Journal of Computational and Applied Mathematics 236 (2012) 3308–3318

Contents lists available at SciVerse ScienceDirect

Journal of Computational and Applied Mathematics

journal homepage: www.elsevier.com/locate/cam

Iterative methods for solving nonlinear equations with finitely many roots in an interval☆

Beong In Yun *

Department of Informatics and Statistics, Kunsan National University, 573-701, South Korea

ARTICLE INFO

Article history:
Received 11 September 2011
Received in revised form 22 February 2012

Keywords:
Numerical integration method
Nonlinear equation
Multiple root
Extremum

ABSTRACT

In this paper we consider a nonlinear equation $f(x) = 0$ having finitely many roots in a bounded interval. Based on the so-called numerical integration method [B.I. Yun, A non-iterative method for solving non-linear equations, Appl. Math. Comput. 198 (2008) 691–699] without any initial guess, we propose iterative methods to obtain all the roots of the nonlinear equation. In the result, an algorithm to find all of the simple roots and multiple ones as well as the extrema of $f(x)$ is developed. Moreover, criteria for distinguishing zeros and extrema are included in the algorithm. Availability of the proposed method is demonstrated by some numerical examples.

1. Preliminaries

Recently many root finding iterative methods [1–25] have been proposed. In particular, iterative methods for finding multiple roots appeared in the literature [26–39]. However, it should be noted that the convergence of most iterative methods depends on an initial approximation and local behavior of the function $f(x)$ near a root. Well-known traditional root finding methods and relevant convergence analysis can be found in the literature [40–45].

In this paper we consider a nonlinear equation $f(x) = 0$ which has finitely many roots, including multiple roots in general, in a given interval. Based on the so-called numerical integration method (NIM) proposed in [24], we develop a new iterative method to find all of the roots which always provides convergent iterates without worry over the initial approximation or the behavior of the function $f(x)$ near a root. Additionally, we extend the method to search every extremum of $f(x)$ on the interval.

In the next section we introduce a basic algorithm combining NIM and existing iterative methods for a unique simple root in an interval and show that the algorithm generates convergent iterates to the searched root if we only take a sufficiently large number of the integration points in implementing NIM. In Section 3 we extend the method to the case of finitely many simple roots via partitioning the given interval. Furthermore, in Section 4, the method is generalized for finding multiple roots by using a transformation and, additionally, we show that it is also available to find the extrema of $f(x)$. Criteria for distinguishing roots and extrema are suggested. We demonstrate the usefulness of the proposed method by performing several numerical examples and, in the last section, summarize the method with a concluding remark.

☆ This research was supported by Basic Science Research program through the National Research Foundation of Korea (NRF) funded by the Ministry of Education, Science and Technology (2012-0004716).

* Tel.: +82 63 469 4615.
E-mail addresses: paulllyun@gmail.com, biyun@kunsan.ac.kr.

0377-0427/$ – see front matter © 2012 Elsevier B.V. All rights reserved.
doi:10.1016/j.cam.2012.02.037

2. A signum based iterative method for a single root

Assume that a function $f(x)$ has a unique simple zero p^* in an interval (α, β). Referring to the signum based method or the NIM given in [24], we recall that p^* can be represented by the formula

$$p^* = \frac{1}{2}\left\{\alpha + \beta + F(\alpha)\int_\alpha^\beta F(x)dx\right\} \tag{1}$$

where $F(x)$ is the signum function of $f(x)$, that is,

$$F(x) = \operatorname{sgn}(f(x)). \tag{2}$$

Thus, for a large integer $N > 0$, p^* can be approximated as

$$p^* \approx q + \delta F(\alpha)\sum_{j=1}^{N-1} F\left(q + (2j - N)\delta\right) := p_0 \tag{3}$$

where

$$\delta = \frac{\beta - \alpha}{2N}, \qquad q = \frac{\alpha + \beta}{2}. \tag{4}$$

Then, using the initial approximation p_0 defined in (3), we perform an iteration

$$p_{k+1} = \phi(p_k), \quad k = 0, 1, 2, \ldots \tag{5}$$

for an iteration function $\phi(x)$.

A basic algorithm for the aforementioned method, which we call a signum based iteration, may be written as follows:

Algorithm-SI (*Signum Based Iteration*)**.**

[S1] For a large integer $N \geq 0$, take an initial approximation by the NIM on the given interval (α, β):

$$F(x) := \operatorname{sgn}(f(x))$$
$$\delta := (\beta - \alpha)/2N, \qquad p := (\alpha + \beta)/2$$
$$p := p + \delta F(\alpha)\sum_{j=1}^{N-1} F\left(p + (2j - N)\delta\right).$$

[S2] For an iteration functional $\phi(x)$ associated with $f(x)$ and for a large integer $K_{\max} > 0$, perform the iteration:

$k := 0$
while $(k := k + 1) \leq K_{\max}$
  $r := p$
  $p := \phi(r)$
  if $|f(p)| < \tau_1$ or $|p - r| < \tau_2$ stop
($\tau_1,\ \tau_2 > 0$ are sufficiently small numbers for stopping criterion).

The following theorem gives a constraint for the iterates obtained by Algorithm-SI to converge to the searched zero.

Theorem 1. *Let a function $f(x)$ have a unique simple zero p^* in an interval (α, β) and suppose that the iterative method $p_{k+1} = \phi(p_k)$ used in* [S2] *of* Algorithm-SI *is of order $d > 1$, that is,*

$$\left|p^* - p_{k+1}\right| \leq C \cdot \left|p^* - p_k\right|^d, \quad k \geq 0 \tag{6}$$

holds for a constant $0 < C < \infty$. Then the iterates $\{p_k\}$ obtained by Algorithm-SI *converge to p^* as long as the number N satisfies*

$$N > \left(\frac{\beta - \alpha}{2}\right) C^{\frac{1}{d-1}}. \tag{7}$$

Proof. From the assumption (6) we have

$$\left|p^* - p_k\right| \leq C^{\frac{-1}{d-1}}\left\{C^{\frac{1}{d-1}}\left|p^* - p_0\right|\right\}^{d^k}$$

by induction. Moreover, in [25] it was shown that the error of the initial approximation p_0 evaluated in [S1] is

$$\left|p^* - p_0\right| < \delta = \frac{\beta - \alpha}{2N}.$$

Therefore, one can see that the iterates $\{p_k\}$ obtained by Algorithm-SI always converge to the zero p^* if

$$C^{\frac{1}{d-1}} \left|p^* - p_0\right| < C^{\frac{1}{d-1}} \frac{\beta - \alpha}{2N} < 1,$$

namely, N satisfies the inequality (7). □

Theorem 1 implies that the convergence requirement of the proposed iterative method is just to take a sufficiently large number N in the step [S1], which provides a good initial approximation in the result. Contrarily, in implementing an existing iterative method with an initial approximation chosen randomly, there are some cases where the obtained iterates converge very slowly or they do not converge. Thus the method proposed by Algorithm-SI may be a solution of the problem of choosing an appropriate initial approximation for the employed iterative method to avoid failing in convergence.

