Numerical Irreducible Decomposition using PHCpack*

Andrew J. Sommese[1], Jan Verschelde[2], and Charles W. Wampler[3]

[1] Department of Mathematics, University of Notre Dame,
Notre Dame, IN 46556-4618, USA.
Email: sommese@nd.edu. *URL:* http://www.nd.edu/~sommese.
[2] Department of Mathematics, Statistics, and Computer Science,
University of Illinois at Chicago, 851 South Morgan (M/C 249),
Chicago, IL 60607-7045, USA.
Email: jan@math.uic.edu or jan.verschelde@na-net.ornl.gov.
URL: http://www.math.uic.edu/~jan.
[3] General Motors Research Laboratories, Enterprise Systems Lab,
Mail Code 480-106-359, 30500 Mound Road, Warren, MI 48090-9055, USA.
Email: Charles.W.Wampler@gm.com.

Abstract. Homotopy continuation methods have proven to be reliable and efficient to approximate all isolated solutions of polynomial systems. In this paper we show how we can use this capability as a blackbox device to solve systems which have positive dimensional components of solutions. We indicate how the software package PHCpack can be used in conjunction with Maple and programs written in C. We describe a numerically stable algorithm for decomposing positive dimensional solution sets of polynomial systems into irreducible components.
2000 Mathematics Subject Classification. Primary 65H10; Secondary 13P05, 14Q99, 68N30, 68W30.
Key words and phrases. homotopy continuation, interfaces, numerical algebraic geometry, polynomial system, software.

1 Introduction

Using numerical algorithms to solve polynomial systems arising in science and engineering with tools from algebraic geometry is the main activity in "Numerical Algebraic Geometry." This is a new developing field on the crossroads of algebraic geometry, numerical analysis, computer science and engineering. One of the key problems in this area (and also in Computational Algebraic Geometry [10]) is to decompose positive dimensional solution sets into irreducible components. A special instance of this problem is the factoring of polynomials in several variables.

* The authors gratefully acknowledge the support of this work by Volkswagen-Stiftung (RiP-program at Oberwolfach), where the writing for this paper began. The first author is supported in part also by NSF Grant DMS - 0105653; and the Duncan Chair of the University of Notre Dame. The second author is supported in part by NSF Grant DMS-0105739.

With the dictionary in Table 1 we show how to translate the key concepts of algebraic geometric to define an irreducible decomposition into data structures used by numerical algorithms. For each irreducible component we find as many generic points as the degree of the component. With this set of generic points we construct filters that provide a probability-one test to decide whether any given point belongs to that component.

We say that an algorithm is a probability-one algorithm if the algorithm depends on a choice of a point in an irreducible variety X and the algorithm works for a Zariski open dense set points $U \subset X$.

For example, consider that we want to check whether a polynomial $p(x)$ on $\mathbb{C}$ is identically zero, We might have as our algorithm: take an explicit random $x_* \in \mathbb{C}$ and check if $p(x_*) = 0$. Then $p(x)$ is identically zero if $p(x_*) = 0$. Here $X := \mathbb{C}$. The algorithm fails precisely when $p(x)$ is not identically zero and $p(x_*) = 0$. Since $p(x)$ is not identically zero, $p^{-1}(0)$ is finite and we need to choose $x_* \in U := \mathbb{C} \setminus p^{-1}(0)$. We are assuming that a random point on $\mathbb{C}$ will not lie in $p^{-1}(0)$. Of course, since we are working with machine numbers, there is an exceedingly small chance we will be wrong, e.g., see [34]. Usually we choose not one but many constants in an algorithm, e.g., the coefficients of the equation of a random hyperplane. In this case, the point will be the vector made up of the coefficients.

Numerical Algebraic Geometry Dictionary		
Algebraic Geometry	example in 3-space	Numerical Analysis
variety	collection of points, algebraic curves, and algebraic surfaces	polynomial system + union of witness point sets, see below for the definition of a witness point
irreducible variety	a single point, or a single curve, or a single surface	polynomial system + witness point set + probability-one membership test
generic point on an irreducible variety	random point on an algebraic curve or surface	point in witness point set; a witness point is a solution of polynomial system on the variety and on a random slice whose codimension is the dimension of the variety
pure dimensional variety	one or more points, or one or more curves, or one or more surfaces	polynomial system + set of witness point sets of same dimension + probability-one membership tests
irreducible decomposition of a variety	several pieces of different dimensions	polynomial system + array of sets of witness point sets and probability-one membership tests

Table 1. Dictionary to translate algebraic geometry into numerical analysis.

The goal of this paper is to elaborate the sentence "all algorithms have been implemented as a separate module of PHCpack" in recent papers [34–38], a sequence that has its origin in [39]. PHCpack [42] is a general purpose numerical solver for polynomials systems. In this paper we describe the extensions to this package, with a special emphasis on interfaces.

Our main tool is homotopy continuation, which has proven to be reliable and efficient to compute approximations to all isolated solutions of polynomials systems. Nowadays [26], this computation involves a combination of computational geometry techniques to calculate the mixed volume of the tuple of Newton polytopes of the polynomial system and numerical methods to follow the solution paths defined by the homotopies.

After outlining the design of PHCpack, reporting on an interface to fast mixed volume calculations, we describe a simple Maple procedure to call the blackbox solver of PHCpack. Via sampling and projecting we obtain a numerical elimination procedure. The algorithms are numeric-symbolic: with numerical interpolation we construct equations to represent the solution components. In section four we illustrate this sampling on a three dimensional Burmester linkage, using Maple as plotting tool. Since factoring of polynomials in several variables is a special instance of our general decomposition algorithms, we developed a low level interface to call the Ada routines from C programs. Section six explains the membership problem and our homotopy test to solve it. This test is then used as one of the tools in the decomposition method, illustrated in section seven. In the last section we list some of our major benchmark applications.

2 Toolbox and Blackbox design of PHCpack

PHCpack [42] was designed to test new homotopy algorithms to solve systems of polynomial equations. In particular, three classes of homotopies [43] have been implemented. For dense polynomial systems, refined versions of Bézout's theorem lead to linear-product start systems [45,47]. Polyhedral homotopies [46,48] are optimal for generic sparse polynomial systems. The third class contains SAGBI [44] and Pieri homotopies [23] implementing a Numerical Schubert Calculus [21].

These three classes of homotopies can be accessed directly when the software is used in toolbox mode. For general purpose polynomial systems, a blackbox solver was designed and tested on a wide variety of systems [42]. Although a blackbox will rarely be optimal and can therefore not beat the particular and sophisticated uses offered by a toolbox, both a toolbox and a blackbox are needed. The writing of PHCpack reflects the dual use of the software: on the one hand as a *pack*age, offering a library of specialized homotopy algorithms, and on the other hand as a blackbox, where only the executable version of the program, i.e.: *PHC*, is needed.

