increases the overlap among the various Minkowski Sums $\mathcal{A}_i + \mathcal{B}_i$ in $\mathbf{Z}^n$ thus reducing the cardinality of $\mathrm{supp}(g)$;

2. or greedily add to each support $\mathcal{B}_i$ one point at a time, by picking the point in $T_i - \mathcal{B}_i$ with the largest $\hat{u}$-distance.

We use the term *round* to describe a phase in the execution of the algorithm during which the supports $\mathcal{B}_i$ are fixed. For the second round, the $\mathcal{B}_i$ are incremented as in scheme (1). If several rounds are required, the two schemes are combined: (2) is used most of the time, with (1) being applied every time a certain number of rounds is completed.

If, after repeatedly incrementing the supports, we have for all $i > 1$, $\mathcal{B}_i = T_i$ then we have failed to find a smaller matrix than that from [CE93] and we have

two choices. Either try another direction $\hat{u}$ or set $\mathcal{B}_1 = T_1$ which implies that M contains a nonzero minor of the proper degree in the coefficients of each polynomial, since it contains, as a submatrix, the matrix defined in [CE93]. In practice the algorithm stops well before all $\mathcal{B}_i = T_i$ and tries a different direction $\hat{u}$.

We summarize now the main algorithm, under the assumption that a direction $\hat{u}$ has been chosen.

1. Support sets $\mathcal{B}_i$ are initialized (only the first time) or incremented, with $\mathcal{B}_1$ fixed; if $\mathcal{B}_i = T_i$, $\forall i > 1$ then pick a different $\hat{u}$ or look for the matrix of [CE93].

2. Construct incrementally the matrix M which represents a linear map (3), for which we know that

   - for every one of its maximal minors D, $\deg_{f_i} D \geq \deg_{f_i} R$, for all i, and
   - at least for $i = 1$ equality holds i.e. $\deg_{f_1} D = \deg_{f_1} R$.

3. If, further, an explicit test indicates that

   - M has at least as many rows as columns and
   - some minor D is not identically zero,

   then D is a non-zero multiple of the sparse resultant; otherwise go to step 1.

In the optimal case the minor D equals the sparse resultant. In general this is not true, and there are two alternative ways to proceed in order to obtain the resultant under a specific specialization of the coefficients, discussed in a general context in [Can88] and adapted in [CE93].

5 Mixed Volume

For unmixed systems it is only needed to calculate the volume of one Newton polytope and then use Definition 2.4. For general cases, there already exist implementations of certain algorithms computing the Mixed Volume of n polytopes in n dimensions, namely of the inclusion/exclusion formula by the first author and an improved one by E. Mahalingam [Mah92]. A more efficient algorithm is sketched below and its implementation is discussed in the next section.

The algorithm, which we call the *Lifting* algorithm, is due to B. Sturmfels; more details are found in [HS92] and [CE93]. Given are convex polytopes $Q_1, \ldots, Q_n \subset \mathbf{R}^n$. First, we choose n sufficiently generic linear functions

$$l_i : \mathbf{R}^n \to \mathbf{R} : \mathbf{Z}^n \to \mathbf{Z}, \quad 1 \leq i \leq n,$$

define the lifted polytopes

$$\hat{Q}_i = \{(q, l_i(q)) | \, q \in Q_i\} \subset \mathbf{R}^{n+1}, \qquad 1 \leq i \leq n,$$

and thus obtain the lifted Minkowski Sum

$$\hat{Q} = \sum_{i=1}^{n} \hat{Q}_i \subset \mathbf{R}^{n+1}.$$

We make use of

Definition 5.1 Given a convex polytope in $\mathbf{R}^{n+1}$, its *lower envelope* is the closure of the subset of all points r on its surface such that, given a point z at infinity on the positive $(n+1)$-st axis, the segment (r, z) intersects the polytope at a point other than r.

The projection of $(n+1)$-dimensional space to its first n coordinates maps the lower envelope of $\hat{Q}$ bijectively onto Q. Moreover, l_i are by definition generic enough so that every point on this lower envelope is uniquely expressed as a sum of points from the n lifted polytopes.

The n-dimensional facets of the envelope induce a *mixed subdivision* of Q into cells that either contribute their volume to the Mixed Volume and are called *mixed*, or contribute zero and are called *unmixed*. From the multilinearity of the Mixed Volume it follows that the mixed cells are exactly those obtained as the Minkowski Sum of n edges. It can be proven that

$$MV = \sum \mathrm{Vol}(\sigma),$$

where the sum ranges over all maximal mixed cells σ of the subdivision.