3. Finitely many simple roots

We suppose that $f(x)$ has M (≥ 2) simple zeros, say, $p^{(1)} < p^{(2)} < \cdots < p^{(M)}$ in an interval (a, b), and set

$$h_{\min} = \min_{1 \leq i \leq M-1} \left|p^{(i+1)} - p^{(i)}\right|. \tag{8}$$

Assume that for a large integer N_0

$$h = \frac{b-a}{N_0} < h_{\min} \tag{9}$$

and take nodes on the interval $[a, b]$ as

$$\xi_k = a + k \cdot h, \quad k = 0, 1, 2, \ldots, N_0. \tag{10}$$

The number of the zeros, M is assumed to be known. If M is unknown, it can be estimated by the formula

$$\frac{1}{2} \sum_{k=1}^{N_0} \{1 - F(\xi_{k-1}) F(\xi_k)\}. \tag{11}$$

Therein $F(x)$ is the signum function of $f(x)$ as defined in (2). One can see that the formula (11) is still useful when ξ_k is a zero of $f(x)$, that is, $f(\xi_k) = 0$ for some $1 \leq k \leq N_0 - 1$.

We introduce an extended algorithm for finding finitely many simple roots as follows.

Algorithm-SIM (*Signum Based Iteration for Many Simple Zeros*)**.**

[SM1] For a function $f(x)$ on an interval (a, b) and for a large integer N_0, set the followings:

$F(x) := \operatorname{sgn}(f(x))$
$h := (b-a)/N_0$.

[SM2] Perform the iterations to find zeros ($p^{(m)}$):

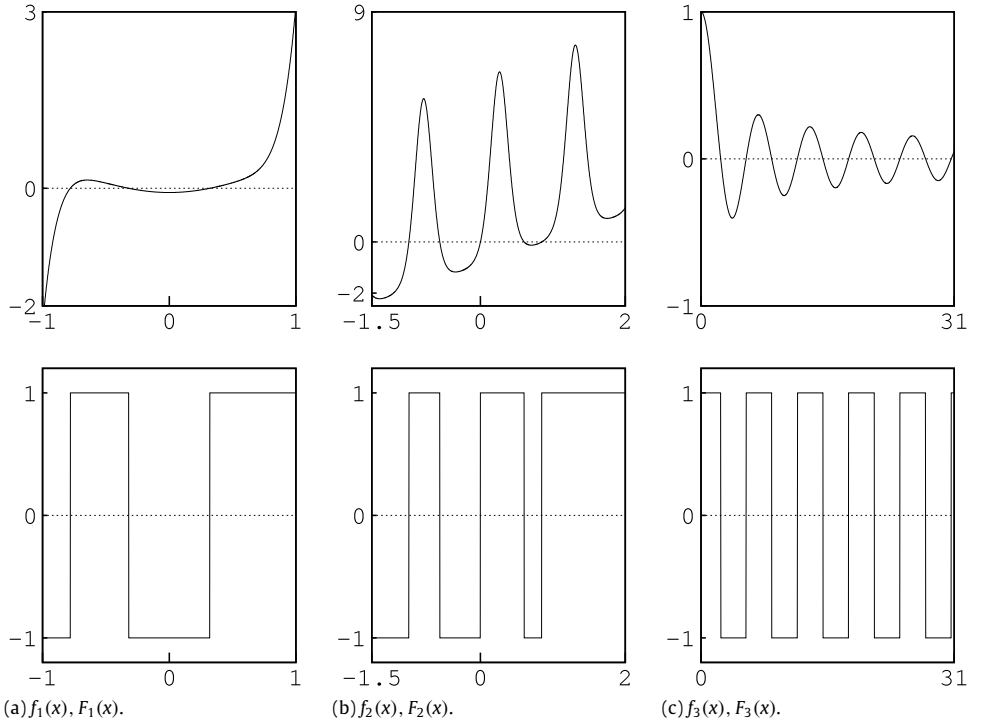
```
j := 0,   m := 0
while (j := j + 1) ≤ N0
    α := a + (j − 1)h,   β := a + jh
    if F(α)F(β) < 0 then
        m := m + 1
        go to Algorithm-SI
        p^(m) := p
    if F(β) = 0 then
        m := m + 1
        p^(m) := β
    end if
```

[SM3] If some zeros are missed, increase N_0 (i.e., decrease h) and repeat [SM2]: (M is a given number of the zeros)

```
if m < M then
    N0 := 2N0
    go to [SM2]
end if
```

Table 1
Numerical results of the proposed method (Algorithm-SIM with $N_0 = 20$ and $N = 10$) for the functions $f_i(x)$, $i = 1, 2, 3$.

$\phi = \phi_N$						
k	$A_k(f_1)$	$ACOC_k$	$A_k(f_2)$	$ACOC_k$	$A_k(f_3)$	$ACOC_k$
2	1.7×10^{-7}	2.0060	1.5×10^{-6}	1.9848	7.7×10^{-10}	2.0755
4	1.4×10^{-26}	2.0000	1.2×10^{-22}	2.0000	2.2×10^{-39}	2.0000
6	6.2×10^{-103}	2.0000	5.2×10^{-87}	2.0000	1.4×10^{-157}	2.0000
$\phi = \phi_O$						
1	9.5×10^{-8}	–	8.0×10^{-7}	–	4.7×10^{-8}	–
2	7.1×10^{-28}	3.9958	4.7×10^{-24}	4.0082	2.8×10^{-30}	4.0336
3	2.2×10^{-108}	4.0000	5.7×10^{-93}	4.0000	3.7×10^{-119}	4.0000



(a) $f_1(x), F_1(x)$. (b) $f_2(x), F_2(x)$. (c) $f_3(x), F_3(x)$.

