Acta Numerica (1993), *pp.* 1–64

Continuation and path following

Eugene L. Allgower* and Kurt Georg*
Department of Mathematics
Colorado State University
Ft. Collins, CO 80523, USA
E-mail: Georg@Math.ColoState.Edu

The main ideas of path following by predictor–corrector and piecewise-linear methods, and their application in the direction of homotopy methods and nonlinear eigenvalue problems are reviewed. Further new applications to areas such as polynomial systems of equations, linear eigenvalue problems, interior methods for linear programming, parametric programming and complex bifurcation are surveyed. Complexity issues and available software are also discussed.

CONTENTS

1. Introduction

Continuation, embedding or homotopy methods have long served as useful theoretical tools in modern mathematics. Their use can be traced back at least to such venerated works as those of Poincaré (1881–1886), Klein (1882–1883) and Bernstein (1910). Leray and Schauder (1934) refined the tool and presented it as a global result in topology, viz. the homotopy invariance of degree. The use of deformations to solve nonlinear systems of equations may be traced back at least to Lahaye (1934). The classical embedding

* Partially supported by the National Science Foundation via grant no. DMS-9104058.

methods were the first deformation methods to be numerically implemented and may be regarded as a forerunner of the predictor–corrector methods for path following which we will discuss here.

Because of their versatility and robustness, numerical continuation or path following methods have now been finding ever wider use in scientific applications. Our aim here is to present some of the recent advances in this subject regarding new adaptations, applications, and analysis of efficiency and complexity. To make the discussion relatively self-contained, we review some of the background of numerical continuation methods. Introductions into aspects of the subject may be found in the books by Garcia and Zangwill (1981), Gould and Tolle (1983), Keller (1987), Rheinboldt (1986), Seydel (1988) and Todd (1976a). The philosophy and notation of the present article will be that of our book Allgower and Georg (1990), which also contains an extensive bibliography up to 1990.

The viewpoint which will be adopted here is that numerical continuation methods are techniques for numerically approximating a solution curve c which is implicitly defined by an underdetermined system of equations. In the literature of numerical analysis, the terms *numerical continuation* and *path following* are used interchangeably.

There are various objectives for which the numerical approximation of c can be used and, depending upon the objective, the approximating technique is adapted accordingly. In fact, continuation is a unifying concept, under which various numerical methods may be subsumed which may otherwise have very little in common. For example, simplicial fixed point methods for solving problems in mathematical economics, the generation of bifurcation diagrams of nonlinear eigenvalue problems involving partial differential equations, and the recently developed interior point methods for solving linear programming problems seem to be quite unrelated. Nevertheless, there is some benefit in considering them as special cases of path following. We personally are struck by the remarkable fact that a technique which was initially developed for solving difficult nonlinear problems now turns out to be extremely useful for treating various problems which are essentially linear: e.g. linear eigenvalue problems, and linear programming and complementarity problems.

The remainder of the article is organized as follows. Section 2 contains the basic ideas of predictor–corrector path following methods. In Section 3 some technical aspects of implementing predictor–corrector methods are addressed, e.g. the numerical linear algebra involved and steplength strategies.

Section 4 deals with various applications of path following methods. We begin with a brief discussion of homotopy methods for fixed point problems and global Newton methods. Then we address the problem of finding multiple solutions. In particular, we discuss recent homotopy methods for finding all solutions of polynomial systems of equations. Next we survey some path

following aspects of nonlinear eigenvalue problems, and address the question of handling bifurcations. Finally, three new developments in path following are discussed: (1) The solution of linear eigenvalue problems via special homotopy approaches; (2) the handling of parametric programming problems by following certain branches of critical points via active set strategies; and (3) the path following aspects involved in the interior point methods for solving linear and quadratic programming problems.

Section 5 presents an introduction to the principles of piecewise linear methods. These methods view path following in a different light: instead of approximately following a smooth solution curve, they exactly follow an approximate curve (i.e. a polygonal path). Some instances where these methods are useful are discussed, e.g. linear complementarity problems or homotopy methods where predictor–corrector methods are not implementable, because of lack of smoothness. We also briefly address the related topic of approximating implicitly defined surfaces.

The issue of the computational complexity of path following is considered in Section 6. This issue is related to the Newton–Kantorovich theory and is currently of considerable interest in the context of interior point methods.

