MATHEMATICS OF COMPUTATION
VOLUME 61, NUMBER 204
OCTOBER 1993, PAGES 733-744

ON ENCLOSING SIMPLE ROOTS OF NONLINEAR EQUATIONS

G. ALEFELD, F. A. POTRA, AND YIXUN SHI

ABSTRACT. In this paper we present two efficient algorithms for enclosing a simple root of the nonlinear equation $f(x) = 0$ in the interval $[a, b]$. They improve recent methods of Alefeld and Potra by achieving higher efficiency indices and avoiding the solution of a quadratic equation per iteration. The efficiency indices of our methods are $1.5537\ldots$ and $1.618\ldots$, respectively. We show that our second method is an optimal algorithm in some sense. Our numerical experiments show that the two methods of the present paper compare well with the above methods of Alefeld and Potra as well as efficient solvers of Dekker, Brent, and Le. The second method in this paper has the best behavior of all, especially when the termination tolerance is small.

1. INTRODUCTION

In a recent paper, Alefeld and Potra [2] proposed three efficient methods for enclosing a simple zero x_* of a continuous function $f(x)$ in the interval $[a, b]$ provided that $f(a)f(b) < 0$. Starting with the initial enclosing interval $[a_1, b_1] = [a, b]$, the methods produce a sequence of intervals $\{[a_n, b_n]\}_{n=1}^{\infty}$ such that

$$x_* \in [a_{n+1}, b_{n+1}] \subseteq [a_n, b_n] \subseteq \cdots \subseteq [a_1, b_1] = [a, b], \tag{1}$$

$$\lim_{n\to\infty} (b_n - a_n) = 0. \tag{2}$$

The asymptotic efficiency indices of each of those three methods, in the sense of Ostrowski [10], are $\sqrt{2} = 1.4142\ldots$, $\sqrt[3]{4} = 1.5874\ldots$, and $\sqrt[3]{(3+\sqrt{13})/2} = 1.4892\ldots$, respectively. The numerical experiments in that paper show that the practical behavior of those methods is comparable to that of the efficient equation solvers of Dekker [6] and Brent [5], although they perform slightly worse on some problems.

Although there are many enclosing methods for solving the equation

$$f(x) = 0, \tag{3}$$

where $f(x)$ is continuous on $[a, b]$ and has a simple root x_* in $[a, b]$, most of them do not have nice asymptotic convergence properties of the diameters $\{(b_n - a_n)\}_{n=1}^{\infty}$. For example, in case of Dekker's method, the diameters $b_n - a_n$ may remain greater than a relative large positive quantity until the last iteration

Received by the editor June 26, 1991 and, in revised form, June 24, 1992.
1991 *Mathematics Subject Classification*. Primary 65H05.

when a "δ-step" is taken. In case of Le's Algorithm LZ4 of [8], the convergence properties of $\{(b_n - a_n)\}_{n=1}^{\infty}$ have not been proved except that the total number of function evaluations will be bounded by four times that needed by the bisection method, which is also the upper bound of the number of function evaluations required by our second method in this paper. For other examples, like Brent's method, the Illinois method, the Anderson-Björck method, the Regula Falsi method, Snyder's method, the Pegasus method, and so on, only the convergence rate of $\{|x_n - x_*|\}_{n=1}^{\infty}$, where x_n is the current estimate of x_*, has been studied and not the convergence rate of the diameters $(b_n - a_n)$.

In case $f(x)$ is convex on $[a, b]$, the classical Newton-Fourier method [10, p. 248], J. W. Schmidt's method [12], and the methods of Alefeld and Potra [1] produce a sequence of enclosing intervals whose diameters are superlinearly convergent to zero. The highest asymptotic efficiency index of those methods, $1.5537\ldots$, is attained by a method of J. W. Schmidt [12] and a slight modification of this method due to Alefeld and Potra [1].

In the paper of Alefeld and Potra [2] three iterative methods are proposed that produce enclosing intervals satisfying (1) and (2) without any convexity assumptions on f. Surprisingly enough, under appropriate smoothness assumptions, one of the methods of [2] has the efficiency index $1.5874\ldots$, which is higher than the efficiency index of the above-mentioned method of J. W. Schmidt [12].

In the present paper two new algorithms for enclosing zeros of nonconvex functions are presented. Our first method requires at most 3, while our second method requires at most 4 function evaluations per step. Both methods reduce the length of the enclosing interval by at least one half at each step, so that in the worst case scenario our methods require 3 times, respectively 4 times, more function evaluations than the bisection method. As the bisection method, or the methods of Brent [5], Dekker [6], or Le [8, 9], our methods are applicable to rather general problems involving discontinuous functions and derivatives, multiple zeros, etc. (see Theorem 3.1). However, in case of simple zeros of C^3-functions we can prove that, asymptotically, our first method requires only 2, and our second method only 3 function evaluations per step. Moreover, in this case the sequence of diameters $\{(b_n - a_n)\}_{n=1}^{\infty}$ converges to zero with R-order at least $1 + \sqrt{2} = 2.414\ldots$ for our first method, and R-order at least $2 + \sqrt{5} = 4.236\ldots$ for our second method. Hence the corresponding efficiency indices are $\sqrt{1+\sqrt{2}} = 1.5537\ldots$ and $\sqrt[3]{2+\sqrt{5}} = (1 + \sqrt{5})/2 = 1.618\ldots$, respectively. As far as we know, the latter is the highest efficiency index for iterative methods that produce monotone enclosing intervals for simple zeros of sufficiently smooth functions.

