Purdue University
Purdue e-Pubs

Department of Computer Science Technical Reports
Department of Computer Science

1998

Continuum: A Homotopy- Continuation Solver for Systems of Algebraic Equations

Cassiano Durand

Christoph M. Hoffmann
Purdue University, cmh@cs.purdue.edu

Report Number:
98-028

CONTINUUM: A HOMOTOPY-CONTINUATION SOLVER FOR SYSTEMS OF ALGEBRAIC EQUATIONS

Cassiano Durand
Christoph M. Hoffman

Purdue University
Department of Computer Science
West Lafayette, IN 47907

CSD-TR #98-028
September 1998

Continuum: A Homotopy-Continuation Solver for Systems of Algebraic Equations

Cassiano Durand Christoph M. Hoffmann

September 1998

Abstract

This work describes *Continuum*, a general-purpose solver for polynomial systems by homotopy continuation, which has been extensively used in [Dur98] to solve systems of geometric constraints.

1 Introduction

Polynomial systems arise frequently in many different fields of science and engineering, such as computer-aided design, inverse kinematics, and molecular modeling. Many standard numerical methods can be used to find one solution of the system. One of the most celebrated is the Newton-Raphson method [SB93, OR70], which is distinguished by the ability to solve large problems. However it is very sensitive, requiring a sufficiently good initial guess. Nonlinear optimization methods [DS83] are global and converge to a solution from almost any initial guess.

Nonetheless, many applications require that all solutions are computed. Göbner bases [Buc85, ALW95, CLO92, Buc79], characteristic sets [Wan89b, Wan89a, Wu94, Rit50], and resultants [Gel94, Stu, Man93, MC93] have been extensively used for that purpose. These methods, however, involve intensive symbolic computations which are very time and space-demanding. In many cases the computations fail due to limitations in memory and speed.

Homotopy continuation is a robust and versatile global method capable of finding all the solutions of a given system [AG93]. Although the theoretical foundations encompass many different areas of mathematics, the idea behind homotopy is rather intuitive: the solutions of a known "easy" system are deformed into the solutions of the wanted system. The method has been applied to problems in various areas, including robotics, kinematics of mechanisms, chemical equilibrium, geometric intersection [Mor87, WMS90, Pat92, Ver96, Ver97, HS95, Hub96, LI97] and, more recently, to constraint solving [LM95].

This work describes *Continuum*, a general-purpose solver for polynomial systems by homotopy continuation, which has been extensively used in [Dur98] to solve systems of geometric constraints. It is assumed that the system to be solved has the same number of equations and variables.

This work is organized as follows. Section 2 provides the theoretical foundations of homotopy continuation. Section 3 describes the implementation of the method. Finally, section 4 describes the syntax shows some examples of usage

2 Homotopy Continuation

For more than a century homotopy has played an important role in many areas of modern mathematics, and its use as a tool to solve systems of linear equations can be traced back at least to Lahaye [Lah34].

Homotopy continuation is a technique for numerically approximating a solution curve c which is implicitly defined by an under-determined system of equations. It deforms the solutions of a "simpler" system (called *start system*) into the solutions of the system which needs to be solved (called *target system* or simply *target*). The solutions of the start system are referred as the *start points* of the homotopy.

2.1 Definitions and Notations

Let $x = (x_1, x_2, \ldots, x_n)$ and let the system of polynomial equations

$$\begin{cases} f_1(x_1, x_2, \ldots, x_n) = 0 \\ f_2(x_1, x_2, \ldots, x_n) = 0 \\ \qquad \vdots \\ f_n(x_1, x_2, \ldots, x_n) = 0 \end{cases}$$

be denoted by $F(x) = 0$.

Notice that all the systems considered here have zero-dimensional solution set, and therefore they have the same number of variables and equations.

The *total degree* of the system F (denoted by $deg(F)$) is defined by

$$deg(F) = \prod_{i=1}^{n} deg(f_i).$$

The *Jacobian* of a F (denoted by $J(x)$ or $dF(x)$) is a $n \times n$ matrix of partial derivatives

$$J(x) = \begin{bmatrix} \frac{\partial f_1}{\partial x_1} & \frac{\partial f_1}{\partial x_2} & \cdots & \frac{\partial f_1}{\partial x_n} \\ \frac{\partial f_2}{\partial x_1} & \frac{\partial f_2}{\partial x_2} & \cdots & \frac{\partial f_2}{\partial x_n} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial f_n}{\partial x_1} & \frac{\partial f_n}{\partial x_2} & \cdots & \frac{\partial f_n}{\partial x_n} \end{bmatrix}.$$

A solution x^* of $F(x) = 0$ is said to be *singular* if $det(J(x^*)) = 0$.

The *homotopy* or *continuation equation* is defined by

$$H(x, \lambda) = (1 - \lambda)G(x) + \lambda\gamma F(x) \tag{1}$$

where $x \in C$, $\lambda \in [0, 1)$, γ is a random complex number, $F(x)$ is the target system, and $G(x)$ is the start system. $G(x)$ is a system with n equations and n variables. Moreover, its structure resembles the structure of $F(x)$ and its solutions are known or can be easily calculated.

The homotopy continuation method proceeds by deforming each solution of system $G(x)$ into a solution of $F(x)$. This describes a path in $C^n \times [0, 1)$ called *homotopy or continuation path*. The issue of building a start system is addressed in section 2.3.

2.2 Solutions at Infinity and Projective Transformations

The *homogeneous part* of F, denoted by F^0 is the system obtained from F by deleting the terms of lower degree in each equation.

Example 1 *Let F be the system defined by*

$$\begin{cases} x_1x_2 - x_2^2 - 3x_1 + 3x_2 = 0 \\ x_1x_2 - 3x_1 - x_2 + 3 = 0 \end{cases}$$

Then F^0 is

$$\begin{cases} x_1x_2 - x_2^2 = 0 \\ x_1x_2 = 0 \end{cases}$$

A *solution at infinity* of a is a solution of F^0 where the first nonzero entry is equal to 1.

Example 2 *The solutions at infinity of F in the example 1 correspond to the solutions of F^0 which have either one of the forms $(1, x2)$ or $(0, 1)$.*

The first form satisfies F^0 for $x_2 = 0$, but the second form does not satisfy F^0. Therefore, the only solution at infinity of F is $(1, 0)$.

If a system has solutions at infinity, then the corresponding homotopy paths are divergent. Deciding if a path diverges is a difficult problem in general.

Fortunately, any polynomial system can be transformed into an equivalent system, which has no solutions at infinity [Mor86b]. This procedure is now explained in detail.

Let $F(x)$ be a system in n variables $x_1, \ldots, x_n$. The *homogenization* of F is the system $\hat{F}(y)$ with n equations and $n + 1$ variables defined by

$$\hat{f}_i(y_1, \ldots, y_{n+1}) = y_{n+1}^{d_i} f_i(\frac{y_1}{y_{n+1}}, \ldots, \frac{y_n}{y_{n+1}}),\ i = 1, \ldots, n \qquad (2)$$

where $d_i = deg(f_i)$. In other words, $\hat{F}$ can be obtained from F by replacing x_i by y_i and completing the degree of each term by multiplying the term by an appropriate power of y_{n+1}.

