COMPUTATIONAL MATHEMATICS
BANACH CENTER PUBLICATIONS, VOLUME 13
PWN–POLISH SCIENTIFIC PUBLISHERS
WARSAW 1984

EFFECTIVE METHODS FOR COMPUTING TURNING POINTS OF CURVES IMPLICITLY DEFINED BY NONLINEAR EQUATIONS*

HUBERT SCHWETLICK

Martin Luther University, Department of Mathematics,
4010 Halle, German Democratic Republic

1. Introduction

Many problems arising in applications can be described by systems of nonlinear equations

$$G(x, t) = 0, \quad G\colon D_G \subset \boldsymbol{R}^n \times \boldsymbol{R}^1 \to \boldsymbol{R}^1. \tag{1.1}$$

The vector $x \in \boldsymbol{R}^n$ denotes the variables of the problem while $t \in \boldsymbol{R}^1$ plays the role of an extra parameter which is of special interest. For some classes of problems the system (1.1) has for all t from a basic interval $[t^*, t^{**}]$ an at least locally unique solution $z(t)$, and $\{(z(t), t)\colon t \in [t^*, t^{**}]\}$ is a smooth solution path, see Fig. 1. Such regular problems can be solved by classical

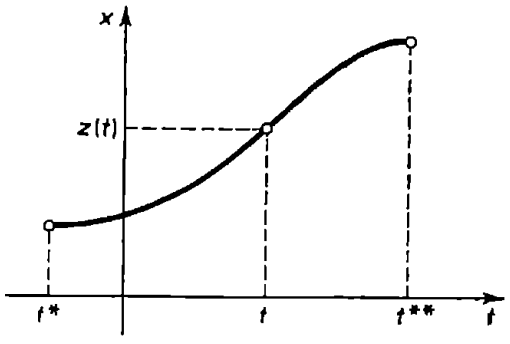

Fig. 1

continuation techniques using t as natural parameter and performing small steps along the t-axis, compare [12], [19] for a detailed discussion.

The simplest singular case is sketched in Figs 2, 3. In both examples there is a smooth one-dimensional solution path, too, but the x-part

* Invited Lecture, Semester on Computational Mathematics, Stefan Banach International Mathematical Center, Warsaw, Poland, May 1980.

is not uniquely determined by the t-part. Especially, the path has one or more so-called turning points $(\underline{x}, \underline{t})$, $(\bar{x}, \bar{t})$.

For applications we refer to [18], [3], [9] where problems of nonlinear mechanics are treated which lead directly or via discretization of certain integral or differential equations to systems of type (1.1). Other examples come from chemical engineering, see [10], or from the analysis of resistive electrical circuits, see [4].

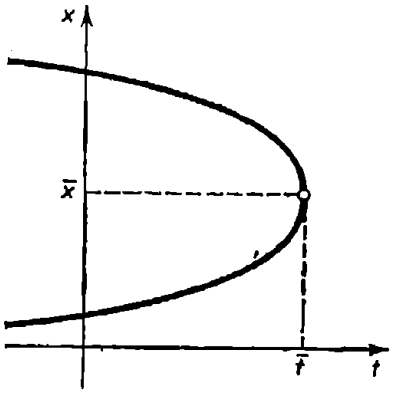

Fig. 2

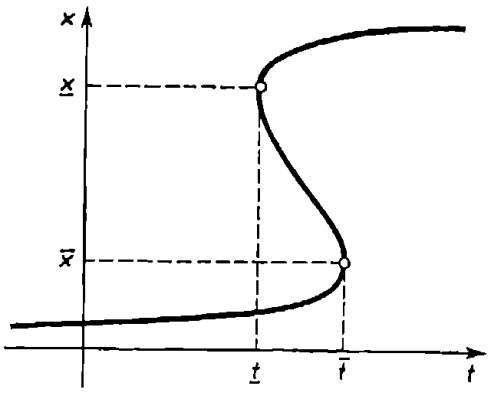

Fig. 3

In all problems mentioned there are of interest the turning points of the solution path characterizing certain critical states of the system. Often it is also desirable to have a pointwise approximation of the path near to the turning point or over the whole basic interval.

In this paper a number of effective methods for computing turning points is described, analyzed, and compared with other methods known from literature. Since the most real life problems are both of high dimension and sparse the following properties should be met by a reasonable algorithm:

(P_1) The subproblems to be solved per step should only be quadratic regular linear systems so that standard software as packages for solving sparse linear systems can be used.

(P_2) Only values of G and of its first partial derivatives should be used in order to avoid the generation of subroutines for computing second order derivatives.

(P_3) The algorithm should converge superlinearly.

(P_4) The number of scalar functions to be evaluated and the combinatorial costs per step should be small in comparison with the order of convergence (low complexity).

In the following only algorithms having all these desirable properties will be considered.

First of all, in Chapter 2 some basic notions and notations are introduced. Chapter 3 is devoted to the so-called indirect methods. These methods are based on a local parametrization $\{(x_k(\tau), t_k(\tau)): \tau \in T_k \subset \boldsymbol{R}^1\}$ of the solution path in a neighborhood of a known solution point (x^k, t_k) of (1.1), the last iterate. By using such a parametrization the turning point is characterized by a scalar condition as, e.g.,

$$t_k(\tau) \to \text{Extremum}. \tag{1.2}$$

The problem (1.2) is approximately solved by applying one or more steps of an appropriate scalar algorithm leading to a new parameter τ_k that defines the next iterate $x^{k+1} = x_k(\tau_k)$, $t_{k+1} = t_k(\tau_k)$ on the path. Due to the implicit definition of the parametrization, the computation of (x^{k+1}, t_{k+1}) requires the solution of a nonlinear system and, hence, is an infinite process. Therefore, indirect methods are at least from a theorctical point of view not implementable in a strong sense. Nevertheless they work well in practice.

In Chapter 4 the direct methods are described. These methods characterize the turning point by (1.1) and an additional scalar equation

$$\varphi(x, t) = 0, \quad \varphi: D_\varphi \subset \boldsymbol{R}^n \times \boldsymbol{R}^1 \to \boldsymbol{R}^1 \tag{1.3}$$

in terms of (x, t) without using any parametrization. The combined system (1.1), (1.3) is solved by a fast algorithm as Newton's method or some modification of it. An example of the bordering equation (1.3) is

$$\det\left(\partial_1 G(x, t)\right) = 0 \tag{1.4}$$

that has to be satisfied at a turning point. Here $\partial_1 G$ denotes the Jacobian of G with respect to x, see Chapter 2. Some other conditions more reliable for computation are proposed and a special Newton-like method is developed which holds the number of scalar function evaluations per step on a low level.

The paper ends with some numerical examples listed in Chapter 5.

Most of the material presented is the result of common research work done by the author and G. Pönisch and can be found in [15], [16] and in the dissertation [13] of the latter. Further references are given in the text.

2. Basic notions and notations

For simplicity of notation we set $u := (x, t)^T \in \boldsymbol{R}^{n+1}$ and $H(u) := G(x, t)$ for all $u = (x, t)^T$ from the domain of G. Then (1.1) reads as

$$H(u) = 0, \quad H: D \subset \boldsymbol{R}^{n+1} \to \boldsymbol{R}^n.$$

Let $H'(u)$ be the $(n, n+1)$-Jacobian $(\partial H_i/\partial u_j)$ of H composed of the (n, n)-Jacobian $\partial_1 G(x, t) = (\partial G_i/\partial x_j)$ of G with respect to the first argument x and the $(n, 1)$-derivative $\partial_2 G(x, t) = (\partial G_i/\partial t)$ of G with respect to t according to

$$H'(u) = \big(\partial_1 G(x, t)\big|\ \partial_2 G(x, t)\big) =: \big(\partial_1 G(u)\big|\ \partial_2 G(u)\big).$$

Further, let e^i denote the ith coordinate vector, and $\|\cdot\|$ means either the Euclidean norm of a vector or the spectral norm of a matrix, respectively. Finally, define

$$\mathscr{L} := \{u \in D\colon\ H(u) = 0\}.$$

2.1. DEFINITION. A point $\bar{u} \in \mathscr{L}$ is called a *turning point (from the right)* of $\mathscr{L}$ if there is a $\delta > 0$ such that

$$(e^{n+1})^T(u - \bar{u}) \leqslant 0 \qquad \forall\ u \in S(\bar{u}, \delta) \cap \mathscr{L}. \tag{2.1}$$

If the "$\leqslant$"-sign is replaced by "$\geqslant$" then $\bar{u}$ is called a *turning point from the left.*

In order to get necessary and sufficient conditions for $\bar{u}$ to be a turning point a parametrization of $\mathscr{L}$ in the neighborhood of $\bar{u}$ is introduced. Since such a parametrization is also used as an algorithmic tool by the indirect methods, it is described for an arbitrary reference point $z \in \mathscr{L}$.