This paper improves the results of [2] in two ways. First, by making better use of available information, we obtain a higher efficiency index. Second, our new algorithms do not use the exact solution of a quadratic equation at each step. Instead, we use 2 or 3 Newton steps to get a convenient approximation. This modification saves the work of computing the square root, makes the subroutine program much simpler, and preserves the good convergence properties. For convenience of comparison, we list the three algorithms of [2] in the Appendix of this paper.

In our numerical experiments we compared our methods with the methods in [2], with the methods of Dekker [6] and Brent [5] which are used in many standard software packages, and also with the Algorithm LZ4 of Le [8]. The results are presented in §5. The numerical results show that the two methods of the present paper compare well with the other six methods. The second method in this paper has the best behavior of all, especially when the termination tolerance is small.

In §6, we show that in a certain sense our second method is an optimal procedure.

2. Preliminary subroutines and lemmas

In this section we present some notations and results to be used later. We assume throughout that $f(x)$ is continuous on $[a, b]$ and that $f(a)f(b) < 0$. We consider a point $c \in [a, b]$.

```
Subroutine bracket(a, b, c, ā, b̄, d).
    If $f(c) = 0$, then print $c$ and stop;
    If $f(a)f(c) < 0$, then $\bar{a} = a$, $\bar{b} = c$, $d = b$;
    If $f(b)f(c) < 0$, then $\bar{a} = c$, $\bar{b} = b$, $d = a$. □
```

After calling the above subroutine, we will have a new interval $[\bar{a}, \bar{b}] \subset [a, b]$ with $f(\bar{a})f(\bar{b}) < 0$. Furthermore, we will have a point $d \notin [\bar{a}, \bar{b}]$ such that if $d < \bar{a}$ then $f(\bar{a})f(d) > 0$; otherwise $f(d)f(\bar{b}) > 0$.

```
Subroutine Newton-Quadratic(a, b, d, r, k).
    Set $A = f[a, b, d]$, $B = f[a, b]$;
    If $A = 0$, then $r = a - B^{-1} \cdot f(a)$;
    If $A \cdot f(a) > 0$, then $r_0 = a$, else $r_0 = b$;
    For $i = 1, 2, \ldots, k$ do:
(4)     $r_i = r_{i-1} - \frac{P(r_{i-1})}{P'(r_{i-1})} = r_{i-1} - \frac{P(r_{i-1})}{B + A(2r_{i-1} - a - b)}$,
    $r = r_k$. □
```

The above subroutine has a, b, d, and k as inputs and r as output. It is assumed that $d \notin [a, b]$ and that $f(d)f(a) > 0$ if $d < a$ and $f(d)f(b) > 0$ if $d > b$. Furthermore, k is a positive integer and r is an approximation of the unique zero z of the quadratic polynomial

$$P(x) = P(a, b, d)(x) = f(a) + f[a, b](x - a) + f[a, b, d](x - a)(x - b)$$

in $[a, b]$, where $f[a, b] = (f(b) - f(a))/(b - a)$, and $f[a, b, d] = (f[b, d] - f[a, b])/(d - a)$; note that, $P(a) = f(a)$ and $P(b) = f(b)$. Hence, $P(a)P(b) < 0$.

Lemma 2.1. (i) *Under the above assumptions,* $r \in (a, b)$.

(ii) *Furthermore, if* $\{a, b, d\} \subseteq [e, f]$, *and if* $f(x)$ *is twice continuously differentiable in* $[e, f]$ *with* $f'(x) \neq 0$ *for all* $x \in [e, f]$ *and*

$$\delta = \min_{e \leq x \leq f} |f'(x)| - (b - a) \max_{e \leq x \leq f} |f''(x)| > 0,$$

then

$$|r - z| \leq \lambda^L (b - a)^{2^k}, \quad \text{where } \lambda = \frac{\max_{e \leq x \leq f} |f''(x)|}{2\delta}, \quad L = 2^k - 1. \tag{5}$$

Proof. (i) follows from the monotone convergence of Newton's method on quadratics, while (ii) follows by remarking that

$$|A| \le \frac{\max_{e \le x \le f} |f''(x)|}{2}$$

and $|P'(x)| \ge \delta > 0$ for all $x \in [a, b]$. Indeed, we have that

$$|r_k - z| = |r_{k-1} - z|^2 \frac{|A|}{|P'(r_{k-1})|} \le |r_{k-1} - z|^2 \lambda \le \lambda^L |r_0 - z|^{2^k}, \tag{6}$$

where $L = 1 + 2 + \cdots + 2^{k-1} = 2^k - 1$. □

The next lemma can be proved in a straightforward manner; it will be needed in §6.