We conclude by listing some available software related to path following and indicate how the reader might access these codes. No attempt to compare or evaluate the various codes is offered. In any case, our opinion is that path following codes always need to be considerably adapted to the special purposes for which they are designed. The path following literature offers various tools for accomplishing such tasks. Although there are some general purpose codes, probably none will slay every dragon.

The extensive bibliography contains only cited items. Space considerations prohibited the addressing of some important topics, and consequently some significant recent contributions to the field are not contained in the bibliography.

2. The basics of predictor–corrector path following

The simplest (and most frequently occurring) case of an underdetermined system of nonlinear equations contains just one degree of freedom:

$$H(u) = 0 \text{ where } H : \mathbb{R}^{N+1} \to \mathbb{R}^N \text{ is a smooth map.} \tag{2.1}$$

When we say that a map is smooth we shall mean that it has as many continuous derivatives as the context of the discussion requires. For convenience, the reader may assume C^∞. In order to apply the Implicit Function Theorem, we need the following standard

Definition 2.1 We call u a *regular point* of H if the Jacobian $H'(u)$ has maximal rank. We call y a *regular value* of H if u is a regular point of

H whenever $H(u) = y$. If a point or value is not regular, then it is called *singular*.

Let $u_0 \in \mathbb{R}^{N+1}$ be a regular point of H such that $H(u_0) = 0$. It follows from the Implicit Function Theorem that the solution set $H^{-1}(0)$ can be locally parametrized about u_0 with respect to some coordinate. By a reparametrization (according to arclength), we obtain a smooth curve $c : J \to \mathbb{R}^{N+1}$ for some open interval J containing zero such that for all $s \in J$:

$$c(0) = u_0 \tag{2.2}$$
$$H'(c(s))\dot{c}(s) = 0, \tag{2.3}$$
$$\|\dot{c}(s)\| = 1, \tag{2.4}$$
$$\det \begin{pmatrix} H'(c(s)) \\ \dot{c}(s)^* \end{pmatrix} > 0. \tag{2.5}$$

These conditions uniquely determine the tangent $\dot{c}(s)$. Here and in the following, $(\,.\,)^*$ denotes the Hermitian transpose and $\|\,.\,\|$ the Euclidean norm. Condition (2.4) normalizes the parametrization to arclength. This is only for theoretical convenience, and it is not an intrinsic restriction. Condition (2.5) chooses one of the two possible orientations.

The preceding discussion motivates the following

Definition 2.2 Let A be an $(N,N+1)$-matrix with maximal rank. For the purpose of our exposition, the unique vector $t(A) \in \mathbb{R}^{N+1}$ satisfying the conditions

$$At = 0, \tag{2.6}$$
$$\|t\| = 1, \tag{2.7}$$
$$\det \begin{pmatrix} A \\ t^* \end{pmatrix} > 0, \tag{2.8}$$

will be called the *tangent vector induced by* A.

Making use of this definition, solution curve $c(s)$ is characterized as the solution of the initial value problem

$$\dot{u} = t(H'(u)), \quad u(0) = u_0 \tag{2.9}$$

which in this context is occasionally attributed to Davidenko (1953), see also Branin (1972). Note that the domain $\{u \in \mathbb{R}^{N+1} : u \text{ is a regular point}\}$ is open. This differential equation is not used in efficient path following algorithms, but it serves as a useful device in analysing the path. Two examples are:

Lemma 2.3 Let (a, b) be the maximal interval of existence for (2.9). If a

is finite, then $c(s)$ converges to a singular zero point of H as $s \to a$, $s > a$. An analogous statement holds if b is finite.

Lemma 2.4 Let zero be a regular value of H. Then the solution curve c is defined on the real line and satisfies one of the following two conditions:

1. The curve c is diffeomorphic to a circle. More precisely, there is a period $T > 0$ such that $c(s_1) = c(s_2)$ if and only if $s_1 - s_2$ is an integer multiple of T.
2. The curve c is diffeomorphic to the real line. More precisely, c is injective, and $c(s)$ has no accumulation point for $s \to \pm\infty$.

See (2.1.13) and (2.1.14) in Allgower and Georg (1990) for proofs. A more topological and global treatment of the Implicit Function Theorem can be found in the books of Hirsch (1976) or Milnor (1969).