2.2. LEMMA. *Let $H\colon D \subset \boldsymbol{R}^{n+1} \to \boldsymbol{R}^n$ have a continuous derivative H' on D, let $z \in \operatorname{int}(D)$ with $H(z) = 0$ be given and suppose*

$$\operatorname{rank} H'(z) = n. \tag{2.2}$$

Let $v \in \boldsymbol{R}^{n+1}$ be the with exception of the sign unique solution of

$$H'(z)v = 0, \qquad \|v\| = 1, \tag{2.3}$$

and let $\omega \in (0, 1]$ be a fixed number. Then there are constants ε, $\delta > 0$ such that the augmented system

$$H(u) = 0, \tag{2.4a}$$

$$r^T(u - z) = \tau \tag{2.4b}$$

has for all $\tau \in T := (-\varepsilon, \varepsilon)$ and for all $r \in R(\omega, v)$,

$$R(\omega, v) := \{r \in \boldsymbol{R}^{n+1}\colon\ \|r\| = 1 \text{ and } |r^T v| \geqslant \omega\}, \tag{2.5}$$

a solution $u = w(\tau) = w_{z,r}(\tau)$ that is unique in $S(z, \delta) := \{u \in \boldsymbol{R}^{n+1}\colon \|u - z\| < \delta\}$. The function $w(\cdot)\colon T \to \boldsymbol{R}^{n+1}$ is continuously differentiable on T, and $\dot{w}$ is the unique solution of

$$\begin{bmatrix} H'\big(w(\tau)\big) \\ r^T \end{bmatrix} \dot{w}(\tau) = e^{n+1}. \tag{2.6}$$

If, in addition, H is twice differentiable on D then the same is true for $\dot{w}$ on T, and w solves

$$\begin{bmatrix} H'(w(\tau)) \\ r^T \end{bmatrix} \ddot{w}(\tau) = -\begin{bmatrix} H''(w(\tau))\dot{w}(\tau)\dot{w}(\tau) \\ 0 \end{bmatrix}. \tag{2.7}$$

These results are direct implications of the implicit function theorem applied to the system (2.4).

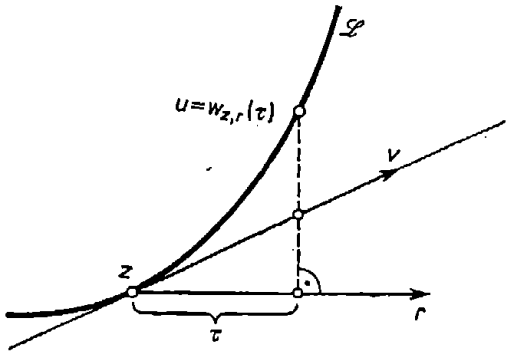

Fig. 4

The arc $\{u = w_{z,r}(\tau)\colon \tau \in T\}$ can be considered as a local parametrization of $\mathscr{L}$ near z, and the parameter τ is the projection of $w(\tau) - z$ on the parameter axis r, see Fig. 4. A parametrization of type (2.4) has been independently proposed in [18], [4], [7], [11], see also [6], [3] for related but somewhat different ideas.

Because of (2.3) there holds

$$\dot{w}(0) = \frac{v}{r^T v} \tag{2.8}$$

so that v has the tangent direction of $\mathscr{L}$ at z. Note, further, that the matrix

$$B(z, r) := \begin{bmatrix} H'(z) \\ r^T \end{bmatrix} \tag{2.9}$$

occurring for $\tau = 0$ on the left side of (2.6), (2.7) is regular if and only if $r^T v \neq 0$. In the regular case there holds

$$\operatorname{cond}(B(z, r)) \leqslant \frac{2}{|r^T v|} \max\{\|H'(z)\|, 1\} \max\{\|H'(z)^+\|, 1\}, \tag{2.10}$$

see [11], [19]. In the sense of (2.10) the tangent direction v is the optimal parameter axis which is intuitively clear from Fig. 4, compare [18] for a similar result using the Hadamard condition number of $B(z, r)$.

Now let the assumptions of Lemma 2.2 be fulfilled at a turning point $z = \bar{u} \in \mathscr{L}$, let $\bar{v}$ be a solution of (2.3) at $z = \bar{u}$, and take $r = \bar{v}$ as parameter axis. Substituting $u = \bar{w}(\tau) := w_{\bar{u},\bar{v}}(\tau)$ in (2.1) we get

$$t(\tau) := (e^{n+1})^T \bar{w}(\tau) \leqslant (e^{n+1})^T \bar{w}(0) = (e^{n+1})^T \bar{u} = \bar{t}$$

for all $\tau \in T$. Therefore,

$$t(\tau) \to \text{Extremum} \tag{2.11}$$

is a necessary condition for $\bar{u}$ to be a turning point of $\mathscr{L}$. Differentiating with respect to τ leads to

$$\dot{t}(\tau) = 0 \tag{2.12}$$

as a further necessary condition to be fulfilled at $\bar{\tau} = 0$. Because of (2.8) there holds $\dot{\bar{w}}(0) = \bar{v}$ so that (2.12) is equivalent to

$$(e^{n+1})^T \bar{v} = 0 \tag{2.13}$$

as a characterization in terms of H.

A sufficient condition for $\bar{\tau} = 0$ to be a strong minimizer or maximizer of $t(\tau)$ is

$$\bar{\theta} := \ddot{t}\,(0) = (e^{n+1})^T \ddot{\bar{w}}(0) = -(e^{n+1})^T B(\bar{u}, \bar{v})^{-1} \begin{bmatrix} H''(\bar{u})\bar{v}^2 \\ 0 \end{bmatrix} \neq 0. \tag{2.14}$$

Therefore, the following assumptions are supposed to be satisfied in future.

2.3. Assumption (TP). The function $H\colon D \subset \boldsymbol{R}^{n+1} \to \boldsymbol{R}^n$ is said to *satisfy assumption* (TP) if H is twice continuously differentiable on D, there is a $u \in \operatorname{int}(D)$ with $H(u) = 0$ and $\operatorname{rank} H'(u) = n$, and with the exception of the sign unique solution $\bar{v}$ of $H'(\bar{u})\bar{v} = 0$, $\|\bar{v}\| = 1$ fulfills the first order condition (2.13) as well as the second order condition (2.14).

Note that the full rank condition can not be dispensed with for $\bar{u}$ may be a bifurcation point in the rankdeficient case.

3. Indirect methods

Let a point $u^k \in \mathscr{L}$ near $\bar{u}$ be given so that $H'(u^k)$ has full rank n. The indirect methods to be discussed in this chapter are based on the local parametrization $u = w_k(\tau) := w_{u^k, r^k}(\tau)$ of $\mathscr{L}$ defined by Lemma 2.2 with $z = u^k$ and an appropriate parameter axis r^k. Due to this lemma there are constants $\varepsilon_k, \delta_k > 0$ such that $u = w_k(\tau)$ is for all $\tau \in T_k := (-\varepsilon_k, \varepsilon_k)$ the unique solution of the system

$$H(u) = 0, \tag{3.1a}$$

$$(r^k)^T (u - u^k) = \tau \tag{3.1b}$$

in $S(u^k, \delta_k)$ provided that r^k satisfies $\|r^k\| = 1$ and

$$|(r^k)^T v^k| \geqslant \omega > 0 \tag{3.2}$$

with a fixed $\omega \in (0, 1]$ and the tangent direction v^k being one solution of

$$H'(u^k)v = 0, \quad \|v\| = 1. \tag{3.3}$$

Obviously, $\bar{\tau}_k := (r^k)^T(\bar{u} - u^k)$ is the parameter belonging to the turning point. If u^k is sufficiently near $\bar{u}$ then we have $\bar{\tau}_k \in T_k$ so that $\bar{u}$ can be described as $\bar{u} = w_k(\bar{\tau}_k)$. The idea of the indirect methods consists now in determining an estimate τ_k for $\bar{\tau}_k$ that defines the next iterate $u^{k+1} = w_k(\tau_k)$ on $\mathscr{L}$ as a better approximation to $\bar{u}$ than the current iterate u^k.