An additional advantage of this algorithm is that it is directly amenable to computing all $n+1$ n-fold Mixed Volumes in a set of $n+1$ polytopes in n dimensions, which is indeed the case here. For this, simply construct the Minkowski Sum of the $n+1$ lifted polytopes; every facet of its lower envelope either contributes the volume of its projection to exactly one Mixed Volume or contributes to none of the Mixed Volumes. Both the facet and the respective cell is called *i-mixed* if it contributes to $MV(i)$, otherwise it is *unmixed*. A facet is i-mixed exactly when it is the sum of a vertex from $\hat{Q}_i$ and of edges from the other n lifted polytopes.

6 Asymptotic Complexity

The complexity of Linear Programming is polynomial in the number of variables and constraints by the Ellipsoid method or Karmarkar's algorithm; in practice, the Simplex method is used, whose complexity is usually similar. Hence, the cost of constructing the vertex

set of each Q^i and of enumerating the T_i is polynomial in the respective cardinalities. The complexity of the Convex Hull algorithm, used to obtain the facets of a polytope, is proportional to the cardinality of the input point set raised to $\lceil (n+1)/2 \rceil$ and to a polynomial factor in n [EC92].

Finding the supports $\mathcal{B}_i$ has cost polynomial in n and the matrix order. Once matrix M is defined, all linear algebra operations have cost polynomial in its order, which is polynomial in the resultant's degree.

The algorithm may have to examine more than one direction $\hat{u}$ and for each direction it requires several steps with $\mathcal{B}_i$ of increasing cardinalities. Strictly speaking, there exist cases when the algorithm is of the Las Vegas variety, which ensures correctness but not speed. Yet, as long as the total number of matrices examined is single exponential in n, the overall complexity is not affected.

Theorem 6.1 If the total number of different matrices examined is single exponential in n then the overall time complexity for unmixed systems is bounded by a polynomial in n and the order of the matrix. For mixed systems, finding the Mixed Volumes has exponential cost in n. If the mixed system is "well-behaved" then the rest of the algorithm has again complexity polynomial in n and the matrix order.

There is an example of a mixed system in [CE93] that would cause the complexity of this algorithm to be significantly higher that predicted in the theorem. However, this is an artificial system and we do not expect to encounter many such systems in practice.

7 Implementation

Our implementation is in ANSI-C and relies on two existing packages. The most frequently used is a Linear Programming package called in order to compute the convex hull vertices for a set of lattice points in arbitrary dimension. In addition, a Convex Hull program is needed to compute the facet structure and volume of the convex hull of a lattice point set in arbitrary dimension; for this, a package developed by the first author in ANSI-C is used, implementing the Beneath-Beyond algorithm and the perturbation of [EC92] for eliminating degenerate configurations.

The input consists of $n+1$ supports $\mathcal{A}_i$, each represented by an array of m_i n-dimensional integer vectors. The first step is to compute the vertices of each Newton polytope Q_i.

The main body of the program computes the support of the polynomials g_i by first finding the respective partial Minkowski sums Q^i. We compute the Q^i's, or more precisely their vertex sets, directly, by successive application of linear programming:

$$\mathrm{vert}(Q^i) = \mathrm{vert}(\sum_{i \neq j} \mathrm{vert}(Q_i)),$$

where $\mathrm{vert}(P) \subset \mathbf{Z}^n$ denotes the *vertex set* of lattice polytope $P \subset \mathbf{R}^n$. The vertex set of a convex hull is always a subset of the given point set and is represented by having an extra bit per point, set exactly when this point is a vertex.

The supports $\mathcal{B}_i$ are computed in practice by first enumerating the T_i. To enumerate the lattice points T_i in each Q^i, we start at any vertex and recursively move on the integer lattice, testing for each new point whether it lies inside or outside Q^i and stopping the recursion at the exterior points. The interior/exterior test can also be implemented with Linear Programming on $\mathrm{vert}(Q^i)$.

The simplest approach in order to obtain the $\mathcal{B}_i$ in successive rounds is to sort all points in T_i by their $\hat{u}$-distance. Then U_i and u_i need not be explicitly constructed once $\hat{u}$ is given. The initial $\mathcal{B}_i$'s have fixed cardinality and thus are well-defined; incrementing them is also straightforward since the points in T_i are sorted.

Given the supports $\mathcal{B}_i$, the matrix M is constructed incrementally. To search, with high probability, for a non-zero maximal minor it suffices to choose a random specialization of the coefficients and test the resulting integer matrix. The randomization makes the method only more conservative. Sufficient random bits, as given by Schwartz's lemma [Sch80] will make the probability of missing some acceptable submatrix arbitrarily low.

