Software for Numerical Irreducible Decomposition

Jan Verschelde

Department of Math, Stat & CS

University of Illinois at Chicago

Chicago, IL 60607-7045

e-mail: **jan@math.uic.edu**

web: **www.math.uic.edu/~jan**

FoCM 2002, August 9, 2002

Joint work with Andrew J. Sommese and Charles W. Wampler

Outline of Talk:

1. Numerical Homotopy Continuation
2. The software package PHCpack
   - 2.1 transition: isolated $\rightarrow$ component solutions
   - 2.2 interfaces with Maple and C
3. Numerical Irreducible Decomposition
   - 3.1 a new way to find nonsingular solutions
   - 3.2 computational experiments
4. Conclusions

1. Numerical Homotopy Algorithms

If we wish to solve $f(\mathbf{x}) = \mathbf{0}$, then we construct a system $g(\mathbf{x}) = \mathbf{0}$ whose solutions are known. Consider the *homotopy*

$$H(\mathbf{x}, t) := (1 - t)g(\mathbf{x}) + tf(\mathbf{x}) = \mathbf{0}.$$

By *continuation*, we trace the paths starting at the known solutions of $g(\mathbf{x}) = \mathbf{0}$ to the desired solutions of $f(\mathbf{x}) = \mathbf{0}$, for t from 0 to 1.

PHCpack is a software package for Polynomial Homotopy Continuation, offering a wide varieties of homotopy methods for solving polynomial systems.

Homotopy Types

Almost all polynomial systems arising in practical applications have a special structure. For instance, most polynomial systems have far fewer isolated solutions than the product of the degrees in the equations.

The homotopies available in PHCpack are

1. **Multi-homogeneous homotopies** for product structures;
2. **Polyhedral homotopies** to exploit sparsity;
3. **SAGBI and Pieri homotopies** for numerical Schubert calculus;
4. **coefficient-parameter homotopies** to study variations of coefficients and natural parameters.

2. Release History of PHCpack

March 1995: Pre-release of PHC and MVC, executable versions on the occasion of the PoSSo Workshop on Software.

August 1997: Release 1.0 of the full Ada 83 sources, executable versions for SUN, IBM AIX, and DEC Workstations, and demonstration database of test polynomial systems.

August 1999: Release 2.0 of the rewritten Ada 95 sources, extended with multi-precision facilities and numerical Schubert calculus. Executables for SUN, SGI, Linux and Windows PCs.

August 2002: Release 2.1 with β-version of the tools for a numerical irreducible decomposition and a C interface.

2.1 PHCpack is menu-driven and file oriented

```
Welcome to PHC (Polynomial Homotopy Continuation) Version 2.1(beta).

Running in full mode.  Note also the following options:
  phc -s : Equation and variable Scaling on system and solutions
  phc -d : Linear and nonlinear Reduction w.r.t. the total degree
  phc -r : Root counting and Construction of start systems
  phc -m : Mixed-Volume Computation by four lifting strategies
  phc -p : Polynomial Continuation by a homotopy in one parameter
  phc -v : Validation, refinement and purification of solutions
  phc -e : SAGBI/Pieri homotopies to intersect linear subspaces
  phc -c : Irreducible decomposition for solution components
  phc -f : Factor pure dimensional solution set into irreducibles
  phc -b : Batch or black-box processing
  phc -z : strip phc output solution lists into Maple format

Is the system on a file ? (y/n/i=info)
```

Documentation at www.math.uic.edu/~jan

J. Verschelde: **PHC and MVC: two programs for solving polynomial systems by homotopy continuation.** *Proceedings of the POSSO Workshop on Software*, pages 165-176. Edited by J.-C. Faugère, J. Marchand, R. Rioboo, Paris 1-4 March 1995.

J. Verschelde: **Homotopy Continuation Methods for Solving Polynomial Systems**. Ph. D. thesis, K.U.Leuven, Dept. of Computer Science, 1996.

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.

A.J. Sommese, J. Verschelde, and C.W. Wampler: **Numerical Irreducible Decomposition using PHCpack.** Proceedings of Dagstuhl Seminar no. 01421, 14–19 October 2001. Edited by M. Joswig and N. Takayama.

Papers documenting the usefulness of PHCpack

C.W. Wampler: **Isotropic coordinates, circularity and Bezout numbers: planar kinematics from a new perspective**. Proceedings of the 1996 ASME Design Engineering Technical Conference. Irvine, CA, Aug 18–22, 1996. (CD-ROM).

