Efficient Incremental Algorithms for the Sparse Resultant and the Mixed Volume

IOANNIS Z. EMIRIS[‡] AND JOHN F. CANNY[§]

[‡]*Projet SAFIR, I.N.R.I.A., B.P. 93, 06902 Sophia-Antipolis, France.*
emiris@sophia.inria.fr

[§]*Computer Science Division, University of California, Berkeley CA 94720, USA.*
jfc@cs.Berkeley.edu

(Received 27 July 1995)

We propose a new and efficient algorithm for computing the sparse resultant of a system of $n+1$ polynomial equations in n unknowns. This algorithm produces a matrix whose entries are coefficients of the given polynomials and is typically smaller than the matrices obtained by previous approaches. The matrix determinant is a nontrivial multiple of the sparse resultant from which the sparse resultant itself can be recovered. The algorithm is incremental in the sense that successively larger matrices are constructed until one is found with the above properties. For multigraded systems, the new algorithm produces optimal matrices, i.e., expresses 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. This computation provides an upper bound on the number of common isolated roots. A publicly available implementation of the algorithm is presented and empirical results are reported which suggest that it is the fastest mixed volume code 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 essentially eliminate the input variables, so they are also called *eliminants*. They also serve in solving systems of equations, for instance by reducing root-finding to an eigenproblem (Auzinger and Stetter, 1988) or by means of the u-resultant construction of Renegar (1992).

This article continues work by Canny and Emiris (1993) for constructing matrix formulae for the sparse resultant. As in that article, we build resultant matrices whose entries are either zero or coefficients of the given polynomials, so the new algorithm can be considered as a generalization

[‡] Work partially conducted while on visit in 1992 at the Mathematics Department of the University of Nice and Projet SAFIR at INRIA Sophia-Antipolis, France. Also supported, as a graduate student at U.C. Berkeley, by a Packard Foundation Fellowship and by NSF PYI Grant IRI-8958577.

[§] Supported by a Packard Foundation Fellowship and by NSF PYI Grant IRI-8958577.

of Sylvester's approach. However, we take a different tack in order to reduce the matrix size and to obtain, for multigraded systems, optimal formulae. This class of systems includes all systems for which an optimal formula *provably* exists. For two polynomials the new algorithm returns Sylvester's matrix, whereas for a linear system it yields the coefficient matrix. The original idea behind the present approach first appeared in (Emiris and Canny, 1993). Under reasonable assumptions, the total worst-case complexity is bounded, in theorem 7.5, by

$$e^{\mathcal{O}(n)}(\deg R)^3: \qquad e < 2.7183,\ \deg R \text{ is the total degree of the sparse resultant.}$$

We also report on experimental results for multihomogeneous systems and the cyclic n-roots benchmark. An important aspect of the algorithm is that it readily extends to systems of more than $n+1$ polynomials in n variables.

A subproblem in our approach is the computation of mixed volume which is also of independent interest since it bounds the degree of a zero-dimensional variety. We present an algorithm with worst-case complexity, under certain mild assumptions,

$$m^{\mathcal{O}(n)}: \qquad m \leq \text{ maximum number of monomials per polynomial.}$$

This bound follows from theorem 6.4, where m bounds the number of extremal monomials or Newton polytope vertices; these concepts are defined in section 3. We also sketch our publicly available implementation of the mixed volume algorithm and report on experimental results; some of these results first appeared in (Emiris and Rege, 1994).

Our motivation stems from the fact that resultant-based methods currently offer the most efficient solution to certain problems in a variety of areas ranging from robotics (Canny, 1988) to modeling (Bajaj, Garrity and Warren, 1988). To illustrate this claim, the remaining of this section discusses some applications.

A concrete example from robot kinematics is the *inverse kinematics* problem for a robot with six rotational joints (6R) and, therefore, six degrees of freedom. It is solved using a customized resultant in 11 milliseconds on a 34 MIPS IBM RS/6000 (Manocha and Canny, 1992). This problem consists in finding the angle at every joint in order to attain a given final position, while the link lengths are fixed. Previous homotopy methods had running times unacceptable of real-time industrial manipulators.

Implicitization of 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 and 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 custom-made resultants have been shown to run faster by a factor of at least 10^3 compared to Gröbner bases and the Ritt-Wu method (Manocha and Canny, 1992).

One limitation of the solutions referred to above is the lack of a general method to attack arbitrary algebraic systems. Since several classes of scientific and engineering problems are expected to reduce to algebraic systems with sparse structure, in a sense to be formalized below, we would like to have a sparse resultant for every problem, and this calls for a general algorithm to construct them. This is the main question studied in this article. To further motivate it, we list some concrete applications of the algorithms and implementations described here to specific problems in vision, robot kinematics and structural biology.

An example of a vision problem is the computation of the camera displacement in a static environment. We are given the coordinates of 5 points in the environment as seen by the two successive positions of the camera. Using the general polynomial system solver described in (Emiris, 1995), which includes the implementations of this article, this problem was solved to at least 5 accurate digits, which is satisfactory for vision applications. The running time of the online solver was 0.2 seconds and 1 second, respectively, on the DEC ALPHA 3300 and the SUN SPARC 20/61 of table 1, for two instances with input parameters of varying genericity.

Another problem solved by Emiris (1995) using the implementations of this article dealt with the kinematics of molecules or, equivalently, of mechanisms. Given a ring with 6 rigid links and prescribing the angle between successive links, each possible configuration is defined by the dihedral angles around the links. Three different instances were solved with absolute error bounded by 10^{-5}, in 0.2 to 0.4 seconds on the SUN SPARC 20/61 of table 1.

The article is organized as follows. The next section puts the new approach into perspective by outlining previous work in the same area. Section 3 provides all definitions. Section 4 presents our approach for the sparse resultant and section 5 defines the algorithm for building sparse resultant matrices. Section 6 discusses our algorithm for computing mixed volumes, sketches its implementation and analyzes the worst-case and average-case complexities. The overall algorithm for the sparse resultant, its implementation and its asymptotic complexity analysis are presented in section 7. Section 8 shows that the algorithm constructs optimal sparse resultant matrices for a class of multihomogeneous systems and lists experimental results for multihomogeneous systems in general. Section 9 examines the practical complexity of our algorithms for the standard benchmark family of cyclic n-roots, and compares them to Gröbner bases and another mixed volume implementation. The paper concludes with a summary and directions for future work.

2. Related Work

This section examines previous work in elimination theory. 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 in the sense that potentially all terms of a given total degree can appear. The simplest system is that of two homogeneous polynomials in two unknowns. This case was studied by Sylvester (1853) who defined the resultant as the determinant of a matrix in the polynomial coefficients. The *multivariate* resultant for a system of n homogeneous polynomials in n variables can be defined in several alternative ways. Cayley (1848) defined it via a series of n divisions of determinants, Macaulay (1902) as the quotient of a determinant divided by one of its minors, while Hurwitz (1913) expressed it as the Greatest Common Divisor (GCD) of n *inertia forms*; see also (van der Waerden, 1950). In all cases, the nonzero entries of the matrices are coefficients of the given polynomials. Various more recent algorithms exist to construct this resultant (Lazard, 1981, Canny, 1988, Renegar, 1992).

The sparse resultant was defined following the study of generalized hypergeometric functions and $\mathcal{A}$-discriminants (Gelfand, Kapranov and Zelevinsky, 1991 and 1994). The exact notion of sparseness is formalized and compared to the dense case in the next section. The first constructive methods for computing and evaluating the sparse resultant were proposed by Sturmfels (1993), the most efficient having complexity super-polynomial in the degree of the resultant and exponential in n with a quadratic exponent.

Canny and Emiris (1993) proposed a general algorithm for computing the sparse resultant of $n+1$ non-homogeneous polynomials in n variables. The worst-case asymptotic cost of this algorithm, under mild assumptions, is polynomial in the resultant's degree and simply exponential in n (Emiris, 1994). This was the first efficient algorithm for the general case in the sense that the lower bound for

computing the resultant is polynomial in its degree and exponential in n. The algorithm constructs a square matrix whose entries are either zero or coefficients of the given polynomials. The determinant of this matrix is not identically zero and is divisible by the sparse resultant.

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, knowing the resultant matrix suffices. For two univariate polynomials the algorithm yields Sylvester's matrix. 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 (1994). A *greedy* implementation has been written in MAPLE by the second author and P. Pedersen (1993) and produces a matrix whose size is at most that given by the original algorithm. The latter is publicly available by `ftp` on `robotics.eecs.Berkeley.edu` in directory `pub/SparseResultant`.

An integral part of the *theory of sparse elimination* is Bernstein's bound on the degree of the toric variety of a square polynomial system. This bound has been extended to count isolated roots even when the variety has positive dimension and also to count common affine roots. This is an active area of research, briefly outlined in the following section (Bernstein, 1975, Fulton, 1993, Rojas, 1994, Li and Wang, 1994, Huber and Sturmfels, 1995). A related question of great interest is to count real roots (Sturmfels, 1992). The calculation of Bernstein's bound requires the computation of the *mixed volume* of the given polynomials. For an overview of algorithms and implementations refer to sections 6 and 9.

Finally we recall that alternative notions of sparseness exist, which may also lead to tight bounds for polynomial systems. One example is the theory of *fewnomials* pioneered by Khovanskii (1991).

There exist other alternatives to elimination and the more particular problem of system solving besides resultants. Gröbner bases provide a general tool for studying arbitrary polynomial ideals, eliminating variables as well as finding the common roots of a system (Buchberger, 1985). They constitute a popular and very general approach and several implementations exist. The main limitations of the method are the large coefficient size and the degree of the basis polynomials. For zero-dimensional affine varieties a tight upper bound on the complexity is $d^{\mathcal{O}(n)}$ (Lakshman Y.N., 1990), while the bit complexity is $d^{\mathcal{O}(n^2)}$ where d is an upper bound on the degree of the input polynomials. Although Gröbner bases in practice perform significantly better than the worst-case bounds, there are no complexity bounds that depend on the Newton polytopes or the mixed volumes.

Another approach is the Ritt-Wu method, whose complexity has been recently shown to be exponential in the number of variables and polynomial in the polynomial degrees (Mishra, 1993, ch. 5).

The principal numerical techniques are continuation methods, which are typically fast but offer little control over the numerical error. Homotopies that follow the optimal number of paths have been proposed for multihomogeneous systems (Morgan, Sommese and Wampler, 1993). A promising development in solving very large polynomial systems comes from *sparse homotopies* which exploit the structure of polynomials as modeled by Newton polytopes (Huber and Sturmfels, 1992, Verschelde, Verlinden and Cools, 1994, Verschelde, Gatermann and Cools, 1995).

3. Preliminaries

This section provides a short introduction to sparse elimination theory. 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 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 $\mathbb{Q}(\{\, c_i \,|\, i \in \{1, \ldots, n\}\,\})$ 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 $\mathbb{C}^* = \mathbb{C} \setminus \{0\}$.

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. $\mathcal{A}_i$ is called the *support* of f_i. Then

$$f_i = \sum_{j=1}^{s_i} c_{ij} x^{a_{ij}}, \qquad c_{ij} \neq 0, \qquad \forall j \in \{1, 2, \ldots, s_i\}, \qquad \forall i \in \{1, 2, \ldots, n+1\}, \tag{3.1}$$

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

We now introduce certain concepts from combinatorial geometry, which can be found in (Grünbaum, 1967, Schneider, 1993).

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

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 the point set*

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

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 $a + b$, where a and b are *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*

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

$A - B$ 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 $B + (A - B) \overset{\subset}{\neq} A$. However, when A is itself a Minkowski sum $B + C$, then $(B + C) - B = C$, for any convex polytope C. We also state the identities $A - (B + C) = (A - B) - C$ and $(A + U) - B = (A - B) + U$, where U is a one-dimensional polytope.

For any polytope $A \subset \mathbb{R}^n$, let $\mathrm{Vol}(A)$ denote the Lebesgue measure of A in n-dimensional Euclidean space. This function assigns the unit volume to the unit cube.