Example 3 *The homogenization of the system of example 1 is*

$$\hat{F}(y_1, y_2, y_3) = \begin{cases} y_1y_2 - y_2^2 - 3y_1y_3 + 3y_2y_3 = 0 \\ y_1y_2 - 3y_1y_3 - y_2y^3 + 3y_3^2 = 0 \end{cases}$$

If y^* is a solution of equation 2, then ky^* is also a solution for any $k \in C$. Therefore the solutions of 2 are complex lines through the origin in C^{n+1}. The set of all this lines form the complex projective space, denoted by CP^n.

There is an explicit relationship between the solutions of the original system and its homogenization. If $(y_1, y_2, \ldots, y_n, y_{n+1})$ with $y_{n+1} \neq 0$ is a solution of $\hat{F}(y) = 0$ then

$$x = (\frac{y_1}{y_{n+1}}, \frac{y_2}{y_{n+1}}, \ldots, \frac{y_n}{y_{n+1}}) \tag{3}$$

is a solution for $F(x) = 0$. Conversely, if $x = (x_1, x_2, \ldots, x_n)$ is a solution to $F(x) = 0$, then

$$y = (x_1, x_2, \ldots, x_n, 1) \tag{4}$$

is a solution for $\hat{F}(y) = 0$. The solutions for $\hat{F}(y) = 0$, where $y_{n+1} = 0$ are the solutions at infinity.

Let $L(x) = \sum_{i=1}^{n} b_ix_i + b_{n+1}$, where the constants b_i, $i = 1, \ldots, n+1$ are random complex numbers and $b_{n+1} \neq 0$. The *projective transformation* of F (denoted by $\bar{F}$) is the polynomial system with n variables defined by

$$\bar{f}_i(x_1, x_2, \ldots, x_n) = \hat{f}_i(x_1, x_2, \ldots, x_n, L(x)),\ i = 1, \ldots, n \tag{5}$$

Theorem 1 **(Bezout)** *Let $F(x) = 0$ be a polynomial system and $d = deg(F)$. Then*

1. *The total number of geometric isolated solutions and solutions at infinity of $F(x) = 0$ is no more than d.*

2. *If $F(x) = 0$ has a finite number of solutions and solutions at infinity, then the total number of solutions and solutions at infinity of $F(x) = 0$ (counting multiplicities) is exactly d.*

Theorem 1 establishes an upper bound for the number of solutions of a given system (sometimes called the *Bezout number or Bezout bound* of the system). In [Mor86b, Mor87] Morgan uses theorem 1 to prove the following statement:

Theorem 2 *Let $\bar{F}(x)$ and $L(x)$ be as in 5. If F has a finite number of solutions and solutions at infinity, then, for almost all $b_i \in C$, $i = 1, \ldots, n+1$, $b_{n+1} \neq 0$, $\bar{F}$ has no solution at infinity.*

The "almost all" condition of theorem 2 is satisfied in practice by selecting random complex numbers for $b_1, \ldots, b_{n+1}$.

Finally, notice that the use of projective transformations does not increase the dimensionality of the problem, since $deg(f_i) = deg(\bar{f}_i)$, $i = 1, \ldots, n$ and $\bar{F}(x)$ has n variables.

2.3 Start System

Let $F(x)$ be the target system, $d = deg(F)$, and $L(x)$ defined as in the previous section. The *standard start system* $\bar{G}(x)$ is defined by

$$\begin{cases} \bar{g}_i(x) = x_i^{d_i} - \alpha^{d_i} \\ \alpha = L(x) \end{cases}, \; i = 1, \ldots, n. \tag{6}$$

Then $\bar{G}(x) = 0$ has $d = deg(F)$ solutions, given by

$$\begin{cases} x_i = k\xi_{d_i,j_i}, \; i = 1, \ldots, n; \; j_i = 0, \ldots, d_{i-1} \\ \alpha = k \end{cases} \tag{7}$$

where

$$k = \frac{b_{n+1}}{1 - \sum_{i=1}^{n} b_i \xi_{d_i,j_i}} \tag{8}$$

and

$$\xi_{d_i,j_i} = \cos(\frac{2\pi j_i}{d_i}) + i\sin(\frac{2\pi j_i}{d_i})$$

(the d_i-th roots of the unity). This can be easily verified by substituting 7 and 8 into 6.

The homotopy can now be restated as

$$\bar{H}(x, \lambda) = (1 - \lambda)\bar{G}(x) + \lambda\gamma\bar{F}(x), \tag{9}$$

and is sometimes referred as *standard homotopy*. It generates $deg(F)$ paths in $C^n \times [0, 1)$ starting at each of the solutions of $\bar{G}(x)$ for $\lambda = 0$.

Theorem 3 *For almost all choices of $b_1, \ldots, b_n$ and γ the homotopy 9 generates a collection of d non-overlapping converging smooth paths. Moreover, if a solution has multiplicity m, then m homotopy paths converge to it (see [Mor86a] and [Mor87]).*

Theorem 3 states that the homotopy paths generated by 9 are "well behaved". A typical situation is presented on figure 1. The issue of singular solutions (multiple paths converging to the same solution) is discussed in the next section.

It should be noted that theorem 3 still holds if different start systems are used. As a general rule, the start system should be selected in such a way that

1. its structure is similar to the structure of $F(x)$, and

2. its roots are known or can be easily computed.

It is clear that the standard start system satisfies the conditions above.

The start system plays a central role in the success of homotopy continuation. Even though theorem 3 guarantees that the homotopy paths are smooth, the more the start system resembles the target, the shorter and smoother the

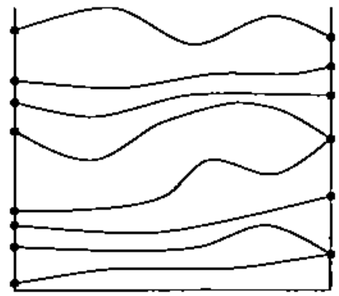

Figure 1: Typical behavior of homotopy paths with 2 double roots and 4 simple roots

homotopy paths are. This is an important issue in practice, since very long or almost singular paths can cause numerical problems[1].

On the other hand, solving these "non-standard" start systems is a problem in itself, where homotopy continuation can also be used. Polyhedral homotopy continuation [HS95, VGC96, HS97] can also be used as an alternate method.

2.4 Parameter Homotopy

As shown in the previous section, the coefficients of the homotopy $H(x, \lambda)$ change as λ varies from 0 to 1. However, in many applications, the coefficients of the system depend on parameters. Systems associated with constraint problems, for instance, exhibit such a parametric structure, since their coefficients are functions of the constraints.

In parameter homotopies, the continuation is generated in the *parameter space* Q, instead of the coefficient space. Therefore, the special structure the parameters impose on the solution set can be exploited. As a practical result, fewer paths have to be tracked to solve the system, significantly reducing the total numerical cost in some cases.