Lemma 2.2. *Let* $I_n = (n + \sqrt{1 + n^2})^{1/(n+1)}$ *for* $n = 1, 2, 3, \ldots$; *then* $I_2 > I_n$ *for all* $n \neq 2$.

3. Algorithms

In this section we present two algorithms for enclosing a simple zero x_* of a continuous function $f(x)$ in $[a, b]$ where $f(a)f(b) < 0$. These two algorithms are improvements of the methods in [2]. The first algorithm requires at most 3, and asymptotically 2, function evaluations per iteration, and the second algorithm requires at most 4, and asymptotically 3, function evaluations per iteration. Under certain assumptions the first algorithm has an asymptotic efficiency index $\sqrt{1 + \sqrt{2}} = 1.5537\ldots$ and the second algorithm has an asymptotic index $(1 + \sqrt{5})/2 = \sqrt[3]{2 + \sqrt{5}} = 1.6180\ldots$. In the following algorithms, $\mu < 1$ is a positive parameter which is usually chosen as $\mu = 0.5$.

Algorithm 1.

```
1.1 set $a_1 = a$, $b_1 = b$, $c_1 = a_1 - f[a_1, b_1]^{-1} f(a_1)$;
1.2 call bracket($a_1, b_1, c_1, a_2, b_2, d_2$);
For $n = 2, 3, \ldots$, do:
1.3 call Newton-Quadratic($a_n, b_n, d_n, c_n, 2$);
1.4 call bracket($a_n, b_n, c_n, \bar{a}_n, \bar{b}_n, \bar{d}_n$);
1.5 if $|f(\bar{a}_n)| < |f(\bar{b}_n)|$, then set $u_n = \bar{a}_n$, else set $u_n = \bar{b}_n$;
1.6 set $\bar{c}_n = u_n - 2f[\bar{a}_n, \bar{b}_n]^{-1} f(u_n)$;
1.7 if $|\bar{c}_n - u_n| > 0.5(\bar{b}_n - \bar{a}_n)$, then $\hat{c}_n = 0.5(\bar{b}_n + \bar{a}_n)$, else $\hat{c}_n = \bar{c}_n$;
1.8 call bracket($\bar{a}_n, \bar{b}_n, \hat{c}_n, \hat{a}_n, \hat{b}_n, \hat{d}_n$);
1.9 if $\hat{b}_n - \hat{a}_n < \mu(b_n - a_n)$, then $a_{n+1} = \hat{a}_n$, $b_{n+1} = b_n$, $d_{n+1} = \hat{d}_n$, else call
    bracket($\hat{a}_n, \hat{b}_n, 0.5(\hat{a}_n + \hat{b}_n), a_{n+1}, b_{n+1}, d_{n+1}$). □
```

Algorithm 2.

```
2.1–2.2: same as 1.1–1.2;
For $n = 2, 3, \ldots$, do:
2.3 call Newton-Quadratic($a_n, b_n, d_n, c_n, 2$);
2.4 call bracket($a_n, b_n, c_n, \tilde{a}_n, \tilde{b}_n, \tilde{d}_n$);
2.5 call Newton-Quadratic($\tilde{a}_n, \tilde{b}_n, \tilde{d}_n, \tilde{c}_n, 3$);
2.6 call bracket($\tilde{a}_n, \tilde{b}_n, \tilde{c}_n, \bar{a}_n, \bar{b}_n, \bar{d}_n$);
2.7–2.11: same as 1.5–1.9. □
```

The following theorem is a basic property of the above two algorithms, whose proof is straightforward and hence will be omitted.

Theorem 3.1. *Let f be a real function defined on $[a, b]$ such that $f(a)f(b) < 0$, and consider one of the Algorithms 1 or 2. Then either a zero of f is found in a finite number of steps, or an infinite sequence of intervals $[a_n, b_n]$ is produced such that*

$$\begin{gathered} f(a_n)f(b_n) < 1, \\ a_n \le a_{n+1} \le b_{n+1} \le b_n, \\ b_{n+1} - a_{n+1} \le \tfrac{1}{2}(b_n - a_n), \\ \lim_{n\to\infty} a_n = x_* = \lim_{n\to\infty} b_n, \\ f(x_* - 0)f(x_* + 0) \le 0. \quad \square \end{gathered}$$

Corollary 3.2. *Under the hypothesis of Theorem 3.1, assume that f is continuous at x_*. Then x_* is a zero of f.* □

4. Convergence results

In §3 it is easy to see that the intervals $\{[a_n, b_n]\}_{n=1}^{\infty}$ produced by either Algorithm 1 or Algorithm 2 satisfy that $b_{n+1} - a_{n+1} \le \mu_1(b_n - a_n)$ for $n \ge 2$, where $\mu_1 = \max\{\mu, 0.5\}$. Since $\mu_1 < 1$, this shows at least linear convergence. In what follows we show that under certain smoothness assumptions, Algorithm 1 and Algorithm 2 produce intervals whose diameters $\{(b_n - a_n)\}_{n=1}^{\infty}$ converge to zero with R-orders at least $1 + \sqrt{2} = 2.414\ldots$ and $2 + \sqrt{5} = 4.236\ldots$, respectively.

