A Method for Tracking Singular Paths with Application to the Numerical Irreducible Decomposition

Andrew J. Sommese[*] *Jan Verschelde*[†]
Charles W. Wampler[‡]

Abstract. In the numerical treatment of solution sets of polynomial systems, methods for sampling and tracking a path on a solution component are fundamental. For example, in the numerical irreducible decomposition of a solution set for a polynomial system, one first obtains a "witness point set" containing generic points on all the irreducible components and then these points are grouped via numerical exploration of the components by path tracking from these points. A numerical difficulty arises when a component has multiplicity greater than one, because then all points on the component are singular. This paper overcomes this difficulty using an embedding of the polynomial system in a family of systems such that in the neighborhood of the original system each point on a higher multiplicity solution component is approached by a cluster of nonsingular points. In the case of the numerical irreducible decomposition, this embedding can be the same embedding that one uses to generate the witness point set. In handling the case of higher multiplicities, this paper, in concert with the methods we previously proposed to decompose reduced solution components, provides a complete algorithm for the numerical irreducible decomposition. The method is applicable to tracking singular paths in other contexts as well.

2000 Mathematics Subject Classification. Primary 65H10, 14Q99; Secondary 68W30.
Key words and phrases. Component of solutions, embedding, interpolation, irreducible component, irreducible decomposition, generic point, homotopy continuation, numerical algebraic geometry, multiplicity, path following, polynomial system, sampling, singularity.

---

[*]Department of Mathematics, University of Notre Dame, Notre Dame, IN 46556-4618, U.S.A. *Email:* sommese@nd.edu *URL:* http://www.nd.edu/˜sommese

[†]Department of Mathematics, Statistics, and Computer Science, University of Illinois at Chicago, 851 South Morgan (M/C 249), Chicago, IL 60607-7045, U.S.A. *Email:* jan@math.uic.edu jan.verschelde@na-net.ornl.gov *URL:* http://www.math.uic.edu/˜jan

[‡]General Motors Research Laboratories, Enterprise Systems Lab, Mail Code 480-106-359, 30500 Mound Road, Warren, MI 48090-9055, U.S.A. *Email:* Charles.W.Wampler@gm.com

1. Introduction

Our main motivation for developing a method to track singular paths is to complete our numerical algorithms for computing the irreducible decomposition of the solution set of a polynomial system of equations. Conventional path tracking methods are limited to nonsingular paths; they are sufficient for handling solution components of multiplicity one, but cannot cope with components having multiplicity greater than one. With the addition of a path tracking algorithm applicable to these components, they can be treated in all other respects in the same manner as the components of multiplicity one. Moreover, the new path tracker can be applied in other similar contexts as well, even those in which the path to be followed is defined by functions that are merely analytic rather than polynomial.

Since it is our main motivation, we begin with a brief description of the irreducible decomposition problem. Let

$$f(\mathbf{x}) := \begin{bmatrix} f_1(x_1, \dots, x_N) \\ \vdots \\ f_n(x_1, \dots, x_N) \end{bmatrix}, \tag{1}$$

be a system of polynomial equations in $\mathbb{C}^N$, where for simplicity we assume that not all of the f_i are identically zero. For this system, the solution set, $Z = \mathbf{V}(f) := \{\mathbf{x} \in \mathbb{C}^N | f(\mathbf{x}) = \mathbf{0}\}$, can be written as the union

$$Z := \bigcup_{i=0}^{N-1} Z_i := \bigcup_{i=0}^{N-1} \bigcup_{j \in \mathcal{I}_i} Z_{ij} \tag{2}$$

where Z_i is the union of all i-dimensional irreducible components Z_{ij} of Z, and the index sets $\mathcal{I}_i$ are finite and possibly empty. Note that Z is the reduction of the possibly nonreduced algebraic set $f^{-1}(\mathbf{0})$ defined by the equations $f_1, \dots, f_n$. The irreducible decomposition problem is to determine all of the irreducible components, Z_{ij}, meaning, at a minimum, to enumerate the components and provide a set membership test for each of them.

In [16], we presented algorithms to numerically describe and manipulate the irreducible decomposition. In that article we gave an implementation of the algorithm for the components Z_{ij} of $\mathbf{V}(f)$, which occur with multiplicity one as components of $f^{-1}(\mathbf{0})$. In this article we present an implementation for the remaining components of $V(f)$ that have multiplicity at least two as components of $f^{-1}(\mathbf{0})$.

Components of multiplicity at least two specialize in the classical case of a one variable polynomial to roots of multiplicity at least two. Beyond the difficulties present in the classical case, there are a number of non-classical difficulties. For example, perturbations of the system can in some situations cause the whole component to disappear, or to change dimension.

One consequence of our new algorithm, is the first implemented homotopy method of finding exactly the isolated solutions of a polynomial system of n polynomials in n variables. All previous implementations deliver a finite set of solutions containing the isolated solutions, but also possibly containing non-isolated solutions. Though previous implementations picked out the nonsingular isolated solutions from this set, they gave little or no information about whether a singular solution was isolated or not.

Computing a numerical irreducible decomposition depends on an algorithm for continuing from a generic point on a solution component to nearby points on the component. In [16] and the related papers [17, 18, 19, 20], widely spaced points on a component are found by tracking paths along linear slices of the component. As we shall describe further below, the numerical exploration of a component by path tracking allows one to classify witness points on each component, thereby determining the degree of the component and creating an efficient membership test for the component.