Before describing some ways for choosing τ_k the problem of computing $u^{k+1} = w_k(\tau_k)$ for a given value of τ_k will be discussed. As, e.g., in [18], [19], [11], u^{k+1} can be computed by applying Newton's method to the system (3.1) using u^k as initial point. Due to the full rank of $H'(u_k)$ and the definition of v^k the first Newton iterate $u^{k,0}$ can be written as

$$u^{k,0} := u^k + \lambda_k v^k, \quad \lambda_k := \frac{\tau_k}{(r^k)^T v^k}, \tag{3.4}$$

and the following iterates $u^{k,i}$ are the solutions of the linear systems

$$\begin{bmatrix} H'(u^{k,i}) \\ (r^k)^T \end{bmatrix}(u^{k,i+1} - u^{k,i}) + \begin{bmatrix} H(u^{k,i}) \\ 0 \end{bmatrix} = 0 \quad (i = 0, 1, \ldots). \tag{3.5}$$

The first Newton step can be considered as a predictor step using Euler's formula whereas the further iterations play the role of a corrector iteration moving the predictor point on the path $\mathscr{L}$. Note that all iterates $u^{k,i}$ $(i = 0, 1, \ldots)$ satisfy (3.1b), see Fig. 5.

Now the basic algorithm of indirect type can be formulated.

3.1. Basic algorithm

Step 0: Choose m initial points $u^j \in \boldsymbol{R}^{n+1}$ with $H(u^j) = 0$ $(j = 0, -1, \ldots, -m+1)$ and an $r^{-1} \in \boldsymbol{R}^{n+1}$ with $\|r^{-1}\| = 1$, set $k := 0$.

Step 1: Determine $v^k \in \boldsymbol{R}^{n+1}$ such that

$$H'(u)v^k = 0, \quad \|v^k\| = 1 \quad \text{and} \quad (r^{k-1})^T v^k > 0. \tag{3.6}$$

Step 2: If $(e^{n+1})^T v^k = 0$, stop.

Step 3: Choose $r^k \in \boldsymbol{R}^{n+1}$ with $\|r^k\| = 1$ and $(r^k)^T v^k > 0$.

Step 4: Determine a stepsize $\tau_k \in \boldsymbol{R}^1$.

Step 5: Compute

$$u^{k,0} := u^k + \lambda_k v^k, \quad \lambda_k := \frac{\tau_k}{(r^k)^T v^k}.$$

Step 6: Determine $u^{k+1} = w_k(\tau_k)$ as limit point of the sequence $\{u^{k,i}\}$ defined by $u^{k,i+1} := u^{k,i} - s^{k,i}$, $s^{k,i}$ solution of the linear system

$$\begin{bmatrix} H'(u^{k,i}) \\ (r^k)^T \end{bmatrix} s^{k,i} = \begin{bmatrix} H(u^{k,i}) \\ 0 \end{bmatrix} \quad (i = 0, 1, \ldots). \tag{3.7}$$

Step 7: Set $k := k+1$ and go to Step 1.

The number m of initial points depends on the specific method for determining τ_k. The vector r^{-1} defines the initial orientation of the tangent vector v^0, and Step 2 is a termination criterion based on (2.13).

In the following some ways for determining τ_k are proposed. Let the algorithm be at stage k. As in Chapter 2 the unknown parameter $\bar{\tau}_k$ belonging to $\bar{u}$ can be defined by the condition

$$\varphi_k(\tau) := t_k(\tau) = (e^{n+1})^T w_k(\tau) \to \text{Extremum}. \tag{3.8}$$

Therefore any method for maximizing or minimizing a scalar function as used in optimization can be taken for defining τ_k. In order to make the whole algorithm superlinearly convergent only interpolative methods will be considered. At first the method of interpolating parabolas is used. The function φ_k is interpolated at

$$\tau_{k,j} := (r^k)^T (u^j - u^k) \tag{3.9}$$

corresponding to the points u^j by a quadratic polynomial p_k according to

$$p_k(\tau_{k,j}) = \varphi_k(\tau_{k,j}) = (e^{n+1})^T u^j = t_j \quad (j = k, k-1, k-2).$$

Then the value $\tau = \tau_k$ that makes $p_k(\tau)$ extremal is determined from the linear equation $\dot{p}_k(\tau) = 0$. This construction yields the

3.2. Stepsize rule (QIP) using quadratic interpolation of φ_k

Step 4: Set

$$\tau_k := -\frac{\delta^{(1)}_{k,k-1} - \tau_{k,k-1}\,\delta^{(2)}_{k,k-1,k-2}}{2\,\delta^{(2)}_{k,k-1,k-2}} \qquad \text{where}$$

$$\delta^{(1)}_{k,k-1} := \frac{t_k - t_{k-1}}{-\tau_{k,k-1}}, \qquad \delta^{(1)}_{k-1,k-2} := \frac{t_{k-1} - t_{k-2}}{\tau_{k,k-1} - \tau_{k,k-2}},$$

$$\delta^{(2)}_{k,k-1,k-2} := \frac{\delta^{(1)}_{k,k-1} - \delta^{(1)}_{k-1,k-2}}{-\tau_{k,k-2}}$$

are the divided differences of φ_k.

In 3.2 only the function values $t_j = \varphi_k(\tau_{k,j})$ of φ_k are used. Note, however, that the tangent directions v^j of $\mathcal{L}$ at u^j are also available.

Because of (2.6) and $u^j = w_k(\tau_{k,j})$ there holds

$$\begin{bmatrix} H'(u^j) \\ (r^k)^T \end{bmatrix} \dot{w}_k(\tau_{k,j}) = e^{n+1}.$$

On the other hand v^j solves $H'(u^j)v = 0$ so that $w_k(\tau_{k,j})$ must be a multiple of v^j due to the full rank of $H'(u^j)$. Therefore, there holds

$$\dot{w}_k(\tau_{k,j}) = \frac{v^j}{(r^k)^T v^j}$$

and, consequently,

$$\dot{\varphi}_k(\tau_{k,j}) = \frac{(e^{n+1})^T v^j}{(r^k)^T v^j} =: \dot{\varphi}_{k,j}. \tag{3.10}$$

This fact suggests the use of a cubic polynomial p_k interpolating φ_k and $\dot{\varphi}_k$ at the last two points according to

$$\begin{aligned} p_k(\tau_{k,j}) &= \varphi_k(\tau_{k,j}) \\ \dot{p}_k(\tau_{k,j}) &= \dot{\varphi}_k(\tau_{k,j}) \end{aligned} \qquad (j = k, k-1). \tag{3.11}$$

Then $\tau = \tau_k$ is the zero of the quadratic equation $\dot{p}_k(\tau) = 0$ which is nearest to $\tau_{k,k} = 0$. This principle gives

3.3. Stepsize rule (CIP) using cubic interpolation of φ_k

Step 4: Set

$$\tau_k := -\frac{2\dot{\varphi}_{k,k}}{a_k(1+\beta_k)} \quad \text{where}$$

$$a_k := \delta_k - 3\tau_{k,k-1}\delta_k^{(3)}, \qquad \beta_k := \sqrt{1 - \frac{12\dot{\varphi}_{k,k}\delta_k^{(3)}}{(a_k)^2}},$$

$$\delta_k := \frac{\dot{\varphi}_{k,k} - \dot{\varphi}_{k,k-1}}{-\tau_{k,k-1}},$$

$$\delta_k^{(3)} := \frac{1}{(\tau_{k,k-1})^2}\left\{\dot{\varphi}_{k,k} - 2\frac{t_k - t_{k-1}}{-\tau_{k,k-1}} + \dot{\varphi}_{k,k-1}\right\}.$$

Differentiating (3.8) we get

$$\dot{\varphi}_k(\tau) = 0. \tag{3.12}$$

In order to solve (3.12) one step of Newton's method from $\tau_{k,k} = 0$ could be used but this would require $\ddot{\varphi}_k(0)$ and, therefore, second derivatives of H, see (2.7). Using the ideas developed in Chapter 4 the terms involving H'' could be approximated by using 4 values of H. However,

this possibility will not be discussed here since there are simpler algorithms avoiding the explicit use of H'' as, e.g., the secant method with nodes $\tau_{k,j}$ $(j = k, k-1)$. This gives the following stepsize algorithm.