First, we have the following two lemmas.

Lemma 4.1 (Alefeld and Potra [2]). *Assume that f is continuously differentiable in $[a, b]$ and $f(a)f(b) < 0$, and x_* is a simple zero of $f(x)$ in $[a, b]$. Suppose that Algorithm 1 (or Algorithm 2) does not terminate after a finite number of iterations. Then there is an n_3 such that for all $n > n_3$, the quantities $\bar{c}_n$ and u_n in step 1.6 (or in step 2.8) satisfy that*

$$f(\bar{c}_n)f(u_n) < 0. \tag{7}$$

Lemma 4.2. *Under the assumptions of Lemma 4.1, also assume that $f(x)$ is three times continuously differentiable on $[a, b]$; then*

(i) *for Algorithm 1, there are an $r_1 > 0$ and an n_1 such that for all $n > n_1$*

$$|f(c_n)| \le r_1(b_n - a_n)^2(b_{n-1} - a_{n-1}), \tag{8}$$

where c_n is defined in step 1.3;

(ii) *for Algorithm 2, there are an $r_2 > 0$ and an n_2 such that for all $n > n_2$*

$$|f(\tilde{c}_n)| \le r_2(b_n - a_n)^4(b_{n-1} - a_{n-1}), \tag{9}$$

where $\tilde{c}_n$ is defined in step 2.5.

Proof. By Theorem 3.1, $b_n - a_n \to 0$ and $x_* \in (a_n, b_n)$. Since x_* is a simple root, $f'(x_*) \ne 0$. Therefore, when n is big enough, $f'(x) \ne 0$ for all $x \in [a_n, b_n]$. For simplicity, we assume that $f'(x) \ne 0$ for all $x \in [a, b]$. Also, it is easy to see that in both algorithms we have that

$$b_n - a_n \le \mu(b_{n-1} - a_{n-1}) < (b_{n-1} - a_{n-1}).$$

Since $\lambda_0 = \min_{a\le x\le b}|f'(x)| > 0$ and $b_n - a_n \to 0$, then for any fixed $0 < \delta < \lambda_0$ there is an n_1 such that for all $n > n_1$ we have that $b_n - a_n < 1$ and

$$\delta_n = \lambda_0 - \max_{a\le x\le b}|f''(x)|(b_n - a_n) > \delta > 0. \tag{10}$$

(i) For Algorithm 1, when $n > n_1$, suppose z_n is the unique zero of $P(a_n, b_n, d_n)(x)$ in $[a_n, b_n]$. Then using the error formula for Lagrange interpolation, we see that

$$\begin{aligned}|f(z_n)| &\le \lambda_1|z_n - a_n|\,|z_n - b_n|\,|z_n - d_n|\\ &\le 0.25\lambda_1(b_n - a_n)^2(b_{n-1} - a_{n-1}), \quad \text{where } \lambda_1 = \frac{1}{3!}\max_{a\le x\le b}|f'''(x)|.\end{aligned} \tag{11}$$

By Lemma 2.1 and (10),

$$\begin{aligned}|c_n - z_n| &\le \left(\frac{\max_{a\le x\le b}|f''(x)|}{2\delta_n}\right)^3 (b_n - a_n)^4 < \lambda_2(b_n - a_n)^4\\ &< \lambda_2(b_n - a_n)^2(b_{n-1} - a_{n-1}), \quad \text{where } \lambda_2 = \left(\frac{\max_{a\le x\le b}|f''(x)|}{2\delta}\right)^3.\end{aligned} \tag{12}$$

Combining (11) and (12), we have that

$$|f(c_n)| \le |f(z_n)| + \left(\max_{a\le x\le b}|f'(x)|\right)|c_n - z_n| \le r_1(b_n - a_n)^2(b_{n-1} - a_{n-1}),$$

where $r_1 = 0.25\lambda_1 + \lambda_2 \max_{a\le x\le b}|f'(x)|$.

(ii) For Algorithm 2, when $n > n_1$, we have that

$$|f(c_n)| < r_1(b_n - a_n)^2(b_{n-1} - a_{n-1}), \tag{13}$$

where c_n is given by 2.3. Suppose $\tilde{z}_n$ is the unique zero of $P(a_n, b_n, c_n)(x) = P(\tilde{a}_n, \tilde{b}_n, \tilde{d}_n)(x)$ in $[\tilde{a}_n, \tilde{b}_n]$; then as in Alefeld and Potra [2], we deduce that there is an n_2 (we can choose $n_2 > n_1$) such that for all $n > n_2$

$$|f(\tilde{z}_n)| < \lambda_3(b_n - a_n)^2|f(c_n)|, \quad \text{where } \lambda_3 = 2\left(0.25\frac{\lambda_1}{\lambda_0}\right). \tag{14}$$

