Efficient Incremental Algorithms for the Sparse Resultant and the Mixed Volume *

IOANNIS Z. EMIRIS‡ AND JOHN F. CANNY§

Computer Science Division
University of California at Berkeley
Berkeley CA 94720
USA
E-mail: {emiris,jfc}@cs.Berkeley.edu

July 16, 1994

Abstract

We propose an efficient method for computing the Sparse Resultant of a system of $n+1$ polynomial equations in n unknowns. The first efficient algorithm was proposed by Canny and Emiris [CE93]. The new algorithm produces a smaller matrix whose determinant is a nontrivial multiple of the Sparse Resultant and from which the latter is easily recovered. It is incremental in the sense that successively larger matrices are constructed until one is found with the above properties. For certain classes of systems, the new algorithm attains optimality by expressing the Sparse Resultant as a single determinant. An implementation of the algorithm is described and experimental results are presented. In addition, we propose an efficient algorithm for computing the Mixed Volume of n polynomials in n variables, which provides an upper bound on the number of common roots. This algorithm has also been implemented and empirical results are reported which suggest that this is the fastest algorithm to date.

Keywords: Sparse Resultant, Mixed Volume, Newton Polytope, Asymptotic Complexity, Experimental Results.

1 Introduction

We are interested in computing the *Sparse Resultant* of a system of $n+1$ polynomial equations in n unknowns. The Sparse Resultant provides a condition for the solvability of the system; it generalizes the determinant of a linear system and the Sylvester resultant of two bivariate forms, as well as the classical resultant for n homogeneous polynomials. Resultants are used for eliminating variables, thus they are also called *eliminants*, and in solving systems of equations, for instance by means of the u-resultant construction.

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

*Some preliminary results in I. Emiris and J. Canny, A Practical Method for the Sparse Resultant, in *Proc. ISSAC '93*, pp. 183–192.

‡Work conducted while on visit at the Mathematics Department of the University of Nice and Projet SAFIR at INRIA Sophia-Antipolis, France.

§Supported by a David and Lucile Packard Foundation Fellowship and by NSF Presidential Young Investigator Grant IRI-8958577.

problem for a 6R robot which is solved using a customized Sparse Resultant in 11 milliseconds [MC92c]. 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 real-time industrial manipulators.

Implicitizing parametric surfaces 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

$$wX(s,t) - xW(s,t) = wY(s,t) - yW(s,t) = wZ(s,t) - zW(s,t) = 0,$$

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 packages 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 the Sparse Resultant is known, several problems can be solved much faster, as seen in these applications.

There exist several classes of problems, such as computing the camera displacement in vision and the kinematics of mechanisms [ER94], the generalized kinematics problem with constraints as well as problems in computational biology [PC94] 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 them. The first efficient algorithm was proposed in [CE93], while here we take a different tack in order to obtain more compact formulae. More precisely, we decrease the order of the matrix defining a multiple of the Sparse Resultant and, for certain classes of systems, obtain an optimal matrix whose determinant equals the resultant. The worst-case asymptotic complexity is polynomial in the degree of the resultant and singly exponential, with a linear exponent, in n, provided that certain mild conditions hold. The size of the constructed matrices is asymptotically given by the same bounds, though our experimental results suggest that the average case is more favorable.

A subproblem in our approach is the computation of Mixed Volume which, based on Bernstein's theorem, provides an upper bound for the number of roots of a system of polynomial equations. This estimate is much tighter than Bezout's for sparse systems, in the sense specified in Section 3. Clearly, computing Mixed Volumes is of independent interest. We present an efficient algorithm which is, to the best of our knowledge, the most efficient to date in terms of empirical complexity; its worst-case asymptotic complexity is singly exponential in n, which matches the known lower bounds.

The next section puts the new approach into perspective by outlining previous work in the area. Section 3 provides preliminary definitions. Section 4 specifies our approach and Section 5 presents the algorithm for building Sparse Resultant matrices. Section 6 discusses computing Mixed Volumes. The Sparse Resultant implementation is presented in Section 7 and the asymptotic complexity analysis in the following section. Section 9 shows how our algorithm constructs optimal Sparse Resultant matrices for a class of multihomogeneous systems and lists some experimental results for multihomogeneous systems in general. The paper concludes with directions for future work and some open questions.

2 Related Work

The classical resultant has been examined in the context of *homogeneous* polynomials; since no *a priori* knowledge on the coefficients is assumed, these can be *dense* polynomials. 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 homogeneous polynomials in n variables can be defined in several alternative ways: Cayley [Cay48] defined it via a series of n divisions of determinants, Macaulay [Mac02] as the quotient of a determinant divided by one of its minors while Hurwitz expressed it as the Greatest Common Divisor (GCD) of n *inertia forms* [Hur13, vdW50]. In all cases, the nonzero entries of the matrices are coefficients of the given polynomials. Various more recent algorithms exist to construct this resultant [Laz81, Can88, Ren92].

The *Sparse Resultant* was defined following the study of generalized hypergeometric functions and $\mathcal{A}$-discriminants [GKZ91, GKZ94]; the exact notion of sparseness is formalized and compared to the dense case in the next section. The first general and efficient algorithm for computing the Sparse Resultant of $n+1$ non-homogeneous polynomials in n variables was proposed in [CE93]. It constructs a square matrix whose determinant is not identically zero and is a multiple of the Sparse Resultant. The complexity, under certain conditions, is polynomial in the degree of the resultant and exponential in n with a linear exponent. The Sparse Resultant is defined through a generalization of Hurwitz's inertia forms, as the GCD of $n+1$ determinants of matrices. It is computed for a particular coefficient specialization through a series of n determinant divisions, though for polynomial system solving the resultant matrix suffices. The algorithm constructs the Multivariate Resultant if the input is comprised of dense polynomials, while for linear systems it correctly computes the determinant of the coefficient matrix. A generalization of the algorithm was presented by Sturmfels [Stu94]. A *greedy* implementation has been written on MAPLE by the second author and P. Pedersen and produces a matrix whose order is at most that of the original algorithm.

An integral part of the Theory of Sparse Elimination is Bernstein's bound on the number of toric roots of a square polynomial system. This is a significantly tighter bound than Bezout's, in general, but its calculation requires the computation of the *Mixed Volume* of the given polynomials. Several algorithms have been proposed and implemented for this problem [HS, VVC94], whereas the algorithm in [Emi93] solves the more general problem of computing a *mixed subdivision*. The first two methods have worst-case asymptotic complexity singly-exponential in n, as does our new algorithm, yet the latter improves upon the empirical complexity of previous approaches.

Special interest has been exhibited for multihomogeneous systems, where for certain cases exact *Sylvester-type* formulae are obtained, i.e. matrices whose determinant equals the resultant exactly [SZ94]. These results are applied in Section 9 and our improved algorithm is shown to construct these formulae. A generalization of these results [WZ92], covering a wider class of systems, has yet to offer a constructive approach.

3 Preliminaries

Suppose that we are given $n+1$ non-homogeneous polynomials $f_1, \ldots, f_{n+1}$ in variables $x_1, \ldots, x_n$ with indeterminate coefficients and that we seek a condition on the coefficients that indicates when the system has a solution. We ignore *trivial* solutions with some $x_i = 0$ for all coefficient specializations, thus we can deal with the more general case of *Laurent* polynomials

$$f_i \in K[x_1, x_1^{-1}, \ldots, x_n, x_n^{-1}] = K[x, x^{-1}]$$

where K is the algebraic closure of $\mathbb{Q}(\{c_i | i = 1 \ldots n\})$ and c_i is the sequence of all nonzero coefficients in f_i. We are interested in common *toric* roots $\xi \in (\mathbb{C}^*)^n$ where $\mathbb{C}^* = \mathbb{C} - \{0\}$.