In this paper we focus on recent developments and interfaces. Combining both the recent research and use of other software, we report on an experiment with PHCpack for which a customized interface to the software of T.Y. Li and Li Xing [27] was used to compute generic points on the two dimensional components of cyclic 9-roots. In the next two paragraphs we provide a situation for the polyhedral methods.

To solve polynomial system with homotopies we deform a start system with known solutions to the system we wish to solve, applying Newton's method to track the solution paths. If we want to approximate all isolated solutions, then one major problem is the construction of a "good" start system that leads to an optimal number of paths. As shown in [1], mixed volumes of the tuples of convex hulls of the exponent sets of the polynomials provide a generically sharp bound for the number of isolated solutions. The theorems of Bernshtein [1] led to polyhedral homotopies [22], [48], and to a renewed interest in resultants, spurred by the field sparse elimination theory [7], [12], [13] and [32]; see also [40].

Mixed volumes for the cyclic n-roots problems were first computed in [12] and [13]. In [14], the cyclic 9-roots problem was reported to be solved with Gröbner bases. This problem has two dimensional components of solutions. Following [34], an embedding of the original polynomial system was constructed.

The first time we used the interface to the software of T.Y. Li and Li Xing was to find all 18 generic points on the two dimensional components of the cyclic 9-roots problem. This type of interface was also used to compute all 184,756 isolated cyclic 11-roots with PHC. The 8,398 generating solutions (with respect to the permutation symmetry) are on display, via the web site of the second author. Note that cyclic 11-roots is numerically "easier" as all solutions are isolated.

Besides [27], other recent computational progress is described in [16,17] and [41], with application to cyclic n-roots in [8]. Polyhedral methods led to a computational breakthrough in approximating all isolated solutions. To deal with positive dimensional solution sets we solve an embedded system that has all its roots isolated. Thus the recent activity on polyhedral root counting methods is highly relevant to the general solution method.

3 A Maple Interface to PHCpack

In this section we describe how to use the blackbox solver of PHCpack from within a Maple worksheet. The interface is file oriented, just like PHC was used with OpenXM [28].

The blackbox solver offered by PHCpack can be invoked on the command line as

```
phc -b input output
```

with `output` the name of the file for the results of the computation. The file `input` contains a system in the format

```
2
  x*y - 1;
  x**2 - 1;
```

where the “2” at the beginning of the file indicates the number of equations and unknowns. We point out that this blackbox only attempts to approximate the isolated solutions.

To bring the solutions into a Maple worksheet, PHC was extended with an addition option `-z`. The command

```
phc -z output sols
```

takes the output of the blackbox solver and creates the file `sols` which contains the list of approximations to all isolated solutions in a format readable by Maple.

The simple Maple procedure listed below has been tested for Maple6 on Linux and Maple7 on Windows 2000. A MapleV version runs on Solaris machines. Besides a list of polynomials, the user should provide a path name for the executable version of PHC.

```
run_phc := proc(phcloc::string,p::list)
  description `Calls phc from Maple6+7 session. \
     The name of file with the executable version of phc \
     should be provided in the string phcloc. \
     The second input argument p is a list of polynomials. \
  On return is a list of approximations to all isolated \
     roots of the system defined by p.`:
  local i,n,sp,semcol,sr,infile,outfile,solfile,cmd1,cmd2,sols:
  n := nops(p):                    # number of polynomials
  semcol := `;`:
  sr := convert(rand(),string):  # to randomize file names
  infile := input||sr: outfile := output||sr:
  solfile := sols||sr:
  fopen(infile,WRITE):
  fprintf(infile,`%d\n`,n):
  for i from 1 to n do
    sp := convert(p[i],string):
    sp := ` `||sp||semcol:    # append semicolon (dot in MapleV)
    fprintf(infile,`%s\n`,sp):
  od:
  fclose(infile):
  cmd1 := phcloc||` -b `||infile||` `||outfile:
  cmd2 := phcloc||` -z `||outfile||` `||solfile:
```

```
  ssystem(cmd1): ssystem(cmd2):
  read(solfile):
  sols := %:        # use " instead of % in older Maple versions
  fremove(infile): fremove(outfile): fremove(solfile):
  RETURN(sols);
end proc:
```

As pointed out earlier, we can only obtain approximations to all isolated solutions with this blackbox. The aim of the remainder of this paper is to sketch the ideas of the algorithms needed in a toolbox to describe also the positive dimensional solution components.

4 Numerical Elimination Methods

Recently we extended the use of homotopies from "just" approximating all isolated roots to describing all solution components of any dimension. In this section we introduce our approach by example. The system

$$\begin{cases} y - x^2 = 0 \\ z - x^3 = 0 \end{cases} \tag{1}$$

defines the so-called "twisted cubic" as the intersection of a quadratic and cubic surface.
For this example, we distinguish three possible orders of elimination:

1. projection onto the (x, y)-plane gives $y - x^2 = 0$;
2. projection onto the (x, z)-plane gives $z - x^3 = 0$;
3. projection onto a random plane gives a cubic curve.

To eliminate we first sample generic points from the curve using the system

$$\begin{cases} y - x^2 = 0 \\ z - x^3 = 0 \\ ax + by + cz + d = 0 \end{cases} \tag{2}$$

where the constants a, b, c, and d are randomly chosen complex numbers. For any general choice of (a, b, c, d) we get exactly three regular solutions which are generic points on the twisted cubic. Moving the last equation of (2) we generate as many samples as desired.

To eliminate z properly (to get the symbolic outcome $y - x^2 = 0$), the last equation of (2) must be parallel to the (x, y)-plane, with a zero coefficient for z. Similarly, for a proper elimination of y, the last equation of (2) must be parallel to the (x, z)-plane, with a zero coefficient for y. To project onto a random plane, we multiply the samples with a random complex two-by-three matrix. Interpolating through the projected samples, we obtain equations as result of the elimination.