Finally, similar to (12), by Lemma 2.1 and (10),

$$\begin{aligned}|\tilde{c}_n - \tilde{z}_n| < \lambda_4(\tilde{b}_n - \tilde{a}_n)^8 &< \lambda_4(b_n - a_n)^4(b_{n-1} - a_{n-1}),\\ &\text{where } \lambda_4 = \left(\frac{\max_{a\le x\le b}|f''(x)|}{2\delta}\right)^7.\end{aligned} \tag{15}$$

Combining (13), (14), and (15), when $n > n_2 > n_1$, we get

$$|f(\tilde{c}_n)| < |f(\tilde{z}_n)| + \max_{a\le x\le b}|f'(x)|\,|\tilde{c}_n - \tilde{z}_n| < r_2(b_n - a_n)^4(b_{n-1} - a_{n-1}) \tag{16}$$

with $r_2 = \lambda_e r_1 + \lambda_4 \max_{a\le x\le b}|f'(x)|$. □

The following two theorems show the asymptotic convergence properties of Algorithm 1 and Algorithm 2, respectively.

Theorem 4.3. *Under the assumptions of Lemma* 4.2, *the sequence of diameters* $\{(b_n - a_n)\}_{n=1}^{\infty}$ *produced by Algorithm* 1 *converges to zero, and there is an* $L_1 > 0$ *such that*

$$(17)\qquad b_{n+1} - a_{n+1} \le L_1(b_n - a_n)^2(b_{n-1} - a_{n-1}), \quad \forall n = 2, 3, \ldots .$$

Moreover, there is an N_1 *such that for all* $n > N_1$ *we have*

$$a_{n+1} = \hat{a}_n \quad and \quad b_{n+1} = \hat{b}_n.$$

Hence, when $n > N_1$, *Algorithm* 1 *requires only two function evaluations per iteration.*

Proof. As in the proof of Lemma 4.2, we assume without loss of generality that $f'(x) \neq 0$ for all $x \in [a, b]$. Take N_1 such that $N_1 > \max\{n_1, n_3\}$. Then by Lemma 4.1, (7) holds for all $n > N_1$. For steps 1.6–1.8 of Algorithm 1 and the fact that $u_n, \bar{c}_n \in [\bar{a}_n, \bar{b}_n]$ we deduce that

$$(18)\qquad \hat{b}_n - \hat{a}_n \le |\bar{c}_n - u_n|, \quad \forall n > N_1.$$

From step 1.6 we also see that

$$(19)\qquad |\bar{c}_n - u_n| = |2f[\bar{a}_n, \bar{b}_n]^{-1} f(u_n)| \le \frac{2}{\lambda_0}|f(u_n)|.$$

Finally, since $c_n \in \{\bar{a}_n, \bar{b}_n\}$, we have that $|f(u_n)| \le |f(c_n)|$. Combining that with (18) and (19), we have

$$(20)\qquad \hat{b}_n - \hat{a}_n \le \frac{2}{\lambda_0}|f(c_n)|, \quad \forall n > N_1.$$

Now by Lemma 4.2, $|f(c_n)| \le r_1(b_n - a_n)^2(b_{n-1} - a_{n-1})$; hence

$$(21)\qquad \hat{b}_n - \hat{a}_n \le \frac{2}{\lambda_0} r_1(b_n - a_n)^2(b_{n-1} - a_{n-1}), \quad \forall n > N_1.$$

Since $\{(b_n - a_n)\}_{n=1}^{\infty}$ converges to zero, if N_1 is large enough, then

$$\hat{b}_n - \hat{a}_n < \mu(b_n - a_n), \quad \forall n > N_1.$$

This shows that for all $n > N_1$ we will have $a_{n+1} = \hat{a}_n$ and $b_{n+1} = \hat{b}_n$. By taking

$$L_1 \ge \max\left\{\frac{2}{\lambda_0} r_1, \frac{(b_{n+1} - a_{n+1})}{(b_n - a_n)^2(b_{n-1} - a_{n-1})}\right\}, \qquad n = 2, 3, \ldots, N_1,$$

and using (21), we obtain (17). □

Corollary 4.4. *Under the assumptions of Theorem* 4.3, $\{\varepsilon_n\}_{n=1}^{\infty} = \{(b_n - a_n)\}_{n=1}^{\infty}$ *converges to zero with an R-order at least* $1 + \sqrt{2} = 2.414\ldots$. *Since, asymptotically, Algorithm* 1 *requires only two function evaluations per iteration, its efficiency index is* $\sqrt{1 + \sqrt{2}} = 1.5537\ldots$.

