Globally Convergent Homotopy Methods: A Tutorial

Layne T. Watson

TR 87-13

GLOBALLY CONVERGENT HOMOTOPY METHODS: A TUTORIAL

Layne T. Watson
Departments of Electrical Engineering and Computer Science,
Industrial and Operations Engineering, and Mathematics
University of Michigan
Ann Arbor, MI 48109 USA

Abstract. The basic theory for probability one globally convergent homotopy algorithms was developed in 1976, and since then the theory, algorithms, and applications have considerably expanded. These are algorithms for solving nonlinear systems of (algebraic) equations, which are convergent for almost all choices of starting point. Thus they are globally convergent with probability one. They are applicable to Brouwer fixed point problems, certain classes of zero-finding problems, unconstrained optimization, linearly constrained optimization, nonlinear complementarity, and the discretizations of nonlinear two-point boundary value problems based on shooting, finite differences, collocation, and finite elements. A mathematical software package, HOMPACK, exists that implements several different strategies and handles both dense and sparse problems. Homotopy algorithms are closely related to ODE algorithms, and make heavy use of ODE techniques. Homotopy algorithms for some classes of nonlinear systems, such as polynomial systems, exhibit a large amount of coarse grain parallelism. These and other topics are discussed in a tutorial fashion.

1. An example.

Consider a thin incompressible elastic rod clamped at the origin and acted on by forces Q, P and torque M (see Figure 1). The governing nondimensionalized equations are

$$\frac{dx}{ds} = \cos\theta, \quad \frac{dy}{ds} = \sin\theta, \quad \frac{d\theta}{ds} = Qx - Py + M, \tag{1.1}$$

$$x(0) = y(0) = \theta(0) = 0, \tag{1.2}$$

$$x(1) = a, \quad y(1) = b, \quad \theta(1) = c. \tag{1.3}$$

The cantilever beam problem, which is to find the position (a, b) of the tip of the rod given the forces $Q \neq 0$ and $P = 0$, has a closed-form solution in terms of elliptic integrals. The inverse problem, where the a, b, c are specified and Q, P, M are to be determined, has no similar closed-form solution. This inverse problem is ferociously nonlinear and extremely difficult. For large deformations, $c = 6\pi$ for example, the rod is wound like a coil spring and its shape is very sensitive to small perturbations in Q, P, or M.

Let

$$v = \begin{pmatrix} Q \\ P \\ M \end{pmatrix} \tag{1.4}$$

and $x(s;v)$, $y(s;v)$, $\theta(s;v)$ be the solution (dependent on v) of the initial value problem (1.1)–(1.2). Then an equivalent formulation of the nonlinear two-point boundary value problem (1.1)–(1.3) is to find a vector v such that

$$F(v) = \begin{pmatrix} x(1;v) - a \\ y(1;v) - b \\ \theta(1;v) - c \end{pmatrix} = 0. \tag{1.5}$$

*This report was completed while Dr. Watson was on sabbatical at the University of Michigan. He is currently on the faculty of the Computer Science Dept. at Virginia Tech.

GLOBALLY CONVERGENT HOMOTOPY METHODS: A TUTORIAL

Layne T. Watson
Departments of Electrical Engineering and Computer Science,
Industrial and Operations Engineering, and Mathematics
University of Michigan
Ann Arbor, MI 48109 USA

Abstract. The basic theory for probability one globally convergent homotopy algorithms was developed in 1976, and since then the theory, algorithms, and applications have considerably expanded. These are algorithms for solving nonlinear systems of (algebraic) equations, which are convergent for almost all choices of starting point. Thus they are globally convergent with probability one. They are applicable to Brouwer fixed point problems, certain classes of zero-finding problems, unconstrained optimization, linearly constrained optimization, nonlinear complementarity, and the discretizations of nonlinear two-point boundary value problems based on shooting, finite differences, collocation, and finite elements. A mathematical software package, HOMPACK, exists that implements several different strategies and handles both dense and sparse problems. Homotopy algorithms are closely related to ODE algorithms, and make heavy use of ODE techniques. Homotopy algorithms for some classes of nonlinear systems, such as polynomial systems, exhibit a large amount of coarse grain parallelism. These and other topics are discussed in a tutorial fashion.

1. An example. Consider a thin incompressible elastic rod clamped at the origin and acted on by forces Q, P and torque M (see Figure 1). The governing nondimensionalized equations are

$$\frac{dx}{ds} = \cos\theta, \quad \frac{dy}{ds} = \sin\theta, \quad \frac{d\theta}{ds} = Qx - Py + M, \tag{1.1}$$

$$x(0) = y(0) = \theta(0) = 0, \tag{1.2}$$

$$x(1) = a, \quad y(1) = b, \quad \theta(1) = c. \tag{1.3}$$

The cantilever beam problem, which is to find the position (a, b) of the tip of the rod given the forces $Q \neq 0$ and $P = 0$, has a closed-form solution in terms of elliptic integrals. The inverse problem, where the a, b, c are specified and Q, P, M are to be determined, has no similar closed-form solution. This inverse problem is ferociously nonlinear and extremely difficult. For large deformations, $c = 6\pi$ for example, the rod is wound like a coil spring and its shape is very sensitive to small perturbations in Q, P, or M.

Let

$$v = \begin{pmatrix} Q \\ P \\ M \end{pmatrix} \tag{1.4}$$

and $x(s; v)$, $y(s; v)$, $\theta(s; v)$ be the solution (dependent on v) of the initial value problem (1.1)–(1.2). Then an equivalent formulation of the nonlinear two-point boundary value problem (1.1)–(1.3) is to find a vector v such that

$$F(v) = \begin{pmatrix} x(1; v) - a \\ y(1; v) - b \\ \theta(1; v) - c \end{pmatrix} = 0. \tag{1.5}$$

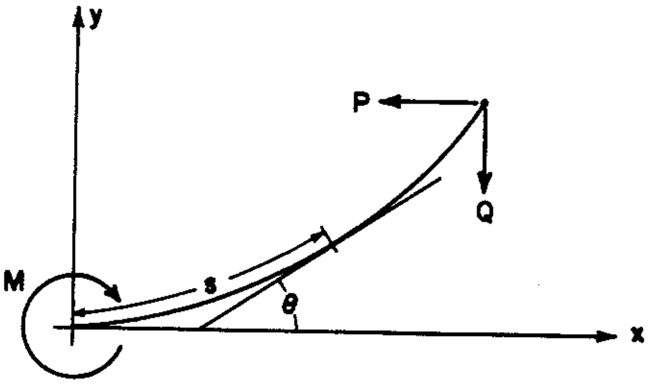

Figure 1. Elastic rod.

This particular formulation is based on shooting. Nonlinear systems different from (1.5) could be derived based on multiple shooting, finite differences, polynomial or spline collocation, spectral methods, or Galerkin methods. The best way to approximate the solution to (1.1)–(1.3) is not the issue here. The issue is how to solve the particular given nonlinear system of equations (1.5).

Newton and quasi-Newton methods, even the very best such as HYBRJ from Argonne's MINPACK package [32], fail dismally when applied to (1.5). A simple continuation scheme, such as tracking the zeros of

$$\rho_w(\lambda, v) = \lambda F(v) + (1-\lambda)(v-w) \tag{1.6}$$

as λ is increased from 0 to 1, also fails. The zero curve of $\rho_w(\lambda, v)$ in (1.6) emanating from $(0, w)$ diverges to infinity. In fact this divergence also occurs if $F(v)$ in (1.6) is replaced by $DF(v)$ for any diagonal orthogonal matrix D. Nonlinear least squares algorithms attempting to minimize $F(v)^t F(v)$ also quickly fail, since there are too many local minima $\tilde{v}$ that are not global minima $(F(\tilde{v}) \neq 0)$.

Consider the function (known as a *homotopy map*) $\rho : E^3 \times [0,1) \times E^3 \to E^3$ defined by

$$\rho(d, \lambda, v) = \rho_d(\lambda, v) = \begin{pmatrix} x(1;v) - \left[\lambda a + (1-\lambda)d_1\right] \\ y(1;v) - \left[\lambda b + (1-\lambda)d_2\right] \\ \theta(1;v) - \left[\lambda c + (1-\lambda)d_3\right] \end{pmatrix}. \tag{1.7}$$

ρ_d is a ***homotopy*** (the technical term from topology) because it continuously deforms one function (in this case $\rho_d(0, v)$) to another function (in this case $\rho_d(1, v) = F(v)$). Note that $\rho(d, \lambda, v)$ is C^2 and that its Jacobian matrix

$$D\rho(d, \lambda, v) = \left[-(1-\lambda)I,\ -(a,b,c)^t + d,\ D_v F(v)\right] \tag{1.8}$$

has full rank (rank $= 3$) on $\rho^{-1}(0)$. In differential geometry jargon, ρ is said to be ***transversal to zero***. The mathematics then says that for almost all vectors $d \in E^3$ (in the sense of Lebesgue measure), the map ρ_d is also transversal to zero. What this means geometrically is that the zero set of ρ_d consists of smooth disjoint curves that do not intersect themselves or bifurcate, and have endpoints only at $\lambda = 0$ or $\lambda = 1$ (see Figure 2). In general under suitable conditions (described in

Section 2) there is a zero curve γ of $\rho_d(\lambda, v)$ stretching from a known solution v_0 at $\lambda = 0$ to the desired solution $\bar{v}$ at $\lambda = 1$.

For the homotopy map (1.7) such a zero curve γ does indeed exist, and a solution $\bar{v}$ to (1.5) can be found by tracking γ starting from $(0, v_0)$. v_0 and d are related by $x(1; v_0) = d_1$, $y(1; v_0) = d_2$, $\theta(1; v_0) = d_3$. Since the theory says that everything works for almost all d, by the implicit function theorem, everything also works for almost all v_0. Thus in practice v_0 was chosen at random and d computed from v_0, rather than vice versa. More details on this example can be found in [59].

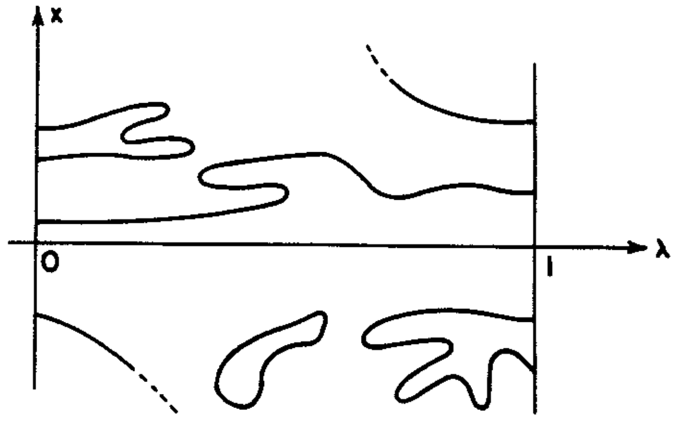

Figure 2. The zero set $\rho_d^{-1}(0)$.

The scheme just described is known as a probability one globally convergent homotopy algorithm. The phrase "probability one" refers to the almost any choice for d, and the "global convergence" refers to the fact that the starting point v_0 need not be anywhere near the solution $\bar{v}$.

Section 2 describes the mathematical theory behind globally convergent homotopy algorithms, and states some typical convergence theorems for various types of problems. Section 3 describes several strategies for tracking the zero curve γ. This is a challenging numerical analysis problem, and has been the focus of much recent research, especially on large sparse problems. The mathematical software package HOMPACK is the topic of Section 4, and Section 5 briefly mentions some of the application areas of homotopy methods. Section 6 presents some current (and incomplete) work on parallel computation.