F. Sottile: **Real Schubert Calculus: Polynomial systems and a conjecture of Shapiro and Shapiro**. Experimental Mathematics 9(2): 161-182, 2000.

B. Haas: **A Simple Counterexample to Kouchnirenko's Conjecture**. Beitraege zur Algebra und Geometrie/Contributions to Algebra and Geometry, 43(1):1-8, 2002.

E. Lee, C. Mavroidis, and J. Morman: **Geometric Design of Spatial 3R Manipulators**. Proceedings of the 2002 NSF Design, Service, and Manufactoring Grantees and Research Conference, San Juan, Puerto Rico, January 7-10, 2002.

E. Lee and C. Mavroidis: **Solving the Geometric Design Problem of Spatial 3R Robot Manipulators Using Polynomial Continuation**. Journal of Mechanical Design, Transactions of the ASME, 2002 (in press).

F. Xie, G. Reid, and S. Valluri: **A numerical method for the one dimensional action functional for FBG structures**. Can J. Phys. 76: 1-21, 2002.

2.2 Multi-lingual programming

Bjarne Stroustrup : *"There never was a single language suitable for all work, and I doubt there ever will be."* Interview, March 2000.

www.ada2cc.com : *"So in converting, you tend to lose the advantages of the source language while not picking up the advantages of the destination languages."* from Realistic Technologies, Inc.

Two kinds of interfaces to PHCpack:

high level : with the computer algebra system Maple.

low level : with the programming language C.

Maple and PHCpack

Computer algebra systems like Maple provide a nice environment to formulate the polynomial equations.

- Interface requires only an executable version of the solver `phc`.
- A small Maple procedure
  1. writes the system from Maple onto a file `input`;
  2. calls the black-box solver `phc -b input output`;
  3. with `phc -z output` the solutions are brought into Maple.
- Lightweight interface, only modification to phc is extra option `-z` to convert formats for solutions.

The solution format conversion has been used to bring lists of samples from a curve into Maple for visualization.

Support for a C interface

We can build a portable interface to the Ada routines in PHCpack with C functions because...

1. The language Ada
   - has the `pragma Import` construction to call routines from other languages such as C;
   - supports conversions for C integers, doubles, and arrays of these C types.
2. The gnu-ada compiler
   - supports a mechanism to call Ada routines from a C main program and to call C functions from Ada;
   - is integrated in the gcc compilation system.

Passing Structured Data Types

polynomial in n variables with complex floating-point coefficients

$\Longleftrightarrow$ (n, #monomials, array of integers (support),

array of doubles (coefficients))

$x^2 + 3xy - 5 \Longleftrightarrow$ (2, 3, 2 0 1 1 0 0, 1.0 0.0 3.0 0.0 -5.0 0.0)

Programs in Ada or C must define exchange protocols of structured data types into basic data types for which automatic conversions are supported.

The next stage in the development of this interface consists in the passing of polynomial *functions* and their derivative *functions*.

$\rightarrow$ implementation of an evaluation-based solving.

C functions calling PHCpack

At this moment, the following C functions call PHCpack :

phc_solver.c calls the black-box solver of PHCpack.

pieri_solver.c invokes the Pieri homotopies to compute feedback laws to control linear systems (joint project with Yusong Wang).

path_tracker.c forms interface to path-tracking routines of PHCpack. Input are target, start system with solutions; the output are solutions computed at the end of the paths.

Typical sequence of calls:

C function with problem data
$\rightarrow$ Ada routine of PHCpack computes
$\rightarrow$ C function processes results of PHCpack.

3. Numerical Irreducible Decomposition

with PHCpack

In computing a numerical irreducible decomposition of a given polynomial system, we typically run through the following steps:

1.	**Embed** (phc -c)	add #random hyperplanes = top dimension,
		add slack variables to make the system square
2.	**Solve** (phc -b)	solve the system constructed above
3.	**WitnessGenerate**	apply a sequence of homotopies to compute
	(phc -c)	witness point sets on all solution components
4.	**WitnessClassify**	filter junk from witness point sets
	(phc -f)	factor components into irreducible components

Especially step 2 is a computational bottleneck.
We recently discovered and implemented a new algorithm.