The essential difficulty, which this paper overcomes, is that standard predictor-corrector path tracking fails on components of multiplicity greater than one. This is because every point on such a set is singular: the Jacobian matrix of the polynomial system evaluated at a point on the set has rank smaller than the co-dimension of the set. In other words, the null space of the Jacobian includes directions that are not in the tangent space of the set. Indeed, in extreme cases the Jacobian matrix can become identically zero, thus yielding no information about the tangent space of the underlying set. To predict a new point along the path, standard path trackers use the Jacobian to compute a tangent vector along the path, so they cannot handle singular sets. These trackers also use the Jacobian to compute corrections to the prediction via Newton's method. Here the ill-conditioning of the Jacobian in the vicinity of the singular solution set slows convergence and limits the final accuracy.

Besides the numerical irreducible decomposition, other problems can create the need to track singular paths. For example, as in [7, 9], suppose we have a family of polynomial systems, $f(\mathbf{x}, \mathbf{q}) = \mathbf{0}$, that depend analytically on some parameters $\mathbf{q} \in \mathbb{C}^M$. For some generic parameters $\mathbf{q}_0$, we may use continuation to find a set of solution points that contains the generically isolated solutions of $f(\mathbf{x}, \mathbf{q}_0) = \mathbf{0}$, some of which may be multiple roots. One may obtain such a set for another set of parameters $\mathbf{q}_1$ by tracking solution paths in the homotopy $f(\mathbf{x}, t\mathbf{q}_0 + (1-t)\mathbf{q}_1) = \mathbf{0}$. With standard path tracking, one could only apply this technique to the generically nonsingular solutions. The approach of this paper removes this restriction.

Although the above discussion has posed the problem in terms of polynomial systems, the method applies more generally to systems of analytic equations. We make no essential use of the polynomial nature of the equations. As prediction and correction both act locally, only the analytic properties of functions of several complex variables are germane.

The method we propose is related to previous work [10, 11, 12] in computing singular solutions of polynomial and analytic systems of equations. For polynomial systems, related recent work on dealing with components of solutions of multiplicity at least two is described in [3] and in [5, 6]. A semi-numerical approach to restore the quadratic convergence of Newton's method by deflation can be found

in [13] and [14]. The main purpose of this paper is outline algorithms that complement our numerical decomposition method in [16]. In section 5, we present a numerical experiment to illustrate feasibility on a small example.
Acknowledgments. We gratefully acknowledge the support of this work by Volkswagen-Stiftung (RiP-program at Oberwolfach). The first author thanks the Duncan Chair of the University of Notre Dame and National Science Foundation (DMS-0105653) for their support. The second author thanks the Department of Mathematics of the University of Illinois at Chicago and National Science Foundation (DMS-0105739) for their support. The third author thanks the General Motors Research Laboratories for their support.

2. Singular Path Tracking

In this section we present a general statement of the singular path-tracking problem, and explain our solution to the problem. In section 3, we explain in more detail how this technique is used as part of a larger algorithm for computing a numerical irreducible decomposition.

Our approach is to meld together the approach we used for path-tracking on a reduced component of the solution set of $f(\mathbf{x}) = \mathbf{0}$ with the classical homotopy continuation approach to isolated singular points of polynomial systems.

2.1. Path Tracking on Components

First we recall the bare bones framework for path-tracking on reduced components. We assume we have a solution $\mathbf{x}_0$ of $f(\mathbf{x}) = \mathbf{0}$ which is a general point of some reduced k-dimensional component Z_{kj} of $f^{-1}(\mathbf{0})$. Indeed, the inductive procedures of [16, 17, 18, 19] lead to such points. Moreover those approaches lead to a new system of equations

$$F(\mathbf{x}) := \begin{bmatrix} F_1(x_1, \ldots, x_N) \\ \vdots \\ F_{N-k}(x_1, \ldots, x_N) \\ L_1(x_1, \ldots, x_N) \\ \vdots \\ L_k(x_1, \ldots, x_N) \end{bmatrix}, \qquad (3)$$

where

1. Z_{kj} is a reduced component of the solution set of the equations $F_1 = \mathbf{0}, \ldots, F_{N-k} = \mathbf{0}$; and

2. the L_i are generic linear functions whose set of common zeroes is a generic k-dimensional linear space L transverse to Z_{kj} and containing $\mathbf{x}_0$.

An appropriate homotopy of the L_is, chosen so that L varies within a $(k+1)$-dimensional linear space L' gives a smooth path on the irreducible reduced curve $L' \cap Z_{kj}$.

In the situation where we have a nonreduced k-dimensional component, we again have the same sort of system F, plus the generic linear spaces L and L', except that the k-dimensional irreducible component Z_{kj} and the irreducible curve $L' \cap Z_{kj}$ are not reduced.

2.2. Singular Isolated Solutions

Next recall how an isolated solution $\mathbf{x}_0$ of multiplicity at least two of a system $f(\mathbf{x}) = \mathbf{0}$ is handled classically. We assume the system is square, i.e., $n = N$, since we usually reduce to such systems at the cost of possibly increasing the multiplicity ν of $\mathbf{x}_0$ when $\nu > 1$. The system is embedded in a larger system $f(\mathbf{x}, t) = \mathbf{0}$ of N equations in $N+1$ unknowns such that $f(\mathbf{x}) = f(\mathbf{x}, 0)$. In this case it follows from the upper semicontinuity of the dimension of fibers of an analytic map, that the zero set of $f(\mathbf{x}, t) = \mathbf{0}$ is a one-dimensional analytic set C in a neighborhood of $(\mathbf{x}_0, 0)$. The system $f(\mathbf{x}, t) = \mathbf{0}$ is chosen so that the projection map of C down to the $\mathbb{C}$ under the projection from $\mathbb{C}^N \times \mathbb{C}$ under to the second factor is ν-sheeted from a neighborhood of $(\mathbf{x}_0, 0)$ down to a neighborhood of 0. This setup gives smooth maps $\mathbf{x}_1(t), \ldots, \mathbf{x}_\nu(t)$ from $(0, t_0]$ to C for some real positive t_0 such that the $\mathbf{x}_i(t)$ are a cluster of distinct smooth isolated solutions of $f(\mathbf{x}, t) = \mathbf{0}$ and $\lim_{t \to 0} \mathbf{x}_i(t) \to \mathbf{x}_0$. It is a useful fact [11] that the centroid of the cluster, $(\mathbf{x}_1(t) + \cdots + \mathbf{x}_\nu(t))/\nu$ is holomorphic.

