Adaptive Steffensen-like Methods with Memory for Solving Nonlinear Equations with the Highest Possible Efficiency Indices

M. J. Lalehchini *, T. Lotfi †‡, K. Mahdiani ‡

Received Date: 2018-11-14 Revised Date: 2019-05-06 Accepted Date: 2019-07-06

Abstract

The primary goal of this work is to introduce two adaptive Steffensen-like methods with memory of the highest efficiency indices. In the existing methods, to improve the convergence order applied to memory concept, the focus has only been on the current and previous iteration. However, it is possible to improve the accelerators, considering the time from the first to the current iterations. Therefore, we achieve superior convergence orders and obtain as high efficiency indices as possible. These are the main contributions of this work.

Keywords: Nonlinear equations; Iterative methods; Steffensen-like method; Methods with memory; Adaptive methods; R-order.

1 Introduction

One of the most important subjects in developing numerical algorithms is to establish optimal algorithms with economic complexity. For example, developing iterative methods for approximating zero(s) of a given nonlinear equation falls within this matter, and many studies have been devoted to it [9, 12]. Inspired by this, we will set up two adaptive Steffensen-like methods with memory which are improvement of the existing methods [1, 5, 6, 12, 16, 22]. To our knowledge, these kinds of adaptive methods have not been studied in the literature. Traub developed the first method with memory from Steffensen’s method [5] as following [11]:

\[
\begin{align*}
  w_k &= x_k + \gamma_k f(x_k), \\
  x_{k+1} &= x_k - \frac{f(x_k)}{f[x_k, w_k]}, \\
  \gamma_{k+1} &= -\frac{1}{N_1(x_{k+1})},
\end{align*}
\]

(1.1)

where \(x_0\) and \(\gamma_0\) are given initially suitable values, and \(N_1(t) = f(x_{k+1}) + (t - x_{k+1})f[x_{k+1}, x_k]\) is the linear Newton’s interpolation. The convergence order of the with-memory method (1.1) is \(1 + \sqrt{2} \approx 2.414\). Also, Džunić and Petković improved Traub’s idea, introducing a better accelerator [12]:

\[
\begin{align*}
  w_k &= x_k + \gamma_k f(x_k), \\
  x_{k+1} &= x_k - \frac{f(x_k)}{f[x_k, w_k]}, \\
  \gamma_{k+1} &= -\frac{1}{N_2(x_{k+1})},
\end{align*}
\]

(1.2)
where $x_0$ and $\gamma_0$ are given initially suitable values, and $N_2(t)$ is the Net-
won’s interpolation polynomial given by $N_2(t) = f(x_{k+1}) + (t-x_{k+1})f[x_{k+1}, w_k] +$
$(t-x_{k+1})(t-w_k)f[x_{k+1}, w_k, x_k].$
The convergence order of the method with
memory (1.2) is $1+\sqrt{3} = 3$. Moreover, Džumić
added another parameter to the Steffensen’s
method and obtained a more efficient method
with memory [9]:
$$\begin{aligned}
x_{k+1} &= x_k - \frac{f(x_k)}{f(x_k) + \lambda_k f(w_k)}, \\
\gamma_{k+1} &= -\frac{1}{N_2(x_{k+1})}, \quad k = 0, 1, 2, \cdots, \\
w_{k+1} &= x_{k+1} + \gamma_{k+1} f(x_{k+1}), \\
\lambda_{k+1} &= -\frac{N_2'(w_{k+1})}{x_{k+1} N_2'(w_{k+1})},
\end{aligned}$$
(1.3)
where $x_0$, $\gamma_0$, and $\lambda_0$ are given initially suitable
values. This method has the convergence order
$3+\sqrt{3} \approx 3.56$.