3.1 Solving Systems Incrementally

- Extrinsic and Intrinsic Deformations

  extrinsic : defined by explicit equations

  intrinsic : following the actual geometry

- Diagonal Homotopies

  $\rightarrow$ to intersect pure dimensional solution sets

- Intersecting with Hypersurfaces

  adding the polynomial equations one after the other we arrive at an incremental polynomial system solver.

Extrinsic Homotopy Deformations

$f(\mathbf{x}) = \mathbf{0}$ has k-dimensional solution components. We cut with k hyperplanes to find isolated solutions = *witness points sets* :

$$a_{i0} + \sum_{j=1}^{n} a_{ij} x_j = 0, \quad i = 1, 2, \ldots, k, \quad a_{ij} \in \mathbb{C} \text{ random}$$

$$\text{Sample} \quad \left\{ \begin{array}{cc} f(\mathbf{x}) + \gamma \mathbf{z} = 0 & \mathbf{z} = \textit{slack} \\ a_{i0}(t) + \displaystyle\sum_{j=1}^{n} a_{ij}(t) x_j = 0 & \textit{moving} \end{array} \right.$$

$$\#\text{witness points} = \sum_{\substack{C \subseteq f^{-1}(0) \\ \dim(C) = k}} \deg(C)$$

Embedding with Slack Variables

The cyclic 4-roots system defines 2 quadrics in $\mathbb{C}^4$:

$$\left\{\begin{array}{l}\left\{\begin{array}{lllllll} x_1 & + x_2 & + x_3 & + x_4 & + & \gamma_1 z & = & 0 \\ x_1x_2 & + x_2x_3 & + x_3x_4 & + x_4x_1 & + & \gamma_2 z & = & 0 \\ x_1x_2x_3 & + x_2x_3x_4 & + x_3x_4x_1 & + x_4x_1x_2 & + & \gamma_3 z & = & 0 \\ x_1x_2x_3x_4 & - \ 1 & & & + & \gamma_4 z & = & 0 \end{array}\right. \\ \qquad a_0 + a_1x_1 + a_2x_2 + a_3x_3 + a_4x_4 + z = 0 \end{array}\right.$$

Original system : 4 equations in x_1, x_2, x_3, and x_4.

Cut with random hyperplane to find isolated points.

Slack variable z with random γ_i, $i = 1, 2, 3, 4$: square system.

Solve embedded system to find $4 = 2{+}2$ witness points as isolated solutions with $z = 0$.

Intrinsic Homotopy Deformations

$f(\mathbf{x}) = \mathbf{0}$ has k-dimensional solution components. We cut with a random affine $(n-k)$-plane to find witness points :

$$\mathbf{x}(\lambda) = \mathbf{b} + \sum_{i=1}^{n-k} \lambda_i \mathbf{v}_i \in \mathbb{C}^n$$

The vectors $\mathbf{b}$ and $\mathbf{v}_i$ are choosen at random.

$$\text{Sample} \quad f\left(\mathbf{x}(\lambda, t) = \mathbf{b}(t) + \sum_{i=1}^{n-k} \lambda_i \mathbf{v}_i(t) \right) = \mathbf{0}$$

Points on the moving $(n-k)$-plane are determined by $n-k$ independent variables λ_i, $i = 1, 2, \ldots, n-k$.

#independent variables = co-dimension

$f(\mathbf{x}) = \mathbf{0}$ is a system with $\mathbf{x} \in \mathbb{C}^n$, $\mathbf{x}$ lies on an affine $(n-k)$-plane:

$$\mathbf{x}(\lambda) = \mathbf{b} + \sum_{i=1}^{n-k} \lambda_i \mathbf{v}_i \in \mathbb{C}^n$$

where $\lambda = (\lambda_1, \lambda_2, \ldots, \lambda_{n-k})$ contains all independent variables.

Correct with Newton on $f(\mathbf{x}(\lambda)) = \mathbf{0}$, a system in λ.

$$\text{Solve} \quad \left[\frac{\partial f}{\partial \lambda}\right] \lambda = -f(\mathbf{x}(\lambda)) \quad \text{with} \quad \frac{\partial f_i}{\partial \lambda_j} = \sum_{l=1}^{n-k} \frac{\partial f_i}{\partial x_l} \frac{\partial x_l}{\partial \lambda_j}.$$

Overdetermined case moved from global to local level!

no slack variables needed...