3.4. Stepsize rule (LIP1) using linear interpolation of $\dot{\varphi}_k$

Step 4: Set

$$\tau_k := -\frac{\dot{\varphi}_{k,k}}{\delta_k}, \qquad \delta_k := \frac{\dot{\varphi}_{k,k} - \dot{\varphi}_{k,k-1}}{-\tau_{k,k-1}}.$$

All the stepsize rules described are based on the characterization of $\bar{\tau}_k$ by the extremal principle (3.8) and the associated necessary condition (3.10). Another basis is the condition

$$\psi_k(\tau) := \det\left(J\left(w_k(\tau)\right)\right) = 0, \qquad J(u) := \partial_1 G(u) \tag{3.13}$$

following from the fact that $J(u)$ must be singular at a turning point. Indeed, define

$$v(u, r^k) := B(u, r^k)^{-1} e^{n+1}. \tag{3.14}$$

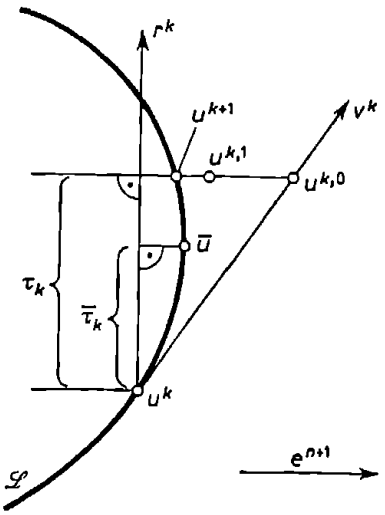

Fig. 5

The matrix $B(u, r^k)$ is regular if u is near $\bar{u}$, and $(r^k)^T \bar{v} \neq 0$ so that v is well-defined. Note that $v = v(u, r^k)$ solves the linear system

$$B(u, r^k) v = \left[\begin{array}{c|c} J(u) & \partial_2 G(u) \\ \hline & (r^k)^T \end{array}\right] v = e^{n+1}.$$

Applying Cramer's rule to the last component

$$\varphi(u, r^k) := (e^{n+1})^T B(u, r^k)^{-1} e^{n+1} \tag{3.15}$$

of v we get

$$\varphi(u, r^k) = \frac{\det\left(B(u, e^{n+1})\right)}{\det\left(B(u, r^k)\right)}$$

and by substituting $u = w_k(\tau)$ the representation

$$\psi_k(\tau) = \det\left(B\left(w_k(\tau), e^{n+1}\right)\right) = \det\left(B\left(w_k(\tau), r^k\right)\right) \cdot \varphi\left(w_k(\tau), r^k\right). \tag{3.16}$$

For $\tau = \bar{\tau}_k$ there holds $\varphi\left(w_k(\bar{\tau}_k), r^k\right) = (e^{n+1})^T \bar{v}/(r^k)^T \bar{v}$ so that (2.13) implies $\psi_k(\bar{\tau}_k) = 0$ as stated above. Note, further, that (2.14) implies the zero $\bar{\tau}_k$ of ψ_k to be simple in the sense of $\dot{\psi}_k(\bar{\tau}_k) \neq 0$. Indeed, differentiating (3.16) we get

$$\dot{\psi}_k(\bar{\tau}_k) = \det\left(B(\bar{u}, r^k)\right) \cdot \nabla_u \varphi(\bar{u}, r^k)^T \dot{w}_k(\bar{\tau}_k)$$

since the second term coming from the product rule vanishes. Considering

$$\nabla_u \varphi(u, r)^T h = -(e^{n+1})^T B(u, r)^{-1} \begin{bmatrix} H''(u) v(u, r) h \\ 0 \end{bmatrix} \tag{3.17}$$

for all $h \in \boldsymbol{R}^{n+1}$ and $\dot{w}_k(\bar{\tau}_k) = v(\bar{u}, r^k) = \bar{v}/(r^k)^T \bar{v}$, we get

$$\dot{\psi}_k(\bar{\tau}_k) = \det\left(B(\bar{u}, r^k)\right) \cdot \bar{\Theta} / \left((r^k)^T \bar{v}\right)^2 \neq 0.$$

Therefore, $\bar{\tau}_k$ is well behaved, and one step of the secant method applied to (3.13) at $\tau_{k,j}$ $(j = k, k-1)$ should give a better parameter τ_k.

3.5. Stepsize rule (LIP2) using linear interpolation of ψ_k

Step 4: Set

$$\tau_k := \left(\frac{\psi_k(\tau_{k,k}) - \psi_k(\tau_{k,k-1})}{\tau_{k,k-1}}\right)^{-1} \psi_k(\tau_{k,k}).$$

Obviously also other stepsize rules are possible but the four given above seem to be the most effective ones.

In the following some remarks concerning the implementation of the methods will be made.

The first problem is to get the 2 (for 3.3, 3.4, 3.5) or 3 (for 3.2) initial points u^0, u^{-1}, u^{-2} required by 3.1. The natural way for doing this is beginning from a u^0 with $H(u^0) = 0$ to use a path following algorithm of the same structure as 3.1 but with a different rule for choosing τ_k, see, e.g., [11], [19]. As a successful rule the formula

$$\tau_k := \min\{\tau_{k-1}/\alpha, \tau_{\max}\} \tag{3.18}$$

has proved where $\tau_{\max}$ is a user provided stepsize bound and $\alpha \in (0, 1)$, say, $\alpha = 1/2$. If Step 6 is successful in the sense that $B(u^{k,i}, r^k)$ is regular and the descent condition

$$\|H(u^{k,i+1})\| \leqslant \beta \|H(u^{k,i})\| \quad (i = 0, 1, \ldots) \tag{3.19}$$

is satisfied with a $\beta \in (0, 1)$, say $\beta = 0.9$, for subsequent i then the corrector iteration is considered as convergent and finished if a termination criterion such as

$$\|H(u^{k,i+1})\| \leqslant \varepsilon \quad \text{or} \quad \|s^{k,i}\| \leqslant \varepsilon$$

is fulfilled. Otherwise Steps 5, 6 are repeated with a smaller stepsize $\alpha\tau_k$.

During the path following phase, the points $u^k \in \mathscr{L}$ are observed with respect to the behaviour of the t-component $t_k = (e^{n+1})^T u^k$. If $(t_{k+1} - -t_k)(t_k - t_{k-1}) < 0$ then there is a turning point between the points u^{k-1} and u^{k+1}. Note that a change of sign of the determinants $\det\big(J(u^k)\big)$ indicates the presence of a turning point, too. The last iterates u^{k+1}, u^k, u^{k-1} may serve as initial points u^0, u^{-1}, u^{-2} for the special turning point algorithms of this chapter.

In the neighborhood of a turning point the stepsizes of the path following algorithm will be small, in general, so that the last iterates will be good enough to make Step 6 successful. Otherwise, the path following algorithm should be restarted at the last point with a smaller stepsize.

Step 1 should be implemented as follows, remind (P_1) claimed in the introduction.