We have also implemented the Lifting algorithm in ANSI-C for computing Mixed Volumes. The l_i functions are represented by random integer n-vectors; an explicit test during the computation of the Minkowski Sum ensures that there are no duplicate points, which implies that the l_i are sufficiently generic. The formula

$$(\cdots(\hat{Q}_1 + \hat{Q}_2) + \cdots + \hat{Q}_{n-1}) + \hat{Q}_n$$

is used to compute $\hat{Q}$ in n steps, the i-th step calculating the Minkowski Sum as the convex hull

$$\mathrm{Conv}(\{a+b | a \in \hat{Q}_1 + \cdots + \hat{Q}_i,\, b \in \hat{Q}_{i+1}\}).$$

This is done in order to eliminate at each phase the interior points; moreover, since only the vertices are needed in order to define any partial Minkowski Sum, we use Linear Programming.

In fact, only the lower envelope of these polytopes are needed. The Convex Hull program that computes the facet structure of $\hat{Q}$ maintains a hash table of all ridges, or faces of codimension 2, each pointing to the two cobounding facets; this is necessary during the hull

construction in order to know the adjacencies of the existing facets. The program constructs a linked list of structures representing the facets by the points that define them; having eliminated degeneracies we know that exactly $n+1$ points will define a facet.

In order to test for a lower envelope facet whether it is mixed or not we need to know the way in which it was created as a Minkowski Sum of faces from the original lifted polytopes. For this, it suffices to keep track of the way every point was produced with a 2-dimensional array of lists `AsSum`, where the first index i indicates the phase in the Minkowski Sum computation and the second is the index j of the point.

$$\texttt{AsSum}[i][j] = [v_1,\ v_2,\ \ldots, v_i], \qquad 2 \le i \le n,$$

signifies that the j-th point in the i-th partial Minkowski Sum is the sum of vertex v_1 of $\hat{Q}_1$, of vertex v_2 of $\hat{Q}_2$ and so on. With this information, a facet is mixed if and only if each lifted polytope has contributed exactly 2 distinct vertices in producing the vertices of the facet.

Since in the present application all n-fold Mixed Volumes are needed, it is more efficient to combine these computations by constructing the lifted Minkowski Sum of $n+1$ lifted Newton polytopes. Array `AsSum` provides sufficient information in order to find the i-mixed facets for $1 \le i \le n+1$.

8 Experimental Results

This section examines multihomogeneous systems in some detail and reports on our experiments with this and other kinds of inputs. Multihomogeneous are called those unmixed systems where the variables can be partitioned into r groups so that each polynomial is homogeneous of degree d_k in each group k, with $k \in \{1, \ldots, r\}$; for the same group, $l_k + 1$ indicates the number of variables. Sturmfels and Zelevinsky [SZ94] studied in particular the subclass of multigraded systems for which, for every $k \in \{1, \ldots, r\}$, we have $l_k = 1$ or $d_k = 1$. We call the system of type $(l_1, \ldots, l_r; d_1, \ldots, d_r)$ if the k-th group has $l_k + 1$ homogeneous variables of degree d_k; notice that the number of equations is $n+1$ where $n = \sum_{k=1}^{r} l_k$.

Sturmfels and Zelevinsky showed that every multigraded system has at least one Sylvester-type formula for its sparse resultant. There should be no confusion from the fact that the polynomials given may be homogeneous; to apply our algorithm we dehomogenize each group of variables by setting the (l_k+1)-st variable to one.

The Newton polytope for every polynomial is then the product of r l_k-dimensional simplices, each on a distinct and non-overlapping set of coordinate axes. Every simplex is denoted by $d_k S^{l_k}$ and is the convex hull of l_k segments of length d_k rooted at the origin and extending along each of the l_k axes corresponding to the variables in this group. Equivalently, S^{l_k} is the convex hull of unit segments. Since we are in the unmixed case, the unique Newton polytope is

$$\prod_{k=1}^{r} d_k S^{l_k} \subset \mathbf{R}^n.$$

Then the n-fold Minkowski Sum Q^i is identical for any $i \in \{1, \ldots, n+1\}$ and equal to integer polytope $P \subset \mathbf{R}^n$ which is simply a copy of the unique input Newton polytope scaled by n, i.e.