In the following discussion, let $F^q(x)$ denote the system obtained for some set of parameter values $q = (q_1, \ldots, q_s) \in Q$.

The results presented in [MS89] can be used as the basis for a two-step method for solving $F^q(x)$.

1. For a random $q^0 \in Q$, solve F^{q^0} using standard homotopy continuation, and

2. Solve F^q using F^{q^0} as the start system and its nonsingular affine solutions, as the start points.

[1] A path is almost singular if the determinant of the Jacobian matrix is smaller than a threshold for one or more points along the path. Those points are referred as *almost singular* points.

Since only the nonsingular affine solutions of F^{q^0} are used as start points, much fewer paths have to be tracked, when compared to standard homotopy. Furthermore, all nonsingular affine solutions of F^q are guaranteed to be computed. Notice, however, that some solutions at infinity might be computed for special sets of parameter values.

The cost of solving F^{q^0} cannot be discounted, but it is acceptable in applications where homotopy continuation will be used repeatedly. The same F^{q^0} can be used for solving many systems. This is consistent with the theory and does not seem to cause numerical difficulties. For further details refer to [MS89].

3 Implementation

This section describes *Continuum*, a homotopy continuation solver for systems of algebraic equations. Table 1 shows the outline of the algorithm.

If standard homotopy continuation is used, $\bar{G}(x)$ is initialized as the system defined on section 2.3. Alternatively, the user can provide the start system and start points. This becomes necessary when parameter homotopy has to be used.

The algorithm generates a new path starting at each solution of $\bar{G}(x) = 0$ and proceeds by computing points along the homotopy path using a *predictor-corrector* scheme, as λ varies from 0 to 1.

Nsteps corresponds to the current number of predictor-corrector steps performed on a given path. $MAXSTEPS$ corresponds to the maximum number of steps allowed per path. The continuation is aborted (Path failure) if more than $MAXSTEPS$ steps are performed[2].

Step corresponds to the current value of the step size. It is initialized to $STEP$, but its value is not fixed. If the corrector does not converge, the *Step* size is divided by two, and is doubled after $SAFE$ successful steps. Appendix A introduces the other parameters used in Continuum, and summarizes their default values, which are used in all the computations performed in this work, unless otherwise stated.

3.1 Predictor-Corrector

The predictor function computes a point on the line tangent to the homotopy path at the point x^*. More specifically, if (x^*, λ) is the current point on the homotopy path, then

$$(x^*, \lambda) + \left(\frac{dx}{d\lambda}(\lambda), 1\right)$$

gives the coordinates of the tip of the tangent vector. The corrector function attempts to solve $\bar{H}(x', \lambda') = 0$ using Newton-Raphson method, where

$$(x', \lambda') = (x^*, \lambda) + \left(\frac{dx}{d\lambda}(\lambda), 1\right) EPS$$

[2]Path failure can also occur when the path is almost singular (see section 4)

Table 1: Homotopy Continuation Algorithm

```
INPUT: F
OUTPUT: All solutions for F(x) = 0

For each solution x* of G̃(x) = 0 do begin
    λ := 0;
    nsteps := 0;
    step := STEP;
    success := 0;
    While λ < 1 and nsteps ≤ MAXSTEPS do begin
        Repeat
            x' := Predictor(x*);
            x* := Corrector(x');
            If correction did not converge then begin
                step := step/2;
                success := 0;
            end
            Else begin
                nsteps := nsteps + 1;
                Update λ;
                success := success + 1;
                If success = NSAFE then begin
                    step := 2step;
                    success := 0;
                end;
            end;
        Until correction converges;
    end;
    If nsteps > MAXSTEPS then
        Path Failure;
    Else
        (x*_1/L(x*), ..., x*_1/L(x*)) is a solution for F(x) = 0;
end
```

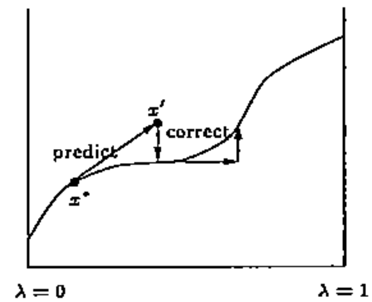

Figure 2: x^* is a point on the curve and x' is the predicted point.

and the scale factor EPS is chosen such that $|(x', \lambda') - (x^*, \lambda)|$ equals the current value of *step*. The situation is depicted on figure 2.

The difficulty with this approach is that now $\frac{dx}{d\lambda}(\lambda)$ must be computed explicitally. Let $x(\lambda)$ be a parametrization of the path as λ goes from 0 to 1, and x a point on the homotopy path. Then

$$\bar{H}(x, \lambda) = 0, \ t \geq 0.$$

Define

$$w(\lambda) = (w_1, w_2, \ldots, w_n, w_{n+1}) = (x(\lambda), \lambda) = (x_1, x_2, \ldots, x_n, \lambda) \tag{10}$$

Therefore

$$\bar{H}(x, \lambda) = \bar{H}(w(\lambda)) \tag{11}$$

Applying the chain rule to 11 yields

$$\frac{d\bar{H}}{dw} \cdot \frac{dw}{d\lambda} = 0, \tag{12}$$

where $\frac{d\bar{H}}{dw}$is an $n \times (n+1)$ matrix and $\frac{dx}{d\lambda}$ a $n \times 1$ vector.

Based on 10, equation 11 can be written in block form

$$\left[\begin{array}{c|c} \frac{d\bar{H}}{dx} & \frac{d\bar{H}}{d\lambda} \end{array} \right] \cdot \left[\begin{array}{c} \frac{dx}{d\lambda} \\ 1 \end{array} \right] = \left[\ 0 \ \right] \tag{13}$$

where $\frac{d\bar{H}}{dx}$ is a $n \times n$ matrix, $\frac{d\bar{H}}{d\lambda}$ is a $n \times 1$ vector and $\frac{dx}{d\lambda}$ is a $n \times 1$ vector.

Finally, equation 13 can be rewritten

$$\frac{d\bar{H}}{dx} \cdot \frac{dx}{d\lambda} = -\frac{d\bar{H}}{d\lambda} \tag{14}$$

Equation 14 is a linear system and can be easily solved using by a variety of methods like LU-factorization and singular value decomposition [SB93]

3.2 Random Parameters

As stated in theorems 2 and 3, the random complex parameters b_1, b_2, ..., b_{n+1} and γ are essential to make the algorithm work, but care must be taken when choosing these values. If these parameters have arbitrarily large norms, numerical problem related to scaling may be introduced.

In practice, this problem is circumvented by selecting $b_1, b_2, \ldots, b_{n+1}$ and γ among the complex numbers of norm 1.

3.3 Singular Solutions

According to theorem 3 the homotopy paths associated to 9 do not have singular points for $\lambda \in [0, 1)$, but the system may have singular solutions for $\lambda = 0$

Moreover, it is important to know if a solution is singular, since special techniques must be required to compute the solution accurately. It is important to emphasize, however, that singular solutions are a "local" or "postprocessing" problem, rather than a homotopy continuation issue.

