stichting
mathematisch
centrum

MC

AFDELING NUMERIEKE WISKUNDE NW 13/74 SEPTEMBER

J.C.P. BUS & T.J. DEKKER

TWO EFFICIENT ALGORITMS WITH GUARANTEED CONVERGENCE FOR FINDING A ZERO OF A FUNCTION

2e boerhaavestraat 49 amsterdam

Printed at the Mathematical Centre, 49, 2e Boerhaavestraat, Amsterdam.

The Mathematical Centre, founded the 11-th of February 1946, is a non-profit institution aiming at the promotion of pure mathematics and its applications. It is sponsored by the Netherlands Government through the Netherlands Organization for the Advancement of Pure Research (Z.W.O), by the Municipality of Amsterdam, by the University of Amsterdam, by the Free University at Amsterdam, and by industries.

AMS (MOS) subject classification scheme (1970): 65H05.

Two efficient algorithms with guaranteed convergence for finding a zero of a function

by

J.C.P. Bus *) and T.J. Dekker **)

ABSTRACT

Two algorithms are presented for finding a zero of a real continuous function defined on a given interval. The methods used are mixtures of linear interpolation, rational interpolation and bisection.
The asymptotic behaviour of these algorithms is completely satisfactory. The number of function evaluations needed to find a zero of a function is bounded by four or five times the number needed by bisection and usually considerably smaller.

---

*) Mathematical Centre, Tweede Boerhaavestraat 49, Amsterdam.

**) University of Amsterdam, Roetersstraat 15, Amsterdam.

CONTENTS

1. INTRODUCTION

Our starting point is an algorithm, published by DEKKER [3], for finding a zero of a real function defined on a given interval.

Section 2 contains a detailed discussion on this algorithm which we call "algorithm A" in the sequel. The method used in algorithm A is a mixture of linear interpolation and bisection. For this algorithm, convergence is guaranteed and the asymptotic behaviour is completely satisfactory. However, the number of function evaluations required by this algorithm may be prohibitively large, in particular, when the zero appears to be multiple. Therefore, BRENT [2] proposed a modified algorithm (called "algorithm B" in section 5). For this algorithm the upper bound of the number of function evaluations needed equals $(t+1)^2-2$, where t is the number of function evaluations needed by bisection.

In section 3 we present a modified algorithm ("algorithm M") having the same asymptotic order of convergence as algorithm A but requiring at most 4t function evaluations. This is achieved by inserting steps in which rational interpolation (see JARRATT & NUDDS [5]) or bisection is performed. ANDERSON and BJORCK [1] present an algorithm (which we call algorithm C in section 5) which uses also linear interpolation and rational interpolation. This algorithm may however require as many function evaluations as algorithm A.

In section 4, we present another algorithm ("algorithm R") having a higher asymptotic order of convergence and requiring at most 5t function evaluations. This algorithm has a similar strategy but uses rational interpolation instead of linear interpolation.

In section 5, we compare some numerical results of the algorithms mentioned and in section 6, we give some conclusions.

A description of our algorithms in the form of ALGOL 60 procedures is given in Appendix.

2. ALGORITHM A

For a detailed description of algorithm A, together with a discussion on its properties and an ALGOL 60 procedure, see DEKKER [3].

2.1. DATA. Given a real continuous function f of one real variable, two distinct argument values x_0 and x_1 satisfying $f(x_0) \times f(x_1) \leq 0$, and a positive tolerance function δ of one real variable satisfying $0 < \tau \leq \delta(x)$, where τ is a given positive constant (for instance, $\delta(x) \equiv \tau$ defines an absolute tolerance τ and $\delta(x) = \alpha|x| + \tau$ defines a relative tolerance α when $|x|$ is large).

2.2. RESULTS. The purpose of algorithm A (and of algorithms M and R presented in the next two sections) is to find two (distinct) real numbers x and y satisfying

$$\begin{aligned} & f(x) \times f(y) \leq 0 \quad , \\ & |f(x)| \leq |f(y)| \quad , \\ & |x-y| \leq 2\delta(x). \end{aligned} \tag{2.2.1}$$

Since f is continuous, the first condition ensures that there exists a zero, z, of f in the closed interval with endpoints x and y; the second condition yields that x is the "best" approximation of z; the third condition states that the required tolerance has been reached.

2.3. DEFINITION OF ALGORITHM A. From the data mentioned in 2.1, algorithm A produces two argument values x and y satisfying (2.2.1). This is achieved by calculating in succession the argument values x_i (for i=2,...,n), and a_i, b_i and c_i (for i=1,...,n) as defined in A1 and A2 below, where n and the results delivered are defined in A3.

A1 (initialisation, i=1).
If $|f(x_1)| \leq |f(x_0)|$,
then $b_1 = x_1$ and $a_1 = c_1 = x_0$;
otherwise $b_1 = x_0$ and $a_1 = c_1 = x_1$.

A2 (iteration step, i=2,...,n).
Let the linear interpolation formula be defined, for $a \neq b$, by

$$\ell = \ell(b,a) = \begin{cases} b - \dfrac{f(b)(b-a)}{f(b)-f(a)} & \text{if } f(b) \neq f(a), \\ \infty & \text{if } f(b) = f(a) \neq 0, \\ b & \text{if } f(b) = f(a) = 0. \end{cases} \tag{2.3.1}$$

Let moreover,

(2.3.2) $h = h(b,c) = b + \mathrm{sign}(c-b) \times \delta(b)$,

(2.3.3) $m = m(b,c) = \frac{1}{2}(b+c)$

and

(2.3.4) $v = v(\ell,b,c) = \ell$ if ℓ is between $h(b,c)$ and $m(b,c)$,
$= h(b,c)$ if $|\ell - b| \leq \delta(b)$,
$= m(b,c)$ otherwise.