2.3. Singular Paths

Now let us explain how we meld these two setups together. As noted at the start of this section, we will explain in more detail in section 3 how this technique is used as part of a larger algorithm for computing a numerical irreducible decomposition. Let

$$g(\mathbf{v}, s, t) := \begin{bmatrix} g_1(v_1, \ldots, v_N, s, t) \\ \vdots \\ g_N(v_1, \ldots, v_N, s, t) \end{bmatrix}, \tag{4}$$

be a system of N functions analytic in $\mathbb{C}^{N+2}$. We call $\mathbf{v}$ the variables and (s, t) the parameters, and we are interested in solutions of the equations $g(\mathbf{v}, s, t) = \mathbf{0}$ for the variables given the values of the parameters. Note that in contrast to (1), this system is assumed to be square: it has the same number of functions as variables. We call a point $(\mathbf{v}, s, t)$ *singular* if the $N \times N$ Jacobian matrix of partial derivatives with respect to $\mathbf{v}$ is less than full rank at the point. In practice

1. we are interested in the irreducible components of $g(\mathbf{v}, 0, 0) = \mathbf{0}$;
2. the analytic set $g(\mathbf{v}, s, 0) = \mathbf{0}$ contains a curve, which would let us do the our continuation if it was reduced; and
3. the variable t plays the same role as in the classical procedure sketched for the isolated solution.

Let us make this precise by listing conditions under which our path tracking algorithm applies. First,

1. the solution set of $g(\mathbf{v}, s, 0) = \mathbf{0}$ is an irreducible one-dimensional curve C in a neighborhood of $\mathbf{v}_0$, whose reduction is isomorphic to a neighborhood of $0 \in \mathbb{C}$ under the projection $\mathbb{C}^N \times \{0\} \times \mathbb{C} \to \mathbb{C}$ given by $(\mathbf{v}, s, t) \to s$.

This in particular implies that $\mathbf{v}_0$ is an isolated solution of the system $g(\mathbf{v}, 0, 0) = \mathbf{0}$. We denote by $\mathbf{v}_0(s)$ the point corresponding to s under the above projection.

Next we assume that

2. the multiplicity ν of C in the analytic set defined by $g(\mathbf{v}, s, 0) = \mathbf{0}$ is the same as the multiplicity ν of $\mathbf{v}_0$ as an isolated solution of $g(\mathbf{v}, 0, 0) = \mathbf{0}$.

Using the above conditions, we know by semicontinuity of dimension of fibers of analytic maps that the solution set of $g(\mathbf{v}, s, t) = \mathbf{0}$ is a pure dimensional two-dimensional analytic set S in a neighborhood of $(\mathbf{v}_0, 0, 0)$. We further assume that,

3. for any fixed s in a neighborhood of $[0, 1]$, the projection $(\mathbf{v}, s, t) \to t$ gives a proper ν-sheeted finite mapping of a neighborhood of $(\mathbf{v}_0, s, 0)$ in the reduction of $C_s := S \cap \left(\mathbb{C}^N \times \{s\} \times \mathbb{C}\right)$ to a neighborhood of $0 \in \mathbb{C}$.

In the situation of section 3, we know this for $s = 0$, and by genericity of construction, for a Zariski open set of $s \in \mathbb{C}$, which by again using the genericity of the construction can be assumed to contain $[0, 1]$.

By the above condition there must exist a real positive t_0 such that

4. for all $s \in [0, 1]$, we have smooth maps $\mathbf{v}_1(s, t), \ldots, \mathbf{v}_\nu(s, t)$ from $\{s\} \times (0, t_0]$ to C_s such that the $\mathbf{v}_i(s, t)$ are a cluster of distinct smooth isolated solutions of $g(\mathbf{v}, s, t) = \mathbf{0}$ and $\lim_{t \to 0} \mathbf{v}_i(s, t) \to \mathbf{v}_0(s)$.

We emphasize some consequences of the above for the cluster $\mathcal{C} = \{\mathbf{v}_1, \ldots, \mathbf{v}_\nu\}$ of ν distinct starting points that satisfy $g(\mathbf{v}, 0, t_0) = \mathbf{0}$ for a positive real $t_0 < \delta$ for a sufficiently small δ. For a sufficiently small δ

5. the centroid of the cluster, $(\mathbf{v}_1(s, t) + \cdots + \mathbf{v}_\nu(s, t))/\nu$, is an analytic function of (s, t) in some neighborhood of $[0, 1] \times [0, \delta]$; and

6. for fixed (s,t) in this neighborhood and with $t \neq 0$, the cluster points $\mathbf{v}_i(s,t)$ are nonsingular isolated solutions of $g(\mathbf{v},s,t)=\mathbf{0}$.