Fig. 1. Graphs of $f_i(x)$ in the upper row and its signum function $F_i(x)$ in the lower row, $i = 1, 2, 3$.

In implementing Algorithm-SIM, we suppose that the number N satisfies the inequality (7) for an iteration $p_{k+1} = \phi(p_k)$ employed in Algorithm-SI. On the other hand, the step [SM3] is added to ready for a pathological function $f(x)$ which possesses so closed zeros that some subinterval $(\xi_{j-1}, \xi_j) = (\alpha, \beta)$ of length $h = (b - a)/N_0$, for an arbitrarily chosen N_0, may contain plural zeros. In other words, the step [SM3] is a procedure for decreasing the length of the subinterval automatically so that the condition (9) is satisfied.

We now carry out numerical examples, using *Mathematica* with 900 digits of precision, for the functions as follows.

$$f_1(x) = \frac{2}{3} - \left(\frac{1}{10} - x^{11}\right) \exp\left(2 - x^2\right), \quad -1 \le x \le 1$$
$$f_2(x) = \exp\left(2 \sin[6(x - \pi)]\right) + x - 1, \quad -1.5 \le x \le 2$$
$$f_3(x) = J_0(x), \quad 0 \le x \le 31,$$

where J_0 is the Bessel function of the first kind of order 0.

Fig. 1 shows graphs of $f_i(x)$ and $F_i(x) = \mathrm{sgn}(f_i(x))$, $i = 1, 2, 3$. One can see that the functions $f_1(x), f_2(x)$ and $f_3(x)$ have 3, 5 and 10 simple zeros, respectively. Numerical results of the presented method, with $N_0 = 20$ and $N = 10$, for these examples are included in Table 1. For the iteration $p_{k+1} = \phi(p_k)$ in Algorithm-SIM (i.e. in the step [S2] in Algorithm-SI), we employed two traditional iterative methods. One is the Newton method

$$p_{k+1} = \phi_N(p_k) = p_k - \frac{f(p_k)}{f'(p_k)}, \quad k \ge 0 \tag{12}$$

and the other is the Ostrowski method of fourth order

$$p_{k+1} = \phi_O(p_k) = p_k - u(p_k)\frac{f(p_k - u(p_k)) - f(p_k)}{2f(p_k - u(p_k)) - f(p_k)}, \quad k \geq 0 \tag{13}$$

where $u(x) = \frac{f(x)}{f'(x)}$. In Table 1 $A_k(f)$ indicates the maximum absolute value of f of the k-th iterate $p_k^{(m)}$ over the zeros $p^{(m)}$, $1 \leq m \leq M$. In other words,

$$A_k(f) := \max_{1\leq m\leq M} \left|f\left(p_k^{(m)}\right)\right| = \left|f\left(p_k^{(m')}\right)\right| \tag{14}$$

for some $1 \leq m' \leq M$, where M denotes the number of simple zeros of $f(x)$ in a given interval. In addition, the table includes the approximated computational order of convergence (ACOC_k) for the approximations $p_k = p_k^{(m')}$,

$$\mathrm{ACOC}_k := \frac{\log|(p_k - p_{k+1})/(p_{k-1} - p_k)|}{\log|(p_{k-1} - p_k)/(p_{k-2} - p_{k-1})|} \tag{15}$$

as defined in [6]. The numerical results demonstrate the availability of the proposed method. It should be noted that, in implementation of Algorithm-SIM for the examples above, the formula (11) was used to estimate the number M of simple zeros.

4. Multiple roots

Suppose that $f(x)$ is continuously differentiable and has L (≥ 2) zeros and extrema in an interval (a, b), say, $r^{(1)} < r^{(2)} < \cdots < r^{(L)}$. In this section the zero indicates a simple or multiple zero of $f(x)$, in general. The extremum means a local maximum or minimum point r with $f(r) \neq 0$. Similarly to the previous section, set

$$h_{\min} = \min_{1\leq i\leq L-1} \left|r^{(i+1)} - r^{(i)}\right| \tag{16}$$

and under the assumption that for a large integer N_0

$$h = \frac{b-a}{N_0} < h_{\min} \tag{17}$$

we take nodes on the interval $[a, b]$ as

$$\xi_k = a + k \cdot h, \quad k = 0, 1, 2, \ldots, N_0. \tag{18}$$

This implies that every subinterval (ξ_{j-1}, ξ_j) includes one zero or extremum of $f(x)$, at most.

Then, for some $\epsilon > 0$ assumed to be sufficiently small, we employ a transformation $f_\epsilon(x)$ of $f(x)$ which was introduced in [38] such as

$$f_\epsilon(x) = \frac{\epsilon f(x)^2}{f(x + \epsilon f(x)) - f(x)} \tag{19}$$

and its signum function

$$F_\epsilon(x) = \mathrm{sgn}(f_\epsilon(x)). \tag{20}$$

Let r be a zero of $f(x)$ in a subinterval (ξ_{j-1}, ξ_j), that is, for an integer $m \geq 1$ and for a proper function $g(x)$ with $g(x) \neq 0$ on (ξ_{j-1}, ξ_j)

$$f(x) = (x - r)^m g(x).$$

Since $f(x + \epsilon f(x)) - f(x) \approx \epsilon f(x) f'(x)$ in the definition (19), we have for every $\xi_{j-1} < x < \xi_j$

$$f_\epsilon(x) \approx \frac{f(x)}{f'(x)} = \frac{x-r}{m} - \frac{g'(\eta)}{m^2 g(\eta)}(x-r)^2 \tag{21}$$

where η is a point between x and r. This implies that $f_\epsilon(x)$ transforms a simple or multiple zero r of $f(x)$ to a simple zero with

$$F_\epsilon(\xi_{j-1}) = -1, \qquad F_\epsilon(\xi_j) = 1 \tag{22}$$

for h small enough. Therefore, all of the zeros of $f(x)$ can be detected by using the property (22) based on the signum function $F_\epsilon(x)$ of the transformed function $f_\epsilon(x)$.