In the neighborhood of a singular solution, Newton-Raphson method still converges, but slower (usually linear convergence), and with fewer (about half) significant digits. In this case, the singular solution is regarded as a *nice singular solution*. However, exceptionally bad behavior is possible. One can experience "arbitrarily slow convergence" or cyclic behavior. These solutions are called *nasty singular solutions.* Only nice singular solutions were encountered in the course of this work.

The *condition* [Gau97] of the Jacobian matrix can be used to classify (independently of scaling) the solutions into *singular* and *possibly singular*. Different algorithms can then be used to refine the solutions depending on the preliminary classification.

A slightly modified version of Newton-Raphson method can be used to refine nice singular solutions. This version should use a larger number of steps, more conservative "epsilons" and a test for singularity (to prevent overflow).

3.4 Computation of $d\bar{F}(x)$

It is more efficient to substitute $L(x)$ numerically, instead of symbolically. Consider, for example, the calculation of $d\bar{F}(x)$.

Let $\bar{F}$ be the defined by composition of two functions:

$$\bar{F} = \hat{F} \circ v : C^n \to C^n,$$

where $\hat{F} : C^{n+1} \to C^n$ is defined as in 2 and

$$\begin{aligned} &v : C^n \to C^{n+1} \\ &v(x) = (x_1, x_2, \ldots, x_n, L(x)). \end{aligned}$$

with $L(x) = b_1x_1 + \cdots + b_nx_n + b_{n+1},\ b_i \in C,\ i = 1, \ldots, n,\ b_{n+1} \neq 0$

The Jacobian of $\bar{F}(x)$ can be calculated by

$$d\bar{F}(x) = d\hat{F}(y) \circ dv(x), \tag{15}$$

where $y = v(x)$,

$$d\hat{F}(y) = \begin{bmatrix} \frac{\partial \hat{f}_1}{\partial y_1} & \frac{\partial \hat{f}_1}{\partial y_2} & \cdots & \frac{\partial \hat{f}_1}{\partial y_n} & \frac{\partial \hat{f}_1}{\partial y_{n+1}} \\ \frac{\partial \hat{f}_2}{\partial y_1} & \frac{\partial \hat{f}_2}{\partial y_2} & \cdots & \frac{\partial \hat{f}_2}{\partial y_n} & \frac{\partial \hat{f}_2}{\partial y_{n+1}} \\ \vdots & \vdots & \ddots & \vdots & \\ \frac{\partial \hat{f}_n}{\partial y_1} & \frac{\partial \hat{f}_n}{\partial y_2} & \cdots & \frac{\partial \hat{f}_n}{\partial y_n} & \frac{\partial \hat{f}_n}{\partial y_{n+1}} \end{bmatrix} = \{\hat{F}\}_{i,j=1}^{n} \tag{16}$$

and

$$dv(x) = \begin{bmatrix} 1 & 0 & \cdots & 0 \\ 0 & 1 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 1 \\ b_1 & b_2 & \cdots & b_n \end{bmatrix}. \tag{17}$$

According to 15, $d\bar{F}(x)$ is obtained by multiplying 16 by 17 Since 17 has a very simple structure, the $n \times n$ matrix $d\bar{F}(x) = \{\bar{F}\}_{i,j=1}^{n}$ is defined by

$$d\bar{F}_{i,j} = d\hat{F}_{i,j} + b_j \cdot d\hat{F}_{i,n+1}, \quad i, j = 1, \ldots, n.$$

3.5 Computation of $d\bar{H}(x)$ and $d\bar{H}(\lambda)$

The formulas for computing $d\bar{H}(x)$ and $d\bar{H}(\lambda)$ are obtained from 9 by differentiation.

$$d\bar{H}(x) = (1-\lambda)d\bar{G}(x) + \lambda\gamma d\bar{F}(x)$$
$$d\bar{H}(\lambda) = d\bar{F}(\lambda) - d\bar{G}(\lambda).$$

Example 4 *Let F be the system.*

$$\begin{cases} x_0 + x_1 + x_2 + x_3 - 1 = 0 \\ x_0 + x_1 - x_2 + x_3 - 3 = 0 \\ x_0^2 + x_1^2 + x_2^2 + x_3^2 - 4 = 0 \\ x_0^2 + x_1^2 + x_2^2 + x_3^2 - 2x_0 - 3 = 0 \end{cases}$$

Standard homotopy continuation tracks 4 *paths. Two of the paths correspond to solutions at infinity. The other two converge to the following real roots*

$$x = (0.5, -0.151388, -1, 1.65139)$$
$$x = (0.5, 1.65139, -1, -0.151388)$$

Example 5 *Let F be the system*

$$\begin{cases} 2x_0 + x_1 + x_2 + x_3 + x_4 - 6 = 0 \\ x_0 + 2x_1 + x_2 + x_3 + x_4 - 6 = 0 \\ x_0 + x_1 + 2x_2 + x_3 + x_4 - 6 = 0 \\ x_0 + x_1 + x_2 + 2x_3 + x_4 - 6 = 0 \\ x_0x_1x_2x_3x_4 - 1 = 0 \end{cases}$$

Standard homotopy continuation tracks 4 *paths. Three of these paths correspond to real solutions and the remaining two, to conjugate complex solutions.*

$x = (1,\ 1,\ 1,\ 1,\ 1)$
$x = (0.916351,\ 0.916351,\ 0.916351,\ 0.916351,\ 1.41825)$
$x = (-0.57902,\ -0.57902,\ -0.57902,\ -0.57902,\ 8.8951)$
$x = (-0.068558 - 0.610028i,\ -0.068558 - 0.610028i,$
$\quad -0.068558 - 0.610028i,\ -0.068558 - 0.610028i,\ 6.34279 + 3.05014i)$
$x = (-0.0695828 + 0.609968i,\ -0.0695828 + 0.609968i,$
$\quad -0.0695828 + 0.609968i,\ -0.0695828 + 0.609968i,\ 6.34791 - 3.04984i)$

4 Usage and Examples

Continuum is a command-line tool. The command

```
Continuum < test.hh > test.out
```

solves the system described in `test.hh` and outputs the results on file `test.out`.

4.1 The Input File

The input file contains the target system and, optionally, the start system and the corresponding start points. If the start system and start points are not included, Continuum generates and uses the standard start system, as defined in section 2.3.

Table 2 shows one input file corresponding to example 4. It is structured into 3 blocks delimited by a **begin**...**end** pair, which define the target system, the start system, and the start points, respectively.

The syntax of the target and start systems is very simple. If they involve n variables, each line must be a polynomial in $x_0, x_1, \ldots, x_{n-1}$. Each polynomial $f_i = f_i(x_0, x_1, \ldots, x_{n-1})$ must be written as a sum of terms

$$\sum_{j=1}^{m_i} c_{ij} x_0^{\alpha_{ij}^0} \ldots x_{n-1}^{\alpha_{ij}^{n-1}},$$

where m_i is the number of terms of f_i. Furthermore,

1. The exponents $\alpha_{ij}^0 \ldots \alpha_{ij}^{n-1}$ are positive integers (A variable with zero exponent should be omitted).

