Finding All Isolated Solutions To Polynomial Systems Using Hompack

Alexander Morgan
Andrew J. Sommese
Layne T. Watson

TR 87-28

FINDING ALL ISOLATED SOLUTIONS TO POLYNOMIAL SYSTEMS USING HOMPACK

by

Alexander Morgan
Mathematics Department
General Motors Research Laboratories
Warren, MI 48090-9057

Andrew J. Sommese †
Mathematics Department
University of Notre Dame
Notre Dame, IN 46556

Layne T. Watson *
Department of Computer Science
Virginia Polytechnic Institute & State University
Blacksburg, VA 24061

Address for correspondence and proofs:

Alexander Morgan

Mathematics Department

General Motors Research Laboratories

Warren, MI 48090-9057

† Partially supported by NSF grant DMS 84-20315.
* Partially supported by AFOSR Grant 85-0250

FINDING ALL ISOLATED SOLUTIONS TO POLYNOMIAL SYSTEMS USING HOMPACK

by

Alexander Morgan, Andrew Sommese, and Layne Watson

ABSTRACT

Although the theory of polynomial continuation has been established for over a decade (following the work of Garcia, Zangwill, and Drexler), it is difficult to solve polynomial systems using continuation in practice. Divergent paths (solutions at infinity), singular solutions, and extreme scaling of coefficients can create catastrophic numerical problems. Further, the large number of paths that typically arise can be discouraging.

In this paper we summarize polynomial-solving homotopy continuation and report on the performance of three standard path-tracking algorithms (as implemented in HOMPACK) in solving three physical problems of varying degrees of difficulty. Our purpose is to provide useful information on solving polynomial systems, including specific guidelines for homotopy construction and parameter settings. The $m-$homogeneous strategy for constructing polynomial homotopies is outlined, along with more traditional approaches. Computational comparisons are included to illustrate and contrast the major HOMPACK options. The conclusions summarize our numerical experience and discuss areas for future research.

1. Introduction.

Let $f(z) = 0$ denote a system of n polynomial equations in n unknowns with complex number coefficients. Generally, such a system has many (complex) solutions. We use homotopy continuation to find all the geometrically isolated solutions of $f(z) = 0$ as follows. We embed f in a system of n polynomial equations in $n+1$ unknowns where this new system includes the variables of f and a new variable, the homotopy parameter. For one value of the homotopy parameter, the new system can be satisfactorily solved, and for another it is identically equal to f. The continuation process attempts to solve $f(z) = 0$ by evolving or "continuing" the full set of known solutions into the full set of solutions to $f(z) = 0$.

We denote the homotopy system by $h(z,t) = 0$ where $h(z,1) = f(z)$ for all z and we know the solutions to $h(z,0) = 0$. Thus we view the homotopy parameter, t, as varying between 0 and 1. We must continue the solutions of $h(z,0) = 0$ into those of $h(z,1) = 0$. We assume sufficient conditions so that $h^{-1}(0)$ consists of smooth paths, and the continuation of solutions becomes a process of "path tracking."

Many issues arise in attempting to implement this concept into a reliable and fast algorithm for computing all solutions to polynomial systems. Basically, there are two steps:

(1) Define the homotopy, $h(z,t)$.

(2) Choose a numerical method for tracking the paths defined by $h(z,t) = 0$.

Step (1) is guided by results from algebraic geometry, while (2) is based on methods for the numerical solution of ordinary differential equations and local methods for the solution of nonlinear systems. Both parts are nontrivial and important.

When homotopy continuation is used to compute the full solution list for a polynomial system, the continuation is carried out in complex projective space in a complex analytic context (as discussed in Section 2). As a consequence, the homotopy paths have a very special structure. For example, $dt/ds > 0$ on paths where s denotes arc length; thus, paths are strictly increasing in t as a function of arc length. On the other hand, singular solutions to $f(z) = 0$ always reduce the real-rank of the Jacobian matrix by a multiple of 2. Therefore, the end game of the continuation process cannot be made nonsingular by generically embedding f in h, and "rank $n-1$ local refinement algorithms" are not applicable. Most sophisticated path-tracking algorithms allow paths to "turn back in t," so these algorithms are more general than is needed for polynomial-solving homotopy continuation, and at the same time they generally do not respond well to highly rank deficient endpoint singularities. It follows that these path trackers are not necessarily well suited to the paths generated by polynomial continuation. This paper studies the behavior of three standard path-tracking algorithms on the special paths generated by polynomial homotopies. Included is a brief summary of the theory of constructing polynomial homotopies (Section 2)

and a brief description of the three path trackers, as implemented in HOMPACK (Section 3). Section 4 describes three test problems and the results of the associated numerical experiments. Section 5 is a summary with conclusions and suggestions for future work.

We shall call polynomial-solving homotopy continuation "polynomial continuation." Any numerical method to find the full set of geometrically isolated solutions to a polynomial system using homotopy continuation is "polynomial continuation."

The POLSYS routine in HOMPACK evokes a traditional polynomial homotopy. POLSYS uses a convenient "tableau" format for inputing the system parameters, along with options to scale (via subroutine SCLGNP) and to apply the projective transformation. In this paper we consider a broader range of options than is available in POLSYS, the most important being the $m-$homogeneous approach to homotopy construction (described in section 2). Also, we consider three path-tracking methods, while POLSYS offers only one. We include in our numerical experiments tests of codes equivalent to POLSYS. See the beginning of section 3 for additional comments on POLSYS.

The idea of polynomial continuation was first suggested in 1977 in papers by Garcia and Zangwill [10] and Drexler [5]. The following year Chow, Mallet-Paret, and York [4] presented an important refinement of the Garcia and Zangwill work. A number of papers have followed: Brunovský and Meravý [2], Drexler [6], Garcia and Li [9], Garcia and Zangwill [11–14], Li, Sauer, and Yorke [19], Morgan [21–24], Morgan and Sommese [25, 26], and Wright [WR].

There have also been some significant physical applications. See Richter and De Carlo [28], Meintjes and Morgan [20], Morgan [24], Morgan and Sarraga [27], Safonov [30], and Tsai and Morgan [32].

Section 2. Constructing Homotopies For Polynomial Systems.

In this section we state the three theorems on which the $m-$homogeneous method of constructing polynomial homotopies is based. (These theorems are proven in [25] and [26].) Then we outline the steps in the homotopy construction process.

Let $f(z) = 0$ be a polynomial system of n equations in n unknowns with complex number coefficients, where $z \in C^n$. First, we need the definitions of a *geometrically isolated* solution and the *multiplicity* of a solution.

A solution to a polynomial system is called *geometrically isolated* (or simply isolated) if there is a ball around the solution that contains no other solution. A solution that is not geometrically isolated is singular, but an isolated solution can be singular also.

Let z^* be a geometrically isolated solution to the polynomial system $f(z) = 0$. Let U be a closed ball about z^* containing no other solution. We can perturb f by adding arbitrarily small complex numbers to each coefficient of f (including the zero coefficients) in such a way that the perturbed system has only nonsingular solutions. (For example, by the Transversality Theorem [16], it suffices to add small random numbers to the constant terms.) For all sufficiently small coefficient perturbations, the perturbed systems have a constant number, m, of solutions in U. This m is (by definition) the *multiplicity* of z^*. See, for example, [7], [8], [15], [31], [37]. We note that a solution z^* to $f(z) = 0$ has multiplicity greater than one exactly when it is singular; that is, when the Jacobian matrix $df(z^*)$ is singular.

The algebraically proper context for generating the full solution list of a polynomial system is complex projective space rather than real or complex Euclidean space. This is because the structure of the solution set to $f(z) = 0$ is generic in projective space. (See, for example, Bezout's theorem in the above algebraic geometry references.) Homotopy continuation methods for generating the full solution list to $f(z) = 0$ have always implicitly acknowledged this by being formulated in complex Euclidean space and allowing paths to diverge to infinity. It is more numerically stable, however, to acknowledge projective space directly. We therefore follow the classical approach from algebraic geometry of homogenizing f and establishing our continuation process in projective space. In many cases it is advantageous to homogenize f so that it has an $m-homogeneous$ structure (defined in this Section below). Then we view the solutions to $f(z) = 0$ as being in a Cartesian product of projective spaces. We will present our discussion in this generality. Since 1-homogeneous systems are merely homogeneous systems, the $m-$homogeneous approach includes all polynomial systems and does not limit us to special cases.

Complex projective space, P^k, consists of the lines through the origin in C^{k+1}, denoted $[(z_0, ..., z_k)]$ where $(z_0, ..., z_k) \in C^{k+1} - \{0\}$; that is, $[(z_0, ..., z_k)]$ is the line through the origin that contains $(z_0, ..., z_k)$. It is natural to view P^k as a disjoint union of points $[(z_0, ..., z_k)]$ with $z_0 \neq 0$ (identified with Euclidean space via $[(z_0, ..., z_k)] \rightarrow (z_1/z_0, ..., z_k/z_0)$) and the "points at infinity," the $[(z_0, ..., z_n)]$ with $z_0 = 0$.

We partition the variables $\{z_1, ..., z_n\}$ into m nonempty collections. It will be notationally simpler here if we rename the variables with double subscripts. Thus, let

$$\{z_1, \ldots, z_n\} = \cup_{j=1}^{m} \{z_{1,j}, ..., z_{k_j,j}\},$$

where $\sum_{j=1}^{m} k_j = n$. Now choose homogeneous variables $z_{0,j}$ for $j = 1$ to m and define

$$Z_j = \{z_{0,j}, z_{1,j}, ..., z_{k_j,j}\}$$

for $j = 1$ to m. Then evoke the substitution $z_{i,j} \leftarrow z_{i,j}/z_{0,j}$ for $i = 1$ to k_j and $j = 1$ to m, generating a system $f' = 0$ of n equations in $n + m$ unknowns (after we clear the denominators of powers of the $z_{0,j}$). Now $f' = 0$ naturally has solutions in

$$P \equiv P^{k_1} \times P^{k_2} \times \ldots \times P^{k_m}.$$

(See [25].) We say f' is *m-homogeneous* because the variables are partitioned into m collections, $Z_1, ..., Z_m$, so that f' is homogeneous as a system in the variables of any one of the collections. We let $d_{j,l}$ denote the j^{th} degree of the l^{th} polynomial; that is, with all variables held fixed except those in Z_j, f'_l has homogeneous degree $d_{j,l}$. Note that "1-homogeneous" is the same as "homogeneous," so theorems about m-homogeneous polynomial systems apply essentially to all polynomial systems. We say that polynomial f_l has *type* $= (d_{1,l}, \ldots, d_{m,l})$.

The *Bezout number*, d, of an $m-$homogeneous polynomial system is defined to be the coefficient of $\prod_{j=1}^{m} \alpha_j^{k_j}$ in the product

$$D = \prod_{l=1}^{n} \sum_{j=1}^{m} d_{j,l}\alpha_j. \tag{1}$$

The significance of the Bezout number is that it is an upper bound on the number of homotopy continuation paths we will track in the space $P \times [0, 1]$ (Theorem 1, below). The smaller d is, the better. Frequently, the m-homogenization of f for $m > 1$ has a (much) smaller Bezout number than the 1-homogenization. If $m = 1$, then $d = d_1 \cdots d_n$, the total degree of f. This is the "traditional" number of paths to track.

