Polynomial System Solving: the Case of a Six-Atom Molecule

Ioannis Z. Emiris, Bernard Mourrain

INRIA

INSTITUT NATIONAL DE RECHERCHE EN INFORMATIQUE ET EN AUTOMATIQUE

Polynomial system solving: the case of a six-atom molecule

Ioannis Z. Emiris & Bernard Mourrain

N° 3075

Décembre 1996

THÈME 2

ISSN 0249-6399

INRIA
SOPHIA ANTIPOLIS

Polynomial system solving: the case of a six-atom molecule

Ioannis Z. Emiris & Bernard Mourrain *

Thème 2 — Génie logiciel
et calcul symbolique
Projet SAFIR

Rapport de recherche n ° 3075 — Décembre 1996 — 28 pages

Abstract: A relatively new branch of computational biology and chemistry has been emerging as an effort to apply successful paradigms and algorithms from geometry and robot kinematics to predicting the structure of molecules, embedding them in Euclidean space, and finding the energetically favorable configurations. We illustrate several efficient algebraic algorithms for enumerating all possible conformations of a cyclic molecule and for studying its singular locus. Recent advances in computational algebra are exploited, including distance geometry, sparse polynomial theory based on Newton polytopes, and matrix methods for solving nonlinear multivariate polynomial systems. With respect to the latter, we compare sparse resultants, Bezoutians, and Sylvester resultants in cascade, in terms of performance and numerical stability.

Key-words: polynomial equation, sparse resultant, Bezoutian, Sylvester resultant, eigenvalue, singular value, singular locus, geometry of lines, cylclohexane, computational biology and chemistry

(Résumé : tsvp)

* INRIA, SAFIR, 2004 route des Lucioles, B.P. 93, 06902 Sophia Antipolis, iemiris, mourrain@sophia.inria.fr

Unité de recherche INRIA Sophia Antipolis
2004 route des Lucioles, BP 93, 06902 SOPHIA ANTIPOLIS Cedex (France)
Téléphone : 04 93 65 77 77 – Télécopie : 04 93 65 77 65

Résolution d'équations polynomiales : le cas d'une molécule à six atomes

Résumé : Une branche relativement récente de la biologie et de la chimie algorithmique consiste à utiliser des paradigmes ou algorithmes venant de la géométrie ou de la robotique et de les appliquer à la recherche de configurations de molécules. Nous illustrons et détaillons ainsi plusieurs algorithmes de résolution d'équations polynomiales permettant de trouver toutes les configurations possibles d'une molécule à 6 atomes. Nous nous intéressons également à l'étude du lieu critique de ces configurations. De récent travaux en algèbre effective sont utilisés, incluant la géométrie des distances, la théorie des polytopes de Newton et les méthodes matricielles pour la résolution d'équations polynomiales. Nous comparons différentes approches telles les résultants « creux », les Bezoutiens et les résultants de Sylvester en cascade, du point de vue de leur performance et de leur stabilité numérique.

Mots-clé : équation polynomiale, résultant creux, Bezoutien, résultant de Sylvester, valeur propre, valeur singulière, lieu singulier, géométrie des droites, cyclohexane, biologie et chimie algorithmiques

1 Introduction

One branch of computational molecular biology and computational chemistry wishes to exploit today's computational power in order to automate and accelerate structure determination for function identification and drug design. A relatively recent effort has been the application of successful paradigms and techniques from geometry, robot kinematics as well as vision to predicting the structure of molecules, embedding them in Euclidean space and finding the energetically favorable configurations [PC94, Emi94]. The main premise for this interaction is the observation that various structural requirements on molecules can be modeled as macroscopic geometric or kinematic constraints.

This work focuses on algebraic algorithms that have been very useful in the areas mentioned above but are not well-known among structural biologists and computational chemists. We give a brief introduction to algorithms for studying basic properties of systems of simultaneous polynomial equations, in particular for finding all their common roots and identifying their singularities. These methods are applied to concrete molecular problems, thus providing an illustrated introduction, while several pointers to specialized publications are provided.

There are two questions studied in depth. The first, in section 2, examines algorithms to identify all possible conformations of a molecule, given some constraints on its primary structure. The problem is first translated into algebraic terms. This is done in sections 2.1.1 and 2.1.2, respectively, by the classical angle formulation applied in a clever way in order to reduce the problem dimension, and by the more recent approach of distance geometry. Sections 2.2 and 2.3 respectively, apply recent advances in computational algebra in order to estimate the number of solutions, and then compute all of them by reducing the nonlinear problem to a problem in linear algebra. For algebraists, the principal interest of section 2.3 lies in the comparison of several alternatives in solving the same multivariate problem, in particular sparse resultants, Bezoutians and repeated application of Sylvester resultants.

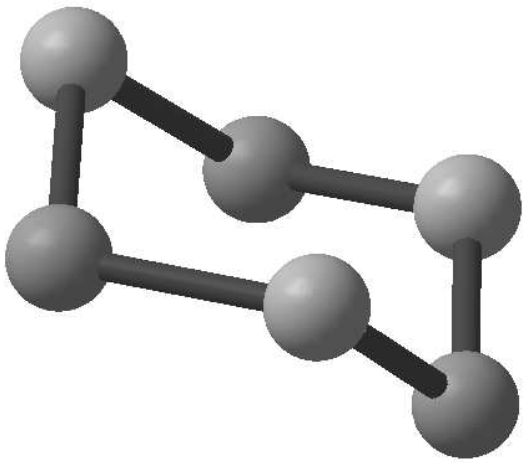
The second problem concerns the singular configurations of the molecule defined by the given data. Section 3 proposes constructive methods for identifying the infinitesimally unstable, or geometrically degenerate, situations.

Most of the implementations referred to in this paper are already publicly available, otherwise they can be obtained by the authors.

Related work is mentioned in each corresponding context.

2 Molecular conformations

This section examines the problem of computing all *conformations* of a molecule. Conformations specify the 3-dimensional structure of the molecule. It has been argued by Gō and Scherga [GS70] that energy minima can be approximated by allowing only the dihedral angles to vary, while keeping bond lengths and bond angles fixed. A *dihedral* angle is the solid angle between two consecutive planes, each defined by an atom and its two links. At a first level of approximation, therefore, solving for the dihedral angles under the assumption of *rigid geometry* provides information for the energetically favorable configurations. We consider *cyclic* molecules of six atoms to illustrate our approach.



The figure shows one configuration of such a molecule, drawn by IZIC [FKM94]. The relationship to geometry and robotics is obvious once bonds are thought of as rigid joints and atoms as the mechanism's links or articulations. The only movement allowed is rotation around the bonds' axes, so the question of identifying a conformation reduces to finding the respective pose. In kinematic terms, the molecule is equivalent to a *serial mechanism* in which each pair of consecutive axes intersects at a link. This connection is spelled out in section 3. This implies that the link offsets are zero for all six links, which will allow us to reduce the 6-dimensional problem to a system of 3 polynomials in 3 unknowns. The product of all link transformation matrices is the identity matrix, since the end-effector is at the same position and orientation as the base link.

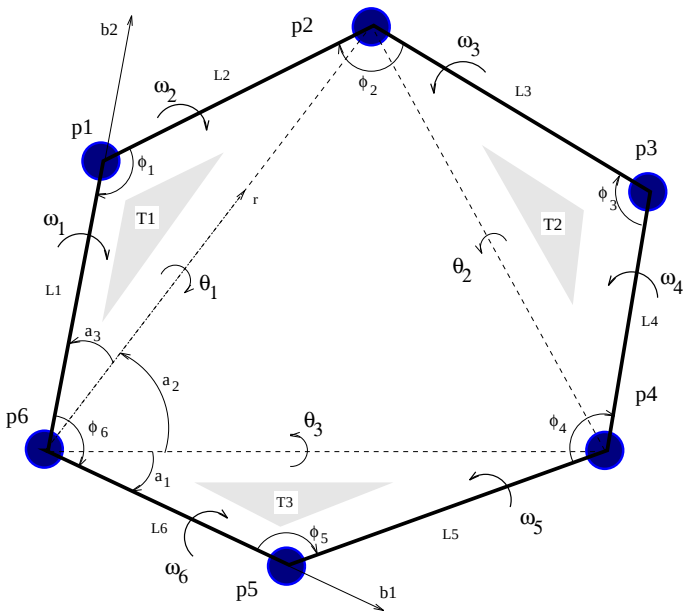


Figure 1: The parameters.

Extending to molecules with more than six varying dihedrals has been approached by a grid search of the space of the two last dihedrals, each grid point giving rise to a six-dimensional subproblem [GS70, MZW94]. For molecules with six free dihedrals and an arbitrary number of atoms, the problem is identical to the inverse kinematics of a general serial manipulator with six rotational joints, which is a settled question for roboticists [MC92].

2.1 Algebraic formulation

The molecule has a cyclic backbone of 6 atoms, typically of carbon. They determine primary structure, the object of our study. Carbon-hydrogen or other bonds outside the backbone are ignored. The bond lengths and angles provide the constraints while the six dihedral angles are allowed to vary.