Item 1 in this list simply asserts that for $t=0$, we have a well-defined solution path parameterized by s. We wish to track this path as s goes from zero to one on the reals and accurately compute the endpoint of this path. However, for $\nu > 1$, this solution path is a multiple solution and is therefore singular. Fortunately, for t in the vicinity of zero, item 4 asserts that the solution component of $g(\mathbf{v},s,0)=\mathbf{0}$ containing $\mathbf{v}_0$ is part of a solution component of $g(\mathbf{v},s,t)=\mathbf{0}$, which is smooth for $s \in [0,1]$ and $t \in (0,t_0]$ for some small positive t_0. It may not be apparent how such an embedding $g(\mathbf{v},s,t)$ of a given system $g(\mathbf{v},0,0)$ is constructed, but we can often arrange it as a consequence of the homotopies we use in solving systems of polynomial equations. It is important to note that the choice of a $\mathbf{v}_0$ on a given component for which this procedure works will depend on the given construction of the system $g(\mathbf{v},s,t)=\mathbf{0}$. A particular instance, the numerical irreducible decomposition problem, is discussed in the next section.

3. Numerical Irreducible Decomposition

In this section, we review the algorithm from [16] and explain how it can lead to a kind of singular path tracking problem. Improvements to the algorithm given in [17, 18, 19] also depend on path tracking and hence require singular path tracking to handle components of multiplicity greater than one. After a quick overview of these algorithms, we show how the particular systems arising in this context can be reformulated as in the foregoing section 2.

3.1. Overview of Decomposition Algorithms

Let $\widehat{W} = \{ \widehat{W}_i \mid i = 0,\ldots,N-1 \}$ denote the witness point superset produced by the routine **WitnessGenerate** of [16]. We refer the reader to [16] for the construction of $\widehat{W}_i$, but note that geometrically $\widehat{W}_i$ is a subset of the intersection of Z with a generic linear space L_{N-i} of dimension $N-i$, which contains $Z_i \cap L_{N-i}$. As explained in detail in [16, 21], we model generic objects, e.g., coefficients of linear equations, by using random number generators. Each $\widehat{W}_i$ contains generic points on each i-dimensional component, plus additional junk points which will be filtered out, i.e., points lying on irreducible components of dimension greater than i. In particular, we showed in [16] how to do a breakup of each $\widehat{W}_i$ of $\widehat{W}$ as $\widehat{W}_i := (\cup_{j\in\mathcal{I}_i} W_{ij}) \cup J_i$, where

1. W_{ij} consists of $\deg Z_{ij}$ generic points of Z_{ij} each occurring ν_{ij} times, where ν_{ij} is a positive integer. Moreover, $\nu_{ij} \geq \mu_{ij}$, where μ_{ij} is the multiplicity of

Z_{ij} in the possibly nonreduced algebraic set $f^{-1}(\mathbf{0})$, and $\nu_{ij} = 1$ if and only if $\mu_{ij} = 1$; and

2. $J_i \subset \cup_{k>i} Z_k$.

In [16, 17], this classification is accomplished by the construction of filtering polynomials

$$\{ \ p_{ij} \mid 0 \leq i \leq N-1 \ ; \ j \in \mathcal{I}_i \ \}. \tag{5}$$

The construction involves the operation of sampling the component containing a given generic point. The samples are used to interpolate and construct a filtering polynomial. The structure of the underlying set of a component of multiplicity at least two does not differ from the corresponding set for a multiplicity one component. If the component of multiplicity at least two could be sampled, the interpolation and further steps of the algorithm of [16] can be dealt with the same routines with no change. To have a stable interpolation method, it is necessary to sample the irreducible component at widely separated points. In [16] we showed how to implement this sampling for multiplicity one components using homotopy continuation. The difficulty is that for a component of multiplicity at least two, this would amount to tracking paths singular at every point.

In [18], the classification of the witness points into components is effected by finding points that are connected by monodromy loops. These loops must be numerically tracked, and any witness points that appear with multiplicity greater than one will require a singular path tracking algorithm. The monodromy method does not stand alone: in many instances an additional check is necessary to confirm that the monodromy classification is correct. In [19], it is shown that the computation of the linear trace is sufficient. As above, this is computed by interpolating points found by path tracking. Path tracking is fundamental to all these operations, and if we can succeed to accurately and reliably track paths on components of higher multiplicity, then these components can be treated exactly as the multiplicity-one components.

3.2. Witness Point Generation

In [15], witness points are generated using a cascade of homotopies, one dimension at a time. In case the number of equations n differs from the number of variables N, we add $N-n$ random hyperplanes to an underdetermined system, or $n-N$ extra "slack" variables to an overdetermined system. In the notation below we thus can take N as the maximum of N and n. The witness point set for dimension i is constructed with the homotopy

$$H(\mathbf{x}, \mathbf{z}, \lambda, t) := \begin{bmatrix} f_1(x_1, \ldots, x_N) + \gamma_{1,1} z_1 + \cdots + \gamma_{1,i+1} z_{i+1} \\ \vdots \\ f_N(x_1, \ldots, x_N) + \gamma_{N,1} z_1 + \cdots + \gamma_{N,i+1} z_{i+1} \\ z_1 + \lambda_{1,0} + \lambda_{1,1} x_1 + \cdots + \lambda_{1,N} x_N \\ \vdots \\ z_i + \lambda_{i,0} + \lambda_{i,1} x_1 + \cdots + \lambda_{i,N} x_N \\ z_{i+1} + t\,(\alpha_0 + \alpha_1 x_1 + \cdots + \alpha_N x_N) \end{bmatrix}. \tag{6}$$

The arguments in H are as follows:

1. $\mathbf{x} \in \mathbb{C}^N$ are the original variables from the system $f(\mathbf{x}) = \mathbf{0}$, $\mathbf{x} = (x_1, x_2, \ldots, x_N)$;
2. $\mathbf{z} \in \mathbb{C}^{i+1}$ are auxiliary variables in the embedding of the component, $\mathbf{z} = (z_1, z_2, \ldots, z_{i+1})$;
3. $t \in (0, 1]$ is the continuation parameter, going from one to zero;
4. $\lambda \in \mathbb{C}^{(i)\times(N+1)}$ is a matrix of generic coefficients for i hyperplanes.