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

$$f_i = \sum_{j=1}^{s_i} c_{ij} x^{a_{ij}}, \qquad c_{ij} \neq 0,\ \forall j \in [1, s_i], \qquad \text{for } i = 1, \ldots, n+1, \tag{1}$$

so that $\mathcal{A}_i$ is uniquely defined given f_i. A polynomial system is *unmixed* if supports $\mathcal{A}_1, \ldots, \mathcal{A}_{n+1}$ are identical, otherwise it is *mixed*.

Definition 3.1 The *Newton Polytope* of f_i is the Convex Hull of support $\mathcal{A}_i$, denoted $Q_i = \mathrm{Conv}(\mathcal{A}_i) \subset \mathbb{R}^n$.

We shall denote the cardinality of the vertex set of Q_i by m_i.

For arbitrary sets there is a natural associative and commutative addition operation called Minkowski Addition.

Definition 3.2 The *Minkowski Sum* $A + B$ of point sets A and B in $\mathbb{R}^n$ is point set

$$A + B = \{a + b \mid a \in A, b \in B\} \subset \mathbb{R}^n.$$

In particular, if A and B are convex polytopes then $A + B$ is a convex polytope.

We are mostly interested in the Minkowski Sums of convex polytopes, for which $A+B$ can be computed as the Convex Hull of all sums $(a + b)$ of *vertices* of A and B respectively. The commutativity of this operation implies that translating A or B is equivalent to translating $A + B$.

Definition 3.3 The *Minkowski Difference* $A - B$ of convex polytopes A and B in $\mathbb{R}^n$ is convex polytope

$$A - B = \{a \in A \mid a + B \subset A\} \subset \mathbb{R}^n.$$

$A - B$ lies in the interior 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 $B + C$, then $(B + C) - B = C$, for any convex polytope C. We also state identities $A - (B + C) = (A - B) - C$ and $(A + U) - B = (A - B) + U$, where U is a one-dimensional polytope.

Let $\mathrm{Vol}(A)$ denote the Lebesgue volume of A in n-dimensional Euclidean space, for polytope $A \subset \mathbb{R}^n$.

Definition 3.4 Given convex polytopes $A_1, \ldots, A_n \subset \mathbb{R}^n$, there is a unique, up to multiplication by a scalar, real-valued function $MV(A_1, \ldots, A_n)$, called the *Mixed Volume* of $A_1, \ldots, A_n$ which is multilinear with respect to Minkowski Addition and scalar multiplication, i.e. for (non-negative) $\mu, \rho \in \mathbb{R}_{\geq 0}$ and convex polytope $A'_k \subset \mathbb{R}^n$

$$MV(A_1, \ldots, \mu A_k + \rho A'_k, \ldots, A_n) = \mu MV(A_1, \ldots, A_k, \ldots, A_n) + \rho MV(A_1, \ldots, A'_k, \ldots, A_n).$$

To define Mixed Volume exactly we require that

$$MV(A_1, \ldots, A_n) = n!\, \mathrm{Vol}(A_1), \qquad \text{when } A_1 = \cdots = A_n.$$

An equivalent definition is

Definition 3.5 For (non-negative) $\lambda_1, \ldots, \lambda_n \in \mathbb{R}_{\geq 0}$ and convex polytopes $A_1, \ldots, A_n \subset \mathbb{R}^n$, the *Mixed Volume* $MV(A_1, \ldots, A_n)$ is 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 $\lambda_1, \ldots, \lambda_n$.

Notice that this definition differs from the classical one [Grü67] by the factor $n!$.

Theorem 3.6 [Ber75] For polynomials $f_1, \ldots, f_n \in K[x, x^{-1}]$ with Newton polytopes $Q_1, \ldots, Q_n \subset \mathbb{R}^n$ the number of common solutions in $(\mathbb{C}^*)^n$ equals $MV(Q_1, \ldots, Q_n)$. For a specific specialization of coefficients in $\mathbb{C}$, the number of roots in $(\mathbb{C}^*)^n$ is either infinite or does not exceed $MV(Q_1, \ldots, Q_n)$.

This is also called the BKK bound, since it relies heavily on work by Kushnirenko [Kus75] and has been alternatively proven by Khovanskii [Kho78], and constitutes the cornerstone of Sparse Elimination Theory. It is at most as high as Bezout's bound and usually significantly tighter for systems encountered in Engineering applications. The two bounds are equal when every Newton polytope is an n-dimensional unit simplex with all vertices on the coordinate axes and scaled by the total degree of the polynomial. This situation is depicted in Figure 1 with dashed lines.

The *Sparse* or *Newton Resultant* provides a necessary and generically sufficient condition for the existence of toric roots for a system of $n+1$ polynomials in n variables; since it applies to mixed systems, it is sometimes called the *Sparse Mixed Resultant*. The complexity of the Multivariate Resultant depends on the Bezout bound, in particular its degree in the coefficients of f_i is $\prod_{j \neq i} d_j$, where d_j is the total degree of f_j. In contrast, the degree of the Sparse Resultant depends on the Mixed Volume of the $n \times n$ subsystems of the given polynomials, according to Theorem 3.8.

To define the Sparse Resultant we regard a polynomial f_i as a generic point $c_i = (c_{i1}, \ldots, c_{im_i})$ in the space of all possible polynomials with the given support $\mathcal{A}_i$. It is natural to identify scalar multiples, so the space of all such polynomials contracts to the projective space $\mathbb{P}_K^{m_i - 1}$ or, simply, $\mathbb{P}^{m_i-1}$. Then the input system (1) can be thought of as a point

$$c = (c_1, \ldots, c_{n+1}) \in \mathbb{P}^{m_1-1} \times \cdots \times \mathbb{P}^{m_{n+1}-1}.$$

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 $(\mathbb{C}^*)^n$ and let $Z = Z(\mathcal{A}_1, \ldots, \mathcal{A}_{n+1})$ denote the Zariski closure of Z_0 in the product of projective spaces. It is proven in [PS93] that Z is an irreducible variety.

Definition 3.7 The *Sparse Resultant* $R = R(\mathcal{A}_1, \ldots, \mathcal{A}_{n+1})$ of system (1) is an irreducible polynomial in $\mathbb{Z}[c]$. 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 and let

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

A consequence of Bernstein's theorem is

Theorem 3.8 [PS93] The Sparse Resultant is separately homogeneous in the coefficients c_i of each f_i and its degree in these coefficients equals the Mixed Volume of the other n Newton polytopes, i.e. $\deg_{f_i} R = MV(i)$.

The Sparse Resultant generalizes the Multivariate Resultant; they coincide when all Newton polytopes are n-simplices scaled by the total degrees of the respective polynomials as described above and the polynomials are homogenized.

Example 3.9 Here is a system of 3 polynomials in 2 unknowns

$$\begin{aligned} f_1 &= c_{11} + c_{12}xy + c_{13}x^2y + c_{14}x \\ f_2 &= c_{21}y + c_{22}x^2y^2 + c_{23}x^2y + c_{24}x \\ f_3 &= c_{31} + c_{32}y + c_{33}xy + c_{34}x \end{aligned} \qquad (2)$$

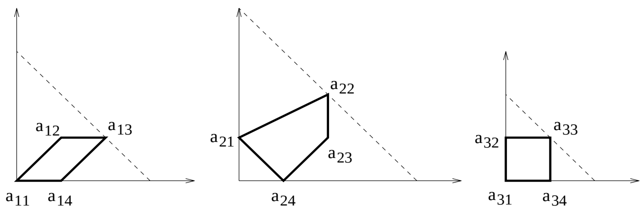

Figure 1: Newton Polytopes for Example 3.9

with Newton polytopes shown in Figure 1. The matrix constructed by the algorithm of [CE93] has size 15, whereas the greedy version of that algorithm and the algorithm in this paper respectively reduce the matrix size to 14 and 12. The Multivariate Resultant has total degree 26 and can be obtained as the Sparse Resultant when the Newton polytopes are the dashed triangles in the figure.