Intersecting Hypersurfaces Extrinsicially

$$\begin{cases} f_1(\mathbf{x}) = 0 \quad \mathbf{x} \in \mathbb{C}^n \\ L_1(\mathbf{x}) = \mathbf{0} \ {\scriptstyle n-1 \text{ hyperplanes}} \end{cases} \qquad \begin{cases} f_2(\mathbf{y}) = 0 \quad \mathbf{y} \in \mathbb{C}^n \\ L_2(\mathbf{y}) = \mathbf{0} \ {\scriptstyle n-1 \text{ hyperplanes}} \end{cases}$$

diagonal homotopy *extrinsic version*

$$\left(\begin{cases} f_1(\mathbf{x}) = 0 \\ f_2(\mathbf{y}) = 0 \\ L_1(\mathbf{x}) = \mathbf{0} \\ L_2(\mathbf{y}) = \mathbf{0} \end{cases} \right) t + \left(\begin{cases} f_1(\mathbf{x}) = 0 \\ f_2(\mathbf{y}) = 0 \\ \mathbf{x} - \mathbf{y} = \mathbf{0} \\ M(\mathbf{y}) = \mathbf{0} \end{cases} \right) (1-t) = \mathbf{0}$$

At $t = 1$ **:** $\deg(f_1) \times \deg(f_2)$ solutions $(\mathbf{x}, \mathbf{y}) \in \mathbb{C}^{n \times n}$.

At $t = 0$ **:** witness points ($\mathbf{x} = \mathbf{y} \in \mathbb{C}^n$) on $f_1^{-1}(0) \cap f_2^{-1}(0)$ cut out by $n-2$ hyperplanes M.

Intersecting Hypersurfaces Intrinsically

Consider a general affine line $\mathbf{x}(\lambda) = \mathbf{b} + \lambda \mathbf{v} \in \mathbb{C}^n$.

$$\underset{\deg(f_1)\ \text{values for}\ \lambda}{f_1(\mathbf{x}(\lambda) = \mathbf{b} + \lambda \mathbf{v})} \quad \bigcap \quad \underset{\deg(f_2)\ \text{values for}\ \mu}{f_2(\mathbf{y}(\lambda) = \mathbf{b} + \mu \mathbf{v})}$$

diagonal homotopy
$$\begin{pmatrix} f_1 \\ f_2 \end{pmatrix} \left(\begin{bmatrix} \mathbf{x}(t) \\ \mathbf{y}(t) \end{bmatrix} \right) = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$$
intrinsic version

$$\begin{bmatrix} \mathbf{x}(t) \\ \mathbf{y}(t) \end{bmatrix} = \begin{bmatrix} \mathbf{b} \\ \mathbf{b} \end{bmatrix} + \lambda \left(\begin{bmatrix} \mathbf{v} \\ \mathbf{0} \end{bmatrix} t + \begin{bmatrix} \mathbf{u}_1 \\ \mathbf{u}_1 \end{bmatrix} (1-t) \right) + \mu \left(\begin{bmatrix} \mathbf{0} \\ \mathbf{v} \end{bmatrix} t + \begin{bmatrix} \mathbf{u}_2 \\ \mathbf{u}_2 \end{bmatrix} (1-t) \right)$$

At $t = 1$ **:** $\deg(f_1) \times \deg(f_2)$ solutions $(\mathbf{x}, \mathbf{y}) \in \mathbb{C}^{n \times n}$.

At $t = 0$ **:** witness points on $\mathbf{x} = \mathbf{b} + \lambda \mathbf{u}_1 + \mu \mathbf{u}_2$, a general 2-plane defined by a random point $\mathbf{b}$ and 2 random vectors $\mathbf{u}_1$ and $\mathbf{u}_2$.

Intersecting with Hypersurfaces

Let $f(\mathbf{x}) = \mathbf{0}$ have k-dimensional solution components described by witness points on a general $(n-k)$-dimensional affine plane, i.e.:

$$f\left(\mathbf{x}(\lambda) = \mathbf{b} + \sum_{i=1}^{n-k} \lambda_i \mathbf{v}_i\right) = \mathbf{0}.$$

Let $g(\mathbf{x}) = 0$ be a hypersurface with witness points on a general affine line, i.e.:

$$g(\mathbf{x}(\mu) = \mathbf{b} + \mu \mathbf{w}) = 0.$$

Assuming $g(\mathbf{x}) = 0$ properly cuts one degree of freedom from $f^{-1}(\mathbf{0})$, we want to find witness points on all $(k-1)$-dimensional components of $f^{-1}(\mathbf{0}) \cap g^{-1}(0)$.