In practice, Fig. 2 depicts the graphs of $F(x)$ and $F_\epsilon(x)$ for a function $f(x)$ which has two simple zeros ($r^{(1)}$, $r^{(7)}$), one multiple zero ($r^{(5)}$) and four extrema ($r^{(2)}$, $r^{(3)}$, $r^{(4)}$, $r^{(6)}$). It should be noted that, contrary to $F(x)$ in (b), the signum function $F_\epsilon(x)$ in (c) does not miss the multiple zero $r^{(5)}$. The zeros and extrema are indicated by filled circles and unfilled ones,

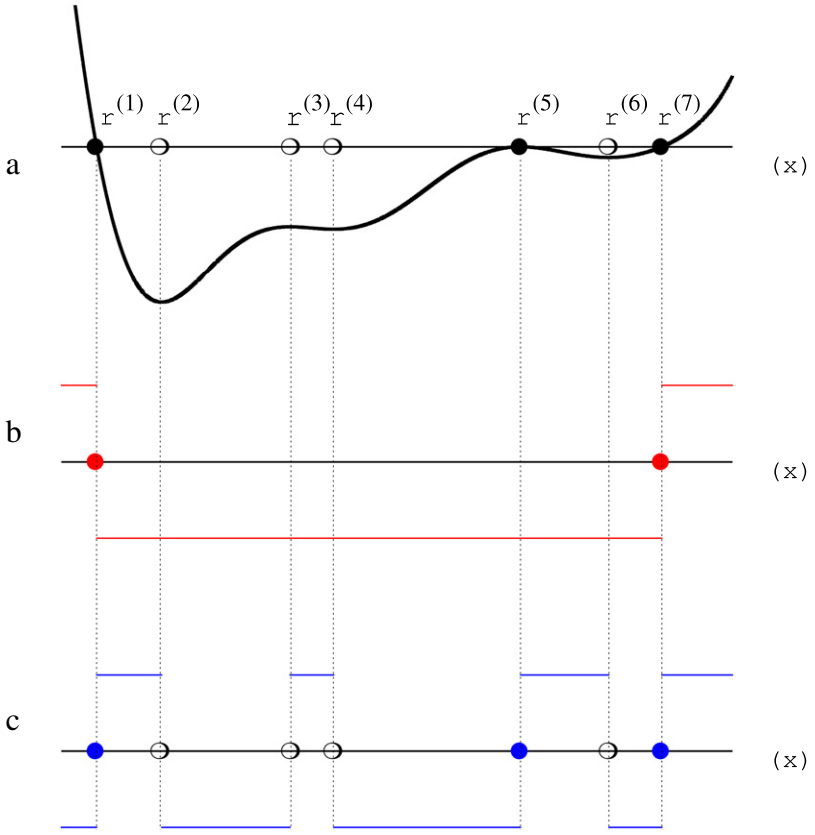


Fig. 2. Graphs of $f(x)$ in (a), $F(x)$ in (b), and $F_\epsilon(x)$ in (c).

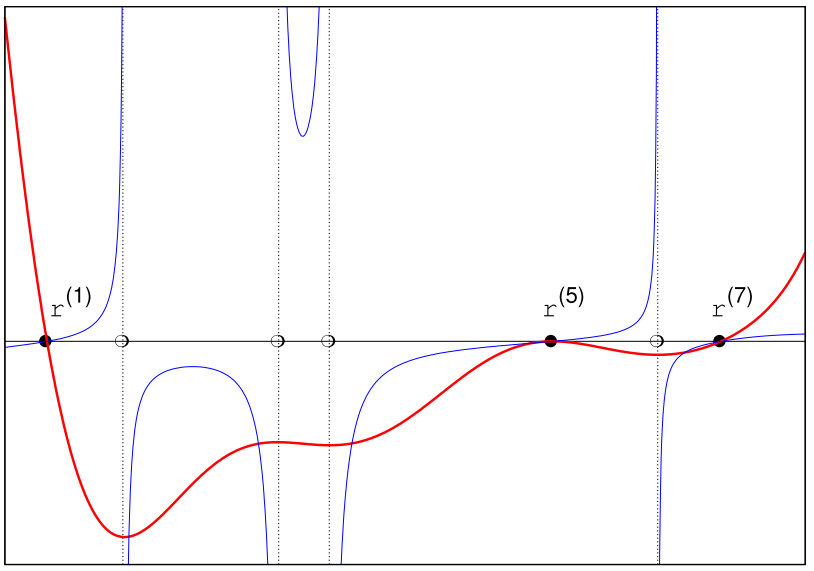


Fig. 3. Graphs of the functions $f(x)$ (: thick line) and $f_\epsilon(x)$ (: thin line).

respectively. Additionally, Fig. 3 shows the graph of $f_\epsilon(x)$ by a thin line where we can observe proper behavior of $f_\epsilon(x)$ near each zero, $r^{(1)}$, $r^{(5)}$ and $r^{(7)}$, while peaks appear near each extremum.

If we search for each subinterval containing a zero by using the property (22), for example, for the case of Fig. 2, the subinterval containing the extremum $r^{(3)}$ will be inevitably searched. In order to avoid this undesirable situation we need another criterion, as provided in the following theorem.

Theorem 2. *Let N_0 be large enough and satisfy (17). Then there exists a unique zero of $f(x)$ in a subinterval (ξ_{j-1}, ξ_j) if and only if, for a sufficiently small $\epsilon > 0$, the function $f_\epsilon(x)$ satisfies*

$$F_\epsilon(\xi_j) - F_\epsilon(\xi_{j-1}) = 2 \tag{23}$$

and

$$f_\epsilon(\xi_j) - f_\epsilon(\xi_{j-1}) < 2(\xi_j - \xi_{j-1}) = 2h. \tag{24}$$