Proof. By Theorem 4.3, $\{\varepsilon_n\}_{n=1}^{\infty}$ converges to zero and $\varepsilon_{n+1} \le L_1 \varepsilon_n^2 \varepsilon_{n-1}$, for $n = 2, 3, \ldots$, and the result follows by invoking Theorem 2.1 of [11]. □

Theorem 4.5. *Under the assumptions of Lemma* 4.2, *the sequence of diameters* $\{(b_n - a_n)\}_{n=1}^{\infty}$ *produced by Algorithm* 2 *converges to zero, and there is an* $L_2 > 0$ *such that*

$$(22)\qquad b_{n+1} - a_{n+1} \le L_2(b_n - a_n)^4(b_{n-1} - a_{n-1}), \quad \forall n = 2, 3, \ldots .$$

Moreover, there is an N_2 *such that for all* $n > N_2$ *we have*

$$a_{n+1} = \hat{a}_n \quad \textit{and} \quad b_{n+1} - \hat{b}_n.$$

Hence, when $n > N_2$, *Algorithm* 2 *requires only three function evaluations per iteration.*

Proof. The proof is almost the same as that of Theorem 4.3. We assume that $f'(x) \neq 0$ for all $x \in [a, b]$. Take N_2 such that $N_2 > \max\{n_2, n_3\}$. Then, when $n > N_2$, as in the proof of Theorem 4.3, we have that

$$\hat{b}_n - \hat{a}_n \leq \frac{2}{\lambda_0}|f(\tilde{c}_n)|. \tag{23}$$

Now by Lemma 4.2, $|f(\tilde{c}_n)| \leq r_2(b_n - a_n)^4(b_{n-1} - a_{n-1})$. Therefore,

$$\hat{b}_n - \hat{a}_n \leq \frac{2}{\lambda_0} r_2(b_n - a_n)^4(b_{n-1} - a_{n-1}), \quad \forall n > N_2. \tag{24}$$

The rest of the proof is similar to the corresponding part of the proof of Theorem 4.3 and is omitted. □

Corollary 4.6. *Under the assumptions of Theorem* 4.4, $\{\varepsilon_n\}_{n=1}^{\infty} = \{(b_n - a_n)\}_{n=1}^{\infty}$ *converges to zero with an R-order at least* $2 + \sqrt{5} = 4.236\ldots$. *Since asymptotically, Algorithm* 2 *requires only three function evaluations per iteration, its efficiency index is* $\sqrt[3]{2 + \sqrt{5}} = 1.618\ldots$. □

5. Numerical experiments

In this section we present some numerical experiments. We compared our methods with the methods in [2], with the methods of Dekker [6] and Brent [5], and also with the Algorithm LZ4 of Le [8]. In our experiments, the parameter μ in all the methods of this paper and [2] was chosen as 0.5. For Dekker's method we translated the ALGOL 60 routine Zeroin presented in [6] into Fortran; for Brent's method we simply used the Fortran routine *Zero* presented in the Appendix of [5], while for the Algorithm LZ4 of Le we used his Fortran code. The machine used was Encore-Multimax, and double precision was used. The test problems are listed in Table 5.1. The termination criterion was the one used by Brent [5], i.e.

$$b - a \leq 2 \cdot tole(a, b), \tag{25}$$

where $[a, b]$ is the current enclosing interval, and

$$tole(a, b) = 2 \cdot |u| \cdot \underline{macheps} + \underline{tol}.$$

Here, $u \in \{a, b\}$ is such that $|f(u)| = \min\{|f(a)|, |f(b)|\}$, $\underline{macheps}$ is the relative machine precision, which in our case is $2.2204460492504 \times 10^{-16}$, and $\underline{tol}$ is a user-given nonnegative number.

Owing to the above termination criterion, a natural modification of the subroutine *bracket* was employed in our implementations of all the methods in this paper and in [2]. The modified subroutine is the following:

Subroutine $bracket(a, b, c, \overline{a}, \overline{b})$ (or $bracket(a, b, c, \overline{a}, \overline{b}, d)$).

```
   Set $\delta = \lambda \cdot tole(a, b)$ for some user-given fixed $\lambda \in (0, 1)$ (in our experiments we took $\lambda = 0.7$);
   if $b - a \leq 4\delta$, then set $c = (a + b)/2$, goto 10;
   if $c \leq a + 2\delta$, then set $c = a + 2\delta$, goto 10;
   if $c \geq b - 2\delta$, then set $c = b - 2\delta$, goto 10;
10 if $f(c) = 0$, then print $c$ and terminate;
   if $f(a)f(c) < 0$, then $\overline{a} = a$, $\overline{b} = c$, $(d = b)$;
   if $f(b)f(c) < 0$, then $\overline{a} = c$, $\overline{b} = b$, $(d = a)$;
   calculate $tole(\overline{a}, \overline{b})$;
   if $\overline{b} - \overline{a} \leq 2 \cdot tole(\overline{a}, \overline{b})$, then terminate. □
```

In our experiments we tested all the problems listed in Table 5.1 with different user-given $\underline{tol}$ ($\underline{tol} = 10^{-7}, 10^{-10}, 10^{-15}$, and 0). The total number of function evaluations in solving all the problems (145 cases) are listed in Table 5.2, where BR and DE stand for Brent's method and Dekker's method, respectively, and "unsolved" means a problem is not solved within 1000 iterations. From there we see that our two methods compare well with the other six methods. The second method in this paper has the best behavior of all, especially when the termination tolerance is small. This reconfirms the fact that the efficiency