2.1.1 A formulation using angles

We adopt an approach proposed by Parsons [PC94]. Notation is defined in figure 1. Backbone atoms are regarded as points $p_1, \ldots, p_6 \in \mathbb{R}^3$; the unknown dihedrals are

the angles $\omega_1, \ldots, \omega_6$ about axes (p_6, p_1) and (p_{i-1}, p_i) for $i = 2, \ldots, 6$.

Each of triangles $T_1 = \triangle(p_1, p_2, p_6)$, $T_2 = \triangle(p_2, p_3, p_4)$ and $T_3 = \triangle(p_4, p_5, p_6)$ is fixed for constant bond lengths $L_1, \ldots, L_6$ and bond angles ϕ_1, ϕ_3, ϕ_5. Then the lengths of (p_2, p_6), (p_2, p_4) and (p_4, p_6) are constant, hence *base* triangle $\triangle(p_2, p_4, p_6)$ is fixed in space, defining the xy-plane of a coordinate frame. Let θ_1 be the (dihedral) angle between the plane of T_1 and the xy-plane. Clearly, for any conformation, θ_1 is well-defined. Similarly we define angles θ_2 and θ_3, as shown in figure 1. We call them *flap* (dihedral) angles to distinguish them from the bond dihedrals.

Conversely, given lengths L_i, angles ϕ_i for $i = 1, \ldots, 6$ and flap angles θ_i for $i = 1, \ldots, 3$ the coordinates of all points p_i are uniquely determined and hence the bond dihedral angles and the associated conformation are all well-defined. We have therefore reduced the problem to computing the three flap angles θ_i which satisfy the constraints on bond angles ϕ_2, ϕ_4, ϕ_6.

Hence we obtain polynomial system

$$\begin{aligned}
\alpha_{11} + \alpha_{12}\cos\theta_2 + \alpha_{13}\cos\theta_3 + \alpha_{14}\cos\theta_2\cos\theta_3 + \alpha_{15}\sin\theta_2\sin\theta_3 &= 0,\\
\alpha_{21} + \alpha_{22}\cos\theta_3 + \alpha_{23}\cos\theta_1 + \alpha_{24}\cos\theta_3\cos\theta_1 + \alpha_{25}\sin\theta_3\sin\theta_1 &= 0,\\
\alpha_{31} + \alpha_{32}\cos\theta_1 + \alpha_{33}\cos\theta_2 + \alpha_{34}\cos\theta_1\cos\theta_2 + \alpha_{35}\sin\theta_1\sin\theta_2 &= 0,\\
\cos^2\theta_1 + \sin^2\theta_1 - 1 &= 0,\\
\cos^2\theta_2 + \sin^2\theta_2 - 1 &= 0,\\
\cos^2\theta_3 + \sin^2\theta_3 - 1 &= 0,
\end{aligned} \tag{1}$$

where the α_{ij} are input coefficients.

For our resultant solvers we prefer an equivalent formulation with a smaller number of polynomials, obtained by applying the standard transformation to half-angles that gives *rational* equations in the new unknowns t_i:

$$t_i = \tan\frac{\theta_i}{2} \;:\quad \cos\theta_i = \frac{1-t_i^2}{1+t_i^2},\ \sin\theta_i = \frac{2t_i}{1+t_i^2}, \qquad i = 1, 2, 3.$$

This transformation captures automatically the last three equations in (1). By multiplying both sides of the i-th equation by $(1+t_j^2)(1+t_k^2)$, where (i, j, k) is a permutation of $\{1, 2, 3\}$, the polynomial system becomes

$$\begin{aligned}
f_1 = \beta_{11} + \beta_{12}t_2^2 + \beta_{13}t_3^2 + \beta_{14}t_2t_3 + \beta_{15}t_2^2t_3^2 &= 0\\
f_2 = \beta_{21} + \beta_{22}t_3^2 + \beta_{23}t_1^2 + \beta_{24}t_3t_1 + \beta_{25}t_3^2t_1^2 &= 0\\
f_3 = \beta_{31} + \beta_{32}t_1^2 + \beta_{33}t_2^2 + \beta_{34}t_1t_2 + \beta_{35}t_1^2t_2^2 &= 0
\end{aligned} \tag{2}$$

where β_{ij} are input coefficients.

2.1.2 A formulation using distances

Another formulation of the problem relying on the geometry of distances is presented here. It aims to show how Computational Symbolic Geometry can help in easily deriving constraints for such problems. Another application of this approach can be found in [Hav91].

Let us first have a glimpse on the space of spheres. A sphere has a unique equation (up to a nonzero scalar) of the form

$$u_0(x^2 + y^2 + z^2) - 2\,u_1\,x - 2\,u_2\,y - 2\,u_3\,z + u_4 = 0, \tag{3}$$

therefore we can identify the space of spheres with a subset of the projective space $\mathbb{P}^4$ of classes $(u_0 : u_1 : \ldots : u_4)$. We also denote this projective space by $\mathcal{S}$.

Conversely, given a point $(u_0 : u_1 : \ldots : u_4) \in \mathcal{S} = \mathbb{P}^4$, if $u_0 \neq 0$, the quadric (3) defines a sphere of center $(1, \frac{u_1}{u_0}, \frac{u_2}{u_0}, \frac{u_3}{u_0})$ (or $(u_0 : \cdots : u_3)$ in $\mathbb{P}^3$) and radius R such that $R^2 = \frac{Q(u_0,\ldots,u_4)}{u_0^2}$ where $Q(u_0, \ldots, u_4) = u_1^2 + u_2^2 + u_3^2 - u_0 u_4$.

If $u_0 = 0$, the quadric obtained by homogenization of (3) is the union of an affine plane and the plane at infinity of $\mathbb{P}^3$ (defined by $t = 0$, where t is the homogenization variable in $\mathbb{P}^3$). It corresponds to a "point at infinity" in $\mathbb{P}^4$. Moreover, if $u_0 = u_1 = \ldots = u_3 = 0$, we obtain the special "sphere" $\omega = (0 : \ldots : 0 : 1)$, whose center is nowhere.

The special spheres of radius 0, will be called *point-spheres*. To any affine point $A = (1 : a_1 : a_2 : a_3) \in \mathbb{A}^3$, we associate the point-sphere $\dot{A} = (1 : a_1 : a_2 : a_3 : a_1^2 + a_2^2 + a_3^2)$. This gives an embedding of $\mathbb{A}^3$ in $\mathcal{S}$.

The space of spheres $\mathcal{S}$ has a natural inner-product given by the following formula: for any spheres $S = (u_0 : \cdots : u_4), S' = (u'_0 : \cdots : u'_4)$ in $\mathcal{S}$,

$$(S|S') \quad = \quad u_1u'_1 + u_2u'_2 + u_3u'_3 - \frac{1}{2}(u_0u'_4 + u_4u'_0).$$

It is not hard to check that if $u_0 = u'_0 = 1$, then

$$(S|S') = \frac{1}{2}(R^2 + R'^2 - d^2(O, O'))$$

where R, R' are the radii of the spheres, O, O' their centers and $d^2(O, O')$ the square of the distance between the two centers. Therefore, for two points $A, B \in \mathbb{A}^3$, the inner-product is

$$(\dot{A}|\dot{B}) = -\frac{1}{2}d^2(A, B) \tag{4}$$

and we also have $(\dot{A}|\dot{A}) = 0$. We also check that for any point $A \in \mathbb{A}^3$,

$$(\dot{A}|\omega) = -\frac{1}{2} \tag{5}$$

(see for instance [MS94] or [DH91] for more information).

Let us come back now to the configurations of $p_1, \ldots, p_6$ and let us denote by $\mathcal{L}$ the list of point-spheres $(\omega, \dot{p}_1, \ldots, \dot{p}_6)$. Consider the 7×7 symmetric matrix whose coefficient (i, j) is the inner-product of the i-th element of $\mathcal{L}$ and its j-th element. It is the *Gramm-Schmidt* matrix, constructed with the list $\mathcal{L}$ and the inner-product $(\,|\,)$. It is also called the *Cayley-Menger* matrix of the points $p_1, \ldots, p_6$ (see for instance [Ber77]). The spheres, being represented by a vector with 5 coordinates, it is not hard to check that this matrix is at most of rank 5 (spheres are represented by vectors with 5 coordinates). Using the equations (4), (5) and dividing by $-\frac{1}{2}$, we obtain a matrix of the form

$$\begin{array}{c} \\ \omega \\ p_1 \\ p_2 \\ p_3 \\ p_4 \\ p_5 \\ p_6 \end{array}
\begin{array}{c}
\begin{array}{ccccccc} \omega & p_1 & p_2 & p_3 & p_4 & p_5 & p_6 \end{array} \\
\left(\begin{array}{ccccccc}
0 & 1 & 1 & 1 & 1 & 1 & 1 \\
1 & 0 & c_{1,2} & c_{1,3} & \mathbf{u}_1 & c_{1,5} & c_{1,6} \\
1 & c_{1,2} & 0 & c_{2,3} & c_{2,4} & \mathbf{u}_2 & c_{2,6} \\
1 & c_{1,3} & c_{2,3} & 0 & c_{3,4} & c_{3,5} & \mathbf{u}_3 \\
1 & \mathbf{u}_1 & c_{2,4} & c_{3,4} & 0 & c_{4,5} & c_{4,6} \\
1 & c_{1,5} & \mathbf{u}_2 & c_{3,5} & c_{4,5} & 0 & c_{5,6} \\
1 & c_{1,6} & c_{2,6} & \mathbf{u}_3 & c_{4,6} & c_{5,6} & 0
\end{array} \right)
\end{array}.$$

In this matrix, all the coefficients $c_{i,j} = d(p_i, p_j)^2$ are known from the input parameters L_i, ϕ_i and there are only 3 unknowns $\mathbf{u}_1, \mathbf{u}_2, \mathbf{u}_2$ (just as in the previous section). Once these unknowns are *known*, we can recover the geometry of the molecule (up to global translations and rotations).