Remark 1.1 If $\gamma$ and $\lambda$ are constants, then
methods (1.2) and (1.3) are without memory
methods with convergence order two having the
following error equations, respectively:
$$\epsilon_{k+1} = c_2(1 + \gamma f'(\alpha))\epsilon_k^2 + O(\epsilon_k^3), \quad (1.4)$$
and
$$\epsilon_{k+1} = (c_2 + \lambda)(1 + \gamma f'(\alpha))\epsilon_k^2 + O(\epsilon_k^3). \quad (1.5)$$

In this work, we will attempt to carry out
two adaptive methods with memory regardless of
(1.2) and (1.3), which are superior [2, 14, 20, 24].
To achieve this end, first, the accelerator param-
eter $\gamma_k$ is updated with the existing information
in the previous and current iterations. We prove
that this method has convergence order 3.4 using
the same function evaluations as (1.2), so its
efficiency index is much better. Similarly, we
derive another adaptive method with memory for
(1.3) which acquires convergence order 3.9 using
the same functional evaluations. Therefore, this
method is better than both our adaptive method
with one accelerator and all the existing methods.

2 Developing adaptive with
memory methods

This section deals with two new adaptive meth-
ods with memory. To this end, we modify and ex-
tend methods (1.2) and (1.3) in such a way that
they consider all previous information to attain
as high as possible convergence order without any
new functional evaluation. In this manner, we use
the adaptive idea which has not been considered
to our best knowledge.

2.1 Mono accelerator adaptive with
memory method

In (1.2), to update accelerator $\gamma_k$ in each itera-
tion, we only use the information from the current
and previous iterations and reach the convergence
order 3. However, the procedure goes ahead, the
old information of current and previous steps can
be used. In another words, we tend to apply the
adaptive idea to construct the new methods with
memory. Accordingly, we introduce the following
new adaptive method with memory
$$\begin{aligned}
w_k &= x_k + \gamma_k f(x_k), \\
x_{k+1} &= x_k - \frac{f(x_k)}{f(x_k) + \lambda_k f(w_k)}, \\
\gamma_{k+1} &= -\frac{1}{N_{2k+2}(x_{k+1})}, \quad k = 0, 1, 2, \cdots, 
\end{aligned}$$
(2.6)
where $x_0$ and $\gamma_0$ are given initially suitable
values, and $N_{2k+2}(t)$ is Netwon’s interpolation
polynomial of degree $2k + 2$ at the points
$x_{k+1}, w_k, x_k, \ldots, w_0, x_0$.

Figure 1: Dynamical Planes for (2.6) for $\gamma = 0.1$.

Referring to the Error Equation (1.4), it is observed that if
$1 + \gamma f'(\alpha) = 0$, then convergence order of the
method without memory (2.6), increases if for a moment we supposed that $\gamma_k$ is fixed [22]. Since
$\alpha$ is unknown, we cannot suppose $\gamma = -1/f'(\alpha)$.
Table 1: Test functions for $\gamma = 0.1, \lambda = 0.1$.

<table>
<thead>
<tr>
<th>Example</th>
<th>$x_0$</th>
<th>$\alpha$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$f_1(t) = e(t^2 - 4) + \sin(t - 2) - t^4 + 15$</td>
<td>2.50</td>
<td>2.00</td>
</tr>
<tr>
<td>$f_2(t) = \frac{1}{t^2} - t^2 - \frac{1}{t} + 1$</td>
<td>2.00</td>
<td>1.00</td>
</tr>
<tr>
<td>$f_3(t) = (t - 2)(t^{10} + t + 2)e^{-5t}$</td>
<td>2.40</td>
<td>2.00</td>
</tr>
<tr>
<td>$f_4(t) = e(t^2 - 3t) + \sin(t) + \log(t^2 + 1)$</td>
<td>0.45</td>
<td>0.00</td>
</tr>
</tbody>
</table>

Table 2: Test functions for $\gamma = 0.1, \lambda = 0.1$.

| Function | $|x_1 - \alpha|$ | $|x_2 - \alpha|$ | $|x_3 - \alpha|$ | COC |
|----------|-----------------|-----------------|-----------------|-----|
| $f_1$    | 0.2371(0)       | 0.9503(-2)      | 0.1967(-7)      | 4.0683 |
| $f_2$    | 0.3417(-1)      | 0.9531(-3)      | 0.1021(-8)      | 3.8404 |
| $f_3$    | 0.2024(0)       | 0.2490(-2)      | 0.8749(-10)     | 3.9025 |
| $f_4$    | 0.2277(-1)      | 0.4799(-3)      | 0.3300(-9)      | 3.6537 |