$$P = \prod_{k=1}^{r} n d_k S^{l_k} \subset \mathbf{R}^n.$$

It is proven in [SZ94] that there exists one matrix M for every permutation π of the indices $\{1, \ldots, r\}$, such that the determinant of M is the resultant. Clearly all supports $\mathcal{B}_i$ are identical, of cardinality equal to the unique n-fold Mixed Volume. Let $B \subset \mathbf{R}^n$ be the convex hull of $\mathcal{B}_i$. The paper defines M by setting

$$B = \prod_{k=1}^{r} m_k S^{l_k} \subset \mathbf{R}^n$$

for the same l_k as above, where

$$m_k = (d_k - 1) l_k + d_k \sum_{j\,:\,\pi(j) < \pi(k)} l_j, \qquad k \in \{1, \ldots, r\}.$$

Since we are in the unmixed case, the initial vectors $u_i \in \mathbf{Q}^n$ are all identical and equal to u. Similarly, all one-dimensional polytopes $U_i \subset \mathbf{R}^n$ are identical and equal to U.

Lemma 8.1 Partition the n coordinates of vector u into r groups following the partition of variables and set every coordinate in the k-th group equal to $(nd_k - m_k)/l_k \in \mathbf{Q}$. Then $P - U = B$.

Proof By using the fact that $\sum_{k=1}^{r} l_k = n$ and that for every k we have $d_k = 1$ or $l_k = 1$, it can be shown that $(nd_k - m_k)/l_k > 0$, $\forall k$. Consider any point $p \in P - U$ with coordinates grouped in r groups, each of cardinality l_k.

For k such that $l_k = 1$ we have two conditions on coordinate c:

$$0 \le c \le n d_k \text{ and } 0 \le c + \frac{n d_k - m_k}{l_k} \le n d_k$$

which is equivalent to $0 \le c \le m_k$.

For k such that $l_k > 1$ and $d_k = 1$, we have two conditions on the sum s of the l_k coordinates in the k-th group:

$$0 \leq s \leq n \text{ and } 0 \leq s + l_k \frac{n - m_k}{l_k} \leq n$$

which is equivalent to $0 \leq s \leq m_k$. Hence $p \in B$ if and only if $p \in P - U$. □

To see how this u was chosen, observe that B is a scaled-down copy of P, where the scaling has occurred by a different factor for every group of l_k coordinates. Given sequence l_k, polytopes P and B are entirely defined by their unique vertex with no zero coordinate; u is the vector between these two vertices.

Theorem 8.2 For any multigraded system, if u is defined as in the above lemma, then the first matrix constructed by our algorithm has determinant equal to the sparse resultant of the system.

Proof It follows from the lemma that the first set of supports $\mathcal{B}_i$ constructed are all identical, since the system is unmixed, and equal to $B \cap \mathbf{Z}^n$, hence they are exactly those required by [SZ94] to define a Sylvester-type formula for the resultant. Note that the formula obtained corresponds to the permutation π used in the definition of m_k. □

We have been able to produce all possible Sylvester-type formulae for various multigraded examples, including the one from [SZ94] with type $(1, 2; 2, 1)$. This ran on a Sun-4 for less than 10 CPU minutes; as with all running times reported here, the complexity is almost entirely due to our MAPLE implementation of Linear Programming.

We also tested conjecture 3.1 from that paper and could not disprove it; however, our results were far from discouraging. For a system of type $(2, 1; 2, 1)$ we obtained a non-zero minor of order 56 in about 20 minutes, while the total degree of the resultant is 48 and the order of the matrix constructed by a greedy version of the algorithm in [CE93] is 103. This matrix was obtained by mimicking the construction in the case of multigraded systems; here $\hat{u}$ was $[50, 50, -1]$.

For a system of type $(2, 1; 2, 2)$, a non-zero minor of order 120 was obtained while the total degree of the resultant is 96.

The method seemed to be less efficient with small mixed systems. For a system of 3 equations in 2 unknowns we obtained a non-singular matrix of order 32, while the resultant's degree is 26 and the greedy version of [CE93] gave an optimal matrix. The algorithm should perform best for large systems.

9 Future Work

There are several related problems, some of independent interest, whose solution may offer room for improvement, such as computing Mixed Volumes and enumerating lattice points in convex polytopes. More experiments like those reported above are expected to contribute towards classifying all systems that admit a Sylvester-type formula of their resultant. Lastly, a theoretical explanation of our algorithm's efficiency should be possible through the theory of Koszul complexes and a generalization of the notion of degree.