As any 6×6 minor of this matrix vanishes, we easily derive new constraints on the parameters $\mathbf{u}_i$. Taking for instance the diagonal minor of column (resp. row) indices $(1, 2, 3, 4, 5, 6)$ of this matrix, we obtain an equation $P_1(\mathbf{u}_1, \mathbf{u}_2) = 0$ in the variables $\mathbf{u}_1, \mathbf{u}_2$, which is of degree 2 in $\mathbf{u}_1$ and $\mathbf{u}_2$. In a similar way, we can derive a constraint $P_2(\mathbf{u}_1, \mathbf{u}_3) = 0$ (resp. $P_3(\mathbf{u}_2, \mathbf{u}_3) = 0$) of the same type by considering the diagonal minor of indices $(1, 2, 3, 4, 5, 7)$ (resp. $(1, 2, 3, 5, 6, 7)$). This yields a system of 3 equations in 3 unknowns

$$P_1(\mathbf{u}_1, \mathbf{u}_2) = 0, P_2(\mathbf{u}_1, \mathbf{u}_3) = 0, P_3(\mathbf{u}_2, \mathbf{u}_3) = 0$$

which has exactly the same set of monomials as system (2).

2.2 Geometric analysis

A first step in the analysis of a polynomial system consists in studying the geometry of common roots and in finding an accurate bound for the number of common solutions. We refer here to some recent advances in this area which exploit a priori knowledge of which monomials appear in the equations. We consider the algebraic closure of the given coefficient field, which is typically the field $\mathbb{C}$ of complex numbers.

The classical theory provides the bound by Bézout's theorem on the number of roots in the projective complex space, which is simply the product of the polynomials' total degrees [Sha77]. The bound given by Bézout's theorem here is $4 \times 4 \times 4 = 64$. This bound is too large, for the system has a very special shape. Not all monomials of degree 4 in the variables t_1, t_2, t_3 are present in the equations and the degree of each equation in each variable is bounded by 2. In other words, regarding only total degree overestimates the total number of solutions.

Sparse elimination theory, initiated with Bernstein's theorem [Ber75], provides a context for exploiting the monomial structure of the system, when roots at infinity are of no interest. To model sparseness, we regard each 3-dimensional exponent vector corresponding to a non-vanishing monomial in t_1, t_2, t_3 as a point in $\mathbb{Z}^3$. The convex hull of all such points defines the *Newton polytope* of the polynomial. The *mixed volume* of 3 polytopes $C_1, C_2, C_3 \in \mathbb{R}^3$ is a real-valued function that generalizes Euclidean volume, formally defined as *the coefficient* of $\lambda_1\lambda_2\lambda_3$ in the polynomial expressing the volume of $\lambda_1 C_1 + \lambda_2 C_2 + \lambda_3 C_3$. More generally, letting $V(\cdot)$ express the Euclidean volume function, we have the following definition: *For* $\lambda_1, \ldots, \lambda_n \in \mathbb{R}_{\geq 0}$ *and convex polytopes* $A_1, \ldots, A_n \subset \mathbb{R}^n$*, their* mixed volume *is precisely the coefficient of* $\lambda_1\lambda_2\cdots\lambda_n$ *in* $V(\lambda_1 A_1 + \cdots + \lambda_n A_n)$ *expanded as a polynomial in* $\lambda_1, \ldots, \lambda_n$.

Equivalent definitions for general dimension, together with an algorithm for computing mixed volumes, can be found in [EC95]. The code is publicly available at locations `http://www.inria.fr/safir/whoswho/emiris/`.

In our case the polytopes C_i are squares of size 2. Bernstein's theorem bounds the number of common roots with no zero coordinates by the mixed volume of the Newton polytopes. The mixed volume of the present system is 16, so Bernstein's theorem says that the number of isolated roots of system (2), counting multiplicities, with $t_1 \neq 0, t_2 \neq 0, t_3 \neq 0$ is at most 16. In fact, this bound is optimal, as shown in the next section by exhibiting an example with 16 solutions, which are moreover all real.

This method is, in our case, equivalent to a variant of Bézout's theorem for multi-homogeneous polynomials. If we homogenize each equation with respect to each variable t_1, t_2, t_3, we obtain a system of 3 multi-homogeneous equations in $\mathbb{P}^1 \times$

$\mathbb{P}^1 \times \mathbb{P}^1$ (where $\mathbb{P}^1$ is the projective space of dimension 1). The multi-degrees of these equations are

$$(0,2,2),(2,0,2),(2,2,0).$$

Using Bézout's theorem in $\mathbb{P}^1 \times \mathbb{P}^1 \times \mathbb{P}^1$, we can bound the number of isolated roots (not necessarily with no zero coordinates) of this system by the coefficient of $h_1\, h_2\, h_3$ in the polynomial

$$(2h_1 + 2h_2)\,(2h_1 + 2h_3)\,(2h_2 + 2h_3)$$

which also yields 16; see [Sha77].

2.3 Solving the equations

This section computes all common solutions for the algebraic system expressing the molecule, thus yielding all possible configurations. Our preferred method uses resultants in order to reduce the nonlinear problem derived in the previous section to a question in linear algebra. Matrix computations are then sufficient, for which we can rely on efficient and stable public implementations.

Let us concentrate on zero-dimensional varieties defined by a polynomial system of n equations in n unknowns. Let R be the polynomial ring and I the ideal of R generated by these n equations. It is known that when the number of roots is finite, as assumed here, the quotient ring $B = R/I$ forms a finite-dimensional vector space [Sha77]. This implies that defining the various operations in B, most notably multiplication, amounts to constructing the relevant matrices. Since finding the points in the variety of I essentially requires being able to compute in B, the abstract question is to compute the matrices expressing multiplication in B. Hence we transform the solution of the given polynomial system into an equivalent linear problem.

The crucial step is to define the matrix expressing multiplication by a variable in the quotient ring B. The matrix of multiplication by this variable has the property that each eigenvalue is equal to the coordinate of a solution vector corresponding to the variable of multiplication. Therefore an eigenvalue computation yields one coordinate of every solution vector. The rest of the coordinates are obtained by the eigenvectors of the multiplication matrix. Root finding is therefore reduced to an eigenproblem and then existing algorithms are employed from numerical linear algebra; sometimes this problem is regarded as the generalized eigenproblem or the problem of computing the matrix kernel vectors. It is not a new result in computational algebra, but it has not widely been used for practical purposes; the interested reader may consult any of [EC95, CM96, Emi96]. An important feature

of our method is precisely that it reduces to matrix operations for which powerful and accurate implementations already exist in the public domain, such as [ABB+92] which is used in the sparse resultant solver implemented in C and available from `http://www.inria.fr/safir/whoswho/emiris/`.

There are several algorithms to construct the matrices whose eigenvalues will give us the roots. A first class of methods is based on Macaulay-type matrices which rely on the idea that the system is generic in some sense [Emi94], and whose entries are linear in the polynomial coefficients. In the classical context, the resultant's degree is a function of the classical Bézout bound, i.e., the product of total degrees. The Macaulay matrix is defined for homogenized polynomials, and its size depends again on Bézout's bound.

In sparse elimination, the sparse resultant has degree dependent on the mixed volume of the Newton polytopes of the equations [GKZ94, PS93]. The sparse resultant approach generalizes the well-known Sylvester resultant for two univariate polynomials into several multivariate polynomials as well as Macaulay's algorithm for dense polynomials (see [Mac02]). Algorithms for constructing *sparse resultant matrices* (also called Newton matrices), whose determinant is a nontrivial multiple of the sparse resultant and which express multiplication in the quotient ring, can be found in [CE93, CP93, EC95]. The size of the Newton matrices scales with mixed volume [Emi96]. Linear algebra methods for reducing the matrix size and removing some genericity requirements, can be found in [Emi94].