Then the new iterate x_i is calculated according to the formula

(2.3.5) $x_i = v(\lambda_i, b_{i-1}, c_{i-1})$,

where

$$\lambda_i = \ell(b_{i-1}, a_{i-1}).$$

Furthermore, let k be the largest (non-negative) integer satisfying $k < i$ and $f(x_k) \times f(x_i) \leq 0$.
Then, b_i, c_i and a_i are defined by

(2.3.6) $b_i = x_i$, $c_i = x_k$, $a_i = b_{i-1}$ if $|f(x_i)| \leq |f(x_k)|$;

(2.3.7) $b_i = x_k$, $a_i = c_i = x_i$ otherwise.

A3 (termination).
Let n be the smallest positive integer satisfying

(2.3.8) $|b_n - c_n| \leq 2\delta(b_n)$.

Then, the algorithm terminates for $i = n$ and delivers as results

(2.3.9) $x = b_n$, $y = c_n$.

2.4. ADDITIONAL DEFINITIONS AND REMARKS.

2.4.1. Let J_i, for i=1,2,...,n denote the closed interval whose endpoints are b_i and c_i. Then, from the *invariant relations*

$$f(b_i) \times f(c_i) \leq 0$$

and

$$|f(b_i)| \leq |f(c_i)|$$

it follows that J_i contains a zero z of f and b_i is the best approximation of z obtained up to and including step i.

2.4.2. The iterates x_i (i=1,2,...,n) are all distinct and their mutual distances are at least τ. Hence, $|b_i - a_i| \geq \tau$ for i=1,2,...,n, so that λ_i and x_i in (2.3.5) are well defined for i=1,2,...,n.

2.4.3. If $a_{i-1} = c_{i-1}$, for certain i, then, the argument values a_{i-1} and b_{i-1}, used to calculate λ_i in (2.3.5), are on different sides of z and we call the i-th step a (*linear*) *intrapolation step*; otherwise, a_{i-1} and b_{i-1} are on the same side of z and we call the i-th step a (*linear*) *extrapolation step*.

2.4.4. Obviously, algorithm A uses the function values $f(x_i)$, for i=0,1,...,n. So, the number of function evaluations needed equals n + 1.

2.5. PROPERTIES. Algorithm A has the following properties (see DEKKER [3]).

2.5.1. If the given function f has a continuous second derivative in J_1 and a unique simple zero in this interval, then the asymptotic order of convergence of algorithm A equals the largest root, p_1, of the equation $x^2 - x - 1 = 0$, thus

$$p_1 = \tfrac{1}{2}(1+\sqrt{5}) \cong 1.618;$$

2.5.2. The number of function evaluations needed is bounded above by T, where

$$T = |x_1 - x_0|/\tau.$$

As BRENT [2] shows, this upper bound may indeed by attained.

2.6. DISCUSSION. If $f(x_i) \times f(b_{i-1}) \leq 0$ for certain i, then

$$|b_i - c_i| = |x_i - b_{i-1}| \leq \tfrac{1}{2}\,|b_{i-1} - c_{i-1}|;$$

otherwise

$$\tfrac{1}{2}|b_{i-1} - c_{i-1}| \leq |b_i - c_i| = |x_i - c_{i-1}| \leq |b_{i-1} - c_{i-1}| - \tau.$$

So, we may have (very) slow convergence only if the latter case occurs frequently.

If f has a continuous second derivative, z is a simple zero of f (i.e. $f'(z) \neq 0$), and a and b are sufficiently close to z to ensure that $f'(\eta) \neq 0$ for η in the smallest interval containing a, b and z, then $\ell = \ell(b,a)$, obtained by the linear interpolation formula (2.3.1), satisfies (see DEKKER [3]).

$$(2.6.1) \qquad \ell - z = (b-z)(a-z)K(\xi,\eta),$$

where

$$K(\xi,\eta) = \tfrac{1}{2}f''(\xi)/f'(\eta)$$

and ξ and η lie in the smallest interval containing a, b and z. Hence, if $|b_{i_0} - c_{i_0}|$ is sufficiently small for certain i_0, then the iterates x_i converge to z and the values $|f(x_i)|$ decrease monotonically for

$i \geq i_0$ as long as

$$(2.6.2) \qquad \delta(x_i) < |\ell(x_i,x_{i-1}) - x_i|.$$

Condition (2.6.2) ensures that, for $i \geq i_0$, the tolerance function does not influence the i-th iteration step. Henceforth in this section (where we consider the asymptotic behaviour of algorithm A), we take $i \geq i_0$ and assume that condition (2.6.2) holds for all $i \geq i_0$. (In fact we consider the process that is obtained if the tolerance function δ tends uniformly to zero on the interval J_1; see also the proof of theorem 3.3.2). Then, by A2, we have $b_i = x_i, a_i = x_{i-1}$ and $c_i = x_k$. Let $\varepsilon_i = b_i - z$ $(= x_i - z)$ denote the error of the i-th iterate. Then, (2.3.5) and (2.6.1) yield

$$(2.6.3) \qquad \varepsilon_{i+1} = \varepsilon_i \varepsilon_{i-1} K(\xi_i,\eta_i),$$

where ξ_i and η_i lie in the smallest interval containing b_i, b_{i-1} and z. Consequently, if $f''(z) \neq 0$, we have $K(z,z) \neq 0$. Hence, for sufficiently large i, $K(\xi_i,\eta_i)$ has the same sign as $K(z,z)$. Therefore, the sign of $K(z,z)$ and of two successive errors ε_i and ε_{i-1} completely determine the signs of the subsequent errors. Then, simple checking yields that, when $f''(z) \neq 0$, there are only the following two (essentially different) possibilities for the asymptotic behaviour:

1. the iteration consists of consecutive cycles of the form IIE, i.e. two intrapolation steps followed by one extrapolation step;
2. the iteration consists of consecutive extrapolation steps.