The next example comes from mechanical design. Suppose N placements of a rigid body in space are given and consider the associated positions, $\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_N$, occupied by a designated point, $\mathbf{x}$, of the body. For four general placements, $\mathbf{x}_1, \ldots, \mathbf{x}_4$ define a sphere having a center point, say $\mathbf{y}$, and this is true for any general point $\mathbf{x}$ of the body. However, for five general placements, $\mathbf{x}_1, \ldots, \mathbf{x}_5$ lie on a sphere only if $\mathbf{x}$ is on a certain surface within the body, and for six general placements, $\mathbf{x}_1, \ldots, \mathbf{x}_6$ lie on a sphere only if $\mathbf{x}$ is on a certain curve in the body. Seven general positions determine 20 center-point/sphere-point pairs, a result proven by Schönflies at the end of the nineteenth century [6]. These points are of interest because we may build a linkage to guide the body through the given placements by connecting a rigid link between point $\mathbf{x}$ and its center $\mathbf{y}$. In the following, we consider the center-point/sphere-point curve arising when only six placements are given.

The polynomial system is given by five quadratic equations in six variables: $\mathbf{x} = (x_1, x_2, x_3)$ the coordinates on the sphere and $\mathbf{y} = (y_1, y_2, y_3)$ points on the centerpoint curve. The equations of the system have the following form:

$$||R_i\mathbf{x} + \mathbf{p}_i - \mathbf{y}||^2 - ||R_0\mathbf{x} + \mathbf{p}_0 - \mathbf{y}||^2 = 0, \quad i = 1, 2, \ldots, 5, \qquad (3)$$

where $\mathbf{p}_i \in \mathbb{R}^3$ are positions of the body and we use $R_i \in \mathbb{R}^{3\times3}$, $R^T R = I$, to denote the rotation matrices for the orientations of the bodies. The problem is to find values for the unknown variables $(\mathbf{x}, \mathbf{y}) \in \mathbb{C}^6$, given the positions and orientations of the body, encoded respectively by $\mathbf{p}_i \in \mathbb{R}^3$ and $R_i \in \mathbb{R}^{3\times3}$. In our experiment the values for $\mathbf{y}$ are separated from those for $\mathbf{x}$ and we construct line segments between the coordinates. Figure 1 shows part of the ruled surface made by Maple.

The data for Figure 1 was generated as follows. While the degree of the solution curve is twenty and we are guaranteed to find twenty generic points as isolated solutions when we add one random *complex* hyperplane to the original system, we cannot be sure to find twenty real points when we use a random *real* hyperplane. Actually, all generic points may still have nonzero imaginary parts, but fortunately we found four real generic points. Starting at the first real generic point, PHC sampled 100 points, moving the constant coefficient in the real hyperplane from 0.0 with steps of size 0.1. Observe that in the Maple session we only use the first 14 samples as the curve started moving too fast for the fixed step size of 0.1.

This last point illustrates that it does not suffice to have a one way communication to export samples for plotting. The plotting program (in our case Maple) must be able to take an active role to control the step size, or equivalently, one should change the sampling for plotting purposes.

Finally, one may wonder about the degrees of the curves for $\mathbf{x}$ and $\mathbf{y}$ in Figure 1. Like with the special planes to cut the twisted cubic, the degree of the curve drops when the hyperplane is parallel to the $\mathbf{x}$ or $\mathbf{y}$ coordinates. In particular, the degree drops from twenty to ten when such special hyperplanes are chosen.

```
> read sbr100x: read sbr100y:  # samples
> with(plottools):
> a := 1: b := 14:
```

The curve for x appears in dashed lines, the curve for y is drawn in solid lines :

```
> x := curve(xl1[a..b],linestyle=4,thickness=3,color=black):
> y := curve(yl1[a..b],thickness=3,color=black):
> T1 := plots[textplot3d]([-.5,-.3,.8,‘curve
  x‘],align=LEFT,color=black):
> T2 := plots[textplot3d]([0,0.3,0.2,‘curve
  y‘],align=RIGHT,color=black):
> l := []:
> for i from a to b do
>   l := [op(l),line(xl1[i],yl1[i],thickness=2)]:
> od:
> plots[display](x,y,T1,T2,l,axes=BOXED);
```

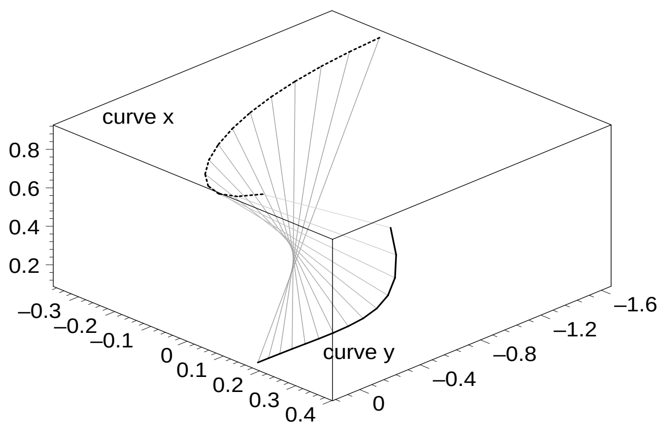

Fig. 1. A Maple session to draw a piece of a ruled surface, from samples generated by PHC.

5 Factoring into Irreducible Components

For the spatial Burmester problem above, it is natural to wonder if the solution curve is a single irreducible piece or whether it breaks up into several irreducible components.

The decomposition of positive dimensional solution sets into irreducible components is similar to that of decomposing (multivariate) polynomial into irreducible factors. We emphasize that our algorithms are closer to geometry than to algebra. We allow the input polynomials to have complex coefficients, and view those as complex multi-valued functions, see Figure 2.

The algorithm to decompose solution sets into irreducible has following specifications:

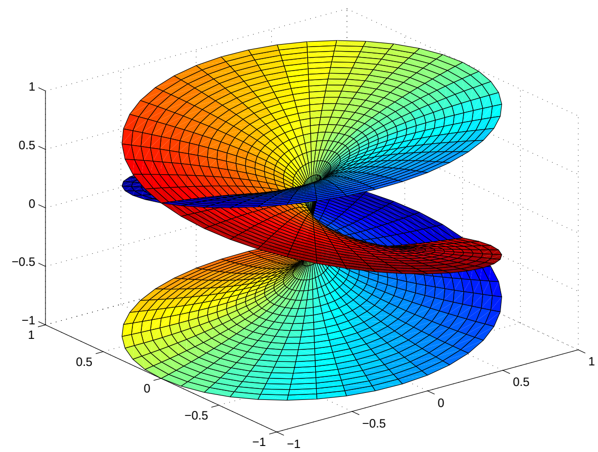

Fig. 2. The Riemann surface produced by `cplxroot` of MATLAB plots $\mathrm{Re}(z)$ with z satisfying $z^3 - x = 0$, for $x \in \mathbb{C}$. These solutions form a connected component. Observe that a loop of x around the origin permutes the order of the three solution points.