TABLE 5.1. Test problems

function $f(x)$	$[a, b]$	parameter
$\sin x - x/2$ $-2\sum_{i=1}^{20}(2i - 5)^2/(x - i^2)^3$	$[\pi/2, \pi]$ $[a_n, b_n]$ $a_n = n^2 + 10^{-9}$ $b_n = (n + 1)^2 - 10^{-9}$	$n = 1(1)19$
axe^{bx}	$[-9, 31]$	$a = -40, b = -1$ $a = -100, b = -2$ $a = -200, b = -3$
$x^n - a$ $\sin x - 0.5$	$[0, 5]$ $[0.95, 4.05]$ $[0, 1.5]$	$a = 0.2, 1, n = 4(2)12$ $a = 1, n = 8(2)14$
$2xe^{-n} - 2e^{-nx} + 1$	$[0, 1]$	$n = 1(1)5, 15, 20$
$[1 + (1 - n)^2]x - (1 - nx)^2$	$[0, 1]$	$n = 1, 2, 5, 10, 15, 20$
$x^2 - (1 - x)^n$	$[0, 1]$	$n = 1, 2, 5, 10, 15, 20$
$[1 + (1 - n)^4]x - (1 - nx)^4$	$[0, 1]$	$n = 1, 2, 4, 5, 8, 15, 20$
$e^{-nx}(x - 1) + x^n$	$[0, 1]$	$n = 1, 5, 10, 15, 20$
$(nx - 1)/((n - 1)x)$	$[0.01, 1]$	$n = 2, 5, 15, 20$
$\begin{cases} 0 & \text{if } x = 0 \\ xe^{-x^{-2}} & \text{otherwise} \end{cases}$	$[-1, 4]$	
$\begin{cases} \frac{n}{20}(\frac{x}{1.5} + \sin x - 1) & \text{if } x \geq 0 \\ \frac{-n}{20} & \text{otherwise} \end{cases}$	$[-10^4, \pi/2]$	$n = 1(1)40$
$\begin{cases} e - 1.859 & \text{if } x > \frac{2\times 10^{-3}}{1+n} \\ e^{(n+1)x/2\times 10^3} - 1.859 & \text{if } x \in [0, \frac{2\times 10^{-3}}{1+n}] \\ -0.859 & \text{if } x < 0 \end{cases}$	$[-10^4, 10^{-4}]$	$n = 20(1)40$ $n = 100(100)1000$

TABLE 5.2. Total number of function evaluations in solving all the problems listed in Table 5.1

tol	B1	B2	B3	Alg.1	Alg.2	BR	DE	LE
10^{-7}	3139	2895	2580	2800	2604	2693	2658	2643
10^{-10}	3447	2995	2773	2990	2708	2794	2819 1 unsolved	2808
10^{-15}	3672	3017	2948	3134	2746	2860	2955 1 unsolved	2971
0	3714	3041	3007	3137	2793	2873	2936 4 unsolved	3025 3025

TABLE 5.3. Total number of function evaluations in solving $x^n = 0$ in $[-1, 10]$ for $n = 3, 5, 7, 9, 19, 25$

tol	B1	B2	B3	Alg.1	Alg.2	BR	DE	LE	BIS
10^{-7}	402	510	384	355	349	434	1340	185	174
10^{-10}	561	718	529	521	461	611	1987	237	234
10^{-15}	785	1034	721	757	746	867	2 unsolved	377	325
0	2219	2959	1793	2208	1830	2624	6 unsolved	1680	921

index is an asymptotic notion. In order to give an interesting example where methods having higher efficiency index are outperformed by methods with lower efficiency indices, we compare those methods with the bisection method, solving the problem

$$x^n = 0, \qquad n = 3, 5, 7, 9, 19, 25$$

with the initial interval $[a, b] = [-1, 10]$. The results are listed in Table 5.3, where BIS stands for the bisection method.

6. DISCUSSION

We notice that our Algorithm 2 is an optimal procedure in the following sense. It is clear that Algorithm 2 improves our Algorithm 1 by repeating 2.3–2.4 in 2.5–2.6. If we repeat 2.3–2.4 a total of k times, then we get an algorithm of the form

Algorithm 3

```
3.1–3.2: same as 2.1–2.2;
    for $n = 2, 3, \dots,$ do
3.3 call Newton-Quadratic $(a_n, b_n, d_n, c_n, 2)$;
3.4 call bracket$(a_n, b_n, c_n, a_n^{(1)}, b_n^{(1)}, d_n^{(1)})$;
  ⋮
  ⋮
3.2$k$ + 1  call Newton-Quadratic$(a_n^{(k-1)}, b_n^{(k-1)}, d_n^{(k-1)}, \tilde{c}_n, k+1)$;
3.2$k$ + 2  call bracket$(a_n^{(k-1)}, b_n^{(k-1)}, \tilde{c}_n, \bar{a}_n, \bar{b}_n, \bar{d}_n)$;
3.2$k$ + 3 – 3.2$k$ + 7: same as 2.7–2.11. □
```

It is clear that Algorithms 1 and 2 are special cases of Algorithm 3. Furthermore, similar to Theorem 4.3 and Theorem 4.5, we see that for Algorithm 3,

$$(b_{n+1} - a_{n+1}) \leq L_k(b_n - a_n)^{2k}(b_{n-1} - a_{n-1}), \qquad n = 2, 3, \ldots,$$

for some $L_k > 0$. Hence, Algorithm 3 has an R-order at least $\tau = k + \sqrt{1+k^2}$, which is the positive root of the equation $t^2 - 2kt - 1 = 0$. Since asymptotically, Algorithm 3 requires $k+1$ function evaluations per iteration, its efficiency index is $I_k = (k + \sqrt{1+k^2})^{1/(k+1)}$. By Lemma 2.2, $I_k < I_2$, for all $k \neq 2$. Therefore, Algorithm 2 is the optimal choice.