The second class of methods relies on the theory of residues, and does not assume that the system is generic,but only that it is a complete intersection (the codimension of the set of solutions is the number of equations). The constructed matrices generalize the constructions proposed by Bézout for the resultant of two polynomials in one variable ([Béz79], see also [Dix08]) and are typically smaller than sparse resultant matrices but the coefficients are no longer linear in the coefficients of the input polynomials. This technique can be used for homogeneous or sparse polynomials. It relies on compression algorithms, reducing the size of the matrix to the optimal size (i.e. the number of roots, counted with multiplicity). The connections between algebraic residue theory and matrix methods for polynomial system solving are another interesting feature of this approach (see for instance [CM96], [EM96]).

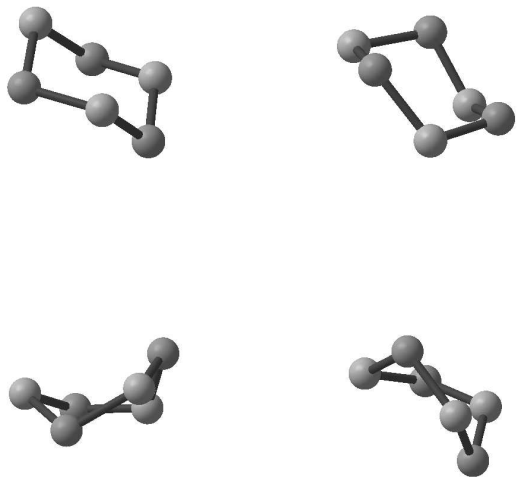
Thirdly, we simply apply Sylvester's resultant successively until all but one variables are eliminated. All three algebraic methods are illustrated in the next section, for the particular instances described below.

We treat three systems of type (2), and detail, in the specific context, the three matrix methods for system solving.

1. For the first one, we consider the *cyclohexane* molecule which has 6 carbon atoms at equal distances and equal bond angles. Usually noise enters in the process that produces the coefficients. To illustrate this phenomenon, starting with the pure cyclohexane, we randomly perturb the data by about 10% to obtain β_{ij} as the entries of matrix

$$\begin{bmatrix} -310 & 959 & 774 & 1389 & 1313 \\ -365 & 755 & 917 & 1451 & 269 \\ -413 & 837 & 838 & 1655 & 1352 \end{bmatrix}.$$

In this case, we find the following 4 solutions, visualized by IZIC [FKM94]. A chemist will immediately recognize two chair and two twisted boat, or crown, configurations



corresponding to the roots

t_1	t_2	t_3
0.3684363946	0.3197251270	0.2969559367
- 0.3684363946	- 0.3197251270	- 0.2969559367
0.7126464332	- 0.01038413185	- 0.6234532743
- 0.7126464332	0.01038413185	0.6234532743

.

2. The second instance has the maximum number of roots, namely 16, and furthermore, they are all *real*. The β_{ij} coefficients are given by matrix

$$\begin{bmatrix} -13 & -1 & -1 & 24 & -1 \\ -13 & -1 & -1 & 24 & -1 \\ -13 & -1 & -1 & 24 & -1 \end{bmatrix}.$$

Among the 16 real roots, four correspond to eigenvalues of unit (geometric) multiplicity, while the rest form four groups, each corresponding to a triple eigenvalue. For the latter the eigenvectors give us no valid information, so we recover the values of t_1, t_2 by substituting t_3 in the original system and by solving the resulting univariate equations.

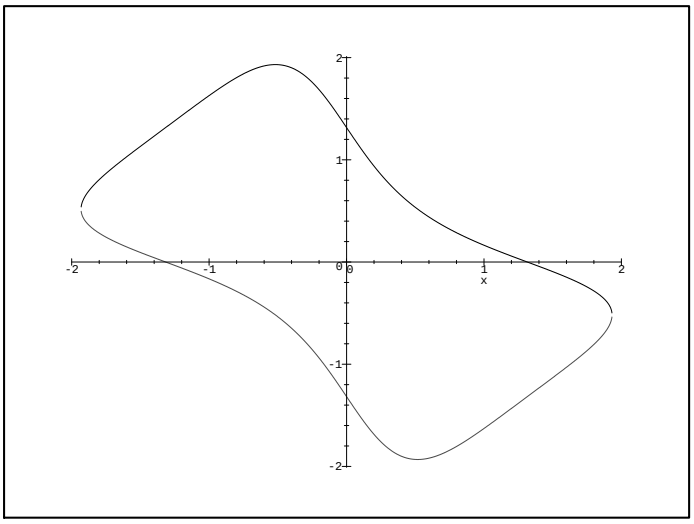
t_1	t_2	t_3
10.85770360	0.7795480449	0.7795480451
- 10.85770360	- 0.7795480449	- 0.7795480451
0.3320730984	4.625181601	4.625181601
- 0.3320730984	- 4.625181601	- 4.625181601
0.7795480449	0.7795480449	0.7795480451
0.7795480449	10.85770360	0.7795480451
0.7795480440	0.7795480457	10.85770360
4.625181601	4.625181601	4.625181601
4.625181601	0.3320730984	4.625181601
4.625181600	4.625181613	0.3320730984
- 0.7795480440	- 0.7795480457	- 10.85770360
- 0.7795480449	- 0.7795480449	- 0.7795480451
- 0.7795480449	- 10.85770360	- 0.7795480451
- 4.625181600	- 4.625181613	- 0.3320730984
- 4.625181601	- 4.625181601	- 4.625181601
- 4.625181601	- 0.3320730984	- 4.625181601

3. The third system is defined by the following matrix of β_{ij}.

$$\begin{bmatrix} -\frac{\sqrt{3}}{2} & \frac{1}{2} & \frac{1}{2} & 2 & \frac{\sqrt{3}}{2} \\ -\frac{\sqrt{3}}{2} & \frac{1}{2} & \frac{1}{2} & 2 & \frac{\sqrt{3}}{2} \\ -\frac{\sqrt{3}}{2} & \frac{1}{2} & \frac{1}{2} & 2 & \frac{\sqrt{3}}{2} \end{bmatrix}.$$

It is not a 0-dimensional system for it is the union of a curve of degree 4 whose projection on the coordinnate plan are defined respectively by the equation

$p_1 = 0$, $p_2 = 0$, $p_3 = 0$ and of 16 points. Among these 16 points, 4 points on the line $t_1 = t_2 = t_3$ are isolated (2 are real) and the other 12 are embedded in the curve. Here is a picture of the projection of the curve on any of the coordinate planes:



The system that we consider is obtained by taking the approximations of the coefficients with 5 digits. The roots that we obtain are

t_1	t_2	t_3
0.5176444559	0.5176444559	0.5176444563
-0.5176444567	-0.5176444567	-0.5176444555
0.5176444559	-1.931851652	0.5176444563
-0.5176444567	1.931851652	-0.5176444555
1.931851652	-0.5176444567	-0.5176444555
-1.931851652	0.5176444559	0.5176444563
0.5176444561	0.5176444561	-1.931851652
-0.5176444561	- 0.5176444561	1.931851652

The 6 last points are not almost embedded in the curve.

We describe now the methods that we have used.

2.3.1 Sparse resultants

Sparse multivariate resultants eliminate one fewer variables than the number of polynomials, all at once, and exploit any a priori knowledge on certain coefficients being zero. Since not all the monomials of degree 2 are present in the given equations, the sparse resultant has lower degree, and hence reduced complexity compared to the classical multivariate resultant (see [Mac02], [VdW48]). We apply this approach to all three systems and report on experimental results in section 2.3.4.

- The first approach in applying resultants for solving well-constrained systems "hides" one variable in the coefficient field. If we consider variable t_3 as part of the coefficient field, the 3 equations can be viewed as bivariate:

$$\begin{aligned} f_1 &= c_{11} + c_{12}t_2 + c_{13}t_2^2 = 0, \\ f_2 &= c_{21} + c_{22}t_1 + c_{23}t_1^2 = 0, \\ f_3 &= c_{31} + c_{32}t_2^2 + c_{33}t_1t_2 + c_{34}t_1^2 + c_{35}t_1^2t_2^2 = 0. \end{aligned} \tag{6}$$

  The resultant matrix of this system has dimension 16 and its entries c_{ij} are naturally functions of t_3; in fact, they are quadratic polynomials in t_3. The sparse resultant has total degree in the coefficients equal to the sum of the mixed volumes of the bivariate well-constrained subsystems. Each of the 3 mixed volumes (in t_1, t_2) is equal to 4, therefore the sparse resultant degree is 12 in the variables $c_{i,j}$ and the size of the constructed matrix will be at least 12. The shown 16×16 matrix M_s is the resultant matrix constructed by both greedy and incremental algorithms, respectively described in [CP93] and [EC95].