Definition 3.4. *Given convex polytopes $A_1, \ldots, A_n \subset \mathbb{R}^n$, there is a real-valued function $MV(A_1, \ldots, A_n)$, unique up to multiplication by a scalar, called the* mixed volume *of $A_1, \ldots, A_n$, which is multilinear with respect to Minkowski addition and scalar multiplication, i.e., for any nonnegative $\mu, \rho \in \mathbb{R}_{\geq 0}$*

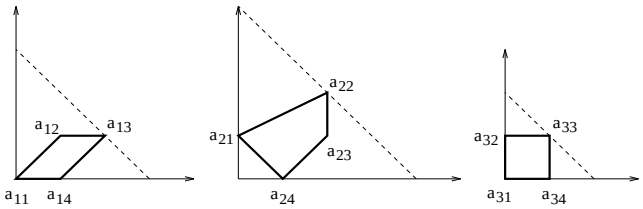

Figure 1. Newton polytopes for example 3.11.

and any 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

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

An equivalent definition is based on the fact that, for nonnegative $\lambda_1, \ldots, \lambda_n \in \mathbb{R}$ and convex polytopes $A_1, \ldots, A_n$, the expression $\mathrm{Vol}(\lambda_1 A_1 + \cdots + \lambda_n A_n)$ expands to a homogeneous polynomial in $\lambda_1, \ldots, \lambda_n$ (Grünbaum, 1967, sect. 15.1).

Definition 3.5. *For nonnegative* $\lambda_1, \ldots, \lambda_n \in \mathbb{R}_{\geq 0}$ *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* $\mathrm{Vol}(\lambda_1 A_1 + \cdots + \lambda_n A_n)$.

Notice that this definition differs from the classic one (Grünbaum, 1967) by a factor of $n!$. These and other basic properties of the mixed volume are demonstrated in (Grünbaum, 1967, Betke, 1992, Schneider, 1993), whereas a more advanced treatment can be found in (Burago and Zalgaller, 1988).

We are now ready to state Bernstein's theorem, the cornerstone of sparse elimination theory.

Theorem 3.6. (Bernstein, 1975) *For polynomials* $f_1, \ldots, f_n \in K[x, x^{-1}]$ *with Newton polytopes* $Q_1, \ldots, Q_n \subset \mathbb{R}^n$ *and generic coefficients, the number of common solutions in* $(\mathbb{C}^*)^n$, *counting multiplicities, 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 (1975) and has been alternatively proven by Khovanskii (1978) . The above bound is guaranteed to be exact for *generic* polynomials. Since its statement the conditions for exactness have been significantly weakened (Canny and Rojas, 1991, Rojas, 1994).

Bernstein's bound is at most as high as Bézout's bound and usually significantly tighter for systems encountered in engineering applications. The two bounds are equal when each Newton polytope is a scaled copy of the n-dimensional unit simplex with vertex set

$$\{(0, \ldots, 0), (1, 0, \ldots, 0), \ldots, (0, \ldots, 0, 1)\}.$$

The scalar factor for every simplex equals the total degree of the respective polynomial. This is depicted in figure 1, where the simplices are drawn with dashed lines for the three polynomials of example 3.11.

The Bernstein bound holds for varieties with positive dimension over arbitrary fields. Consider the intersection multiplicity of the hypersurfaces defined by the given polynomials at an isolated common root. Recall that this is a positive integer and is equal to unity exactly when the hypersurfaces meet transversally.

THEOREM 3.7. (Fulton, 1993, sect. 5.5) *Given are polynomials* $f_1, \ldots, f_n \in K[x, x^{-1}]$ *with Newton polytopes* $Q_1, \ldots, Q_n$, *where* K *is an arbitrary field and* $\overline{K}$ *is its algebraic closure. For any isolated common zero* $\alpha \in (\overline{K}^*)^n$, *let* $i(\alpha)$ *denote the intersection multiplicity at this point. Then* $\sum_\alpha i(\alpha) \leq MV(Q_1, \ldots, Q_n)$, *where the sum ranges over all isolated roots.*

Moreover, the bound has recently been extended to count all isolated roots. By abuse of language we refer to the same function when we speak of the mixed volume of a system of polynomials, the mixed volume of the respective supports or the mixed volume of the respective Newton polytopes.

THEOREM 3.8. (Li and Wang, 1994) *For polynomials* $f_1, \ldots, f_n \in \mathbb{C}[x, x^{-1}]$ *with supports* $\mathcal{A}_1, \ldots, \mathcal{A}_n$ *the number of common isolated zeros in* $\mathbb{C}^n$, *counting multiplicities, is either infinite or bounded by* $MV(\mathcal{A}_1 \cup \{0\}, \ldots, \mathcal{A}_n \cup \{0\})$.

In the rest of this article we concentrate for simplicity on polynomials over $\mathbb{C}$.

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*. We shall see below that the degree of the sparse resultant depends on the mixed volume of the n-dimensional subsystems, whereas the degree of the classical resultant depends on the Bézout bound.

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 can be identified with the projective space $\mathbb{P}_K^{m_i-1}$ or, simply, $\mathbb{P}^{m_i-1}$. Then the input system (3.1) 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 above product of projective spaces. It is proven in (Pedersen and Sturmfels, 1993) that Z is an irreducible algebraic variety of positive codimension.

DEFINITION 3.9. *The* sparse resultant $R = R(\mathcal{A}_1, \ldots, \mathcal{A}_{n+1})$ *of system (3.1) is a polynomial in* $\mathbb{Z}[c]$. *If* $codim(Z) = 1$ *then* $R(\mathcal{A}_1, \ldots, \mathcal{A}_{n+1})$ *is the defining irreducible polynomial of the hypersurface* Z. *If* $codim(Z) > 1$ *then* $R(\mathcal{A}_1, \ldots, \mathcal{A}_{n+1}) = 1$.

Although the sparse resultant is defined as a condition on the existence of toric roots, it can be used to recover all roots of nontrivial systems, including those with zero coordinates, under certain mild conditions on the variety (Emiris, 1994b).

Let $\deg_{f_i} R$ denote the total degree of the resultant R in the coefficients of polynomial f_i and let

$$MV_{-i} = MV(Q_1, \ldots, Q_{i-1}, Q_{i+1}, \ldots, Q_{n+1}), \qquad \forall i \in \{1, \ldots, n+1\}.$$

A consequence of Bernstein's theorem is

THEOREM 3.10. (Pedersen and Sturmfels, 1993) *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 classical multivariate resultant; the two coincide when all Newton polytopes are n-simplices scaled by the total degrees of the respective polynomials. However,

the degree and, hence, the complexity of computing the classical multivariate resultant depends on the Bézout bound as follows: the degree of the resultant in the coefficients of f_i is $\prod_{j\neq i} d_j$, where d_j is the total degree of f_j.

EXAMPLE 3.11. Here is a system of 3 polynomials in 2 unknowns:

$$\begin{aligned} f_1 &= c_{11} + c_{12}xy + c_{13}x^2y + c_{14}x, \\ f_2 &= c_{21}y + c_{22}x^2y^2 + c_{23}x^2y + c_{24}x, \\ f_3 &= c_{31} + c_{32}y + c_{33}xy + c_{34}x, \end{aligned} \tag{3.2}$$

with Newton polytopes shown in figure 1. For each subsystem of two polynomials we may compute the mixed volume. The three twofold mixed volumes are

$$MV_{-1} = 4, \qquad MV_{-2} = 3, \qquad MV_{-3} = 4.$$

By theorem 3.10 the total degree of the sparse resultant is $4 + 3 + 4 = 11$.

To motivate the present approach, we compare the various resultant algorithms on this example. The matrix constructed by the algorithm of (Canny and Emiris, 1993) has size 15, whereas its greedy version (Canny and Pedersen, 1993) 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 figure 1.

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. Our construction uses certain facts from ideal theory and combinatorial geometry; the interested reader may consult, respectively, (Cox, Little and O'Shea, 1992) and the previous section and the citations thereof.

To exploit sparseness and achieve the degree bounds of theorem 3.10 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 $\mathcal{A}_i$ (Sturmfels, 1994). Suppose this sublattice has rank n and is thus identified with $\mathbb{Z}^n$. In what follows, it is assumed that this has already been done by means of the *Smith normal form*; for computing this form see (Hafner and McCurley, 1991).

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

$$\begin{aligned} M \;:\; \mathcal{P}(\mathcal{B}_1) \times \cdots \times \mathcal{P}(\mathcal{B}_{n+1}) &\rightarrow \mathcal{P}(\bigcup_{i=1}^{n+1} \mathcal{B}_i + \mathcal{A}_i), \\ M \;:\; (g_1, \ldots, g_{n+1}) &\mapsto g = \sum_{i=1}^{n+1} g_i f_i, \end{aligned} \tag{4.1}$$

where addition between supports stands for Minkowski addition. The matrix we wish to build is precisely the matrix of this transformation and to define it fully we specify supports $\mathcal{B}_i$ at the end of this section.

We shall abuse notation and denote by M the matrix representing the linear transformation. 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 the 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 the polynomial $x^b f_i$ so that the coefficient of the monomial x^q appears in the column indexed by x^q, where $b \in \mathcal{B}_i = \mathrm{supp}(g_i)$, $q \in \mathrm{supp}(g)$. 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 rank deficient.*

PROOF. If a common solution ξ exists, then it is a solution for all g in the image of the linear transformation M. This implies that the image of M cannot contain any monomials, because the monomial value at a toric ξ cannot be zero. Therefore, the image of M is a proper subset of the range. Since M is not a surjective transformation, it follows that matrix M does not have full rank. □

The number of rows equals the sum of the cardinalities of the supports $\mathcal{B}_i$, while the number of columns equals the cardinality of $\mathrm{supp}(g)$. 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 lemma 4.1, the rank of M is strictly 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. A maximal minor of M is the determinant of a square submatrix of M with the same number of columns as M. Thus, any maximal minor D is zero on Z_0. Hence D 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 is the vector of coefficients of f_i. □

It is possible that every maximal minor D vanishes, in which case the theorem still holds but the constructed matrix is of little use. In our main algorithm, below, we shall explicitly enforce the existence of a non-vanishing maximal minor D.

COROLLARY 4.3. *Let $\deg_{f_i} D$ denote the degree of D in the coefficients of polynomial f_i. If R divides D and $D \neq 0$ then $\deg_{f_i} D \geq MV_{-i}$ for all i.*

PROOF. From theorems 3.10 and 4.2. □

We now specify the construction of the 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 (Canny and Emiris, 1993). 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 vector

$$v \in \mathbb{Q}^n \setminus \{0, \ldots, 0\}$$

we define a family of one-dimensional polytopes $V \subset \mathbb{R}^n$, each being the convex hull of the origin and of a point $\beta u \in \mathbb{R}^n$, where β is a nonzero real variable. In other words,

$$|\beta| = \text{length}(V) \in \mathbb{R} \setminus \{0\}.$$

The sign of β determines the direction in which V lies and its magnitude determines the length of V. For a fixed V we define

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

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

An interesting aspect of this approach is that it readily extends to the case of simultaneously eliminating n variables from more than $n+1$ polynomials. In this case the resultant is not defined but some of its properties are still valid. In particular, the non-vanishing minor of the constructed matrix provides a solvability condition for the system and reduces its solution to a linear algebra problem, just as the resultant matrix does for the system of $n+1$ polynomials.

5. Matrix Construction

This section presents the algorithm for constructing the sets $\mathcal{B}_i$ and the resultant matrix M satisfying the requirements set in the previous section.

DEFINITION 5.1. *Given a convex polytope $A \subset \mathbb{R}^n$ and a vector $v \in \mathbb{Q}^n \setminus \{(0,\ldots,0\}$, we define the* v-distance *of any point $p \in A \cap \mathbb{Z}^n$ to be the maximum nonnegative $s \in \mathbb{R}_{\geq 0}$ such that $p + sv \in A$. In other words, it is the distance of p from the boundary of A in the direction v.*

Integer points on the boundary of the polytope A which are extremal with respect to vector v have zero v-distance. Figure 2 shows different subsets of T_2 for system (3.2) with respect to v-distance, as explained in the running example 3.11 at the end of this section. An equivalent definition of supports $\mathcal{B}_i$ is by ordering $Q_{-i} \cap \mathbb{Z}^n$ by v-distance, then selecting the points whose v-distance exceeds some bound. The vector v here and in the definition of V at the end of the previous section is the same and β can be used as the bound on v-distance. This is formalized in the following proposition.