2. The coefficients c_{ij} can be either integer, real, or complex numbers. A complex number $a + bi$ is represented by the ordered pair (a, b). Real numbers follow the standard IEEE format.

3. Multiplication is denoted by juxtaposition.

4. Each polynomial is delimited by a semicolon.

5. The sign of the first term of each polynomial is *not* optional.

The ordered pair at the beginning of the third block informs the number of start points and the number of variables, respectively. The following lines list the actual start points. For instance, the input file shown on table 2 has 2 start points involving 4 variables.

Table 2: Input file corresponding to example 4.

```
begin
   +x0 + x1 + x2 + x3 - 1;
   +x0 + x1 - x2 + x3 - 3;
   +x0^2 + x1^2 + x2^2 + x3^2 - 4;
   +x0^2 + x1^2 + x2^2 + x3^2 - 2 x0 -3;
end

begin
   +x0 + x1 + x2 + x3 - (0.371234,0.928539);
   +x0 + x1 - x2 + x3 - (0.685677,0.727906);
   +x0^2 + x1^2 + x2^2 + x3^2 - (0.888296,0.459271);
   +x0^2 + x1^2 + x2^2 + x3^2 - 2x0 -(0.628353,0.777928);
end

begin
   (2 4)

   (0.129971500000,-0.159328500000)
   (-0.605147336515,0.450665817131)
   (-0.157221500000,0.100316500000)
   (1.003631336515,0.536885182869)

   (0.129971500000,-0.159328500000)
   (1.003631336515,0.536885182869)
   (-0.157221500000,0.100316500000)
   (-0.605147336515,0.450665817131)

end
```

4.2 The Output File

The output file summarizes the computation. Appendix B shows the output generated for the input file of table 2. It is divided into four sections. The first section shows the parameter values used (refer to appendix A). The second section shows the target and start systems (after homogenization), the random vector `b`, and the random constant `a`, which corresponds to γ. The vector `deg` contains the degrees of the polynomials of the target system.

The third section shows the number of paths to be tracked. If the start system is not provided, then standard homotopy is used. In this case, the number of paths equals the total degree of the target system. Otherwise, it corresponds to the number of start points given. Additional information for each individual path is also provided inside of the square brackets. The following notation is used:

`!` : The refinement step at the end of the path converged.

`x` : The refinement step at the end of the path did not converge.

`+` : Successful path tracking.

`-` : Path failure.

`(S)` : Almost singular point found during the tracking.

`(A)` : More than $MAXSINGULAR$ almost singular points found.

If the $REFINEMENT$ flag is off (refer to appendix A), `!` and `x` are not used. Notice that `x` does not imply in path failure (`-`). It only indicates that the refinement step did not converge given the parameters $REFIN\_EPS$ and $REFIN\_NITER$.

Finally, the fourth section lists all the solutions computed.

Appendix C presents another complete example.

4.3 Setting the Parameters

Continuum is called from a shell script, where the parameters are set to the default values. These values can be modified simply by editing the script.

References

[AG93] E.L. Allgower and K. Georg. Continuation and Path Following. *Acta Numerica*, pages 1–64, 1993.

[ALW95] I.A. Ajwa, Z. Liu, and P.S. Wang. Gröbner Bases Algorithm. Technical Report ICM-1999502-00, ICM, Department of Mathematics and Computer Science, Kent State University, 1995.

[Buc79] B. Buchberger. A Criterion for Detecting Unnecessary Reductions in the Construction of Gröbner Cases. In *EUROSAM'79, An International Symposium on Symbolic and Algebraic Manipulation*, pages 3–21, 1979.

[Buc85] B. Buchberger. Gröbner Bases: An Algorithmic Method in Polynomial Ideal Theory. In N.K. Bose, editor, *Recent Trends in Multidimensional Systems Theory*, pages 184–232. D. Reidel, 1985.

[CLO92] D. Cox, J. Little, and D. O'Shea. *Ideals, Varieties and Algorithms.* Springer-Verlag, 1992.

[DS83] J.E. Dennis and R.B. Schnabel. *Numerical Methods for Unconstrained Optimization and Nonlinear Equations.* Prentice-Hall, Englewood Cliffs, New Jersey, 1983.

[Dur98] C. Durand. *Symbolic and Numerical Techniques for Constraint Solving.* PhD thesis, Department of Purdue University, Purdue University, December 1998.

[Gau97] W. Gautschi. *Numerical Analysis: An Introduction.* Birkhäuser, 1997.

[Gel94] I.M. Gelfand. *Discriminants, Resultants, and Multidimensional Determinants.* Birkhauser, 1994.

[HS95] B. Huber and B. Sturmfels. A Polyhedral Method for Solving Sparse Polynomial Systems. *Mathematics of Computation*, 64(212):1541–1555, October 1995.

[HS97] B. Huber and B. Sturmfels. Bernstein's Theorem in Affine Space. *Discrete and Computational Geometry*, 17:137–141, 1997.

[Hub96] B. Huber. *Solving Sparse Polynomial Systems.* PhD thesis, Cornell University, 1996.

[Lah34] E. Lahaye. Une Méthode de Resolution d'une Categorie d'Equations Transcendantes. *Comptes Rendus des Séances de l'Acadmie des Sciences*, 198:1840–1842, 1934.

[LI97] T.Y. LI. Numerical Solutions of Multivariate Polynomial Systems by Homotopy Continuation Methods. *Acta Numerica*, 6:399–436, 1997.

[LM95] H. Lamure and D. Michelucci. Solving Geometric Constraints by Homotopy. In *Third Symposium on Solid Modeling and its Applications*, pages 263–269, Salt Lake City, Utah, 1995. ACM.

[Man93] D. Manocha. Efficient Algorithms for Multipolynomial Resultant. *The Computer Journal*, 36(5):485–496, 1993. Special issue on Quantifier Elimination.

[MC93] D. Manocha and J. Canny. Multipolynomial Resultant Algorithms. *Journal of Symbolic Computation*, 15(2):99–122, 1993.

[Mor86a] A.P. Morgan. A Homotopy for Solving Polynomial Systems. *Applied Mathematics and Computation*, 18:87–92, 1986.

[Mor86b] A.P. Morgan. A Transformation to Avoid Solutions at Infinity for Polynomial Systems. *Applied Mathematics and Computation*, 18:77–86, 1986.

[Mor87] Alexander Morgan. *Solving Polynomial Systems using Continuation for Engineering and Scientific Problems*. Prentice-Hall, Englewood Cliffs, NJ, 1987.

[Mor92] A.P. Morgan. Polynomial Continuation and its Relationship to the Symbolic Reduction of Polynomial Systems. In B.R. Donald, D. Kapur, and J.L. Mundy, editors, *Symbolic and Numerical Computation for Artificial Inteligence*. Academic Press, 1992.

[MS89] A.P. Morgan and A.J. Sommese. Coefficient-Parameter Polynomial Continuation. *Applied Mathematics and Computation*, 29:123–160, 1989.