Step 1′: Solve the quadratic system

$$B(u^k, r^{k-1})\,\tilde{v} = e^{n+1}$$

$$\text{for } \tilde{v} = v(u^k, r^{k-1}) =: \tilde{v}^k.$$

Step 1″: Set $v^k := \tilde{v}^k / \|\tilde{v}^k\|$.

Clearly $(r^{k-1})^T v^k > 0$ is then satisfied because of $(r^{k-1})^T \tilde{v}^k = 1$. Having computed u^k and v^k all information required by 3.1 and the rules 3.2, 3.3, 3.4 is readily available. For realizing 3.5 the determinants $\psi_k(\tau_{k,j})$ must be evaluated. Note, first of all, that the implementation of Step 1 as described requires a LR-factorization of $B(u^k, r^{k-1})$ and hence

$$\delta_k := \det\big(B(u^k, r^{k-1})\big)$$

can directly be calculated during the Gaussian elimination. Since the identity

$$\det\big(B(u, r)\big) = \det\big(B(u, s)\big) \cdot [r^T v(u, s)]$$

holds for all $u, r, s \in \boldsymbol{R}^{n+1}$, $\|r\| = \|s\| = 1$, such that $B(u, s)$ is regular because of $v(u^k, r) = \tilde{v}^k / r^T \tilde{v}^k$, it follows that

$$\psi_k(\tau_{k,k}) = \delta_k \cdot [(e^{n+1})^T \tilde{v}^k],$$

$$\psi_k(\tau_{k,k-1}) = \delta_{k-1} \cdot [(e^{n+1})^T \tilde{v}^{k-1}].$$

Moreover, Step 6 should be monitored by (3.19) as mentioned in the path following algorithm.

Finally the choice of r^k will be discussed. The optimal choice in the sense of (2.10) would be

Step 3: Set

$$r^k := v \tag{3.20}$$

which should be used for problems of small or medium dimension. If n is large and H' is sparse a choice as

Step 3: Determine $j_k \in \{1, \ldots, n+1\}$ such that

$$|(e^{j_k})^T v^k| = \max\{|(e^j)^T v^k| : j = 1, \ldots, n+1\}.$$

Set

$$r^k := \operatorname{sign}\left((e^{j_k})^T v^k\right) \cdot e^{j_k} \tag{3.21}$$

would be preferable because r^k is then sparse which is not the case for v^k.

In sparse problems it is often desirable to use the same pivoting sequence for the Gaussian elimination in consecutive steps. In this case (3.21) can be weakened to

Step 3: Determine j_k as in (3.21), set

$$r^k := \textbf{if } |(r^{k-1})^T v^k| \geqslant \omega_0 |(e^{j_k})^T v^k| \textbf{ then } r^{k-1} \textbf{ else } \operatorname{sign}\left((e^{j_k})^T v^k\right) \cdot e^{j_k} \tag{3.22}$$

with a fixed $\omega_0 \in (0, 1)$, say $\omega_0 = 0.1$, and an initial $r^{-1} = e^{j-1}$.

In all cases the condition $|(r^k)^T v^k| \geqslant \omega_0/\sqrt{n} =: \omega > 0$ is satisfied so that $r^k \in R(\omega, v^k)$.

A reduction of work is possible, in general, by replacing Newton's method in Step 6 by its modified form, i.e. by replacing (3.7) by

$$\begin{bmatrix} H'(u^k) \\ (r^k)^T \end{bmatrix} s^{k,i} = \begin{bmatrix} H(u^{k,i}) \\ 0 \end{bmatrix} \quad (i = 0, 1, \ldots) \tag{3.23}$$

so that only one evaluation of H' and one LR-factorization are to be performed per step. Note, further, that the interpolation principles used in 3.2 and 3.3 can also be used to replace the tangent line at u^k by a polynomial fit $\tilde{w}_k$ to w_k in order to get a better predictor point $u^{k,0} := \tilde{w}_k(\tau_k)$ and, therefore, to reduce the number of corrector iterations.

The convergence behaviour of the algorithms dicussed in this chapter is described by the following theorem.

3.6. THEOREM. *Let* $H: D \subset \mathbf{R}^{n+1} \to \mathbf{R}^n$ *satisfy assumption* (TP) *and have Lipschitz continuous derivatives up to order* 2 (*for* 3.4 *and* 3.5), 3 (*for* 3.2), *and* 4 (*for* 3.3), *respectively. Then the basic algorithm* 3.1 *combined with the stepsize rules* 3.3, 3.4, 3.5 *and with the choice of* r^k *according to* (3.20), (3.21) *or* (3.22) *is well-defined and terminates after a finite number of steps at* $\bar{u}$, *or else it is infinite and converges Q-superlinearly toward* $\bar{u}$

provided that the initial points u^0, u^{-1} are sufficiently near $\bar{u}$. The speed of convergence is characterized by the inequalities

$$\|u^{k+1}-\bar{u}\| \leqslant \begin{cases} C\,\|u^k-\bar{u}\|\,\|u^{k-1}-\bar{u}\|^2 & \text{for 3.3,} \\ C\,\|u^k-\bar{u}\|\,\|u^{k-1}-\bar{u}\| & \text{for 3.4, 3.5} \end{cases}$$

corresponding to the R-orders $\varkappa = 2$ (for 3.3) and $\varkappa = (1+\sqrt{5})/2 = 1.62$ (for 3.4, 3.5).

If 3.1 combined with 3.2 is well defined for all k and convergent toward $\bar{u}$ then the R-order is $\varkappa = 1.32$ which is the dominant zero of $\varkappa^3 = \varkappa+1$.

For the proof in case $r^k = v^k$ see [13], [15], the generalization for $r^k \in R(\omega, v^k)$ is straightforward.

Note, at this point, that the first indirect method for computing turning points seems to be described in [18] but without any analysis. In [22] an interpolatory method is given that uses an instable parametrization and, again, without analysis. The combination 3.1/3.4 has been proposed independently in [8] without analysis and in [13], [20] in the form given here. Further indirect methods are sketched in [21].

4. Direct methods

As mentioned in the introduction the direct methods are based on a characterization of the turning point $\bar{u}$ by a nonlinear system of the structure

$$H(u) = 0, \tag{4.1}$$

$$\varphi(u) = 0 \tag{4.2}$$

with a scalar function $\varphi\colon D \subset \boldsymbol{R}^{n+1} \to \boldsymbol{R}^1$ without any explicit reference to the path $\mathscr{L}$ and a parametrization of it. Because of the first order property (2.13) the function φ must involve first derivatives of H as is the case for

$$\varphi(u) := \det\big(J(u)\big) \tag{4.3}$$

proposed in [10]. Applying Newton's method to the system (4.1), (4.3) we get an algorithm that requires $\nabla_u \varphi(u)$ and, therefore, the computation of all second derivatives of H and of all minors of $J(u)$. Hence, as mentioned in [10], this strategy can only be recommended for small n.

An essential reduction of work is possible by following an idea described in [1]. There only the first part (4.1) is linearized at u^k yielding

$$H(u^k)+H'(u^k)(u-u^k) = 0. \tag{4.4}$$

If u^k is near $\bar{u}$ then $H'(u^k)$ has full rank, and (4.4) has a straight line

$$u = u^k + s^k + \lambda v^k, \quad \lambda \in \boldsymbol{R}^1 \tag{4.5}$$

as solution where v^k is a solution of the homogeneous system, say, of unit norm

$$H'(u^k)v^k = 0, \quad \|v^k\| = 1, \tag{4.6}$$

and s^k is a solution of the inhomogeneous system, say, the minimum norm solution $s^k = -H'(u^k)^+ H(u^k)$ defined by

$$H(u^k) + H'(u^k)s^k = 0, \quad (v^k)^T s^k = 0. \tag{4.7}$$

Note that v^k is the tangent direction of the translated path $\{u \in \boldsymbol{R}^{n+1}: H(u) = H(u^k)\}$ at u^k, see Fig. 6.