  Matrix M_s expresses a quadratic matrix polynomial in t_3, therefore its determinant is quadratic in t_3 and a multiple of the sparse resultant. Solving this determinant for t_3 gives us the values of t_3 at the roots, while the kernel vectors correspond to the values of t_1, t_2. More specifically, each kernel vector expresses the evaluation of the following vector of monomials at the roots of t_1, t_2. The monomial vector is precisely the sequence of monomials indexing the columns of M_s:

$$\left[1, t_2, t_2^2, t_2^3, t_1, t_1t_2, t_1t_2^2, t_1t_2^3, t_1^2, t_1^2t_2, t_1^2t_2^2, t_1^2t_2^3, t_1^3, t_1^3t_2, t_1^3t_2^2, t_1^3t_2^3\right].$$

  To recover the value of t_1, t_2 at the roots, it suffices to divide certain vector entries. To find, for instance, the value of t_2, we divide the second entry by the first, i.e., $t_2/1$. However innocuous this operation may seem, the choice

$$M_s = \begin{bmatrix}
c_{11} & c_{12} & c_{13} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & c_{11} & c_{12} & c_{13} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
c_{31} & 0 & c_{32} & 0 & 0 & c_{33} & 0 & 0 & c_{34} & 0 & c_{35} & 0 & 0 & 0 & 0 & 0 \\
0 & c_{31} & 0 & c_{32} & 0 & 0 & c_{33} & 0 & 0 & c_{34} & 0 & c_{35} & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & c_{11} & c_{12} & c_{13} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & c_{11} & c_{12} & c_{13} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & c_{31} & 0 & c_{32} & 0 & 0 & c_{33} & 0 & 0 & c_{34} & 0 & c_{35} & 0 \\
0 & 0 & 0 & 0 & 0 & c_{31} & 0 & c_{32} & 0 & 0 & c_{33} & 0 & 0 & c_{34} & 0 & c_{35} \\
c_{21} & 0 & 0 & 0 & c_{22} & 0 & 0 & 0 & c_{23} & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & c_{21} & 0 & 0 & 0 & c_{22} & 0 & 0 & 0 & c_{23} & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & c_{21} & 0 & 0 & 0 & c_{22} & 0 & 0 & 0 & c_{23} & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & c_{21} & 0 & 0 & 0 & c_{22} & 0 & 0 & 0 & c_{23} & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & c_{21} & 0 & 0 & 0 & c_{22} & 0 & 0 & 0 & c_{23} & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & c_{21} & 0 & 0 & 0 & c_{22} & 0 & 0 & 0 & c_{23} & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & c_{21} & 0 & 0 & 0 & c_{22} & 0 & 0 & 0 & c_{23} & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & c_{21} & 0 & 0 & 0 & c_{22} & 0 & 0 & 0 & c_{23}
\end{bmatrix}$$

of which entries to use is an important numerical issue that has to be studied further. Results vary significantly according to this choice, and certain simple strategies examined do not work. We have, namely, tried averaging over several ratios, e.g. $(t_2/1 + t_1 t_2/t_1)/2$, or taking roots, e.g. $\sqrt{t_2^2/1}$.

For the second system the specialized determinant is of degree 24:

$$-\frac{186624}{169}\left({t_3}^4 - 22\,{t_3}^2 + 13\right)^3 \left({t_3}^2 + 1\right)^4 \left({t_3}^4 - 118\,{t_3}^2 + 13\right).$$

- The second way to define an overconstrained system from the given one is to add to the given system any polynomial of our choice. This gives the u-resultant approach if the extra polynomial is linear in the variables with indeterminate constant term u. This method distinguishes among all simple roots at the expense of increasing the problem's dimension by 1.

  If we choose polynomial $a + b\,t_1$, with a, b indeterminate, the mixed volumes of the 3-polynomial systems are $4, 4, 4, 16$, hence the sparse resultant degree is 4 in the variables $\beta_{i,1}, \ldots, \beta_{i,5}$, 16 in the variables a, b and its total degree is 28 The incremental algorithm was applied and the smallest matrix found is of size 48. Its determinant, which is a multiple of the resultant, when *specialized* to the coefficients of the second system, yields the following polynomial of degree

22:

$$3869835264\,b^2\left(a^4-118\,b^2a^2+13\,b^4\right)\left(4\,a^2+b^2\right)^2\left(a^4-22\,b^2a^2+13\,b^4\right)^3.$$

The mixed volume information indicates that the degree of the specialized sparse resultant in each of a, b is 16. In fact, the specialized sparse resultant is

$$\left(a^4-118\,b^2a^2+13\,b^4\right)\left(a^4-22\,b^2a^2+13\,b^4\right)^3,$$

which shows that 12 roots will come in triplets with t_1 constant.

In the rest of the section we report on the application of the public-domain solver `http://www.inria.fr/safir/whoswho/emiris/`, in conjunction with hiding t_3 in the coefficient field. This solver is in C therefore it runs faster than the MAPLE solver examined in detail below. It constructs the matrix by the incremental algorithm with generic coefficients, therefore independent of the instance and the precision. The time for constructing the resultant matrix is less than half a second on the DEC AlphaStation of section 2.3.4. Then, the program constructs a 32×32 companion matrix whose eigenvalues and eigenvectors are the t_3 values and the kernel vectors of M_s. Since there are 32 eigenvalue and eigenvector pairs, exactly half of them will not correspond to solutions. The program applies the standard eigendecomposition routines of [ABB+92] over double-precision floating-point numbers, which have 53 binary digits reserved for the mantissa. It yields the following results on the first two instances; more information is included in [Emi94].

1. For the first system, after rejecting false candidates, the recovered roots cause the maximum absolute value of the input polynomials to be 10^{-5}. We check the computed solutions against those obtained by an exact Gröbner bases computation over the integers and observe that each contains at least 8 correct digits. The total CPU time on a SPARC is 0.2 seconds on average for transforming the matrix and computing its eigenvalue-eigenvector pairs.

2. For the second system, the computed roots evaluate the input polynomials to values no larger than 10^{-6} and, compared to the exact roots, they are correct to at least 7 decimal digits. The average CPU time for computing the roots from the resultant matrix is 0.2 seconds on a SPARC.

2.3.2 Bézout's resultants

We consider here two methods, analogous to those of the previous section.

$$M_b = \begin{bmatrix}
-14\,t_1^2 - 13 - t_1^4 & -2\,t_1^2 - 1 - t_1^4 & 24\,t_1 + 24\,t_1^3 & 0 \\
-7488\,t_1 - 864\,t_1^3 & 0 & 312 + 24\,t_1^4 + 336\,t_1^2 & 7488\,t_1^2 \\
24\,t_1 + 24\,t_1^3 & 0 & -14\,t_1^2 - 13 - t_1^4 & 11\,t_1^4 - 13 - 2\,t_1^2 \\
-240\,t_1^2 + 24\,t_1^4 + 312 & 0 & 0 & -288\,t_1^3 \\
13822\,t_1^2 + 11\,t_1^4 - 13 & -14\,t_1^2 - 13 - t_1^4 & -552\,t_1^3 - 264\,t_1 & -7488\,t_1 - 864\,t_1^3 \\
-552\,t_1^3 - 264\,t_1 & 24\,t_1 + 24\,t_1^3 & 24 + 24\,t_1^4 - 528\,t_1^2 & 312 + 24\,t_1^4 + 336\,t_1^2 \\
24\,t_1 + 24\,t_1^3 & 0 & -2\,t_1^2 - 1 - t_1^4 & -14\,t_1^2 - 13 - t_1^4 \\
-576\,t_1^2 & 0 & 24\,t_1 + 24\,t_1^3 & 24\,t_1^3 + 312\,t_1
\end{bmatrix}$$

$$\begin{bmatrix}
0 & 0 & 0 & 0 \\
24\,t_1^3 + 312\,t_1 & 11\,t_1^4 - 13 - 2\,t_1^2 & -288\,t_1^3 & -14\,t_1^2 - 13 - t_1^4 \\
24\,t_1 + 24\,t_1^3 & -24 - 24\,t_1^4 - 48\,t_1^2 & 24\,t_1 + 24\,t_1^3 & 0 \\
-14\,t_1^2 - 13 - t_1^4 & 24\,t_1 + 24\,t_1^3 & 11\,t_1^4 - 13 - 2\,t_1^2 & 0 \\
-576\,t_1^2 & 24\,t_1 + 24\,t_1^3 & -240\,t_1^2 + 24\,t_1^4 + 312 & 24\,t_1 + 24\,t_1^3 \\
24\,t_1 + 24\,t_1^3 & -14\,t_1^2 - 13 - t_1^4 & 0 & -2\,t_1^2 - 1 - t_1^4 \\
0 & 0 & 0 & 0 \\
-2\,t_1^2 - 1 - t_1^4 & 24\,t_1 + 24\,t_1^3 & -14\,t_1^2 - 13 - t_1^4 & 0
\end{bmatrix}$$

- First, we consider t_3 as a parameter and compute the Bezoutian of the 3 polynomial in the variables t_1, t_2. It yields the shown 8×8 matrix, whose entries are polynomials of degree ≤ 4 in t_3. For the second system, we obtain the matrix M_b given below:

  The determinant of matrix M_b is of degree 32:

  $$186624\,\left(t_3^4 - 118\,t_3^2 + 13\right)\left(t_3^4 - 22\,t_3^2 + 13\right)^3\left(t_3^2 + 1\right)^8$$

  The correct candidates for the solutions of the system are given by the roots of the first two factors. Four of them will be of multiplicity 3. This is the formulation used in the experiments reported in section 2.3.4.