[OR70] J. Ortega and W. Rheinboldt. *Iterative Solution of Nonlinear Equations in Several Variables*. Academic Press, New York, 1970.

[Pat92] N.M. Patrikalakis. Surface-to-Surface Intersections. *IEEE Computer Graphics and Applications*, 13(1):89–95, 1992.

[Rit50] J.F. Ritt. *Differential Algebra*. American Mathematical Society, 1950.

[SB93] J. Stoer and R. Bulirsch. *Introduction to Numerical Analysis*. Springer-Verlag, New York, 2nd edition, 1993.

[Stu] B. Sturmfels. Introduction to Resultants. Lecture notes at the AMS short course on Applications of Computational Algebraic Geometry, San Diego.

[Ver96] J. Verschelde. *Homotopy Continuation Methods for Solving Polynomial Systems*. PhD thesis, Katholieke Universiteit Leuven, 1996.

[Ver97] J. Verschelde. PHCPACK: A general-purpose solver for polynomial systems by homotopy continuation. Technical Report TW 265, Department of Computer Science, Katholieke Universiteit Leuven, 1997.

[VGC96] J. Verschelde, K. Gatermann, and R. Cools. Mixed-volume Computation by Dynamic Lifting Applied to Polynomial System Solving. *Discrete Computational Geometry*, 16(1):69–112, 1996.

[Wan89a] D. Wang. A Generalization of Characteristic Sets Algorithm. RISC-Linz Series 89-51.0, Johannes Kepler University, Austria, 1989.

[Wan89b] D. Wang. Characteristic Sets and Zero Structure of Polynomial Sets. RISC-Linz Lecture Notes, Johannes Kepler University, Austria, 1989.

[WMS90] C.W. Wampler, A.P. Morgan, and A.J. Sommese. Numerical Continuation Methods for Solving Systems arising in Kinematics. *Journal of Mechanical Design*, 112:59–68, 1990.

[Wu94] W. Wu. *Mechanical Theorem Proving in Geometries: Basic Principles.* Springer-Verlag, 1994.

A Parameters used by Continuum and their Default Values

In addition to $MAXSTEPS$, $STEP$ and $SAFE$, Continuum requires the following parameters to be specified, These parameters can be tuned to each individual problem.

1. $DETAIL$: Level of detail of the output.
2. $REFINEMENT$: If set to 1, a refinement step is performed at the end of each homotopy path.
3. $NEWTON\_EPS$: Tolerance used to check the convergence of the Newton-Raphson routine during correction.
4. $REFIN\_EPS$: Tolerance used to check the convergence of the Newton-Raphson routine during refinement. It is not used if $REFINEMENT = 0$.
5. $PIVOT\_EPS$: Tolerance used to to check for zero pivot during LU decomposition. If the pivot is smaller than $PIVOT\_EPS$, the matrix is considered singular.
6. $IMAG\_EPS$: Tolerance used in deciding if the imaginary part of a solution coordinate is 0.
7. $REAL\_EPS$: Tolerance used in deciding if the real part of a solution coordinate is 0.
8. $COMP\_EPS$: Tolerance used in deciding if a solution coordinate is 0.
9. $INFTY\_EPS$: Tolerance used to decide if a solution is a solution at infinity.
10. $NITER$: Maximum number of iterations of the Newton-Raphson routine during correction.
11. $REFIN\_NITER$: Maximum number of iterations of the Newton-Raphson routine during refinement.

12. *FILTER*: If set to 1, the solutions are filtered according to *FILTER_LIMIT*.

13. *FILTER_LIMIT*: If any coordinate of a solution has norm greater than *FILTER_LIMIT*, then the solution is considered a solution at infinity. It is not used if $FILTER = 0$.

14. *MAXSINGULAR*: Maximum number of almost singular points found on a path. If more than *MAXSINGULAR* points are found the path is aborted.

15. *TOO_BIG*: If set to 1 the solutions are not stored in memory for future post-processing. They are printed as they are computed. It is useful to solve large systems.

Parameter	Value
DETAIL	4
REFINEMENT	1
NEWTON_EPS	$1.0e-8$
REFIN_EPS	$1.0e-12$
IMAG_EPS	$1.0e-8$
REAL_EPS	$1.0e-8$
COMP_EPS	$1.0e-8$
PIVOT_EPS	$1.0e-20$
INFTY_EPS	$1.0e-4$
NITER	5
REFIN_NITER	100
STEP	0.01
NSAFE	5
MAXSTEPS	100
FILTER	1
FILTER_LIMIT	100
MAXSINGULAR	5
TOO_BIG	0

B Output file for example 4

```
Continuum 1.8 - Continuation-based Solver for Algebraic Systems
Computer Science, Purdue University, 1996-98

Parameters:
DETAIL      = 4
REFINEMENT  = 1
NEWTON_EPS  = 1.000000e-08
```

```
REFIN_EPS   = 1.000000e-12
IMAG_EPS    = 1.000000e-08
REAL_EPS    = 1.000000e-08
COMP_EPS    = 1.000000e-08
PIVOT_EPS   = 1.000000e-20
INFTY_EPS   = 1.000000e-04
NITER       = 5
REFIN_NITER = 100
STEP        = 1.000000e-02
NSAFE       = 5
MAXSTEPS    = 100
FILTER      = 0
FILTER_LIMIT= 1.000000e+02

Parsing Input System...No Errors on input!
Equations = 4

 **********************************************
 Homotopy system
a = (0.387237,0.92198)
bezout number = 2
b = [ (0.791597,0.611043) (0.564735,0.825273)
(0.865966,0.500103) (0.651847,0.75835) (1,0) ]
Target System
(1,0)*x0^1+(1,0)*x1^1+(1,0)*x2^1+(1,0)*x3^1+(-1,0)*x4^1+
(1,0)*x0^1+(1,0)*x1^1+(-1,0)*x2^1+(1,0)*x3^1+(-3,0)*x4^1+
(1,0)*x0^2+(1,0)*x1^2+(1,0)*x2^2+(1,0)*x3^2+(-4,0)*x4^2+
(1,0)*x0^2+(1,0)*x1^2+(1,0)*x2^2+(1,0)*x3^2+(-2,0)*x0^1*x4^1+(-3,0)*x4^2+
Start System
(1,0)*x0^1+(1,0)*x1^1+(1,0)*x2^1+(1,0)*x3^1+(-0.371234,-0.928539)*x4^1+
(1,0)*x0^1+(1,0)*x1^1+(-1,0)*x2^1+(1,0)*x3^1+(-0.685677,-0.727906)*x4^1+
(1,0)*x0^2+(1,0)*x1^2+(1,0)*x2^2+(1,0)*x3^2+(-0.888296,-0.459271)*x4^2+
(1,0)*x0^2+(1,0)*x1^2+(1,0)*x2^2+(1,0)*x3^2+(-2,0)*x0^1*x4^1+
    (-0.628353,-0.777928)*x4^2+
deg = [ 1 1 2 2 ]

 *********************************************
Number of Paths to be Tracked: 2

Tracking 2 possible paths...

[1!+] [2!+]

=====>>> Elapsed Time: 0.375294 seconds
Number of paths aborted: 0
```

```
==========================================================

                    SOLUTIONS

Real (2) Complex (0) At Infinity (0) Limits (0)

==========================================================

                  REAL SOLUTIONS
1-th Path, nsteps = 26
(0.500000000000,0.000000000000) (-0.151387818866,0.000000000000)
(-1.000000000000,0.000000000000) (1.651387818866,0.000000000000)

2-th Path, nsteps = 26
(0.500000000000,0.000000000000) (1.651387818866,0.000000000000)
(-1.000000000000,0.000000000000) (-0.151387818866,0.000000000000)

==========================================================

                 COMPLEX SOLUTIONS

==========================================================

                 LIMIT SOLUTIONS

==========================================================

                 SOLUTIONS AT INFINITY

==========================================================
```