In the first case, the length of J_i is smaller than 0.25 times the length of J_{i-3}. So, in this case, we find a small upper bound (viz. $\frac{3}{2}$ t) for the number, N, of function evalutions needed. In the second case, convergence may be very slow (N may attain the upper bound T). Therefore, we modify algorithm A such that more than two consecutive extrapolation steps can no longer occur in an iteration, while an interation consisting of consecutive cycles of the form IIE remains undisturbed.

3. ALGORITHM M

3.1. DEFINITION. From the data mentioned in 2.1, algorithm M produces two argument values x and y satisfying (2.2.1). This is achieved by calculating in succession the argument values x_i, d_i (for i=2,...,n) and a_i,b_i,c_i (for i=1,...,n) as defined in A1 (see 2.3) and M2 (below), where n and the results deliverd are defined in A3 (see 2.3).

M2 (iteration step, i=2,...,n).

Let $j = j_i$ be the largest positive integer satisfying $j = 1$ or, if $1 < j < i$, then

$$(3.1.1) \quad |b_j - c_j| \le \tfrac{1}{2} |b_{j-1} - c_{j-1}|.$$

Then the new iterate x_i is calculated as follows (for the definitions of h,m and λ_i see A2).

Let

$$(3.1.2) \quad w = w(\ell,b,c) = \begin{cases} \ell & \text{if } \ell \text{ is between } h(b,c) \text{ and } m(b,c), \\ h(b,c) & \text{if } |\ell - b| \le \delta(b) \text{ and } \ell \text{ lies not outside the interval bounded by b and } m(b,c), \\ m(b,c) & \text{otherwise.} \end{cases}$$

Then,

$$(3.1.3) \quad x_i = \begin{cases} w(\lambda_i,b_{i-1},c_{i-1}) & \text{if } j_i \ge i-2, \\ w(\rho_i,b_{i-1},c_{i-1}) & \text{if } j_i = i-3, \\ m(b_{i-1},c_{i-1}) & \text{otherwise,} \end{cases}$$

where ρ_i is defined as follows: for $a \neq b$ let

$$(3.1.4) \quad f[a,b] = \frac{f(a)-f(b)}{a-b}$$

(i.e. the first divided difference of f at a and b);
for distinct a,b and d, using the abbreviations

$$\alpha = f[b,d] \times f(a), \qquad \beta = f[a,d] \times f(b),$$

define

$$(3.1.5)\quad r = r(b,a,d) = b - \frac{\beta(b-a)}{\beta-\alpha} \qquad \text{if } \beta \neq \alpha,$$

$$= \infty \qquad \text{if } \beta = \alpha \neq 0,$$

$$= 0 \qquad \text{if } \beta = \alpha = 0;$$

then

$$(3.1.6)\quad \rho_i = r(b_{i-1},a_{i-1},d_{i-1}).$$

Furthermore, let k be the largest (non-negative) integer satisfying $k < i$ and $f(x_k) \times f(x_i) \leq 0$, then b_i, c_i, a_i and d_i are defined by

$$(3.1.7)\quad b_i = x_i,\ c_i = x_k,\ a_i = b_{i-1} \qquad \text{if } |f(x_i)| \leq |f(x_k)|;$$

$$(3.1.8)\quad b_i = x_k,\ a_i = c_i = x_i \qquad \text{otherwise;}$$

$$(3.1.9)\quad d_i = a_{i-1} \qquad \text{if } b_i = x_i \text{ or } b_i = b_{i-1};$$

$$d_i = b_{i-1} \qquad \text{otherwise.}$$

3.2. ADDITIONAL DEFINITIONS AND REMARKS. The definitions and remarks 2.4 are also valid for algorithm M.

3.2.1. Formula (3.1.5) is obtained by 3 - point rational interpolation, where the interpolating function is

$$\phi(x) = \frac{x-r}{px+q}$$

and the parameters p,q and r are determined such that $\phi(x) = f(x)$ for $x = a,b,d$ (see also JARRATT & NUDDS [5]).

3.2.2. In addition to 2.4.2 it is obvious that for all $i \geq 2$, the argument values b_i, a_i and d_i are distinct and have a mutual distance which is

bounded below by τ. So, ρ_i and x_i in (3.1.6) an (3.1.3) are well defined.

3.2.3. In addition to 2.4.3 we speak about *rational interpolation* if $x_i = w(\rho_i, b_{i-1}, c_{i-1})$. Moreover, if in this case b_{i-1} and a_{i-1} lie on different sides of z, then we call the i-th step a *rational intrapolation step;* otherwise we call the i-th step a *rational extrapolation step.*

3.2.4. Comparing the definitions of w and v ((3.1.2) and (2.3.4) respectively) we note that $w(\ell,b,c) \neq v(\ell,b,c)$, only if $|\ell - b| \leq \delta(b)$ and ℓ lies not in the interval bounded by b and m(b,c). We have replaced v by w in algorithm M, because we think it is preferable from a theoretical point of view, and it sometimes yields better results.

3.3. PROPERTIES. We state and prove the following two theorems on algorithm M.

3.3.1. THEOREM. Let data be given as mentioned in 2.1. Then the number of function evaluations needed by algorithm M to obtain two values x and y satisfying (2.2.1) is bounded by 4t, where

$$t = {}^2\log(|x_1 - x_0|/\tau).$$

(Note that t is the number of function evaluations needed by bisection).

PROOF. This follows from the definition of the algorithm, in particular from formulas (3.1.1) and (3.1.2). A bisection step is performed whenever none of the last three steps has reduced the length of the interval by a factor ≤ 0.5. Hence, the length of J_i is smaller than half the length of J_{i-4}, which proves the theorem. □

3.3.2. THEOREM. Let data be given as mentioned in 2.1. Let moreover, the given function f have a continuous fourth derivative and an unique simple zero, z, in the interval J_1. Then the asymptotic order of convergence of algorithm M, finding an approximation of z equals p_1.
(For definitions of J_I and p_1 see 2.4.1 and 2.5.1).