While it is natural to extend our context from Euclidean to projective space, it is often a nuisance to acknowledge this change of context with precise changes of notation. Thus, when we start with a system of polynomials $f(z) = 0$ with $z \in C^n$ and generate the $m-$homogenization $f'(z') = 0$ with $z' \in P$, we generally will abuse the notation by dropping the primes. Usually, our systems arise naturally as non-homogeneous systems of n equations in n unknowns, and it will be implicit that an appropriate m-homogenization will be carried out so that "z" is now in P and the systems under discussion are the m-homogeneous forms of the original systems.

Define a homotopy

$$h(z,t) = (1-t)\gamma g(z) + tf(z), \tag{2}$$

where g is an $m-$homogeneous system of n polynomials in n variables, and γ is a randomly chosen complex number. Let g be chosen so that its m-homogeneous structure matches that of f; that is, $Z_1, \ldots, Z_m$ are specified and the $d_{j,l}$ for g are exactly the same as those for f. Naturally, many such g will exist. We can always choose $m = 1$ and g diagonal (e.g., $g_j = p_j z_{l,1}^{d_{1,l}} - q_j z_{0,1}^{d_{1,l}}$), but it is important to note that in practice we can often do better. When specific examples of systems f and g are presented below, we give the non-homogeneous forms with the understanding that the appropriate $m-$homogenization is to be applied, as noted above.

Let $S \subseteq P$ be a set of common solutions of $f(z) = 0$ and $g(z) = 0$. For each $s \in S$, we require that the following conditions hold. For $s \in S$ let K denote the full connected component of solutions of $g(z) = 0$ with $s \in K$.

If s is a geometrically isolated solution of $g(z) = 0$ (i.e., $K = \{s\}$), we assume that:

- s is a geometrically isolated solution of $f(z) = 0$, and
- the multiplicity of s as a solution of $g(z) = 0$ is less than or equal to the multiplicity of s as a solution of $f(z) = 0$.

If s is not a geometrically isolated solution of $g(z) = 0$, then we assume that:

- K is contained in S,
- K is the full solution component of $f(z) = 0$ containing s,
- K is a smooth manifold (see [16]), and
- at each point $z^0 \in K$ the rank of $dg(z^0)$ is the codimension of K (that is, $n-$(the dimension of K)).

Let Σ denote the solutions of $g(z) = 0$ in $P - S$. Then the following theorem holds.

Theorem 1. Assume the points in Σ are nonsingular solutions of $g(z) = 0$. For any positive r and for all but a finite number of angles, θ, if $\gamma = re^{i\theta}$, then $h^{-1}(0) \cap ((P - S) \times [0,1))$ consists of smooth paths and every geometrically isolated solution of $f(z) = 0$ not in S has a path in $(P - S) \times [0,1)$ converging to it. In fact, if m_0 is the multiplicity of a geometrically isolated solution, z^0, that is not in S, then z^0 has exactly m_0 paths converging to it. Further, the paths are strictly increasing in t, and $dt/ds > 0$, where s denotes arc length.

This theorem is given in [26] as Theorem 1 (including the remarks after the statement of the theorem).

Notes:

1. Naturally, S may be empty, in which case the only restriction on the solutions of $g(z) = 0$ is that there be exactly d of them. (This implies they are nonsingular.)

2. The theorem can be generalized to allow the K in S not to be smooth manifolds. However, computing the multiplicity of higher-dimensional non-manifold solution components is much more difficult than for geometrically isolated solutions and higher-dimensional manifold solution components.

3. Applications in which one might allow S to contain infinite solution components arise naturally. For example, S might be a disjoint unions of lines, in which case the rank-codimension requirement is that the rank of $dg(z^0)$ be $n-1$ for each z^0 on each of the lines. See the manipulator example in Section 4.

Here is a simple example to illustrate the case when the multiplicities of a common solution are different for $f(z) = 0$ and $g(z) = 0$. Let

$$g(z) = (z-1)^2(z-2)(z-3)(z-4)$$

and

$$f(z) = (z-1)^4(z-5).$$

Take $S = \{1\}$. We track paths beginning at $z = 2$, 3, and 4. One of these will converge to $z = 5$, the other two to $z = 1$.

Let $L = (L_1, \ldots, L_m)$ with

$$L_j = \sum_{i=0}^{k_j} b_{i,j} z_{i,j} \tag{3}$$

where $b_{i,j} \neq 0$ for some i, for each j. Then we say that

$$U_L = U_{L_1} \times U_{L_2} \times \ldots \times U_{L_m},$$

is the *Euclidean coordinate patch defined by* L, where

$$U_{L_j} = \{[z] \in P^{k_j} \,|\, L_j(z) \neq 0\}$$

is the Euclidean coordinate patch on P^{k_j} defined by L_j. Note that U_L, which is an open dense submanifold of P, can be identified with

$$C^n = C^{k_1} \times C^{k_2} \times \ldots \times C^{k_m}$$

by identifying U_{L_j} with C^{k_j} via

$$[(z_0, \ldots, z_{k_j})] \to \frac{1}{L_j(z)}(z_0, \ldots, z_{i_j-1}, z_{i_j+1}, \ldots, z_{k_j}),$$

where $\beta_{i_j,j} \neq 0$ for $j = 1$ to m.

Note that if $z \in P$, then $z \in U_L$ unless $L_j(z_{0,j}, \ldots, z_{k_j,j}) = 0$ for some j. Thus z is in virtually all such coordinate patches. This also holds if we restrict our choices of the constants $b_{i,j}$ to be real numbers. (We say that "virtually all" parameters obey a stated condition if the set of parameters that do not obey the condition are contained in a lower-dimensional algebraic variety of parameter space. The result is that parameters "chosen at random" from parameter space will obey the condition "with probability one." Compare with the similar topological ideas in [3].)

The following two theorems from [26] show us how to keep the continuation process in Euclidean space, even though our basic theorem (Theorem 1) is formulated in P, a Cartesian product of projective spaces.

Theorem 2. Let U_L be a given Euclidean patch on P defined by L, as above. If the solutions of $g(z) = 0$ in $P - S$ are all in U_L, then

$$h^{-1}(0) \cap \big((P - S) \times [0, 1)\big) \subset U_L \times [0, 1),$$

except for a finite number of θ.

Theorem 3. Assume the points in Σ are nonsingular solutions of $g(z) = 0$. Then

$$\overline{h^{-1}(0) \cap \big((P - S) \times [0, 1)\big)} \subset U_L \times [0, 1]$$

for virtually all U_L and all but a finite number of θ, where the overbar indicates topological closure.

Note that $\overline{h^{-1}(0) \cap \big((P - S) \times [0, 1)\big)}$ is the set of continuation paths, including end points. It equals $h^{-1}(0) \cap \big((P - S) \times [0, 1]\big)$ if and only if $f(z) = 0$ has only a finite number of solutions in $P - S$. (Because the half open interval $[0, 1)$ is used here instead of $[0, 1]$, the parts of the higher-dimensional solution components of $h^{-1}(0)$ that are not the endpoints of continuation paths are excluded from $\overline{h^{-1}(0) \cap \big((P - S) \times [0, 1)\big)}$.)

We can make the phrase "virtually all U_L" in Theorem 3 more precise. The result holds for virtually all $(b_{i,j}) \in \prod C^{k_j+1}$ and also for virtually all $(b_{i,j}) \in \prod R^{k_j+1}$. In fact, the only L for which the result fails are those for which $T \not\subset U_L$ where T is the finite set containing both the geometrically isolated solutions of $f(z) = 0$ in $P - S$ and the limits of homotopy paths in $(P - S) \times [0, 1)$ going to infinite solution-components of $f(z) = 0$. In other words, if $L_j(z_{0,j}, \ldots, z_{k_j,j}) = 0$ for some $z \in T$ and some j, then U_L will not work. Otherwise, it will.

For computations, we need a convenient way to realize the U_L. We do this via "the projective transformation," as follows. With m-homogeneous h in the variables $z_{i,j}$ for $i = 0$ to k_j and $j = 1$ to m, we let

$$z_{0,j} = \sum_{i=1}^{k_j} \beta_{i,j} z_{i,j} \; + \; \beta_{0,j} \tag{4}$$

for $j = 1$ to m where the $\beta_{i,j}$ are constants and $\beta_{i,j} \neq 0$ for some i, for each j. (Compare equation (3).) The *projective transformation of* h (given by (4)) is the system H of n equations in the n variables $z_{i,j}$ for $i = 1$ to k_j, $j = 1$ to m where $H_j = h_j$ for $j = 1$ to n with (4) defining the $z_{0,j}$ in terms of the other variables. By Theorem 3,

$$\overline{h^{-1}(0) \cap \left((P - S) \times [0, 1)\right)}$$

(the homotopy paths, including end points) are completely represented in C^n via H, for virtually all $\beta = (\beta_{i,j})$ and all but a finite number of θ. Thus, for computations, we need not acknowledge projective space except by solving H with randomly chosen $\beta_{i,j}$ and θ. The finite solutions of $f(z) = 0$ are recovered via $z_{i,j} \leftarrow z_{i,j}/z_{0,j}$ for $i = 1$ to k_j and $j = 1$ to m. (If any $z_{0,j} = 0$, then the solution is at infinity.)

In creating a computer code to implement H, our usual procedure is to write a subroutine for h as a system of n equations in the $n+m$ variables $z_{i,j}$ but include the formulas (4) which make $z_{0,j}$ for $j = 1$ to m implicitly defined functions of the other variables. The partial derivatives of H with respect to $z_{i,j}$ for $i = 1$ to k_j, $j = 1$ to m are then generated from those of h with respect to $z_{i,j}$ for $i = 0$ to k_j, $j = 1$ to m using the chain rule. To make use of Theorem 3, the $\beta_{i,j}$ are chosen at random from C or from R.

In the 1-homogeneous case one option is to apply the projective transformation to f rather than to h, and then solve the resulting $F(z) = 0$ using a homotopy in Euclidean space (perhaps to make use of an existing homotopy code). In fact, this is recommended in [22]. Since this $F(z) = 0$ has all its geometrically isolated solutions in Euclidean space (for virtually all coefficients of the projective transformation), we are guaranteed that the resulting homotopy paths will find all geometrically isolated solutions of $f(z) = 0$ (including those at infinity). In our tests we tried both approachs (projective transformation of h and projective transformation of f) when we were studying traditional homotopies. The practical difference in most cases was slight. For simplicity, in Section 4 we report the results with traditional homotopies using the projective transformation of f only.

Let H be the projective transformation of homotopy h, where h is given by theorem 1. We now give a summary of the structure of the solution set $H^{-1}(0)$ as given by theorems 1–3. Then, we complete this section with a step-by-step description of the $m-$homogeneous solution strategy based on theorems 1–3.