- Secondly, we consider the variables t_1, t_2, t_3, but we add a generic linear form $a + b\,t_1$ and we compute the Bezoutian of these 4 polynomials. We obtain a 32×36 matrix, whose rank is 20. Using Bareiss's method, we obtain on the last row, the polynomial of degree 20 :

  $$-2579890176\,b^4\left(13\,b^4 - 118\,b^2a^2 + a^4\right)\left(13\,b^4 - 22\,b^2a^2 + a^4\right)^3.$$

  This is a multiple of the resultant, which has degree 16. The sparse resultant method, described above, had given another multiple of degree 22.

The time for computing the roots from the Bezoutian (by compression and eigenvector computations) is 0.1 seconds on a SPARC. We have used the library `ALP` (`http://www.inria.fr/safir/SAM/ALP/`, still under development).

2.3.3 Sylvester resultants

A simpler approach applies Sylvester's resultant in *cascade*. This is the resultant of two univariate polynomials and is implemented in most modern computer algebra systems, including AXIOM, MAPLE and MATHEMATICA.

The cascade method applies Sylvester's resultant several times, each time eliminating one variable. In general it produces matrices larger than with the multivariate resultant, which eliminates all the variables in one step. The example at hand, though, is favorable for this algorithm since each polynomial is independent of one variable. We can, thus, define S_{23} as the Sylvester matrix of f_2 and f_3, regarded as univariate polynomials in t_1 in the second instance of system (2),

$$S_{23} = \begin{bmatrix} -1-{t_3}^2 & 24\,t_3 & -13-{t_3}^2 & 0 \\ 0 & -1-{t_3}^2 & 24\,t_3 & -13-{t_3}^2 \\ -1-{t_2}^2 & 24\,t_2 & -13-{t_2}^2 & 0 \\ 0 & -1-{t_2}^2 & 24\,t_2 & -13-{t_2}^2 \end{bmatrix}$$

so the determinant $R_{23}(t_2, t_3) = \det S_{23}$ is a polynomial in t_2, t_3:

$$576\,{t_2}^4{t_3}^2 - 1152\,{t_2}^3{t_3}^3 + 576\,{t_2}^2{t_3}^4 + 144\,{t_2}^4 \\ -8064\,{t_2}^3t_3 + 15840\,{t_2}^2{t_3}^2 - 8064\,t_2{t_3}^3 + 144\,{t_3}^4 + 7488\,{t_2}^2 - 14976\,t_2t_3 + 7488\,{t_3}^2.$$

Then, the Sylvester resultant of R_{23} and f_1, regarded as polynomials in t_2, is a polynomial in t_3 and a multiple of the resultant of f_1, f_2, f_3. Letting S be the last Sylvester matrix and R its determinant,

$$R = 186624\,\left({t_3}^4 - 118\,{t_3}^2 + 13\right)\left({t_3}^4 - 22\,{t_3}^2 + 13\right)^3,$$

we can solve numerically for t_3, thus obtaining a superset of the values of t_3 at the common roots of the original system. Thanks to the special shape of this system, in this special case we get exactly the minimal condition in t_3, i.e., R above is the sparse resultant within a scalar factor. However, this is not usually the case. Recall that the sparse resultant has degree 16; note that the Newton and Bézout matrices have, respectively, led to nontrivial multiples of degrees 24 and 32.

	S_1				S_2				S_3			
D	T_1	T_2	T_3	ϵ	T_1	T_2	T_3	ϵ	T_1	T_2	T_3	ϵ
	7.97				7.44				7.74			
5		0.13	1.33	0.31		0.18	1.70	12.61		0.35	0.85	$4.11\,10^{-3}$
10		0.08	1.35	$2.15\,10^{-7}$		0.30	1.67	$1.14\,10^{-4}$		0.28	1.17	$4.17\,10^{-3}$
15		1.38	1.48	$1.49\,10^{-11}$		0.22	1.68	$1.55\,10^{-9}$		0.38	1.13	$4.17\,10^{-3}$
20		1.67	18.92	$5.58\,10^{-16}$		0.25	17.07	$7.51\,10^{-15}$		1.23	10.88	$4.17\,10^{-3}$
30		1.27	21.68	$1.87\,10^{-25}$		0.25	19.37	$2.73\,10^{-24}$		0.80	11.75	$4.17\,10^{-3}$
50		0.20	26.17	$1.93\,10^{-45}$		0.30	23.87	$3.51\,10^{-44}$		0.55	15.28	$4.15\,10^{-3}$

Table 1: Time and accuracy with sparse resultant matrices (method 2.3.1)

	S_1				S_2				S_3			
D	T_1	T_2	T_3	ϵ	T_1	T_2	T_3	ϵ	T_1	T_2	T_3	ϵ
5	0.19	2.0	0.35	0.18	0.21	0.72	2.05	0.32	0.30	2.82	0.43	$2\,10^{-4}$
10	0.18	2.05	0.50	10^{-6}	0.20	0.88	1.92	$9\,10^{-7}$	0.25	3.15	1.0	$1.48\,10^{-4}$
15	0.18	2.02	0.50	$3.1\,10^{-11}$	0.35	0.68	1.87	$3.0\,10^{-12}$	0.20	25.38	1.02	$1.48\,10^{-4}$
20	0.20	2.02	4.27	$4.5\,10^{-16}$	0.22	0.88	13.05	$6.5\,10^{-16}$	0.20	8.00	7.83	$1.48\,10^{-4}$
30	0.37	2.04	5.02	$3.9\,10^{-26}$	0.35	0.70	14.32	$9.0\,10^{-26}$	0.20	3.10	4.30	$1.48\,10^{-4}$
50	0.37	2.12	6.45	$2.1\,10^{-46}$	0.20	0.88	16.63	$4.4\,10^{-46}$	0.22	2.90	5.57	$1.48\,10^{-4}$

Table 2: Time and accuracy with Bézout's matrices (method 2.3.2)

For each root of R, we compute the kernel of S, yielding the corresponding values of t_2 at the roots. Knowing t_2 and t_3, we recover t_1 by computing the kernel of S_{23}. This is the approach on which we present experimental results in section 2.3.4.

If we wish to avoid solving univariate polynomial $R(t_3)$ (which is a rather unstable operation), we may build the companion matrix of S and compute its eigenvalue-eigenvector pairs. In practice, the kernel can be computed by a Singular Value Decomposition (SVD) of a singular matrix, namely S with t_3 specialized to the roots of R. These are essentially the pairs of eigenvalues and kernel vectors of S, and we obtain again the values of t_1, t_2. The same limitation as with resultant matrices in the previous subsection is also present here concerning eigenspaces of high dimension.

2.3.4 Results

We report on experimental results obtained on MAPLE V, on a DEC AlphaStation 200 4/233 with Spec ratings 157.7 92Int, 183.9 92FP and 96 MBytes of memory. All relevant MAPLE code is available by the authors or publicly available at `http://www.inria.fr/safir/whoswho/emiris/`, `http://www.inria.fr/safir/whoswho/Bernard.Mourrain/`. Several C programs used in this work can also be found in these locations.

- D is the number of decimal digits used in the computation. (precision),

	S_1				S_2				S_3			
D	T_1	T_2	T_3	ϵ	T_1	T_2	T_3	ϵ	T_1	T_2	T_3	ϵ
5	0.45	0.08	0.48	0.025	0.20	0.10	1.4	0.23	0.55	0.15	0.42	$1.3\,10^{-3}$
10	0.40	0.083	0.32	$4.9\,10^{-5}$	0.18	0.12	1.5	$8\,10^{-7}$	0.55	0.17	0.43	$1.5\,10^{-4}$
15	0.60	0.10	0.30	$4.8\,10^{-5}$	0.18	0.13	1.4	$2.2\,10^{-9}$	0.58	0.82	0.28	$1.5\,10^{-4}$
20	0.43	0.083	2.4	$1.8\,10^{-8}$	0.37	0.15	6.9	$5.2\,10^{-13}$	0.38	0.33	1.8	$1.5\,10^{-4}$
30	0.42	0.10	2.3	$8.6\,10^{-19}$	0.38	0.15	8.1	$5.3\,10^{-24}$	0.38	0.15	2.3	$8.4\,10^{-3}$
50	0.62	0.12	2.7	$2.6\,10^{-38}$	0.20	0.17	9.5	$7.6\,10^{-44}$	0.40	4.6	2.9	$2.2\,10^{-9}$

Table 3: Time and accuracy with Sylvester resultants (method 2.3.3)

- T_1 is the time for constructing the matrix(ces) in seconds, including simplifying these matrices through either Gaussian or Bareiss's fraction-free elimination and, finally, computing the determinant as a univariate polynomial. For the sparse resultant method this includes 6.12 seconds for building the generic-coefficient matrix, once for all systems.
- T_2 is the time for solving the univariate resultant polynomial in the hidden variable t_3 (in this case with `fsolve`) in seconds.
- T_3 is the time for computing the matrix kernel and then the other coordinates of the roots, or directly the latter, in seconds.
- ϵ is the maximal absolute value of the input polynomials at the computed roots.