2. Theory.

The theoretical foundation of all probability one globally convergent homotopy methods is given in the following differential geometry theorem:

Definition. Let E^n denote n-dimensional real Euclidean space, let $U \subset E^m$ and $V \subset E^n$ be open sets, and let $\rho : U \times [0,1) \times V \to E^n$ be a C^2 map. ρ is said to be transversal to zero if the Jacobian matrix $D\rho$ has full rank on $\rho^{-1}(0)$.

Parametrized Sard's Theorem. If $\rho(a, \lambda, x)$ is transversal to zero, then for almost all $a \in U$ the map

$$\rho_a(\lambda, x) = \rho(a, \lambda, x)$$

is also transversal to zero; i.e., with probability one the Jacobian matrix $D\rho_a(\lambda, x)$ has full rank on $\rho_a^{-1}(0)$.

A typical zero set $\rho_a^{-1}(0)$ is shown in Figure 2. The recipe for constructing a globally convergent homotopy algorithm to solve the nonlinear system of equations

$$F(x) = 0, \tag{2.1}$$

where $F : E^n \to E^n$ is a C^2 map, is as follows: For an open set $U \subset E^m$ construct a C^2 homotopy map $\rho : U \times [0, 1) \times E^n \to E^n$ such that

1) $\rho(a, \lambda, x)$ is transversal to zero,
2) $\rho_a(0, x) = \rho(a, 0, x) = 0$ is trivial to solve and has a unique solution x_0,
3) $\rho_a(1, x) = F(x)$,
4) $\rho_a^{-1}(0)$ is bounded.

Then for almost all $a \in U$ there exists a zero curve γ of ρ_a, along which the Jacobian matrix $D\rho_a$ has rank n, emanating from $(0, x_0)$ and reaching a zero $\bar{x}$ of F at $\lambda = 1$. This zero curve γ does not intersect itself, is disjoint from any other zeros of ρ_a, and has finite arc length in every compact subset of $[0, 1) \times E^n$. Furthermore, if $DF(\bar{x})$ is nonsingular, then γ has finite arc length.

The general idea of the algorithm is now apparent: just follow the zero curve γ emanating from $(0, a)$ until a zero $\bar{x}$ of $F(x)$ is reached (at $\lambda = 1$). Of course it is nontrivial to develop a viable numerical algorithm based on that idea, but at least conceptually, the algorithm for solving the nonlinear system of equations $F(x) = 0$ is clear and simple. The homotopy map (usually, but not always) is

$$\rho_a(\lambda, x) = \lambda F(x) + (1 - \lambda)(x - a), \tag{2.2}$$

which has the same form as a standard continuation or embedding mapping. However, there are two crucial differences. In standard continuation, the embedding parameter λ increases monotonically from 0 to 1 as the trivial problem $x - a = 0$ is continuously deformed to the problem $F(x) = 0$. The present homotopy method permits λ to both increase and decrease along γ with no adverse effect; that is, turning points present no special difficulty. The second important difference is that there are never any "singular points" which afflict standard continuation methods. The way in which the zero curve γ of ρ_a is followed and the full rank of $D\rho_a$ along γ guarantee this.

It should be mentioned that the form of the homotopy map $\rho_a(\lambda, x)$ in (2.2) is just a special case used here for clarity of exposition. The more general theory can be found in [69, 73–75, 82], and practical engineering problems requiring a ρ_a nonlinear in λ are in [59] and [66].

The following sections give some typical theorems for various classes of problems.

2.1. Brouwer fixed points. This area represents one of the first successes for both simplicial [4, 13–14, 30, 41–43, 45] and continuous homotopy methods [8, 48]. Brouwer fixed point problems can be very nasty, and often cause locally convergent iterative methods a great deal of difficulty.

Theorem. Let $B = \{x \in E^n \mid \|x\|_2 = 1\}$ be the closed unit ball, and $f : B \to B$ a C^2 map. Then for almost all $a \in \operatorname{int} B$ there exists a zero curve γ of

$$\rho_a(\lambda, x) = \lambda(x - f(x)) + (1 - \lambda)(x - a),$$

along which the Jacobian matrix $D\rho_a(\lambda, x)$ has full rank, emanating from $(0, a)$ and reaching a fixed point $\bar{x}$ of f at $\lambda = 1$. Furthermore, γ has finite arc length if $I - Df(\bar{x})$ is nonsingular.

2.2. General zero finding problems. Typically a problem (such as a partial differential equation) reduces to a finite dimensional nonlinear system of equations, and what is desired are conditions on the original problem, not on the final discretized problem. Thus the results in this section are used to derive, working backwards, useful conditions on the original problem, whatever it might be. The following four lemmas from [74], [74], [73], [75] respectively, which follow from the results of [8], are used for that purpose.

Lemma 1. Let $g : E^p \to E^p$ be a C^2 map, $a \in E^p$, and define $\rho_a : [0,1) \times E^p \to E^p$ by

$$\rho_a(\lambda, y) = \lambda g(y) + (1 - \lambda)(y - a).$$

Then for almost all $a \in E^p$ there is a zero curve γ of ρ_a emanating from $(0, a)$ along which the Jacobian matrix $D\rho_a(\lambda, y)$ has full rank.

Lemma 2. If the zero curve γ in Lemma 1 is bounded, it has an accumulation point $(1, \bar{y})$, where $g(\bar{y}) = 0$. Furthermore, if $Dg(\bar{y})$ is nonsingular, then γ has finite arc length.

Lemma 3. Let $F : E^p \to E^p$ be a C^2 map such that for some $r > 0$, $x\,F(x) \geq 0$ whenever $\|x\| = r$. Then F has a zero in $\{x \in E^p \mid \|x\| \leq r\}$, and for almost all $a \in E^p$, $\|a\| < r$, there is a zero curve γ of

$$\rho_a(\lambda, x) = \lambda F(x) + (1 - \lambda)(x - a),$$

along which the Jacobian matrix $D\rho_a(\lambda, x)$ has full rank, emanating from $(0, a)$ and reaching a zero $\bar{x}$ of F at $\lambda = 1$. Furthermore, γ has finite arc length if $DF(\bar{x})$ is nonsingular.

Lemma 3 is a special case of the following more general lemma.

Lemma 4. Let $F : E^p \to E^p$ be a C^2 map such that for some $r > 0$ and $\tilde{r} > 0$, $F(x)$ and $x - a$ do not point in opposite directions for $\|x\| = r$, $\|a\| < \tilde{r}$. Then F has a zero in $\{x \in E^p \mid \|x\| \leq r\}$, and for almost all $a \in E^p$, $\|a\| < \tilde{r}$, there is a zero curve γ of

$$\rho_a(\lambda, x) = \lambda F(x) + (1 - \lambda)(x - a),$$

along which the Jacobian matrix $D\rho_a(\lambda, x)$ has full rank, emanating from $(0, a)$ and reaching a zero $\bar{x}$ of F at $\lambda = 1$. Furthermore, γ has finite arc length if $DF(\bar{x})$ is nonsingular.

2.3. Optimization. Consider first the unconstrained optimization problem

$$\min_x f(x). \tag{2.3}$$

Theorem. Let $f : E^n \to E$ be a C^3 convex map with a minimum at $\tilde{x}$, $\|\tilde{x}\|_2 \leq M$. Then for almost all a, $\|a\|_2 < M$, there exists a zero curve γ of the homotopy map

$$\rho_a(\lambda, x) = \lambda\, \nabla f(x) + (1 - \lambda)(x - a),$$

along which the Jacobian matrix $D\rho_a(\lambda, x)$ has full rank, emanating from $(0, a)$ and reaching a point $(1, \bar{x})$, where $\bar{x}$ solves (2.3).

A function is called uniformly convex if it is convex and its Hessian's smallest eigenvalue is bounded away from zero. Consider next the constrained optimization problem

$$\min_{x\geq 0} f(x). \tag{2.4}$$

This is more general than it might appear because the general convex quadratic program reduces to a problem of the form (2.4).

Theorem. Let $f : E^n \to E$ be a C^3 uniformly convex map. Then there exists $\delta > 0$ such that for almost all $a \geq 0$ with $\|a\|_2 < \delta$ there exists a zero curve γ of the homotopy map

$$\rho_a(\lambda, x) = \lambda\, K(x) + (1-\lambda)(x-a),$$

where

$$K_i(x) = -\left|\frac{\partial f(x)}{\partial x_i} - x_i\right|^3 + \left(\frac{\partial f(x)}{\partial x_i}\right)^3 + x_i^3,$$

along which the Jacobian matrix $D\rho_a(\lambda, x)$ has full rank, connecting $(0, a)$ to a point $(1, \bar{x})$, where $\bar{x}$ solves the constrained optimization problem (2.4).

2.4. Nonlinear complementarity. Given $F : E^n \to E^n$, the nonlinear complementarity problem is to find a vector $x \in E^n$ such that

$$x \geq 0, \quad F(x) \geq 0, \quad x^t F(x) = 0. \tag{2.5}$$

At a solution $\bar{x}$, $\bar{x}$ and $F(\bar{x})$ are "complementary" in the sense that if $\bar{x}_i > 0$, then $F_i(\bar{x}) = 0$, and if $F_i(\bar{x}) > 0$, then $\bar{x}_i = 0$. This problem is difficult because there are linear constraints $x \geq 0$, nonlinear constraints $F(x) \geq 0$, and a combinatorial aspect from the complementarity condition $x^t F(x) = 0$. It is interesting that homotopy methods can be adapted to deal with nonlinear constraints and combinatorial conditions.

Define $G : E^n \to E^n$ by

$$G_i(z) = -\left|F_i(z) - z_i\right|^3 + \left(F_i(z)\right)^3 + z_i^3, \quad i = 1, \ldots, n, \tag{2.6}$$

and let

$$\rho_a(\lambda, z) = \lambda\, G(z) + (1-\lambda)(z-a).$$

Theorem. Let $F : E^n \to E^n$ be a C^2 map, and let the Jacobian matrix $DG(z)$ be nonsingular at every zero of $G(z)$. Suppose there exists $r > 0$ such that $z > 0$ and $z_k = \|z\|_\infty \geq r$ imply $F_k(z) > 0$. Then for almost all $a > 0$ there exists a zero curve γ of $\rho_a(\lambda, z)$, along which the Jacobian matrix $D\rho_a(\lambda, z)$ has full rank, having finite arc length and connecting $(0, a)$ to $(1, \bar{z})$, where $\bar{z}$ solves (2.5).

Theorem. Let $F : E^n \to E^n$ be a C^2 map, and let the Jacobian matrix $DG(z)$ be nonsingular at every zero of $G(z)$. Suppose there exists $r > 0$ such that $z \geq 0$ and $\|z\|_\infty \geq r$ imply $z_k F_k(z) > 0$ for some index k. Then there exists $\delta > 0$ such that for almost all $a \geq 0$ with $\|a\|_\infty < \delta$ there exists a zero curve γ of $\rho_a(\lambda, z)$, along which the Jacobian matrix $D\rho_a(\lambda, z)$ has full rank, having finite arc length and connecting $(0, a)$ to $(1, \bar{z})$, where $\bar{z}$ solves (2.5).