First, assume that Σ is the full solutions set of $g(z) = 0$. (The start points for the homotopy will be the associated ("projectively transformed") solutions to $H(z,0) = 0$.) The set $H^{-1}(0)$ is a union of two parts: Part A (the paths) and Part B (described below). Part A consists of bounded, non-bifurcating, non-intersecting (except perhaps when $t = 1$) paths in $C^n \times [0,1]$, with start points in $C^n \times \{0\}$ and end points in $C^n \times \{1\}$. There will be one path for each point in Σ , except that some paths may come together when $t = 1$. The paths are smooth 1-real-dimensional submanifolds of $C^n \times [0,1]$ (with boundary), except that where paths come together non-manifold points may be generated. All the paths are strictly increasing in t as a function of arc length, s, and $dt/ds > 0$ on paths. Part B consists of the points in the set

$$\{(z,1)|H(z,1) = 0\}$$

not in Part A; that is, solutions to $H(z,1) = 0$ that are not the endpoints of homotopy paths. All geometrically isolated solutions are the endpoints of paths, and therefore in Part A. Part B is empty if $H(z,1) = 0$ has only a finite number of solutions. If $H(z,1) = 0$ has an infinite number of solutions, then the solution set consists of the geometrically isolated solutions and the solutions that are parts of higher-dimensional solution components. (These are algebraic varieties, in the sense of algebraic geometry, but are not necessarily smooth manifolds.) Part B consists of the pieces of the infinite solution components that are not in the set of endpoints of the paths. Each infinite solution component contains at least two points that are the endpoints of paths.

If Σ is a proper subset of the set of solutions of $g(z) = 0$, then the description of Part A given above remains valid, and we may take Part B to consist of the pieces of the infinite solution components of $H(z,1) = 0$ not the endpoints of paths that are in connected components in $H^{-1}(0)$ of points in Σ . There is a third part to the solution set of $H^{-1}(0)$, defined as follows. The common solution set S (see Theorem 1) may contain infinite solutions components, K. Then $K \times [0,1]$ is in $h^{-1}(0)$ for each K in S. (The homotopy is constant on these K, and we don't track any paths associated with them.) Part C of the solution set to $H^{-1}(0)$ consists of the union of $(K \cap U_L) \times [0,1]$ over all $K \in S$, where U_L is defined by the projective transformation, as above.

Here is a summary of the $m-$homogeneous solution process. We want to find all geometrically isolated solutions to $f(z) = 0$. The solution process described below finds all such solutions as well as at least two points on every infinite solution component. There are two steps:

Step 1: **Constructing the Homotopy.**

Stage a. Fix an m-homogeneous form for f. Generally, we choose an m-homogeneous form for f that minimizes the Bezout number, although other considerations may be relevant.

Stage b. Choose the system g so that:

- g has the same m-homogeneous form as f
- The solutions to $g(z) = 0$ are known
- The singular solutions to $g(z) = 0$ are also singular solutions to $f(z) = 0$. See Theorem 1 for conditions that must hold (i.e., restrictions on the set S.)
- If possible, $g(z) = 0$ and $f(z) = 0$ should share solutions and/or the coefficients of g should be "close to" the coefficients of f. (There are generic ways to choose g that always "work." However, these generic choices are rarely the most effective.)

Stage c. Define H to be the projective transformation of h. Choose a set S of common solutions of $g = 0$ and $f = 0$. Let Σ be the solutions of $g = 0$ not in S. (Σ and S must obey the conditions of Theorem 1.)

Step 2: **Numerical Path Tracking.** For each point in Σ , the associated path in $H^{-1}(0)$ will be tracked numerically from $t = 0$ to $t = 1$. This will yield the full list of geometrically isolated solutions to $H(z, 1) = 0$. No paths diverge to infinity, so the numerical path tracking will terminate in a finite number of steps without abandoning any path.

Section 3. Smooth Path-Tracking Algorithms.

There are many different algorithms for tracking smooth paths. The CONSOL8 code from [24] was developed specifically for polynomial continuation, and its performance is discussed in [24] and [26]. HOMPACK [35] supports three general path-tracking methods, not customized to the polynomial problem. These methods are: ordinary differential equation based, normal flow, and augmented Jacobian matrix, denoted DF, NF, and QF, respectively. These are described in detail in [35]. Here, we summarize. Although the distinction of sparse vs. dense Jacobian matrix methods is important for some applications, the polynomial applications we have in mind generate dense Jacobian matrices. We will not acknowledge the sparse techniques further here, but see [35].

The POLSYS driver provided with HOMPACK (and described in [35]) is essentially the traditional generic homotopy with the NF method, including easy-to-evoke options for scaling and using the projective transformation of f. The computational results in section 4 for the traditional homotopy and NF method can be interpreted as POLSYS results, with the exception that the CPU times for POLSYS will generally be longer (but not the WORK values, which are based on operation counts). This is because the coding of the FFUNP subroutine for POLSYS (using the convenient tableau input format) is less efficient than the customized code used in this study. (See [35] for a discussion of the POLSYS input tableau.)

We will use the notation $w = (z, t)$ in this section. It is valid to view w either as a vector of n complex numbers $(z_1, \ldots, z_n)$ and one real number (t) or $2n+1$ real numbers $(Re(z_1), Im(z_1), \ldots, Re(z_n), Im(z_n), t)$. It is standard in polynomial continuation to move from the context of $2n$ equations in $2n+1$ real unknowns to the context of n equations in n complex and one real unknown as convenient, without loss of rigor. (See, for example, [24, App.3].) In this section, we will generally adopt the convention that we are dealing with $2n$ real equations in $2n+1$ real unknowns. We will use the notation

$$x = (Re(z_1), Im(z_1), \ldots, Re(z_n), Im(z_n)),$$

so that $w = (z, t) = (x, t)$.

Ordinary differential equation based algorithm.

Let Γ denote a zero path of $H(w) = 0$. We can always parametrize Γ by arc length s. Thus $w = w(s)$ along Γ, and

$$H(w(s)) = 0$$

identically in s. Therefore

$$\frac{dH}{ds}(w(s)) = dH(w(s))\frac{dw}{ds} = 0 \tag{5}$$

and

$$\left\| \frac{dw}{ds} \right\| = 1 \tag{6}$$

identically in s, where dH denotes the $2n \times (2n+1)$ Jacobian matrix of H with respect to w and the norm in (6) is the Euclidean norm. With the initial conditions

$$t(0) = 0, \quad z(0) = z^0, \tag{7}$$

for some $z^0 \in \Sigma$ (in the notation of Theorem 1), the zero curve Γ is one of the homotopy paths defined by $H = 0$. When $t(\bar{s}) = 1$, the corresponding $z(\bar{s})$ is a zero of $F(z) \equiv H(z,1) = 0$. Thus techniques for numerically solving ordinary differential equations can be brought to bear on the problem of tracking Γ .

Typical ordinary differential equation software requires dw/ds explicitly, but (5–6) only implicitly define dw/ds. The derivative dw/ds, which is a unit tangent vector to the zero curve Γ , can be calculated by finding the one-dimensional kernel of $dH(w(s))$, which has full rank according to theory, except perhaps when $t = 1$. Once the kernel has been calculated, the derivative dw/ds is uniquely determined by (6) and continuity. Complete details for solving the initial value problem (5–7) and obtaining $z(\bar{s})$ are in [33] and [36]. A discussion of the kernel computation is given in [35] and [34], along with a summary flow chart for the ODE-based algorithm.

Normal flow algorithm.

Consider the family of homotopy continuation systems $H(w) = a$, for $a \in C^n$. As a varies, each of the corresponding sets of homotopy curves varies. The resulting family of sets of zero curves is known as the Davidenko flow. The normal flow algorithm is so called because the corrector iterates converge to the zero curves along the flows normal to the Davidenko flow.

The normal flow algorithm has four phases: prediction, correction, step size estimation, and end game (computation of the solution at $t = 1$). Let Γ be a continuation curve. For the prediction phase, assume that several points, $P^{(1)}$ and $P^{(2)}$, on Γ have been found, with corresponding tangent vectors, $T^{(1)}$ and $T^{(2)}$ respectively. Let η be an estimate of the optimal step size (in arc length) to take along Γ . The prediction of the next point on Γ is

$$Z^{(0)} = p(s_2 + \eta), \tag{8}$$

where $p(s)$ is the Hermite cubic interpolating $w(s)$ at s_1 and s_2. Precisely,

$$p(s_1) = P^{(1)}, \quad (\frac{dp}{ds})(s_1) = T^{(1)},$$
$$p(s_2) = P^{(2)}, \quad (\frac{dp}{ds})(s_2) = T^{(2)},$$

and each component of $p(s)$ is a polynomial in s of degree less than or equal to 3.

Starting at the predicted point $Z^{(0)}$, the corrector iteration is

$$Z^{(k+1)} = Z^{(k)} - \left[dH(Z^{(k)})\right]^{\dagger} H(Z^{(k)}), \qquad k = 0, 1, \ldots \tag{9}$$

where $\left[dH(Z^{(k)})\right]^{\dagger}$ is the Moore-Penrose pseudoinverse of the $2n \times (2n+1)$ Jacobian matrix dH.

A corrector step ΔZ is the unique minimum norm solution of the equation

$$dH\ \Delta Z = -H.$$

Fortunately ΔZ can be calculated at the same time as the kernel of dH, and with just a little more work.

When the iteration (9) converges, the final iterate $Z^{(k+1)}$ is accepted as the next point on Γ and the tangent vector to the integral curve through $Z^{(k)}$ is used for the tangent. This saves a Jacobian matrix evaluation and factorization at $Z^{(k+1)}$. The step size estimation attempts to balance progress along Γ with the effort expended on the iteration (9).

The end game is evokes when $t \geq 1$. Cubic interpolation is used (as in prediction) to obtain an estimate of a point with $t = 1$, and this estimate is refined using a variant of Newton's method.

Augmented Jacobian matrix algorithm. The augmented Jacobian matrix algorithm also has the four phases: prediction, correction, step size estimation, and end game. The algorithm is based on Rheinboldt [29], but with some significant differences: (1) a Hermite cubic rather than a linear predictor is used; (2) a tangent vector rather than a standard basis vector is used to augment the Jacobian matrix of the homotopy map; (3) updated QR factorizations and quasi-Newton updates are used rather than Newton's method; (4) different step size control, necessitated by the use of quasi-Newton iterations, is used; (5) a different scheme for locating the target point at $t = 1$ is used. See [1] for a fuller discussion.

The prediction phase is exactly the same as in the normal flow algorithm. Having the points $P^{(1)}$ and $P^{(2)}$ on Γ with corresponding tangent vectors $T^{(1)}$ and $T^{(2)}$, respectively, the prediction $Z^{(0)}$ of the next point on Γ is given by (8).