Table 3: Results of (2.25) for different test functions.

| Function | $|x_1 - \alpha|$ | $|x_2 - \alpha|$ | $|x_3 - \alpha|$ | COC |
|----------|-----------------|-----------------|-----------------|-----|
| $f_1$    | 0.2902(0)       | 0.1028(-1)      | 0.1581(-7)      | 4.0078 |
| $f_2$    | 0.9073(-1)      | 0.2554(-2)      | 0.2200(-8)      | 3.9115 |
| $f_3$    | 0.1681(0)       | 0.1006(-2)      | 0.1582(-11)     | 3.9603 |
| $f_4$    | 0.4153(-1)      | 0.4718(-3)      | 0.5529(-11)     | 4.0784 |

Figure 2: Dynamical Planes for (2.6) for $\gamma = -0.31093$.

Figure 3: Dynamical Planes for (2.6) for $\gamma = -0.33330$.

Even if we assumed that $\alpha$ was known, we could not use it to evaluate $f'(\alpha)$, since it increased the functional evaluation, and optimality of the methods would be destroyed. It is assumed that the sequence $\{x_k\}$ converges to $\alpha$. Moreover, $f'$ is at least continuous, so there is $\lim f'(x_k) = f'(\alpha)$ as $k \to \infty$. Thus, we can use $N'_{2k+2}(x_k)$ instead of $f'(x_k)$ to our mission, i.e., $\gamma_k = -1/N'_{2k+2}(x_k)$.

To discuss the convergence order of (2.6), we need:

**Lemma 2.1** If $\gamma_{k+1} = -1/N'_{2k+2}(x_k)$, and
\[
\lambda_{k+1} = \frac{-N_{2k+3}'(w_{k+1})}{2N_{2k+3}(w_{k+1})}
\text{then}
\]

\[
1 + \gamma f'(\alpha) \sim \prod_{i=0}^{k} e_{w,i} e_i, \quad c_2 + \lambda \sim \prod_{i=0}^{k} e_{w,i} e_i,
\]

(2.7)

where \( e_i = x_i - \alpha \) and \( e_{w,i} = w_i - \alpha \).

**Proof.** According to Newton’s interpolation formula for nodes \( t_0, t_1, \ldots, t_s \), we have

\[
f(t) - N_s(t) = f^{(s+1)}(\xi) \prod_{i=0}^{s} (t - t_i).
\]

(2.8)

where \( s \in [\min \{t_0, t_1, \ldots, t_s\}, \max \{t_0, t_1, \ldots, t_s\}] \)

and

\[
\gamma = -\frac{1}{f'(\alpha)} \approx -\frac{1}{N_{2k+2}'(x_k)} = \gamma_{k+1}.
\]

(2.9)

By differentiating (2.8) and setting \( t = x_{k+1} \) like proving Lemma 1 in [12], we have

\[
N_{2k+2}'(x_{k+1}) = f'(x_{k+1}) - \frac{f^{(2k+3)}(\xi)}{(2k+3)!} \prod_{i=0}^{k} (x_{k+1} - x_i)(x_{k+1} - w_i)
\]

(2.10)

\[
\sim f'(\alpha)(1 + c_{2k+3} \prod_{i=0}^{k} e_{w,i} e_i).
\]

Consequently,

\[
1 + \gamma_{k+1} f'(\alpha) = 1 - \frac{f'(\alpha)}{N_{2k+2}'(x_{k+1})} = \frac{1}{1 - c_{2k+3} \prod_{i=0}^{k} e_{w,i} e_i} \sim \prod_{i=0}^{k} e_{w,i} e_i.
\]

(2.11)
To prove the second relation with respect to (2.8), (2.10) and Lemma 1 in [9] we have

\[
\begin{align*}
\epsilon_2 + \lambda_{k+1} & = \frac{f''(\alpha)}{2f'(\alpha)} - \frac{N_{2k+2}(w_{k+1})}{2N_{2k+3}(w_{k+1})} \\
& \approx \prod_{i=0}^{k} \epsilon_{w,i} \epsilon_i
\end{align*}
\] (2.12)