2.5. Two-point boundary value problems: shooting. The next few sections consider nonlinear two-point boundary value problems of the form

$$y''(t) = g\big(t, y(t), y'(t)\big), \quad 0 \le t \le 1, \tag{2.7}$$

$$y(0) = 0, \qquad y(1) = 0 \quad (\text{or } y'(1) = 0), \tag{2.8}$$

where $y = (y_1, \ldots, y_n)$, $g(t, u, v)$ satisfies a Lipschitz condition in (u, v) for $0 \le t \le 1$, and has continuous second partials with respect to u and v for $0 \le t \le 1$. These technical assumptions vary slightly from theorem to theorem and method to method; consult the references for complete and precise statements of the theorems. The intent here is to provide the flavor of applying homotopy methods to approximations to nonlinear two-point boundary value problems, and not get bogged down in technical detail.

Let $u(t)$ be the unique solution of

$$y''(t) = g\big(t, y(t), y'(t)\big), \quad 0 \le t \le 1,$$

$$y(0) = 0, \quad y'(0) = v,$$

and define $f : E^n \to E^n$ by

$$f(v) = u(1) \quad (\text{or } u'(1)\,). \tag{2.9}$$

Observe that the original problem (2.7)–(2.8) is equivalent to solving

$$f(v) = 0 \tag{2.10}$$

for the correct initial condition $\bar{v}$.

Theorem. Suppose there exists a constant $M > 0$ such that $\big\| g\big(t, x(t), x'(t)\big) \big\|_\infty \le M$ along every trajectory $x(t)$ for which $x(0) = 0$, $\|x'(0)\|_\infty = M$. Then for almost all $w \in E^n$ with $\|w\|_\infty < M$ there exists a zero curve γ of the homotopy map

$$\rho_w(\lambda, v) = \lambda\, f(v) + (1 - \lambda)(v - w),$$

along which $D\rho_w$ has full rank, lying in $[0,1] \times \big\{ v \in E^n \mid \|v\|_\infty \le M \big\}$ and connecting $(0, w)$ to $(1, \bar{v})$, where $\bar{v}$ is a zero of f. If $Df(\bar{v})$ is nonsingular, then γ has finite arc length.

2.6. Two-point boundary value problems: finite differences. The problem under consideration is (2.7)–(2.8). Using standard centered second order accurate finite difference approximations for y' and y'' in (2.7) with the boundary conditions (2.8) at mesh points $0 = t_0 < t_1 < \cdots < t_N < t_{N+1} = 1$ results in the finite difference approximation

$$G(Y) = A\,Y + h^2 F^h(Y) = 0, \tag{2.11}$$

where $h = \max_i (t_{i+1} - t_i)$ is the mesh size, A is a constant positive definite matrix, and $F^h(Y)$ is the nonlinear part of $G(Y)$ due to g.

Theorem. Let $F^h(Y)$ be a C^2 mapping, and suppose that

$$\limsup_{\|Y\|_2\to\infty} \frac{\|F^h(Y)\|_2}{\|Y\|_2} \le 9.$$

Then for almost $W \in E^N$ there is a zero curve γ of the homotopy map

$$\rho_W(\lambda, Y) = \lambda\, G(Y) + (1-\lambda)(Y-W),$$

along which the Jacobian matrix $D\rho_W(\lambda, Y)$ has full rank, emanating from $(0, W)$ and reaching a zero $\tilde{Y}$ of G at $\lambda = 1$. Furthermore, γ has finite arc length if $DG(\tilde{Y})$ is nonsingular.

More general boundary conditions than (2.8) have the form

$$By(0) + B'y'(0) + Cy(1) + C'y'(1) = b, \tag{2.12}$$

where rank $(B\ B'\ C\ C') = 2n$, and there are other technical restrictions (see Keller [23]). These boundary conditions lead to a different nonlinear system of equations

$$G(Y) = A\,Y + h^2 F^h(Y) = 0, \tag{2.13}$$

where A and F^h are different from those in (2.11).

Theorem. Let $F^h(Y)$ be a C^2 mapping, and suppose that G from (2.13) satisfies one of the following:

1) there exist $r > 0$ and $\tilde{r} > 0$ such that $Y - W$ and $G(Y)$ do not point in opposite directions for $\|Y\|_2 = r$, $\|W\|_2 < r$;
2) there exists $r > 0$ such that $Y^t G(Y) \ge 0$ for $\|Y\|_2 = r$;
3) A is positive semidefinite, and there exists $r > 0$ such that $Y^t F^h(Y) \ge 0$ for $\|Y\|_2 = r$.

Then for almost all $\|W\|_2 < \tilde{r}$ there is a zero curve γ of the homotopy map

$$\rho_W(\lambda, Y) = \lambda\, G(Y) + (1-\lambda)(Y-W),$$

along which the Jacobian matrix $D\rho_W(\lambda, Y)$ has full rank, emanating from $(0, W)$ and reaching a zero $\tilde{Y}$ of G at $\lambda = 1$. Furthermore, γ has finite arc length if $DG(\tilde{Y})$ is nonsingular.

Theorem. The conclusion of the previous theorem holds if A from (2.13) is positive definite and $g(t, u, v)$ is a bounded C^2 mapping.

Theorem. The conclusion of the previous theorem also holds if $g(t, u, v)$ is a bounded C^2 mapping and the boundary conditions are of the form $y(0) = b_1$, $y(1) = b_2$.

2.7. Two-point boundary value problems: collocation. The idea here is to approximate $y(t)$ from some convenient, finite dimensional vector space S_m with basis $\varphi_1, \ldots, \varphi_m$. These basis functions could be orthogonal polynomials (a popular approach in chemical engineering) or B-splines (the numerical analysts' choice) or something tailored for a specific problem. The approximations

$$y_k(t) \approx A_k(t) = \sum_{i=1}^{m} \alpha_{ki}\varphi_i(t), \quad k = 1, \ldots, n, \tag{2.14}$$

are substituted into equations (2.7)–(2.8) evaluated at discrete points, the collocation points, in the interval $[0,1]$. This results in the nonlinear system of equations

$$F(Y) = MY + N(Y) = 0, \tag{2.15}$$

where

$$Y = (\alpha_{11}, \alpha_{12}, \ldots, \alpha_{1m}, \alpha_{21}, \ldots, \alpha_{2m}, \ldots, \alpha_{n1}, \ldots, \alpha_{nm})^t,$$

M is a constant matrix, and $N(Y)$ is the nonlinear part due to g. The dimension of the problem is $p = nm$.

Theorem. Let $N(Y)$ in (2.15) be a C^2 mapping, and suppose there exist constants C and ν such that

$$\limsup_{\|Y\|_\infty \to \infty} \frac{\|N(Y)\|_\infty}{\|Y\|_\infty^\nu} = C, \qquad 0 \le \nu < 1. \tag{2.16}$$

For $W \in E^p$, define $\rho_W : [0,1) \times E^p \to E^p$ by

$$\rho_W(\lambda, Y) = \lambda F(Y) + (1-\lambda)(Y - W).$$

Then for almost all $W \in E^p$ there exists a zero curve γ of ρ_W, along which the Jacobian matrix $D\rho_W(\lambda, Y)$ has full rank, emanating from $(0, W)$ and reaching a zero $\tilde{Y}$ of F (at $\lambda = 1$). Furthermore, if $DF(\tilde{Y})$ is nonsingular, then γ has finite arc length.

Theorem. Let $N(Y)$ in (2.15) be a C^2 mapping and the matrix M be such that

$$\min_{\|Y\|_\infty = 1} \max_{1 \le j \le p} Y_j (MY)_j = \Gamma > 0.$$

If

$$\limsup_{\|Y\|_\infty \to \infty} \frac{\|N(Y)\|_\infty}{\|Y\|_\infty} = C < \Gamma,$$

then the conclusion of the above Theorem holds.

Theorem. If $g(t,u,v)$ in (2.7) is C^2 and bounded, then the conclusion of the above Theorem holds.

Theorem. Let $g(t,u,v)$ in (2.7) be a C^2 mapping, and suppose there exist constants μ and ν such that

$$\limsup_{\|y\|_\infty \to \infty} \max_{\substack{0 \le t \le 1 \\ u \in E^n}} \frac{\|g(t,y,u)\|_\infty}{\|y\|_\infty^\nu} = \mu, \quad 0 \le \nu < 1. \tag{2.17}$$

Then the conclusion of the above Theorem holds.

2.8. Two-point boundary value problems: finite elements. The finite element (or Rayleigh-Ritz-Galerkin) approach is similar to collocation in that an approximation is sought from a finite dimensional space S_m. Here the elements of S_m automatically satisfy the boundary conditions (2.8), which collocation doesn't require. Instead of satisfying the differential equation (2.7) at discrete points, the finite element method satisfies (2.7) in an average sense by requiring certain inner product integrals to be equal. In some contexts the finite element formulation can be viewed as minimizing some functional over a finite dimensional space, where the minimum over an infinite dimensional space gives the exact solution (a variational formulation).

Using the approximations (2.14), the Galerkin approximation to (2.7)–(2.8) is

$$\int_0^1 A_j''(x)\varphi_k(x)\,dx = \int_0^1 g_j\bigl(x, A(x), A'(x)\bigr)\varphi_k(x)\,dx, \qquad k = 1, \ldots, m, \quad j = 1, \ldots, n. \tag{2.18}$$

This is a system of equations of exactly the same form as (2.15). The convergence theorems are similar to those for collocation.

Theorem. Let $N(Y)$ in (2.15) be a C^2 mapping, and suppose there exist constants C and ν such that

$$\limsup_{\|Y\|_2 \to \infty} \frac{\|N(Y)\|_2}{\|Y\|_2^\nu} = C, \qquad 0 \le \nu < 1.$$

For $W \in E^p$, define $\rho_W : [0,1) \times E^p \to E^p$ by

$$\rho_W(\lambda, Y) = \lambda F(Y) + (1-\lambda)(Y - W).$$

Then for almost all $W \in E^p$ there exists a zero curve γ of ρ_W, along which the Jacobian matrix $D\rho_W(\lambda, Y)$ has full rank, emanating from $(0, W)$ and reaching a zero $\tilde{Y}$ of F (at $\lambda = 1$). Furthermore, if $DF(\tilde{Y})$ is nonsingular, then γ has finite arc length.

Theorem. Let $N(Y)$ in (2.15) be a C^2 mapping and $\Gamma > 0$ the smallest eigenvalue of M. If

$$\limsup_{\|Y\|_2 \to \infty} \frac{\|N(Y)\|_2}{\|Y\|_2} = C < \Gamma,$$

then the conclusion of the above Theorem holds.

Theorem. If $g(t,u,v)$ in (2.7) is C^2 and bounded, then the conclusion of the above Theorem holds.

Theorem. Let $g(t,u,v)$ in (2.7) be a C^2 mapping, and suppose there exist constants μ, ξ, and ν such that

$$\|g(t,u,v)\|_2 \le \mu\bigl(\xi + (\|u\|_2 + \|v\|_2)^\nu\bigr), \quad 0 \le \nu < 1, \tag{2.19}$$

for all $0 \le t \le 1$ and $u, v \in E^n$. Then the conclusion of the above Theorem holds.