4 Matrix Definition

In this section we describe how to obtain a matrix such that some maximal minor is a nontrivial multiple of the Sparse Resultant. The entries of this resultant matrix are chosen among the indeterminate coefficients of the original polynomials.

To exploit sparseness and achieve the degree bounds of Theorem 3.8 we must work on the sublattice of $\mathbb{Z}^n$ generated by the union of all input supports $\cup \mathcal{A}_i$, i.e. on the coarsest common refinement of the sublattices generated by each $\mathcal{A}_i$. Suppose this sublattice has rank n and is thus identified with $\mathbb{Z}^n$ [Stu94]; in what follows, it is assumed that this has already been done by means of the Smith Normal Form.

Let $\mathcal{P}(\mathcal{A}) \subset K[x, x^{-1}]$ be the set of all Laurent polynomials in n variables with support $\mathcal{A} \subset \mathbb{Z}^n$. Clearly, $f_i \in \mathcal{P}(\mathcal{A}_i)$. Now fix supports $\mathcal{B}_1, \ldots, \mathcal{B}_{n+1} \subset \mathbb{Z}^n$ and consider the following linear transformation:

$$M : \mathcal{P}(\mathcal{B}_1) \times \cdots \times \mathcal{P}(\mathcal{B}_{n+1}) \to \mathcal{P}(\bigcup_{i=1}^{n+1} \mathcal{B}_i + \mathcal{A}_i) : (g_1, \ldots, g_{n+1}) \mapsto g = \sum_{i=1}^{n+1} g_i f_i, \tag{3}$$

where addition between supports stands for Minkowski Addition. The matrix we are constructing is precisely the matrix of this transformation and to define it fully we specify supports $\mathcal{B}_i$ at the end of this section. Every row of M is indexed by an element of some $\mathcal{B}_i$ and every column by an element of $\mathcal{B}_i + \mathcal{A}_i$ for some i; equivalently, the rows and columns are indexed respectively by the monomials of g_i and the monomials of g. We fill in the matrix entries *à la* Macaulay: The row corresponding to monomial x^b of g_i contains the coefficients of polynomial $x^b f_i$ so that the coefficient of monomial x^q appears in the column indexed by x^q, where $b \in \mathcal{B}_i$, $q \in \operatorname{supp}(g)$. Columns indexed by monomials which do not explicitly appear in $x^b f_i$ have a zero entry.

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

Proof If a common solution ξ exists, then it is a solution for all g in the image of linear transformation M. This implies that the image of M cannot contain any monomials and is, therefore, a proper subset of the range. Since M is not surjective it is singular. □

The number of rows equals the sum of the cardinalities of supports $\mathcal{B}_i$ while the number of columns equals the cardinality of $\operatorname{supp}(g)$. Throughout this article we restrict ourselves to matrices M with *at least as many rows as columns.*

Theorem 4.2 Every maximal minor D of M is a multiple of the Sparse Resultant $R(\mathcal{A}_1, \ldots, \mathcal{A}_{n+1})$.

Proof By the lemma, the rank of M is less than the number of columns on the set Z_0 of coefficient specializations such that $f_1, \ldots, f_{n+1}$ have a common solution. Any maximal minor D is zero on Z_0, thus it 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 D in $\mathbb{Z}[c_1, \ldots, c_{n+1}]$ where c_i are the coefficients of f_i. $\square$

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 Theorem 3.8 that if R divides D then $\deg_{f_i} D \geq MV(i)$ for all i.

We now specify the construction of supports $\mathcal{B}_i$. Let

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

be the Minkowski Sum of the input Newton polytopes. Consider all n-fold partial Minkowski Sums

$$Q^i = Q - Q_i = \sum_{j \neq i} Q_j \ \subset \mathbb{R}^n \qquad \text{and let} \qquad T_i = Q^i \cap \mathbb{Z}^n = (Q - Q_i) \cap \mathbb{Z}^n.$$

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

Given a direction $u \in \mathbb{Q}^n$ we define a family of one-dimensional polytopes $U \subset \mathbb{R}^n$, each being the Convex Hull of the origin and of a point $\beta u \in \mathbb{R}^n$, where β is a varying nonzero real number. The sign of β determines the direction in which U lies and its magnitude determines the length of U. For a fixed U we define

$$\mathcal{B}_i \ = \ (Q^i - U) \cap \mathbb{Z}^n \ \subset T_i.$$

As the length of U decreases the cardinality of $\mathcal{B}_i$ tends to that of T_i. So for fixed u and β or, simply, for fixed U, matrix M is well defined. In the next section we specify an algorithmic way for computing $\mathcal{B}_i$.

5 Matrix Construction

This section presents the algorithm for constructing sets $\mathcal{B}_i$ and the resultant matrix.

Definition 5.1 Given convex polytope $A \subset \mathbb{R}^n$ and a vector $u \in \mathbb{Q}^n$, we define the u-*distance* of every point $p \in A \cap \mathbb{Z}^n$ to be the maximum non-negative $s \in \mathbb{R}_{\geq 0}$ such that $p + su \in A$, i.e. it is the distance of p from the boundary of A on direction u.

Integer points on the boundary of polytope A which are extremal with respect to vector u have zero u-distance. Figure 2 shows different subsets of T_2 for system (3.9) with respect to u-distance, for $u = (11, 1)$. An equivalent definition of supports $\mathcal{B}_i$ is by ordering $Q^i \cap \mathbb{Z}^n$ by u-distance, then selecting the points whose u-distance exceeds some bound. Vector u here and in the definition of U at the end of the previous section is the same and β can be used as the bound on u-distance. This is formalized in

Proposition 5.2 For convex polytope A and one-dimensional polytope $U \in \mathbb{R}^n$,

$$(A - U) \cap \mathbb{Z}^n \ = \ \{a \ \in \ A \cap \mathbb{Z}^n \mid u\text{-distance}(a) > \beta = \ \mathrm{length}(U)\}.$$

We now turn to the question of enumerating all integer lattice points T_i in n-fold Minkowski Sum Q^i, for $1 \le i \le n+1$, together with their u-distances for some $u \in \mathbb{Q}^n$. Our *Mayan Pyramid Algorithm* is recursive and computes, at every stage, the range of values for the k-th coordinate in T_i when the first $k-1$ coordinates are fixed i.e. when the algorithm is computing coordinate x_k for $1 < k \le n$, all previous coordinates are set to $p_1, \ldots, p_{k-1}$.

```
Input:   Integers i,k in [1,n] and u ∈ Q^n.
         If k > 1 then a set of integer coordinates p_1,...,p_{k-1} is also given.
Output:  T_i ⊂ Z^n together with the u-distance of the points.
```

```
Mayan Pyramid Algorithm:
        1.  Compute mn,mx ∈ Z which are, respectively, the minimum and maximum x_k-
            coordinates in Q^i when the first k-1 coordinates are fixed to p_1,...,p_{k-1}.
        2.  If k < n, for each p_k ∈ [mn,mx]
            set x_k = p_k and recursively call the algorithm with input i,k+1
            and coordinates p_1,...,p_k.
        3.  If k = n, for each p_k ∈ [mn,mx]
            set x_k = p_k, compute the u-distance of point (p_1,...,p_n) and add it to T_i.
```

The recursion terminates if $[mn, mx]$ is empty for any k. Linear Programming is used to compute mn, mx; here is how we find mn, for some $k > 1$:

$$\text{minimize} \quad s \in \mathbb{R}: \qquad (p_1, \ldots, p_{k-1}, s) = \sum_{l=1, l\neq i}^{n+1} \sum_{j=1}^{m_i} \lambda_{lj} v_{lj}^k; \qquad \sum_{j=1}^{m_l} \lambda_{lj} = 1,\ \lambda_{lj} \ge 0,\ \forall l \neq i, j = 1 \ldots m_l;$$

where v_{lj} are the vertices of Q_l, v_{lj}^k is the k-vector consisting of the first k coordinates of v_{lj} and m_l is the respective cardinality. Then mn is the ceiling of the optimal value of s; the same setup with s maximized gives mx as the floor of the optimum.