Proof. Let r be a unique simple or multiple zero of $f(x)$ in a subinterval (ξ_{j-1}, ξ_j), and thus let

$$f(x) = (x-r)^m g(x)$$

for an integer $m \geq 1$ and a function $g(x)$ with $g(x) \neq 0$ on (ξ_{j-1}, ξ_j). Then from (21) we have, for sufficiently small $\epsilon > 0$,

$$\begin{aligned} f_\epsilon(\xi_j) - f_\epsilon(\xi_{j-1}) &\leq \frac{1}{m}\left(\xi_j - \xi_{j-1}\right) + \frac{1}{m^2} \max_{\xi_{j-1}<\eta<\xi_j} \left|\frac{g'(\eta)}{g(\eta)}\right| \left\{\left(\xi_j - r\right)^2 + \left(\xi_{j-1} - r\right)^2\right\} \\ &\leq \left(\xi_j - \xi_{j-1}\right) + \max_{\xi_{j-1}<\eta<\xi_j} \left|\frac{g'(\eta)}{g(\eta)}\right| \left(\xi_j - \xi_{j-1}\right)^2. \end{aligned}$$

By the given assumption, we may let N_0 be so large, that is, $h = \xi_j - \xi_{j-1} = (b-a)/N_0$ is so small that

$$\xi_j - \xi_{j-1} < \left(\max_{\xi_{j-1}<\eta<\xi_j} \left|\frac{g'(\eta)}{g(\eta)}\right|\right)^{-1}.$$

Therefore

$$\max_{\xi_{j-1}<\eta<\xi_j} \left|\frac{g'(\eta)}{g(\eta)}\right| \left(\xi_j - \xi_{j-1}\right)^2 < \left(\xi_j - \xi_{j-1}\right)$$

and we have the inequality (24). The Eq. (23) is clear from (22).

For the proof of the converse, under the assumptions that $f_\epsilon(x)$ satisfies (23) and (24) with a sufficiently small $\epsilon > 0$, we recall the relation given in (21) as

$$f_\epsilon(x) \approx \frac{f(x)}{f'(x)}, \qquad \xi_{j-1} \leq x \leq \xi_j.$$

Assume that there is no zero in the interval (ξ_{j-1}, ξ_j). Furthermore, suppose that $f'(x) \neq 0$ for any $x \in (\xi_{j-1}, \xi_j)$. Then there is no change of the sign of $f_\epsilon(x)$ over the interval (ξ_{j-1}, ξ_j), and thus $F_\epsilon(\xi_j) - F_\epsilon(\xi_{j-1}) \neq 2$ which is contradictory to the assumption (23). Next, suppose that $f'(r) = 0$ for some $r \in (\xi_{j-1}, \xi_j)$ with $f(r) \neq 0$. Then form (23) and (21), since the sign of $f(x)$ is not changed on (ξ_{j-1}, ξ_j), we have

$$f'(x) \approx c(x-r)^{2n-1}$$

for some integer $n \geq 1$ and a constant $c \neq 0$. Thus $f(x) \approx \frac{c}{2n}(x-r)^{2n} + f(r)$, and it follows that

$$f_\epsilon(x) \approx \frac{f(r)}{c(x-r)^{2n-1}}.$$

Setting $\frac{f(r)}{c} = d$, since $d > 0$ from (23), we have

$$f_\epsilon(\xi_j) - f_\epsilon(\xi_{j-1}) \approx d\left\{\frac{1}{(\xi_j - r)^{2n-1}} + \frac{1}{(r - \xi_{j-1})^{2n-1}}\right\} > \frac{2d}{h^{2n-1}}.$$

We may suppose that $h < d^{1/2n}$, that is, $h^{2n-1} < d/h$. Then we have

$$f_\epsilon(\xi_j) - f_\epsilon(\xi_{j-1}) > 2h$$

which is contradictory to the assumption (24). Therefore, there should be a zero of $f(x)$ in the interval (ξ_{j-1}, ξ_j). Uniqueness of the root is clear from the assumption in (17). □

Further, similarly to Theorem 2, we can have the following theorem which provides a criterion to search the subinterval containing an extremum.

Theorem 3. *Let N_0 be large enough and satisfy (17). Then there exists a unique extremum of $f(x)$ in a subinterval (ξ_{j-1}, ξ_j) if and only if, for a sufficiently small $\epsilon > 0$, the function $f_\epsilon(x)$ satisfies*

$$\left|F_\epsilon(\xi_j) - F_\epsilon(\xi_{j-1})\right| = 2 \tag{25}$$

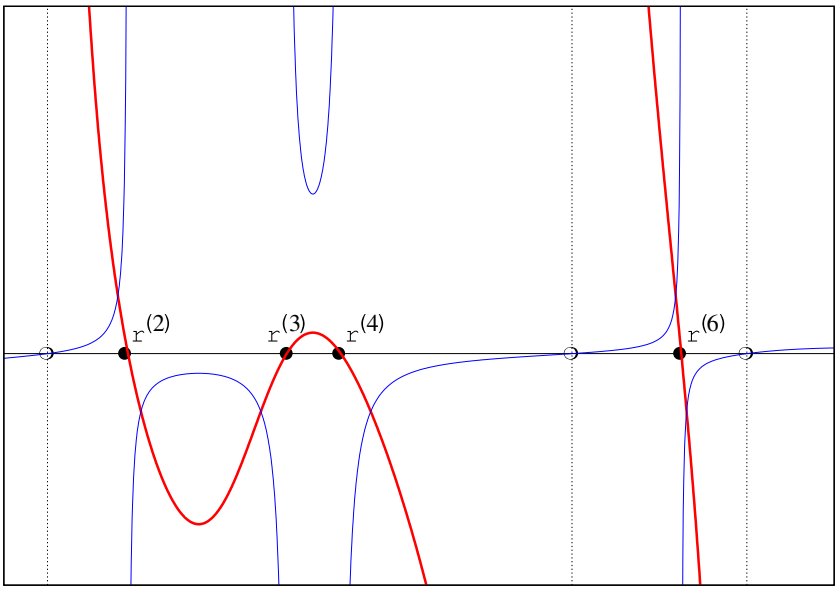


Fig. 4. Graphs of the functions $f_\epsilon(x)$ (: thin line) and $1/f_\epsilon(x)$ (: thick line).

and

$$\left|f_\epsilon(\xi_j) - f_\epsilon(\xi_{j-1})\right| > 2(\xi_j - \xi_{j-1}). \tag{26}$$

We note that, for $\epsilon > 0$ sufficiently small and an integer $n \geq 1$,

$$\frac{1}{f_\epsilon(x)} \approx \frac{f'(x)}{f(x)} \approx c(x-r)^{2n-1} \tag{27}$$

for all x in each subinterval detected by the criteria (25) and (26). Therefore, we may obtain the approximation to the extremum r in the subinterval by substitution, $f(x) = 1/f_\epsilon(x)$ in Algorithm SI. For example, the thick line in Fig. 4 illustrates behavior of $1/f_\epsilon(x)$ for the function $f(x)$ considered in Figs. 2 and 3.