Acknowledgments

The first author has conducted this work while on visit at the Mathematics Department of the University of Nice and Projet S.A.F.I.R. at I.N.R.I.A. Sophia-Antipolis, France. The second author acknowledges support by a David and Lucile Packard Foundation Fellowship and by NSF Presidential Young Investigator Grant IRI-8958577.

References

[Ber75] D.N. Bernstein. The Number of Roots of a System of Equations. *Funct. Anal. and Appl.*, 9(2):183–185, 1975.

[BGW88] C. Bajaj, T. Garrity, and J. Warren. On the applications of multi-equational resultants. Technical Report 826, Purdue Univ., 1988.

[Can88] J.F. Canny. *The Complexity of Robot Motion Planning.* M.I.T. Press, Cambridge, Mass., 1988.

[CE93] J. Canny and I. Emiris. An Efficient Algorithm for the Sparse Mixed Resultant. In G. Cohen, T. Mora, and O. Moreno, editors, *Proc. 10th Intern. Symp. on Applied Algebra, Algebraic Algorithms and Error-Correcting Codes, Lect. Notes in Comp. Science 263*, pages 89–104, Puerto Rico, May 1993. Springer Verlag. Submitted to *SIAM J. Computing.*

[Cha92] M. Chardin. The resultant via a Koszul complex. In F. Eyssette and A. Galligo, editors, *Proc. MEGA '92*, pages 29–39, Nice, April 1992.

[EC92] I. Emiris and J. Canny. An efficient approach to removing geometric degeneracies. In *Proc.*

8th ACM Symp. on Computational Geometry, pages 74–82, Berlin, June 1992.

[HS92] B. Huber and B. Sturmfels. A polyhedral method for solving sparse polynomial systems. To appear in Mathematics of Computation. A preliminary version presented at the "Workshop on Real Algebraic Geometry", August 1992.

[Kho78] A.G. Khovanskii. Newton Polyhedra and the Genus of Complete Intersections. *Funktsional'nyi Analiz i Ego Prilozheniya*, 12(1):51–61, Jan.–Mar. 1978.

[KSZ92] M.M. Kapranov, B. Sturmfels, and A.V. Zelevinsky. Chow polytopes and general resultants. *Duke Math. J.*, 67:189–218, 1992.

[Kus75] A.G. Kushnirenko. The Newton polyhedron and the number of solutions of a system of k equations in k unknowns. *Uspekhi Mat. Nauk.*, 30:266–267, 1975.

[Laz81] D. Lazard. Résolution des systèmes d'équations algébriques. *Theor. Comp. Science*, 15:77–110, 1981.

[Mac02] F.S. Macaulay. Some formulae in elimination. *Proc. London Math. Soc.*, 1(33):3–27, 1902.

[Mah92] E. Mahalingam. A way to compute mixed volumes. Master's thesis, U.C. Berkeley, October 1992.

[MC92a] D. Manocha and J. Canny. The impicit representation of rational parametric surfaces. *J. Symbolic Computation*, 13:485–510, 1992.

[MC92b] D. Manocha and J. Canny. Multipolynomial Resultants and Linear Algebra. In *Proc. ACM Intern. Symp. on Symbolic and Algebr. Computation*, pages 96–102, Berkeley, July 1992.

[MC92c] D. Manocha and J. Canny. Real Time Inverse Kinematics of General 6R Manipulators. In *Proc. IEEE Intern. Conf. Robotics and Automation*, pages 383–389, Nice, May 1992.

[PS93] P. Pedersen and B. Sturmfels. Product Formulas for Resultants and Chow Forms. *Math. Zeitschrift*, 214:377–396, 1993.

[Sal85] G. Salmon. *Modern Higher Algebra.* G.E. Stechert and Co., New York, 1885. reprinted 1924.

[Sch80] J.T. Schwartz. Fast probabilistic algorithms for verification of polynomial identities. *J. ACM*, 27(4):701–717, 1980.

[Stu92] B. Sturmfels. Combinatorics of the sparse resultant. Technical Report 020-93, MSRI, Berkeley, November 1992.

[Stu93] B. Sturmfels. Sparse elimination theory. In D. Eisenbud and L. Robbiano, editors, *Proc. Computat. Algebraic Geom. and Commut. Algebra 1991*, pages 377–396, Cortona, Italy, 1993. Cambridge Univ. Press.

[SZ94] B. Sturmfels and A. Zelevinsky. Multigraded Resultants of Sylvester Type. *J. of Algebra*, 163(1):115–127, 1994.

[vdW50] B.L. van der Waerden. *Modern Algebra.* Ungar Publishing Co., New York, 3rd edition, 1950.