Computing u-distances is accomplished by Linear Programming as well:

$$\text{minimize} \quad s \in \mathbb{R}: \qquad (p_1, \ldots, p_n) + su = \sum_{l=1, l\neq i}^{n+1} \sum_{j=1}^{m_i} \lambda_{lj} v_{lj}; \qquad \sum_{j=1}^{m_l} \lambda_{lj} = 1,\ \lambda_{lj} \ge 0,\ \forall l \neq i, j = 1 \ldots m_l.$$

This Linear Program can be used with $k < n$ coordinates for pruning the set of integer points T_i, since in practice we concentrate on points with a positive u-distance. We can then test the point projections as they are constructed and eliminate all points $(p_1, \ldots, p_k)$ whose u^k-distance is zero; the u^k-distance is the analogue of u-distance for the projection of Q^i and of u in the first k dimensions. Further tests of this flavor can achieve a more extensive pruning that decreases the empirical complexity.

Incrementing the supports $\mathcal{B}_i$ is done either by decreasing the length of U or, equivalently, by lowering bound β on the u-distance. The degree of maximal minor D in M must be at least $MV(i)$. Hence we pick the initial sets $\mathcal{B}_i$ to be of cardinality exactly equal to $MV(i)$. If D is generically nonzero then it equals the Sparse Resultant and we have obtained a Sylvester-type formula. Otherwise, points from T_i are added to $\mathcal{B}_i$ and they correspond to additional rows which are appended to the existing matrix; in general, more columns will have to be added as well.

We summarize now the matrix construction algorithm, under the assumption that a direction u has been chosen.

```
Input:   A_i, MV(i), T_i with the u-distance of every point, for i = 1,...,n+1.
Output:  Maximal minor D of matrix M, such that D is a nontrivial multiple of the
         Sparse Resultant, or an indication that such a minor cannot be found.
```

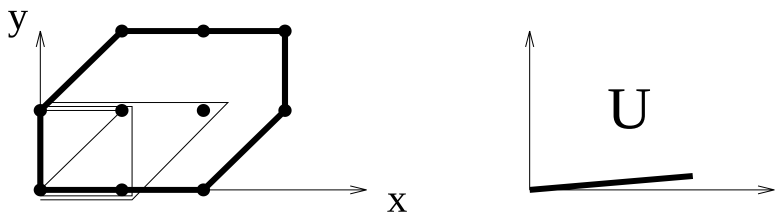

Figure 2: T_2 subsets with different u-distance bounds and U

```
Incremental Matrix Construction Algorithm:
        1.  Initialize supports B_i to include MV(i) points from T_i
            with largest possible u-distance.
        2.  Construct matrix M with random coefficients.
        3.  If M has at least as many rows as columns and it is nonsingular
            then return any maximal minor in it.
        4.  Otherwise, if B_i = T_i for i = 1,...,n+1 i.e. the supports cannot be
            incremented, return with an indication that minor D cannot be found.
        5.  Otherwise, let B_i = {p ∈ T_i | u-distance(p) ≥ β} where β ∈ ℝ is chosen
            so that the minimum number of new points are added to supports B_i and
            at least one B_i is incremented; go to step 2.
```

The nonsingularity test considers a generic matrix M whose nonzero entries are symbolic coefficients. Genericity is simulated by picking uniformly distributed random coefficients from an interval of integers. If we consider the determinant of M as a polynomial in one coefficient from each f_i then the probability that a generically nonsingular M is singular under the specialization is bounded by the ratio of the number of rows of M divided by the size of the interval. The size of M is typically less than 10^4 so one-word integer coefficients lead to an upper bound of 10^{-5} on the probability.

In many situations, a deterministic u which guarantees the construction of a compact matrix formula can be found; such cases include systems whose structure is or resembles the multihomogeneous structure, as demonstrated in Section 9. For arbitrary systems, a random vector u is chosen.

In several applications, including polynomial system solving, an exact matrix formula for the resultant is not required [ER94], though when minor D equals the Sparse Resultant efficiency is optimized. In general $D \neq R$ and there are two alternative ways to proceed in order to obtain the resultant under a specific specialization of the coefficients [CE93]; for both we fix the cardinality of $\mathcal{B}_1$ to $MV(1)$ so that $\deg_{f_1} D = \deg_{f_1} R$. This will enable us to define R as the GCD of at most $n+1$ such minors.

Example 3.9 (cont'd): Figure 2 shows Q^2 of system (2) and polytope U between the origin and $(2, 2/11)$ such that $Q^2 - U$ is the thin-line triangle with vertex set $\{(0,1),(1,1),(0,0)\}$; this is $\mathcal{B}_2$ for the final matrix M. Equivalently, this is the set of integer points in Q^2 whose u-distance is larger than or equal to $5\sqrt{5}/11$, where $u = (2, 2/11)$. The thin polygonal lines in Figure 2 define some subsets of $T_2 \subset \mathbb{Z}^2$ for different cutoff values on the u-distance.

This u leads to a 13×12 nonsingular matrix M shown below for system (2) with $\mathcal{B}_i$ cardinalities $5, 3, 5$, from which any 12×12 minor serves as D. The Sparse Resultant's total degree is the sum of

Mixed Volumes $4+3+4=11$. The first line below displays the integer points indexing the columns.

$$(3,2)(4,3)(5,3)(4,2)(3,1)(5,2)(4,1)(2,2)(3,3)(2,1)(5,4)(4,4)$$

$$M = \begin{bmatrix}
c_{11} & c_{12} & c_{13} & c_{14} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
c_{12} & 0 & 0 & c_{13} & c_{14} & 0 & 0 & 0 & 0 & c_{11} & 0 & 0 \\
c_{14} & c_{13} & 0 & 0 & 0 & 0 & 0 & c_{11} & c_{12} & 0 & 0 & 0 \\
0 & c_{14} & 0 & 0 & 0 & 0 & 0 & 0 & c_{11} & 0 & c_{13} & c_{12} \\
0 & 0 & 0 & c_{12} & c_{11} & c_{13} & c_{14} & 0 & 0 & 0 & 0 & 0 \\
c_{21} & 0 & c_{22} & 0 & 0 & c_{23} & c_{24} & 0 & 0 & 0 & 0 & 0 \\
0 & c_{22} & 0 & c_{23} & c_{24} & 0 & 0 & c_{2}1 & 0 & 0 & 0 & 0 \\
0 & 0 & c_{23} & c_{24} & 0 & 0 & 0 & 0 & c_{21} & 0 & c_{22} & 0 \\
c_{31} & c_{33} & 0 & c_{34} & 0 & 0 & 0 & 0 & c_{32} & 0 & 0 & 0 \\
c_{32} & 0 & 0 & c_{33} & c_{31} & 0 & c_{34} & 0 & 0 & 0 & 0 & 0 \\
c_{33} & 0 & 0 & 0 & c_{34} & 0 & 0 & c_{32} & 0 & c_{31} & 0 & 0 \\
0 & c_{31} & c_{34} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & c_{33} & c_{32} \\
0 & c_{32} & c_{33} & c_{31} & 0 & c_{34} & 0 & 0 & 0 & 0 & 0 & 0
\end{bmatrix}$$