Input: A set of witness points on a positive dimensional solution set.
Output: A partition of the set; points in same subset of the partition belong to the same irreducible component.

To find out which points lie on the same irreducible component, we use homotopies to make the connections. For a system $f(\mathbf{x}) = \mathbf{0}$ we cut out witness points by adding a set of hyperplanes $L(\mathbf{x}) = \mathbf{0}$. The number of hyperplanes in L equals the dimension of the solution set. With the homotopy

$$h_{KL}(\mathbf{x}, t) = \lambda \begin{pmatrix} f \\ K \end{pmatrix} (1-t) + \begin{pmatrix} f \\ L \end{pmatrix} t = \mathbf{0}, \quad \lambda \in \mathbb{C}, \tag{4}$$

we find new witness points on the hyperplanes $K(\mathbf{x}) = \mathbf{0}$, starting at those witness points satisfying $L(\mathbf{x}) = \mathbf{0}$, letting t move from one to zero.

The role of the random complex constant λ is very important in what follows. Suppose we move back from K to L, using the homotopy

$$h_{LK}(\mathbf{x}, t) = \mu \begin{pmatrix} f \\ L \end{pmatrix} (1-t) + \begin{pmatrix} f \\ K \end{pmatrix} t = \mathbf{0}, \quad \mu \in \mathbb{C}, \tag{5}$$

using some random constant $\mu \neq \lambda$.

After using $h_{KL}(\mathbf{x}, t) = \mathbf{0}$ and $h_{LK}(\mathbf{x}, t) = \mathbf{0}$, we moved the hyperplanes added to $f(\mathbf{x}) = \mathbf{0}$, from L to K and back from K to L. The set of solutions of $f(\mathbf{x}) = \mathbf{0}$ at L is the same with the important difference that the order of the solutions in the set may have been changed. In particular, we obtain a permutation from the witness points in the set. Points that are permuted (e.g., the first point went to the third, the third to the second, and the second to the first), belong the same irreducible component, as illustrated on Figure 2.

The example above contains the main idea of the monodromy breakup algorithm developed in [36]. While the connections found in this way indicate which points lie on the same irreducible component, we may miss certain loops. Therefore, we recently developed an efficient test [38], based on the linear trace, to certify the partitions obtained by the monodromy algorithm. Besides the efficiency, the test is numerically stable as it only requires to set up a linear function.

For the spatial Burmester problem, the solution components was found to be an irreducible curve of degree 20.

At this stage we wish to show how a low level interface to the numerical factorization routines with PHC is planned. PHCpack is developed in Ada 95, which provides mechanisms to interface with other languages, such as Fortran, C, and Cobol. In our example we wish to call the factorization routines in Ada from a C program. This multi-lingual programming is supported by the gcc compilation system. A polynomial is represented in human readable format like Maple input, e.g. for $x^2 - y^2$, we use the string `"x**2 - y**2;"`, using the semicolon as terminator (also according the conventions in Maple). The sample C program is listed below.

```
#include <stdio.h>
extern char *_ada_phc_factor(int n, char *s);
extern void adainit();
extern void adafinal();
int main() {
   char *f;
   adainit();
   f = _ada_phc_factor(2,"x**2-y**2;");
   adafinal();
   printf("The factors are %s \n",f);
}
```

A more elaborate interface to C programs allows to pass numerical data (such as matrices of doubles), directly from C to Ada. This kind of interface is currently under construction.

Also in [15] the authors propose monodromy to factor multivariate polynomials numerically.

6 A Membership Test

From the dictionary in Table 1 we see that a membership test figures prominently. Traditional uses of homotopies discard roots as nonisolated when the

Jacobian matrix is sufficiently close to being singular. But this approach fails in the presence of isolated roots of multiplicity greater than one.
The membership problem can be formulated as follows:

Given: A solution $\mathbf{x} \in \mathbb{C}^n$ of $f(\mathbf{x}) = \mathbf{0}$ and
a witness point set for an irreducible solution component V.
Wanted: To determine whether $\mathbf{x}$ lies on V.

In our example, we present the equations of f in factored form, so we can read off the equations for the solution components. We emphasize that we only take this for the sake of presentation, our method does not require this factorization. Consider,

$$f(x, y, z) = \begin{cases} (y - x^2)(y + z) = 0 \\ (z - x^3)(y - z) = 0 \end{cases} \tag{6}$$

From the factored form, we read off the four solution components of the system $f(x, y, z) = 0$:

1. $V_1 = \{\ (x, y, z) \mid y - x^2 = 0,\ z - x^3 = 0\ \}$ is the twisted cubic;
2. $V_2 = \{\ (x, y, z) \mid y - x^2 = 0,\ y - z = 0\ \}$ is a quadratic curve in the plane $y - z = 0$;
3. $V_3 = \{\ (x, y, z) \mid y + z = 0,\ z - x^3 = 0\ \}$ is a cubic curve in the plane $y + z = 0$;
4. $V_4 = \{\ (x, y, z) \mid y + z = 0,\ y - z = 0\ \}$ is the x-axis.

While the symbolic solution of the system is given by the factored form of the equations, the numerical solution is given by nine generic points: three on the first, two on the second, three on the third, and one on the last component. These generic points are solutions of the system

$$e(x, y, z) = \begin{cases} (y - x^2)(y + z) = 0 \\ (z - x^3)(y - z) = 0 \\ c_0 + c_1 x + c_2 y + c_3 z = 0 \end{cases} \tag{7}$$

where the constants c_0, c_1, c_2, and c_3 are randomly chosen complex numbers. The partition of the set of nine witness points corresponding to the components is achieved by running the monodromy algorithm on the nine solutions of the system $e(x, y, z) = \mathbf{0}$.
The homotopy membership test consists of the following three steps:

1. Adjust the constant term c_0 in (7) to c_0' so that the plane defined by $c_0' + c_1 x + c_2 y + c_3 z = 0$ passes through the test point.
2. Use the homotopy $h(x, y, z, t) = \mathbf{0}$ to track paths with t going from 1 to 0 in the system

$$h(x, y, z, t) = \begin{cases} (y - x^2)(y + z) = 0 \\ (z - x^3)(y - z) = 0 \\ c_0 t + c_0'(1 - t) + c_1 x + c_2 y + c_3 z = 0 \end{cases} \tag{8}$$

At $t = 1$ we start the path tracking at the witness point set of one of the irreducible components V_i of $f(\mathbf{x}) = \mathbf{0}$ and find their end points at $t = 0$.
3. If the test point lies on the component V_i, then it will be one of the endpoints at $t = 0$.
4. Repeat steps (2) and (3) for each of the components V_i.