In addition, both $\gamma \in \mathbb{C}^{N\times(i+1)}$ and $\alpha \in \mathbb{C}^{N+1}$ are generic parameters. The hyperplanes defined by λ slice any i-dimensional components down to isolated points; these are the witness points for the components. The hyperplanes also meet components of dimension greater than i, so some endpoints of the homotopy may lie on these sets. If so, these are the "junk" points J_i mentioned above.

In this homotopy we have smooth paths, indexed by k, of the form $(\mathbf{x}_k(t), \mathbf{z}_k(t)) : (0, 1] \to \mathbb{C}^N \times \mathbb{C}^{i+1}$, defined by $H(\mathbf{x}_k(t), \mathbf{z}_k(t), \lambda, t) = \mathbf{0}$. The witness point superset (which consists of both the proper witness points and the junk points) for dimension i is given by the endpoints of those paths that have all slack variables equal to zero; that is, those which are of the form

$$\lim_{t\to 0} (\mathbf{x}_k(t), \mathbf{z}_k(t)) = (\mathbf{w}_k, \mathbf{0}).$$

Since the slack variables are zero, one sees from (6) that $\mathbf{w}_k$ is both a solution point of $f = \mathbf{0}$ and lies on the linear slice defined by λ.

3.3. Sampling Components

In the homotopy above, the matrix λ defines a linear slice that cuts out the witness points. To classify the witness points and otherwise explore the components, one may move λ in a secondary homotopy and track the movements of the witness points. A common example is a linear homotopy going from $s = 0$ to $s = 1$ between the original slice $\lambda^{(1)}$ and a new slice $\lambda^{(2)}$ as

$$H(\mathbf{w}, \mathbf{0}, (1 - s)\lambda^{(1)} + s\lambda^{(2)}, 0) = \mathbf{0}, \tag{7}$$

where H is as defined in (6). For nonsingular components, this can be tracked by conventional methods to give a new sample on the component. These methods fail, however, when $\mathbf{w}$ is on a multiple component, because of the singularity of the Jacobian matrix for all points on the component.

This leads us back to the singular path tracking problem as defined in §2. We must begin with the cluster of ν points that approach $\mathbf{w}$ as $t \to 0$ and use the doubly parameterized homotopy

$$H(\mathbf{x}, \mathbf{z}, (1-s)\lambda^{(1)} + s\lambda^{(2)}, t) = \mathbf{0}. \tag{8}$$

This is of the same form as (4), with variables $\mathbf{v} = (\mathbf{x}, \mathbf{z})$. The witness point generator tracks all solution paths, so we are assured that the cluster is complete. The existence of a one-dimensional path to track, is known to be true by the method of constructing of the witness point set, and because the slicing coefficients, $\lambda^{(1)}$ and $\lambda^{(2)}$ are chosen generically.

4. The Algorithm

Before presenting the new algorithm for tracking singular paths, it is useful to sketch out existing predictor-corrector algorithms for tracking nonsingular paths. Our singular path tracker builds on the same framework, but we replace both the predictor and the corrector with new routines that work for singular paths.

4.1. Nonsingular Tracking Algorithm

A generic path-tracking algorithm proceeds as follows [2], (see also [1, 8]). In our homotopies, we may assume that the path parameter, s, is strictly increasing, that is, the path has no turning points.

- **Given:** System of full-rank equations, $g(\mathbf{v}, s) = \mathbf{0}$, initial point $\mathbf{v}_0$ at $s_0 = 0$ such that $g(\mathbf{v}_0, 0) \approx \mathbf{0}$, and initial step length h.
- **Find:** Sequence of points $(\mathbf{v}_i, s_i)$, $i = 1, 2, \ldots$, along the path such that $g(\mathbf{v}_i, s_i) \approx \mathbf{0}$, $s_{i+1} > s_i$, terminating with $s_n = 1$. Return a high-accuracy estimate of $\mathbf{v}_n$.
- **Procedure:**
  - **Loop:** For $i = 1, 2, \ldots$
    1. **Predict:** Predict solution $(\mathbf{u}, s')$ such that $||(\mathbf{u}, s') - (\mathbf{v}_{i-1}, s_{i-1})|| \approx h$ with $s' > s_{i+1}$.
    2. **Correct:** In the vicinity of $(\mathbf{u}, s')$, attempt to find a corrected solution $(\mathbf{w}, s'')$ such that $g(\mathbf{w}, s'') \approx \mathbf{0}$.

3. **Update:** If correction step was successful, update $(\mathbf{v}_i, s_i) = (\mathbf{w}, s'')$. Increment i.
4. **Adjust:** Adjust the step length h.

– **Terminate:** Terminate when $s_i = 1$.

– **Refine endpoint:** At $s_i = 1$, correct $\mathcal{C}_i$ to high accuracy.

There are many possible choices for the implementation of each step. Some useful choices are as follows.