Intrinsic Hypersurface Intersection

The **diagonal homotopy** for (f, g) on $(\mathbf{x}, \mathbf{y}) \in \mathbb{C}^{n \times n}$ starts at

$$\left[\begin{array}{c} \mathbf{x}(1) \\ \mathbf{y}(1) \end{array}\right] = \left[\begin{array}{c} \mathbf{b} \\ \mathbf{b} \end{array}\right] + \sum_{i=1}^{n-k} \lambda_i \left[\begin{array}{c} \mathbf{v}_i \\ \mathbf{0} \end{array}\right] + \mu \left[\begin{array}{c} \mathbf{0} \\ \mathbf{w} \end{array}\right]$$

and ends at

$$\left[\begin{array}{c} \mathbf{x}(0) \\ \mathbf{y}(0) \end{array}\right] = \left[\begin{array}{c} \mathbf{b} \\ \mathbf{b} \end{array}\right] + \sum_{i=1}^{n-k} \lambda_i \left[\begin{array}{c} \mathbf{v}_i \\ \mathbf{v}_i \end{array}\right] + \mu \left[\begin{array}{c} \mathbf{w} \\ \mathbf{w} \end{array}\right].$$

The diagonal homotopy

$$\left(\begin{array}{c} f \\ g \end{array}\right) \left(\left[\begin{array}{c} \mathbf{x}(t) \\ \mathbf{y}(t) \end{array}\right] = \left[\begin{array}{c} \mathbf{x}(1) \\ \mathbf{y}(1) \end{array}\right] t + \left[\begin{array}{c} \mathbf{x}(0) \\ \mathbf{y}(0) \end{array}\right] (1-t) \right) = \left[\begin{array}{c} \mathbf{0} \\ 0 \end{array}\right]$$

has $n - k + 1$ independent variables $(\lambda_1, \lambda_2, \ldots, \lambda_{n-k}, \mu)$.

Computing Nonsingular Solutions Incrementally

Suppose $(f_1, f_2, \ldots, f_k)$ defines the system $f(\mathbf{x}) = \mathbf{0}$, $\mathbf{x} \in \mathbb{C}^n$, whose solution set is pure dimensional of multiplicity one for all $k = 1, 2, \ldots, N \leq n$, i.e.: we find only nonsingular roots if we slice the solution set of $f(\mathbf{x}) = \mathbf{0}$ with a generic linear space of dimension $n - k$.

Main loop in the solver :

```
for $k = 2, 3, \ldots, N-1$ do
  use a diagonal homotopy to intersect
        $(f_1, f_2, \ldots, f_k)^{-1}(\mathbf{0})$ with $f_{k+1}(\mathbf{x}) = 0$,
  to find witness points on all $(n-k-1)$-dimensional
  solution components.
```

Outcomes of Hypersurface Intersections

Let V be an $(n-k)$-dimensional irreducible component of $(f_1, ..., f_k)^{-1}(\mathbf{0})$ and $g^{-1}(0)$ be an irreducible hypersurface.

Three cases for $V \cap g^{-1}(0)$:

1. $V \subseteq g^{-1}(0)$

   All witness points of V satisfy $g(\mathbf{x}) = 0$.

2. $\dim(V \cap g^{-1}(0)) = k-1$

   The diagonal homotopy gives witness points on all $(k-1)$-dimensional components of the intersection.

3. $V \cap g^{-1}(0) = \emptyset$

   All paths in the diagonal homotopy diverge.

3.2 Test Polynomial Systems

- all adjacent minors of a general matrix
- the cyclic n-roots problem
- a 7-bar mechanism in the plane
- a spatial Burmester problem
- the Griffis-Duffy platform

Adjacent 2×2-minors of a $2 \times (n+1)$-matrix

$$n = 3 : \quad \begin{bmatrix} \overbrace{x_{11} \quad x_{12}}^{(1)} & \overbrace{x_{13} \quad x_{14}}^{(3)} \\ x_{21} \quad \underbrace{x_{22} & x_{23}}_{(2)} \quad x_{24} \end{bmatrix} \qquad \begin{cases} x_{11}x_{22} - x_{21}x_{12} = 0 & (1) \\ x_{12}x_{23} - x_{22}x_{13} = 0 & (2) \\ x_{13}x_{24} - x_{23}x_{14} = 0 & (3) \end{cases}$$

Theorem (Diaconis, Eisenbud, Sturmfels 1998):

The ideal of all adjacent 2×2-minors of a generic $2 \times (n+1)$-matrix is the intersection of $f(n)$ prime ideals. In particular, the ideal is radical.