This test is numerically stable because the comparison of the test point with the end points at $t = 0$ does not require any extra precision.

This numerical stability is very important for components of high degree. In [35], multi-precision arithmetic was needed to present filtering polynomials [35] to identify components. Evaluating those high degree components is numerically unstable. Unless one knows the test points with a sufficiently high accuracy, the result of the evaluation in the filtering polynomials cannot be certified.

The membership test is a crucial component in treating solution sets of different dimensions. To find generic points on all solution components of all dimensions, we apply a sequence of homotopies, introduced in [34]. The decomposition starts at the top dimension. At each step, the sets of generic points are partitioned, after removing superfluous points, using the membership test. We explain this procedure in the next section.

7 A Numerical Blackbox Decomposer

So far we have discussed the following tools:

1. use of monodromy to partition the sets of generic points into subsets of points that lie on the same irreducible solution component, or in more general terms, to decompose pure dimensional varieties into irreducibles; and
2. a homotopy membership test to separate isolated solutions from nonisolated points on solution components, or in general, to separate generic points on one component from those on higher dimensional components.

We still have to explain how to obtain the sets of generic points as the solutions of the original polynomial equations with additional linear constraints representing the random hyperplanes. As above we explain by example:

$$f(\mathbf{x}) = \begin{cases} (x_1 - 1)(x_2 - x_1^2) = 0 \\ (x_1 - 1)(x_3 - x_1^3) = 0 \\ (x_1^2 - 1)(x_2 - x_1^2) = 0 \end{cases} \tag{9}$$

From its factored form we see that $f(\mathbf{x}) = \mathbf{0}$ has two solution components: the two dimensional plane $x_1 = 1$ and the twisted cubic $\{\ (x_1, x_2, x_3) \mid x_2 - x_1^2 = 0,\ x_3 - x_1^3 = 0\ \}$.

To describe the solution set of this system, we use a sequence of homotopies, the chart in Figure 3 illustrates the flow of data for this example.

Because the top dimensional component is of dimension two, we add two random hyperplanes to the system and make it square again by adding two slack variables z_1 and z_2:

$$e(\mathbf{x}, z_1, z_2) = \begin{cases} (x_1 - 1)(x_2 - x_1^2) + a_{11} z_1 + a_{12} z_2 = 0 \\ (x_1 - 1)(x_3 - x_1^3) + a_{21} z_1 + a_{22} z_2 = 0 \\ (x_1^2 - 1)(x_2 - x_1^2) + a_{31} z_1 + a_{32} z_2 = 0 \\ \quad c_{10} + c_{11} x_1 + c_{12} x_2 + c_{13} x_3 + z_1 = 0 \\ \quad c_{20} + c_{21} x_1 + c_{22} x_2 + c_{23} x_3 + z_2 = 0 \end{cases} \tag{10}$$

where all constants a_{ij}, $i = 1,2,3$, $j = 1,2$, and c_{kl}, $k = 1,2$, $l = 0,1,2,3$ are randomly chosen complex numbers. Observe that when $z_1 = 0$ and $z_2 = 0$ the solutions to $e(\mathbf{x}, z_1, z_2) = \mathbf{0}$ satisfy $f(\mathbf{x}) = \mathbf{0}$. So if we solve $e(\mathbf{x}, z_1, z_2) = \mathbf{0}$ we will find the generic point on the two dimensional solution component $x_1 = 1$ as a solution with $z_1 = 0$ and $z_2 = 0$. Using polyhedral homotopies, this requires the tracing of six solutions paths.

The embedding was proposed in [34] to find generic points on all positive dimensional solution components with a sequence of homotopies. In [34] it was proven that solutions with slack variables $z_i \neq 0$ are regular and, moreover, that those solutions can be used as start solutions in a homotopy to find all generic points on lower dimensional solution components. We call those solutions *nonsolutions*.

In the solution of $e(\mathbf{x}, z_1, z_2) = \mathbf{0}$, one path ended with $z_1 = 0 = z_2$, the five other paths ended in regular solutions with $z_1 \neq 0$ and $z_2 \neq 0$. These five solutions are start solutions in the homotopy

$$\begin{aligned} & h_2(\mathbf{x}, z_1, z_2, t) \\ & = \begin{cases} (x_1 - 1)(x_2 - x_1^2) + a_{11} z_1 + a_{12} z_2 = 0 \\ (x_1 - 1)(x_3 - x_1^3) + a_{21} z_1 + a_{22} z_2 = 0 \\ (x_1^2 - 1)(x_2 - x_1^2) + a_{31} z_1 + a_{32} z_2 = 0 \\ c_{10} + c_{11} x_1 + c_{12} x_2 + c_{13} x_3 + z_1 = 0 \\ z_2(1-t) + (c_{20} + c_{21} x_1 + c_{22} x_2 + c_{23} x_3 + z_2) t = 0 \end{cases} \end{aligned} \tag{11}$$

where t goes from one to zero, replacing the last hyperplane with $z_2 = 0$. Of the five paths, four of them converge to solutions with $z_1 = 0$. Of those four solutions, one of them is found to lie on the two dimensional solution component $x_1 = 1$, the other three are generic points on the twisted cubic. As there is one solution with $z_1 \neq 0$ we have one candidate left for being an isolated solution of $f(\mathbf{x}) = \mathbf{0}$. This one solution with $z_1 \neq 0$ is used as start solution in the homotopy

$$h_1(\mathbf{x}, z_1, t) = \begin{cases} (x_1 - 1)(x_2 - x_1^2) + a_{11} z_1 = 0 \\ (x_1 - 1)(x_3 - x_1^3) + a_{21} z_1 = 0 \\ (x_1^2 - 1)(x_2 - x_1^2) + a_{31} z_1 = 0 \\ z_1(1-t) + (c_{10} + c_{11} x_1 + c_{12} x_2 + c_{13} x_3 + z_1) t = 0 \end{cases} \tag{12}$$

replacing the last hyperplane at $t = 1$ by $z_1 = 0$ at $t = 0$. At $t = 0$, the solution is found to lie on the twisted cubic, so there are no isolated solutions.

The calculations are summarized in Figure 3.