In order to use this predictor, a means of calculating the tangent vector $T^{(2)}$ at a point $P^{(2)}$ is required. This is done by solving the system

$$\begin{bmatrix} dH\left(P^{(2)}\right) \\ T^{(1)\,t} \end{bmatrix} y = \begin{pmatrix} 0 \\ \vdots \\ 0 \\ 1 \end{pmatrix} \tag{10}$$

for y. (Here, superscript t denotes the matrix transpose.) Normalizing y gives

$$T^{(2)} = \frac{y}{\|y\|}.$$

The last row of (10) insures that the tangent $T^{(2)}$ makes an acute angle with the previous tangent $T^{(1)}$. It is the augmentation of the Jacobian matrix with this additional row which motivates the name "augmented Jacobian matrix algorithm." The solution to (10) is found by computing a QR factorization of the matrix, and then using back substitution.

Starting with the predicted point $Z^{(0)}$, the correction is performed by a quasi-Newton iteration defined by

$$Z^{(k+1)} = Z^{(k)} - \begin{bmatrix} A^{(k)} \\ T^{(2)\,t} \end{bmatrix}^{-1} \begin{pmatrix} H\left(Z^{(k)}\right) \\ 0 \end{pmatrix}, \qquad k = 0, 1, \ldots \tag{11}$$

where $A^{(k)}$ is an approximation to the Jacobian matrix $dH\left(Z^{(k)}\right)$. The last row of the matrix in (11) insures that the iterates lie in a hyperplane perpendicular to the tangent vector $T^{(2)}$. Equation (11) is the quasi-Newton iteration for solving the augmented nonlinear system

$$\begin{pmatrix} H(w) \\ T^{(2)\,t}\left(w - Z^{(0)}\right) \end{pmatrix} = 0.$$

When the iteration (11) converges within some tolerance, the final iterate $Z^{(*)}$ is accepted as the next point on the zero curve Γ . The step size estimation algorithm is an adaptation of a procedure developed by Rheinboldt. The final phase of the algorithm, computation of the solution at $t = 1$, is entered when a point $P^{(2)} = (x^{(2)}, t^{(2)}$ is generated such that $t^{(2)} \geq 1$. Since $P^{(2)}$ is the first such point, the solution must lie on Γ somewhere between $P^{(2)}$ and the previous point $P^{(1)}$. The algorithm for finding this solution is a two step process which is repeated until the solution is found. First, starting from a point $P^{(k)}$, a prediction $Z^{(k-2)}$ for the solution is generated such that $t^{(k-2)} = 1$. Second, a single quasi-Newton iteration is performed to produce a new point $P^{(k+1)}$ close to Γ but not necessarily on the hyperplane $t = 1$.

Section 4. Numerical Results.

This section is devoted to a description of numerical experiments. It is divided into two parts. In the first part, we present a discussion of certain key numerical issues:

- path failure,
- singular solutions,
- run parameters (first run and rerun),
- CPU time vs. WORK.

In the second part we present the actual experiments, organized in three subsections corresponding to the three physical problems studied. In fact, we shall consider:

- three problems, one each from geometric modeling, chemical equilibrium studies, and the kinematics of mechanisms;
- three path-tracking techniques, the QF, NF, and DF methods from HOMPACK, as discussed in Section 3;
- two approaches to constructing homotopies, one "special" (customized, nongeneric, m-homogeneous) and one "traditional" (generic, uncustomized).

One of the purposes of this numerical work is to compare the path-tracking approaches, another is to compare the homotopy construction strategies. In addition, the issue of scaling (which is critical for polynomial continuation) is clarified through the use of the SCLGNP scaling algorithm (see [20], [35], or [24] Chap. 5) and the projective transformation, both of which are optional for the traditional homotopy and have a powerful beneficial scaling effect on most problems.

In [26] the same problem set is studied with respect to the CONSOL8 code given in [24].

Here is a summary of results. See Section 5 for more details. NF and DF generally outperformed QF. In some cases NF did better than DF and in others DF did better than NF. The special (m-homogeneous) way of constructing homotopies leads to solver performance that is faster and more reliable than the traditional (generic) way. However, it requires more (human) time and ingenuity. Also, the SCLGNP scaling and the projective transformation improve solver performance, sometimes considerably.

Section 4.1. Key Numerical Issues.

Section 4.1.1. Path Failure.

Path failure can be divided into two types:

1. Recognized HOMPACK failures, signified by the output variable FLAG.
2. Endpoint accuracy and path crossing failures.

The recognized HOMPACK failures are:

- FLAG=3: not enough steps allowed for the path. When a FLAG=3 is returned, three cases are possible:
  - There is nothing wrong with the path. If more steps are allowed, the tracker will converge.
  - The FLAG=3 is the result of path crossing. See the discussion below.
  - The tracker has encountered a singularity on the path. For properly constructed polynomial homotopies, this can occur only as t approaches 1. Sometimes by allowing more steps, a FLAG=1 can be obtained (signifying a successful HOMPACK convergence). Sometimes no amount of steps will induce FLAG=1. Even if FLAG=1, the accuracy of the solution estimate may be poor. At this time, singular endpoints are dealt with most effectively by postprocessing the endpoint with a local solver, as discussed in the subsection below on Singular Solutions.
- FLAG=5: the path fails at the beginning (t near 0). The path tracker refuses to get started. Often, a negative t is returned. One frequently effective fix is to put the following code at the beginning of subroutines RHO and RHOJAC:

```
IF ( LAMBDA .LT. 0.D0 ) LAMBDA = 0.D0.
```

(The homotopy parameter t is denoted "LAMBDA" in HOMPACK.) We recommend doing this routinely, but we avoided it in our tests to clarify the issues. Otherwise, rerunning with tighter run parameters often works. (See the Run Parameters subsection below.)

- FLAG=6: The path cannot attain the requested endpoint accuracy. This generally occurs because the solution is ill conditioned. The best fix is postprocessing (as discussed in the Singular Solutions subsection below), if the solution seems important, and accepting the run as satisfactory, if the solution can be judged unimportant (or sufficiently accurate) without further refinement. However, rerunning with ANS larger will sometimes eliminate the error condition. (ANS is defined in the Run Parameters subsection below.)

Endpoint accuracy and path crossing failures are more difficult to diagnose and fix than HOMPACK FLAG = 3, 5, or 6 conditions. When a solution is ill conditioned, a FLAG=1 can be obtained with reasonable run parameters for a solution that is not accurate to even one digit! (For example, see the discussion of Butler's problem below.) Within the context of HOMPACK, such a difficulty can sometimes be discovered by setting

the answer-closeness parameters (ANS parameters) very small. But a more direct, reliable, and convenient approach to the issue of endpoint accuracy is to test the accuracy and conditioning of the solution estimate as a postprocessing step (i.e., post-HOMPACK) and to refine important inaccurate solutions, as discussed in the Singular Solutions subsection below.

By "path crossing" we mean that the path tracker begins on one path but crosses to a different path for some t between 0 and 1 and continues on the second path thereafter. We might say that "the paths cross numerically" even though they don't cross in exact arithmetic. Unfortunately, there is no foolproof way to detect path crossing. The possibility of path crossing is the main motivation for setting path-tracking parameters conservatively. (For an example of path crossing, see the discussion below of solving the mechanism problem with the DF method.) If we suspect a path has crossed we can rerun it with tighter run parameters. But how can we detect path crossing? Basically, only by detecting deficiencies in the computed full solution set. For example, if two paths converge to the same nonsingular solution, a path crossing has occurred and the two paths should be rerun.

Section 4.1.2. Singular Solutions.

The most satisfactory polynomial continuation occurs in solving a system without singular solutions. In this case, we know that exactly the Bezout number of distinct solutions should be obtained. Thus we have a simple double check on the run. At the same time, nonsingular solutions are handled satisfactorily by all the HOMPACK path trackers. The picture is not so nice for singular solutions. First, to obtain a Bezout count, we must count the solutions by their multiplicities, and we don't know the multiplicities independent of the HOMPACK computation. Second, singular solutions are expensive to compute and hard to compute accurately. Judging the accuracy and success of a path computation is difficult when the path ends in a singular solution. Even deciding if a solution is singular or not is a judgement call. HOMPACK is not constructed to deal with the higher-order singularities generated by polynomial continuation. The rank of the Jacobian matrix at a singular solution is always real-deficient by a multiple of 2. The theories on which the HOMPACK codes are based break down at such singular endpoints. (CONSOL8 from [24] does better, perhaps because of the special way in which the homotopy parameter is managed. See [24, Chap. 4] and [26].)

Our experience with HOMPACK suggests that a singular solution may generate a FLAG=1, FLAG=3, or FLAG=6 condition. We recommend that when HOMPACK has completed, all solution estimates z^0 be tested for accuracy and singularity by computing

$$cond = cond(\frac{dH}{dz}(z^0, 1))$$

(the condition number of the Jacobian matrix at z^0 when $t = 1$) and

$$resid = \left\| \left[\frac{dH}{dz}(z^0, 1) \right]^{-1} H(z^0, 1) \right\|$$

(the norm of the Newton's method residual). If *cond* is large and *resid* is not small, then z^0 should be refined by a local method if it is judged to be of interest. Plain old Newton's method can be used to improve the accuracy of a singular solution. We have been surprised at how well a (complex) Newton's method performs in refining a singular solution if allowed enough iterations. See [18] for some alternative methods and caveats. We specifically recommend that FLAG=6 solutions and FLAG=3 solutions when t is near 1 be refined in this way rather than by rerunning HOMPACK. A final check of the norm of the difference between the unrefined and refined solution estimates helps weed out "endpoints" that were not actually close to a solution.

Section 4.1.3. Run Parameters.

We distinguish between "first run" and "reruns." Reruns may be reruns of the whole problem ("full reruns") or of individual paths. The process we envision is a first run, followed (perhaps) by several full reruns, followed (perhaps) by reruns of several individual paths.

How conservatively should the first run parameters be set? The critical question is: can the computed solution list be double checked external to HOMPACK? For example, is the physical-solution sublist easy to characterize? Is it easy to see if all the sought solutions have been obtained? Using the special homotopy for the mechanism problem (below) we expect 32 finite solutions and 32 solutions at infinity. Further 16 of the finite solutions will not satisfy the actual manipulator system. (There are several additional equations to be satisfied.) Half of the solutions at infinity will have $z_9 = 0$ and half will have $z_{10} = 0$ and none both. Generally (depending on the manipulator, hand position, and orientation) the solutions will be nonsingular and 16 will be actual solutions to the manipulator. With this much information about the final solution list, we are in a good position to evaluate the run and to rerun unsuccessful paths. Another case in point is provided by chemical equilibrium systems. Each such system has a single physical solution and it is the only positive real solution. In cases like these, we can "take chances" in choosing "loose" run parameters. If we cannot double check a run by the characteristics of the solution list, we must set tighter run parameters to start and be prepared to rerun the entire problem until path crossing (for example) can be ruled unlikely from general principles.

Now we will state how run parameters were set for the experiments in this paper, for the first run and for reruns on all problems, for the three HOMPACK methods. First, a list of the main parameters, with method-specific variable names and first run values, are given in Table 1.