3. Algorithms.

The zero curve γ of the homotopy map $\rho_W(\lambda, Y)$ in (2.2) can be tracked by many different techniques; refer to the excellent survey [4] and recent work by Rheinboldt and Burkhardt [39], Mejia [28], and Watson [82]. There are three primary algorithmic approaches to tracking γ : 1) an ODE-based algorithm based on that in [68], with several refinements; 2) a predictor-corrector algorithm whose corrector follows the flow normal to the Davidenko flow [16–18] (a "normal flow" algorithm); 3) a version of Rheinboldt's linear predictor, quasi-Newton corrector algorithm [39] (an "augmented Jacobian matrix" method). The algorithms for dense and sparse Jacobian matrices are qualitatively different. Only algorithms for dense Jacobian matrices are discussed here; see [77] for sparse homotopy algorithms.

3.1. Ordinary differential equation based algorithm (dense Jacobian matrix).

For the sake of brevity, only the zero finding problem (2.1) with homotopy map (2.2) will be presented. Assuming that $F(x)$ is C^2, a is such that the Jacobian matrix $D\rho_a(\lambda, x)$ has full rank along γ, and

γ is bounded, the zero curve γ is C^1 and can be parametrized by arc length s. Thus $\lambda = \lambda(s)$, $x = x(s)$ along γ, and

$$\rho_a(\lambda(s), x(s)) = 0 \tag{3.1}$$

identically in s. Therefore

$$\frac{d}{ds}\rho_a(\lambda(s), x(s)) = D\rho_a(\lambda(s), x(s)) \begin{pmatrix} \dfrac{d\lambda}{ds} \\ \dfrac{dx}{ds} \end{pmatrix} = 0, \tag{3.2}$$

$$\left\| \left(\frac{d\lambda}{ds}, \frac{dx}{ds} \right) \right\|_2 = 1. \tag{3.3}$$

With the initial conditions

$$\lambda(0) = 0, \quad x(0) = a, \tag{3.4}$$

the zero curve γ is the trajectory of the initial value problem (3.2–3.4). When $\lambda(\bar{s}) = 1$, the corresponding $x(\bar{s})$ is a zero of $F(x)$. Thus all the sophisticated ordinary differential equation techniques currently available can be brought to bear on the problem of tracking γ [47], [69].

Typical ordinary differential equation software requires $(d\lambda/ds, dx/ds)$ explicitly, and (3.2), (3.3) only implicitly define the derivative $(d\lambda/ds, dx/ds)$. (It might be possible to use an implicit ordinary differential equation technique for (3.2–6), but that seems less efficient than the method proposed here.) The derivative $(d\lambda/ds, dx/ds)$, which is a unit tangent vector to the zero curve γ, can be calculated by finding the one-dimensional kernel of the $n \times (n+1)$ Jacobian matrix

$$D\rho_a(\lambda(s), x(s)),$$

which has full rank according to the theory [69]. It is here that a substantial amount of computation is incurred, and it is imperative that the number of derivative evaluations be kept small. Once the kernel has been calculated, the derivative $(d\lambda/ds, dx/ds)$ is uniquely determined by (3.3) and continuity. Complete details for solving the initial value problem (3.2–3.4) and obtaining $x(\bar{s})$ are in [68] and [69]. A discussion of the kernel computation follows.

The Jacobian matrix $D\rho_a$ is $n \times (n+1)$ with (theoretical) rank n. The crucial observation is that the last n columns of $D\rho_a$, corresponding to $D_x\rho_a$, may not have rank n, and even if they do, some other n columns may be better conditioned. The objective is to avoid choosing n "distinguished" columns, rather to treat all columns the same (not possible for sparse matrices). Kubicek [26] and Mejia [28] have kernel finding algorithms based on Gaussian elimination and n distinguished columns. Choosing and switching these n columns is tricky, and based on *ad hoc* parameters. Also, computational experience has shown that accurate tangent vectors $(d\lambda/ds, dx/ds)$ are essential, and the accuracy of Gaussian elimination may not be good enough. A conceptually elegant, as well as accurate, algorithm is to compute the QR factorization with column interchanges [7] of $D\rho_a$,

$$Q\, D\rho_a\, P^t P z = \begin{pmatrix} * & \cdots & * & * \\ & \ddots & \vdots & \vdots \\ 0 & & * & * \end{pmatrix} P z = 0,$$

where Q is a product of Householder reflections and P is a permutation matrix, and then obtain a vector $z \in \ker D\rho_a$ by back substitution. Setting $(Pz)_{n+1} = 1$ is a convenient choice. This scheme provides high accuracy, numerical stability, and a uniform treatment of all $n+1$ columns. Finally,

$$\left(\frac{d\lambda}{ds}, \frac{dx}{ds}\right) = \pm \frac{z}{\|z\|_2},$$

where the sign is chosen to maintain an acute angle with the previous tangent vector on γ. There is a rigorous mathematical criterion, based on a $(n+1) \times (n+1)$ determinant, for choosing the sign, but there is no reason to believe that would be more robust than the angle criterion.

Several features which are a combination of common sense and computational experience should be incorporated into the algorithm. Since most ordinary differential equation solvers only control the local error, the longer the arc length of the zero curve γ gets, the farther away the computed points may be from the true curve γ. Therefore when the arc length gets too long, the last computed point $(\bar\lambda, \bar x)$ is used to calculate

$$\bar a = \frac{\bar\lambda\, F(\bar x) + (1-\bar\lambda)\, \bar x}{1-\bar\lambda}. \tag{3.5}$$

Then $\rho_{\bar a}(\bar\lambda, \bar x) = 0$ exactly, and the zero curve of $\rho_{\bar a}(\lambda, x)$ is followed starting from $(\bar\lambda, \bar x)$. A rigorous justification for this strategy was given in [69].

Remember that tracking γ was merely a means to an end, namely a zero $\tilde x$ of $F(x)$. Since γ itself is of no interest (usually), one should not waste computational effort following it too closely. However, since γ is the only sure way to $\tilde x$, losing γ can be disastrous [69]. The strategy proposed in [82] estimates the curvature of each component of γ using finite differences, and reduces the tolerance used by the ordinary differential equation solver whenever the estimated curvature exceeds some threshold. The tradeoff between computational efficiency and reliability is very delicate, and a fool-proof strategy appears difficult to achieve.

In summary, the algorithm is:

1. Set $s := 0$, $y := (0, a)$, $ypold := yp := (1, 0, \ldots, 0)$, $restart :=$ false, $error :=$ initial error tolerance for the ODE solver.
2. If $y_1 < 0$ then go to 23.
3. If $s >$ some constant then
    4. $s := 0$.
    5. Compute a new vector a from (3.5). If

    $$\|\text{new } a - \text{old } a\| > 1 + \text{constant} * \|\text{old } a\|,$$

    then go to 23.
6. $ode\ error := error$.
7. If $\|yp - ypold\|_\infty >$ (last arc length step) $*$ constant, then $ode\ error := tolerance \ll error$.
8. $ypold := yp$.
9. Take a step along the trajectory of (3.2–3.4) with the ODE solver. $yp = y'(s)$ is computed for the ODE solver by 10–12:

```
    10. Find a vector $z$ in the kernel of $D\rho_a(y)$ using Householder reflections.
    11. If $z^t\, ypold < 0$, then $z := -z$.
    12. $yp := z/\|z\|$.
13. If the ODE solver returns an error code, then go to 23.
14. If $y_1 < 0.99$, then go to 2.
15. If $restart$ = true, then go to 20.
16. $restart$ := true.
17. $error$ := final accuracy desired.
18. If $y_1 \geq 1$, then set $(s, y)$ back to the previous point (where $y_1 < 1$).
19. Go to 4.
20. If $y_1 < 1$ then go to 2.
21. Obtain the zero (at $y_1 = 1$) by interpolating mesh points used by the ODE solver.
22. Normal return.
23. Error return.
```

3.2. Normal flow algorithm (dense Jacobian matrix). As the homotopy parameter vector a varies, the corresponding homotopy zero curve γ also varies. This family of zero curves is known as the Davidenko flow. The normal flow algorithm is so called because the iterates converge to the zero curve γ along the flow normal to the Davidenko flow (in an asymptotic sense).

The normal flow algorithm has three phases: prediction, correction, and step size estimation. (2.1) and (2.2) are the relevant equations here. For the prediction phase, assume that several points $P^{(1)} = (\lambda(s_1), x(s_1))$, $P^{(2)} = (\lambda(s_2), x(s_2))$ on γ with corresponding tangent vectors $(d\lambda/ds(s_1), dx/ds(s_1))$, $(d\lambda/ds(s_2), dx/ds(s_2))$ have been found, and h is an estimate of the optimal step (in arc length) to take along γ. The prediction of the next point on γ is

$$Z^{(0)} = p(s_2 + h), \tag{3.6}$$

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

$$\begin{aligned} p(s_1) &= (\lambda(s_1), x(s_1)), \quad & p'(s_1) &= (d\lambda/ds(s_1), dx/ds(s_1)), \\ p(s_2) &= (\lambda(s_2), x(s_2)), \quad & p'(s_2) &= (d\lambda/ds(s_2), dx/ds(s_2)), \end{aligned}$$

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

$$Z^{(k+1)} = Z^{(k)} - \left[D\rho_a(Z^{(k)})\right]^{\dagger} \rho_a(Z^{(k)}), \qquad k = 0, 1, \ldots \tag{3.7}$$

where $\left[D\rho_a(Z^{(k)})\right]^{\dagger}$ is the Moore-Penrose pseudoinverse of the $n \times (n+1)$ Jacobian matrix $D\rho_a$. Small perturbations of a produce small changes in the trajectory γ , and the family of trajectories γ for varying a is known as the "Davidenko flow". Geometrically, the iterates given by (3.7) return to the zero curve along the flow normal to the Davidenko flow, hence the name "normal flow algorithm".

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

$$[D\rho_a]\Delta Z = -\rho_a. \tag{3.8}$$

Fortunately ΔZ can be calculated at the same time as the kernel of $[D\rho_a]$, and with just a little more work. Normally for dense problems the kernel of $[D\rho_a]$ is found by computing a QR factorization of $[D\rho_a]$, and then using back substitution. By applying this QR factorization to $-\rho_a$ and using back substitution again, a *particular* solution v to (3.8) can be found. Let $u \neq 0$ be any vector in the kernel of $[D\rho_a]$. Then the minimum norm solution of (3.8) is

$$\Delta Z = v - \frac{v^t u}{u^t u} u. \tag{3.9}$$

Since the kernel of $[D\rho_a]$ is needed anyway for the tangent vectors, solving (3.8) only requires another $O(n^2)$ operations beyond those for the kernel. The number of iterations required for convergence of (3.7) should be kept small (say < 4) since QR factorizations of $[D\rho_a]$ are expensive. The alternative of using $[D\rho_a(Z^{(0)})]$ for several iterations, which results in linear convergence, is rarely cost effective.

When the iteration (3.7) 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 described next attempts to balance progress along γ with the effort expended on the iteration (3.7).