PROOF. Let

$$(3.3.3) \qquad c_k = f^{(k)}(z)/k! \qquad k > 0.$$

Then $c_1 \neq 0$, because z is a simple zero of f by assumption.
We need more terms in the error formula (2.6.1). By straightforward calculation, using Newton's interpolation formula and the assumption that f has a continuous fourth derivative, we find

$$(3.3.4) \qquad \ell - z = (b-z)(a-z)[K_0 - K_1(b-z+a-z) + O(|b-z| + |a-z|)^2],$$

where

$$K_0 = c_2/c_1 \text{ and } K_1 = (c_2/c_1)^2 - c_3/c_1.$$

Similarly, for the 3-point rational interpolation formula (3.1.5) we find (see also JARRATT & NUDDS [5]):

$$(3.3.5) \qquad r - z = (b-z)(a-z)(d-z)[K_1 + O(|b-z| + |a-z| + |d-z|)].$$

From (3.3.5), it follows that the asymptotic order of convergence of the 3-point rational interpolation formula equals p_2, where p_2 is the largest root of the equation $x^3 - x^2 - x - 1 = 0$; hence $p_2 \approx 1.839$, cf. JARRATT & NUDDS [5].

We consider the asymptotic order of convergence of the iteration process, that is obtained if we let the tolerance function δ tend uniformly to zero on the interval J_1. (We assume, of course, that exact arithmetic is used.) This limit process is a well defined iteration process which does, however, not terminate. (Here, we use the fact that the divided difference f[a,b] converges to f'(a) when b converges to a). The intervals J_i (i=1,2,...) are monotonically non-increasing (i.e. $J_{i+1} \subset J_i$, for all i) and the length of the interval J_i converges to zero for i tending to infinity. (Indeed the length decreases by a factor ≤ 0.5 in every 4 steps, cf. the proof of the

previous theorem). We choose i_0 such that $f'(x) \neq 0$ for $x \in J_{i_0}$.

From the definition of the algorithm, in particular (3.1.1) and (3.1.3), and the error formulas (3.3.4) and (3.3.5) we know that an integer $i_1 \geq i_0$ exists, such that

a. for all $i > i_1$ satisfying $j_i > i - 3$, a bisection step is performed to obtain the i-th iterate x_i (i.e. $x_i = m(b_{i-1},c_{i-1})$); so, $|f(x_i)| > |f(b_{i-1})|$ and $f(x_i) \times f(b_{i-1}) \leq 0$; in this case, a_i,b_i and c_i are chosen according to (3.1.8) and the (i+1)-th step will be an intrapolation step;

b. for all $i > i_1$ satisfying $j_i \leq i-3$ we have $|f(x_i)| \leq |f(b_{i-1})|$ and $|x_i-z| \leq |x_{i-1}-z|$; now, b_i,a_i and c_i are obtained by (3.1.7); substituting $\varepsilon_k = b_k - z$ for arbitrary k in (3.3.4) and (3.3.5), we obtain

$$\lambda_i - z = \varepsilon_{i-1}\,\varepsilon_{i-2}[K_0 - K_1(\varepsilon_{i-1}+\varepsilon_{i-2}) + O(|\varepsilon_{i-1}| + |\varepsilon_{i-2}|)], \tag{3.3.6}$$

$$\rho_i - z = \varepsilon_{i-1}\,\varepsilon_{i-2}\,\varepsilon_{i-3}[K_1 + O(|\varepsilon_{i-1}| + |\varepsilon_{i-2}| + |\varepsilon_{i-3}|)]. \tag{3.3.7}$$

We distinguish between two cases.

A. There exists an $i_2 \geq i_1$, such that $j_i \geq i - 3$ for all $i \geq i_2$. Then, for all $i \geq i_2$, the iterate x_i is obtained by linear interpolation (with asymptotic order of convergence equal to p_1) or by 3-point rational interpolation (with asymptotic order of convergence equal to $p_2 > p_1$). This leads immediately to the required result.

B. For each $i_2 \geq i_1$, there exists an $i \geq i_2$ such that $j_i < i - 3$. We distinguish between two subcases.

B.1. $c_2 \neq 0$. So, $K_0 \neq 0$. Hence an integer $\nu \geq i_2$ exists, such that $j_\nu < \nu - 3$ and K_0 in formula (3.3.6) dominates. Consequently, using (a), the $(\nu+1)$ -th step is an intrapolation step and the sign of $\varepsilon_i (i>\nu)$ is determined by the sign of $\varepsilon_\nu,\varepsilon_{\nu-1}$ and K_0. Then it is easily checked that, from the $(\nu+1)$-th step , the iteration consists of consecutive cycles of the form IIE, i.e. two linear intrapolation steps followed by one linear extrapolation step. This contradicts our assumption (B).

B.2. $c_2 = 0$. Then, also $K_0 = 0$.

We again distinguish between two subcases.

B.2.1. $c_3 \neq 0$. So, $K_1 \neq 0$. Hence, as in (B.1.) an integer $\nu \geq i_2$ exists such that the $(\nu+1)$-th step is a linear intrapolation step and the term $K_1(\varepsilon_{i-1}+\varepsilon_{i-2})$ in formula (3.3.6) and the term K_1 in (3.3.7) dominate. Consequently, the sign of ε_i $(i>\nu)$ is completely determined by the sign of $\varepsilon_\nu, \varepsilon_{\nu-1}$ and K_1. (Note that ε_i $(i>\nu)$ equals either $\lambda_i - z$ or $\rho_i - z$ and that a rational extrapolation step always yields an iterate on the other side of z. So, this step is always followed by a linear intrapolation step.) It can be shown that from the $(\nu+1)$-th step the iteration consists of either only linear intrapolation steps (viz. when $K_1 > 0$) or cycles of the form IEE', i.e. a linear intrapolation step, a linear extrapolation step and a rational extrapolation step. This also contradicts our assumption (B).