- The simplest predictor is just $\mathbf{u} = \mathbf{v}_{i-1}$, but it is much better to use a prediction along the tangent direction as $\mathbf{u} = \mathbf{v}_i - \alpha(\partial g/\partial \mathbf{v})^{-1}(\partial g/\partial s)$, where α is calculated to give the desired step length. Higher-order predictions can also be used.
- The step length can be measured as a weighted two-norm or simply as $s' - s_{i-1}$, in which case $s' = s_{i-1} + h$.
- A common corrector strategy is to hold s constant, that is, $s'' = s'$, and compute $\mathbf{w}$ by Newton's method, allowing a fixed number of iterations. The correction is deemed successful if Newton's method converges within a prespecified path-tracking tolerance within the allowed number of iterations.
- A good step length adjustment strategy is to cut the step length in half on failure of the corrector and to double it if several successive corrections at the current step size have been successful.
- Near the end of the path-tracking interval, the step length is adjusted to land exactly on $s = 1$.

By keeping the number of iterations in the corrector small (no larger than three) and the path tracking tight, all intermediate points are kept close to the exact path, minimizing any chance that a solution will jump tracks. However, to save computation time, the path tracking tolerance is generally looser than that used in the final refinement at $s = 1$.

4.2. Singular Tracking Algorithm

When the path to be tracked is singular, Newton's method cannot be used effectively for correction. In the formulation of §2, this can be overcome by considering the cluster of nonsingular points that approach the singular path as $t \to 0$.

For our new corrector, we turn to methods that have been proposed for the accurate estimation of a singular endpoint, known as the "endgame." The simplest approach is to detect a cluster of paths approaching each other as the paths are tracked as close as possible to $t = 0$. In [10], it was shown that the centroid of such a cluster is an analytic function of t, which can be extrapolated to $t = 0$ to get an accurate approximation of the solution. More advanced "endgames" are proposed in [4, 11, 12, 22]. All of these depend, directly or indirectly, on a breakup of the ν roots into subsets of cardinality $c_1, \dots, c_k$ with $c_1 + \cdots + c_k = \nu$, each subset forming a cycle (see [12]). For each cycle, there exists a fractional power series

$$\mathbf{v}(t) = \mathbf{v}_0 + \mathbf{a}_1 t^{1/c_i} + \mathbf{a}_2 t^{2/c_i} + \dots \tag{9}$$

which can be used for accurate estimation of $\mathbf{v}(0)$.

In [12], the notion of an "endgame operating range" was introduced. This simply recognizes that on the one hand, the fractional power series (9), is convergent only inside a certain radius, ρ_1, while on the other hand, ill-conditioning prevents accurate numerical accuracy inside a second radius, ρ_0, dependent on the precision of arithmetic used. If the precision is sufficient so that $\rho_0 < \rho_1$, the solution can be sampled in the annular region $\rho_0 < |t| < \rho_1$ to estimate the endpoint as the constant term in the power series.

In the singular path tracking problem, the endgame operating range changes as s advances along the path. Our algorithm attempts to move the cluster of points $\mathcal{C}(s,t)$ forward in s, adjusting t at every step to stay within the endgame operating range. For intermediate values of s, the endgame can use loose tolerances—just enough to verify that t is within range. Upon reaching $s = 1$, the endgame is applied a final time with tight tolerance to compute an accurate endpoint.

We begin by specifying the singular path tracker at a high level, similar to above, and then describe the basic options for implementing the predictor and corrector steps for the singular case. For a cluster of points $\mathcal{C} = \{\mathbf{v}_1, \dots, \mathbf{v}_n\}$, let $g(\mathcal{C}, s, t) = \mathbf{0}$ be an abbreviation for $g(\mathbf{v}_i, s, t) = \mathbf{0}$, $i = 1, \dots, n$. For simplicity, the algorithm to follow assumes that step length is measured by the change in s rather than by some more general notion of arc length.

- **Given:** System of equations, $g(\mathbf{v}, s, t) = \mathbf{0}$, initial cluster $\mathcal{C}_0$, such that $g(\mathcal{C}_0, 0, t_0) \approx \mathbf{0}$, meeting conditions (1,2,3) in §2. Also, initial step length h.
- **Find:** Sequence of clusters $(\mathcal{C}_i, s_i, t_i)$, $i = 1, 2, \dots$, along the path such that $g(\mathcal{C}_i, s_i, t_i) \approx \mathbf{0}$ meeting the same cluster conditions, with $s_{i+1} > s_i$, terminating with $s_n = 1$. Return the final cluster and a high-accuracy estimate of the path point at $s = 1$.
- **Procedure:**
  - **Loop:** For $i = 1, 2, \dots$
    1. **Predict:** Predict cluster $(\mathcal{U}, s', t')$ with $s' = s_{i-1} + h$ and $t' = t_{i-1}$.
    2. **Correct:** In the vicinity of $\mathcal{U}$, attempt to find a corrected cluster $\mathcal{W}$ such that $g(\mathcal{W}, s', t') \approx \mathbf{0}$.
    3. **Recondition:** If correction is successful, play a singular endgame in t' to verify the cluster conditions and compute an approximation to the point $\lim_{t' \to 0} \mathcal{W}(s', t')$, where $g(\mathcal{W}(s', t'), s', t') = \mathbf{0}$. If the endgame is successful, then
       * **Adjust t:** Pick a new t_i in the endgame operating zone.

  - $*$ **Update:** Set $s_i = s'$ and generate the corresponding cluster $\mathcal{C}_i$. Increment i.

4. **Adjust h:** Adjust the step length h.

– **Terminate:** Terminate when $s_i = 1$.

– **Refine endpoint:** Play the endgame at $s = 1$ to compute the final path point to high accuracy.

Note that this algorithm assumes that the initial cluster has already been properly reconditioned. If not, the main loop should be preceded by an initial reconditioning.