Define a contraction factor

$$L = \frac{\|Z^{(2)} - Z^{(1)}\|}{\|Z^{(1)} - Z^{(0)}\|}, \tag{3.10}$$

a residual factor

$$R = \frac{\|\rho_a(Z^{(1)})\|}{\|\rho_a(Z^{(0)})\|}, \tag{3.11}$$

a distance factor ($Z^* = \lim_{k\to\infty} Z^{(k)}$)

$$D = \frac{\|Z^{(1)} - Z^*\|}{\|Z^{(0)} - Z^*\|}, \tag{3.12}$$

and ideal values $\bar{L}$, $\bar{R}$, $\bar{D}$ for these three. Let h be the current step size (the distance from Z^* to the previous point found on γ), and $\bar{h}$ the "optimal" step size for the next step. The goal is to achieve

$$\frac{\bar{L}}{L} \approx \frac{\bar{R}}{R} \approx \frac{\bar{D}}{D} \approx \frac{\bar{h}^q}{h^q} \tag{3.13}$$

for some q. This leads to the choice

$$\bar{h} = \left(\min\{\bar{L}/L, \bar{R}/R, \bar{D}/D\}\right)^{1/q} h, \tag{3.14}$$

a worst case choice. To prevent chattering and unreasonable values, constants $h_{\min}$ (minimum allowed step size), $h_{\max}$ (maximum allowed step size), $B_{\min}$ (contraction factor), and $B_{\max}$ (expansion factor) are chosen, and $\bar{h}$ is taken as

$$\bar{h} = \min\left\{\max\{h_{\min}, B_{\min}h, \hat{h}\}, B_{\max}h, h_{\max}\right\}. \tag{3.15}$$

There are eight parameters in this process: $\bar{L}$, $\bar{R}$, $\bar{D}$, $h_{\min}$, $h_{\max}$, $B_{\min}$, $B_{\max}$, q. The choice of $\bar{h}$ from (3.15) can be refined further. If (3.7) converged in one iteration, then $\bar{h}$ should certainly not be smaller than h, hence set

$$\bar{h} := \max\{h, \bar{h}\} \tag{3.16}$$

if (3.7) only required one iteration.

To prevent divergence from the iteration (3.7), if (3.7) has not converged after K iterations, h is halved and a new prediction is computed. Every time h is halved the old value h_{old} is saved. Thus if (3.7) has failed to converge in K iterations sometime during this step, the new $\bar{h}$ should not be greater than the value h_{old} known to produce failure. Hence in this case

$$\bar{h} := \min\{h_{\text{old}}, \bar{h}\}. \tag{3.17}$$

Finally, if (3.7) required the maximum K iterations, the step size should not increase, so in this case set

$$\bar{h} := \min\{h, \bar{h}\}. \tag{3.18}$$

The logic in (3.16–3.18) is rarely invoked, but it does have a stabilizing effect on the algorithm.

In summary, the algorithm is:

```
1. $s := 0$, $y := (0, a)$, $h := 0.1$, $firststep :=$ true, $arcae, arcre :=$ absolute, relative error
   tolerances for tracking $\gamma$, $ansae, ansre :=$ absolute, relative error tolerances for the answer.
2. If $firststep :=$ false then
     3. Compute the predicted point $Z^{(0)}$ using (3.6).
   else
     4. Compute the predicted point $Z^{(0)}$ using a linear predictor based on $y = (0, a)$ and the
        tangent there.
5. Iterate with equation (3.7) until either
        $\|\Delta Z^{(k)}\| \le arcae + arcre \, \|Z^{(k)}\|$
        or
        4 iterations have been performed
6. If the Newton iteration (3.7) did not converge in 4 steps, then
     7. $h := h/2$.
     8. If $h$ is unreasonably small, then return with an error flag.
     9. Go to 2.
```

```
10. $firststep$ := false.
11. If $y_1 < 1$, then compute a new step size $h$ by (3.10–3.18) and go to 2.
12. Do 13.–18. some fixed number of times.
    13. Find $\bar{s}$ such that $p(\bar{s})_1 = 1$, using $yold$, $ypold$, $y$, $yp$ in (3.6).
    14. Do two iterations of (3.7) starting with $Z^{(0)} = p(\bar{s})$, ending with $Z^{(2)}$.
    15. If
            $\left|Z_1^{(2)} - 1\right| + \left\|\Delta Z^{(1)}\right\| \le ansae + ansre \left\|Z^{(1)}\right\|$,
        then return (solution has been found).
    16. If $Z_1^{(2)} \ge 1$, then
            17. $y := Z^{(2)}$, $yp :=$ tangent at $Z^{(2)}$.
        else
            18. $yold := Z^{(2)}$, $ypold :=$ tangent at $Z^{(2)}$.
19. Return with an error flag.
```

3.3. Augmented (dense) Jacobian matrix algorithm. The augmented Jacobian matrix algorithm has four major phases: prediction, correction, step size estimation, and computation of the solution at $\lambda = 1$. Again, only the zero finding case is described here. The algorithm here is based on Rheinboldt [36–39], 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 $\lambda = 1$ is used which allows the Jacobian matrix of F to be singular at the solution.

The prediction phase is exactly the same as in the normal flow algorithm. Having the points $P^{(1)} = (\lambda(s_1), x(s_1)), P^{(2)} = (\lambda(s_2), x(s_2))$ on γ with corresponding tangent vectors

$$T^{(1)} = \begin{pmatrix} \dfrac{d\lambda}{ds}(s_1) \\ \dfrac{dx}{ds}(s_1) \end{pmatrix}, \qquad T^{(2)} = \begin{pmatrix} \dfrac{d\lambda}{ds}(s_2) \\ \dfrac{dx}{ds}(s_2) \end{pmatrix},$$

the prediction $Z^{(0)}$ of the next point on γ is given by (3.6).

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

$$\begin{bmatrix} D\rho_a\left(P^{(2)}\right) \\ T^{(1)t} \end{bmatrix} z = \begin{pmatrix} 0 \\ \vdots \\ 0 \\ 1 \end{pmatrix} \qquad (3.19)$$

for z, where $D\rho_a$ is the $n \times (n+1)$ Jacobian of ρ_a. Normalizing z gives

$$T^{(2)} = \frac{z}{\|z\|}. \tag{3.20}$$

The last row of (3.19) 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 (3.19) is found by computing a QR factorization of the matrix, and then using back substitution [7].

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

$$Z^{(k+1)} = Z^{(k)} - \begin{bmatrix} A^{(k)} \\ T^{(2)\,t} \end{bmatrix}^{-1} \begin{pmatrix} \rho_a\left(Z^{(k)}\right) \\ 0 \end{pmatrix}, \qquad k = 0, 1, \ldots \tag{3.21}$$

where $A^{(k)}$ is an approximation to the Jacobian matrix $D\rho_a\left(Z^{(k)}\right)$. The last row of the matrix in (3.21) insures that the iterates lie in a hyperplane perpendicular to the tangent vector $T^{(2)}$. (3.21) is the quasi-Newton iteration for solving the augmented nonlinear system

$$\begin{pmatrix} \rho_a(y) \\ T^{(2)\,t}\left(y - Z^{(0)}\right) \end{pmatrix} = 0. \tag{3.22}$$

A corrector step $\Delta Z^{(k)}$ is the unique solution to the equation

$$\begin{bmatrix} A^{(k)} \\ T^{(2)\,t} \end{bmatrix} \Delta Z^{(k)} = \begin{pmatrix} -\rho_a\left(Z^{(k)}\right) \\ 0 \end{pmatrix}. \tag{3.23}$$

The matrix on the left side of this equation is produced by successive Broyden rank one updates [7] of the matrix in (3.19). Precisely, letting $Z^{(-1)} = P^{(2)}$, $A^{(-1)} = D\rho_a\left(P^{(2)}\right)$, and

$$M^{(k)} = \begin{bmatrix} A^{(k)} \\ T^{(2)\,t} \end{bmatrix},$$

the update formulas are

$$M^{(-1)} = \begin{bmatrix} A^{(-1)} \\ T^{(2)\,t} \end{bmatrix} = \begin{bmatrix} D\rho_a\left(P^{(2)}\right) \\ T^{(1)\,t} \end{bmatrix} + e_{n+1}\left(T^{(2)} - T^{(1)}\right)^t, \tag{3.24}$$

and

$$M^{(k+1)} = M^{(k)} + \frac{\left(\tilde{\Delta}\rho_a - M^{(k)}\Delta Z^{(k)}\right)\Delta Z^{(k)\,t}}{\Delta Z^{(k)\,t}\Delta Z^{(k)}}, \qquad k = -1, 0, \ldots \tag{3.25}$$

where

$$\tilde{\Delta}\rho_a = \begin{pmatrix} \rho_a\left(Z^{(k+1)}\right) - \rho_a\left(Z^{(k)}\right) \\ 0 \end{pmatrix}.$$

These updates can be done in QR factored form, requiring a total of $O(n^2)$ operations for each iteration in the correction process [7]. When the iteration (3.21) 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 [39]. At each point $P^{(k)}$ with tangent $T^{(k)}$ along γ, the curvature is estimated by the formula

$$\left\| w^{(k)} \right\| = \frac{2}{\Delta s_k} \left| \sin\left(\alpha_k/2\right) \right|, \tag{3.26}$$

where

$$w^{(k)} = \frac{T^{(k)} - T^{(k-1)}}{\Delta s_k}, \qquad \alpha_k = \arccos\left(T^{(k)\,t} T^{(k-1)}\right), \qquad \Delta s_k = \left\| P^{(k)} - P^{(k-1)} \right\|.$$

Intuitively, α_k represents the angle between the last two tangent vectors, and the curvature is approximated by the Euclidean norm of the difference between these two tangents divided by Δs_k.

This curvature data can be extrapolated to produce a prediction for the curvature for the next step

$$\hat{\xi}_k = \left\| w^{(k)} \right\| + \frac{\Delta s_k}{\Delta s_k + \Delta s_{k-1}} \left(\left\| w^{(k)} \right\| - \left\| w^{(k-1)} \right\| \right). \tag{3.27}$$

Since $\hat{\xi}_k$ can be negative, use

$$\xi_k = \max\{\xi_{min}, \hat{\xi}_k\} \quad \text{for some small} \quad \xi_{min} > 0, \tag{3.28}$$

as the predicted curvature for the next step.

The goal in estimating the optimal step size is to keep the error in the prediction $\|Z^{(0)} - Z^{(*)}\|$ relatively constant, so that the number of iterations required by the corrector will be stable. This is achieved by choosing the step size as

$$\hat{h} = \sqrt{\frac{2\delta_k}{\xi_k}}, \tag{3.29}$$

where δ_k represents the ideal starting error desired for the prediction step. δ_k is chosen as a function of the tolerance for tracking the curve and is also restricted to be no larger than half of Δs_k.

As with the normal flow algorithm, additional refinements on the optimal step size are made in order to prevent chattering and unreasonable values. In particular, $\bar{h}$ is chosen to satisfy equations (3.15) and (3.17). This $\bar{h}$ is then used as the step size for the next step.

The final phase of the algorithm, computation of the solution at $\lambda = 1$, is entered when a point $P^{(2)}$ is generated such that $P_1^{(2)} \geq 1$. $P^{(2)}$ is the first such point, so 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 $Z_1^{(k-2)} = 1$. 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 $\lambda = 1$.

Normally, the prediction $Z^{(k-2)}$ is computed by a secant method using the last two points $P^{(k)}$ and $P^{(k-1)}$:

$$Z^{(k-2)} = P^{(k)} + \left(P^{(k-1)} - P^{(k)}\right) \frac{\left(1 - P_1^{(k)}\right)}{\left(P_1^{(k-1)} - P_1^{(k)}\right)}. \tag{3.30}$$

However, this formula can potentially produce a disastrous prediction (e.g., if $|P_1^{(k-1)} - P_1^{(k)}| \ll |1 - P_1^{(k)}|$), so an additional scheme is added to ensure that this does not happen. In order to implement this scheme, a point $P^{(opp)}$ must be saved. This point is chosen as the last point computed from a quasi-Newton step which is on the opposite side of the hyperplane $\lambda = 1$ from $P^{(k)}$. Thus, the points $P^{(opp)}$ and $P^{(k)}$ bracket the solution. The prediction $Z^{(k-2)}$ may be bad whenever the inequality

$$\left\|Z^{(k-2)} - P^{(k)}\right\| > \left\|P^{(k)} - P^{(opp)}\right\| \tag{3.31}$$

is true. In this case, $Z^{(k-2)}$ is recomputed from the equation

$$Z^{(k-2)} = P^{(k)} + \left(P^{(opp)} - P^{(k)}\right) \frac{\left(1 - P_1^{(k)}\right)}{\left(P_1^{(opp)} - P_1^{(k)}\right)}. \tag{3.32}$$

This chord method, while much safer than the secant method (3.30), is used only in the special case (3.31) because it has a much slower rate of convergence than the secant method.