Since the solution curve c is characterized by the initial value problem (2.9), it is evident that the numerical methods for solving initial value problems could immediately be used to numerically trace c. However, in general this is not an efficient approach, since it ignores the contractive properties which the curve c has in view of the fact that it satisfies the equation $H(u) = 0$. Instead, a typical path following method consists of a succession of two different steps:

Predictor step. An approximate step along the curve, usually in the general direction of the tangent of the curve. The initial value problem (2.9) provides motivation for generating predictor steps in the spirit of the technology of numerical solution of initial value problems.

Corrector steps. One or more iterative steps which aim to bring the predicted point back to the curve by an iterative procedure (typically of Newton or gradient type) for solving $H(u) = 0$.

It is usual to call such procedures *predictor–corrector* path following methods. However, let us note that this name should not be confused with the predictor–corrector multistep methods for initial value problems, since the latter do not converge back to the solution curve.

The following pseudocode (in **MATLAB** format) shows the basic steps of a generic predictor–corrector method.

Algorithm 2.5 u = generic_pc_method(u, h)

```
% u ∈ R^{N+1} such that H(u) ≈ 0 is an initial point, input
% h > 0 is an initial steplength, input
   WHILE a stopping criterion is not met
      % predictor step
      predict v such that H(v) ≈ 0 and ||u − v|| ≈ h
      and v − u points in the direction of traversing
```

% *corrector step*
let $w \in \mathbb{R}^{N+1}$ approximately solve ...
$\min_w \{ \|v - w\| : H(w) = 0 \}$
% *new point along* $H^{-1}(0)$
$u = w$
% *steplength adaptation*
choose a new steplength $h > 0$
END

The predictor–corrector type of algorithms for curve following seem to date to Haselgrove (1961). In contrast to the modern predictor–corrector methods, the classical embedding methods assume that the solution path is parametrized with respect to an explicit parameter which is identified with the last variable in H. Hence, we consider the equation (2.1) in the form

$$H(x, \lambda) = 0. \tag{2.10}$$

If we assume that the partial derivative $H_x(x, \lambda)$ does not vanish, then the solution curve can be parametrized in the form $(x(\lambda), \lambda)$. This assumption has the drawback that *folds* are excluded, i.e. points such that $H(x, \lambda) = 0$ and $H_x(x, \lambda) = 0$. Such points are sometimes called turning points in the literature. The assumption has, however, the advantage that the corrector steps can be more easily handled, in particular if the partial derivative of H with respect to x is sparse. In some applications it is known *a priori* that no folds are present, and then the embedding method is applicable. For purposes of illustration we present an analogous generic embedding method:

Algorithm 2.6 x = generic_embedding_method(x, λ, h)

% $(x, \lambda) \in \mathbb{R}^{N+1}$ *such that* $H(x, \lambda) \approx 0$ *is an initial point, input*
% $h > 0$ *is an initial steplength, input*
WHILE a stopping criterion is not met
let $y \in \mathbb{R}^N$ approximately solve $H(y, \lambda + h) = 0$
$(x, \lambda) = (y, \lambda + h)$
choose a new steplength $h > 0$
END

The predictor step is hidden; the predictor point would correspond to the starting point of an iterative method for solving $H(y, \lambda + h) = 0$. The most commonly used starting point is the previous point x.

It is common to blend aspects of these two algorithms. A simple example is to use a predictor tangent to the curve $(x(\lambda), \lambda)$ in the embedding algorithm. A more sophisticated example is the use of the bordering algorithm introduced in Keller (1977, 1983) in the corrector phase of the predictor–

corrector method. To avoid dealing with the arclength parameter, one can adopt a strategy of parameter switching, see, e.g., Rheinboldt (1980, 1981).

3. Aspects of implementations

Let us now turn to some of the practical aspects of implementing a predictor–corrector method.

3.1. Newton steps as corrector

A straightforward way of approximating a solution of the minimization problem in the predictor–corrector method (2.5) is given by the Newton step

$$\mathcal{N}_H(v) := v - H'(v)^+ H(v), \tag{3.1}$$

where $H'(v)^+$ denotes the Moore–Penrose inverse of $H'(v)$, see, e.g., Golub and van Loan (1989). Very commonly, an *Euler predictor*, i.e. a predictor step in the direction of the tangent to the curve is used:

$$v = u + ht(H'(u)), \tag{3.2}$$

where $h > 0$ represents the current stepsize.