In contrast to the nonsingular path tracker, this time the correction step does not attempt to compute a point on the path. Instead, it serves to compute the nonsingular cluster of points for a nonzero $t = t'$ after a step forward in s. The endgame in the reconditioning step now plays the role formerly played by the corrector: it computes the path point. In doing so successfully, it verifies that the path is being followed to sufficient precision. It also builds a local model of the solution cluster's dependence on t, so that t_i can be safely placed in the endgame operating range.

Since the paths of the cluster points are all nonsingular for small enough nonzero t, prediction and correction can proceed as in the nonsingular path tracker. In particular, we may use a first-order predictor and Newton's method for correction. For added protection against path crossing in the predictor-corrector step, one could check that the distances between the corrected cluster points $\mathcal{W}$ is greater than their Newton residuals. Adjustment of the step length proceeds as in the nonsingular tracker, except that a step is now successful only if both the corrector and the reconditioner succeed.

A simple version of reconditioning is to monitor the condition number while tracking the cluster paths versus t with s held constant. Along these paths, determine when the maximal condition number of the points in the cluster hits a predetermined value. The condition number can be estimated either from the Jacobian matrix of partial derivatives or as the inverse of the distance between nearest points in the cluster. Note that if the initial condition number is already past the mark, t must be increased. If it is far too high, the correction step may fail, and the step length will then be decreased.

A more sophisticated version of reconditioning is to use a fit to the fractional power series (9), to estimate the path point. By monitoring the estimate as t is decreased, one may verify that the sequence is converging accurately. This is safer than the condition number criterion, which can fail if the specified condition number lower than that encountered at the outer edge of the endgame operating range (the radius of convergence of the power series). Furthermore, detection of convergence at larger t, where the condition number is smaller, will allow the procedure to advance more quickly.

The final refinement may also take advantage of the fractional power series to compute a high-order estimate of the path point. A simple version, though, is to track t towards zero until the cluster radius is smaller than a specified tolerance, then take the centroid of the cluster as the estimate.

These comments notwithstanding, our current level of implementation is very basic. In the numerical experiments to follow, we use the condition number criterion for reconditioning. We also only recondition at the end ($s = 1$) to prepare the cluster for further tracking. We used 10^5 as a practical value for double-precision arithmetic, as it still leaves about 8 digits of accuracy in the cluster points.

5. A Numerical Experiment

The algorithms outlined above have been implemented with the aid of the path following routines in PHCpack [23]. Recently the package has been upgraded with a module (see [20] for an overview) to numerically decompose solution sets of polynomial systems into irreducible components.

We tested the implementation on a rational normal curve of degree four, of multiplicity two:

$$\left\{ \begin{array}{r} x_1^2 - x_2 = 0 \\ x_2^2 - x_3 = 0 \\ (x_3 - x_4)^2 = 0 \end{array} \right. \tag{10}$$

We can read off the solution set: $(x_1, x_1^2, x_1^4, x_1^4)$, for any $x_1 \in \mathbb{C}$. The system is underdetermined: three equations in four unknowns. We use the following embedding of the system in (10):

$$\left\{ \begin{array}{r} x_1^2 - x_2 + \gamma_{1,1} z_1 = 0 \\ x_2^2 - x_3 + \gamma_{2,1} z_1 = 0 \\ (x_3 - x_4)^2 + \gamma_{3,1} z_1 = 0 \\ z_1 - t = 0 \\ z_1 + \lambda_{1,0} + \lambda_{1,1} x_1 + \lambda_{1,2} x_2 + \lambda_{1,3} x_3 + \lambda_{1,4} x_4 = 0 \end{array} \right. \tag{11}$$

where the γ and λ constants are randomly generated complex numbers. The variable z_1 is an added "slack" variable. Clearly, for $z_1 = 0$ and $t = 0$, we obtain generic points on the curve. In particular, we have four clusters of two points each. The last equation is a generic slice, so that the limits of the solution points as $t \to 0$ are witness points for the set.

For the purposes of this numerical test, we use the interpolation procedure of [16] to determine the degree of the space curve. Sampling is accomplished by a sequence of homotopies to randomly generated slices, $\lambda^{(1)}, \lambda^{(2)}, \ldots, \lambda^{(n)}$, each homotopy having the form of (8). The samples are projected onto a randomly-oriented plane, $\mathbb{C}^2$, and fit with polynomials ranging in degree from 1 to 4. Extra samples serve as independent test points to see when the interpolant fits the entire component.

In final refinement, we use 64 decimal places. With this precision, the magnitude of the cluster radius ranges between 10^{-34} and 10^{-33}. We did not recondition except at the endpoints, so t was held constant during each leg of the homotopy. However, the condition numbers of the paths were monitored: they ranged between 10^5 and 10^6. The calculations are summarized in Table 1. At degree four, the interpolant fits the test points to full precision.

degree of the interpolator	number of sampled clusters	magnitude of condition number	magnitude of residual at test points
1	3	3	–6
2	7	8	–10
3	12	13	–20
4	16	21	$-\infty$

Table 1: Summary of incremental interpolation to determine the degree of a space curve. For every degree of the interpolator, we list the total number of sampled clusters, the magnitude of the condition number (3 stands for 10^3) of the interpolation problem and the magnitude of the residual at some test points.

In Table 1 we observe a steady worsening of the condition of the interpolation problem as the degree rises. For degrees higher than four, one would need to sharpen the clusters with more than 64 decimal places as working precision or implement a more sophisticated endgame. Alternatively, instead of ordinary linear projections, one can (if the linear span of the component permits) project from a point on the component. These central projections [17] reduce the degree of the interpolating polynomial and thus improve the numerical conditioning of the problem.