**Theorem 2.1** Let the initial approximation \(x_0\) be sufficiently close to the zero \(\alpha\) of \(f\). Also \(R\) and \(p\) denote the convergence order of the sequences \(\{x_k\}\) and \(\{w_k\}\), respectively, as obtained in adaptive method with memory (2.6). Then, we will have

\[
\begin{align*}
R^k p - R^k - (p + 1) \sum_{i=0}^{k-1} R^i & = 0, \\
R^{k+1} - R^k - (p + 1) \sum_{i=0}^{k-1} R^i & = 0.
\end{align*}
\] (2.13)

**Proof.** We can assume

\[
e_{k+1} \sim e_k^R.
\] (2.14)

Hence,

\[
e_{k+1} \sim (e_{k-1}^R)^R = e_{k-1}^{R^2}.
\] (2.15)

Inductively,

\[
e_{k+1} \sim e_0^{R^{k+1}}.
\] (2.16)

Similarly, we have

\[
e_{w,k} \sim e_k^p = (e_{k-1}^R)^p = e_{k-1}^{R^p}.
\] (2.17)

Thus,

\[
e_{w,k} \sim e_0^{R^k p}.
\] (2.18)

By (2.14) and (2.17), Lemma 2.1 results in

\[
1 + \gamma f'(\alpha) \sim e_0^{(p+1) \sum_{i=0}^{k-1} R^i}.
\] (2.19)
and with-memory method:

consider the following two new accelerators adap-

2.2 Bi accelerators adaptive with memory method

Figure 12: Dynamical Planes for (2.25) for $\gamma = -0.25005, \lambda = -1.49993$

On the other hand, since $e_{w,k} \sim (1 + \gamma f'(\alpha))e_k$
and $e_{k+1} \sim (1 + \gamma f'(\alpha))e_k^2$, taking into account (2.19), we have

$$e_{w,k} \sim e_k^R + (p+1) \sum_{i=0}^{k-1} R^i,$$  \hspace{1cm} (2.20)

and

$$e_{k+1} \sim e_k^{2R} + (p+1) \sum_{i=0}^{k-1} R^i.$$  \hspace{1cm} (2.21)

From (2.18)-(2.20) and (2.16)-(2.21), we conclude that

$$e_k^R \sim e_k^0 + \sum_{i=0}^{k-1} R^i,$$  \hspace{1cm} (2.22)

$$e_k^{R+1} \sim e_k^{0} + 2R + (p+1) \sum_{i=0}^{k-1} R^i.$$  \hspace{1cm} (2.23)

Consequently,

$$\begin{cases} R^k p - R^k - (p+1) \sum_{i=0}^{k-1} R^i = 0, \\ R^{k+1} - 2R^k - (p+1) \sum_{i=0}^{k-1} R^i = 0. \end{cases}$$  \hspace{1cm} (2.24)

2.2 Bi accelerators adaptive with memory method

We now introduce bi accelerators adaptive with memory method [7, 14, 19, 20, 21, 24]. Since most of the details are similar to the descriptions of (2.6), we confine ourselves to repeat them. We consider the following two new accelerators adaptive with-memory method:

$$\begin{cases} x_{k+1} = x_k - \frac{f(x_k)}{f_x(x_k,w_k) + \lambda_k f(w_k)}, \\ \gamma_{k+1} = \frac{1}{N_{2k+2}(x_{k+1})}, \quad k = 0, 1, 2, \ldots, \\ w_{k+1} = x_{k+1} + \gamma_{k+1} f(x_{k+1}), \\ \lambda_{k+1} = \frac{1}{2N_{2k+3}(w_{k+1})}, \end{cases}$$  \hspace{1cm} (2.25)

where $x_0$, $\gamma_0$ and $\lambda_0$ are given suitably. Then, we have

**Theorem 2.2** Let the initial approximation $x_0$ be sufficiently close to the zero $\alpha$ of $f$. Also $R$ and $p$ denote the convergence order of the sequences $\{x_k\}$ and $\{w_k\}$, respectively, as obtained in adaptive method with memory (2.25). Then, we will have

$$\begin{cases} R^k p - R^k - (p+1) \sum_{i=0}^{k-1} R^i = 0, \\ R^{k+1} - 2R^k - 2(p+1) \sum_{i=0}^{k-1} R^i = 0. \end{cases}$$  \hspace{1cm} (2.26)