B.2.2. $c_3 = 0$. Then, also $K_1 = 0$ and the most unfavourable situation is an iteration consisting of consecutive cycles of the form IEE'B, i.e. a linear intrapolation step, a linear extrapolation step, a rational extrapolation step and a bisection step. Let the i-th step be a bisection step yielding argument values $a_i = c_i = x_i$ and $b_i = x_{i-1}$. Then $a_i - z = O(1)$ and, according to (3.3.6) and (3.3.7), the cycle IEE'B asymptotically yields:

$$I : \varepsilon_{i+1} = \lambda_{i+1} - z = O(\varepsilon_i(c_i-z)^3) = O(\varepsilon_i),$$

$$E : \varepsilon_{i+2} = \lambda_{i+2} - z = O(\varepsilon_{i+1}\varepsilon_i^3) = O(\varepsilon_i^4),$$

$$E' : \varepsilon_{i+3} = \rho_{i+3} - z = O(\varepsilon_{i+2}\varepsilon_{i+1}\varepsilon_i^2) = O(\varepsilon_i^7),$$

$$B : \varepsilon_{i+4} = \varepsilon_{i+3} = O(\varepsilon_i^7)$$

and $a_{i+4} = c_{i+4} = x_{i+4}$.

So, in this case, the effective asymptotic order of convergence equals $\sqrt[4]{7} \tilde{=} 1.626$, which is greater than p_1. This completes the proof of the theorem. □

4. ALGORITHM R

4.1. DEFINITION. From the data mentioned in (2.1), algorithm R produces two argument values x and y satisfying (2.2.1), by successively calculating

argument values x_i, and d_i (for i=2,...,n) and a_i, b_i and c_i (for i=1,...,n) as defined in A1 (see 2.3) and R2 (below), where n and the results delivered are defined in A3 (see 2.3).

R_2 (iteration step, i=2,...,n).

Let j_i be defined as in M2. Then, the new iterate x_i is calculated as follows (for the definitions of λ_i and m see A2, and for the definitions of w and ρ_i see M2):

$$
\begin{aligned}
(4.1.1) \qquad x_i &= w(\lambda_i, b_{i-1}, c_{i-1}) && \text{if } i = 2, \\
&= w(\rho_i, b_{i-1}, c_{i-1}) && \text{if } i \geq 3 \text{ and } j_i \geq i-3, \\
&= w(2\rho_i - b_{i-1}, b_{i-1}, c_{i-1}) && \text{if } i \geq 3 \text{ and } j_i = i-4, \\
&= m(b_{i-1}, c_{i-1}) && \text{otherwise.}
\end{aligned}
$$

Furthermore, b_i, c_i, a_i and d_i are defined as in M2.

4.2. ADDITIONAL DEFINITIONS AND REMARKS. The definitions and remarks 2.4 and 3.2 are also valid for algorithm R.

4.2.1. In algorithm M a bisection step is performed ($x_i = m(b_{i-1}, c_{i-1})$) when $j_i = i - 4$, but in algorithm R a bisection step is performed when $j_i = i - 5$. The reason for this difference lies in the different asymptotic behaviour of the algorithms M and R. Using 3-point rational interpolation the errors satisfy (3.3.5). Assuming $K_1 \neq 0$, then the iteration may asymptotically consist of consecutive cycles of the form IIEE, i.e. two intrapolation steps followed by two extrapolation steps. (see also proof of theorem 4.3.2). We do not want to disturb such an asymptotic behaviour. So, we have to allow two consecutive extrapolation steps in algorithm R. Therefore, in algorithm R, we modify the third of three consecutive extrapolation steps ($j_i = i-4$) by doubling the step-length obtained with rational interpolation and a bisection step is inserted if $j_i < i - 4$.

4.2.2. In addition to 2.4.3 and 3.2.3 we call an iteration step a *modified extrapolation step* if $x_i = w(2\rho_i - b_{i-1}, b_{i-1}, c_{i-1})$.

4.3. PROPERTIES. We state and prove the following two theorems on algorithm R.

4.3.1. THEOREM. Let data be given as mentioned in 2.1. Then the number of function evaluations needed by algorithm R to produce two argument values x and y satisfying (2.2.1) is at most 5t. (For the definition of t see 3.3.1.)

PROOF. This follows immediately from the definition of the algorithm. □

4.3.2. THEOREM. Let data be given as mentioned in 2.1. Let, moreover, the given function f have a continuous fifth derivative and a unique simple zero, z, in the interval J_1.
Then, the asymptotic order of convergence of algorithm R, to find an approximation of z, equals p_2.
(For the definition of J_1 see 2.4.1 and of p_2 see the proof of theorem 3.3.2.).

PROOF. This proof is very much alike that of theorem 3.3.2.
Let c_k, $k > 0$, be defined by (3.3.3). Then $c_1 \neq 0$ by assumption. As in the proof of theorem 3.3.2 we consider the asymptotic order of convergence of the iteration process that is obtained if we let the tolerance function δ tend uniformly to zero on the interval J_1. The length of the intervals J_i converges to 0 for i tending to infinity. So, we may choose i_0 such that $f'(x) \neq 0$ for all $x \in J_{i_0}$. From the definition of the algorithm and the error formula (3.3.5) we may conclude that an integer $i_1 \geq i_0$ exists such that

a. for all $i \geq i_1$, satisfying $j_i = i - 4$, a modified extrapolation step is performed; then, using the notation $\varepsilon_k = b_k - z$ for arbitrary k, we obtain the following error formula:

$$(4.3.3) \qquad \varepsilon_i = 2\rho_i - b_{i-1} - z = -\varepsilon_{i-1}[1 + O(\varepsilon_{i-2}\varepsilon_{i-3})];$$

hence, $f(x_i) \times f(b_{i-1}) \leq 0$ and the next step will be an intrapolation step;

b. for all $i \geq i_1$, satisfying $j_i \geq i - 3$ the relations $|f(x_i)| \leq |f(b_{i-1})|$ and $|x_i - z| \leq |b_{i-1} - z|$ hold; consequently, b_i, a_i and c_i are obtained by (3.1.7).