C Another Example

The system presented in this section is extensivelly studied in [Dur98]. It corresponds to the problem of constructing a line tangent to 4 known spheres.

- Input file:

```
begin
   +x3^2+x4^2+x5^2-1;
   +x0x3+x1x4+x2x5;
   +x0^2+x1^2+6x1+x2^2-9x4^2+8;
   +x0^2+x1^2-6x1+x2^2-9x4^2+8;
   +x0^2+x1^2-14x1+x2^2-2x2-49x4^2-14x4x5-x5^2+49;
```

```
   +x0^2-2x0+x1^2-10x1+x2^2-4x2-x3^2-10x3x4-4x3x5-25x4^2
        -20x4x5-4x5^2+29;
end
```

- Output file:

```
Continuum 1.8 - Continuation-based Solver for Algebraic Systems
Computer Science, Purdue University, 1996-98

Parameters:
DETAIL       = 4
REFINEMENT   = 1
NEWTON_EPS   = 1.000000e-08
REFIN_EPS    = 1.000000e-12
IMAG_EPS     = 1.000000e-08
REAL_EPS     = 1.000000e-08
COMP_EPS     = 1.000000e-08
PIVOT_EPS    = 1.000000e-20
INFTY_EPS    = 1.000000e-04
NITER        = 5
REFIN_NITER = 100
STEP         = 1.000000e-02
NSAFE        = 5
MAXSTEPS     = 100
FILTER       = 0
FILTER_LIMIT= 1.000000e+02

Parsing Input System...No Errors on input!
Equations = 6

 *********************************************
 Homotopy system
a = (0.839398,0.543518)
bezout number = 64
b = [ (0.536953,0.843612) (0.518618,0.855006) (0.705336,0.708873)
(0.880625,0.473813) (0.384871,0.92297) (0.407086,0.91339) (1,0) ]
Target System
(1,0)*x3^2+(1,0)*x4^2+(1,0)*x5^2+(-1,0)*x6^2+
(1,0)*x0^1*x3^1+(1,0)*x1^1*x4^1+(1,0)*x2^1*x5^1+
(1,0)*x0^2+(1,0)*x1^2+(6,0)*x1^1*x6^1+(1,0)*x2^2+(-9,0)*x4^2
     +(8,0)*x6^2+
(1,0)*x0^2+(1,0)*x1^2+(-6,0)*x1^1*x6^1+(1,0)*x2^2+(-9,0)*x4^2
     +(8,0)*x6^2+
(1,0)*x0^2+(1,0)*x1^2+(-14,0)*x1^1*x6^1+(1,0)*x2^2+(-2,0)*x2^1*x6^1
     +(-49,0)*x4^2+(-14,0)*x4^1*x5^1+(-1,0)*x5^2+(49,0)*x6^2+
(1,0)*x0^2+(-2,0)*x0^1*x6^1+(1,0)*x1^2+(-10,0)*x1^1*x6^1+(1,0)*x2^2
```

```
     +(-4,0)*x2^1*x6^1+(-1,0)*x3^2+(-10,0)*x3^1*x4^1+(-4,0)*x3^1*x5^1
     +(-25,0)*x4^2+(-20,0)*x4^1*x5^1+(-4,0)*x5^2+(29,0)*x6^2+
Start System
(1,0)*x0^2+(-1,0)*x6^2+
(1,0)*x1^2+(-1,0)*x6^2+
(1,0)*x2^2+(-1,0)*x6^2+
(1,0)*x3^2+(-1,0)*x6^2+
(1,0)*x4^2+(-1,0)*x6^2+
(1,0)*x5^2+(-1,0)*x6^2+
deg = [ 2 2 2 2 2 2 ]

 *******************************************
Number of Paths to be Tracked: 64

Tracking 64 possible paths...

[1x+] [2x+] [3x+] [4x+] [5x+] [6x+] [7x+] [8!+] [9!+] [10!+]
[11x+] [12!+] [13!+] [14!+] [15x+] [16!+] [17!+] [18x+] [19!+] [20!+]
[21!+] [22x+] [23x+] [24x+] [25x+] [26x+] [27!+] [28!+] [29x+] [30!+]
[31x+] [32!+] [33!+] [34x+] [35x+] [36x+] [37!+] [38x+] [39x+] [40x+]
[41x+] [42x+] [43x+] [44x+] [45x+] [46x+] [47x+] [48!+] [49!+] [50!+]
[51x+] [52x+] [53!+] [54x+] [55!+] [56x+] [57x+] [58!+] [59x+] [60x+]
[61x+] [62!+] [63x+] [64x+]

=====>>> Elapsed Time: 76.6876 seconds
Number of paths aborted: 0

=======================================================

                   SOLUTIONS

Real (8) Complex (16) At Infinity (40) Limits (0)

=======================================================
                 REAL SOLUTIONS
12-th Path, nsteps = 39
(-0.541030812433,0.000000000000) (0.000000000000,0.000000000000)
(0.294941920212,0.000000000000) (0.125658294384,0.000000000000)
(0.964924008309,0.000000000000) (0.230503039549,0.000000000000)
14-th Path, nsteps = 34
(0.565454517450,0.000000000000) (0.000000000000,0.000000000000)
(0.187172830160,0.000000000000) (-0.084140101598,0.000000000000)
(0.963487443247,0.000000000000) (0.254189673292,0.000000000000)
19-th Path, nsteps = 62
(0.565454517450,0.000000000000) (0.000000000000,0.000000000000)
(0.187172830160,0.000000000000) (0.084140101598,0.000000000000)
```

```
(-0.963487443247,0.000000000000) (-0.254189673292,0.000000000000)
27-th Path, nsteps = 41
(-0.225724345720,0.000000000000) (0.000000000000,0.000000000000)
(0.811917097587,0.000000000000) (-0.172898246526,0.000000000000)
(-0.983766054710,0.000000000000) (-0.048068138593,0.000000000000)
33-th Path, nsteps = 36
(-0.225724345720,0.000000000000) (0.000000000000,0.000000000000)
(0.811917097587,0.000000000000) (0.172898246526,0.000000000000)
(0.983766054710,0.000000000000) (0.048068138593,0.000000000000)
37-th Path, nsteps = 65
(0.777804377366,0.000000000000) (0.000000000000,0.000000000000)
(-0.073855672310,0.000000000000) (0.019666790154,0.000000000000)
(0.978118051827,0.000000000000) (0.207119033543,0.000000000000)
49-th Path, nsteps = 49
(-0.541030812433,0.000000000000) (0.000000000000,0.000000000000)
(0.294941920212,0.000000000000) (-0.125658294384,0.000000000000)
(-0.964924008309,0.000000000000) (-0.230503039549,0.000000000000)
53-th Path, nsteps = 74
(0.777804377366,0.000000000000) (0.000000000000,0.000000000000)
 (-0.073855672310,0.000000000000) (-0.019666790154,0.000000000000)
(-0.978118051827,0.000000000000) (-0.207119033543,0.000000000000)
===================================================
                 COMPLEX SOLUTIONS
8-th Path, nsteps = 50
(-20.591388173181,16.798652742161) (0.000000000000,0.000000000000)