We note the following on run parameters and initialization issues. (Refer to Table 1.)

1. In table 1, "–" indicates that the variable is not used in this method.

2. LIMITD, the "maximum number of steps" parameter, is initialized at the beginning of subroutines FIXPQF, FIXPNF, and FIXPDF (the main HOMPACK subroutines for the QF, NF, and DF methods, respectively.) LIMITD is nominally set to 1000.

3. With reference to the START variable: We modified FIXPQF so that "H=0.1" (in the initialization section at the beginning of the program) was replaced by "H=SSPAR(5)," so that we could easily change this constant.

4. We always take ARCRE=ARCAE, and refer to these two variables together as ARC. ARC specifies how close the path tracker will try to keep the path estimate to the actual path, for $0 \leq t < 1.0$. Similarly, we always take ANSRE=ANSAE, called ANS. ANS is the error tolerance for the solution at $t = 1.0$. The strategy for using ARC and ANS is rather complicated and differs for DF, NF, and QF. See [33], [34], [35], and [36] for the details.

5. The RHOA subroutine (which DF uses but QF and NF do not) is coded as a Newton's method which refines z rather than the "random" parameters. It is necessary that z be adjusted for path correction, because the special homotopies do not use an array of random parameters.

Now here is the rerun strategy for the QF method. The rerun strategies for the other methods are modifications of this, as noted below.

1. If more than than half the paths do not return FLAG=1 (or are not otherwise satisfactorily accounted for), then rerun the whole problem reducing MAXSS and/or ARC and/or START by one order of magnitude. (If, to simplify parameter adjustment, only one parameter is to be changed, use MAXSS.)

2. If path crossing is suspected, rerun the doubtful paths with MAXSS and/or ARC and/or START reduced by one order of magnitude. (If, to simplify parameter adjustment, only one parameter is to be changed, use MAXSS.)

3. Rerun FLAG=5 paths with START reduced by an order of magnitude. If two reruns don't suffice, begin reducing MAXSS and/or ARC also.

4. Consider not rerunning FLAG=6 paths. Examine the endpoint, z^0, of such a path to see if it is close enough to a solution to justify post-HOMPACK refinement. That is, check the *cond* and *resid* test values as defined in the subsection above on Singular Solutions. Otherwise, rerun with ANS increased to find an endgame tolerance that works. Note that difficulty generally occurs with ill conditioned solutions.

5. FLAG=3 paths may be the result of path crossing, in which case the appropriate action is described in 2. above. Otherwise, rerun a FLAG=3 path with LIMITD

increased, if t is not near 1. If t is near 1, judge if the path is converging to an ill conditioned solution. If so, consider refining post-HOMPACK rather than rerunning, as described in the Singular Solutions section.

The rerun strategy for NF is the same as QF, except that START cannot be adjusted. Use ARC instead.

The rerun strategy for DF is the same as for QF, except that START and MAXSS cannot be adjusted. Use ARC instead.

On QF and NF: Making MAXSS smaller is an extreme change. It tends to increase the computational work by a big increment (like, say, an order of magnitude). On the other hand, it is also a powerful change. Rarely will setting MAXSS smaller and smaller fail to induce eventual convergence (at a cost in CPU time). Making ARC smaller can also improve a run, but this has a more indirect effect on performance. The advantage of adjusting only one parameter for reruns will be obvious to anyone with multivariable optimization experience.

Ideally, most paths will be satisfactory on the first run, and we would (at most) need to rerun only a few paths. However, if a number of paths fail, the danger of path crossing will be great. Thus, we use as a (flexible) rule of thumb to rerun all paths if half the paths fail.

Section 4.1.4. Run Time vs. WORK.

We use the following measure of computational work:

$$\text{WORK} = \text{NOFUN} + (\text{N} + 1) * \text{NOJAC}$$

where NOFUN is the number of function evaluations (calls to subroutine RHO) and NOJAC is the number of Jacobian matrix evaluations (each N+1 calls to subroutine RHOJAC equals one Jacobian matrix evaluation). The NFE value returned by HOMPACK equals NOJAC. We should consider two issues:

(1) How does WORK compare to CPU time?

(2) Does WORK let us compare the computational cost of the QF, NF, and DF methods?

First, note the following:

- QF calls RHO more than RHOJAC. Thus WORK >(N+1)*NFE.
- NF calls RHO exactly as often as RHOJAC. Thus WORK = (N+2)*NFE.
- DF never calls RHO. Thus WORK = (N+1)*NFE.

It is reasonable to use WORK to compare two runs using the same method. This is not so clear for comparisons between methods. Our tests suggest the following rule of

thumb for estimating CPU times from WORK values. In Tables 5, 7, 8, and 12, WORK and CPU time values (in minutes on IBM 3090) are given. The WORK/CPU ratios are within a factor of 2 for DF and NF (usually much closer). Comparing WORK/CPU for NF to WORK/CPU for QF we obtain factors between 2 and 4 (approximately). We conclude that we can compare WORK values for NF and DF, but to compare these with QF values for WORK we should multiply the latter by a factor of 3.

Section 4.2. The Problems and Numerical Results.

We describe each problem below and then report the result of solving the problem using each path-tracker with a generic homotopy and a new homotopy. For the generic homotopy we use $g^0(z)$ in place of $\gamma g(z)$ in (2) where

$$g_j^0(z) = p_j^{d_j} z_j^{d_j} - q_j^{d_j}$$

with the "random" coefficients p_j and q_j given in Table 2. (Here $d_j = \deg(f_j)$.) This is a traditional homotopy as defined in [23] and [24], similar to the variety of other generic ways of defining H. (See the list of references cited in Section 1.) With this generic homotopy, we use the 1-homogeneous projective transformation of f realized by $z_j \leftarrow z_j/z_{n+1}$ for $j = 1$ to n and

$$z_{n+1} = \sum_{j=1}^{n} \beta_j z_j \quad + \quad \beta_{n+1}$$

with the β_j taken from Table 3. (HOMPACK does not support the double-subscript notation useful for the general discussion of the m-homogeneous case. To simplify setting up the various codes for our experiments, we have adopted the convention of ordering the variables in sequence with the homogeneous variables last. Thus

$$z_{0,1}, z_{1,1}, z_{2,1}, z_{3,1}, z_{0,2}, z_{1,2}, z_{2,2}$$

become

$$z_6, z_1, z_2, z_3, z_7, z_4, z_5,$$

respectively.) It has been our experience that the traditional homotopy is more efficient when used with such a projective transformation, and it is specifically the case with the test problems in this paper. (Note the computational comparisons for the first problem below.)

Although we tested the alternative to the traditional homotopy of taking the projective transformation of the homotopy (rather than of f), the differences between these variants were minor. For simplicity, we do not include the numerical results here of this alternative.

For the $m-$homogeneous homotopies (called "special" homotopies), we define γ by $r = 1.0$ and $\theta = 0.876534$. The g's themselves are customized to each problem, as given

below. The exact projective transformation formulas for each problem are specified below also.

Note that the HOMPACK path-trackers do not specifically acknowledge the fact that the homotopy paths are strictly increasing in t. The path-tracking parameters for each path-tracker were the same for each run and as comparable as possible between methods. (See Section 4.1.3 on Run Parameters.)

Problem 1. **Geometric Intersection Problem.**

Background:

This system of two quadrics arose in a geometric-modeling context. (See [26] or [24, Chap. 8].) It has two real solutions and a complex conjugate pair of solutions. It has no solutions at infinity. Thus it has total degree equal to 4, and the special homotopy will be 1-homogeneous. Three of the solutions are in the circle of radius 2, and the other has norm approximately equal to 2343. This is the last problem presented in [35], and the solutions are listed there.

This relatively mild problem is included here to show that even when a system exhibits no special numerical difficulties, it can be solved with less work by the special homotopy approach. We have solved the problem with and without the SCLGNP scaling and with and without the projective transformation of f (when the traditional homotopy is used) to illustrate the value of these options. In the more difficult two problems presented below (the chemical problem and the Mechanism problem), we have omitted running some of these variants, for simplicity.

Definition of f:

$$f_l = a_{l,1}z_1^2 + a_{l,2}z_2^2 + a_{l,3}z_1z_2 + a_{l,4}z_1 + a_{l,5}z_2 + a_{l,6}$$

for $l = 1, 2$ where the $a_{l,j}$ are given in Table 4.

The Special g:

The special g is the same as f, except that we take $a_{1,1} = a_{1,3} = a_{1,4} = 0$. The logic of this choice of g is:

1. g is triangular. We can solve $g_1(z_2) = 0$ for z_2 using the quadratic formula, substitute for z_2 in $g_2(z) = 0$, and solve for z_1 with two more applications of the quadratic formula.

2. We can make f into a triangular system in four ways, via a choice of an equation and of a variable to delete in the chosen equation. g_1 includes the terms of f of greatest magnitude that can be produced in this way.

The projective transformation is defined via $z_j \leftarrow z_j/z_3$ for $j = 1$ to 2 where

$$z_3 = \beta_1 z_1 + \beta_2 z_2 + \beta_3$$

with the β_j chosen from Table 3.

Experimental Results:

The results of this experiment are listed in Table 5. Almost every run "worked," in the sense that the correct solution set was obtained without the need for reruns. The exceptions are the two runs of the traditional homotopy without scaling and without the projective transformation using the QF and NF methods. Not enough steps were allowed for the paths converging to the large solution, and a FLAG=3 was returned in each case. Increasing LIMITD was the easy fix. Note the significant difference in WORK and CPU time in solving the traditional homotopy with and without the projective transformation and with and without scaling. The special homotopy is not particularly sensitive to scaling, but the computations are quick in any case.

Problem 2. **Chemical Equilibrium System (Butler's Problem).**

Background:

Chemical equilibrium systems generate polynomial systems. They are characterized by extreme scaling and unique model formulation issues. The problem included here is the second example in [20], "Butler's Problem." See [20] for a discussion of its chemical significance.

This system of a quartic and a cubic has seven finite real solutions and a finite complex conjugate pair of solutions. It has one solution at infinity, of multiplicity three. (The finite solutions are listed in [20], Table 6.) Without the SCLGNP-scaling and the projective transformation, these solutions are very ill conditioned. Their conditioning is much improved when these options are used. Here is a summary: With scaling and the transformation, the solutions have condition numbers less than 3000, except the complex conjugate pair, whose condition number is about 8×10^8. The unscaled transformed solutions all have condition greater than 10^{10}, except the complex pair whose condition is about 3×10^6. Without scaling and without the transformation, the solutions all have condition greater than 10^{10}. With scaling but without the transformation, the solutions have condition number less than 3×10^4, except one with condition about 8×10^5 and the complex pair whose condition is greater than 10^{10}.

This system has total degree equal to 12. For the special homotopy we use a 2-homogeneous homotopy with associated Bezout number of 9. Thus on $P = P^1 \times P^1$ the system has no solutions at infinity. It is especially significant to eliminate a triple singularity, because this solution is expensive to compute and, with a traditional homotopy, three paths converge to it.