The following algorithm sketches one version of the predictor–corrector method incorporating an approximate Euler predictor and one Newton-type iteration as a corrector step.

Algorithm 3.1 $u = \text{Euler\_Newton}(u, h)$

```
WHILE a stopping criterion is not met
        approximate A ≈ H'(u)
        v = u + ht(A)              % predictor step
        u = v − A⁺H(v)             % corrector step
        choose a new steplength h > 0
END
```

Discussions of Newton's method using the Moore–Penrose inverse can be found in several text books, e.g. Ortega and Rheinboldt (1970) or Ben-Israel and Greville (1974).

Let us first state a convergence result, see (5.2.1) in Allgower and Georg (1990), which ensures that this algorithm safely follows the solution curve under reasonable assumptions.

Theorem 3.2 Let $H : \mathbb{R}^{N+1} \to \mathbb{R}^N$ be a smooth map having zero as a regular value and let $H(u_0) = 0$. Denote by $c_h(s)$ the polygonal path, starting at u_0, going through all points u generated by Algorithm 3.1 with fixed steplength $h > 0$. Denote by $c(s)$ the corresponding curve in $H^{-1}(0)$ given by the initial value problem (2.9). For definiteness, we assume that

$c_h(0) = c(0) = u_0$, and that both curves are parametrized with respect to arclength. If the estimate $\|A - H'(u)\| = \mathcal{O}(h)$ holds uniformly for the approximation in the loop of the algorithm, then the following quadratic bounds hold uniformly for $0 \le s \le s_0$ and s_0 sufficiently small:

$$\|H(c_h(s))\| \le \mathcal{O}(h^2), \quad \|c_h(s) - c(s)\| \le \mathcal{O}(h^2).$$

Some major points which remain to be clarified are:

— How do we efficiently handle the numerical linear algebra involved in the calculation of $t(A)$ and $A^+H(v)$?
— How do we formulate efficient steplength strategies?

3.2. *The numerical linear algebra involved*

A straightforward and simple (but not the most efficient) way to handle the numerical linear algebra would be to use a QR factorization:

$$A^* = Q \begin{pmatrix} R \\ 0^* \end{pmatrix}, \tag{3.3}$$

where Q is an $(N+1, N+1)$ orthogonal matrix, and R is a nonsingular (N, N) upper triangular matrix. We assume that A is an $(N, N+1)$ matrix with maximal rank. If q denotes the last column of Q, then $t(A) = \sigma q$, where the orientation defined in (2.5) leads to the choice

$$\sigma = \operatorname{sign}\Big(\det Q \det R\Big). \tag{3.4}$$

Hence σ is easy to determine. The Moore–Penrose inverse of A can be obtained from the same decomposition in the following way:

$$A^+ = A^*(AA^*)^{-1} = Q \begin{pmatrix} (R^*)^{-1} \\ 0^* \end{pmatrix}. \tag{3.5}$$

Similar ideas apply if an LU decomposition is given:

$$PA^* = L \begin{pmatrix} U \\ 0^* \end{pmatrix}, \tag{3.6}$$

where L is a lower triangular $(N+1, N+1)$ matrix, U is an (N, N) upper triangular matrix, and P is a permutation matrix corresponding to partial pivoting which is, in general, necessary to improve the numerical stability. Let us first consider the calculation of $t(A)$. If y denotes the last column of $P^*(L^*)^{-1}$, then

$$t(A) = \sigma y / \|y\|, \quad \text{where } \sigma = \operatorname{sign}\Big(\det P \det L \det U\Big). \tag{3.7}$$

The Moore–Penrose inverse is obtained by

$$A^+ = (I - t(A)t(A)^*)P^*(L^*)^{-1} \begin{pmatrix} (U^*)^{-1} \\ 0^* \end{pmatrix}. \tag{3.8}$$

Hence, a calculation of $w = A^+ z$ amounts to essentially one forward-solving with U^*, one back-solving with L^*, and one scalar product with $t(A)$.