As a result, using (23)–(26) for the criteria of the searched subinterval, we have the following general algorithm to find all the zeros including multiple zeros as well as all the extrema of $f(x)$.

Algorithm-SIG (*Signum Based Iteration for General Cases*)**.**

[SG1] For a function $f(x)$ on an interval (a, b), a large integer N_0, and a small number $\epsilon > 0$, set the followings:

$$f_\epsilon(x) := \epsilon f(x)^2 / \{f(x + \epsilon f(x)) - f(x)\}$$
$$F_\epsilon(x) := \mathrm{sgn}(f_\epsilon(x))$$
$$h := (b-a)/N_0.$$

[SG2] Perform the iterations to find zeros ($p^{(m)}$) and extrema ($q^{(n)}$):

$j := 0, \quad m := 0, \quad n := 0$
while $(j := j+1) \leq N_0$
  $\alpha := a + (j-1)h, \quad \beta := a + jh$
  if $F_\epsilon(\beta) - F_\epsilon(\alpha) = 2$ and $f_\epsilon(\beta) - f_\epsilon(\alpha) < 2h$ then
    $m := m + 1$
    $\boldsymbol{f(x) := f_\epsilon(x)}$
    go to Algorithm-SI
    $p^{(m)} := p$
  if $|F_\epsilon(\beta) - F_\epsilon(\alpha)| = 2$ and $|f_\epsilon(\beta) - f_\epsilon(\alpha)| > 2h$ then
    $n := n + 1$
    $\boldsymbol{f(x) := 1/f_\epsilon(x)}$
    go to Algorithm-SI
    $q^{(n)} := p$
  end if

Table 2
Numerical results of the proposed method (Algorithm-SIG with $N_0 = 20$ and $N = 10$) for the function $f_4(x)$ which has two simple zeros $p^{(1)}$ and $p^{(3)}$, a multiple zero $p^{(2)}$, and four extrema $q^{(n)}$, $n = 1, 2, 3, 4$.

Roots	$\phi = \phi_N$			$\phi = \phi_O$		
m	k	$\lvert p_k^{(m)} - p^{(m)}\rvert$	COC_k	k	$\lvert p_k^{(m)} - p^{(m)}\rvert$	COC_k
1	2	1.2×10^{-12}	2.0004	1	1.0×10^{-12}	–
	4	1.1×10^{-46}	2.0000	2	4.6×10^{-47}	4.0012
	6	6.6×10^{-183}	2.0000	3	1.8×10^{-184}	4.0000
2	2	3.7×10^{-11}	2.0052	1	2.3×10^{-10}	–
	4	2.4×10^{-42}	2.0000	2	2.3×10^{-38}	4.0022
	6	3.8×10^{-167}	2.0000	3	2.1×10^{-150}	4.0000
3	2	1.5×10^{-12}	2.0002	1	6.1×10^{-13}	–
	4	3.3×10^{-46}	2.0000	2	3.4×10^{-48}	4.0006
	6	7.2×10^{-181}	2.0000	3	3.2×10^{-189}	4.0000
Extrema	$\phi = \phi_N$			$\phi = \phi_O$		
n	k	$\lvert q_k^{(n)} - q^{(n)}\rvert$	COC_k	k	$\lvert q_k^{(n)} - q^{(n)}\rvert$	COC_k
1	4	2.2×10^{-53}	2.0000	2	2.7×10^{-58}	4.0002
	6	3.0×10^{-209}	2.0000	3	6.2×10^{-230}	4.0000
2	4	9.4×10^{-30}	2.0000	2	2.8×10^{-29}	3.9935
	6	4.2×10^{-114}	2.0000	3	4.1×10^{-112}	4.0000
3	4	3.8×10^{-29}	2.0000	2	1.6×10^{-28}	4.0085
	6	6.3×10^{-112}	2.0000	3	2.7×10^{-109}	4.0000
4	4	6.0×10^{-35}	2.0000	2	2.0×10^{-30}	3.9894
	6	9.8×10^{-137}	2.0000	3	8.0×10^{-118}	4.0000

[SG3] If some zeros or extrema are missed, increase N_0 and repeat [SG2]: (L is a given number of the zeros and extrema)

```
if m + n < L then
    N_0 := 2N_0
    go to [SG2]
end if
```

In implementing above algorithm, we suppose that the number N satisfies the inequality (7) for an iteration $p_{k+1} = \phi(p_k)$ employed in the step [S2] in Algorithm-SI. When the number L of all zeros and extrema is not known, we may estimate it by using the formula (11) with $F(x)$ replaced by $F_\epsilon(x)$.

We consider two examples:

$$f_4(x) = \left(64x^4 - 16\pi x^3 - 3\pi^2 x^2 + \pi^3 x - \frac{\pi^4}{16}\right)\left(\sin(5x) + \frac{x}{2} + 2\right), \quad -1 \le x \le 1$$

and

$$f_5(x) = (3x-2)^4(2x-3)^2(96x^3 - 332x^2 + 325x - 75), \quad \frac{1}{5} \le x \le 2.$$

The function $f_4(x)$ has two simple zeros $p^{(1)} = -\frac{\pi}{4}$ and $p^{(3)} = \frac{\pi}{4}$, a multiple zero $p^{(2)} = \frac{\pi}{8}$, and four extrema $q^{(n)}$, $n = 1, 2, 3, 4$. In fact, Figs. 2–4 are illustrations for $f(x) = f_4(x)$. The other function $f_5(x)$ has three simple zeros, $p^{(1)} = \frac{1}{3}$, $p^{(3)} = \frac{5}{4}$ and $p^{(5)} = \frac{15}{8}$, two multiple zeros $p^{(2)} = \frac{2}{3}$ and $p^{(4)} = \frac{3}{2}$, and four extrema $q^{(n)}$, $n = 1, 2, 3, 4$.