Note that for all $i \geq i_1$, the inequality $j_i \geq i - 4$ holds because of (a). So, no bisection steps occur.
Instead of (3.3.7) we need for this proof a more elaborate error formula which can be obtained by straightforward calculation using the assumption that f has a continuous fifth derivative.

$$(4.3.4) \qquad \rho_i - z = \varepsilon_{i-1}\varepsilon_{i-2}\varepsilon_{i-3}[K_1 + K_2(\varepsilon_{i-1}+\varepsilon_{i-2}+\varepsilon_{i-3}) + \\ + O(|\varepsilon_{i-1}| + |\varepsilon_{i-2}| + |\varepsilon_{i-3}|)^2],$$

where K_1 is defined by (3.3.4) and

$$K_2 = c_2c_3/c_1^2 - c_4/c_1.$$

We distinguish between two cases.

A. There exists an integer $i_2 \geq i_1$, such that $j_i \geq i - 3$ for all $i \geq i_2$. Then, for all $i \geq i_2$, the iterate x_i is obtained by rational interpolation (with asymptotic order of convergence equal to p_2).
This proves the required result.

B. For each $i_2 \geq i_1$, there exists an $i \geq i_2$, such that $j_i = i - 4$. Hence, the i-th step is a modified step.
We distinguish between two subcases.

B.1. $K_1 \neq 0$.
By assumption (B) we may choose an integer $\nu \geq i_2$ such that the ν-th step is a modified extrapolation step and the term K_i in formula (4.3.4) dominates. Consequently, using (a), the $(\nu+1)$-th step is an intrapolation step and the sign of $\varepsilon_i (i>\nu)$ is completely determined by the sign of ε_k $(k=\nu,\nu-1,\nu-2)$ and K_1. Then, it is easily checked that, from the $(\nu+1)$-th step, the iteration can only consist of cycles of the form I or IE, when $K_1 > 0$, and IIEE, when $K_1 < 0$; here I denotes a rational intrapolation step and E denotes a rational extrapolation step. This contradicts our assumption (B).

B.2. $K_1 = 0$.
Then, the most unfavourable situation is an iteration consisting of

cycles IEEE', i.e. a rational intrapolation step, two rational extrapolation steps and a modified extrapolation step. Then, according to (4.3.4) we have

$$\varepsilon_i = -\varepsilon_{i-1} + O(\varepsilon_{i-1}\varepsilon_{i-2}\varepsilon_{i-3}^2),$$

and the cycle IEEE' yields:

$$E': \varepsilon_i = -\varepsilon_{i-1} + O(\varepsilon_{i-1}\varepsilon_{i-2}\varepsilon_{i-3}^2) = O(\varepsilon_{i-1});$$

$$I: \varepsilon_{i+1} = O(\varepsilon_i\varepsilon_{i-1}\varepsilon_{i-2}^2) = O(\varepsilon_{i-1}^2\varepsilon_{i-2}^2);$$

$$E: \varepsilon_{i+2} = \varepsilon_{i+1}\varepsilon_i\varepsilon_{i-1}[K_2(\varepsilon_{i+1}+O(\varepsilon_{i-1}\varepsilon_{i-2}\varepsilon_{i-3}^2)) + O(\varepsilon_{i-1}^2)] =$$
$$= O(\varepsilon_{i-1}^5\varepsilon_{i-2}^2(\varepsilon_{i-2}\varepsilon_{i-3}^2+\varepsilon_{i-1}));$$

$$E: \varepsilon_{i+3} = O(\varepsilon_{i+2}\varepsilon_{i+1}\varepsilon_i^2) = O(\varepsilon_{i-1}^9\varepsilon_{i-2}^4(\varepsilon_{i-2}\varepsilon_{i-3}^2+\varepsilon_{i-1})).$$

Using similar relations for the (i+4)-th up to the (i+7)-th iteration step we obtain

$$\varepsilon_{i+7} = O(\varepsilon_{i+3}^9\varepsilon_{i+2}^4(\varepsilon_{i+2}\varepsilon_{i+1}^2+\varepsilon_{i+3})) < O(\varepsilon_{i+3}^9\varepsilon_{i-1}^{29}) .$$

Therefore, the effective asymptotic order of convergence is at least equal to $\sqrt[4]{\zeta}$, where ζ denotes the largest positive root of the equation $x^2 - 9x - 29 = 0$, which approximately equals 11.52. So, $\sqrt[4]{\zeta} \cong 1.842$, which is larger than p_2. This completes the proof of the theorem. □

REMARK. In fact, for analytic functions having a simple zero, it can be shown that no modified steps will asymptotically occur in the iteration of algorithm R. So, the asymptotic order of convergence of algorithm R is as large as that of an iteration process using 3-point rational interpolation throughout.

5. NUMERICAL RESULTS

We have compared five algorithms for calculating a zero of a function of one variable.

- Algorithm A, published by DEKKER [3] and described in section 2.
- Algorithm M, defined in section 3.
- Algorithm R, defined in section 4.
- Algorithm B, published by BRENT [2] (see section 1).
- Algorithm C, published by ANDERSON and BJORCK [1] (see section 1).

For testing these algorithms we have chosen four groups of test functions.