6 Mixed Volume Computation

Computing the Mixed Volume of n Newton polytopes in n dimensions is an indispensable subproblem for the Sparse Resultant matrix construction but also a fundamental question of independent interest in Sparse Elimination. The main idea behind our method is the *Lifting Algorithm* by B. Sturmfels [Stu94]. Given convex polytopes $Q_1, \ldots, Q_n \subset \mathbb{R}^n$, we choose a sufficiently generic lifting defined by n linear functions $l_i : \mathbb{Z}^n \rightarrow \mathbb{Q}$ and define the *lifted polytopes*

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

We define the *lifted Minkowski Sum* as the Minkowski Sum of the lifted polytopes $\widehat{Q} = \sum_{i=1}^n \widehat{Q}_i \subset \mathbb{R}^{n+1}$.

Definition 6.1 Consider distinct vertex sets $\{p_1, \ldots, p_n\}$ and $\{q_1, \ldots, q_n\}$ such that $p_i, q_i \in Q_i$ and $\sum_{i=1}^n p_i = \sum_{i=1}^n q_i$. The lifting defined by functions $l_1, \ldots, l_n$ is *sufficiently generic* if and only if $\sum_{i=1}^n (p_i, l_i(p_i)) \neq \sum_{i=1}^n (q_i, l_i(q_i))$ or, equivalently, $\sum_{i=1}^n l_i(p_i) \neq \sum_{i=1}^n l_i(q_i)$.

The *lower envelope* of a convex polytope in $\mathbb{R}^{n+1}$ is the closure of the subset of all n-dimensional faces, or facets, whose outward normal has a negative x_{n+1}-coordinate.

For a sufficiently generic lifting, the lower envelope of $\widehat{Q}$ is in bijective correspondence with the Minkowski Sum Q of the original polytopes, since every vertex on the lower envelope is expressed uniquely as a Minkowski Sum. This is a weaker requirement on the genericity of l_i since it has to remove any ambiguity in picking the unique point lying on the lower envelope of $\widehat{Q}$. If all $2n^2$ coordinates of l_i are chosen uniformly at random from an interval of size $\lceil 2^{L_l} \rceil$, the probability that genericity fails is bounded by $\prod_i m_i / n^2 2^{L_l}$, where m_i is the vertex cardinality of Q_i and $L_l \in \mathbb{R}_{\geq 0}$. For most problems in practice it suffices to use one-word values for the l_i coordinates. It is straightforward to check deterministically whether a particular choice of lifting functions is sufficiently generic.

The facets of the lower envelope project to maximal cells in a *mixed subdivision* of Q so that each cell either contributes its volume to the Mixed Volume or contributes zero. In the first case, the cell is called *mixed* and is the Minkowski Sum of n edges; in the second case it is *unmixed*. Demonstrations of these facts can be found in [BS92, Stu94]; the essential property

$$MV(Q_1, \ldots, Q_n) = \sum \mathrm{Vol}(\sigma), \qquad \text{over all mixed cells } \sigma \text{ of a mixed subdivision of } Q,$$

relies on the multilinearity of the Mixed Volume from Definition 3.4. Mixed cells are parallelepipeds in n dimensions, hence their volume is the determinant of a matrix whose rows are the edges defining the cell.

Several algorithms exist for the calculation of Mixed Volumes. One of the first approaches [Emi93] computes the entire subdivision and simultaneously all n-fold Mixed Volumes required for a system of $n+1$ polynomials in n variables, but has to construct explicitly the lower envelope of $\widehat{Q}$. The method of [HS] takes advantage of repeated polytopes, while that of [VG94] exploits symmetry; general implementations have been described in [VVC94]. These algorithms have the same worst-case asymptotic complexity, as does our own algorithm defined below and analyzed in Section 8, namely singly-exponential in n. However, based on experimental results, our algorithm appears to be the most efficient to date, for the general problem.

The idea is to test, for every combination of n edges from the given polytopes, whether their Minkowski Sum lies on the lower envelope of $\widehat{Q}$. If so its volume is computed and added to the Mixed Volume. To *prune* the combinatorial search, we make use of

Proposition 6.2 Fix a lifting and let $J \subset \{1, \ldots, n\}$ be an index set such that e_j is an edge of Q_j for all $j \in J$. If the Minkowski Sum of lifted edges $\sum_{j \in J} \widehat{e}_j$ lies on the lower envelope of $\sum_{j \in J} \widehat{Q}_j$ then, for any subset of $L \subset J$, the Minkowski Sum $\sum_{l \in L} \widehat{e}_l$ lies on the lower envelope of the Minkowski Sum $\sum_{l \in L} \widehat{Q}_l$.

Our algorithm constructs n-tuples of edges from Q_i by starting with edge pairs and adding one edge from another polytope at a time. As each edge is added, the k-tuple for $2 \leq k \leq n$ is tested on whether it lies on the lower envelope of the corresponding lifted Minkowski Sum or not; only k-tuples that pass the test continue to be augmented. Further pruning is achieved by eliminating from the edge sets of polytopes not yet considered those edges that cannot extend the current k-tuple. This means that for index set J, we let $L = J \cup \{i\}$, where i ranges over $\{1, \ldots, n\} \setminus J$ and check the edge tuples corresponding to L. This process employs several "small" tests to decrease the number of "large" and expensive tests that must be ultimately performed.

Every test for a k-tuple of edges $e_{i_1}, \ldots, e_{i_k}$ is implemented as a Linear Programming problem. Let $\widehat{p}_i \in \mathbb{Q}^{n+1}$ be the midpoint of the lifted edge $\widehat{e}_i$ of $\widehat{Q}_i$ and let $\widehat{p} = \widehat{p}_{i_1} + \cdots + \widehat{p}_{i_k} \in \mathbb{Q}^{n+1}$ be an interior point of their Minkowski Sum. The test of interest is equivalent to asking whether $\widehat{p}$ lies on the lower envelope or not, which is formulated as follows:

$$\text{maximize } \ s \in \mathbb{R}: \qquad \widehat{p} - sz = \sum_{l \in \{i_1, \ldots, i_k\}} \sum_{j=1}^{m_l} \lambda_{lj} \widehat{v}_{lj}; \ \sum_{j=1}^{m_l} \lambda_{lj} = 1, \ \lambda_{lj} \geq 0, \ \forall\, l \in \{i_1, \ldots, i_k\}, j = 1 \ldots m_l;$$

where $z = (0, \ldots, 0, 1) \in \mathbb{Z}^{n+1}$, $\widehat{v}_{lj}$ are the vertices of $\widehat{Q}_l$ and m_l their cardinality, as before. Then $\widehat{p}$ lies on the lower envelope if and only if the maximal value of s is 0.

The Mixed Volume is invariant under permutation of the polytopes. For a given permutation the algorithm is:

Input: Convex polytopes $Q_1, \ldots, Q_n \subset \mathbb{R}^n$ with integer vertices.
Output: $MV(Q_1, \ldots, Q_n) \in \mathbb{Z}$.