These methods are useful for small dense matrices A. However, in many applications of path following methods, the corresponding matrix A is large and sparse, and then this procedure is inefficient. Among such applications are the approximation of branches of nonlinear eigenvalue problems or the central path methods of linear and nonlinear programming. Let us point out some ideas which are useful in dealing with such situations.

In many applications, one encounters matrices A with the following structure:

$$A = \begin{pmatrix} L & b \end{pmatrix}, \tag{3.9}$$

where equations of the form $Lx = y$ permit a fast linear solver. If $\begin{pmatrix} c^* & d \end{pmatrix}$ denotes an additional row (typically generated via the last predictor direction), then a standard block elimination may be employed via the Schur complement.

Lemma 3.3 Let

$$s = d - c^* L^{-1} b$$

denote the Schur complement of L in the augmented matrix

$$\tilde{A} = \begin{pmatrix} L & b \\ c^* & d \end{pmatrix}.$$

Then

$$\det \tilde{A} = \det L \det s. \tag{3.10}$$

Furthermore, if $\tilde{A}$ is nonsingular, then

$$\tilde{A}^{-1} = \begin{pmatrix} L^{-1} + L^{-1} b s^{-1} c^* L^{-1} & -L^{-1} b s^{-1} \\ -s^{-1} c^* L^{-1} & s^{-1} \end{pmatrix}.$$

As an easy consequence, the tangent $t(A)$ is obtained via

$$t(A) = \sigma y / \|y\|, \tag{3.11}$$

where y denotes the last column of $\tilde{A}^{-1}$. The sign $\sigma \in \{\pm 1\}$ can either be obtained from an angle test with the previous predictor direction or from (3.10), since it can be shown that

$$\sigma = \operatorname{sign}\Big(\det L \det s\Big). \tag{3.12}$$

Note that the computational expense of determining $t(A)$ is roughly one application of the fast solver and a scalar product.

The Moore–Penrose inverse is obtained via

$$A^+ = (I - t(A)t(A)^*)(\tilde{A}^{-1})_N, \tag{3.13}$$

where $(\tilde{A}^{-1})_N$ denotes the submatrix consisting of the first N columns of $\tilde{A}^{-1}$. Hence, a calculation of $w = A^+ z$ amounts to essentially one additional call of the fast solver and two additional scalar products.

Among the fast solvers which are of importance here are direct solvers for sparse linear systems, or preconditioned iterative solvers such as conjugate-gradient or other Krylov methods, see, e.g., Freund, Golub and Nachtigal (1992).

This Schur complement construction is also valid if b, c and d are matrices (of appropriate size). This is of interest in parametric optimization, see Lundberg and Poore (1993). Watson (1986) and deSa, Irani, Ribbens, Watson and Walker (1992) discuss some numerical linear algebra aspects in the context of path following.

The popular bordering algorithm of Keller (1977), see also Chan (1984a), Keller (1983), Menzel and Schwetlick (1978, 1985), is related to these ideas. These approaches are akin to Keller's pseudo arclength method, in which the equation $H(v) = 0$ is extended by an additional parametrization condition $\mathcal{N}(u, v, h) = 0$ which is at least transversal to $H(v) = 0$ for small h, and often models an approximate arclength parametrization. This viewpoint is often convenient, in particular for structured problems.

3.3. Step length control and higher order predictors

The convergence considerations of Theorem 3.2 were carried out under the assumption that the steplength of the Algorithm 3.1 was uniformly constant throughout. This assumption is also typical for complexity studies, see Section 6. Such an approach is inefficient for any practical implementation. An efficient algorithm needs to incorporate an automatic strategy for controlling the steplength. In this respect the predictor–corrector methods are similar to the methods for numerically integrating initial value problems in ordinary differential equations. To some extent, the steplength strategy depends upon the accuracy with which it is desired to numerically trace a solution curve. Path following methods usually split into two categories:

— either the solution curve is to be approximated with some given accuracy, e.g. for plotting purposes; or
— the objective is just to safely follow the curve as fast as possible, until a certain point is reached, e.g. a zero point or critical point with respect to some additional functional defined on the curve.

We briefly sketch some ideas which are used to adjust the steplength.

Steplength control via error models. One method, due to Den Heijer and Rheinboldt (1981), is based upon an error model for the corrector iteration. For Newton corrector steps, such error models can be obtained by analysing the Newton–Kantorovich theory. The steplength is controlled by