PROPOSITION 5.2. *For a convex polytope A, one-dimensional polytope $V \in \mathbb{R}^n$ and vector $v \in \mathbb{Q}^n$ such that v lies in the interior of V,*

$$(A - V) \cap \mathbb{Z}^n = \{a \in A \cap \mathbb{Z}^n \mid v\text{-}distance(a) \geq \beta = length(V)\}.$$

We now turn to the question of enumerating all integer lattice points T_i in the n-fold Minkowski sum Q_{-i}, for $i \in \{1,\ldots,n+1\}$, together with their v-distances for some $v \in \mathbb{Q}^n$. We propose a recursive algorithm, called the Mayan Pyramid algorithm, which computes, at its k-th stage, the range of values for the k-th coordinate in T_i when the first $k-1$ coordinates are fixed and denoted by $(k-1)$-dimensional vector p.

Input: The vertex sets of convex polytopes $Q_1,\ldots,Q_{n+1} \in \mathbb{R}^n$ and $v \in \mathbb{Q}^n$.
Output: $T_i \subset \mathbb{Z}^n$, for $i \in \{1,\ldots,n+1\}$, together with the v-distance of each point.
Mayan Pyramid algorithm:

1 For all $i \in \{1,\ldots,n+1\}$ run the following steps:

2 Initialize $T_i = \emptyset$, let $k = 1$ and let the vector of known coordinates $p = ()$ be 0-dimensional.
3 Compute $mn, mx \in \mathbb{Z}$ which are, respectively, the minimum and maximum k-th coordinates in Q_{-i} when the first $k-1$ coordinates are fixed to the coordinates in $p = (p_1, \ldots, p_{k-1})$.
4 If $k < n$, for each $p_k \in [mn, mx]$
append p_k at the end of vector p which becomes $p = (p_1, \ldots, p_k)$, increment k and recurse at step 3.
5 If $k = n$, for each $p_k \in [mn, mx]$
append p_k at the end of vector p which becomes $p = (p_1, \ldots, p_k)$, compute the v-distance of point $p \in \mathbb{Z}^n$ and insert it, together with its v-distance, in T_i.

The recursion terminates, in general, when $k = n$ or if $[mn, mx]$ is empty for any k. Notice that it is possible to remove the recursion.

Linear programming is used to compute mn, mx. For a general introduction on the uses of linear programming in combinatorial geometry the reader may consult (Grötschel, Lovász and Schrijver, 1993). Here is how we find mn, for some $k > 1$:

$$\text{minimize} \quad s \in \mathbb{R} : \qquad (p_1, \ldots, p_{k-1}, s) = \sum_{t=1, t\neq i}^{n+1} \sum_{j=1}^{m_i} \lambda_{tj} v_{tj}^k, \tag{5.1}$$
$$\sum_{j=1}^{m_t} \lambda_{tj} = 1, \qquad \lambda_{tj} \geq 0, \qquad \forall t \in \{1, \ldots, n+1\} \setminus \{i\}, \quad j \in \{1, \ldots, m_t\},$$

where v_{tj} are the vertices of Q_t, v_{tj}^k is the k-vector consisting of the first k coordinates of v_{tj}, and m_t is the cardinality of the vertex set of Q_t. Then mn is the ceiling of the optimal value of s. The same setup, with s maximized instead, gives mx as the floor of the optimum.

Computing v-distances is accomplished by linear programming as well:

$$\text{minimize} \quad s \in \mathbb{R}_{\geq 0} : \qquad (p_1, \ldots, p_n) + su = \sum_{t=1, t\neq i}^{n+1} \sum_{j=1}^{m_i} \lambda_{tj} v_{tj},$$
$$\sum_{j=1}^{m_t} \lambda_{tj} = 1, \qquad \lambda_{tj} \geq 0, \qquad \forall t \in \{1, \ldots, n+1\} \setminus \{i\}, \quad j \in \{1, \ldots, m_t\}.$$

This last linear program can be used with $k < n$ coordinates for pruning the set of integer points T_i, since in practice we concentrate only on those points with a positive v-distance. To do this, define the v^k-distance as the analogue of v-distance for the projection of Q_{-i} and of v into the first k dimensions. The v^k-distance of a point is a non-increasing function of k. We can now test the point projections as they are enumerated by the Mayan Pyramid algorithm and eliminate all points $(p_1, \ldots, p_k)$ whose v^k-distance is zero, for any k.

Incrementing the supports $\mathcal{B}_i$ is done either by decreasing the length of V or, equivalently, by lowering the bound β on the v-distance. The maximal minor D in M must satisfy $\deg_{f_i} D \geq MV_{-i}$. Hence we pick the initial sets $\mathcal{B}_i$ to be of cardinality exactly equal to MV_{-i}. In this case, if D is a nonzero polynomial then it equals the sparse resultant and we have obtained a Sylvester-type formula, i.e., one of optimal size equal to the resultant degree. 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 now summarize the matrix construction algorithm, under the assumption that a direction v has been chosen. The question of how to choose a vector v to produce a small matrix is addressed in section 7.

Input: $\mathcal{A}_i, MV_{-i}, T_i$ with the v-distance of every point, for $i \in \{1, \ldots, n+1\}$.
Output: A maximal minor D of the matrix M, such that D is a nontrivial multiple of the sparse resultant or an indication that such a minor cannot be found for the given v after some random specialization of the coefficients.
Incremental Matrix Construction algorithm:

1 Specialize the polynomial coefficients to independently and identically distributed random values.
2 For all $i \in \{1, \ldots, n+1\}$, initialize the supports $\mathcal{B}_i$ to include MV_{-i} points from T_i with largest possible v-distance.
3 Construct the matrix M containing the specialized coefficients.
4 If M has at least as many rows as columns and is also of full rank then return a non-vanishing maximal minor D of M.
5 Otherwise, if $\mathcal{B}_i = T_i$ for $i = 1, \ldots, n+1$, i.e., the supports cannot be incremented, then return with an indication that the minor D cannot be found with the current choice of coefficient specialization and vector v.
6 Otherwise, let $\mathcal{B}_i = \{p \in T_i \,|\, v\text{-distance}(p) \geq \beta\}$ where $\beta \in \mathbb{R}_{\geq 0}$ is chosen so that the minimum number of new points are added to the supports $\mathcal{B}_i$ and at least one $\mathcal{B}_i$ is incremented; go to step 3.

The termination of the algorithm relies on the fact that M contains as a maximal submatrix the resultant matrix of Canny and Emiris (1993). We need the following notion from the latter paper: Vector $v \in \mathbb{Q}^n$ is *sufficiently generic* with respect to $Q_1, \ldots, Q_{n+1}$ if it does not lie on any $(n-1)$-dimensional face defined as the Minkowski sum $F_1 + \cdots + F_{n+1}$, where every F_i is a face of Newton polytope Q_i, for $i \in \{1, \ldots, n+1\}$. The actual requirement on v is weaker: it asks that it does not lie on the boundary of any maximal cell in some mixed decomposition of $Q_1 + \ldots + Q_{n+1}$. These notions are formalized in section 6 but will not be used here.

THEOREM 5.3. (TERMINATION) *If vector v is sufficiently generic as defined in the preceding paragraph, then the matrix construction always terminates with a matrix M that has a maximal minor D which is a nontrivial multiple of the sparse resultant.*

PROOF. The proof relies on the algorithm of Canny and Emiris (1993), which constructs a square matrix whose determinant is a nontrivial multiple of the resultant. Let $\mathcal{B}_i$ contain all lattice points in T_i with positive v-distance, for all $i \in \{1, \ldots, n+1\}$. Then the set of column monomials $\cup_{i=1}^{n+1}(\mathcal{A}_i + \mathcal{B}_i)$ equals the set of column monomials in the matrix of Canny and Emiris (1993), if in the latter construction $\delta = -\epsilon v$, where ϵ is a positive infinitesimal that guarantees that the magnitude of δ is sufficiently small. Canny and Emiris have shown that when δ is sufficiently generic, as in the definition above, the constructed matrix is square, generically nonsingular and its determinant is a multiple of the resultant. This matrix is a maximal submatrix of M constructed by the present algorithm, hence there exists D corresponding to this submatrix satisfying the claim of the theorem. □

The same argument shows that the matrix construction terminates with a valid matrix M also in the case that a specific polynomial f_i appears in the minimum number of rows. Then, the maximal minor D has degree in the coefficients of f_i equal to the respective degree of the sparse resultant. If one such minor is computed for every f_i, $i \in \{1, \ldots, n+1\}$, then the sparse resultant can be obtained as the GCD of the $n+1$ minors.

For random $v \in \mathbb{Q}^n$, a rough bound on the probability that v is inadequate is given below.

LEMMA 5.4. (Emiris, 1994b, sect. 3.1.8) *Assume that the numerators of the entries of $v \in \mathbb{Q}^n$ are chosen uniformly and independently from a set of S integers and all denominators are identical and equal to an integer relatively prime to the n numerators. Then the probability that v is not sufficiently generic is bounded by $(cn!)^2/S$ where c is the number of matrix columns.*

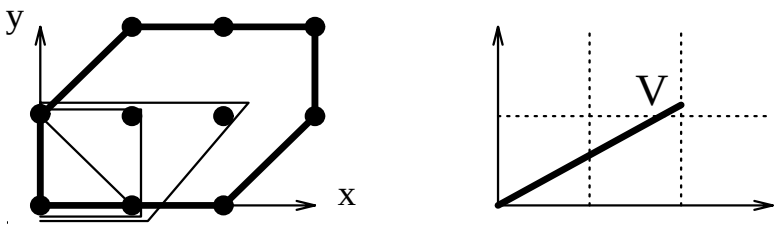

Figure 2. Q_{-2} and T_2 subsets with different v-distance bounds and one-dimensional polytope V for $\mathcal{B}_2$ in example 3.11.

This implies that if the final matrix has at most 10^3 columns, $n < 10$ and 64-bit integers are used, then the probability that v is not sufficiently generic is bounded by 0.7%. In practice 32-bit integers suffice since the algorithm typically terminates well before reaching its next to last step, where the preceding analysis applies. In general, the algorithm produces valid matrices of small size even if v is not sufficiently generic.

Step 4 can always find and return D because M has full rank. Overall, the algorithm may fail to construct M either due to the coefficient specialization or due to the choice of v. Since the most common reason is the latter, the algorithm does not try a different coefficient specialization but calls for a different v.

The rank test should consider the matrix M whose nonzero entries are symbolic coefficients. Instead, we pick random values for the coefficients from an integer interval. Checking the specialized matrix will give the correct answer with high probability, as shown in the following lemma.

LEMMA 5.5. *Suppose that M has r rows, at most that many columns, and has full rank when its nonzero entries are generic polynomial coefficients. Assume that we specialize the polynomial coefficients to uniformly and independently distributed integers in an interval of size S. Then the probability that the specialized matrix is rank-deficient is bounded by r/S.*

PROOF. Let $c \leq r$ be the number of columns and let us consider one maximal nonsingular submatrix M' of M. For each polynomial f_i, we may specialize all coefficients except one and regard $\det M'$ as a polynomial in the $n+1$ remaining generic coefficients. This polynomial has at least one nonzero term, namely the power product of all generic coefficients, each raised to a power equal to the number of rows corresponding to the respective polynomial. Hence, $\det M'$ is nonzero and has total degree of c.

The probability that $\det M'$ vanishes when all coefficients are specialized is bounded by c/S (Schwartz, 1980, lem. 1). Clearly, this bounds also the probability that M drops rank under the specialization. We complete the proof by applying $c \leq r$. □

In most cases, the size of M is less than 10^4 so 32-bit integer coefficients lead to an upper bound of 10^{-5} on the probability of error. Notice that even in the unlikely event of a bad choice of parameters, the algorithm does not produce an erroneous result, but just yields larger matrices.