$f(0) = f(1) = 1, f(n) = f(n-1) + f(n-2)$, Fibonacci numbers

P. Diaconis, D. Eisenbud, and B. Sturmfels. **Lattice Walks and Primary Decomposition.** Vol 161 of *Progress in Mathematics*, 173–193, 1998.

S. Hosten and J. Shapiro: **Primary Decomposition of Lattice Basis Ideals.** *Journal of Symbolic Computation* 29(4&5): 625–639, 2000.

Witness Points for Adjacent Minors

A general $2 \times (n+1)$-matrix gives a system of n equations in $2n+2$ variables whose $(n+2)$-dimensional solution set has degree 2^n.

n	m	d	user cpu time	n	m	d	user cpu time
3	8	8	301ms	8	18	256	1m 46s 162ms
4	10	16	962ms	9	20	512	7m 1s 596ms
5	12	32	3s 395ms	10	22	1,024	17m 57s 269ms
6	14	64	11s 497ms	11	24	2,048	40m 15s 723ms
7	16	128	31s 425ms	12	26	4,096	1h 58m 22s 282ms

n = #equations, m = #variables, d = degree

user cpu time for Pentium III 1Ghz Windows 2000

A spatial Burmester Problem

Classical in the plane : Given five placements of a moving body, find the points of the moving body that lie on a common fixed circle.

Spatial body guidance : Seven general positions determine 20 center-point/sphere-point pairs. (Schönflies) O. Bottema and B. Roth: **Theoretical Kinematics.** North-Holland, Amsterdam, 1979.

Instead of seven, take only six placements. We solve

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

where $\mathbf{p}_i$ are positions of the body and R_i are rotation matrices.

For given positions, determine $(\mathbf{x}, \mathbf{y})$, with $\mathbf{x}$ the sphere-point curve and $\mathbf{y}$ the center-point curve.

Witness Points

for the Spatial Burmester Problem

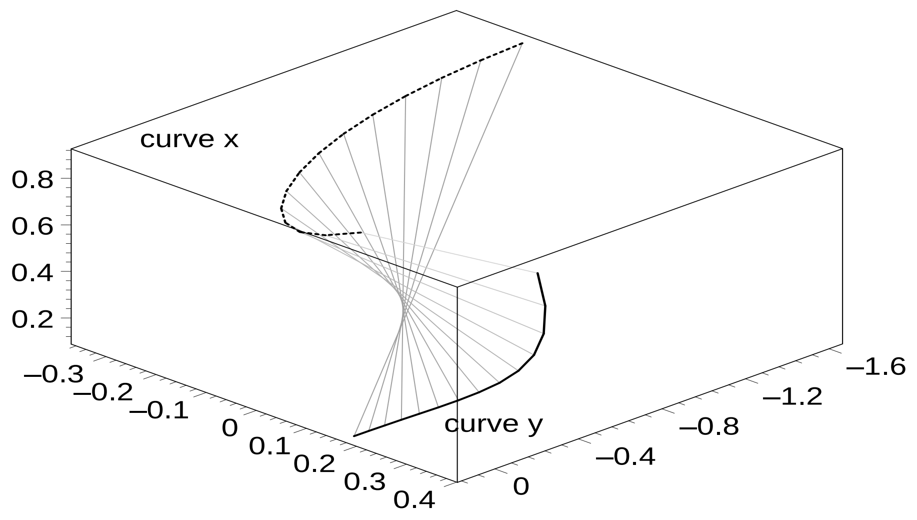
- The input polynomial system consists of five quadrics in six unknowns $(\mathbf{x}, \mathbf{y})$.
- The new incremental solver computes 20 witness points in 7s 181ms on Pentium III 1Ghz Windows 2000 PC.
- Projection onto $\mathbf{x}$ or $\mathbf{y}$ reduces the degree from 20 to 10.

Visualization of Samples with Maple

```
> 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);
```



4. Conclusions

- Feasible in practice to decompose the solution set of a polynomial system by standard machine arithmetic.

  multi-precision arithmetic is needed for singular components...

- The incremental solving method with diagonal homotopies promises to unify solvers for isolated *and* solvers for components of solutions.

  exploitation of structure in progress...