An exception to these linear prediction schemes occurs with the first step of the final phase. Since the tangents $T^{(1)}$ and $T^{(2)}$ at $P^{(1)}$ and $P^{(2)}$ are available, this information is used to generate a Hermite cubic polynomial $p(s)$ for calculating the first prediction point $Z^{(0)}$. This is done by finding the root $\bar{s}$ of the equation $p_1(s) = 1$. $Z^{(0)}$ is then given by

$$Z^{(0)} = p(\bar{s}). \tag{3.33}$$

After the predictor $Z^{(k-2)}$ has been determined, a quasi-Newton step is taken to get the point $P^{(k+1)}$. This step is defined by

$$P^{(k+1)} = Z^{(k-2)} + \Delta Z^{(k-2)}, \tag{3.34}$$

where $\Delta Z^{(k-2)}$ is the solution to (3.23). Again, the matrix in (3.23) is produced by the rank one updates (3.24) and (3.25).

The alternating process of computing a prediction and taking a quasi-Newton step is repeated until the solution is found.

In summary, the algorithm is:

1. $s := 0$, $y := (0, a)$, $ypold := (1, 0)$, $h := 0.1$, *failed* := false, *firststep* := true, *arcae, arcre* := absolute, relative error tolerances for tracking γ, *ansae, ansre* := absolute, relative error tolerances for the answer.
2. Compute the tangent *yp* at *y*, using (3.19) and (3.20), and update the augmented Jacobian matrix using (3.24).
3. If *firststep* = false then
    4. Compute the predicted point $Z^{(0)}$ with the cubic predictor (3.6) based on *yold*, *ypold*, *y*, *yp*.

   else

    5. Compute the predicted point $Z^{(0)}$ using a linear predictor based on *y* and *yp*.

```
6. If $failed$ = true then
    7. Compute the augmented Jacobian matrix at $Z^{(0)}$.
    8. Compute the next iterate $Z^{(1)}$ using (3.21).
9. $limit := 2\left(\lfloor -\log_{10}(arcae + arcre\|y\|) \rfloor + 1\right)$. Repeat steps 10–11 until either
       $\left\|\Delta Z^{(k)}\right\| \le arcae + arcre \left\|Z^{(k)}\right\|$
       or
       $limit$ iterations have been performed.
    10. Update the augmented Jacobian matrix using (3.25).
    11. Compute the next iterate using (3.21).
12. If the quasi-Newton iteration did not converge in $limit$ steps, then
    13. $h := h/2$; $failed :=$ true.
    14. If $h$ is unreasonably small, then return with an error flag.
    15. Go to 3.
16. Compute the tangent at the accepted iterate $Z^{(*)}$ using (3.19) and (3.20), and update the
    augmented Jacobian matrix using (3.24).
17. Compute the angle $\alpha$ between the current and previous tangents by (3.26).
18. If $\alpha > \pi/3$, then
    19. $h := h/2$; $failed :=$ true.
    20. If $h$ is unreasonably small, then return with an error flag.
    21. Go to 3.
22. $yold := y$, $ypold := yp$, $y := Z^{(*)}$, $yp :=$ tangent computed in step 16, $firststep :=$ false,
    $failed :=$ false.
23. If $y_1 < 1$, then compute a new step size $h$ by Equations (3.15, 3.17, 3.26–3.29) with $\xi_{min} = 0.01$,
       $\delta_k = \min\left\{(arcae + arcre\|y\|)^{1/4},\ \frac{1}{2}\|y - yold\|\right\}$,
    and go to 3.
24. Find $\bar{s}$ such that $p(\bar{s})_1 = 1$, using $yold$, $ypold$, $y$, and $yp$ in (3.6). $yopp := yold$, $Z^{(0)} := p(\bar{s})$.
25. $limit := 2\left(\lfloor -\log_{10}(ansae + ansre\|y\|) \rfloor + 1\right)$. Do steps 26–33 for $k = 2, \ldots, limit + 2$.
    26. Update the augmented Jacobian matrix using (3.25).
    27. Take a quasi-Newton step with (3.34).
    28. If
           $\left|P_1^{(k+1)} - 1\right| + \left\|\Delta Z^{(k-2)}\right\| \le ansae + ansre \left\|Z^{(k-2)}\right\|$,
        then return (solution has been found).
```

29. If $\left|P_1^{(k+1)} - 1\right| \leq ansae + ansre$,
    then
    $$Z^{(k-1)} := P^{(k+1)}$$
    else do steps 30–33.
30. $yold := y$, $y := P^{(k+1)}$.
31. If $yold_1$ and y_1 bracket $\lambda = 1$, then $yopp := yold$.
32. Compute $Z^{(k-1)}$ with the linear predictor (3.30) using y and $yold$.
33. If $\|Z^{(k-1)} - y\| > \|y - yopp\|$, then compute $Z^{(k-1)}$ with the linear predictor (3.32) using y and $yopp$.

34. Return with an error flag.

4. Software. This section describes HOMPACK [82], a software package currently under development at Sandia National Laboratories, General Motors Research Laboratories, Virginia Polytechnic Institute and State University, and the University of Michigan. HOMPACK is organized in two different ways: by algorithm/problem type and by subroutine level. There are three levels of subroutines. The top level consists of drivers, one for each problem type and algorithm type. Normally these drivers are called by the user, and the user need know nothing beyond them. They allocate storage for the lower level routines, and all the arrays are variable dimension, so there is no limit on problem size. The second subroutine level implements the major components of the algorithms such as stepping along the homotopy zero curve, computing tangents, and the end game for the solution at $\lambda = 1$. A sophisticated user might call these routines directly to have complete control of the algorithm, or for some other task such as tracking an arbitrary parametrized curve over an arbitrary parameter range. The lowest subroutine level handles the numerical linear algebra, and includes some BLAS routines. All the linear algebra and associated data structure handling are concentrated in these routines, so a user could incorporate his own data structures by writing his own versions of these low level routines. Also, by concentrating the linear algebra in subroutines, HOMPACK can be easily adapted to a vector or parallel computer.

The organization of HOMPACK by algorithm/problem type is shown in Table 1, which lists the driver name for each algorithm and problem type.

Table 1. Taxonomy of homotopy subroutines.

$x = f(x)$		$F(x) = 0$		$\rho(a, \lambda, x) = 0$		algorithm
dense	sparse	dense	sparse	dense	sparse	
FIXPDF	FIXPDS	FIXPDF	FIXPDS	FIXPDF	FIXPDS	ordinary differential equation
FIXPNF	FIXPNS	FIXPNF	FIXPNS	FIXPNF	FIXPNS	normal flow
FIXPQF	FIXPQS	FIXPQF	FIXPQS	FIXPQF	FIXPQS	augmented Jacobian matrix

The naming convention is

$$FIXP \left\{ \begin{matrix} D \\ N \\ Q \end{matrix} \right\} \left\{ \begin{matrix} F \\ S \end{matrix} \right\},$$

where $D \approx$ ordinary differential equation algorithm, $N \approx$ normal flow algorithm, $Q \approx$ augmented Jacobian matrix algorithm, $F \approx$ dense Jacobian matrix, and $S \approx$ sparse Jacobian matrix. Using brackets to indicate the three subroutine levels described above, the natural grouping of the HOMPACK routines is:

[FIXPDF] [FODE, ROOT, SINTRP, STEPS] [DCPOSE]

[FIXPDS] [FODEDS, ROOT, SINTRP, STEPDS] [GMFADS, MFACDS, MULTDS, PCGDS, QIMUDS, SOLVDS]

[FIXPNF] [ROOTNF, STEPNF, [TANGNF]] [ROOT]

[FIXPNS] [ROOTNS, STEPNS, TANGNS] [GMFADS, MFACDS, MULTDS, PCGDS, PCGNS, QIMUDS, ROOT, SOLVDS]

[FIXPQF] [ROOTQF, STEPQF, TANGQF] [QRFAQF, QRSLQF, R1UPQF, UPQRQF]

[FIXPQS] [ROOTQS, STEPQS, TANGQS] [GMFADS, MULTDS, PCGQS, SOLVDS]

The BLAS subroutines used by HOMPACK are DAXPY, DCOPY, DDOT, DNRM2, DSCAL, D1MACH, IDAMAX.

The user written subroutines, of which exactly two must be supplied depending on the driver chosen, are F, FJAC, JFACS, RHO, RHOA, RHOJAC, RHOJS.

The special purpose polynomial system solver POLSYS is essentially a high level driver for HOMPACK. POLSYS requires special versions of RHO and RHOJAC (subroutines normally provided by the user). These special versions are included in HOMPACK, so for a polynomial system the user need only call POLSYS, and define the problem directly to POLSYS by specifying the polynomial coefficients. POLSYS scales and computes partial derivatives on its own. Thus the user interface to POLSYS and HOMPACK is clean and simple. The only caveat is that FFUNP cannot recognize patterns of repeated expressions in the polynomial system, and so may be less efficient than a hand crafted version. If great efficiency is required, the user can modify the default FFUNP; the sections in the code which must be changed are clearly marked. The grouping is:

[POLSYS] [POLYNF, POLYP, ROOTNF, STEPNF, TANGNF] [DIVP, FFUNP, GFUNP, HFUNP, HFUN1P, INITP, MULP, OTPUTP, POWP, RHO, RHOJAC, ROOT, SCLGNP, STRPTP]

5. Applications.

The globally convergent probability one homotopy algorithms described here have been successfully applied to a wide range of practical scientific and engineering problems. A survey of some engineering applications prior to 1980 is given in [76]. Some problems for which locally convergent iterative methods either totally or partially failed, and for which homotopy methods succeeded, are given in the references [12], [20], [22], [44], [49–67], [70], [76], [78–81], [83]. The example at the beginning of this paper is typical.

The fields where homotopy methods have been successfully applied include:
nonlinear structural mechanics (limit point and postbuckling analysis),
fluid mechanics,
statistics (nonlinear regression),

chemistry (reaction and equilibrium equations),
chemical engineering (reactor modelling and process control),
robotics (inverse kinematics),
computer vision (shape from shading, structure from motion),
economics (fixed points),
industrial engineering (nonlinear complementarity),
control theory (pole placement),
CAD/CAM (solid modelling),

6. Parallel computation. This section describes some ongoing work to implement HOMPACK on a parallel computer. The serial version of HOMPACK was structured so as to be easily adapted to a vector machine, by segregating most of the linear algebra into subroutines. The appropriate algorithmic decomposition for a parallel computer like the hypercube is not so clear. Here the implementation of HOMPACK on a hypercube machine for one special class of problems, polynomial systems, is addressed.