(-16.811600210558,-20.574158420389) (2.199597369133,-2.354763798077)
 (1.007995700888,-0.002541166928) (2.353487613284,2.201878485049)

9-th Path, nsteps = 42
(0.909583750877,2.699528493742) (0.000000000000,0.000000000000)
 (2.904508769077,-0.833754389715) (-0.089264218475,-0.469651829608)
 (1.015493115843,0.003698292674) (-0.438458291966,0.104180249553)

10-th Path, nsteps = 44
(0.909583750877,-2.699528493742) (0.000000000000,0.000000000000)
 (2.904508769077,0.833754389715) (-0.089264218475,0.469651829608)
 (1.015493115843,-0.003698292674) (-0.438458291966,-0.104180249553)

13-th Path, nsteps = 41
(3.462981149984,-5.725434540256) (0.000000000000,0.000000000000)
 (6.048846369144,3.476649224401) (-1.183400957819,1.741279050875)
 (1.146748895756,0.116527026379) (-1.644271210718,-1.171950250816)

16-th Path, nsteps = 42
(-20.591388173181,-16.798652742157) (0.000000000000,0.000000000000)
 (-16.811600210554,20.574158420389) (2.199597369133,2.354763798077)
```

```
(1.007995700888,0.002541166928) (2.353487613284,-2.201878485049)

17-th Path, nsteps = 33
(3.462981149984,5.725434540256) (0.000000000000,0.000000000000)
 (6.048846369144,-3.476649224401) (1.183400957819,1.741279050875)
 (-1.146748895756,0.116527026379) (1.644271210718,-1.171950250816)

20-th Path, nsteps = 45
(3.462981149984,-5.725434540256) (0.000000000000,0.000000000000)
 (6.048846369144,3.476649224401) (1.183400957819,-1.741279050875)
(-1.146748895756,-0.116527026379) (1.644271210718,1.171950250815)

21-th Path, nsteps = 46
(-47.963546243070,0.568028395867) (0.000000000000,0.000000000000)
(-0.161254780193,49.437226046515) (16.525305119684,0.031339318357)
 (-0.976811100897,4.005851878591) (-0.211763377903,-16.032369803857)

28-th Path, nsteps = 38
(0.909583750877,-2.699528493742) (0.000000000000,0.000000000000)
 (2.904508769077,0.833754389715) (0.089264218475,-0.469651829608)
 (-1.015493115843,0.003698292674) (0.438458291966,0.104180249553)

30-th Path, nsteps = 35
(-47.963546243071,0.568028395866) (0.000000000000,0.000000000000)
 (-0.161254780194,49.437226046515) (-16.525305119685,-0.031339318357)
(0.976811100897,-4.005851878592) (0.211763377903,16.032369803858)

32-th Path, nsteps = 37
(-20.591388173177,-16.798652742156) (0.000000000000,0.000000000000)
 (-16.811600210553,20.574158420384) (-2.199597369133,-2.354763798077)
 (-1.007995700888,-0.002541166928) (-2.353487613284,2.201878485049)

48-th Path, nsteps = 46
(3.462981149984,5.725434540256) (0.000000000000,0.000000000000)
 (6.048846369144,-3.476649224401) (-1.183400957819,-1.741279050875)
 (1.146748895756,-0.116527026379) (-1.644271210718,1.171950250815)

50-th Path, nsteps = 88
(-47.963546243071,-0.568028395866) (0.000000000000,0.000000000000)
 (-0.161254780193,-49.437226046515) (16.525305119685,-0.031339318357)
 (-0.976811100897,-4.005851878591) (-0.211763377903,16.032369803857)

55-th Path, nsteps = 36
(0.909583750877,2.699528493742) (0.000000000000,0.000000000000)
(2.904508769077,-0.833754389715) (0.089264218475,0.469651829608)
(-1.015493115843,-0.003698292674) (0.438458291966,-0.104180249553)
```

```
58-th Path, nsteps = 40
(-47.963546243071,-0.568028395866) (0.000000000000,0.000000000000)
 (-0.161254780194,-49.437226046515) (-16.525305119684,0.031339318357)
 (0.976811100897,4.005851878591) (0.211763377903,-16.032369803857)

62-th Path, nsteps = 43
(-20.591388173181,16.798652742161) (0.000000000000,0.000000000000)
 (-16.811600210559,-20.574158420388) (-2.199597369133,2.354763798077)
(-1.007995700888,0.002541166928) (-2.353487613284,-2.201878485049)
============================================================

                  LIMIT SOLUTIONS

============================================================

                  SOLUTIONS AT INFINITY
1-th Path, nsteps = 42


2-th Path, nsteps = 57


3-th Path, nsteps = 49


4-th Path, nsteps = 47


5-th Path, nsteps = 38


6-th Path, nsteps = 45


7-th Path, nsteps = 44


11-th Path, nsteps = 40


15-th Path, nsteps = 47


18-th Path, nsteps = 55
```

```
22-th Path, nsteps = 55

23-th Path, nsteps = 58

24-th Path, nsteps = 41

25-th Path, nsteps = 59

26-th Path, nsteps = 50

29-th Path, nsteps = 39

31-th Path, nsteps = 41

34-th Path, nsteps = 60

35-th Path, nsteps = 43

36-th Path, nsteps = 93

38-th Path, nsteps = 44

39-th Path, nsteps = 56

40-th Path, nsteps = 51

41-th Path, nsteps = 45

42-th Path, nsteps = 43
```

```
43-th Path, nsteps = 59

44-th Path, nsteps = 44

45-th Path, nsteps = 41

46-th Path, nsteps = 47

47-th Path, nsteps = 44

51-th Path, nsteps = 53

52-th Path, nsteps = 39

54-th Path, nsteps = 46

56-th Path, nsteps = 40

57-th Path, nsteps = 78

59-th Path, nsteps = 62

60-th Path, nsteps = 36

61-th Path, nsteps = 53

63-th Path, nsteps = 48

64-th Path, nsteps = 34
==================================================
```