Lift-and-Prune Algorithm:

1. Enumerate the edges of all polytopes $Q_1, \ldots, Q_n$ in sets $E_1, \ldots, E_n$.
2. Compute random lifting vectors $l_1, \ldots, l_n \in \mathbb{Q}^n$.
3. Initialize the Mixed Volume to 0.
4. For every edge e_1 in E_1 create current tuple (e_1); let $k = 1$.
5. Let i range from $k+1$ to n:

```
For every $e_i \in E_i$, if $\sum_{j=1}^{k} \widehat{e}_j + \widehat{e}_i$ does not lie on the lower envelope of
$\sum_{j=1}^{k} \widehat{Q}_j + \widehat{Q}_i$ then $e_i$ is removed from $E_i$.
```

6. Increment k.
7. If $k > n$
   then add the volume of the Minkowski Sum of $(e_1, \ldots, e_n)$ to the Mixed Volume; continue at step 4.
8. If $k \leq n$
   then add new edge e_k in E_k to the current tuple $(e_1, \ldots, e_{k-1})$.
   If the Minkowski Sum $\widehat{e}_1 + \cdots + \widehat{e}_k$ lies on the lower envelope of the Minkowski Sum $\widehat{Q}_1 + \cdots + \widehat{Q}_k$ then go to step 5;
   otherwise continue at step 4.

The Lift-and-Prune Algorithm is incremental in the sense that partial results are available at every stage of execution. This is particularly useful in long runs of the program, when a loose bound can be detected long before termination. In addition, the tree structure of the combinatorial search permits to restart the algorithm in the middle of a computation and allows for a distributed version.

The algorithm given above does not exploit the fact that Mixed Volume is invariant under permutation of the polytopes. In our implementation, we change the order of the polytopes, or rather their edge sets, in a dynamic fashion so that when the algorithm at step 8 picks a new edge set, it chooses the one with minimum cardinality.

Asymptotic complexity is analyzed in Section 8; here we discuss empirical results. Table 1 displays the running times of our implementation on the problem of cyclic n-roots for an ALPHA DECSTATION with an 80 MHz processor, rounded to the nearest integer number of seconds. This is a standard benchmark for algebraic geometry software, encountered in Fourier analysis. The exact bounds on the number of isolated complex roots were derived in a series of articles including [BF91a, BF91b, BF94] plus some Gröbner bases calculations on J. Backelin's program BERGMAN. For $n = 8$ there is a one-dimensional variety, which is found by GB, plus 1152 isolated roots; for $n = 9$ it is also known that the solution includes a positive-dimensional variety but for $n \geq 9$ the precise number of isolated roots is still unknown. The polynomial system is the following:

$$\begin{aligned}
x_1 + x_2 + \cdots + x_n &= 0 \\
x_1x_2 + x_2x_3 + \cdots + x_nx_1 &= 0 \\
&\cdots \\
x_1 \cdots x_{n-1} + x_2 \cdots x_n + \cdots + x_nx_1 \cdots x_{n-2} &= 0 \\
x_1x_2 \cdots x_n &= 1
\end{aligned}$$

We compare running times with the Gröbner Bases package GB by Faugère, since it outperformed BERGMAN. GB was executed on a 40 MHz SUN SPARC 10. All running times should be solely viewed as rough indications of the problem's intrinsic complexity and the algorithms' performances. The Mixed Volume computation constructs a monomial basis for the coordinate ring which allows us to solve the polynomial system [ER94], still Gröbner Bases provide significantly more information, including a tighter root count in general. For $n = 8$ the Gröbner Basis was computed over a field of a large prime characteristic, while for larger n the problem was infeasible [Fau94]. Exploiting the symmetry, the algorithm of [VG94] requires 16.4 seconds on an ALPHA DECSTATION 5000/240 for the cyclic 5-root problem.

Our implementation is available through anonymous `ftp` from `robotics.eecs.berkeley.edu`. Parallelization of our algorithm is straightforward and an implementation is being prepared on the CM-5 for public distribution.

Table 1: Mixed Volume algorithm performance for the cyclic n-roots problem on an ALPHA DECSTATION.

n	known #roots	Lift-Prune Algorithm Mixed Volume	time (80 MHz DECSTATION)	GB package #roots	time (40 MHz SPARC 10)
5	70	70	0s		
6	156	156	2s	156	3s
7	924	924	27s	924	6h 0m 4s
8	1152	2560	4m 19s	infinite	(char> 0) 3h 5m 12s
9	?	11016	40m 59s	-	-
10	?	35940	4h 50m 14s	-	-
11	?	184756	38h 26m 44s	-	-

7 Implementation

This section presents the overall algorithm for constructing Sparse Resultant matrices given $n+1$ supports $\mathcal{A}_i \subset \mathbb{Z}^n$. For computing Newton polytopes Q_i we may use the implementation, developed by the first author, of the the Beneath-Beyond algorithm in arbitrary dimension that handles degenerate configurations by the optimal perturbation scheme of [EC92]. However, since only the Convex Hull vertices and edges are needed, we choose to use Linear Programming on every support point to decide whether it is a vertex or not. Then the Mixed Volume algorithm of the previous section uses again Linear Programming on every pair of vertices to identify those that define edges on the Newton polytopes.

```
Input:   Supports A_1,...,A_{n+1} and direction vector u ∈ Q^n.
Output:  Maximal minor D of matrix M, such that D is a nontrivial multiple of the
         Sparse Resultant, or an indication that such a minor cannot be found.

Main Algorithm:
        1.  Compute the vertex sets of Newton polytopes Q_1,...,Q_{n+1}.
        2.  Use the Mayan Pyramid Algorithm to compute sets T_1,...,T_{n+1} ∈ Z^n
            and all u-distances.
        3.  Use the Lift-and-Prune Algorithm to compute Mixed Volumes
            MV(1),...,MV(n+1).
        4.  Use the Matrix Construction Algorithm to construct matrix M whose
            maximal minor D is a nontrivial multiple of R and return D if found.
            Otherwise, return with an indication that minor D cannot be found.
```

A useful feature is that, as the matrix construction is incremental, the nonsingularity test is also incremental. We have implemented an incremental algorithm for LU decomposition of rectangular matrices which, given a partially decomposed matrix, will attempt to continue and complete the decomposition. It uses partial pivoting and stops when a pivot and the subcolumn below it are all zero, thus calling for a larger matrix M. Arithmetic is carried out over a large finite field, which allows for efficient and exact arithmetic. This has the disadvantage that the constructed matrix M may be larger than what would be possible over the integers.

For Linear Programming, we use a publicly available implementation of the Simplex algorithm, since our goal is to release our software for distribution. It is evident that more efficient implementations of Linear Programming would significantly speed up our algorithm. As for the stability of the Simplex algorithm, it is not an issue because the inputs are all integers.

8 Asymptotic Complexity

Let s be the maximum number of points in any of the given supports $\mathcal{A}_i$, e the maximum number of Newton polytope edges, m the maximum number of Newton polytope vertices and v the maximum coordinate of any vertex, assuming that the Newton polytopes have been translated to lie in the first orthant and touch the coordinate axes. Let L_l be the maximum bit-size of a coordinate in any lifting vector l_i and $L_v = \log v$ be the maximum coordinate size of any Newton polytope vertex.