7. Appendix

In what follows we list the three algorithms proposed in Alefeld and Potra [2], assuming that $f(x)$ is continuous on $[a, b]$ and $f(a)f(b) < 0$. For convenience, we use the names B1, B2, and B3 for the first, the second, and the third method in [2], respectively.

Algorithm B1

```
set $a_1 = a$, $b_1 = b$, for $n = 1, 2, \ldots$ do:
B1.1 $c_n = a_n - f[a_n, b_n]^{-1} f(a_n)$;
B1.2–B1.5: same as 1.4–1.7 in Algorithm 1 of this paper;
B1.6 call $bracket(\overline{a}_n, \overline{b}_n, \hat{c}_n, \hat{a}_n, \hat{b}_n)$;
B1.7 if $\hat{b}_n - \hat{a}_n < \mu(b_n - a_n)$, then set $a_{n+1} = \hat{a}_n$, $b_{n+1} = \hat{b}_n$;
         else call $bracket(\hat{a}_n, \hat{b}_n, 0.5(\hat{a}_n + \hat{b}_n), a_{n+1}, b_{n+1})$.  □
```

Algorithm B2

```
set $a_1 = a$, $b_1 = b$, for $n = 1, 2, \ldots$ do:
B2.1 $c_n = a_n - f[a_n, b_n]^{-1} f(a_n)$;
B2.2 call $bracket(a_n, b_n, c_n, \tilde{a}_n, \tilde{b}_n)$;
B2.3 $\tilde{c}_n$ = the unique zero of $P(a_n, b_n, c_n)(x)$ in $[\tilde{a}_n, \tilde{b}_n]$;
B2.4 call $bracket(\tilde{a}_n, \tilde{b}_n, \tilde{c}_n, \overline{a}_n, \overline{b}_n)$;
B2.5–B2.9: same as B1.3–B1.7.  □
```

Algorithm B3

```
set $a_1 = a$, $b_1 = b$, for $n = 1, 2, \ldots$ do:
B3.1 $c_n = 0.5(a_n + b_n)$;
B3.2–B3.6: same as B2.2–B2.6;
B3.7 call $bracket(\overline{a}_n, \overline{b}_n, \overline{c}_n, a_{n+1}, b_{n+1})$.  □
```

Acknowledgment

We thank Dr. D. Le for kindly sending us the Fortran code of his Algorithm LZ4, and an anonymous referee whose comments led to an improved presentation of our results.

Bibliography

1. G. Alefeld and F. A. Potra, *On two higher order enclosing methods of J. W. Schmidt*, Z. Angew. Math. Mech. **68** (1988), 331–337.

2. ——, *Some efficient methods for enclosing simple zeroes of nonlinear equations*, BIT **32** (1992), 334–344.

3. Ned Anderson and Ake Björck, *A new high order method of regula falsi type for computing a root of an equation*, BIT **13** (1973), 253–264.
4. Ken Atkinson, *An introduction to numerical analysis*, Wiley, New York, 1989.
5. R. P. Brent, *Algorithms for minimization without derivatives*, Prentice-Hall, Englewood Cliffs, NJ, 1972.
6. T. J. Dekker, *Finding a zero by means of successive linear interpolation*, Constructive Aspects of the Fundamental Theorem of Algebra (B. Dejon and P. Henrici, eds.), Wiley-Interscience, London, 1969, pp. 37–48.
7. R. F. King, *Methods without secant steps for finding a bracketed root*, Computing **17** (1976), 49–57.
8. D. Le, *An efficient derivative-free method for solving nonlinear equations*, ACM Trans. Math. Software **11** (1985), 250–262.
9. ——, *Three new rapidly convergent algorithms for finding a zero of a function*, SIAM J. Sci. Statist. Comput. **16** (1985), 193–208.
10. A. M. Ostrowski, *Solution of equations in Banach spaces*, Academic Press, New York, 1973.
11. F. A. Potra, *On Q-order and R-order of convergence*, J. Optim. Theory Appl. **63** (1989), 415–431.
12. J. W. Schmidt, *Eingrenzung der Lösungen nichtlinearer Gleichungen mit höherer Konvergenzgeschwindigkeit*, Computing **8** (1971), 208–215.

Institut für Angewandte Mathematik, Universität Karlsruhe, Kaiserstrasse 12, W-7500 Karlsruhe, Germany

Department of Mathematics, University of Iowa, Iowa City, Iowa 52242

Department of Mathematics and Computer Science, Bloomsburg University, Bloomsburg, Pennsylvania 17815