EXAMPLE 3.11 (CONTINUED). The bold polygon in figure 2 is Q_{-2} of system (3.2), whereas the bold segment is the polytope V, with endpoints at the origin and $(2, 11/10)$. Equivalently, $v = (2, 11/10)$ and $\beta = 1$.

The Mayan Pyramid algorithm produces the following integer point sets with the respective approximate v-distances:

$$\begin{aligned}
T_1 &= \{(0,1;0.150),(1,0;0.100),(1,1;0.100),(1,2;0.091),(2,1;0.050),(2,2;0.050),\\
&\quad (0,2;0),(0,3;0),(2,0;0),(2,3;0),(3,1;0),(3,2;0),(3,3;0)\},\\
T_2 &= \{(0,0;0.150),(1,0;0.100),(0,1;0.091),(1,1;0.091),(2,1;0.050),\\
&\quad (1,2;0),(2,2;0),(2,0;0),(3,2;0),(3,1;0)\},\\
T_3 &= \{(0,1;0.182),(1,1;0.150),(1,0;0.111),(2,1;0.100),(2,2;0.091),(3,2;0.050),\\
&\quad (1,2;0),(2,0;0),(3,1;0),(3,3;0),(4,2;0),(4,3;0)\}.
\end{aligned}$$

Let us focus on the rows containing multiples of f_2. The first matrix constructed contains 3 rows with multiples of f_2 corresponding to the points

$$\{(0,0;0.150),(1,0;0.100),(0,1;0.091)\} \subset T_2,$$

which define the thin-line triangle at the origin, in figure 2. The second and final matrix has 4 rows containing f_2 multiples:

$$\mathcal{B}_2 = \{(0,0;0.150),(1,0;0.100),(0,1;0.091),(1,1;0.091)\} \subset T_2.$$

These points define the thin-line square at the origin in the figure. This is the set of integer points in Q_{-2} whose v-distance is larger than or approximately equal to 0.091. The third thin-line polygon in figure 2 defines an even larger subsets of $T_2 \subset \mathbb{Z}^2$, for a smaller cutoff value β on v-distance. This set, though, is never needed.

This v leads to a 13×12 matrix M of full rank for system (3.2) with $\mathcal{B}_i$ cardinalities $4, 4$ and 5, from which a 12×12 generically nonsingular minor serves as D. Recall that the sparse resultant's total degree is 11 and the classical resultant's degree is 26, whereas the algorithm in (Canny and Emiris, 1993) and its greedy variant (Canny and Pedersen, 1993) yield, respectively, a 15×15 and a 14×14 matrix.

Below, we show the 13×12 matrix and also the monomials indexing its columns (on top, denoted by their integer exponent vector) as well as the polynomial multiples filling each row (to the right).

$$(0,1)(0,2)(1,0)(1,1)(1,2)(2,0)(2,1)(2,2)(2,3)(3,1)(3,2)(3,3)$$

$$M = \begin{bmatrix}
c_{11} & 0 & 0 & c_{14} & c_{12} & 0 & 0 & c_{13} & 0 & 0 & 0 & 0 \\
0 & 0 & c_{11} & 0 & 0 & c_{14} & c_{12} & 0 & 0 & c_{13} & 0 & 0 \\
0 & 0 & 0 & c_{11} & 0 & 0 & c_{14} & c_{12} & 0 & 0 & c_{13} & 0 \\
0 & 0 & 0 & 0 & c_{11} & 0 & 0 & c_{14} & c_{12} & 0 & 0 & c_{13} \\
c_{21} & 0 & c_{24} & 0 & 0 & 0 & c_{23} & c_{22} & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & c_{21} & 0 & c_{24} & 0 & 0 & 0 & c_{23} & c_{22} & 0 \\
0 & c_{21} & 0 & c_{24} & 0 & 0 & 0 & c_{23} & c_{22} & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & c_{21} & 0 & c_{24} & 0 & 0 & 0 & c_{23} & c_{22} \\
c_{31} & c_{32} & 0 & c_{34} & c_{33} & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & c_{31} & c_{32} & 0 & c_{34} & c_{33} & 0 & 0 & 0 & 0 \\
0 & 0 & c_{31} & c_{32} & 0 & c_{34} & c_{33} & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & c_{31} & c_{32} & 0 & c_{34} & c_{33} & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & c_{31} & c_{32} & 0 & c_{34} & c_{33}
\end{bmatrix} \begin{matrix}
yf_1 \\ xf_1 \\ xyf_1 \\ xy^2f_1 \\ f_2 \\ xf_2 \\ yf_2 \\ xyf_2 \\ yf_3 \\ xyf_3 \\ xf_3 \\ x^2yf_3 \\ x^2y^2f_3
\end{matrix}$$

6. Mixed Volume Computation

Computing the mixed volume of n Newton polytopes in n dimensions is not only an important subproblem for the sparse resultant matrix construction but also a fundamental question of independent interest in sparse elimination. The main idea behind our method for computing the mixed volume is the *Lifting* algorithm by B. Sturmfels (1994). Given convex polytopes $Q_1, \ldots, Q_n \subset \mathbb{R}^n$, we define a *lifting* by choosing n linear functions $l_i : \mathbb{Z}^n \to \mathbb{Q}$. Below we formalize certain requirements on these functions. We define the *lifted polytopes*

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

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

The *lower envelope* of a convex polytope in $\mathbb{R}^{n+1}$ is the closure of the subset of all its n-dimensional faces, or facets, whose outward normals have a negative x_{n+1}-coordinate. The lifting must be sufficiently generic in order for the lower envelope of $\widehat{Q}$ to be in bijective correspondence with the Minkowski sum Q of the original polytopes. This is equivalent to the requirement that every vertex on the lower envelope be expressed uniquely as a Minkowski sum. A stronger and sufficient condition is the following.

DEFINITION 6.1. *Consider a pair of distinct vertex sets* $(\{p_1, \ldots, p_n\}, \{q_1, \ldots, q_n\})$, *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 for every such pair* $\sum_{i=1}^n (p_i, l_i(p_i)) \neq \sum_{i=1}^n (q_i, l_i(q_i))$ *or, equivalently,* $\sum_{i=1}^n l_i(p_i) \neq \sum_{i=1}^n l_i(q_i)$.

LEMMA 6.2. *If all* $2n^2$ *coordinates of* l_i *are chosen independently and uniformly from an interval of size* 2^{L_l}, *where* $L_l \in \mathbb{Z}_{>0}$, *then the probability that the lifting is not sufficiently generic is bounded by*

$$Prob[\, l_1, \ldots, l_n \text{ not sufficiently generic}] \;\leq\; \frac{1}{2^{L_l}} \frac{1}{2} m^n (m-1)^{n-1},$$

where m *is the maximum vertex cardinality of any* Q_i *over all* $i \in \{1, \ldots, n\}$.

PROOF. Fix two distinct sets of points $\{p_1, \ldots, p_n\}$ and $\{q_1, \ldots, q_n\}$ with p_i, q_i vertices of Q_i, for which $\sum_i p_i = \sum_i q_i$. We must bound the probability that the lifting fails to distinguish between the lifted images of these two points, namely that $\sum_i l_i(p_i) = \sum_i l_i(q_i)$. Assume, without loss of generality, that p_1 and q_1 differ in their first coordinate and fix all rational coefficients in the n lifting forms, except for the numerator in the first coefficient of l_1. In choosing the value of this numerator, the probability of picking the single integer value that does not distinguish between the two lifted points is $\leq 1/2^{L_l}$.

By multiplying this probability with the number of pairs $(\{p_i\}, \{q_i\})$ the result is proven. In counting the number of pairs the maximum number is found when, for every i, $p_i \neq q_i$, because otherwise the possible pairs are constrained. If the p_i's are chosen at will, there are at most $m-1$ choices for each q_i and no choice for q_n since the points must satisfy $\sum_i p_i = \sum_i q_i$. Hence the number of pairs is at most $1/2\, m^n (m-1)^{n-1}$. □

For typical values $n < 10$, $m_i < 15$ and $L_l = 64$, the probability of failure is about 45%, while for $L_l = 96$, the probability becomes smaller than 10^{-9}. The above bound is pessimistic; moreover, it is straightforward to check deterministically whether a particular choice of lifting functions is sufficiently generic (Emiris, 1994b).

For a sufficiently generic lifting, the projection of the lower envelope of $\widehat{Q}$ induces a partition of Q

into *cells*, where each cell is the image of a lower envelope face of the same dimension. In particular, facets give rise to *maximal* cells of dimension n. It can be shown that the induced subdivision is a *mixed subdivision* of the Minkowski sum Q. Mixed subdivisions are characterized by the fact that every maximal 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 from distinct Newton polytopes. In the second case it is *unmixed*. Demonstrations of these facts can be found in (Billera and Sturmfels, 1992, Sturmfels, 1994). 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 the matrix whose rows are the edge vectors generating the cell. An important property of the mixed cells is that, generically, they define a monomial basis of the coordinate ring of the ideal generated by n polynomials in n variables (Emiris and Rege, 1994), and they specify the start system for a sparse homotopy.

Several algorithms exist for the calculation of mixed volumes. One of the first approaches (Emiris, 1993) computes the entire mixed 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 Huber and Sturmfels (1992) takes advantage of repeated polytopes, while that of Verschelde and Gatermann (1994) exploits symmetry; a general implementation has been described in (Verschelde, Verlinden and Cools, 1994). These algorithms have the same worst-case asymptotic complexity as our own algorithm defined below. This complexity is analyzed below and turns out to be simply exponential in n. However, based on experimental results, our algorithm appears to be the fastest 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 the following obvious fact.

LEMMA 6.3. *Fix a lifting, let $J \subset \{1,\ldots,n\}$, and suppose that e_j is an edge of Q_j for all $j \in J$. If the Minkowski sum of the 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 $T \subset J$, the Minkowski sum $\sum_{t\in T} \widehat{e}_t$ lies on the lower envelope of the Minkowski sum $\sum_{t\in T} \widehat{Q}_t$.*

Our algorithm constructs n-tuples of edges from Q_i by starting with a pair of edges and then adding one edge from a new polytope at a time. As each edge is added, the k-tuple for $2 \le k \le n$ is tested on whether it lies on the lower envelope of the corresponding lifted Minkowski sum or not; the k-tuples that pass this test are called *valid* and are precisely those tuples that will continue to be augmented. Further pruning is achieved by eliminating those edges that cannot extend the current k-tuple from the edge sets of polytopes not yet considered. This means that for an index set J, we let $T = J \cup \{t\}$, where t ranges over $\{1,\ldots,n\} \setminus J$ and check the edge tuples corresponding to T. 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 the Minkowski sum $\widehat{e}_{i_1} + \cdots + \widehat{e}_{i_k}$. The test of interest is equivalent to asking whether $\widehat{p}$ lies on the lower envelope or not, which is formulated as follows:

$$\text{maximize } s \in \mathbb{R}_{\ge 0}: \qquad \widehat{p} - sz = \sum_{t\in\{i_1,\ldots,i_k\}} \sum_{j=1}^{m_t} \lambda_{tj} \widehat{v}_{tj}, \tag{6.1}$$

$$\sum_{j=1}^{m_t} \lambda_{tj} = 1, \qquad \lambda_{tj} \geq 0, \qquad \forall\, t \in \{i_1, \ldots, i_k\},\ \ j \in \{1, \ldots, m_t\},$$

where $z = (0, \ldots, 0, 1) \in \mathbb{Z}^{n+1}$, $\widehat{v}_{tj}$ is the j-th vertex of $\widehat{Q}_t$ and m_t the cardinality of Q_t or, equivalently, $\widehat{Q}_t$, due to the linearity of the lifting. Then $\widehat{p}$ lies on the lower envelope if and only if the maximal value of s is 0, for s expresses the vertical distance of $\widehat{p}$ from the lower envelope. Otherwise s is positive.