Definition of f:

$$f_1 = a_{1,1}z_1^4 + a_{1,2}z_1^3 z_2 + a_{1,3}z_1^3 + a_{1,4}z_1 + a_{1,5}$$
$$f_2 = a_{2,1}z_1 z_2^2 + a_{2,2}z_2^2 + a_{2,3}$$

where the $a_{i,j}$ are given in Table 6. We 2-homogenize the system via the transformation $z_1 \leftarrow z_1/z_3$ and $z_2 \leftarrow z_2/z_4$. Thus z_3 and z_4 are the homogeneous variables. We can compute the Bezout number from the combinatorial product (1) for this case:

$$D = (4\alpha_1 + 1\alpha_2)(1\alpha_1 + 2\alpha_2).$$

We then confirm that the coefficient of $\alpha_1\alpha_2$ is 9. The projective transformation of the special homotopy is realized by:

$$z_3 = \beta_1 z_1 + \beta_3$$
$$z_4 = \beta_2 z_2 + \beta_4$$

where the β_j are taken from Table 3.

Special choice of g:

The special g is the same as f except we set $a_{1,2} = 0$. This makes g triangular, and we can solve as in problem 1. (We use the method from [17] to solve the quartic, rather than the quartic formula. The computational cost of this is trivial compared to the total cost of the continuation run.) However, there is a "trick." g_1 should be type (4,1), because f_1 is type (4,1). (The "type" is defined at the end of Section 2.) However, if we homogenize g by the prescription given in Section 2, then g_1 will be type (4,0), because g_1 contains no z_2. The trick is to multiply g_1 by z_4. Then g has the same 2-homogeneous structure as f, and $g(z) = 0$ has exactly 9 solutions in $P^1 \times P^1$, as required. There are eight finite solutions and one solution at infinity:

$$(z_1, z_2, z_3, z_4) = (-a_{2,2}/a_{2,1}, 1, 1, 0)$$

corresponding to

$$([(-a_{2,2}/a_{2,1}, 1)], [1, 0]) \in P^1 \times P^1.$$

Experimental Results:

Refer to Tables 7 and 8. The special homotopy is relatively easy to solve, whether scaled or unscaled. In spite of the ill conditioning of the solutions, none of the methods had any significant trouble. However, the scaled problem required no reruns at all, while the unscaled problem needed some reruns. The failed paths were clearly marked (FLAG=5, FLAG=6) and corrected easily (by reducing MAXSS).

The traditional homotopy is difficult to solve. The unscaled version defeated all three solvers, for all reasonable choices of rerun parameters. Naturally, we could have forced

the issue by heroic efforts (say, by using extended precision). However, this seemed pointless. Chemical problems should be scaled, as noted in [20] and [24, Chap. 9]. And the traditional homotopy, which forces a blending of f with a generic g, is more sensitive to scaling and other mismatching issues than a well chosen special homotopy. Note that the ill conditioning of the solutions, per se, is not the difficulty, but rather the ill conditioning of the homotopy. Each of the methods required three full reruns of the scaled problem, for different reasons, and with different outcomes. Still, we can conclude that the solution process (including the rerun strategies) is successful, allowing that post-HOMPACK refinement of the complex solution pair is required for reasonable accuracy. In fact, the physical solution (the positive real solution) is relatively easy to find and is found with reasonable accuracy without refinement.

Here are some additional notes on Table 8:

1. For the QF method: For the all-path reruns, we reduced (only) MAXSS (twice). The second all-path rerun had four paths for which FLAG $\neq 1$. They were settled as follows. The FLAG=6 path is close enough not to rerun. The FLAG=5 path converges when rerun with MAXSS=0.01 and START=0.001. The WORK for this rerun is 1219, CPU=0.008 min. The two FLAG=3 paths were rerun with LIMITD = 10^6, MAXSS=10^{-2}, and START=0.1. They converged but with WORK = 1934360, CPU = 11.581 minutes . This "convergence" is pathelogical both for the amount of WORK and also for the inaccuracy of the solution estimates. See Note 4 below.

2. For the NF method: For the all-path reruns, we reduced (only) MAXSS (twice). No reasonable parameter settings for NF caused the final two FLAG=3 paths to change to FLAG=1. These paths should converge to the complex solution pair. See Note 4 below.

3. For the DF method: These three runs are a good example of how path crossing and endpoint inaccuracy may not be detected by HOMPACK. For each run, every path returns a FLAG=1. However, runs 1 and 2 include crossed paths and the complex solution pair is not computed accurately. An easy post-HOMPACK check shows for run 1 that a path has crossed; a nonsingular solution is found twice. Nothing in the solution list for run 2 indicates that the run is less than perfect (except the inaccuracy in the complex pair). In fact, two paths have crossed with each other. The solution list is correct, but the path computation is faulty. The reruns here were done reducing ARC.

4. No run of the traditional homotopy returned even one digit of accuracy for the complex solution pair:

$$z_1 = -9.354537 \times 10^3 \pm 1.367812 \times 10^{-2} i$$

$$z_2 = 5.345006 \times 10^{-17} \pm 7.310953 \times 10^{-11} i$$

The condition number of the Jacobian matrix at these solutions is about 8×10^8 (scaled and transformed). Eventually (after reruns) the QF and DF methods return a FLAG=1 for this solution pair. In this sense, the performance of NF is better, because it insists on returning a FLAG=3, clearly indicating failure. The Newton's method residual *resid* shows that the path endpoint is (badly) inaccurate. However, Newton's method (implemented in double precision) converges from this endpoint to single precision accuracy, in spite of the ill conditioning.

Problem 3. **Mechanism Problem from Tsai and Morgan [32].**

Background:

The inverse position problem (IPP) for six-revolute-joint manipulators is a basic problem in mechanisms. Given parameters describing a manipulator and a desired hand position and orientation, the problem is to find all the sets of joint angles that put the hand in this position and orientation. The IPP is analyzed in [32] and described for a more general audience in [24], Chapter 10.

The IPP might have an infinite number of physical solutions, in cases where the hand position and orientation can be attained by the free rotation of a joint and in other degenerate configurations. We focus on finding the geometrically isolated solutions. Usually these are nonsingular, but not always. In P^8 the 1-homogeneous form of the system has total degree 256, an infinite number of solutions at infinity, and (we can prove) at most 64 finite solutions (unless there are an infinite number), while we observe at most 32 finite solutions (unless there are an infinite number). (The complicated formulas by which manipulator data generates polynomial coefficients make exact analysis here difficult.) In $P^4 \times P^4$ the 2-homogeneous form of the system has Bezout number 96, 8 solutions at infinity (each of multiplicity 4), and (we observe) 32 finite solutions and 32 (additional) solutions at infinity. The test results confirm that the problem should be solved in its 2-homogeneous form rather than the 1-homogeneous. However, we have some choice of 2-homogeneous approaches.

In [26] the IPP was solved using three homotopies, the traditional and two 2-homogeneous homotopies. Here we will use only the traditional and the second 2-homogeneous homotopy from [26] (denoted by its start system G^b in [26]). The homotopy we will use has as a start system the IPP system with a particular choice of coefficients. Thus, to use this homotopy, we must solve the IPP once using a different homotopy, thereafter using the solutions obtained from that one run as start points to solve the IPP for a variety of other choices of coefficients.

Definition of f:

This is system (7) in [25], reproduced here for convenience:

$$
\begin{aligned}
f_l = {} & a_{l,1}z_1z_3 + a_{l,2}z_1z_4 + a_{l,3}z_2z_3 + a_{l,4}z_2z_4 \\
& + a_{l,5}z_5z_7 + a_{l,6}z_5z_8 + a_{l,7}z_6z_7 + a_{l,8}z_6z_8 \\
& + a_{l,9}z_1 + a_{l,10}z_2 + a_{l,11}z_3 + a_{l,12}z_4 \\
& + a_{l,13}z_5 + a_{l,14}z_6 + a_{l,15}z_7 + a_{l,16}z_8 \\
& + a_{l,17} \qquad (l = 1, \ldots, 4) \\
f_l = {} & z_{2l-9}^2 + z_{2l-8}^2 - 1 \qquad (l = 5, \ldots, 8)
\end{aligned}
$$

The values of the coefficients are given in Table 9. These are defined by the second manipulator example in [32]. We 2-homogenize via $z_j \leftarrow z_j/z_9$ for $j = 1, 2, 5, 6$ and $z_j \leftarrow z_j/z_{10}$ for $j = 3, 4, 7, 8$, letting z_9 and z_{10} be the 2-homogeneous variables. There are four equations of type (1,1), two of type (2,0), and two of type (0,2). The Bezout number calculation requires that we find the coefficient of $\alpha_1^4\alpha_2^4$ in equation (1), which in this case is

$$D = (1\alpha_1 + 1\alpha_2)^4(2\alpha_1 + 0\alpha_2)^2(0\alpha_1 + 2\alpha_2)^2.$$

Thus, the Bezout number is 96. The projective transformation yields

$$
\begin{aligned}
z_9 &= \beta_1 z_1 + \beta_2 z_2 + \beta_5 z_5 + \beta_6 z_6 + \beta_9 \\
z_{10} &= \beta_3 z_3 + \beta_4 z_4 + \beta_7 z_7 + \beta_8 z_8 + \beta_{10}
\end{aligned}
$$

with coefficients from Table 3.

Choice of special g:

The special g has exactly the same form as f but with coefficients given in Table 10. These are defined by the third manipulator example in [32].

We could simply choose a 2-homogeneous g with 96 nonsingular solutions, such as system (10) in [25]. (This defines the "G^a" homotopy for this problem in [26].) We might do this if we wanted to solve the IPP only once. However, if we wish to solve this problem many times, we can create a much more efficient homotopy. It turns out that $f(z) = 0$ has exactly the same set of 8 multiplicity-four solutions at infinity in $P^4 \times P^4$, independent of the choice of coefficients. (This is proven as Theorem 4 in [26].) Therefore, by Theorem 1 we can choose g to have the same form as f and ignore these solutions as start points. This leaves just 64 paths to track. We obviate the issue of finding a g that is easily solvable by simply choosing one and solving it using another homotopy. This becomes the start system for all other IPP systems. In fact we use the third example in [32] as our start system. Note that this is not generic or random at all. We use it to solve the second example in [32]. While these two manipulators themselves are not particularly related to each other, we specify the same hand position and orientation. Thus the two problems are

linked by this physical relationship and are not "randomly chosen." But that is the point. In practice, problems are never randomly chosen.

Experimental Results:

Refer to Table 11. The special homotopy uses much less WORK and is much more reliable than the traditional homotopy. Although we have reduced the number of paths from 256 (traditional) to 64 (special), a 32 path homotopy does not follow, because the finite start points do not yield paths converging to finite solutions. Let us consider this in a little more detail.