6.1. The hypercube. The word "hypercube" refers to an n-dimensional cube. Think of a cube in n dimensions as sitting in the positive orthant, with vertices at the points

$$(v_1, \ldots, v_n), \quad v_i \in \{0, 1\}, \quad i = 1, \ldots, n.$$

There are thus 2^n vertices, and two vertices v and w are "adjacent", i.e., connected by an edge, if and only if $v_i = w_i$ for all i except one. The associated graph, also sometimes referred to an an "n-cube", has 2^n vertices (which can be labelled as above with binary n-tuples) and edges between vertices whose labels differ in exactly one coordinate (see Figure 3).

A "hypercube computer architecture" is a computer system with 2^n (node) processors, corresponding to the 2^n vertices (nodes), and a communication link corresponding to each edge of the n-cube. Thus each processor has a direct communication link to exactly n other processors. The distance between any two of the $P = 2^n$ processors is at most $n = \log_2 P = \log_2(2^n)$, considered an ideal compromise between total connectivity (distance = 1) and ring connectivity (distance = $P/2$). Figure 1 shows how a 4-cube is built up from two 3-cubes.

Typically the node label $(v_1, \ldots, v_n)$ is viewed as a binary number $v_1 v_2 \ldots v_n$, and in this view two nodes are adjacent if and only if their binary representations differ in exactly one bit. Typically node addresses are computed in programs by a gray code, a bijective function

$$g : \{0, \ldots, 2^n - 1\} \to \{0, \ldots, 2^n - 1\}$$

such that the binary representations of $g(k \pmod{2^n})$ and $g(k + 1 \pmod{2^n})$ differ in exactly one bit for all k.

Realizations of this abstract architecture have one additional feature: a "host" processor with communication links to *all* the node processors. This host typically loads programs into the nodes, starts and stops processes executing in the nodes, and interchanges data with the nodes. In current hardware implementations only the host has external I/O and peripheral storage; the nodes consists of memory, a CPU, and possibly communication and floating-point hardware.

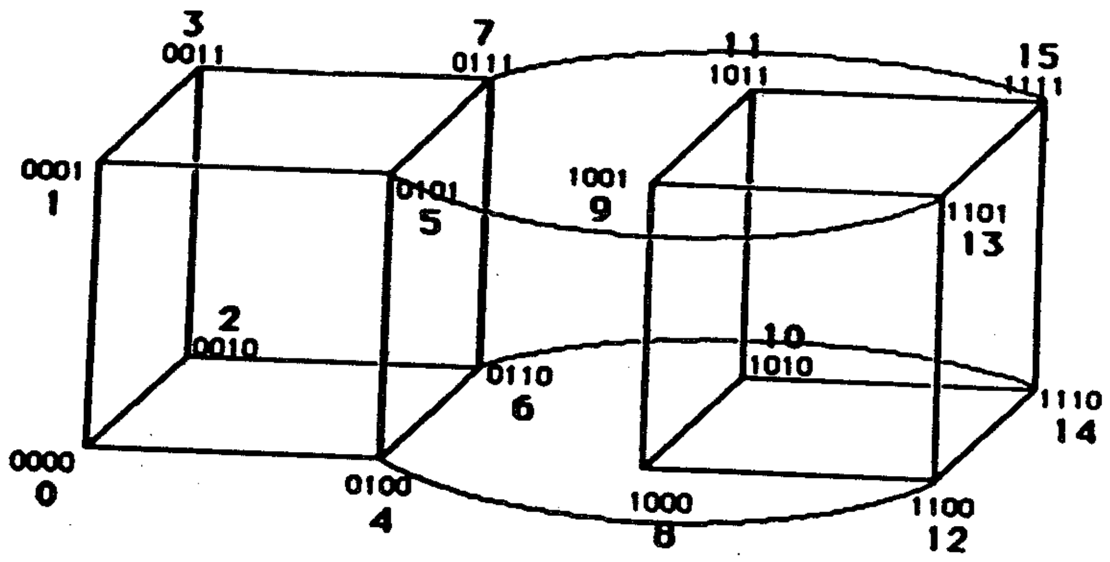

Figure 3. 4-cube structure and node labelling.

The Intel iPSC has 32, 64, or 128 nodes. Each node is an 80286/80287 with 512K bytes of memory. The host is also an 80286/80287, but with 4 MB of memory, a floppy disk drive, a hard disk, an Ethernet connection, and Xenix. The nodes have only a minimal monitor for communication and process management.

The NCUBE has up to 1024 nodes in multiples of 64, each with 128K of memory and communication and floating-point hardware. The host is an 80286, running NCUBE's operating system, a primitive version of UNIX. The node chip is NCUBE's own design, with a unique feature being communication hardware.

6.2. Polynomial systems. Suppose that the components of the nonlinear system of equations (2.1) have the form

$$F_i(x) = \sum_{k=1}^{n_i} a_{ik} \prod_{j=1}^{n} x_j^{d_{ijk}}, \quad i = 1, \ldots, n. \tag{6.1}$$

The ith component $F_i(x)$ has n_i terms, the a_{ik} are the (real) coefficients, and the degrees d_{ijk} are nonnegative integers. The total degree of F_i is

$$d_i = \max_k \sum_{j=1}^{n} d_{ijk}. \tag{6.2}$$

The *total degree* of the entire system (6.1) is

$$d = d_1 \cdots d_n. \tag{6.3}$$

The homotopy map $\rho_c : [0,1) \times C^n \to C^n$ has the form

$$\rho_c(\lambda, x) = (1-\lambda)G(x) + \lambda F(x), \tag{6.4}$$

where $F(x)$ is now regarded as a complex map $F : C^n \to C^n$, and $G : C^n \to C^n$ is defined by

$$G_j(x) = b_j\, x_j^{d_j} - a_j, \quad j = 1, \ldots, n, \tag{6.5}$$

where a_j and b_j are nonzero complex numbers, d_j was defined in (6.2), and

$$c = (a_1, \ldots, a_n, b_1, \ldots, b_n)$$

is the parameter vector for the homotopy map. Through a series of ingenious tricks – converting $F(x)$ to a complex map and then back to a real map, converting to projective coordinates, and adding a linear equation in projective space – the following attributes can be achieved (with probability one):

1) there are exactly d zero curves of ρ_c emanating from $\lambda = 0$;
2) $d\lambda/ds \geq 0$ along each zero curve;
3) all the zero curves are bounded;
4) every isolated solution of (2.1) is reached by some zero curve of ρ_c.

There is thus a rigorous theoretical algorithm [33–34, 82] for finding *all* solutions to a polynomial system (2.1). This algorithm is implemented in subroutine POLSYS from HOMPACK.

6.3. Computational results. Polynomial systems arise in such diverse areas as solid modelling, robotics, chemical engineering, mechanical engineering, and computer vision. A small problem has total degree $d < 100$ and a large problem has $d > 1000$. Given that d homotopy paths are to be tracked, there are two extreme approaches using a hypercube.

The first extreme, with the coarsest granularity possible, is to assign one path to each node processor, with the host controlling the assignment of paths to the nodes, keeping as many nodes busy as possible, and post-processing the answers computed by the nodes.

The second extreme, with the finest granularity, is to track all d paths on the host processor, distributing the numerical linear algebra, polynomial system evaluation, Jacobian matrix evaluation, and possibly other things amongst the nodes. Because of the high communication overhead (whether hardware or software) on a hypercube, this approach requires an immense amount of sophistication and analysis to prevent the communication costs from overwhelming the computational costs. Also the algorithm at this granularity would have to be a major modification of the serial algorithm. A possible advantage is that the load could be balanced better, resulting in an overall speedup over a coarser grained algorithm.

These issues are not simple, and much more research is needed on these and intermediate approaches. Neither extreme can be declared inferior *a priori*. The first approach, having a node track an entire path, has been tried, and some results are shown in Table 1. The problem number

Table 1. Execution time (secs).

Problem number	total degree	80286/ 80287	iPSC-32	VAX 11/750	VAX 11/780
102	256	16257	645	2438	1248
103	625	34692	1616	5260	2656
402	4	255	54	41	18
403	4	84	19	14	6
405	64	3669	335	703	334
601	60	9450	257	1707	796
602	60	28783	2795	4332	2054
603	12	1200	243	325	152
803	256	—		29779	16221
1702	16	1655	163	216	112
1703	16	1657	162	216	112
1704	16	1628	108	216	112
1705	81	14336	378	1384	999
5001	576	—		49736	27815

refers to an internal numbering scheme used at General Motors Research Laboratories; problem data is available on request. These problems are all real engineering problems that have arisen at GM and elsewhere.

The parallel computation scheme of assigning a path to each node seems completely transparent and straightforward. Why this is not so is illustrative of the difficulty of the whole field of parallel computation. A detailed discussion of tasks that should be trivial, but were not, is contained in [35].

7. References.

[1] J. C. ALEXANDER, *The topological theory of an imbedding method*, in Continuation Methods (H. G. Wacker, Ed.), Academic Press, New York, 1978, pp. 37–68.

[2] J. C. ALEXANDER AND J. A. YORKE, *The homotopy continuation method: numerically implementable topological procedures*, Trans. Amer. Math. Soc., 242 (1978), pp. 271–284.

[3] J. C. ALEXANDER, T.-Y. LI, AND J. A. YORKE, *Piecewise smooth homotopies*, in Homotopy Methods and Global Convergence, B. C. Eaves, F. J. Gould, H.-O. Peitgen, and M. J. Todd, eds., Plenum, New York, 1983, pp. 1–14.

[4] E. ALLGOWER AND K. GEORG, *Simplicial and continuation methods for approximating fixed points*, SIAM Rev., 22 (1980), pp. 28–85.

[5] S. C. BILLUPS, *An augmented Jacobian matrix algorithm for tracking homotopy zero curves*, M.S. Thesis, Dept. of Computer Sci., VPI & SU, Blacksburg, VA, Sept., 1985.

[6] P. BOGGS, *The solution of nonlinear systems of equations by A-stable integration techniques*, SIAM J. Numer. Anal., 8 (1971), pp. 767–785.

[7] P. BUSINGER AND G. H. GOLUB, *Linear least squares solutions by Householder transformations*, Numer. Math., 7 (1965), pp. 269–276.

[8] S. N. CHOW, J. MALLET-PARET, AND J. A. YORKE, *Finding zeros of maps: homotopy methods that are constructive with probability one*, Math. Comp., 32 (1978), pp. 887–899.

[9] S. N. CHOW, J. MALLET-PARET, AND J. A. YORKE, *A homotopy method for locating all zeros of a system of polynomials*, in Functional Differential Equations and Approximation of Fixed Points, H.-O. Peitgen and H. O. Walther, eds., Springer-Verlag Lecture Notes in Math. 730, New York, 1979, pp. 228–237.

[10] D. F. DAVIDENKO, *On the approximate solution of systems of nonlinear equations*, Ukrain. Mat. Ž., 5 (1953), pp. 196–206.

[11] J. E. DENNIS AND R. SCHNABEL, *Numerical Methods for Unconstrained Optimization and Nonlinear Equations*, Prentice-Hall, Englewood Cliffs, NJ, 1983.

[12] J. P. DUNYAK, J. L. JUNKINS, AND L. T. WATSON, *Robust nonlinear least squares estimation using the Chow-Yorke homotopy method*, J. Guidance, Control, Dynamics, 7 (1984), pp. 752–755.

[13] B. C. EAVES, *Homotopies for the computation of fixed points*, Math. Programming, 3 (1972), pp. 1–22.

[14] B. C. EAVES AND R. SAIGAL, *Homotopies for computation of fixed points on unbounded regions*, Math. Programming, 3 (1972), pp. 225–237.