Complementary to using central projections is the application of extrapolation techniques to improve the centroid of the cluster. For this example, extrapolation is used on the fractional power series (9) with cycle number $c = 2$ at values 10^{-20}, 10^{-32}, and 10^{-44} for t in the homotopy (11). The working precision is 64 decimal places. This second order extrapolator achieves a cluster radius of magnitude in the range between 10^{-45} and 10^{-44}. In comparison, if we approximate the centroid of the cluster just by taking the average of the solution vectors at $z_1 = 10^{-72}$, we obtain a radius in the range between 10^{-34} and 10^{-33}.

6. Conclusions

We have proposed a tracking algorithm for singular paths. The system of equations defining the singular path are embedded in a perturbed system having a cluster of nearby nonsingular paths. These can be handled effectively with nonsingular path tracking techniques. A singular endgame can then be applied to the cluster to estimate points on the original singular path. A bare-bones version of the tracker is shown to be effective on a simple, fourth-degree curve having multiplicity two.

Section 4 indicates several possible enhancements to the safety and efficiency of the algorithm. Chief among these is to use a higher-order singular endgame instead of just the centroid of the cluster, which is correct only to first order in t. Higher-order approximations will decrease the error propagation caused by the multiplicity and thus increase the numerical stability. To get a point on a singular path having cycle number c accurate up to d decimal places, a rough rule of thumb is that with first order approximation we need to take t as low as 10^{-cd}, whereas with extrapolation of order c, we can stop t at 10^{-d}. This behavior is confirmed in the experiment in Section 5.

This algorithm, although ripe for further improvement, serves to show feasibility for extending our techniques for irreducible decomposition to components of multiplicity greater than one. In this sense, the irreducible decomposition algorithm is now complete, although many improvements are still possible.

References

[1] E.L. Allgower and K. Georg. *Numerical Continuation Methods, an Introduction*, volume 13 of *Springer Ser. in Comput. Math.* Springer–Verlag, 1990.

[2] E.L. Allgower and K. Georg. Numerical Path Following. In *Techniques of Scientific Computing (Part 2)*, edited by P.G. Ciarlet and J.L. Lions, volume 5 of *Handbook of Numerical Analysis*, pages 3–203. North-Holland, 1997.

[3] A. Galligo and D. Rupprecht. Semi-numerical determination of irreducible branches of a reduced space curve. In *Proceedings of the 2001 International Symposium on Symbolic and Algebraic Computation*, edited by B. Mourrain, pages 137–142, ACM 2001.

[4] B. Huber and J. Verschelde. Polyhedral end games for polynomial continuation. *Numerical Algorithms* 18(1):91–108, 1998.

[5] G. Lecerf. *Une alternative aux méthodes de réécriture pour la résolution des systèmes algébriques.* PhD Thesis, École Polytechnique, France, 2001.

[6] G. Lecerf. Quadratic Newton iteration for systems with multiplicity. To appear in *J. FoCM.*

[7] T.Y. Li and X. Wang. Nonlinear homotopies for solving deficient polynomial systems with parameters. *SIAM J. Numer. Anal.* 29(4):1104–1118, 1992.

[8] A. Morgan. *Solving polynomial systems using continuation for engineering and scientific problems.* Prentice-Hall, Englewood Cliffs, N.J., 1987.

[9] A.P. Morgan and A.J. Sommese. Coefficient-parameter homotopy continuation. *Appl. Math. Comput.* 29(2):123–160, 1989. Errata: *Appl. Math. Comput.* 51:207(1992).

[10] A.P. Morgan, A.J. Sommese, and C.W. Wampler. Computing singular solutions to nonlinear analytic systems. *Numer. Math.* 58(7):669–684, 1991.

[11] A.P. Morgan, A.J. Sommese, and C.W. Wampler. Computing singular solutions to polynomial systems. *Adv. Appl. Math.* 13(3):305–327, 1992.

[12] A.P. Morgan, A.J. Sommese, and C.W. Wampler. A power series method for computing singular solutions to nonlinear analytic systems. *Numer. Math.* 63(3):391–409, 1992.

[13] T. Ojika. Modified deflation algorithm for the solution of singular problems. I. A system of nonlinear algebraic equations. *J. Math. Anal. Appl.* 123:199–221, 1987.

[14] T. Ojika, S. Watanabe, and T. Mitsui. Deflation algorithm for the multiple roots of a system of nonlinear equations. *J. Math. Anal. Appl.* 96:463–479, 1983.

[15] 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.

[16] 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.

[17] 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.

[18] 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. Proceedings of a NATO Conference, February 25 - March 1, 2001, Eilat, Israel, pages 297–315, Kluwer Academic Publishers.

[19] A.J. Sommese, J. Verschelde, and C.W. Wampler. Symmetric functions applied to decomposing solution sets of polynomial systems. Submitted for Publication, 2001.

[20] A.J. Sommese, J. Verschelde, and C.W. Wampler. Numerical irreducible decomposition using PHCpack. Submitted for Publication, 2001.

[21] 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, pages 749–763, 1996. Proceedings of the AMS-SIAM Summer Seminar in Applied Mathematics, Park City, Utah, July 17-August 11, 1995, Park City, Utah.

[22] M. Sosonkina, L.T. Watson, and D.E. Stewart. Note on the end game in homotopy zero curve tracking. *ACM Trans. Math. Software* 22(3):281–287, 1996.

[23] J. Verschelde. Algorithm 795: PHCpack: A general-purpose solver for polynomial systems by homotopy continuation. *ACM Trans. Math. Software* 25(2):251–276, 1999. Software available at `http://www.math.uic.edu/~jan`.