Each Mixed Volume computation requires $\mathcal{O}(e^n)$ Linear Programming problems, each of complexity $\mathcal{O}(n^7 m^6 (L_l + L_v))$. For Linear Programming any polynomial-time algorithm can be applied; Karmarkar's results [Kar84] are used in this section. This complexity dominates Step 1 of the algorithm where the edges are enumerated. From Section 6, $m^n / n^2 2^{L_l}$ is a constant, hence $L_l = \mathcal{O}(n \log m)$.

Theorem 8.1 The complexity of the Mixed Volume calculation is $e^{\mathcal{O}(n)} L_v$, where e is the maximum number of edges in any Newton polytope and L_v the logarithm of the maximum coordinate of any Newton polytope vertex or, equivalently, the maximum degree of any input polynomial in any variable. For most systems this bound is $e^{\mathcal{O}(n)}$.

This is asymptotically optimal because Mixed Volume generalized the Convex Hull Volume problem which is known to be #P-hard.

Computing one Convex Hull vertex set requires at most s Linear Programming tests for a total complexity in $(sn)^{\mathcal{O}(1)} L_v$. The cardinality of an integer point set is asymptotically bounded by the volume of their Convex Hull [Ehr67], hence s is bounded by the maximum $\mathrm{Vol}(Q_i)$ which is bounded by the maximum $\mathrm{Vol}(Q^i)$. In short, the total complexity of finding all Newton polytope vertex sets is

$$(n \mathrm{Vol}(Q^i))^{\mathcal{O}(1)} L_v \qquad \text{with } \mathrm{Vol}(Q^i) \text{ maximized over } i = 1, \ldots, n+1.$$

Each Linear Programming problem in the enumeration of integer point sets T_i has worst-case complexity $\mathcal{O}(n^6 m^5 L_v)$. An asymptotic upper bound on the number of Linear Programming problems is the cardinality of T_i or, by the relation of point cardinality to volume, $\mathrm{Vol}(Q^i)$. The complexity of Algorithm I is dominated by the LU decomposition, hence has complexity $\mathcal{O}(r^3)$, where r is the total number of rows in matrix M which is at most equal to the total number of points in all sets T_i. Hence the complexity of the Mayan Pyramid and the Matrix Construction Algorithms is bounded by the complexity above. This discussion leads to

Theorem 8.2 Assuming that a constant number of vectors u is used and the Mixed Volumes have been computed, the complexity of the Sparse Resultant algorithm is $(n \mathrm{Vol}(Q^i))^{\mathcal{O}(1)} L_v$, where L_v is as above and $\mathrm{Vol}(Q^i)$ is maximized over all $i = 1, \ldots, n+1$. For systems with every $MV(i) > 0$, the L_v factor can be ignored. Another bound is $(nv)^{\mathcal{O}(n)}$, where v is the maximum coordinate in any Newton polytope.

Proof L_v is bounded by the maximum of n and $\mathrm{Vol}(Q^i)$ provided that $\mathrm{Vol}(Q^i) > 0$ which follows from $MV(i) > 0$, $\forall\, i \in \{1, \ldots, n\}$. Otherwise, the input system has a very special structure and the Sparse Resultant is trivially 1. To avoid the dependence on $\mathrm{Vol}(Q^i)$, consider that $\mathrm{Vol}(Q_i) \leq v^n$ and that $\mathrm{Vol}(Q^i) \leq n^n \mathrm{Vol}(Q_i)$, where i is always chosen so as to maximize the respective volumes. □

An important special case is unmixed systems, as well as other systems which may behave similarly in the following sense. For unmixed systems $\mathrm{Vol}(Q^i) = \mathcal{O}(n^n \mathrm{Vol}(Q_1))$, $MV(i) = n!\, \mathrm{Vol}(Q_1)$

and $\deg R = (n+1)MV(i)$, where $\deg R$ is the total degree of the Sparse Resultant. This implies $\mathrm{Vol}(Q^i) = \mathcal{O}(n^n \deg R/(n+1)!) = \mathcal{O}(2^n \deg R)$ by Stirling's approximation. *Well-behaved* systems are exactly those for which these relations hold and include those mixed systems whose Newton polytopes sufficiently resemble each other.

Corollary 8.3 Given the Mixed Volumes, for a constant number of u vectors and for well-behaved systems, including unmixed ones, the total complexity is $2^{\mathcal{O}(n)}(\deg R)^{\mathcal{O}(1)} L_v$. L_v is asymptotically dominated and can be ignored, unless $\mathrm{Vol}(Q_i) = 0$ for all i.

We expect to formalize the notion of well-behaved systems, extend the latter bounds to arbitrary mixed systems and derive tighter bounds for the Mixed Volume problem. Empirical results show that the above bounds are overly pessimistic for the complexity of our Sparse Resultant algorithm.

9 Multihomogeneous Systems

We concentrate on unmixed homogeneous 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. We call the system of type $(l_1, \ldots, l_r; d_1, \ldots, d_r)$ with the number of equations being $n+1$ where $n = \sum_{k=1}^r l_k$; there is no relation between these l_i and the lifting functions of previous sections. 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 Minkowski Sum of r l_k-dimensional simplices, each on a disjoint 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 n-fold Minkowski Sum Q^i is the same for any $i \in \{1, \ldots, n+1\}$ and equal to integer polytope $P \subset \mathbb{R}^n$ which is simply a copy of the unique input Newton polytope scaled by n i.e.

$$Q_1 = \cdots = Q_{n+1} = \sum_{k=1}^{r} d_k S_{l_k}, \; P = \sum_{k=1}^{r} n d_k S_{l_k} \quad \subset \mathbb{R}^n.$$

Both summations express Minkowski Addition of lower-dimensional polytopes in complementary subspaces, such that their Sum is a full-dimensional polytope.

Sturmfels and Zelevinsky [SZ94] studied in particular the subclass of systems for which, for every $k \in \{1, \ldots, r\}$, we have $l_k = 1$ or $d_k = 1$. They showed that every such system has a number of Sylvester-type formulae for its Sparse Resultant, called *multigraded resultants*, one for every permutation π of the indices $\{1, \ldots, r\}$. For this resultant matrix, all supports $\mathcal{B}_i$ are identical, of cardinality equal to the unique n-fold Mixed Volume. Let $B \subset \mathbb{R}^n$ be the convex hull of $\mathcal{B}_i$. Matrix M is defined by setting

$$B = \sum_{k=1}^{r} m_k S_{l_k} \subset \mathbb{R}^n \qquad \text{where} \;\; m_k = (d_k - 1) l_k + d_k \sum_{j:\pi(j)<\pi(k)} l_j, \qquad k \in \{1, \ldots, r\}.$$

Lemma 9.1 Partition the n coordinates of vector $u \in \mathbb{Q}^n$ into r groups following the partition of variables and set every coordinate in the k-th group equal to $(nd_k - m_k)/l_k \in \mathbb{Q}$. 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:

$$0 \leq c \leq nd_k \qquad \text{and} \qquad 0 \leq c + \frac{nd_k - m_k}{l_k} \leq nd_k$$

which is equivalent to $0 \leq c \leq m_k$. 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:

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

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

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 9.2 Given a multihomogeneous system of type $(l_1, \ldots, l_r; d_1, \ldots, d_r)$ such that $l_k = 1$ or $d_k = 1$ for $k = 1, \ldots, r$, we define $u \in \mathbb{Q}^n$ with the k-th group of coordinates equal to $(nd_k - m_k)/l_k$. 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 $B \cap \mathbb{Z}^n$, hence they are exactly those required 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. $\square$

We have been able to produce all possible Sylvester-type formulae for various multihomogeneous examples with $l_k = 1$ or $d_k = 1$ for all k. Further, for systems that do not fall within this class we have used u defined similarly and obtained near-optimal resultant matrices. Experimental results and preliminary running times on a 40 MHz Sun Sparc 10 are displayed in Table 2, rounded to the nearest integer number of seconds. $\deg R$ and $\deg D$ respectively indicate the total degree of the Sparse Resultant and the size of the maximal minor D that our algorithm constructs.