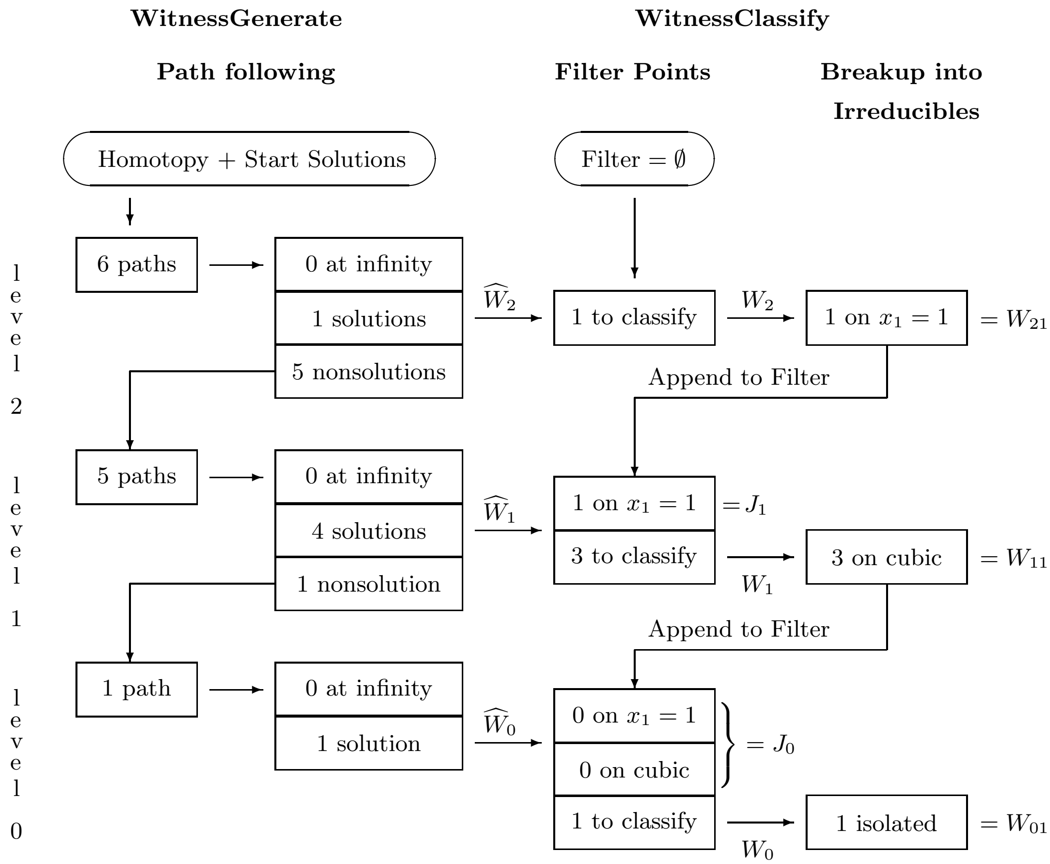

Fig. 3. Numerical Irreducible Decomposition of a system whose solutions are the 2-dimensional plane $x_1 = 1$, the twisted cubic, and one isolated point. At level i, for $i = 2, 1, 0$, we filter candidate witness point sets $\widehat{W}_i$ into junk sets J_i and witness point sets W_i. The sets W_i are partitioned into witness point sets W_{ij} for the irreducible components.

In making the transition to the benchmark applications section, we wish to illustrate the importance of the membership test on the cyclic 9-roots problem. This problem has besides six irreducible two dimensional cubic surfaces, 5,994 isolated regular solutions, also 648 roots of multiplicity four. After application of the homotopy membership test, the multiplicity four was determined by grouping clusters after refining the roots using Newton's method with multi-precision arithmetic (32 decimal places) until 13 decimal places were correct.

8 Benchmark Applications

To measure progress of our algorithms we executed the algorithms on a wide variety of polynomial systems. The second author maintains at `http://www.math.uic.edu/~jan/demo` a collection of 120 polynomial systems. We focus below on three of our most significant benchmarks. The systems come from relevant application fields and are challenging.

cyclic n-roots: This system is probably the most notorious benchmark, popularized by Davenport in [9] and coming from an application involving Fourier transforms [2,3]. Already for $n = 4$, the system

$$\left\{ \begin{array}{c} x_1 + x_2 + x_3 + x_4 = 0 \\ x_1x_2 + x_2x_3 + x_3x_4 + x_4x_1 = 0 \\ x_1x_2x_3 + x_2x_3x_4 + x_3x_4x_1 + x_4x_1x_2 = 0 \\ x_1x_2x_3x_4 - 1 = 0 \end{array} \right. \tag{13}$$

is numerically troublesome when directly fed to traditional homotopy continuation methods because there are no isolated solutions and all paths terminate close to points on the two quadratic solution curves.

Haagerup confirmed in [19] a conjecture of Fröberg (mentioned in [30]): for n prime, the number of solutions is finite and equals $\dfrac{(2n-2)!}{(n-1)!^2}$. Fröberg furthermore conjectured that for a quadratic divisor of n, there are infinitely many solutions.

Progress with computer algebra methods is described for dimensions 7 in [4], 8 in [5] and 9 in [14]. Unpublished (mentioned in [19]) is the result of Björck, who found all distinct isolated 184,756 cyclic 11-roots.

To solve this problem efficiently by numerical homotopy solvers, we need fast polyhedral methods to bound the number of isolated solutions by means of mixed volumes. These mixed volumes were computed for all dimensions up to $n = 11$ by Emiris in [12]; see also [13].

With PHCpack, all isolated 35,840 isolated cyclic 10-roots were computed. As reported earlier in the this paper, with the aid of the software of T.Y. Li and Li Xing [27], PHC also found all 184,756 cyclic 11-roots. While those cases are computationally very intensive, they are numerically "easy" to handle as all solutions are isolated. Recently, we found in [37] the decomposition of the one dimensional solution component of the cyclic 8-roots system into 16 pieces, 8 quadrics and 8 curves of degree 16. There are also 1,152 isolated cyclic 8-roots. In addition to 6,642 isolated solutions, the cyclic 9-roots problem has a two dimensional component of degree 18, which breaks up into six cubic surfaces.

As the technology for computing mixed volumes advances, (see [16,17] and [41]), we may expect further cases to be solved. In particular, [8] reports the approximation of all nonsingular cyclic 12-roots.

adjacent minors: This application is taken from [11], where ideals are decomposed to study the connectivity of certain graphs arising in a random walks. One particular system in this paper is defined by all adjacent minors of a $2 \times (n+1)$-matrix. For example, for $n = 3$ we have as matrix and polynomial system:

$$\begin{bmatrix} x_{11} & x_{12} & x_{13} & x_{14} \\ x_{21} & x_{22} & x_{23} & x_{24} \end{bmatrix} \qquad \begin{cases} x_{11}x_{22} - x_{21}x_{12} = 0 \\ x_{12}x_{23} - x_{22}x_{13} = 0 \\ x_{13}x_{24} - x_{23}x_{14} = 0 \end{cases} \tag{14}$$

In [11], it is proven that the ideal of adjacent 2×2-minors of a generic $2 \times (n+1)$-matrix is the intersection of F_n prime ideals (where F_n is the nth Fibonacci number), and that the ideal is radical. All irreducible solution components have the same dimension are the sum of their degrees equals 2^n.