For the method of sparse resultants, the matrix is computed only once by the greedy algorithm, for generic coefficients, in 6.12 seconds; this, obviously, does not depend on the chosen precision. For every system the coefficients are specialized, respectively, to integer polynomials in the hidden variable t_3. The time T_1 in the first line of table 1 includes the 6.12 seconds to construct the generic matrix, and the time of Gaussian elimination on the specialized matrix, which yields the determinant. The incremental algorithm for sparse resultant matrices is expected to be considerably faster in constructing the generic matrices, i.e. with respect to the timings in column T_1, which correspond to the original subidivision-based algorithm. Our preliminary Maple implementation suggests that this time would be *less than half* of the reported running times. This is intuitively obvious because the main step in the incremental method is a rank test on the resultant matrix, which takes about 2 seconds.

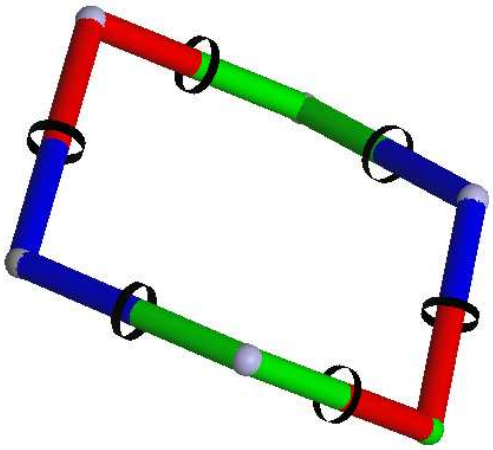
Gaussian pivoting is slightly slower than fraction-free elimination, but does not increase the size of the entries, so that the triangular matrix may be used in computing the kernel. To increase accuracy, the kernel is computed for the original matrix, which leads to an increase in running time.

In all three methods, we solve for the real roots of the univariate polynomial expressing a multiple of the resultant. We use the maple command `fsolve`, whose efficiency can probably be ameliorated. It takes about 2 seconds to solve a polynomial of degree 16 and for the last system, with 15 digits, more than 25 seconds. This strange variation is probably due to testing out several methods (by try and error) in the function `fsolve`. For each root, we substitute in the matrix and compute its kernel vector by SVD. When the root projections on the t_3-axis are not simple, the vectors do not necessarily correspond to the values of t_1, t_2. In this case, we substitute t_3 in the original system and solve the equations in t_1 and t_2 only. Some experience is required in choosing the threshold value that decides whether the vectors are valid. If not, we substitute directly in the original system and solve the resulting equations. Then a second threshold is used for deciding which candidate solutions to keep.

In conclusion, we observe that the Sylvester cascade method is fastest for the particular problem. This is explained by the equations' structure. However, it is not expected to happen in larger systems, unless they possess some similar strong structure. Nonetheless, the Sylvester method offers no advantage in terms of accuracy. The Bézout's matrices method yields smaller matrices, which are easier and faster to manipulate than the sparse resultant matrices, Accuracy of Bezout's method is slightly but not significantly better than the sparse resultant method on the example. On the other hand sparse resultant matrices are easier to instantiate than the Bézout matrices. The higher running times of the sparse resultant method are due to the large matrix size.

3 The critical locus

Obviously, there exits an analogy between molecular configurations and robotics or mechanics. Various structural requirements on molecules can be modeled as geometric or kinematic constraints, as in robotics. This analogy consists in associating a serial robot to a molecule, by considering the rotation ω_i around the axis L_i, as a rotation articulation between two solids S_{i-1} (centered at p_{i-1}) and S_i (centered at p_i), as illustrated in the following figure; for notation refer to figure 1.



The circles indicate the articulations of rotation in this robot. It has the particularity that two consecutive axes of articulation are intersecting at a point. The cyclic molecule corresponds to a situation where the last solid is in a fixed configuration with respect to the first one.

This analogy can be carried on further, in order to analyze the configurations that are infinitesimally unstable. For any solid S moved by a displacement $D(t)$ (where t is, for instance, time), the velocity of a point $M(t)$ of this solid is given by

$$d_t(M(t)) = V_0 + OM(t) \times \Omega$$

where O is the origin, V_0 the velocity of the origin, considered as point of the solid, $\times$ is the vector product of the physicists and Ω the angular velocity. In other words, (V_0, Ω) represents the cinematic torque. We will identify it with the element $\mathcal{I}$ of $\wedge^2\mathbb{E}$ (where $\mathbb{E}$ is the 4-dimensional vector space underlying the affine space $\mathbb{A}^3$ and $\wedge\mathbb{E}$ the exterior algebra on $\mathbb{E}$, see [Lan80, Bou70]):

$$\mathcal{I} = V_1\, e_2 \wedge e_3 - V_2\, e_1 \wedge e_3 + V_3\, e_1 \wedge e_2 + \Omega_1\, e_1 \wedge e_4 + \Omega_2\, e_2 \wedge e_4 + \Omega_3\, e_3 \wedge e_4,$$

where the $e_i = (0, \ldots, 0, 1, 0, \ldots, 0)$ are the vectors in the canonical basis. It represents the *infinitesimal* movement of the solid. For instance, for a rotation along an

axis L containing $A = (1 : a_1 : a_2 : a_3), B = (1 : b_1 : b_2 : b_3)$, we have $\Omega = \lambda \overrightarrow{AB}$ and $V_A = 0$. Thus the infinitesimal movement for this rotation is $\lambda A \wedge B \in \wedge^2\mathbb{E}$.

If $M = (1 : m_1 : m_2 : m_3)$, we check that the first 3 coordinates of $\mathcal{I} \wedge M$ (ie. the coefficients of $e_0 \wedge e_2 \wedge e_3$, $e_0 \wedge e_1 \wedge e_3$, $e_0 \wedge e_1 \wedge e_2$) are precisely the coordinates of $d_t(M(t))$. As we have $\mathcal{I} \wedge M \wedge M = 0$, the last coordinate (the coefficient of $e_1 \wedge e_2 \wedge e_3$ of $\mathcal{I} \wedge M$ is determined, when we know its 3 first coordinates and M. Thus, the velocity $d_t(M(t))$ can be identified with the vector $\mathcal{I} \wedge M \in \wedge^3\mathbb{E}$.

Consider now several solids connected by articulations of rotation along axes $L_1, \ldots, L_n$. The velocity of any point M of the last solid is the sum of the velocities of M attached to each solid. It is given by the formula

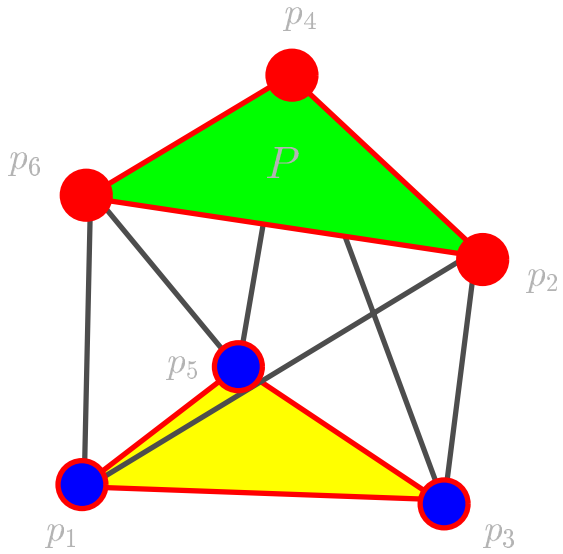
$$d_t(M(t)) = \sum_{i=1}^{N} \lambda_i \, L_i \wedge M.$$

Coming back to our problem of molecule (or robot), the serial robot will be unstable (see [MS94],[Mer90]), if the velocity of any point M in the last solid is 0, but the infinitesimal movement of the 6 solids is not zero. This means that we can fix the first and last solid and find however an infinitesimal motion of the chain of solids. In other words, the chain will be infinitesimally unstable if there exists $(\lambda_1, \ldots, \lambda_6) \neq (0, \ldots, 0)$ such that $\forall M, \sum_{i=1}^{N} \lambda_i \, L_i \wedge M$ or equivalently

$$\lambda_1 L_1 + \cdots + \lambda_6 L_6 = 0.$$

where $L_i = p_i \wedge p_{i+1} \in \wedge^2\mathbb{E}$ represent the axes of rotations of the robot.

Therefore, the cyclic molecule is infinitesimally unstable if these axes of rotations are linearly dependent in $\wedge^2\mathbb{E}$. We are going now to characterize geometrically, the configurations of the molecule which are infinitesimally unstable.