Input: The vertex sets of convex polytopes $Q_1, \ldots, Q_n \subset \mathbb{R}^n$, which are all subsets of the integer lattice $\mathbb{Z}^n$.
Output: $MV(Q_1, \ldots, Q_n) \in \mathbb{Z}_{>0}$.
Lift-Prune algorithm:

1 Enumerate the edges of all polytopes $Q_1, \ldots, Q_n$ respectively into sets $E_1, \ldots, E_n$.
2 Compute random lifting vectors $l_1, \ldots, l_n \in \mathbb{Q}^n$.
3 Compute lifted edge $\widehat{e}_i$ for every edge $e_i \in E_i$, $i = 1, \ldots, n$.
4 Initialize the mixed volume to 0.
5 If $E_1 = \emptyset$ then terminate.
Otherwise, pick any edge $e_1 \in E_1$, remove it from E_1, create current tuple (e_1), let sets $E'_2, \ldots, E'_n$ be copies of $E_2, \ldots, E_n$ and let $k = 1$.
6 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.
7 Increment k.
8 If $k > n$, then add the volume of the Minkowski sum of $(e_1, \ldots, e_n)$ to the mixed volume; continue at step 5.
9 If $k \leq n$ and $E'_k = \emptyset$ then continue at step 5. If $k \leq n$ and $E'_k \neq \emptyset$ then add some edge $e_k \in E'_k$ to the current tuple $(e_1, \ldots, e_{k-1})$, remove e_k from E'_k and go to step 6.

Our implementation of the Lift-Prune algorithm is publicly available from

`ftp://robotics.eecs.Berkeley.edu/pub/MixedVolume`.

Computation of the Newton polytope edges is accomplished by well-known techniques based on linear programming; see (Grötschel et.al.) for details. Notice that the edges of every original and lifted polytope are the same due to the linearity of the lifting.

The algorithm, as 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 9 picks a new edge set, it chooses the one with minimum cardinality.

In implementing the algorithm, we have used an existing implementation of the Simplex algorithm from the Numerical Recipes in C package (Press, Flannery, Teukolsky and Vetterling, 1988). Unfortunately, these sources are not free for distribution, although most sites today have access to them. The `ftp` site of the Lift-Prune program contains our corrections to the Numerical Recipes in C sources, along with our own code and the executables. It is easy to see that the bottleneck of the mixed volume program is linear programming, therefore a more efficient implementation for this problem would significantly speed up our algorithm.

As for the stability of the Simplex algorithm over double precision floating point arithmetic, it is not a major issue because the inputs are usually 32-bit integers. The crucial point here is to ensure that the lifting values, which are typically larger than the polynomial exponents, are not too large. In

any case, we must be careful in choosing a threshold value to distinguish between zero and positive optimal values for s in linear program (6.1).

The Lift-Prune algorithm is incremental in the sense that partial results are available at rather regular intervals and well before termination. This successively tighter lower bound on mixed volume is particularly useful in long runs of the program. Furthermore, we propose the following scheme for either coarse-grain or fine-grain parallelization. An initial sequential phase examines the first few polytopes. Then each valid edge combination is given to a different processor and the rest of the algorithm proceeds as before on every processor. We would typically have the initial phase examine enough polytopes in order to produce sufficiently many combinations so that each processor is given at least one valid combination. Clearly, this scheme works in a distributed environment too. When one processor completes its computation, it may examine some of the valid edge combinations that wait to be examined at another processor.

Asymptotic complexity is analyzed below and empirical results for the benchmark of cyclic n-roots are reported in section 9.

6.1. Asymptotic Complexity

The worst-case asymptotic complexity of the Lift-Prune algorithm is analyzed in this section. Motivated by the empirical observation that this bound is overly pessimistic (see experimental results in section 9), we also attempt to model the algorithm's average-case behavior. The computational model used is the real RAM (Aho, Hopcroft and Ullman, 1974), on which two different cost functions are employed. Namely, we consider the worst-case *arithmetic* (number of instructions) as well as the worst-case *bit complexity* (number of bit operations) of the algorithm.

Let g denote the maximum number of Newton polytope edges, m the maximum number of Newton polytope vertices, which is bounded by the number of nonzero monomials per polynomial, and d the maximum coordinate of any vertex, assuming that the Newton polytopes have been translated to lie in the first orthant and touch all the coordinate hyperplanes. Let L_l be the maximum bit-size of a coordinate in any lifting form l_i and $L_d = \log d$ be the maximum bit-size of any Newton polytope vertex coordinate.

The bottleneck is the combinatorial search for the valid edge tuples. Ignoring the pruning, the algorithm has to test g^n combinations, where g is an upper bound on the number of edges in every Newton polytope. Clearly, $g \leq m^2$ and the number of tests is not larger than m^{2n}.

Linear programming may be solved by any polynomial-time algorithm. In what immediately follows as well as in later sections we use Karmarkar's (1984) polynomial-time algorithm in order to derive our complexity bounds. For linear programs with V variables, C constraints and at most B bits per coefficient, the bit complexity is

$$\mathcal{O}^*(C^2V^{5.5}B^2), \tag{6.2}$$

where $\mathcal{O}^*(\cdot)$ indicates that we have ignored polylogarithmic factors in C, V, B. Every problem of the form (5.1) has $V = \mathcal{O}(nm)$ variables, $C = \mathcal{O}(n)$ constraints and $B = \mathcal{O}(L_l + L_d)$ bits per coefficient. Applying Karmarkar's result we can bound the bit complexity of every linear programming test by $\mathcal{O}^*(n^{7.5}m^{5.5}(L_l + L_d)^2)$.

Theorem 6.4. *Let m be the maximum vertex cardinality per polytope, d be the maximum degree in any variable and $\epsilon < 1$ be the probability of failure of the lifting scheme. The worst-case bit complexity of the Lift-Prune algorithm for computing $MV(Q_1, \ldots, Q_n)$ is*

$$\mathcal{O}^*(m^{2n+5.5}n^{7.5}(\log d - \log \epsilon)^2).$$

For a constant probability ϵ and systems with d bounded by a polynomial in m and n, the Lift-Prune algorithm complexity is

$$\mathcal{O}^*(m^{2n+5.5}n^{7.5}).$$

PROOF. The first step identifies the edges of the Newton polytopes by applying linear programming to every pair of vertices. For the i-th polytope there are $\mathcal{O}(m_i^2)$ pairs and the bit complexity for each is $\mathcal{O}^*(n^2 m_i^{5.5} L_d^2)$, where $L_d = \lceil \log d \rceil$. Hence the total bit complexity of this phase is $\mathcal{O}^*(n^3 m^{7.5} \log^2 d)$ and it is dominated as shown below.

There are at most m^{2n} edge tests, each reducing to a linear programming application with bit complexity $\mathcal{O}^*(n^{7.5} m^{5.5}(L_l + L_d)^2)$. The maximum coordinate is bounded by d hence $L_d \leq \log d$. From lemma 6.2, $\epsilon = m^n(m-1)^{n-1}/(2 \cdot 2^{L_l})$ is the probability that the lifting fails, therefore $L_l = \mathcal{O}(n \log m - \log \epsilon)$. Hence the total complexity is $\mathcal{O}^*(m^{2n+5.5} n^{7.5}(n \log m - \log \epsilon + \log d)^2)$, and the first claim follows. Under the additional hypotheses, the last factor is dominated and the second claim follows. □

We put the analysis in the perspective of complexity classes; for definitions see (Garey and Johnson, 1979).

THEOREM 6.5. *Computing the mixed volume is in #P.*

PROOF. The Lift-Prune algorithm puts mixed volume in complexity class #P because a nondeterministic machine could guess an edge combination that leads to a mixed cell with positive volume, then spend polynomial time to check this guess. This process is repeated for every edge tuple corresponding to a mixed cell. If we subdivide each edge to unit-length segments and restrict ourselves to these unit-length edges, then each guess leads to a subcell of unit volume and the number of distinct guesses equals the mixed volume. □

In terms of lower bounds, recall that the mixed volume problem is equivalent, for unmixed systems, to computing the volume of the convex hull of a point set. The latter is known to be #P-hard (Khachiyan, 1993), hence computing mixed volumes is also #P-hard. Moreover, it has recently been shown that mixed volume is #P-complete (Pedersen, 1994) by a reduction of computing the permanent.

To model the average-case behavior and account for the effects of pruning we should estimate the number of edge combinations that pass the test at the various stages. We define ρ_k to be the ratio of *valid* edge tuples at step k of the Lift-Prune algorithm over the total number of edge tuples for the partial Minkowski sum $\widehat{Q}_1 + \cdots + \widehat{Q}_k$. In other words, the number of valid tuples for this sum is $\rho_k m^{2k}$. This is a worst-case bound assuming that all polytopes have m vertices, therefore at most m^2 edges.

HYPOTHESIS 6.6. *Consider the k-th stage of the Lift-Prune algorithm, for $k \in \{2, \ldots, n\}$, namely the stage at which $\widehat{Q}_k$ is considered for the first time. The fraction of edge tuples that pass the validity test at this stage over the maximum possible number of edge tuples depends only on k and the number of facets of $\widehat{Q}_k$ and grows linearly with each.*

To estimate the number of facets of $\widehat{Q}_k$ we use the well-known bound on the number of facet of the convex hull of m points in $n+1$ dimensions, namely $\mathcal{O}(m^{\lfloor (n+1)/2 \rfloor})$ (Grünbaum, 1967). By the

above hypothesis,

$$\rho_k = \mathcal{O}\left(\frac{k m^{\lfloor (n+1)/2 \rfloor}}{m^{2k}}\right), \qquad k \in \{2, \ldots, n\}.$$

This assumption is supported by experimental evidence from the cyclic n-roots problem examined in section 9.

COROLLARY 6.7. *Assume that the probability of failure ϵ for the lifting scheme is constant and that the maximum coordinate of any polytope vertex d is bounded by a polynomial in m and n. Then, under hypothesis 6.6 on the number of valid edge combinations, the Lift-Prune algorithm has complexity*

$$\mathcal{O}^*(m^{\lceil n/2 \rceil + 7.5} n^{9.5}).$$

PROOF. Since the number of edges per polytope is bounded by m^2, the number of linear programming problems is bounded by

$$\begin{aligned} m^4 + \rho_2 m^4 m^2 + \cdots + \rho_{n-1} m^{2(n-1)} m^2 = \\ = m^4 + 2m^{\lfloor (n+1)/2 \rfloor + 2} + \cdots + (n-1) m^{\lfloor (n+1)/2 \rfloor + 2} \\ = \mathcal{O}(n^2 m^{\lfloor (n+1)/2 \rfloor + 2}). \end{aligned}$$

As analyzed above, the cost of every linear programming run is at most $\mathcal{O}^*(n^{7.5} m^{5.5} (\log d - \log \epsilon)^2)$. By the current hypothesis the last factor is at most polylogarithmic in m and n. The claim follows by applying the identity $\lfloor (n+1)/2 \rfloor = \lceil n/2 \rceil$. □

7. Sparse Resultant Algorithm

This section presents the overall algorithm for constructing sparse resultant matrices given $n+1$ supports $\mathcal{A}_i \subset \mathbb{Z}^n$. As already explained, the determinant of the resultant matrix is, in general, a multiple of the resultant. In several applications, including polynomial system solving, an exact matrix formula for the resultant is not required (Emiris, 1994, 1994b and 1995), though efficiency is optimized when the minor D equals the sparse resultant. 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 (Canny and Emiris, 1993). For both methods we fix the cardinality of $\mathcal{B}_1$ to MV_{-1} so that $\deg_{f_1} D = \deg_{f_1} R$. This enables us to define R as the GCD of at most $n+1$ such minors.

A crucial question in our approach is the choice of vector v. In many situations, a deterministic v which guarantees the construction of a compact matrix formula can be found. Such cases include systems whose structure is or resembles a multigraded structure, as demonstrated in section 8. Moreover, the resultant matrix constructed by our algorithm generalizes the Sylvester matrix (Sylvester, 1853) and the coefficient matrix for linear systems.