Applying the monodromy breakup algorithm of [37], we found the breakup of a curve of degree 2048 into 144 irreducible components, as solution set to a system defined by all adjacent minors of a general 2×12-matrix.

We have limited our calculations to the case of matrices with 2 rows, see [11] for results on more general matrices. We expect methods exploiting the binomial structure of the equations, like the ones in [20] to be superior over general-purpose solvers.

Griffis-Duffy platform: In mechanical engineering we study the direct kinematics of a so-called parallel robot (also known as a Stewart-Gough platform), consisting of two platforms attached to each other by six links. The problem is to find all possible positions of the top platform for a given position of the base platform and given lengths of the links. Numerical continuation methods [31] first established that this problem has 40 isolated solutions. This result was later confirmed analytically in [24], [33] and [49]. These parallel robots are an important area of study, see for example [29].

In [18] a special type of the Stewart-Gough platform was proposed. In [25], it was shown that this platform permits motion, i.e., instead of 40 isolated solutions we now find a solution curve. Following [25], we call this type of platform a Griffis-Duffy platform.

Instead of 40 isolated solutions, we now have a one dimensional solution curve of degree 40. We investigated two cases of this Griffis-Duffy platform, see [37,38]. In both cases the curve of degree 40 contains twelve degenerate lines. For general choices of the parameters we have an irreducible curve of degree 28, while a more special case (also analyzed in [25]) breaks up in several components of lower degree.

We note that for this system we took approximate coefficients of the input polynomials.

9 Conclusions

In this paper we gave a description of our software tools to decompose positive dimensional solution sets of polynomial systems into irreducible components, emphasizing the geometrical and numerical aspects.

We reported on the first steps to let the software interact with a computer algebra system, such as Maple. This interaction is currently limited to passing polynomials from Maple into PHCpack and passing results from PHCpack (solution vectors or interpolating polynomials) back into Maple. We experienced that efficient visualization requires more advanced communication protocols. These protocols will be built, using the C interface to PHCpack.

Acknowledgment. The authors wish to thank Jean-Charles Faugère for confirming the number of isolated cyclic 9-roots. The authors are grateful for the comments of the referee.

References

1. D.N. Bernshteǐn. The number of roots of a system of equations. *Functional Anal. Appl.* 9(3):183–185, 1975. Translated from *Funktsional. Anal. i Prilozhen.* 9(3):1–4,1975.
2. G. Björck. Functions of modulus one on Z_p whose Fourier transforms have constant modulus. In *Proceedings of the Alfred Haar Memorial Conference, Budapest*, volume 49 of *Colloquia Mathematica Societatis János Bolyai*, 193–197. 1985.
3. G. Björck. Functions of modulus one on Z_n whose Fourier transforms have constant modulus, and "cyclic n-roots". In *Recent Advances in Fourier Analysis and its Applications*, edited by J.S. Byrnes and J.F. Byrnes. Volume 315 of *NATO Adv. Sci. Inst. Ser. C: Math. Phys. Sci.*, 131–140. Kluwer, 1989.
4. 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(3):329–336, 1991.
5. G. Björck and R. Fröberg. Methods to "divide out" certain solutions from systems of algebraic equations, applied to find all cyclic 8-roots. In *Analysis, Algebra and Computers in Math. research*, edited by M. Gyllenberg and L.E. Persson. Volume 564 of *Lecture Notes in Applied Mathematics*, 57–70. Marcel Dekker, 1994.
6. O. Bottema and B. Roth. *Theoretical Kinematics.* North-Holland, Amsterdam, 1979.
7. J. Canny and J.M. Rojas. An optimal condition for determining the exact number of roots of a polynomial system. In *Proceedings of the 1991 International Symposium on Symbolic and Algebraic Computation*, 96–101. ACM, 1991.

8. Y. Dai, S. Kim, and M. Kojima. Computing all nonsingular solutions of cyclic-n polynomial using polyhedral homotopy continuation methods. Research Reports on Mathematical and Computing Sciences. Series B: Operations Research. Department of Mathematical and Computing Sciences. Tokyo Institute of Technology. Technical Report B-373, September 2001.
9. J. Davenport. Looking at a set of equations. Technical report 87-06, Bath Computer Science, 1987.
10. W. Decker and F.-O. Schreyer. Computational algebraic geometry today. In *Application of Algebraic Geometry to Coding Theory, Physics and Computation*, edited by C. Ciliberto, F. Hirzebruch, R. Miranda, and M. Teicher, 65–119, 2001. Proceedings of a NATO Conference, February 25 - March 1, 2001, Eilat, Israel. Kluwer Academic Publishers.
11. P. Diaconis, D. Eisenbud, and B. Sturmfels. Lattice walks and primary decomposition. In *Mathematical Essays in Honor of Gian-Carlo Rota*, edited by B.E. Sagan, R.P. Stanley. Volume 161 of *Progress in Mathematics*, 173–193, Birkhäuser, 1998.
12. 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, 1994. Available at `http://www-sop.inria.fr/galaad/emiris`.
13. 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, 1995. Software available at `http://www-sop.inria.fr/galaad/emiris`.
14. J.C. Faugère. A new efficient algorithm for computing Gröbner bases (F_4). *Journal of Pure and Applied Algebra* 139(1-3):61–88, 1999. Proceedings of MEGA'98, 22–27 June 1998, Saint-Malo, France.
15. A. Galligo, I.S. Kotsireas, R.M. Corless, and S.M. Watt. A geometric numeric algorithm for multivariate factorization. Poster presented at ISSAC'01.
16. T. Gao and T.Y. Li. Mixed volume computation via linear programming. *Taiwan J. of Math.* 4, 599–619, 2000.
17. T. Gao and T.Y. Li. Mixed volume computation for semi-mixed systems. Manuscript, 2001.
18. M. Griffis and J. Duffy. Method and apparatus for controlling geometrically simple parallel mechanisms with distinctive connections. US Patent #5,179,525, 1993
19. U. Haagerup. Orthogonal maximal abelian *-algebras of the n x n matrices and cyclic n-roots. In *Operator Algebras and Quantum Field Theory*, 296–322. International Press, Cambridge, MA, 1996.
20. S. Hosten and J. Shapiro. Primary decomposition of lattice basis ideals. *Journal of Symbolic Computation* 29(4&5):625–639, 2000. Special Issue on Symbolic Computation in Algebra, Analysis and Geometry. Edited by E. Cattani and R.C. Laubenbacher.
21. B. Huber, F. Sottile, and B. Sturmfels. Numerical Schubert calculus. *J. of Symbolic Computation* 26(6):767–788, 1998.
22. B. Huber and B. Sturmfels. A polyhedral method for solving sparse polynomial systems. *Math. Comp.* 64(212):1541–1555, 1995.
23. B. Huber and J. Verschelde. Pieri homotopies for problems in enumerative geometry applied to pole placement in linear systems control. *SIAM J. Control Optim.* 38(4): 1265–1287, 2000.