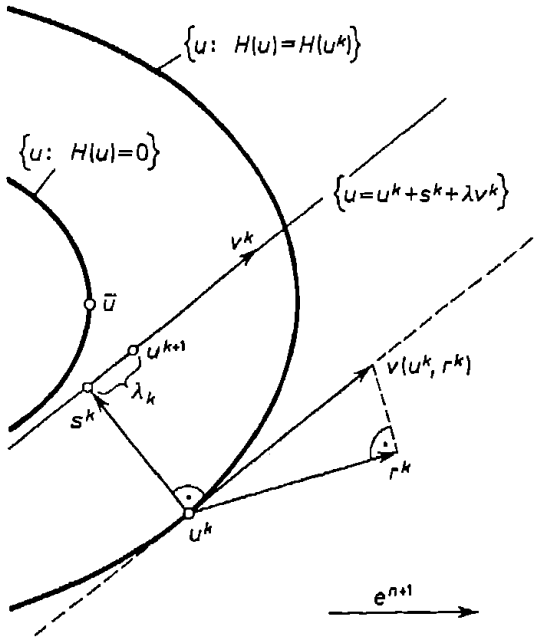

Fig. 6

Now (4.5) is substituted into (4.2) leading to the scalar equation

$$\varphi(u^k + s^k + \lambda v^k) = 0 \tag{4.8}$$

to be solved for λ. In order to avoid the computation of $\partial\varphi/\partial\lambda$ and, hence, of second derivatives one step of the secant method with the nodes $\lambda = 0$ and $\lambda = \mu_k$, $\mu_k \neq 0$ being an appropriate discretization stepsize, is applied to (4.8). This gives

$$\lambda_k := -\mu_k \frac{\varphi(u^k + s^k)}{\varphi(u^k + s^k + \mu_k v^k) - \varphi(u^k + s^k)}$$

as expression for λ_k defining the next iterate

$$u^{k+1} := u^k + s^k + \lambda_k v^k,$$

see Fig. 6. The computation of λ_k requires the evaluation of φ at two points and, hence, two Jacobians H'.

In the following a somewhat modified algorithm is proposed in which the number of additional function evaluations is reduced to that of 4 values of H, see [16] for a detailed treatment. The basis is again the system (4.1), (4.2) but with

$$\varphi(u, r) := (e^{n+1})^T v(u, r) = (e^{n+1})^T B(u, r)^{-1} e^{n+1} = 0 \tag{4.9}$$

from (3.15) as bordering equation where $r \in R(\omega, \bar{v})$ is fixed for the moment. One step of Newton's method applied to (4.1), (4.9) gives $u = u^{k+1}$ as solution of the linear $(n+1, n+1)$-system consisting of (4.4) and the linearized equation (4.9),

$$\varphi(u^k, r) + \nabla_u \varphi(u^k, r)^T (u - u^k) = 0. \tag{4.10}$$

Now (4.4), (4.10) is not treated as an $(n+1, n+1)$-system with the matrix $B\left(u^k, \nabla_u \varphi(u^k, r)\right)$ but solved in two steps. The first one consists of solving (4.4) as described by (4.5), (4.6), (4.7). In the second step (4.5) is substituted into (4.10) yielding a linear equation for λ that has the solution

$$\lambda = \tilde{\lambda}_k := -\frac{\varphi(u^k, r) + \nabla_u \varphi(u^k, r)^T s^k}{\nabla_u \varphi(u^k, r)^T v^k}. \tag{4.11}$$

Considering (3.17) and the fact that $v(u^k, r)$ is a multiple of v^k we get

$$\tilde{\lambda}_k := \frac{(e^{n+1})^T (v^k - \tilde{y}^k)}{(e^{n+1})^T \tilde{z}^k}, \tag{4.12}$$

$$\tilde{y}^k := B(u^k, r)^{-1} \begin{bmatrix} H''(u^k) s^k v^k \\ 0 \end{bmatrix}, \quad \tilde{z}^k := B(u^k, r)^{-1} \begin{bmatrix} H''(u^k) v^k v^k \\ 0 \end{bmatrix}.$$

From (4.12) it is seen that $H''(u^k)$ is only used in the form

$$H''(u^k) p v^k, \quad p := s^k, v^k. \tag{4.13}$$

The terms (4.13) can be considered as limits of certain second order divided differences. Because of the different sizes of s^k and v^k ($\|s^k\| = O(\|u^k - \bar{u}\|)$ is small whereas $\|v^k\| = 1$) the arguments of $H''(u^k)$ are assumed to be normed which is no restriction since H'' is bilinear. Therefore, for $v, d \in \boldsymbol{R}^{n+1}$, $\|d\| = \|v\| = 1$, the term $H''(u) dv$ can be approximated by

$$q(u, v, d, \mu) := [H(u + \mu v) - H(u + \mu v - \mu d) + H(u - \mu d) - H(u)]/\mu^2 \tag{4.14}$$

with a discretization stepsize $\mu \neq 0$ and an error

$$\|H''(u)vd - q(u, v, d, \mu)\| \leqslant C|\mu|$$

if H'' is Lipschitz continuous.

Using the approximation (4.14) for estimating the vectors $\tilde{y}^k$ and $\tilde{z}^k$ in (4.12) we get the basic step of the new direct algorithm.

4.1. Basic step

Step 0: Let $u^k \in \boldsymbol{R}^{n+1}$, $r^k \in \boldsymbol{R}^{n+1}$ with $\|r^k\| = 1$, and $\mu_k \in \boldsymbol{R}^1$, $\mu_k \neq 0$, be given.

Step 1: Determine v^k and s^k according to (4.6), (4.7).

Step 2: Set

$$a^k := \textbf{if } s^k \neq 0 \textbf{ then } \|s^k\| q(u^k, v^k, s^k/\|s^k\|, \mu_k) \textbf{ else } 0,$$
$$b^k := q(u^k, v^k, v^k, \mu_k).$$

Step 3: Compute y^k, z^k as solutions of the linear equations

$$B(u^k, r^k)y = \begin{bmatrix} a^k \\ 0 \end{bmatrix}, \quad B(u^k, r^k)z = \begin{bmatrix} b^k \\ 0 \end{bmatrix}.$$

Step 4: Set

$$\lambda_k := \frac{(e^{n+1})^T(v^k - y^k)}{(e^{n+1})^T z^k},$$

$$u^{k+1} := u^k + s^k + \lambda_k v^k =: \Phi(u^k, r^k, \mu_k).$$

There are many ways to extend the basic step to a reasonable turning point algorithm, see [16]. In the following a version is described in which Step 1 is implemented in a similar manner as Step 1 in 3.1, compare (P_1) again.

4.2. Direct Newton-like algorithm (DNA)

Step 0: Choose $u^0 \in \boldsymbol{R}^{n+1}$ (not necessarily with $H(u^0) = 0$), $r^0 \in \boldsymbol{R}^{n+1}$ with $\|r^0\| = 1$, set $k := 0$.

Step 1.1: Compute $\tilde{v}^k$ and $\tilde{s}^k$ as solutions of the linear systems

$$B(u^k, r^k)\tilde{v} = e^{n+1}, \quad B(u^k, r^k)\tilde{s} = -\begin{bmatrix} H(u^k) \\ 0 \end{bmatrix},$$

$$\text{set } v^k := \tilde{v}^k/\|\tilde{v}^k\|, \; s^k := \tilde{s}^k - [(v^k)^T\tilde{s}^k]v^k.$$

Step 1.2: If $\sigma_k := \operatorname{sqrt}\{\|H(u^k)\|^2 + |(e^{n+1})^T v^k|^2\} = 0$, stop.

Steps 2, 3, 4: As Steps 2, 3, 4 of 4.1.

Step 5: Choose $r^{k+1} \in \boldsymbol{R}^{n+1}$ with $\|r^{k+1}\| = 1$ and $(r^{k+1})^T v^k > 0$, set $k := k+1$, and go to Step 1.

Note that the four linear systems to be solved in Step 1.1 and Step 3 have all the same matrix $B(u^k, r^k)$ so that only one LR-factorization is needed.

Of course, the choice of r^{k+1} in Step 5 should be made as in (3.20), (3.21), or (3.22) with r^k there being replaced by r^{k+1}.

The convergence of 4.2 is described by the following theorem.