THEOREM 7.1. *Given a system of two univariate polynomials and one-dimensional vector $v \in \mathbb{R}_{>0}$, our Matrix Construction algorithm of section 5 produces the Sylvester matrix of this system. Given a system of $n+1$ linear polynomials in n variables and vector $v = (1, \ldots, 1) \in \mathbb{R}^n$, the algorithm produces the coefficient matrix of this system.*

PROOF. For two univariate polynomials, the sets T_1 and T_2 are subsets of $\mathbb{Z}$ and, for positive v, the points with positive v-distance are exactly the monomials that define the rows of the Sylvester matrix. Since MV_{-1} and MV_{-2} are equal to the degree of the second and the first polynomial, respectively, the points with positive v-distance constitute $\mathcal{B}_1$ and $\mathcal{B}_2$ in the first round of the algorithm and

the constructed matrix is the Sylvester matrix. It is known that the Sylvester matrix is generically nonsingular, therefore for specialized coefficients the constructed matrix is also nonsingular with very high probability, as determined in lemma 5.5. Hence the algorithm returns the Sylvester matrix and terminates.

For linear systems, each Q_{-i} is an n-dimensional unit simplex scaled by n and $MV_{-i} = 1$ for all $i \in \{1, \ldots, n+1\}$. Furthermore, for $v = (1, \ldots, 1)$, each $\mathcal{B}_i$ contains exactly one point at the first round, namely the origin. The corresponding matrix is the $(n+1) \times (n+1)$ coefficient matrix and it is generically nonsingular. For specialized coefficients the matrix is still nonsingular with very high probability, as given by lemma 5.5. Hence, this matrix is output and the algorithm terminates. □

For arbitrary systems, a random vector v is chosen. From theoretical arguments and empirical observations it follows that it is preferable to choose vectors $v \in (\mathbb{Q}^*)^n$ with all coordinates distinct. The goal is to have as few integer points as possible with the same v-distance in any set T_i. Theorem 5.3 establishes the fact that any sufficiently generic v will lead to a valid matrix.

In general, when no deterministic choice for v exists, the algorithm is of the Las Vegas type, in the sense that any bad probabilistic choice for v which does not lead to a valid matrix M cannot lead to wrong results but shall only increase execution time.

Input: Supports $\mathcal{A}_1, \ldots, \mathcal{A}_{n+1}$ and direction vector $v \in \mathbb{Q}^n \setminus (0, \ldots, 0)$.
Output: A maximal minor D of the matrix M, such that D is a nontrivial multiple of the sparse resultant.
Main algorithm:

1 Compute the vertex sets of Newton polytopes $Q_1, \ldots, Q_{n+1}$.
2 Use the Mayan Pyramid algorithm to compute sets $T_1, \ldots, T_{n+1} \in \mathbb{Z}^n$ and the v-distance of all points in them.
3 Use the Lift-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, report that minor D cannot be found with the current choice of v; then either a new v is supplied or one is chosen randomly, and the algorithm restarts at step 2.

A preliminary implementation of this algorithm together with the back end of a system solver using the resultant matrix is publicly available from

`ftp://robotics.eecs.Berkeley.edu/pub/emiris/res_solver`.

The first step is to compute the Newton polytope vertices. For this task we use linear programming on every support point to decide whether it is a vertex or not. This is a well-known technique; see (Grötschel et.al.) for details.

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 efficiency and exactness. The only disadvantage is that, with some very small probability, the final matrix M may be larger than over the integers.

In computing the sets T_i, we have implemented an option that allows the user to limit the maximum number of points in these sets. Typically, these sets contain many more points than eventually needed.

The user can guess an upper limit on the number of points in each $\mathcal{B}_i$, and hence in the respective T_i, and pass it to the program. Moreover, after computing each T_i, the program updates the smallest v-distance and delimits the search of subsequent point sets to points whose v-distance is at least as large.

Parallelization of the matrix construction is straightforward. discussed in the previous section Moreover, the integer point enumeration may be easily parallelized by assigning each set T_i on a different processor. Further parallelization may be achieved by having different "slices" of each Q_{-i} assigned to different processors. The most expensive phase is testing whether M has full rank. There exists a rich literature on parallel LU decomposition; see for instance (Modi, 1990).

7.1. Asymptotic Complexity

Here we analyze the worst-case asymptotic complexity of the algorithm, based on the real RAM model of computation (Aho et.al.). Again, two complexity functions are used, namely the arithmetic and the bit complexity. We make repeated use of the following theorem.

Theorem 7.2. (Emiris, 1994) *Given are convex polytopes $Q_1, \ldots, Q_n \in \mathbb{R}^n$, all of which have positive volume, and let Q_μ be the polytope of minimum volume, for some $\mu \in \{1, \ldots, n\}$. Assume that there is a constant $c \geq 1$ such that, for every $i \in \{1, \ldots, n\}$, there exists vector $b_i \in \mathbb{R}^n$ such that $b_i + Q_i \subset cQ_\mu$. Then,*

$$\mathrm{Vol}(Q_1 + \cdots + Q_n) = \mathcal{O}\left(\frac{e^n}{\sqrt{n}}\right) MV(Q_1, \ldots, Q_n),$$

where $e \approx 2.7183$ denotes the exponential base.

Let s be the maximum number of points in any of the given supports $\mathcal{A}_i$, g the maximum number of Newton polytope edges, m the maximum number of Newton polytope vertices and d the maximum coordinate of any vertex, assuming that the Newton polytopes have been translated to lie in the first orthant and touch all the coordinate hyperplanes. Let L_l be the maximum bit-size of a coordinate in any lifting form l_i and $L_d = \lceil \log d \rceil$ be the maximum bit-size of any Newton polytope vertex coordinate.

Lemma 7.3. *The complexity of the Mayan Pyramid algorithm to compute one integer point set T_i, for some $i \in \{1, \ldots, n+1\}$, and the respective v-distances is $\mathcal{O}(\mathrm{Vol}(Q_{-i}) n^{7.5} m^{5.5} L_d^2)$. If, further, the hypothesis of theorem 7.2 holds, then the complexity is $\mathcal{O}(e^n MV_{-i} n^7 m^{5.5} L_d^2)$, where e denotes the exponential base.*

Proof. Each linear programming problem in the enumeration of integer point sets T_i is expressed as in (5.1). The number of variables and constraints is $V = \mathcal{O}(nm)$ and $C = \mathcal{O}(n)$ respectively and the bit size of the coefficients is $B = L_d$. Hence, by bound (6.2) (Karmarkar, 1984), the worst-case bit complexity per linear program is $\mathcal{O}(n^{7.5} m^{5.5} L_d^2)$.

An asymptotic upper bound on the number of linear programming problems is the cardinality of T_i. By the famous result of Ehrart (1967) which bounds asymptotically the cardinality of an integer point set by the volume of its convex hull, the number of linear programs is asymptotically bounded by $\mathrm{Vol}(Q_{-i})$. The first bound is now obvious; to obtain the second bound we apply theorem 7.2. □

Now we turn to the complexity of step 4. Recall that arithmetic is carried out over a finite field, hence the bit complexity per operation is constant.

LEMMA 7.4. *Assume that the hypothesis of theorem 7.2 holds for the given Newton polytopes. Then the bit complexity of the incremental Matrix Construction algorithm of section 5 (i.e.step 4 of the Main algorithm) is* $\mathcal{O}(e^{3n}(\deg R)^3)$, *where* $e \approx 2.7183$ *denotes the exponential base and* $\deg R$ *denotes the total degree of the sparse resultant.*

PROOF. The matrix construction is dominated by the LU decomposition which is at worst cubic in the maximum number of rows. The final number of rows is bounded by the total number of points in all the T_i sets. By Ehrart's bound the complexity becomes $\mathcal{O}((\sum \mathrm{Vol}(Q_{-i}))^3)$. By theorems 7.2 and 3.10,

$$\sum_{i=1}^{n+1} \mathrm{Vol}(Q_{-i}) = \mathcal{O}\left(\frac{e^n}{\sqrt{n}}\right) \sum_{i=1}^{n+1} MV_{-i} = \mathcal{O}\left(\frac{e^n}{\sqrt{n}}\right) \deg R,$$

and the claim follows. □

We can now sum up the complexities of the different stages. Our assumptions attempt to model to some extent the average-case behavior of the algorithm and are justified by experimental evidence.

THEOREM 7.5. *Let* s, m *and* g *be the maximum number of support points, Newton polytope vertices and Newton polytope edges respectively, let* d *be the maximum coordinate of any vertex and let* ϵ *denote the probability of failure for the Lift-Prune lifting;* $\deg R$ *denotes the total degree of the sparse resultant. Suppose that the algorithm uses a constant number of vectors* v *and that the hypothesis of theorem 7.2 holds for the given Newton polytopes. Also suppose that* $s^2 = \mathcal{O}(me^n \deg R)$, d *is bounded by a polynomial in* m *and* n *and* ϵ *is constant. Then the total bit complexity of the sparse resultant algorithm is*

$$\mathcal{O}^*(e^{3n}m^{5.5})(\deg R)^3 + m^{\mathcal{O}(n)}.$$

If, moreover, $m^{2n} = e^{\mathcal{O}(n)}(\deg R)^3$, *then the total complexity is bounded by*

$$e^{\mathcal{O}(n)}(\deg R)^3.$$

PROOF. The first step of the overall algorithm computes the vertex sets of the Newton polytopes by applying linear programming to each point in the every support. Each linear program has at most $V = s$ variables, $C = \mathcal{O}(n)$ constraints and $B = L_d$ bits per coefficients, hence the total bit complexity of this step is $\mathcal{O}^*(n^3 s^{6.5} L_d^2)$ by bound 6.2; see (Karmarkar, 1984, Grötschel et.al.) for details. By the assumption on s this step is dominated by the cost of the Mayan Pyramid algorithm.

The total cost of the Mayan Pyramid algorithm is $\mathcal{O}(e^n n^7 m^{5.5} L_d^2 \deg R)$. By the hypothesis on d the L_d factor is dominated. Therefore the bit complexity of this step and the matrix construction step can be summed up to $\mathcal{O}^*(e^{3n}(\deg R)^3 m^{5.5})$. Under the current hypothesis the bound of theorem 6.4 on the bit complexity of the Lift-Prune algorithm is $\mathcal{O}^*(m^{2n+5.5}n^{7.5})$. The first bound now follows.

The second bound on total complexity models the fact that the complexity is largely dominated by the matrix construction phase. Under the additional hypothesis the computation of mixed volumes is dominated. □

8. Multihomogeneous Systems

We concentrate on unmixed homogeneous systems where the variables can be partitioned into groups so that every polynomial is homogeneous in each group. Such polynomials, and the resulting systems, are called *multihomogeneous*. We focus on a subclass of multihomogeneous systems called

multigraded, which includes all systems for which exact sparse resultant matrices are known to exist. Our algorithm produces these matrices for certain deterministic choices of vector v. Moreover, it produces optimal matrices for systems approximating the multigraded structure, as exemplified later. Hence the importance of multihomogeneous systems for sparse elimination.

We partition the variables into r groups so that each polynomial is homogeneous of degree d_k in the k-th group, with $k \in \{1, \ldots, r\}$. For the k-th group, $n_k + 1$ indicates the number of variables. Such a system is said to be of *type*

$$(n_1, \ldots, n_r; d_1, \ldots, d_r),$$

where the number of equations is $n + 1$ and $n = \sum_{k=1}^{r} n_k$. There should be no confusion from the fact that the polynomials given may be multihomogeneous. To apply our algorithm we simply dehomogenize each group of variables by setting the $(n_k + 1)$-st variable to one.

An improved Bézout bound exists on the number of isolated roots for arbitrary systems of multihomogeneous polynomials (also called m-homogeneous). If the degree of polynomial i is d_{ij} in the j-th variable subset, then the number of common isolated solutions for the system of n polynomials is bounded by

$$\text{the coefficient of } \prod_{j=1}^{r} x_j^{n_j} \text{ in polynomial } \prod_{i=1}^{n} \left(\sum_{j=1}^{r} d_{ij} x_j \right).$$

For a recent generalization see (Morgan et.al.).