[15] D. K. FADEEV AND V. N. FADEEVA, *Computational Methods of Linear Algebra*, Freeman, London, 1963.

[16] K. GEORG, private communication, 1981.

[17] K. GEORG, *A note on stepsize control for numerical curve following*, in Homotopy Methods and Global Convergence, B. C. Eaves, F. J. Gould, H.-O. Peitgen, and M. J. Todd, eds., Plenum, New York, 1983, pp. 145–154.

[18] K. GEORG, *Numerical integration of the Davidenko equation*, in Numerical Solution of Nonlinear Equations, E. Allgower, K. Glashoff, H.-O. Peitgen, eds., Springer-Verlag Lecture Notes in Math. 878, New York, 1981.

[19] P. E. GILL AND W. MURRAY, *Newton type methods for unconstrained and linearly constrained optimization*, Math. Programming, 7 (1974), pp. 311–350.

[20] M. W. HERUSKA, L. T. WATSON, AND K. K. SANKARA, *Micropolar flow past a porous stretching sheet*, Comput. & Fluids, to appear.

[21] M. R. HESTENES, *The conjugate gradient method for solving linear systems*, in Proc. Symp. Appl. Math., 6, Numer. Anal., AMS, New York, 1956, pp. 83–102.

[22] M. P. KAMAT, L. T. WATSON, AND V. B. VENKAYYA, *A quasi-Newton versus a homotopy method for nonlinear structural analysis*, Comput. & Structures, 17 (1983), pp. 579–585.

[23] H. B. KELLER, *Numerical Solution of Two-point Boundary Value Problems*, Society for Industrial and Applied Mathematics, Philadelphia, 1976.

[24] ——, *Numerical solution of bifurcation and nonlinear eigenvalue problems*, in Applications of Bifurcation Theory, Academic Press, New York, 1977.

[25] R. W. KLOPFENSTEIN, *Zeros of nonlinear functions*, J. Assoc. Comput. Mech., 8 (1961), pp. 336–373.

[26] M. KUBICEK, *Dependence of solutions of nonlinear systems on a parameter*, ACM-TOMS, 2 (1976), pp. 98–107.

[27] H. H. KWOK, M. P. KAMAT, AND L. T. WATSON, *Location of stable and unstable equilibrium configurations using a model trust region quasi-Newton method and tunnelling*, Comput. & Structures, 21 (1985), pp. 909–916.

[28] R. MEJIA, *CONKUB: A conversational path-follower for systems of nonlinear equations*, J. Comput. Phys., to appear.

[29] R. MENZEL AND H. SCHWETLICK, *Zur Lösung parameterabhängiger nichtlinearer Gleichungen mit singulären Jacobi-Matrizen*, Numer. Math., 30 (1978), pp. 65–79.

[30] O. MERRILL, *Applications and extensions of an algorithm to compute fixed points of upper semicontinuous mappings*, Doctoral thesis, I. O. E. Dept., Univ. of Michigan, Ann Arbor, 1972.

[31] G. MEYER, *On solving nonlinear equations with a one-parameter operator embedding*, SIAM J. Numer. Anal., 5 (1968), pp. 739–752.

[32] J. J. MORÉ, B. S. GARBOW, AND K. E. HILLSTROM, *User Guide for MINPACK-1*, ANL-80-74, Argonne National Laboratory, Argonne, IL (1980).

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

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

[35] A. P. MORGAN AND L. T. WATSON, *Solving nonlinear equations on a hypercube*, Proc. ASCE Structures Congress, New Orleans, Sept., 1986.

[36] W. C. RHEINBOLDT AND C. DEN HÉYÉR, *On steplength algorithms for a class of continuation methods*, SIAM J. Numer. Anal., 18 (1981), pp. 925–947.

[37] W. C. RHEINBOLDT, *On the computation of critical boundaries on equilibrium surfaces*, SIAM J. Numer. Anal., 19 (1982), pp. 653–669.

[38] W. C. RHEINBOLDT AND R. MELHEM, *A comparison of methods for determining turning points of nonlinear equations*, Computing, 29 (1982), pp. 103–114.

[39] W. C. RHEINBOLDT AND J. V. BURKARDT, *Algorithm 596: A program for a locally parameterized continuation process*, ACM Trans. Math. Software, 9 (1983), pp. 236-241.

[40] E. RIKS, *An incremental approach to the solution of snapping and buckling problems*, Internat. J. Solids Structures, 15 (1979), pp. 529–551.

[41] R. SAIGAL, *On the convergence rate of algorithms for solving equations that are based on methods of complementary pivoting*, Math. Operations Res., 2 (1977), pp. 108–124.

[42] R. SAIGAL AND M. J. TODD, *Efficient acceleration techniques for fixed point algorithms*, SIAM J. Numer. Anal., 15 (1978), pp. 997–1007.

[43] R. SAIGAL, *An efficient procedure for traversing large pieces in fixed point algorithms*, in Homotopy Methods and Global Convergence, B. C. Eaves, F. J. Gould, H.-O. Peitgen, and M. J. Todd, eds., Plenum, New York, 1983, pp. 239–248.

[44] K. K. SANKARA AND L. T. WATSON, *Micropolar flow past a stretching sheet*, Z. Angew. Math. Phys., 36 (1985), pp. 845–853.

[45] H. SCARF, *The approximation of fixed points of a continuous mapping*, SIAM J. Appl. Math., 15 (1967), pp. 1328–1343.

[46] M. R. SCOTT, *Invariant Imbedding and its Applications to Ordinary Differential Equations*, Addison-Wesley, Reading, MA, 1973.

[47] L. F. SHAMPINE AND M. K. GORDON, *Computer Solution of Ordinary Differential Equations: The Initial Value Problem*, W. H. Freeman, San Francisco, 1975.

[48] S. SMALE, *Convergent process of price adjustment and global Newton methods*, J. Math. Econom., 3 (1976), pp. 107–120.

[49] C. Y. WANG AND L. T. WATSON, *Squeezing of a viscous fluid between elliptic plates*, Appl. Sci. Res., 35 (1979), pp. 195–207.

[50] ——, *Viscous flow between rotating discs with injection on the porous disc*, Z. Angew. Math. Phys., 30 (1979), pp. 773–787.

[51] ——, *On the large deformations of C-shaped springs*, Internat. J. Mech. Sci., 22 (1980), pp. 395–400.

[52] ——, *Theory of the constant force spring*, J. Appl. Mech., 47 (1980), pp. 956–958.

[53] ——, *The fluid-filled cylindrical membrane container*, J. Engrg. Math., 15 (1981), pp. 81–88.

[54] ——, *Equilibrium of heavy elastic cylindrical shells*, J. Appl. Mech., 48 (1981), pp. 582–586.

[55] ——, *The elastic catenary*, Internat. J. Mech. Sci., 24 (1982), pp. 349–357.

[56] ——, *Free rotation of a circular ring about a diameter*, Z. Angew. Math. Phys., 34 (1983), pp. 13–24.

[57] C. Y. WANG, L. T. WATSON, AND M. P. KAMAT, *Buckling, postbuckling and the flow through a tethered elastic cylinder under external pressure*, J. Appl. Mech., 50 (1983), pp. 13–18.

[58] L. T. WATSON, T. Y. LI AND C. Y. WANG, *Fluid dynamics of the ellilptic porous slider*, J. Appl. Mech., 45 (1978), pp. 435–436.

[59] L. T. WATSON AND C. Y. WANG, *A homotopy method applied to elastica problems*, Internat. J. Solids Structures, 17 (1981), pp. 29–37.

[60] ——, *Deceleration of a rotating disc in a viscous fluid*, Phys. Fluids, 22 (1979), pp. 2267–2269.

[61] ——, *The circular leaf spring*, Acta Mech., 40 (1981), pp. 25–32.

[62] ——, *Hanging an elastic ring*, Internat. J. Mech. Sci., 23 (1981), pp. 161–168.

[63] ——, *Overhang of a heavy elastic sheet*, Z. Angew. Math. Phys., 33 (1982), pp. 17–23.

[64] ——, *Periodically supported heavy elastic sheet*, J. Engrg. Mech., 109 (1983), pp. 811–820.

[65] ——, *The equilibrium states of a heavy rotating column*, Internat. J. Solids Structures, 19 (1983), pp. 653–658.

[66] L. T. WATSON AND W. H. YANG, *Optimal design by a homotopy method*, Applicable Anal., 10 (1980), pp. 275–284.

[67] ——, *Methods for optimal engineering design problems based on globally convergent methods*, Comput. & Structures, 13 (1981), pp. 115–119.

[68] L. T. WATSON AND D. FENNER, *Chow-Yorke algorithm for fixed points or zeros of C^2 maps*, ACM TOMS, 6 (1980), pp. 252–260.

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

[70] ——, *Numerical study of porous channel flow in a rotating system by a homotopy method*, J. Comput. Appl. Math., 7 (1981), pp. 21–26.

[71] ——, *Computational experience with the Chow-Yorke algorithm*, Math. Programming, 19 (1980), pp. 92–101.

[72] ——, *Fixed points of C^2 maps*, J. Comput. Appl. Math., 5 (1979), pp. 131–140.

[73] ——, *Solving the nonlinear complementarity problem by a homotopy method*, SIAM J. Control Optim., 17 (1979), pp. 36–46.

[74] ——, *An algorithm that is globally convergent with probability one for a class of nonlinear two-point boundary value problems*, SIAM J. Numer. Anal., 16 (1979), pp. 394–401.

[75] ——, *Solving finite difference approximations to nonlinear two-point boundary value problems by a homotopy method*, SIAM J. Sci. Stat. Comput., 1 (1980), pp. 467–480.

[76] ——, *Engineering applications of the Chow-Yorke algorithm*, Appl. Math. Comput., 9 (1981), pp. 111–133.

[77] ——, *Numerical linear algebra aspects of globally convergent homotopy methods*, Tech. Rep. 86–7, Dept. of Industrial and Operations Eng., Univ. of Michigan, Ann Arbor, 1986.

[78] L. T. Watson, S. M. Holzer, and M. C. Hansen, *Tracking nonlinear equilibrium paths by a homotopy method*, Nonlinear Anal., 7 (1983), pp. 1271–1282.

[79] L. T. Watson, K. K. Sankara, and L. C. Mounfield, *Deceleration of a porous rotating disk in a viscous fluid*, Internat. J. Engrg. Sci., 23 (1985), pp. 131–137.

[80] L. T. Watson, M. P. Kamat, and M. H. Reaser, *A robust hybrid algorithm for computing multiple equilibrium solutions*, Engrg. Comput., 2 (1985), pp. 30–34.

[81] L. T. Watson and M. R. Scott, *Solving spline collocation approximations to nonlinear two-point boundary value problems by a homotopy method*, Tech. Rep. CS84015-R, Dept. of Computer Sci., VPI&SU, Blacksburg, VA, 1984.

[82] L. T. Watson, S. C. Billups, and A. P. Morgan, *HOMPACK: A suite of codes for globally convergent homotopy algorithms*, Tech. Rep. 85–34, Dept. of Industrial and Operations Eng., Univ. of Michigan, Ann Arbor, MI, 1985.

[83] L. T. Watson and L. R. Scott, *Solving Galerkin approximations to nonlinear two-point boundary value problems by a globally convergent homotopy method*, Tech. Rep. 86–16, Dept. of Industrial and Operations Eng., Univ. of Michigan, Ann Arbor, MI, 1986.