24. M.L. Husty. An algorithm for solving the direct kinematics of general Stewart-Gough Platforms. *Mech. Mach. Theory,* 31(4):365–380, 1996.
25. M.L. Husty and A. Karger. Self-motions of Griffis-Duffy type parallel manipulators *Proc. 2000 IEEE Int. Conf. Robotics and Automation*, CDROM, San Francisco, CA, April 24–28, 2000.
26. T.Y. Li. Numerical solution of multivariate polynomial systems by homotopy continuation methods. *Acta Numerica* 6:399–436, 1997.
27. T.Y. Li and X. Li. Finding mixed cells in the mixed volume computation. *Found. Comput. Math.* 1(2): 161–181, 2001. Software available at `http://www.math.msu.edu/~li`.
28. M. Maekawa, M. Noro, K. Ohara, Y. Okutani, N. Takayama, and Y. Tamura, OpenXM – an open system to integrate mathematical softwares. Available at `http://www.OpenXM.org/`.
29. J-P. Merlet. Parallel robots. Kluwer, Dordrecht, 2000. For example pictures, see `http://www-sop.inria.fr/coprin/equipe/merlet/merlet.html.`
30. H.M. Möller. Gröbner bases and numerical analysis. In *Gröbner Bases and Applications*, edited by B. buchberger and F. Winkler. Volume 251 of *London Mathematical Lecture Note Series*, 159–178. Cambridge University Press, 1998.
31. M. Raghavan. The Stewart platform of general geometry has 40 configurations. *ASME J. Mech. Design* 115:277–282, 1993.
32. J.M. Rojas. A convex geometric approach to counting the roots of a polynomial system. *Theoret. Comput. Sci.* 133(1):105–140, 1994.
33. F. Ronga and T. Vust. Stewart platforms without computer? In *Real Analytic and Algebraic Geometry.* Proceedings of the International Conference, (Trento, 1992), 196–212. Walter de Gruyter, 1995.
34. A.J. Sommese and J. Verschelde. Numerical homotopies to compute generic points on positive dimensional algebraic sets. *Journal of Complexity* 16(3):572–602, 2000.
35. A.J. Sommese, J. Verschelde, and C.W. Wampler. Numerical decomposition of the solution sets of polynomial systems into irreducible components. *SIAM J. Numer. Anal.* 38(6):2022–2046, 2001.
36. A.J. Sommese, J. Verschelde, and C.W. Wampler. Numerical irreducible decomposition using projections from points on the components. In *Symbolic Computation: Solving Equations in Algebra, Geometry, and Engineering*, volume 286 of *Contemporary Mathematics*, edited by E.L. Green, S. Hoşten, R.C. Laubenbacher, and V. Powers, pages 37–51. AMS 2001.
37. A.J. Sommese, J. Verschelde, and C.W. Wampler. Using monodromy to decompose solution sets of polynomial systems into irreducible components. In *Application of Algebraic Geometry to Coding Theory, Physics and Computation*, edited by C. Ciliberto, F. Hirzebruch, R. Miranda, and M. Teicher, 297–315, 2001. Proceedings of a NATO Conference, February 25 - March 1, 2001, Eilat, Israel. Kluwer Academic Publishers.
38. A.J. Sommese, J. Verschelde, and C.W. Wampler. Symmetric functions applied to decomposing solution sets of polynomial systems. Submitted for Publication, 2001.
39. A.J. Sommese and C.W. Wampler. Numerical algebraic geometry. In *The Mathematics of Numerical Analysis*, volume 32 of *Lectures in Applied Mathematics*, edited by J. Renegar, M. Shub, and S. Smale, 749–763, 1996. Pro-

ceedings of the AMS-SIAM Summer Seminar in Applied Mathematics, Park City, Utah, July 17-August 11, 1995, Park City, Utah.

40. B. Sturmfels. Polynomial equations and convex polytopes. *Amer. Math. Monthly* 105(10):907–922, 1998.
41. A. Takeda, M. Kojima, and K. Fujisawa. Enumeration of all solutions of a combinatorial linear inequality system arising from the polyhedral homotopy continuation method. Research Reports on Mathematical and Computing Sciences. Series B: Operations Research. Department of Mathematical and Computing Sciences. Tokyo Institute of Technology. Technical Report B-364, November 2000.
42. J. Verschelde. Algorithm 795: PHCpack: A general-purpose solver for polynomial systems by homotopy continuation. *ACM Transactions on Mathematical Software* 25(2): 251–276, 1999. Software available at `http://www.math.uic.edu/~jan`.
43. J. Verschelde. Polynomial homotopies for dense, sparse and determinantal systems. MSRI Preprint #1999-041. Available at `http://www.math.uic.edu/~jan`.
44. J. Verschelde. Numerical evidence for a conjecture in real algebraic geometry. *Experimental Mathematics* 9(2): 183–196, 2000.
45. J. Verschelde and R. Cools. Symbolic homotopy construction. *Applicable Algebra in Engineering, Communication and Computing* 4(3):169–183, 1993.
46. J. Verschelde, K. Gatermann, and R. Cools. Mixed-volume computation by dynamic lifting applied to polynomial system solving. *Discrete Comput. Geom.* 16(1):69–112, 1996.
47. J. Verschelde and A. Haegemans. The *GBQ*-Algorithm for constructing start systems of homotopies for polynomial systems. *SIAM J. Numer. Anal.* 30(2):583–594, 1993.
48. J. Verschelde, P. Verlinden, and R. Cools. Homotopies exploiting Newton polytopes for solving sparse polynomial systems. *SIAM J. Numer. Anal.* 31(3):915–930, 1994.
49. C.W. Wampler. Forward displacement analysis of general six-in-parallel SPS (Stewart) platform manipulators using soma coordinates. *Mech. Mach. Theory* 31(3): 331–337, 1996.