The Newton polytope for every polynomial is the Minkowski sum of r n_k-dimensional simplices, each on a disjoint coordinate subspace. Every simplex is denoted by $d_k S_{n_k}$ and is the convex hull of n_k segments of length d_k rooted at the origin and extending along each of the n_k axes corresponding to the variables in this group. Equivalently, S_{n_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 the integer polytope $P \subset \mathbb{R}^n$ which is simply the copy of the (unique) input Newton polytope scaled by n, i.e.,

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

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

DEFINITION 8.1. *Consider an unmixed system of $n+1$ multihomogeneous polynomials in n variables of type $(n_1, \ldots, n_r; d_1, \ldots, d_r)$, where $n_1 + \cdots + n_r = n$. This system is called* multigraded *if, for every $k \in \{1, \ldots, r\}$, $n_k = 1$ or $d_k = 1$.*

Multigraded systems include all systems for which Sylvester-type formulae exist and, in particular, linear systems, systems of two univariate polynomials and bihomogeneous systems of three polynomials whose resultant is, respectively, the coefficient determinant, the Sylvester resultant and the Dixon resultant (Dixon, 1908).

For the resultant matrix of a multigraded system, 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

$$B = \sum_{k=1}^{r} m_k S_{n_k} \subset \mathbb{R}^n, \qquad \text{where } \; m_k = (d_k - 1) n_k + d_k \sum_{j:\, \pi(j) < \pi(k)} n_j, \qquad k \in \{1, \ldots, r\}. \quad (8.1)$$

Table 1. Hardware specifications

machine	clock rate [MHz]	memory [MB]	Spec 92Int	Spec 92FP
DEC 5240	40	64	28	36
DEC Alpha 3300	150	64	66	92
DEC Alpha 3600	175	320	114	162
Sun Sparc 10/40	40	32	50	60
Sun Sparc 10/51	50	32	65	83
Sun Sparc 20/61	60	32	95	93

Moreover, for every multigraded system an optimal matrix formula can be constructed based on the following result.

Theorem 8.2. (Sturmfels and Zelevinsky, 1994) *For a multigraded system there exists a Sylvester-type matrix formula for the sparse resultant for every permutation π of the indices $\{1,\ldots,r\}$. The matrix is defined by the multiindex $(m_1,\ldots,m_r)$ of expression (8.1).*

We shall prove that, for a given permutation, the incremental algorithm constructs the corresponding Sylvester-type matrix at the first round.

Lemma 8.3. *Partition the n coordinates of vector $v \in \mathbb{Q}^n$ into r groups following the partition of variables and set every coordinate in the k-th group equal to $(nd_k - m_k)/n_k \in \mathbb{Q}$. Then $P - V = B$.*

Proof. By using the fact that $\sum_{k=1}^{r} n_k = n$ and that, for every k, we have $d_k = 1$ or $n_k = 1$, it can be shown that $(nd_k - m_k)/n_k > 0$, $\forall k$. Consider any point $p \in P - V$ with coordinates grouped in r groups, each of cardinality n_k. For a specific group, all coordinates are equal to c. For any k such that $n_k = 1$ we have two conditions on c:

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

which is equivalent to $0 \leq c \leq m_k$. For any k such that $n_k > 1$ and $d_k = 1$, we have two conditions on the sum s of the n_k coordinates in the k-th group:

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

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

To see how this v was chosen, observe that B is a scaled-down copy of P, where the scaling has occurred by a different factor for each group of n_k coordinates. Given the sequence $(n_1,\ldots,n_r)$, polytopes P and B are entirely defined by their unique vertex with no zero coordinate; v is the vector between these two vertices.

Theorem 8.4. *Given a multihomogeneous system of type $(n_k,\ldots,n_r;d_1,\ldots,d_r)$ such that $n_k = 1$ or $d_k = 1$ for $k = 1,\ldots,r$, define $v \in \mathbb{Q}^n$ with the k-th group of coordinates equal to*

$$(nd_k - m_k)/n_k, \qquad \textit{where } m_k \textit{ is defined by expression (8.1).}$$

Table 2. The performance of the sparse resultant algorithm on a SUN SPARC 10/40.

type	vector $v \in \mathbb{Z}^n$	$\deg R$	$\deg D$	#rows in M	greedy	CPU time
$(2,1,1;1,2,2)$	$(2,2,3,1)$	240	240	240	> 670	42s
$(1,1,1,1;2,2,1,1)$	$(7,5,2,1)$	480	480	480		1m 0s = 60s
$(1,1,1,1;3,3,1,1)$	$(10,7,2,1)$	1080	1080	1080		2m 11s = 131s
$(1,1,1,1;3,3,2,1)$	$(10,7,3,1)$	2160	2160	2160		4m 3s = 243s
$(1,1,1,1;3,3,3,1)$	$(10,7,4,1)$	3240	3240	3240		6m 29s = 389s
$(2,1;2,1)$	$(97,103,300)$	48	52	52	103	0s
$(2,1;2,2)$	$(101,99,500)$	96	104	104	206	6s
$(2,1,1;2,1,1)$	$(301,299,200,100)$	240	295	340		2m 43s = 163s
$(2,1,1;2,2,1)$	$(301,299,304,100)$	480	592	690		18m 56s = 1136s
$(2,1,1;2,2,2)$	$(300,310,290,100)$	960	1120	1200		2h 58m = 10680s

Then the first matrix constructed by the resultant matrix 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 by theorem 8.2 and the results of (Sturmfels and Zelevinsky, 1994). Note that the formula obtained corresponds to the permutation π used in the definition of m_k. □

We now report on some experiments. Table 1 contains the hardware specifications of the machines employed.

We have been able to produce all possible Sylvester-type formulae for various multihomogeneous examples with $n_k = 1$ or $d_k = 1$ for all k. Furthermore, for systems that do not fall within this class we have used v defined similarly and obtained near-optimal resultant matrices. The input for the experiments in table 2 is the $n+1$ supports and the vector v shown in the table. The output is the $n+1$ n-fold mixed volumes, the sum of which gives the total degree of the sparse resultant, the point sets T_i with their v-distance, for all $i \in \{1, \ldots, n+1\}$, and a square resultant matrix with generically nonzero determinant D. The symbols $\deg R$ and $\deg D$ respectively indicate the total degree of the sparse resultant and of the maximal minor D that our algorithm constructs. Since $\deg D$ expresses only the number of columns in the final matrix M and since the algorithm's complexity also depends on the final number of rows in M, we also report the latter.

Table 2 also reports preliminary running times on the SUN SPARC 10/40 of table 1, rounded to the nearest integer number of seconds. For the first set of examples, which are all multigraded, the algorithm uses the fact that these systems have Sylvester-type formulae to avoid testing for singularity; this explains the fast execution times. However, for the second class of examples the algorithm not only builds the matrix but also tests whether it has full rank and, typically, has to increment it before it finds a generically nonsingular maximal submatrix.

For the systems for which there exists k such that $n_k > 1$ and $d_k > 1$, we used the same recipe as above to calculate m_i and v, and then have perturbed the latter v to obtain the results shown. For types $(2,1;2,1)$ and $(2,1;2,2)$ the smallest matrix is obtained for $\pi = (2,1)$ and $v = (1,1,3)$

and $(1,1,5)$ respectively. For the last three types we used permutation $(1,2,3)$, resulting in vectors $v=(3,3,2,1)$, $(3,3,3,1)$ and $(3,3,3,1)$ respectively.

It is interesting to compare these resultant matrices to those computed by the greedy algorithm in (Canny and Pedersen, 1993), with sizes shown in column "greedy". There is a randomization step in this algorithm that might lead to matrices of slightly smaller size. Nonetheless, the results from a single run of the greedy algorithm suggest that the present approach yields more economical formulae.

We have applied the results on multigraded systems and systems resembling the multigraded structure in studying concrete problems in vision, robotics, structural biology as well as game theory and computational economics (Emiris, 1994b).

9. Cyclic n-Roots

This section discusses the practical performance of our algorithms applied to the standard benchmark problem of cyclic n-roots, a family of systems encountered in Fourier analysis. First we examine empirical results of the Lift-Prune implementation and then we look at the matrices constructed by our resultant code. Table 3 displays the running times of the Lift-Prune program on the problem of cyclic n-roots for the DEC Alpha 3300 of table 1. The times have been rounded to the nearest integer number of seconds.

The polynomial system is the following:

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

For small values of n the exact cardinalities of isolated roots, appearing in the first column of table 3, were derived in a series of articles. For $n=4$ the variety has unit dimension and no isolated roots. Björck (1990) credits L. Lovász with settling the case $n=5$. The same article states that $n=6$ was essentially solved in (Björck and Fröberg, 1991) except for an error corrected by D. Lazard. The problem for $n=7$ was solved in (Backelin and Fröberg, 1991) with the help of Gröbner bases calculations on J. Backelin's program Bergman. For $n=8$ there are 1152 isolated roots in addition to the one-dimensional variety (Björck and Fröberg, 1994).

For $n\geq 9$ the precise number of isolated roots is unknown and for $n=10$ even finiteness is open. Our bound for $n=11$ verifies the conjecture by Fröberg and Björck that the number of roots for prime n is $\binom{2n-2}{n-1}$. This value was shown to be an upper bound for every n (Pottier, 1995).

Our experimental results support the following conjecture.

Conjecture 9.1. *When the variety of the cyclic n-root system has zero dimension, then the mixed volume gives the exact number of affine solutions.*

All running times should be solely viewed as rough indications of the problem's intrinsic complexity and the algorithms' performances. In addition, it must always be remembered that different algorithms compute different outputs. In particular, the mixed volume computation constructs a monomial basis for the coordinate ring and the mixed cells computed define the start system of a sparse homotopy. On the other hand, Gröbner bases lead to a method for computing the system's roots and provide more information, in particular when the dimension of the variety is positive.

Table 3. Lift-Prune algorithm performance for the cyclic n-roots problem; timings are on a DEC Alpha 3300. Timings for GB are on a DEC Alpha 3600 and for the algorithm in (Verschelde, Gatermann and Cools, 1995) on a DEC 5240.

n	#isolated roots	mixed volume	Lift-Prune	(Verschelde et.al. 1995) static	DRL by GB #isol. roots	DRL by GB time
3	6	6	0s			
4	0	16	0s		16	0s
5	70	70	0s	0s	70	0s
6	156	156	2s	5s	156	1s
7	924	924	27s	2m 36s = 156s	924	2m 52s = 172s
8	1152	2560	4m 19s = 259s	1h 57m 37s = 7057s	∞ or 2560	56m 6s = 3366s
9	unknown	11016	40m 59s = 2459s	–	–	–
10	unknown	35940	4h 50m 14s = 17414s	–	–	–
11	unknown	184756	38h 26m 44s = 138404s	–	–	–

We first compare our algorithm with the lifting algorithm of Verschelde, Gatermann and Cools (1995), namely their *static* technique, which is the fastest in their article. The mixed volumes computed by this algorithm agree with the output of the Lift-Prune algorithm. The experiments were conducted on a DEC 5240 with performance ratings shown in table 1.

We compare running times with the Gröbner bases package GB by Faugère (1995), since it seems to be the fastest available system; for a comparison with other systems see the (Faugère, 1995) and the next paragraph for some concrete examples. GB was executed on a DEC Alpha 3600 and computed the ideal basis with respect to the degree reverse lexicographic (DRL) ordering over a finite field. Of course, this computation has a small probability of error, but so does our mixed volume algorithm. For $n = 8$ the computed basis implies that the associated variety is one-dimensional. Substituting random coefficients, GB computes a bound on the number of isolated roots equal to the mixed volume. To solve the system, further computation is necessary to transform the DRL basis to a lexicographic basis. For $n = 6$ and 7, for the same computation over the integers, GB took 3 seconds and 6 hours, respectively, on a Sun Sparc 10/40.

We also used Macaulay (Stillman, Stillman and Bayer, 1992) to compute a reverse lexicographic Gröbner basis (mod 31991) for $n = 6$ and 7 on the Sun Sparc 10/51 of table 1. The CPU timings were, respectively, 4 seconds and 18 minutes and 48 seconds.