I. Some functions with a simple zero in the interval considered. These functions are (see also DOWELL & JARRATT [4]):

1. $f(x) = \sin(x) - 0.5$,
   on the interval [0,1.5];
2. $f(x) = 2x\exp(-n) + 1 - 2\exp(-nx)$,
   on the interval [0,1] and n=1,2,3 and 4;
3. $f(x) = (1+(1-n)^2)x - (1-nx)^2$,
   on the interval [0,1], and n=1,5 and 10;
   these functions have one turning point on [0,1];
4. $f(x) = x^2 - (1-x)^n$,
   on the interval [0,1], and n=1,5 and 10;
   these function have one inflexion on [0,1];
5. $f(x) = (1+(1-n)^4)x - (1-nx)^4$,
   on the interval [0,1], and n=1,4 and 8;
   these functions have one turning point and one inflexion on [0,1];
6. $f(x) = (x-1)\exp(-nx) + x^n$,
   on the interval [0,1], and n=1,5 and 10;
   this is a family of curves increasingly close to the x-axis for large n.

II. Some functions of the form

$f(x) = x^n + ax + b$,
where n=3,5,9 and 19, and

1. a = 1 and b = 0;

2. $a = 0$ and $b = {}_{10}-4$;

3. $a = 1$ and $b = {}_{10}-4$.

These functions have a simple zero and an inflexion point of the order n-1 or n at the zero or in its neighbourhood.

III. Some simple polynomials with a multiple zero.
$f(x) = x^n$,
on the interval [-1,10] and
n=3,5,7,9,19 and 25;
these functions have a zero of multiplicity n.

IV. A function given by BRENT [2] for which all the derivatives vanish at the zero of the function ("multiplicity ∞").
This function is defined by

$$f(x) = \begin{cases} 0 & \text{if } x = 0 \\ x \exp(-x^{-2}) & \text{otherwise.} \end{cases}$$

The interval is chosen to be [-1,4].

The testing has been performed on a Cyber 73 computer, which has a machine precision of 48 bits. In all examples the tolerance function is chosen to be $\delta(x) = |x| \times {}_{10}-14 + {}_{10}-14$.

The results for these groups of testfunctions are given in tables 5.1 to 5.4. In these tables we give the number of function evaluations needed by the various algorithms to find a zero of the given function within the given precision.

Tabel 5.1 illustrates that algorithm M behaves almost the same as algorithm A for simple zeroes, while algorithm R, B and C are slightly better. The better results for algorithm R are due to the use of the higher order rational interpolation formula (3.1.5) throughout. The better behaviour of algorithm B and C is caused by replacing each linear extrapolation step by an inverse quadratic interpolation step (in algorithm B, see BRENT [2]) or a rational extrapolation step (in algorithm C, see ANDERSON & BJORCK [1]). Hence in algorithms R, B and C we save roughly 10% of the number of

function evaluations at the cost of slightly more complicated calculations.

table 5.1

testfunctions of group I

function	n	number of function evaluations				
		A	M	R	B	C
1	-	10	10	9	8	9
2	1	9	9	7	8	7
	2	10	10	8	9	8
	3	11	11	9	10	9
	4	12	12	10	10	10
3	1	10	9	8	8	9
	5	10	10	9	9	8
	10	9	9	9	9	8
4	1	9	10	8	9	9
	5	10	10	9	9	10
	10	11	11	11	10	11
5	1	10	10	8	9	9
	4	9	9	9	8	8
	8	7	7	8	7	8
6	1	9	9	8	9	9
	5	9	9	9	9	9
	10	10	10	10	9	10
total		165	165	149	150	151

From table 5.2 we see that algorithm R,C and M are better than algorithm B for finding a simple zero of a function with a high order inflexion point at or near the zero.

table 5.2

testfunctions of group II

			number of function evaluations				
a	b	n	A	M	R	B	C
1	0	3	11	12	11	15	12
		5	10	10	10	14	12
		9	10	13	11	16	12
		19	10	13	13	16	12
0	10^{-4}	3	21	26	17	26	21
		5	22	26	18	27	23
		9	23	27	19	25	24
		19	23	27	19	24	24
1	10^{-4}	3	11	12	11	14	12
		5	10	10	10	14	11
		9	10	10	11	16	11
		19	10	13	13	16	11
total			171	199	163	223	185

Finally, tables 5.3 and 5.4 show clearly that algorithm A and also algorithm C are not efficient for calculating multiple zeroes. They may cause a computer program running out of time very quickly.

table 5.3

testfunctions of group III

n	number of function evaluations				
	A	M	R	B	C
3	117	151	91	147	118
5	206	149	163	122	207
7	293	161	206	138	294
9	380	160	196	137	381
19	802	179	206	141	759
25	1320	159	174	123	961
total	3118	959	1036	808	2720

table 5.4

function IV

number of function evaluations				
A	M	R	B	C
>5000	27	23	18	969

6. CONCLUSIONS

From the results given in section 5 it is obvious that algorithm A and C are not efficient for practical use on a computer if the multiplicity of the zero is not known in advance.

Although, in most cases, the results of algorithm B are slightly better than those of algorithm M, this is only due to the use of a more complicated formula in roughly 30% of the iteration steps. Moreover, there are examples (see table 5.2) for which algorithm M requires fewer function evaluations than algorithm B. So, for rather simple functions, whose evaluation is cheap

with respect to the calculations performed in one iteration step of algorithm M, we recommend the use of algorithm M, also, because the upper bound of the number of function evaluations needed is better than for algorithm B (see theorem 3.3.1). Algorithm R is to be preferred for more expensive functions, because of the higher asymptotic order of convergence of the interpolation formula used in this algorithm (see theorem 4.3.2). This statement is affirmed by the numerical results in section 5. For functions having poles near the zero we also advise the use of algorithm R, because of the special character of the interpolating function used in this algorithm.

7. REFERENCES

[1] ANDERSON, N. & BJORCK, A., *A new high order method of regula falsi type for computing a root of an equation*, BIT 13 (1973) 253-264.