Tables 2 and 3 include numerical results of the absolute errors for the searched zeros $p^{(m)}$ and extrema $q^{(n)}$ as well as the computational order of convergence (COC_k) defined by

$$COC_k = \frac{\log|(r_k - r)/(r_{k-1} - r)|}{\log|(r_{k-1} - r)/(r_{k-2} - r)|} \tag{28}$$

for the exact value $r = p^{(m)}$ or $r = q^{(n)}$. Therein the exact value of each extremum $q^{(n)}$ is replaced by an iterate $q_K^{(n)}$ with K large enough. The results, with $\epsilon = 10^{-6}$, given in Tables 2 and 3 show that the proposed method according to Algorithm-SIG provides favorable iterates to each zero and extremum. Moreover, the method results in the consistent computational order of convergence for each iteration method employed, regardless of the multiplicity of the searched zero. In general, for further accurate approximation, one may take higher order iterative methods such as, for example, those proposed in the literature [2,6–10,13,30,31,17,18,22,23,39].

Table 3
Numerical results of the proposed method (Algorithm-SIG with $N_0 = 40$ and $N = 10$) for the function $f_5(x)$ which has three simple zeros $p^{(1)}$, $p^{(3)}$, $p^{(5)}$, two multiple zeros $p^{(2)}$ and $p^{(4)}$, and four extrema $q^{(n)}$, $n = 1, 2, 3, 4$.

Roots	$\phi = \phi_N$			$\phi = \phi_O$		
m	k	$\lvert p_k^{(m)} - p^{(m)}\rvert$	COC_k	k	$\lvert p_k^{(m)} - p^{(m)}\rvert$	COC_k
	2	4.3×10^{-10}	1.9994	1	6.9×10^{-11}	–
1	4	1.2×10^{-34}	2.0000	2	1.4×10^{-38}	3.9996
	6	8.1×10^{-133}	2.0000	3	2.1×10^{-149}	4.0000
	2	7.8×10^{-14}	1.9981	1	1.3×10^{-12}	–
2	4	4.3×10^{-54}	2.0000	2	5.6×10^{-48}	3.9994
	6	4.0×10^{-215}	2.0000	3	1.8×10^{-189}	4.0000
	2	1.6×10^{-12}	2.0064	1	2.6×10^{-11}	–
3	4	2.6×10^{-47}	2.0000	2	3.6×10^{-41}	4.0027
	6	2.1×10^{-186}	2.0000	3	1.3×10^{-160}	4.0000
	2	4.1×10^{-10}	2.0029	1	5.0×10^{-10}	–
4	4	1.2×10^{-36}	2.0000	2	3.1×10^{-36}	4.0014
	6	9.8×10^{-143}	2.0000	3	5.0×10^{-141}	4.0000
	2	3.2×10^{-9}	1.9990	1	4.9×10^{-10}	–
5	4	1.4×10^{-31}	2.0000	2	1.2×10^{-35}	3.9993
	6	5.3×10^{-121}	2.0000	3	4.5×10^{-138}	4.0000
Extrema	$\phi = \phi_N$			$\phi = \phi_O$		
n	k	$\lvert q_k^{(n)} - q^{(n)}\rvert$	COC_k	k	$\lvert q_k^{(n)} - q^{(n)}\rvert$	COC_k
1	4	4.1×10^{-26}	2.0043	2	4.4×10^{-29}	4.0012
	6	9.6×10^{-99}	2.0000	3	3.1×10^{-111}	4.0000
2	4	2.1×10^{-43}	1.9972	2	1.8×10^{-40}	3.9988
	6	1.2×10^{-170}	2.0000	3	2.7×10^{-158}	4.0000
3	4	5.6×10^{-56}	1.9757	2	9.7×10^{-44}	3.9892
	6	1.3×10^{-222}	2.0000	3	2.8×10^{-171}	4.0000
4	4	8.5×10^{-30}	1.9973	2	2.0×10^{-32}	3.9994
	6	5.9×10^{-114}	2.0000	3	5.7×10^{-125}	4.0000

5. Conclusions

In this paper we considered a nonlinear equation $f(x) = 0$ which has finitely many simple or multiple roots in general. We ultimately aimed to achieve iterates which always converge to the root invariantly, without worry over the initial approximation, and we proposed some algorithms based on the numerical integration method. The main procedure of the final method described by Algorithm-SIG can be summarized as follows.

1. *Regularization*: Transformation of the function $f(x)$ to $f_\epsilon(x)$ in (19), which converts all of the simple and multiple zeros to the simple ones.
2. *Discrimination*: Searching for each subinterval containing a zero or extremum of $f(x)$, based on the criteria (23)–(26).
3. *Approximation*: Evaluating iterates for the zero or extremum in each initial interval by using Algorithm-SI.

In the result, we have had the advantage of the new method (Algorithm-SIG):

i. It is available for finding all of the zeros and extrema of any function $f(x)$ as long as the assumption (7) is satisfied.
ii. It maintains the order of convergence of the employed iteration $r_{k+1} = \phi(r_k)$, regardless of the multiplicity of the zero.
iii. It will always provide convergent iterates without requiring any proper initial approximation.

Acknowledgments

The author shows his sincere thanks to Professor Miodrag S. Petković for his valuable comments and suggestions on the first draft of this paper. In addition, the author would like to show his gratitude to Professor Anthony Peirce who invited the author to work as a visiting scholar at the University of British Columbia.

References

[1] M. Basto, V. Semiao, F.L. Calheiros, A new iterative method to compute nonlinear equations, Appl. Math. Comput. 173 (2006) 468–483.
[2] W. Bi, H. Ren, Q. Wu, Three-step iterative methods with eighth-order convergence for solving nonlinear equations, J. Comput. Appl. Math. 225 (2009) 105–112.
[3] J. Chen, New modified regula falsi method for nonlinear equations, Appl. Math. Comput. 184 (2007) 965–971.
[4] C. Chun, A family of composite fourth-order iterative methods for solving nonlinear equations, Appl. Math. Comput. 187 (2007) 951–956.
[5] C. Chun, Y. Ham, Some fourth-order modifications of Newton's method, Appl. Math. Comput. 197 (2008) 654–658.
[6] A. Cordero, J.R. Torregrosa, Variants of Newton's method using fifth-order quadrature formulas, Appl. Math. Comput. 190 (2007) 686–698.