We have also applied our resultant implementation to this problem. To view the system as an overconstrained one, we "hide" one of the variables in the coefficient field. In other words, we consider the polynomials as functions on $n-1$ variables with coefficients in $\mathbb{R}[x_n]$. Both the resultant and the resultant matrix have entries in $\mathbb{R}[x_n]$. This method to system solving is formalized in (Emiris, 1995).

Experimental results are shown in table 4, with running times rounded to the nearest integer, on the Sun Sparc 10/51 of table 1. The performance of our code is not optimized, because this is the *offline* phase of the polynomial system solver and we have focused on the *online* part that takes the resultant matrix and computes the common roots.

All v vectors are random perturbations of vector $(1, \ldots, 1)$, since this works best for small dimensions. We also compare the greedy algorithm of (Canny and Pedersen, 1993), which yields matrices of comparable size. We must note that a thorough study of this family of systems goes beyond the

Table 4. Sparse resultant algorithm performance on a SUN SPARC 10/51 for the cyclic n-roots.

n	vector $v \in \mathbb{Z}^n$	$\deg R$	$\deg D$	# rows in M	greedy	CPU time
3	$(82, 71)$	6	6	6	6	0s
4	$(82, 71, 98)$	20	25	29	26	0s
5	$(91, 59, 211, 5)$	85	147	206	150	21s
6	$(82, 71, 98, 64, 77)$	290	887	1516		1h 30m 48s = 5448s

scope of this paper. In particular, it is possible that a change of variables may lead to more tractable systems and, in particular, faster calculation of the number of roots and smaller resultant matrices. Some ideas can be found in (Emiris, 1994b, Faugère, 1995).

10. Conclusion

We have proposed a new incremental algorithm for constructing sparse resultant matrices, namely square matrices in the polynomial coefficients which are generically nonsingular and whose determinant is a multiple of the sparse resultant. Under reasonable assumptions, the algorithm has asymptotic complexity simply exponential in n and polynomial in the total resultant degree. This behavior is also observed in practice from a series of experiments.

The main limitation of the new algorithm is the existence of a randomized step in the choice of direction v. In general, there is no guarantee that a vector v will produce smaller matrices than previous algorithms. However, in practice, we have never encountered this problem and the constructed matrices are considerably smaller as the problem dimension increases. Moreover, there are several classes of systems for which deterministic choices for v exist, leading to optimal matrices. These systems include the multigraded systems which include, in turn, all algebraic systems for which optimal matrix formulae provably exist. Our algorithm is able to construct these optimal matrices.

We also present an efficient algorithm for computing mixed volumes which is, to the best of our knowledge, the fastest to date in terms of empirical complexity. Its worst-case asymptotic complexity is simply exponential in n, which matches asymptotically the known lower bound, whereas its speed in terms of empirical complexity is illustrated by a series of benchmarks.

To derive a priori bounds on the size of the resultant matrices we may study the incremental method in relation to the theory of Koszul complexes. Another intriguing relationship is between our approach to building multiplication tables and Gröbner bases. Specifically, the monomial sets that we define resemble those specified by a Gröbner basis, since they are concentrated near the origin or, in Gröbner bases terminology, "under the staircase". Another feature in common with Gröbner bases is that our algorithm can treat systems with a number of polynomials larger than the number of variables.

The results on multigraded systems have been generalized in (Weyman and Zelevinsky, 1994), though a constructive approach that would exploit this generalization has yet to be found. An important merit of the work on resultants is its practical application in solving algebraic systems in kinematics, vision, modeling as well as game theory and computational economics; see e.g. (Emiris, 1994b). In this respect, an open question is the transformation of arbitrary systems to an equivalent

form that is amenable to sparse elimination and, in particular, to the efficient computation of mixed volumes and sparse resultants. Ultimately, it is desirable to have an algorithm that transforms the given system to an equivalent one possessing the minimum possible mixed volume.

Acknowledgment

We thank the anonymous referees for suggesting a number of expository and stylistic improvements.

References

A.V. Aho, J.E. Hopcroft, and J.D. Ullman. *The Design and Analysis of Computer Algorithms.* Addison-Wesley, Reading, Mass., 1974.

W. Auzinger and H.J. Stetter. An elimination algorithm for the computation of all zeros of a system of multivariate polynomial equations. In *Proc. Intern. Conf. on Numerical Math., Intern. Series of Numerical Math., 86*, pages 12–30. Birkhäuser Verlag, Basel, 1988.

D.N. Bernstein. The number of roots of a system of equations. *Funct. Anal. and Appl.*, 9(2):183–185, 1975.

U. Betke. Mixed volumes of polytopes. *Arch. der Math.*, 58:388–391, 1992.

J. Backelin and R. Fröberg. How we proved that there are exactly 924 cyclic 7-roots. In *Proc. ACM Intern. Symp. on Symbolic and Algebraic Computation*, pages 103–111, Bonn, 1991.

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.

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.

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

G. Björck. Functions of modulus 1 on z_n whose Fourier transforms have constant modulus, and "cyclic n-roots". In J.S. Byrnes and J.F. Byrnes, editors, *Recent Advances in Fourier Analysis and Its Applications*, pages 131–140. Kluwer Academic, 1990. NATO Adv. Sci. Inst. Ser. C.

L.J. Billera and B. Sturmfels. Fiber polytopes. *Annals of Math.*, 135:527–549, 1992.

B. Buchberger. Gröbner bases: An algebraic method in ideal theory. In N.K. Bose, editor, *Multidimensional System Theory*, pages 184–232. Reidel Publishing Co., 1985.

Y.D. Burago and V.A. Zalgaller. *Geometric Inequalities.* Grundlehren der mathematischen Wissenschaften, 285. Springer, Berlin, 1988.

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

A. Cayley. On the theory of elimination. *Dublin Math. J.*, II:116–120, 1848.

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, 1993. Springer Verlag.

D. Cox, J. Little, and D. O'Shea. *Ideals, Varieties, and Algorithms.* Undergraduate Texts in Mathematics. Springer-Verlag, New York, 1992.

J. Canny and P. Pedersen. An algorithm for the Newton resultant. Technical Report 1394, Computer Science Dept., Cornell University, 1993.

J. Canny and J.M. Rojas. An optimal condition for determining the exact number of roots of a polynomial system. In *Proc. ACM Intern. Symp. on Symbolic and Algebraic Computation*, pages 96–102, Bonn, July 1991.

A.L. Dixon. The eliminant of three quantics in two independent variables. *Proceedings of London Mathematical Society*, 6:49–69, 209–236, 1908.

I. Emiris and J. Canny. A practical method for the sparse resultant. In *Proc. ACM Intern. Symp. on Symbolic and Algebraic Computation*, pages 183–192, Kiev, 1993.

E. Ehrart. Sur un problème de géométrie diophantienne, i. polyèdres et réseaux. *J. Reine Angew. Math.*, 226:1–29, 1967.

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

I.Z. Emiris. On the complexity of sparse elimination. Technical Report 840, Computer Science Division, U.C. Berkeley, 1994. Submitted for publication.

I.Z. Emiris. *Sparse Elimination and Applications in Kinematics.* PhD thesis, Computer Science Division, Dept. of Electrical Engineering and Computer Science, University of California, Berkeley, December 1994.

I.Z. Emiris. A general solver based on sparse resultants. In *Proc. PoSSo (Polynomial System Solving) Workshop on Software*, pages 35–54, Paris, March 1995.

I.Z. Emiris and A. Rege. Monomial bases and polynomial system solving. In *Proc. ACM Intern. Symp. on Symbolic and Algebraic Computation*, pages 114–122, Oxford, July 1994.

J.-C. Faugère. State of GB and tutorial. In *Proc. PoSSo (Polynomial System Solving) Workshop on Software*, pages 55–71, Paris, March 1995.

W. Fulton. *Introduction to Toric Varieties.* Number 131 in Annals of Mathematics. Princeton University Press, Princeton, 1993.

M.R. Garey and D.S. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness.* Freeman, New York, 1979.

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, pp. 1–62).

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

M. Grötschel, L. Lovász, and A. Schrijver. *Geometric Algorithms and Combinatorial Optimization.* Springer-Verlag, Berlin, 2nd edition, 1993.

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

J.L. Hafner and K.S. McCurley. Asymptotically fast triangularization of matrices over rings. *SIAM J. Computing*, 20(6):1068–1083, 1991.

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, Aug. 1992.

B. Huber and B. Sturmfels. Bernstein's theorem in affine space. *Discr. and Computational Geometry*, 1995. To appear.

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

N. Karmarkar. A new polynomial-time algorithm for linear programming. *Combinatorica*, 4:373–395, 1984.

L. Khachiyan. Complexity of polytope volume computation. In János Pach, editor, *New Trends in Discrete and Computational Geometry*, chapter IV. Springer-Verlag, 1993.

A.G. Khovanskii. Newton polyhedra and the genus of complete intersections. *Funktsional'nyi Analiz i Ego Prilozheniya*, 12(1):51–61, Jan.–Mar. 1978.

A.G. Khovanskii. *Fewnomials.* AMS Press, Providence, Rhode Island, 1991.

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.

Y.N. Lakshman. *On the complexity of computing Gröbner bases for zero-dimensional polynomial ideals.* PhD thesis, Computer Science Department, Rensselaer Polytechnic Institute, Troy, New York, December 1990.

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

T.Y. Li and X. Wang. The BKK root count in C^N. Manuscript, 1994.

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

D. Manocha and J. Canny. Multipolynomial resultants and linear algebra. In *Proc. ACM Intern. Symp. on Symbolic and Algebraic Computation*, pages 96–102, 1992.

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.

B. Mishra. *Algorithmic Algebra.* Springer Verlag, New York, 1993.

J.J. Modi. *Parallel Algorithms and Matrix Computation.* Oxford Applied Mathematics and Computing Science Series. Clarendon Press, Oxford, 1990.

A.P. Morgan, A.J. Sommese, and C.W. Wampler. A product-decomposition theorem for bounding Bézout numbers. *SIAM J. Numerical Analysis.* To appear. Manuscript, 1993.

P. Pedersen. AMS–IMS–SIAM Summer Conference on Continuous Algorithms and Complexity. Mt. Holyoke, Mass., July 1994.

W.H. Press, B.P. Flannery, S.A. Teukolsky, and W.T. Vetterling. *Numerical Recipes in C : the Art of Scientific Computing.* Cambridge University Press, Cambridge, 1988.

L. Pottier. Bounds for degree of the n-cyclic system. INRIA Sophia-Antipolis, Manuscript, 1995.

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

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.

J.M. Rojas. A convex geometric approach to counting the roots of a polynomial system. *Theor. Comp. Science*, 133(1):105–140, 1994.

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

R. Schneider. *Convex Bodies: The Brunn-Minkowski Theory.* Cambridge University Press, Cambridge, 1993.

M. Stillman, M. Stillman, and D. Bayer. Macaulay user manual. NSF Regional Geometry Institute, Amherst College, Amherst, Mass., 1992.

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.

B. Sturmfels. On the Newton polytope of the resultant. *J. of Algebr. Combinatorics*, 3:207–236, 1994.

B. Sturmfels. *On the Number of Real Roots of a Sparse Polynomial System*, volume 3, pages 137–143. AMS, Fields Inst. Commun., Providence, RI, 1994.

J.J. Sylvester. On a theory of syzygetic relations of two rational integral functions, comprising an application to the theory of Sturm's functions, and that of the greatest algebraic common measure. *Philosophical Trans.*, 143:407–548, 1853.

B. Sturmfels and A. Zelevinsky. Multigraded resultants of Sylvester type. *J. of Algebra*, 163(1):115–127, 1994.

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

J. Verschelde and K. Gatermann. Symmetric Newton polytopes for solving sparse polynomial systems. *Adv. Appl. Math.*, 16(1):95–127, 1995.

J. Verschelde, K. Gatermann, and R. Cools. Mixed volume computation by dynamic lifting applied to polynomial system solving. Technical Report TW 222, Katholieke Universiteit Leuven, Dept. of Computer Science, 1995.

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.

J. Weyman and A. Zelevinsky. Determinantal formulas for multigraded resultants. *J. Algebraic Geom.*, 3(4):569–597, 1994.