[2] BRENT, R.P., *An algorithm with guaranteed convergence for finding a zero of a function*, Comp. J. 14 (1971) 422-425.

[3] DEKKER, T.J., *Finding a zero by means of successive linear interpolation*. In: Dejon, B. & Henrici, P. (eds.), *Constructive aspects of the fundamental theorem of algebra*, Wiley Interscience, London, 1969.

[4] DOWELL, M. & JARRATT, P., *A modified regula falsi method for computing the root of an equation*, BIT 11 (1971) 168-174.

[5] JARRATT, P. & NUDDS, D., *The use of rational functions in the iterative solution of equations on a digital computer*, Comp. J. 8 (1965) 62-65.

8. APPENDIX: ALGOL 60 procedures

In this appendix we give the text of two ALGOL 60 procedures, implementing algorithms M and R, defined in sections 3 and 4.

The heading of the procedure implementing algorithm M reads:

```
Boolean procedure zeroin (x,y,fx,tolx);
real x,y,fx,tolx;
```

The heading of the procedure implementing algorithm R reads:

```
Boolean procedure zeroinrat (x,y,fx,tolx);
real x,y,fx,tolx;
```

The meaning of the formal parameters is:

x,y : real variables;
entry: the endpoints of the interval J_1 (see 2.4.1);
exit : if the value of the procedure identifier is *true*, then the values of x and y satisfy (2.2.1);

fx : real expression depending on x; the actual value of fx should be equal to the function value at the point given by the actual value of x;

tolx : real expression depending on x; the actual value of tolx should be equal to the value of the tolerance function at the point given by the actual value of x;

the procedure identifier will have the value true on exit if two argument values x and y are found which satisfy (2.2.1), otherwise the value of the procedure identifier will be false on exit. The last case can only occur if, on entry, the values of x and y do not satisfy $f(x) \times f(y) \leq 0$.

Note that in the procedures we have written

```
... if p × 1 = 0 v ...
```

instead of

```
... if p = 0 v ... .
```

This is done because of the poor arithmetic of the Cyber 73 for values around the smallest positive representable number.
On this computer, it can occur that the Boolean expression p = 0 has the value false while the expressions p/1 and p × 1 have the value 0.
So, replacing the expression p = 0 by p × 1 = 0 removes the difficulty, at least in those cases that we checked.

```
Boolean procedure zeroin(x, y, fx, tolx);
real x, y, fx, tolx;
begin integer ext;
    real c, fc, b, fb, a, fa, d, fd, fdb, fda, w, mb,
    tol, m, p, q;
    b:= x; fb:= fx; a:= x:= y; fa:= fx;
interpolate: c:= a; fc:= fa; ext:= 0;
extrapolate: if abs(fc) < abs(fb) then
    begin if c ≠ a then begin d:= a; fd:= fa end;
        a:= b; fa:= fb; b:= x:= c; fb:= fc; c:= a; fc:= fa
    end interchange;
    tol:= tolx; m:= (c + b) × 0.5; mb:= m - b;
    if abs(mb) > tol then
    begin if ext > 2 then w:= mb else
        begin tol:= tol × sign(mb);
            p:= (b - a) × fb; if ext ≤ 1 then
            q:= fa - fb else
            begin fdb:= (fd - fb) / (d - b);
                fda:= (fd - fa) / (d - a);
                p:= fda × p; q:= fdb × fa - fda × fb
            end; if p < 0 then
            begin p:= -p; q:= -q end;
            w:= if p × 1 = 0 ∨ p ≤ q × tol then tol else
            if p < mb × q then p / q else mb
        end; d:= a; fd:= fa; a:= b; fa:= fb;
        x:= b:= b + w; fb:= fx;
        if (if fc ≥ 0 then fb ≥ 0 else fb ≤ 0) then
        goto interpolate else
        begin ext:= if w = mb then 0 else ext + 1;
            goto extrapolate
        end
    end; y:= c;
    zeroin:= if fc ≥ 0 then fb ≤ 0 else fb ≥ 0
end zeroin;
```

```
Boolean procedure zeroinrat(x, y, fx, tolx);
real x, y, fx, tolx;
begin integer ext; boolean first;
    real b, fb, a, fa, d, fd, c, fc, fdb, fda, w,
    mb, tol, m, p, q;
    b:= x; fb:= fx; a:= x:= y; fa:= fx; first:= true;
interpolate: c:= a; fc:= fa; ext:= 0;
extrapolate: if abs(fc) < abs(fb) then
    begin if c ≠ a then begin d:= a; fd:= fa end;
        a:= b; fa:= fb; b:= x:= c; fb:= fc; c:= a; fc:= fa
    end interchange;
    tol:= tolx; m:= (c + b) × .5; mb:= m - b;
    if abs(mb) > tol then
    begin if ext > 3 then w:= mb else
        begin tol:= tol × sign(mb);
            p:= (b - a) × fb; if first then
            begin q:= fa - fb; first:= false end else
            begin fdb:= (fd - fb) / (d - b);
                fda:= (fd - fa) / (d - a);
                p:= fda × p; q:= fdb × fa - fda × fb
            end; if p < 0 then
            begin p:= -p; q:= -q end;
            if ext = 3 then p:= p × 2;
            w:= if p × 1 = 0 ∨ p ≤ q × tol then tol else
            if p < mb × q then p / q else mb
        end; d:= a; fd:= fa; a:= b; fa:= fb;
        x:= b:= b + w; fb:= fx;
        if (if fc ≥ 0 then fb ≥ 0 else fb ≤ 0) then
        goto interpolate else
        begin ext:= if w = mb then 0 else ext + 1;
            goto extrapolate
        end
    end; y:= c;
    zeroinrat:= if fc ≥ 0 then fb ≤ 0 else fb ≥ 0
end zeroinrat;
```