Note, first, that the system (4.1), (4.9) has the regular solution $\bar{u}$ uniformly for all $r \in R(\omega, \bar{v})$ so that Newton-like convergence is to be expected.

4.3. THEOREM. *Let H: $D \subset \boldsymbol{R}^{n+1} \to \boldsymbol{R}^n$ satisfy assumption* (TP) *and have a Lipschitz continuous second derivative. Then algorithm 4.2 combined with the rules* (3.20), (3.21), *or* (3.22) *for the choice of r^{k+1} is well-defined and terminates after a finite number of steps at $\bar{u}$, or else it is infinite and converges toward $\bar{u}$ provided that $\|u^0 - \bar{u}\|$, $\|r^0 - \bar{v}\|$, and the discretization stepsizes $|\mu_k|$ are sufficiently small. The speed of convergence is described by the inequality*

$$\|u^{k+1} - \bar{u}\| \leqslant C \|u^k - \bar{u}\| \{\|u^k - \bar{u}\| + |\mu_k|\}. \tag{4.15}$$

The estimate (4.15) *implies the Q-superlinear convergence in case of*

$$\lim \mu_k = 0,$$

and if μ_k is chosen according to

$$|\mu_k| \leqslant \beta \sigma_k, \qquad \beta > 0 \text{ fixed},$$

say, as

$$\mu_k := \min\{\varepsilon_1 \|u^k\| + \varepsilon_2, \sigma_k\}, \qquad \varepsilon_1 := 10^{-3}, \ \varepsilon_2 := 10^{-5}, \tag{4.16}$$

then the convergence is Q-quadratic.

Note, at this point, that the system (4.1), (4.9) can be written in the equivalent form

$$\begin{aligned} H(u) &= 0, \\ r^T v &= 1, \\ H'(u) v &= 0, \\ (e^{n+1})^T v &= 0, \end{aligned} \tag{4.17}$$

where $v := v(u, r)$. For the choice $r = e^j$ ($j \in \{1, \ldots, n+1\}$), (4.17) reduces to a system of dimension $2n$ for the unknown variables $(u_1, \ldots, u_{n+1}, v_1, \ldots$ $\ldots, v_{j-1}, v_{j+1}, \ldots, v_n)$. Such a reduced system has been used in [21] and solved by applying Newton's method. In this form second derivatives are to be evaluated or approximated using n Jacobians H', and a linear system of dimension $2n$ is to be solved per step. On the other hand in

the basic step 4.1 the vector v occurring linearly is eliminated in advance yielding (4.1), (4.9), and after that Newton's method is applied. This approach corresponds to the idea of the so-called "variable projection methods" used for solving nonlinear least squares problems whose variables separate, see, e.g., [5].

Note, further, that in the definition of φ in (4.9) the vector r is only used to make v as a solution of $H'(u)v = 0$ unique and not too small. Of course also other conditions with the same effect as

$$\|v\| = 1, \quad r^T v > 0 \tag{4.18}$$

or, as proposed in [17],

$$\|v\| = 1, \quad \det\big(B(u, v)\big) > 0 \tag{4.19}$$

could be used but the condition $r^T v = 1$ seems to be somewhat simpler since it is linear in v.

5. Numerical examples

The methods described in this paper have been implemented and used for the solution of a number of academic and real life problems. In all cases the methods were able to follow the solution path and to indicate and calculate the turning points with high accuracy and low costs. In order to illustrate the performance of the algorithms we consider the trigger circuit shown in Fig. 7. If the diodes are modeled by an expo-

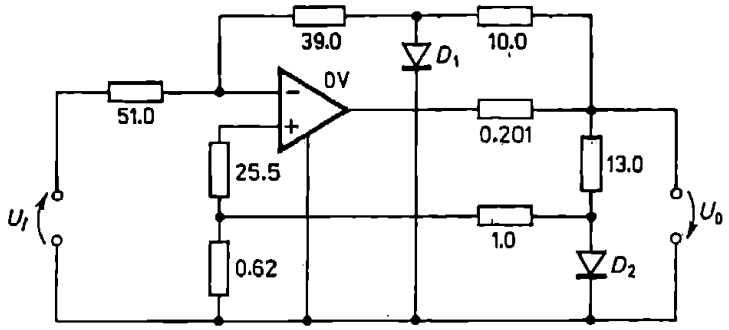

Fig. 7

nential and the operational amplifier by an arctan then the following nonlinear system results:

$$H_1 := \frac{u_1 - u_3}{10000} + \frac{u_1 - u_2}{39} + \frac{u_1 + u_7}{51},$$

$$H_2 := \frac{u_2 - u_6}{10} + \frac{u_2 - u_1}{39} + 5.6 \cdot 10^{-8} \cdot \big(\exp(25u_2) - 1\big),$$

$$H_3 := \frac{u_3 - u_4}{25.5} + \frac{u_3 - u_1}{10000},$$

$$H_4 := \frac{u_4 - u_3}{25.5} + \frac{u_4}{0.62} + u_4 - u_5,$$

$$H_5 := \frac{u_5 - u_6}{13} + u_5 - u_4 + 5.6 \cdot 10^{-8} \cdot \left(\exp(25u_5) - 1\right),$$

$$H_6 := \frac{u_6 - u_5}{13} + \frac{u_6 - u_2}{10} + \frac{u_6 - 7.65 \arctan\left(1962\,(u_3 - u_1)\right)}{0.201}.$$

The quantities $u_1, \ldots, u_7$ are voltages, especially, $u_7 = U_i$ is the input voltage and $u_6 = U_0$ is the output voltage of the trigger. The circuit is characterized by its transfer characteristic curve defined as the projection of the solution path on the (u_6, u_7)-plane. This curve has two turning points $\bar{u}^{(1)}$ (from the left) and $\bar{u}^{(2)}$ (from the right) and looks like the curve of Fig. 3. The turning points as well as the initial points used by 3.1 and generated by a path following algorithm are given in table 1.

Table 1

	$\bar{u}^{(1)}$	u^{-1}	u^0	$\bar{u}^{(2)}$	u^{-1}	u^0
u_1	0.235777668	0.2341	0.2356	0.049366971	0.0481	0.0680
u_2	0.662968764	0.6603	0.6627	0.547358409	0.5450	0.5728
u_3	0.237597699	0.2355	0.2374	0.049447207	0.0482	0.6915
u_4	0.237602341	0.2355	0.2374	0.049447411	0.0482	0.6915
u_5	0.620832106	0.6153	0.6202	0.129201309	0.1258	0.1807
u_6	9.608996879	9.0454	9.5453	1.166019152	1.1357	1.6307
u_7	0.322866124	0.3233	0.3228	0.601853012	0.6018	0.5898

The results for algorithm 3.1 with $r^k := v^k$ and the different stepsize rules are summarized in Table 2 using the following abbreviations:

NS — Number of steps to be performed in order to satisfy $|(e^{n+1})^T v^k| \leqslant 10^{-8}$,

NC — Number of corrector steps,

NH — Number of evaluations of H,

NH′ — Number of evaluations of H'.

For each stepsize rule there are two numbers in the columns, the left one for the Newton corrector (3.7) and the right one for the modified Newton corrector (3.23), respectively. In both cases we used the criterion $\|H(u^{k,i})\| \leqslant 10^{-8}$ as termination rule for the corrector iteration.

It is seen from Table 2 that the stepsize rules (CIP) and (LIP2) perform a little better than (QIP) and (LIP1) on this example. Further

Table 2

Alg. 3.1		(QIP)		(CIP)		(LIP1)		(LIP2)	
$\bar{u}^{(1)}$	NS	6	7	5	5	6	6	4	5
	NC	7	9	3	6	5	10	1	5
	NH	13	16	8	11	11	16	5	10
	NH′	15	10	10	7	13	8	7	7
$\bar{u}^{(2)}$	NS	6	6	4	5	6	6	5	5
	NC	6	8	5	5	6	8	5	5
	NH	12	14	9	10	12	14	10	10
	NH′	14	8	11	7	14	8	12	7

tests have shown, however, that the differences between (CIP), (LIP1), and (LIP2) seem to be not essential whereas the use of the modified Newton corrector brings a significant reduction of costs, in general.