For the second class of systems for which there exists k such that $l_k > 1$ and $d_k > 1$, we have used the same recipe as above to calculate m_i and u, then have perturbed the latter to obtain the results shown. For type $(2, 1; 2, 1)$ we used $\pi = (2, 1)$; the greedy implementation of the Sparse Resultant algorithm in [CE93] produces a matrix of size 103. Similarly, for type $(2, 1; 2, 2)$ the smallest matrix is obtained for $\pi = (2, 1)$.

The last section of the table refers to specific applications, namely the motion-from-points problem in vision and the forward kinematics of the Stewart-platform parallel robot. Neither is exactly a multihomogeneous system of the shown type but both approximate this structure. The kinematics problem is very sparse, which significantly lowers the Sparse Resultant degree; in actuality, we have used a perturbation of the shown u to produce a 745×745 matrix that can be reduced to a 372×372 resultant matrix which is then manipulated in order to solve the corresponding algebraic system. More results on system solving using resultant matrices are reported in [ER94], where further references can also be found for these applications.

10 Future Work

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 to sparse polynomials. The determination

Table 2: Sparse Resultant algorithm performance on a SUN SPARC 10.

type	application	vector $u \in \mathbb{Z}^n$	deg R	deg D	CPU time
$(2,1,1;1,2,2)$		$(2,2,3,1)$	240	240	42s
$(1,1,1,1;2,2,1,1)$		$(7,5,2,1)$	480	480	1m 0s
$(1,1,1,1;3,3,1,1)$		$(10,7,2,1)$	1080	1080	2m 11s
$(1,1,1,1;3,3,2,1)$		$(10,7,3,1)$	2160	2160	4m 3s
$(1,1,1,1;3,3,3,1)$		$(10,7,4,1)$	3240	3240	6m 29s
$(2,1;2,1)$		$\sim (1,1,3)$	48	52	0s
$(2,1;2,2)$		$\sim (1,1,5)$	96	104	6s
$(2,1,1;2,1,1)$		$\sim (3,3,2,1)$	240	295	2m 43s
$(2,1,1;2,2,1)$		$\sim (3,3,3,1)$	480	592	18m 56s
$\sim (2,3,1,1)$	vision	$(5,5,2,2,2)$	60	60	24s
$\sim (3,4,2,2)$	kinematics	$\sim (11,11,11,3,3,3,3)$	246	745	35m 0s

of favorable vectors u for different classes of systems would automate the process of constructing compact matrix formulae. Lastly, a more careful complexity analysis for Mixed Volumes and for the entire algorithm on mixed polynomial systems might yield tighter asymptotic bounds.

An important merit of this work is its practical application in solving algebraic systems in kinematics and vision [ER94], modeling [BGW88] as well as computational biology [PC94]. In this respect, one open question is the transformation of arbitrary systems to an equivalent form that is amenable to Sparse Elimination and in particular to the computation of Mixed Volumes and Sparse Resultants.

References

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

[BF91a] J. Backelin and R. Fröberg. How we Proved that there are exactly 924 Cyclic 7-Roots. In *Proc. ACM Intern. Symp. on Symbolic and Algebr. Computation*, pages 103–111, Bonn, 1991.

[BF91b] G. Björck and R. Fröberg. A Faster Way to Count the Solutions of Inhomogeneous Systems of Algebraic Equations, with Applications to Cyclic n-roots. *J. Symbolic Computation*, 12:329–336, 1991.

[BF94] G. Björck and R. Fröberg. Methods to "Divide out" certain Solutions from Systems of Algebraic Equations, Applied to Find all Cyclic 8-Roots. Manuscript, Dept. of Math., Stockholm University, 1994.

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

[BS92] L.J. Billera and B. Sturmfels. Fiber Polytopes. *Annals of Math.*, 135:527–549, 1992.

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

[Cay48] A. Cayley. On the Theory of Elimination. *Cambridge and Dublin Math. J.*, 3:116–120, 1848. Reprinted in "Collected Papers", Vol. 1, No. 59, pp. 370–374, Cambridge Univ. Press, Cambridge, 1989.

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

[EC92] I. Emiris and J. Canny. An efficient approach to removing geometric degeneracies. In *Proc. 8th ACM Symp. on Computational Geometry*, pages 74–82, 1992.

[Ehr67] E. Ehrart. Sur un problème de géométrie diophantienne, I. Polyèdres et réseaux. *J. Reine Angew. Math.*, 226:1–29, 1967.

[Emi93] I. Emiris. An efficient computation of mixed volume. Technical Report 734, Computer Science Division, U.C. Berkeley, Berkeley, CA, 1993.

[ER94] I.Z. Emiris and A. Rege. Monomial Bases and Solution of Polynomial Systems. In *Proc. ACM Intern. Symp. on Symbolic and Algebr. Computation*, Oxford, July 1994. To Appear.

[Fau94] J.-C. Faugère, 1994. Personal Communication.

[GKZ91] I.M. Gelfand, M.M. Kapranov, and A.V. Zelevinsky. Discriminants of Polynomials in Several Variables and Triangulations of Newton Polytopes. *Leningrad Math. J.*, 2(3):449–505, 1991. (Translated from *Algebra i Analiz* **2**, 1990, 1–62).

[GKZ94] I.M. Gelfand, M.M. Kapranov, and A.V. Zelevinsky. *Discriminants and Resultants.* Birkhäuser, Boston, 1994.

[Grü67] B. Grünbaum. *Convex Polytopes.* Wiley-Interscience, New York, 1967.

[HS] B. Huber and B. Sturmfels. A polyhedral method for solving sparse polynomial systems. *Math. Comp.* To appear. A preliminary version presented at the "Workshop on Real Algebraic Geometry", 8/92.

[Hur13] A. Hurwitz. Über die Trägheitsformen Eines Algebraischen Moduls. *Annali di Mat.*, Tomo XX(Ser. III):113–151, 1913.

[Kar84] N. Karmarkar. A New Polynomial-Time Algorithm for Linear Programming. *Combinatorica*, 4:373–395, 1984.

[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.

[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.

[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, 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.

[PC94] D. Parsons and J. Canny. Geometric problems in molecular biology and robotics. In *Proceedings of the Second International Conference on Intelligent Systems for Molecular Biology*, Palo Alto, CA, August 1994. To Appear.

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

[Ren92] J. Renegar. On the Computational Complexity of the First-Order Theory of the Reals, parts I, II, III. *J. Symbolic Computation*, 13(3):255–352, 1992.

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

[Stu94] B. Sturmfels. On the Newton Polytope of the Resultant. *J. of Algebr. Combinatorics*, 3:207–236, 1994.

[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.

[VG94] J. Verschelde and K. Gatermann. Symmetric Newton Polytopes for Solving Sparse Polynomial Systems. Technical Report 3, Konrad-Zuse-Zentrum für Informationstechnik Berlin, 1994.

[VVC94] J. Verschelde, P. Verlinden, and R. Cools. Homotopies Exploiting Newton Polytopes for Solving Sparse Polynomial Systems. *SIAM J. Numerical Analysis*, 31(3):915–930, 1994.

[WZ92] J. Weyman and A. Zelevinsky. Determinantal formulas for multigraded resultants. Manuscript, 1992.