The start points for the special homotopy consist of a set, A, of finite points and a set, B, of points at infinity, where $|A| = |B| = 32$. We observe 19 of the points in A and 13 of the points in B lead to the 32 finite solutions of $f = 0$ in $P^4 \times P^4$. The remaining points of A and B lead to infinite solutions of $f = 0$ in $P^4 \times P^4$. Thus *the finite solutions of $g = 0$ are not necessarily connected to the finite solutions of $f = 0$*, even though f and g have exactly the same structure.

The special homotopy is relatively easy to solve, because:

- The 64 solutions are nonsingular.
- The solution set has well known characteristics. This makes double checking the run and knowing which paths to rerun easy. For example, there are 32 finite solutions and 32 solutions at infinity. The 32 solutions at infinity divide into two groups of 16, those with $z_9 = 0$ and those with $z_{10} = 0$. No solutions have both $z_9 = 0$ and $z_{10} = 0$. The finite solutions divide into two groups of 16, those that obey the additional relations that actual manipulator solutions must obey, and the others. (See "extraneous solutions" in [32].)

For the traditional homotopy, 32 paths should converge to finite solutions, while 226 should converge to (singular) solutions at infinity. Because of the singularity of the endpoints of these 226 paths, it is unrealistic to expect HOMPACK to return "FLAG=1." Many of the paths will be labeled FLAG=6 or FLAG=3 (while some will have FLAG=1). Rather than force these paths to converge, it is more reasonable to call a run successful when 32 finite solutions are satisfactorily computed. This is less definitive than obtaining 64 distinct nonsingular solutions to the special homotopy. A path crossing to a path going to infinity is particularly unpleasant, since, while one easily detects a deficiency in the solution list (because we expect 32 finite solutions), every path not converging to a finite solution may have to be rerun to find the crossed path. For these tests, we took the expedient of merely rerunning the paths we knew should have converged to finite solutions. We conclude that, while completely correct solution lists can be obtained using the traditional homotopy, the special homotopy is not only less work, but it is also more reliable.

Let us make several observations related to Table 11. The FLAG=6 endpoints are sufficiently close to solutions, so that reruns are not necessary. The FLAG=3 runs are actually crossed paths. The superior performance of DF on the traditional homotopy may be related to its relative efficiency on singular endpoints.

With reference to the special homotopy reruns: For QF, both FLAG=5 paths converged when rerun with MAXSS=0.01. The NF run had one crossed path, in addition to the two FLAG=5 paths. Rerunning with ARC=10^{-6} corrected these. The DF run had one path cross, and it converged correctly with ARC=10^{-6}.

With reference to the traditional homotopy reruns: For QF, the FLAG=3 runs were corrected with START=0.01. The NF FLAG=3 path converged with MAXSS=0.1. The DF run was perfect.

Section 5. Summary and Conclusions

It is not easy to compute the full solution set to small systems of polynomials using homotopy continuation. In spite of the seeming completeness of theoretical results on polynomial continuation, divergent paths (solutions at infinity), singular solutions, and extreme scaling of coefficients can create catastrophic numerical problems. Further, the large number of paths that typically arise can be discouraging.

We have summarized the m-homogeneous strategy for constructing polynomial homotopies and reported on the performance of three standard path-tracking algorithms (as implemented in HOMPACK) in solving three physical problems of varying degrees of difficulty using both $m-$homogeneous and traditional homotopies. We have shared our experiences (and the resulting rules of thumb) in the hope that they might act as guidelines for homotopy construction and choice of parameter settings. Computational comparisons illustrate and contrast the major options.

HOMPACK is deficient in its handling of singular endpoints. Currently, the best fix is to post-process (external to HOMPACK) endpoints judged to be singular . This is not just a local issue (endpoint refinement). Butler's problem shows a system with ill conditioned solutions generating an ill conditioned homotopy, and difficulties occur long before t is close to 1. Scaling and the projective transformation improve, but don't cure, these difficulties. The resilience of Newton's method in the face of ill conditioning suggests that continuation approaches that appropriately exploit Newton's method may have an edge. To some extent the NF and QF methods do this, because their correctors are based on Newton's method. The CONSOL8 code from [24] is another example. It solves Butler's problem unscaled with the traditional homotopy (as reported in [26]), which none of the HOMPACK codes can do.

If the system being solved is suspected of having a number of unwanted singular solutions (say, a collection of singular solutions at infinity), then to save CPU time one might consider terminating paths that seem to be converging to singular solutions. This makes a heuristic out of the algorithm, but singular solutions do that anyway, in implementation if not in theory. The method whereby one truncates paths could be as simple as carefully choosing the LIMITD parameter (large enough to capture the finite solutions and no larger), or something more elaborate, like monitoring the condition number of the Jacobian matrix as t approaches 1, or monitoring the reduction in the step size as t approaches 1 (usually a strong indicator).

A well chosen special m-homogeneous homotopy can be much faster and more reliable than a traditional homotopy. Any problem that will be solved many times for various choices of coefficients should be solved with a special homotopy.

The SCLGNP scaling and the projective transformation frequently improve the speed and reliability of path tracking. We believe that they should be used as a matter of course.

How do the QF, NF, and DF methods compare with each other? Because the run parameters are not precisely comparable, one should be cautious in comparing methods. In our tests, the QF method was consistently outperformed by the NF and DF methods. We don't know why. The NF method seems to be faster on "quick and easy" problems, while the DF method sometimes (but not always) did better on the more challenging runs. Because run times and WORK are sensitive to parameter settings, rerunning the experiments with different nominals would produce different results, although (we suspect) not significantly different results. The DF method seems to spend less time on singular endpoints, which accounts for its faster performance on the traditional homotopy runs of Butler's problem and the Tsai-Morgan manipulator problem. Thus, if a new singular end game is developed for the NF method, the advantage may disappear. DF is less prone to the FLAG=5 difficulty, in which the path tracking fails at the beginning. However, we were able to cure all FLAG=5 paths by tightening the run parameters, suggesting that with tighter nominal values, this "difference" might not have come up. (An exception is the traditional homotopy with the unscaled Butler's problem, where most paths failed to converge, usually with a FLAG=5, for all methods.)

HOMPACK was not developed specifically for polynomial continuation. It neither exploits the fact that paths are strictly increasing in the homotopy parameter nor anticipates the need for dealing with highly rank deficient Jacobian matrices at the endpoints. In spite of this, we have succeeded in solving some difficult polynomial problems using HOMPACK, and we believe that with attention to the issues and options we have discussed, HOMPACK can be a powerful method for finding all geometrically isolated solutions to polynomial systems.

References.

[1] S.C. Billups, *An augmented Jacobian matrix algorithm for tracking homotopy zero curves*, M.S. Thesis, Dept. of Computer Science, VPI and SU, Blacksburg, VA, Sept. 1985.

[2] P. Brunovský and P. Meravý, *Solving systems of polynomial equations by bounded and real homotopy*, Numer. Math., 43 (1984), 397–418.

[3] S. N. Chow, J. Mallet-Paret, and J. A. Yorke, *Finding zeros of maps: Homotopy methods that are constructive with probability one*, Math. Comput., 32 (1978), 887–899.

[4] ——, *A homotopy method for locating all zeros of a system of polynomials*, in Functional Differential Equations and Approximation of Fixed Points, Lecture Notes in Math. #730, H. O. Peitgen and H. O. Walther, Eds., Springer-Verlag, 1979, 228–237.

[5] F. J. Drexler, *Eine Methode zur Berechung sämtlicher Lösunger von Polynomgleichungessystemen*, Numer. Math., 29(1977), 45–58.

[6] ——, *A homotopy method for the calculation of zeros of zero-dimensional polynomial ideals*, In Continuation Methods, H. G. Wacker (Ed.), Academic Press, NY, 1978, 69–93.

[7] G. Fischer, *Complex Analytic Geometry*, Lecture Notes in Mathematics No. 538, Springer-Verlag, New York, 1977.

[8] W. Fulton, *Intersection Theory*, Springer-Verlag, New York, 1984.

[9] C. B. Garcia and T. Y. Li, *On the number of solutions to polynomial systems of equations*, SIAM J. Numer. Anal., 17(1980), 540–546.

[10] C. B. Garcia and W. I. Zangwill, *Global continuation methods for finding all solutions to polynomial systems of equations in N variables*, Center for Math. Studies in Business and Economics Report No. 7755, University of Chicago, 1977.

[11] ——, *Finding all solutions to polynomial systems and other systems of equations*, Math. Programming, 16 (1979), 159–176.

[12] ——, *An approach to homotopy and degree theory*, Math. Oper. Res. 4(1979), 390–405.

[13] ——, *Determining all solutions to certain systems of nonlinear equations*, Math. Oper. Res. 4(1979), 1–14.

[14] ——, *Pathways to Solutions, Fixed Points, and Equilibria*, Prentice-Hall, Englewood Cliffs, NJ, 1981.

[15] P. Griffiths and J. Harris, *Principles of Algebraic Geometry*, John Wiley and Sons, New York, 1978.

[16] V. Guillemin and A. Pollack, *Differential Topology*, Prentice-Hall, Englewood Cliffs, NJ, 1981.

[17] M.A. Jenkins and J.F. Traub, *A three-stage variable-shift iteration for polynomial zeros and its relation to generalized Rayleigh iteration*, Numer. Math., 14 (1970), 252-263.

[18] A. Griewank, *On solving nonlinear equations with simple singularities or nearly singular solutions*, SIAM Rev., 27(1985), 537–563.

[19] T. Y. Li, T. Sauer, and J. A. Yorke, *Numerical solution of a class of deficient polynomial systems*, SIAM J. Numer. Anal., 24(1987), 435–451.

[20] K. Meintjes and A. P. Morgan, *A methodology for solving chemical equilibrium systems*, Appl. Math. Comput., 22(1987), 333–361.

[21] A. P. Morgan, *A method for computing all solutions to systems of polynomial equations*, ACM Trans. Math Software, 9(1983), 1–17.

[22] ——, *A transformation to avoid solutions at infinity for polynomial systems*, Appl. Math. Comput., 18(1986), 77–86.

[23] ——, *A homotopy for solving polynomial systems*, Appl. Math. Comput., 18(1986), 87–92.

[24] ——, *Solving Polynomial Systems Using Continuation for Scientific and Engineering Problems*, Prentice-Hall, Englewood Cliffs, NJ, 1987.

[25] A. P. Morgan and A. J. Sommese, *A homotopy for solving general polynomial systems that respects m-homogeneous structures*, Appl. Math. Comput., 24(1987), 101–113.

[26] ——, *Computing all solutions to polynomial systems using homotopy continuation*, Appl. Math. Compu., 24(1987), 115–138.

[27] A. P. Morgan and R. F. Sarraga, *A method for computing three surface intersection points in GMSOLID*, ASME paper 82-DET-41 (1982). Also available as Res. Pub. GMR-3964, Mathematics Dept., G.M. Research Labs., Warren, MI 49090, 1981.

[28] S. Richter and R. De Carlo, *A homotopy method for eigenvalue assignment using decentralized state feedback*, IEEE Trans. Auto. Control, AC-29(1984), 148–158.