Let P be the plane (p_2, p_4, p_6). The linear system generated by $p_1 \wedge p_2$, $p_1 \wedge p_3$ is the same as the one generated by $p_1 \wedge p_2$, $p_1 \wedge p'_3$ where p'_3 is any point of the plane p_1, p_2, p_3 outside the line (p_1, p_2). In particular, we can take p'_3 on the plan P. Let us call this line $L'_1 = p_2 \wedge p'_3$. By the same operations at the points p_4, p_6, we can replace the system $(p_i \wedge p_{i+1})_{i=1\ldots6}$ by an equivalent system

$$p_1 \wedge p_2, p_3 \wedge p_4, p_5 \wedge p_6, L'_1, L'_2, L'_3$$

where L'_1 (resp. L'_2, L'_3) is the intersection of (p_1, p_2, p_3) (resp. (p_3, p_2, p_4), (p_5, p_4, p_6)) with P.

Note that if these 3 lines are concurrent, then they are linearly dependent in $\wedge^2\mathbb{E}$, for they are in a same plane. In this case, the 6 elements $p_1 \wedge p_2, p_3 \wedge p_4, p_5 \wedge p_6, L'_1, L'_2, L_3$. are also linearly dependent.

By construction, if L'_1, L'_2, L'_3 meet in a common point, so do the planes

$$(p_2, p_4, p_6), (p_1, p_2, p_3), (p_3, p_4, p_5), (p_5, p_6, p_1).$$

We can translate this condition, using the Cayley operators (see [BBR85]), in a polynomial in the determinants of the points p_i as follows:

$$\begin{aligned}&(p_2, p_4, p_6) \cap (p_1, p_2, p_3)\\ &\quad = \quad p_2 \wedge p_4[p_6, p_1, p_2, p_3] - p_2 \wedge p_6[p_4, p_1, p_2, p_3] + p_4 \wedge p_6[p_2, p_1, p_2, p_3]\end{aligned}$$

$$
\begin{aligned}
&(p_2,p_4,p_6)\cap(p_1,p_2,p_3)\cap(p_3,p_4,p_5)\\
&\quad = \quad -p_4[p_6,p_1,p_2,p_3][p_2,p_3,p_4,p_5]-p_2[p_4,p_1,p_2,p_3][p_6,p_3,p_4,p_5]\\
&\qquad +p_6[p_4,p_1,p_2,p_3][p_2,p_3,p_4,p_5]+p_4[p_2,p_1,p_2,p_3][p_6,p_3,p_4,p_5]
\end{aligned}
$$

which yields

$$
\begin{aligned}
&[p_4,p_1,p_5,p_6][p_6,p_1,p_2,p_3][p_2,p_3,p_4,p_5]+[p_2,p_1,p_5,p_6][p_4,p_1,p_2,p_3][p_6,p_3,p_4,p_5] \qquad (7)\\
&- \quad [p_6,p_1,p_5,p_6][p_4,p_1,p_2,p_3][p_2,p_3,p_4,p_5]-[p_4,p_1,p_5,p_6][p_2,p_1,p_2,p_3][p_6,p_3,p_4,p_5]=0
\end{aligned}
$$

where $[A,B,C,D]$ is the determinant (or oriented volume) of the 4 affine points A,B,C,D.

The determinant of the 6 axis $p_i \wedge p_{i+1}$ and this polynomial (7) are of the same degree (i.e. 2) in the coordinates each point p_i, and vanish on the same open subset of the unstable configurations $(p_1,\ldots,p_6)$. Therefore, they are equal within multiplication by a nonzero scalar.

Therefore, the necessary and sufficient conditions for which the molecule is unstable is given by the equation (7).

3.1 Acknowledgment

This work is partially supported by the European project Frisco (LTR 21.024).

References

[ABB+92] E. Anderson, Z. Bai, C. Bischof, J. Demmel, J. Dongarra, J. Du Croz, A. Greenbaum, S. Hammarling, A. McKenney, S. Ostrouchov, and D. Sorensen. *LAPACK Users' Guide.* SIAM, Philadelphia, 1992.

[BBR85] M. Barnabei, A. Brini, and G.C. Rota. On the exterior calculus of invariant theory. *J. of Algebra*, 96:120–160, 1985.

[Ber75] D.N. Bernstein. The number of roots of a system of equations. *Funct. Anal. and Appl.*, 9(2):183–185, 1975.

[Ber77] M. Berger. *Géométrie*, volume 5, L'espace des sphères. Cedic/Fernand Nathan, 1977.

[Béz79] E. Bézout. *Théorie générale des Équations Algébriques.* Paris, 1779.

[Bou70] N. Bourbaki. *Eléments de Mathématiques, Algèbre, Ch. 3.* Hermann, Paris, 1970.

[CE93] J. Canny and I. Emiris. An efficient algorithm for the sparse mixed resultant. In G. Cohen, T. Mora, and O. Moreno, editors, *Proc. Intern. Symp. Applied Algebra, Algebraic Algor. and Error-Corr. Codes, Lect. Notes in Comp. Science 263*, pages 89–104, Puerto Rico, 1993. Springer.

[CM96] J.-P. Cardinal and B. Mourrain. Algebraic approach of residues and applications. In J. Renegar, M. Shub, and S. Smale, editors, *Proc. AMS-SIAM Summer Seminar on Math. of Numerical Analysis, July 1995, Park City, Utah*, volume 32 of *Lectures in Applied Math.*, pages 189–210, 1996.

[CP93] J. Canny and P. Pedersen. An algorithm for the Newton resultant. Technical Report 1394, Comp. Science Dept., Cornell University, 1993.

[DH91] A.W.M. Dress and T.F. Havel. Distance geometry and Geometric algebra. *Foundations of Physics*, 23(10):1357–1374, October 1991.

[Dix08] A.L. Dixon. The eliminant of three quantics in two independent variables. *Proc. London Math. Society*, 6:49–69, 209–236, 1908.

[EC95] I.Z. Emiris and J.F. Canny. Efficient incremental algorithms for the sparse resultant and the mixed volume. *J. Symbolic Computation*, 20(2):117–149, August 1995.

[EM96] M. Elkadi and B. Mourrain. Approche effective des résidus algébriques. Technical Report 2884, INRIA Sophia-Antipolis, France, 1996.

[Emi94] 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.

[Emi96] I.Z. Emiris. On the complexity of sparse elimination. *J. Complexity*, 12:134–166, 1996.

[FKM94] R. Fournier, N. Kajler, and B. Mourrain. Visualization of mathematical surfaces: the IZIC server approach. *J. Symbolic Computation*, 19:159–173, 1994.

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

[GS70] N. Gō and H.A. Scheraga. Ring closure and local conformational deformations of chain molecules. *Macromolecules*, 3(2):178–187, 1970.

[Hav91] T.F. Havel. An evaluation of computational strategies for use in the determination of protein structure from distance constraints obtained by nuclear magnetic resonance. *Pog. Biophys. Molec. Biol.*, 56:43–78, 1991.

[Lan80] S. Lang. *Algebra*. Addison-Wesley, 1980.

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

[MC92] D. Manocha and J. Canny. Real time inverse kinematics of general 6R manipulators. In *Proc. IEEE Conf. Robotics and Automation*, pages 383–389, Nice, May 1992.

[Mer90] J.P. Merlet. *Les robots parallèles*. Traités de nouvelles technologiques. Hermes, 1990.

[MS94] B. Mourrain and N. Stolfi. *Invariants methods in Discrete and Computational Geometry*, chapter Computational Symbolic Geometry, pages 107–139. Kluwer Acad. Pub., 1994.

[MZW94] D. Manocha, Y. Zhu, and W. Wright. Conformational analysis of molecular chains using nano-kinematics. In *Proc. 1st IEEE Workshop on Shape and Pattern Recognition in Computational Biology*, pages 3–23, Seattle, 1994. Also Tech. Report 94-036, Dept. of Computer Science, Univ. of North Carolina.

[PC94] D. Parsons and J. Canny. Geometric problems in molecular biology and robotics. In *Proc. 2nd Intern. Conf. on Intelligent Systems for Molecular Biology*, pages 322–330, Palo Alto, CA, August 1994.

[PS93] P. Pedersen and B. Sturmfels. Product formulas for resultants and Chow forms. *Math. Zeitschrift*, 214:377–396, 1993.

[Sha77] I.R. Shafarevich. *Basic Algebraic Geometry*. Springer, Berlin, 1977.

[VdW48] B.L. Van der Waerden. *Modern algebra Vol II*. Frederick Ungar Publishing Co, 1948.

Unité de recherche INRIA Lorraine, Technopôle de Nancy-Brabois, Campus scientifique,
615 rue du Jardin Botanique, BP 101, 54600 VILLERS LÈS NANCY
Unité de recherche INRIA Rennes, Irisa, Campus universitaire de Beaulieu, 35042 RENNES Cedex
Unité de recherche INRIA Rhône-Alpes, 655, avenue de l'Europe, 38330 MONTBONNOT ST MARTIN
Unité de recherche INRIA Rocquencourt, Domaine de Voluceau, Rocquencourt, BP 105, 78153 LE CHESNAY Cedex
Unité de recherche INRIA Sophia Antipolis, 2004 route des Lucioles, BP 93, 06902 SOPHIA ANTIPOLIS Cedex

Éditeur
INRIA, Domaine de Voluceau, Rocquencourt, BP 105, 78153 LE CHESNAY Cedex (France)
ISSN 0249-6399