In order to give an impression of the behaviour of the single iterations the results for (LIP2) are described more detailed in Table 3. The numbers in the columns refer to the turning points $\bar{u}^{(1)}$ and $\bar{u}^{(2)}$, respectively.

Table 3

Algorithm 3.1 with stepsize rule (LIP2)						
Newton corrector (3.7)						
k	$\lvert(e^{n+1})^T v^k\rvert$	$\lvert\det J(u^k)\rvert$	$\lVert H(u^k)\rVert$	NC	NH	NH′
−1	1.7−01 7.3−03	3.7+01 8.8+01	8.3−10 2.1−12			1 1
0	2.1−01 6.4−01	2.3+01 9.3+01	7.7−11 5.3−11			1 1
1	1.9−05 3.2−03	2.7−03 4.2−01	1.3−12 8.7−12	1 2	2 3	2 3
2	8.3−08 2.1−05	1.2−06 4.4−02	3.6−13 1.2−13	0 2	1 3	1 3
3	1.0−12 4.4−07	1.4−10 9.0−04	3.6−13 2.5−13	0 1	1 2	1 2
4	2.2−14 6.9−10	3.0−14 1.4−06	3.6−13 2.5−13	0 0	1 1	1 1
5	2.0−14	4.2−11	2.5−13	0	1	1
Modified Newton corrector (3.23)						
−1	1.7−01 7.3−03	3.7+01 8.8+01	8.3−10 2.1−12			1 1
0	2.1−01 6.4−01	2.3+01 9.3+01	7.7−11 5.3−11			1 1
1	1.4−04 1.9−03	2.0−02 4.1−01	8.1−12 8.7−12	3 2	4 3	1 1
2	4.7−06 2.2−05	6.6−04 4.4−02	2.1−12 1.3−12	1 2	2 3	1 1
3	2.2−08 4.4−07	3.1−07 8.9−04	2.6−11 2.1−12	1 1	2 2	1 1
4	5.3−11 7.3−10	1.1−10 1.6−06	1.7−11 1.6−12	0 0	1 1	1 1
5	8.2−13 3.5−13	5.3−12 7.2−09	1.5−11 1.6−12	0 0	1 1	1 1

The direct method 4.2 has been realized with $r^k := v^{k-1}$ and μ_k according to (4.16). The results for the initial points

$$u^0 := (0.20, 0.60, 0.20, 0.20, 0.60, 9.50, 0.30)^T \quad \text{for} \quad \bar{u}^{(1)}$$

and

$$u^0 := (0.05, 0.50, 0.05, 0.05, 0.15, 1.30, 0.50)^T \quad \text{for} \quad \bar{u}^{(2)}$$

are listed in Table 4 in the same manner as in Table 3. To conclude this paper let us remark that the indirect algorithms appear to be especially useful if the turning points as well as the entire solution path are of interest because of the possibility of changing from the path tracing strategy to the turning point strategy only by changing the stepsize rules. If, however, only the turning points and the dependence of them on further quantities involved as additional parameters in H are of interest then the direct methods appear to be advantageous. Remark that in this case continuation techniques with respect to one of such an additional parameter can be applied to compute curves of turning points but this problem is out of the scope of this paper.

Table 4

Algorithm 4.2		
k	$\|(e^{n+1})^T v^k\|$	$\|H(u^k)\|$
0	2.1 −01 7.0 −01	2.2 +01 2.8 +01
1	4.0 −02 4.2 −03	1.9 −01 1.7 −01
2	5.9 −03 2.2 −04	8.1 −03 1.8 −02
3	7.1 −05 1.1 −06	8.5 −05 1.1 −04
4	2.4 −08 1.3 −09	5.8 −08 3.0 −09
5	2.1 −11 9.8 −13	2.2 −11 1.9 −13

References

[1] J. P. Abbott, *An efficient algorithm for the determination of certain bifurcation points*, J. Comput. Appl. Math. 4 (1978), 19–27.

[2] E. Allgower, K. Georg, *Simplicial and continuation methods for approximating fixed points and solutions to systems of equations*, SIAM Rev. 22 (1980), 28–85.

[3] P. Anselone, R. Moore, *An extension of the Newton–Kantorovich method for solving nonlinear equations with an application to elasticity*, J. Math. Anal. Appl. 13 (1966), 476–501.

[4] L. O. Chua, A. Ushida, *A switching-parameter algorithm for finding multiple solutions of nonlinear resistive circuits*, IEEE Trans. Circuit Theory and Applications 4 (1976), 215–239.

[5] G. H. Golub, V. Pereyra, *The differentiation of pseudoinverses and nonlinear least squares problems whose variables separate*, SIAM J. Numer. Anal. 10 (1973), 413–432.

[6] C. Haselgrove, *Solution of nonlinear equations and of differential equations with two-point boundary conditions*, Comput. J. 4 (1961), 255–259.

[7] H. B. Keller, *Numerical solution of bifurcation and nonlinear eigenvalue problems*, in: P. Rabinowitz (ed.), *Applications of bifurcation theory*, Academic Press, New York, 1977, pp. 359–384.

[8] H. B. Keller, *Global homotopies and Newton methods*, in: C. de Boor, G. H. Golub (eds.), *Recent advances in numerical analysis*, Academic Press, New York, 1978, pp. 73–94.

[9] F. Kikuchi, *Finite element approximations to bifurcation problems of turning point type*, in: R. Glowinski, J. L. Lions (eds.), *Computing methods in applied sciences and engineering*, 1977, I. Springer, Berlin, 1979, pp. 252–266.

[10] M. Kubiček, *Evaluation of branching points for nonlinear boundary-value problems based on the GPM-technique*, Appl. Math. Comput. 1 (1975), 341–352.

[11] R. Menzel, H. Schwetlick, *Zur Lösung parameterabhängiger nichtlinearer Gleichungen mit singulären Jacobi-Matrizen*, Numer. Math. 30 (1978), 65–79.

[12] J. M. Ortega, W. C. Rheinboldt, *Iterative solution of nonlinear equations in several variables*, Academic Press, New York, 1970.

[13] G. Pönisch, *Verfahren zur numerischen Bestimmung von Rückkehrpunkten implizit definierter Raumkurven*, Dissertation A, Technische Universität Dresden, Sektion Mathematik, Dresden, 1979.

[14] —, *Ein implementierbares ableitungsfreies Verfahren zur Bestimmung von Rückkehrpunkten implizit definierter Raumkurven*, Beiträge zur Numerischen Mathematik 9 (1980).

[15] G. Pönisch, H. Schwetlick, *Ein lokal überlinear konvergentes Verfahren zur Bestimmung von Rückkehrpunkten implizit definierter Raumkurven*, Numer. Math. 38 (1982), 455–466; see also Preprint 07-30-77, Technische Universität Dresden, Sektion Mathematik, Dresden, 1977.

[16] —, —, *Computing turning points of curves implicitly defined by nonlinear equations depending on a parameter*, Computing 26 (1980), 107–121.

[17] W. C. Rheinboldt, *Solution field of nonlinear equations and continuation methods*, SIAM J. Numer. Anal. 17 (1980), 221–237.

[18] E. Riks, *The application of Newton's method to the problem of elastic stability*, Trans. ASME J. Appl. Mech. 39 (1972), 1060–1065.

[19] H. Schwetlick, *Numerische Lösung nichtlinearer Gleichungen*, Deutsch. Verlag der Wissenschaften, Berlin, 1979; Oldenbourg, München, 1979.

[20] —, *Einbettungstechniken zur Lösung parameterabhängiger nichtlinearer Gleichungssysteme*, Wiss. Z. Techn. Univers. Dresden 29 (1980), 445–449.

[21] R. Seydel, *Numerical computation of branch points in nonlinear equations*, Numer. Math. 33 (1979), 339–352.

[22] R. B. Simpson, *A method for numerical determination of bifurcation states of nonlinear systems of equations*, SIAM J. Numer. Anal. 12 (1975), 439–451.

Presented to the Semester
Computational Mathematics
February 20 – May 30, 1980