[29] W.C. Rheinboldt and J.V. Burkardt, *Algorithm 596: A program for a locally parameterized continuation process*, ACM Trans. Math. Software, 9(1983), 236–241.

[30] M. G. Safonov, *Exact calculation of the multivariable structured-singular-value stability margin*, IEEE Control and Decision Conference, Las Vegas, NV, Dec. 12-14, 1984.

[31] I. R. Shafarevich, *Basic Algebraic Geometry*, Springer-Verlag, New York, 1977.

[32] L.-W. Tsai and A. P. Morgan, *Solving the kinematics of the most general six- and five-degree-of-freedom manipulators by continuation methods*, ASME J. of Mechanisms, Transmissions and Automation in Design, 107(1985), 48–57.

[33] L. T. Watson, *A globally convergent algorithm for computing fixed points of C^2 maps*, Appl. Math. Comput., 5(1979), 297–311.

[34] ——, *Numerical linear algebra aspects of globally convergent homotopy methods*, SIAM Review, 28(1986), 529–545.

[35] L. T. Watson, S. C. Billups, and A. P. Morgan, *HOMPACK: A suite of codes for globally convergent homotopy algorithms*, ACM Trans. Math. Software, to appear.

[36] L. T. Watson and D. Fenner, *Chow-Yorke algorithm for fixed points or zeros of C^2 maps*, ACM Trans. Math. Software, 6(1980), 252–260).

[37] B. L. van der Waerden, *Modern Algebra*, Vols. 1, 2, Frederick Ungar Pub. Co., New York, 1953.

Table 1. Notation for HOMPACK parameters and nominal (first run) values.

Notation in this paper	In words	Variables			First Run Values
		QF	NF	DF	
MAXSS	max step size	SSPAR(2)	SSPAR(5)	–	1.0
START	start step size	SSPAR(5)	–	–	0.1
ARC	beg path tolerance	ARCRE	ARCRE	ARCRE	10^{-5}
ANS	end path tolerance	ANSRE	ANSRE	ANSRE	10^{-10}
LIMITD	max num steps	LIMITD	LIMITD	LIMITD	1000

Table 2. Coefficients for generic g^0.

Index j	Real Part of p_j	Imaginary Part of p_j
1	0.12324754231	0.76253746298
2	0.93857838950	−0.99375892810
3	−0.23467908356	0.39383930009
4	0.83542556622	−0.10192888288
5	−0.55763522521	−0.83729899911
6	−0.78348738738	−0.10578234903
7	0.03938347346	0.04825184716
8	−0.43428734331	0.93836289418

Index j	Real Part of q_j	Imaginary Part of q_j
1	0.58720452864	0.01321964722
2	0.97884134700	−0.14433009712
3	0.39383737289	0.41543223411
4	−0.03938376373	−0.61253112318
5	0.39383737388	−0.26454678861
6	−0.00938376766	0.34447867861
7	−0.04837366632	0.48252736790
8	0.93725237347	−0.54356527623

Table 3. Coefficients for projective transformation.

Index j	Real Part of β_j	Imaginary Part of β_j
1	−0.03485644332	0.28554634336
2	0.91453454766	0.35354566613
3	−0.36568737635	0.45634642477
4	−0.89089767544	0.34524523544
5	0.13523462465	0.43534535555
6	−0.34523544445	0.00734522256
7	−0.80004678763	−0.00938712364
8	−0.87543212424	0.00045687651
9	0.65256352333	−0.12356777452
10	0.09986798322	−0.56753456577

Table 4. Coefficients for f in problem 1.

Index j	$a_{1,j}$	$a_{2,j}$
1	−0.00098	−0.01
2	978000.0	−0.984
3	−9.8	−29.7
4	−235.0	0.00987
5	88900.0	−0.124
6	−1.0	−0.25

Table 5. WORK and CPU minutes for Geometric Modeling Problem. SC and/or PT indicate that the SCLGNP scaling and/or the projective transformation, respectively, were used.

Special Homotopy

METHOD	SC,PT		no SC, PT	
	WORK	CPU	WORK	CPU
QF	269	0.002	310	0.002
NF	352	0.001	340	0.001
DF	975	0.003	918	0.003

Traditional Homotopy

METHOD	SC,PT		no SC, PT		SC, no PT		no SC, no PT	
	WORK	CPU	WORK	CPU	WORK	CPU	WORK	CPU
QF	771	0.005	2332	0.014	10108	0.054	51446	0.578
NF	888	0.002	2572	0.006	8316	0.021	85624	0.223
DF	1821	0.005	4734	0.011	7737	0.017	61239	0.159

Table 6. Coefficients for f in problem 2. The "D" notation indicates powers of 10. Thus 0.2000D+17 denotes 0.2000×10^{17}.

Index j	$a_{1,j}$	$a_{2,j}$
1	0.1069D−03	0.2000D+17
2	0.2000D+05	0.1000D+15
3	0.1000D+01	−0.1000D+01
4	−0.1800D−09	
5	−0.1283D−23	

Table 7. WORK and CPU minutes for Unscaled Chemical Problem.

Special Homotopy

METHOD	FULL RUNS (ALL PATHS)						RERUNS OF FAILED PATHS	
	WORK	CPU	No. paths with FLAG= 1	3	5	6	WORK	CPU
QF	340	0.005	6	0	1	2	1372	0.013
NF	332	0.003	8	0	1	0	400	0.001
DF	1713	0.007	9	0	0	0	0	0.0

Traditional Homotopy

ALL FAILED

Table 8. WORK and CPU minutes for Scaled Chemical Problem.

Special Homotopy

METHOD	FULL RUNS (ALL PATHS)						RERUNS OF FAILED PATHS	
			No. paths with FLAG=					
	WORK	CPU	1	3	5	6	WORK	CPU
QF	247	0.005	9	0	0	0	0	0.0
NF	328	0.003	9	0	0	0	0	0.0
DF	1524	0.007	9	0	0	0	0	0.0

Traditional Homotopy

METHOD	FULL RUNS (ALL PATHS)							RERUNS OF FAILED PATHS	
				No. paths with FLAG=					
	Run No.	WORK	CPU	1	3	5	6	WORK	CPU
QF	1	2225	0.018	1	0	10	1		
	2	23098	0.150	4	2	4	2		
	3	27695	0.169	8	2	1	1	1935579	11.589
NF	1	2972	0.009	4	0	8	0		
	2	6964	0.022	6	0	6	0		
	3	24008	0.079	10	2	0	0	0	0.0
DF	1	12315	0.035	12	0	0	0		
	2	15648	0.044	12	0	0	0		
	3	19143	0.053	12	0	0	0	0	0.0

Table 9. Coefficients for f in problem 3.

Index j	$a_{1,j}$	$a_{2,j}$	$a_{3,j}$	$a_{4,j}$
1	−0.24915068D+00	0.12501635D+00	−0.63555007D+00	0.14894773D+01
2	0.16091354D+01	−0.68660736D+00	−0.11571992D+00	0.23062341D+00
3	0.27942343D+00	−0.11922812D+00	−0.66640448D+00	0.13281073D+01
4	0.14348016D+01	−0.71994047D+00	0.11036211D+00	−0.25864503D+00
5	0.00000000D+00	−0.43241927D+00	0.29070203D+00	0.11651720D+01
6	0.40026384D+00	0.00000000D+00	0.12587767D+01	−0.26908494D+00
7	−0.80052768D+00	0.00000000D+00	−0.62938836D+00	0.53816987D+00
8	0.00000000D+00	−0.86483855D+00	0.58140406D+00	0.58258598D+00
9	0.74052388D−01	−0.37157270D−01	0.19594662D+00	−0.20816985D+00
10	−0.83050031D−01	0.35436896D−01	−0.12280342D+01	0.26868320D+01
11	−0.38615961D+00	0.85383482D−01	0.00000000D+00	−0.69910317D+00
12	−0.75526603D+00	0.00000000D+00	−0.79034221D−01	0.35744413D+00
13	0.50420168D+00	−0.39251967D−01	0.26387877D−01	0.12499117D+01
14	−0.10916287D+01	0.00000000D+00	−0.57131430D−01	0.14677360D+01
15	0.00000000D+00	−0.43241927D+00	−0.11628081D+01	0.11651720D+01
16	0.40026384D+00	0.00000000D+00	0.12587767D+01	0.10763397D+01
17	0.49207290D−01	0.13873010D−01	0.21625750D+01	−0.69686809D+00

Table 10. Coefficients for g in problem 3.

Index j	$a_{1,j}$	$a_{2,j}$	$a_{3,j}$	$a_{4,j}$
1	−0.61255849D−01	0.34078258D−01	−0.12694273D+01	0.21210809D+01
2	0.26076366D+00	−0.15606219D+00	−0.27719637D+00	0.49826195D+00
3	0.45281134D−01	−0.27099914D−01	−0.15963103D+01	0.28693762D+01
4	0.35275837D+00	−0.19624886D+00	0.22043375D+00	−0.36832183D+00
5	0.00000000D+00	0.22073862D+00	−0.11617822D+00	0.17313019D+00
6	−0.14746307D+00	0.00000000D+00	0.25915994D+00	0.77612143D−01
7	−0.56975356D+00	0.00000000D+00	0.67075527D−01	0.29987029D+00
8	0.00000000D+00	−0.85286853D+00	0.44887817D+00	−0.44809391D−01
9	−0.12965114D+01	0.72128377D+00	−0.45191642D−01	0.87146435D−03
10	0.95839839D+00	−0.57358356D+00	−0.12669584D+01	0.22180094D+01
11	−0.27565432D+00	−0.63198845D−01	0.00000000D+00	−0.22160665D+01
12	−0.27244110D+00	0.00000000D+00	0.51406650D+00	0.22422032D+01
13	0.88054348D+00	−0.14525953D+00	0.76452385D−01	−0.11718355D+00
14	−0.11395071D+01	0.00000000D+00	−0.44139805D−01	0.86731007D+00
15	0.00000000D+00	−0.47562562D+00	−0.15019756D+01	0.86565097D+00
16	0.31773876D+00	0.00000000D+00	0.12957997D+01	0.10033856D+01
17	−0.50218200D−01	0.19116983D−01	0.11530545D+01	−0.15033239D+01

Table 11. WORK and CPU minutes for the Mechanism Problem.

Special Homotopy

METHOD	FULL RUNS (ALL PATHS)						RERUNS OF FAILED PATHS	
			No. paths with FLAG=					
	WORK	CPU	1	3	5	6	WORK	CPU
QF	33878	0.937	62	0	2	0	1397	0.044
NF	54590	0.474	62	0	2	0	3320	0.027
DF	134775	0.866	64	0	0	0	4986	0.035

Traditional Homotopy

METHOD	FULL RUNS (ALL PATHS)						RERUNS OF FAILED PATHS	
			No. paths with FLAG=					
	WORK	CPU	1	3	5	6	WORK	CPU
QF	7515514	202.0	29	2	0	1	2342	0.064
NF	7921580	63.0	30	1	0	1	2300	0.025
DF	3426516	21.0	32	0	0	0	0	0.0