[7] A. Cordero, J.R. Torregrosa, M.P. Vassileva, Three-step iterative methods with optimal eighth-order convergence, J. Comput. Appl. Math. 235 (2011) 3189–3194.
[8] J. Dzunić, M.S. Petković, L.D. Petković, A family of optimal three-point methods for solving nonlinear equations using two parametric functions, Appl. Math. Comput. 217 (2011) 7612–7619.
[9] Y.H. Geum, Y.I. Kim, A multi-parameter family of three-step eighth-order iterative methods locating a simple root, Appl. Math. Comput. 215 (2010) 3375–3382.
[10] Y.H. Geum, Y.I. Kim, A uniparametric family of three-step eighth-order multipoint iterative methods for simple roots, Appl. Math. Lett. 24 (2011) 929–935.
[11] H.H.H. Homeier, A modified Newton method with third-order convergence, J. Comput. Appl. Math. 157 (2003) 227–230.
[12] J. Kou, Y. Li, A. Wang, A modification of Newton method with third-order convergence, Appl. Math. Comput. 181 (2006) 1106–1111.
[13] J. Kou, X. Wang, Y. Li, Some eighth-order root-finding three-step methods, Commun. Nonlinear Sci. Numer. Simul. 15 (2010) 536–544.
[14] M.A. Noor, New class of iterative methods for nonlinear equations, Appl. Math. Comput. 191 (2007) 128–131.
[15] Y. Peng, H. Feng, Q. Li, X. Zhang, A fourth-order derivative-free algorithm for nonlinear equations, J. Comput. Appl. Math. 235 (2011) 2551–2559.
[16] L.D. Petković, M.S. Petković, A note on some recent methods for solving nonlinear equations, Appl. Math. Comput. 185 (2007) 368–374.
[17] M.S. Petković, On a general class of multipoint root-finding methods of high computational efficiency, SIAM J. Numer. Anal. 47 (2010) 4402–4414.
[18] M.S. Petković, L.D. Petković, Families of optimal multipoint methods for solving nonlinear equations: a survey, Appl. Anal. Discrete Math. 4 (2010) 1–22.
[19] M.S. Petković, L. Rancić, M.R. Milosević, On the new fourth-order methods for the simultaneous approximation of polynomial zeros, J. Comput. Appl. Math. 235 (2011) 4059–4075.
[20] H. Ren, Q. Wu, W. Bi, A class of two-step Steffensen type methods with fourth-order convergence, Appl. Math. Comput. 209 (2009) 206–210.
[21] J.R. Sharma, R.K. Guha, Second-derivative free methods of third and fourth order for solving nonlinear equations, Int. J. Comput. Math. 88 (2011) 163–170.
[22] X. Wang, L. Liu, Modified Ostrowski's method with eighth-order convergence and high efficiency index, Appl. Math. Lett. 23 (2010) 549–554.
[23] X.Y. Wu, D.S. Fu, New higher-order convergence iteration methods without employing derivatives for solving nonlinear equations, Comput. Math. Appl. 41 (2001) 489–495.
[24] B.I. Yun, A non-iterative method for solving non-linear equations, Appl. Math. Comput. 198 (2008) 691–699.
[25] B.I. Yun, M.S. Petković, Iterative methods based on the signum function approach for solving nonlinear equations, Numer. Algorithms 52 (2009) 649–662.
[26] C. Chun, B. Neta, A third-order modification of Newton's method for multiple roots, Appl. Math. Comput. 211 (2009) 474–479.
[27] C. Dong, A basic theorem of constructing an iterative formula of the higher order for computing multiple roots of an equation, Math. Numer. Sin. 11 (1982) 445–450.
[28] C. Dong, A family of multipoint iterative functions for finding multiple roots of equations, Int. J. Comput. Math. 21 (1987) 363–367.
[29] S. Li, L. Cheng, B. Neta, Some fourth-order nonlinear solvers with closed formulae for multiple roots, Comput. Math. Appl. 59 (2010) 126–135.
[30] B. Neta, New third order nonlinear solvers for multiple roots, Appl. Math. Comput. 202 (2008) 162–170.
[31] B. Neta, A.N. Johnson, High order nonlinear solvers for multiple roots, Comput. Math. Appl. 55 (2008) 2012–2017.
[32] B. Neta, Extension of Murakami's high order nonlinear solver to multiple roots, Int. J. Comput. Math. 87 (2010) 1023–1031.
[33] N. Osada, An optimal multiple root finding method of order three, J. Comput. Appl. Math. 51 (1994) 131–133.
[34] P.K. Parida, D.K. Gupta, An improved method for finding multiple roots and it's multiplicity of nonlinear equations in $\mathbb{R}$, Appl. Math. Comput. 202 (2008) 498–503.
[35] M.S. Petković, L.D. Petković, J. Dzunić, Accelerating generators of iterative methods for finding multiple roots of nonlinear equations, Comput. Math. Appl. 59 (2010) 2784–2793.
[36] J.R. Sharma, R. Sharma, Modified Jarratt method for computing multiple roots, Appl. Math. Comput. 217 (2010) 878–881.
[37] H.D. Victory, B. Neta, A higher order method for multiple zeros of nonlinear functions, Int. J. Comput. Math. 12 (1983) 329–335.
[38] B.I. Yun, Transformation methods for finding multiple roots of nonlinear equations, Appl. Math. Comput. 217 (2010) 599–606.
[39] B.I. Yun, New higher order methods for solving nonlinear equations with multiple roots, J. Comput. Appl. Math. 235 (2011) 1533–1555.
[40] P. Jarratt, Some efficient fourth order multipoint methods for solving equations, BIT 9 (1969) 119–124.
[41] R.F. King, A family of fourth order methods for nonlinear equations, SIAM J. Numer. Anal. 10 (1973) 876–879.
[42] B. Neta, Several new methods for solving equations, Int. J. Comput. Math 23 (1988) 265–282.
[43] A.M. Ostrowski, On approximation of equations by algebraic equations, SIAM J. Numer. Anal. 1 (1964) 104–130.
[44] A.M. Ostrowski, Solution of Equations and System of Equations, third ed., Academic Press, New York, 1973.
[45] J.F. Traub, Iterative Methods for the Solution of Equations, Prentice Hall, New York, 1964.