**Proof.** We can assume

$$e_{k+1} \sim e_k^R.$$  \hspace{1cm} (2.27)

Hence,

$$e_{k+1} \sim (e_{k-1}^R)^R = e_{k-1}^{R^2}.$$  \hspace{1cm} (2.28)

Inductively,

$$e_{k+1} \sim e_{k-1}^{R^{k+1}}.$$  \hspace{1cm} (2.29)

Similarly, we have

$$e_{w,k} \sim e_k^p = (e_k^{R-1})^p = e_k^{R^p}.$$  \hspace{1cm} (2.30)

Thus,

$$e_{w,k} \sim e_k^{R^p}.$$  \hspace{1cm} (2.31)

By (2.14) and (2.17), Lemma 2.1 results

$$(e_2 + \lambda)(1 + \gamma f'(\alpha)) \sim e_0^2(p+1) \sum_{i=0}^{k-1} R^i.$$  \hspace{1cm} (2.32)

By (1.5) and $e_{w,k} \sim (1 + \gamma f'(\alpha))e_k$, taking into account (2.32), we have

$$e_{w,k} \sim e_0^R + (p+1) \sum_{i=0}^{k-1} R^i.$$  \hspace{1cm} (2.33)

and

$$e_{k+1} \sim e_0^{2R + 2(p+1) \sum_{i=0}^{k-1} R^i}.$$  \hspace{1cm} (2.34)

From (2.31)-(2.33) and (2.29)-(2.34), we conclude that

$$e_0^R \sim e^R + (p+1) \sum_{i=0}^{k-1} R^i.$$  \hspace{1cm} (2.35)

$$e_0^{R+1} \sim e_0^{2R + 2(p+1) \sum_{i=0}^{k-1} R^i}.$$  \hspace{1cm} (2.36)

Consequently,

$$\begin{cases} R^k p - R^k - (p+1) \sum_{i=0}^{k-1} R^i = 0, \\ R^{k+1} - 2R^k - 2(p+1) \sum_{i=0}^{k-1} R^i = 0. \end{cases}$$  \hspace{1cm} (2.37)
3 Numerical Computations

In this section, to show the efficiency of the new adaptive methods with memory (2.6) and (2.25), we report their numerical results. To this end, among many tested problems, we confine to report the results of four test functions (See Table 1). Moreover, the initial values are given in Tables 1. It should be noted that $|x_k - a|$ shows the error in each iteration and $a(b)$ stands for $a^b$, and the computational order of convergence (COC) can be approximated by the following formula:

$$\text{COC} \approx \frac{\log |x_{k+1} - a|}{\log |x_k - a|}.$$

Tables 2 and 3 report the numerical implementations of the adaptive methods with memory in this work. Table 2 shows that the numerical results support the developed theory for method (2.6). As can be seen in Table 3, the (COC) of the tested functions using the method (2.25) is very good, and has the highest amount. Indeed, there is not any work in the literature that could compete with method (2.25). On the other hand, this is the highest and the best efficiency index which comes from the point of the efficiency index. Let us discuss this contribution a little more. It is well known that any general optimal multipoint method without memory, using $n + 1$ functional evaluations has optimal convergence order $2^n$, so it could reach the optimal efficiency index $E^* = \lim_{n \to \infty} 2^{\frac{\log n}{n}} = 2$. On the other hand, we have proved that the new adaptive method with memory (2.25) could reach the same efficiency index with only two functional evaluations (See Theorem 2.2). This means that this method competes with any optimal multipoint method without memory.

In what follows, we disclose the mathematica code for the numerical implementations.

4 Dynamical Behavior

Here we focus on the stability behavior of the adaptive methods with memory (2.6) and (2.25). To this end, we utilize visual dynamical approach [13, 15, 17, 19, 24]. Although we have tested many examples, we have reached the same conclusion. Therefore, we only report the results for the function $p(z) = z^3 + 1$ and $p(z) = z^4 - 1$. To show the stability of the one-parameter (2.6) and two-parameter (2.25) methods, we analyze their dynamical properties and focus on the of dynamic planes related to iterative methods. Mathematica software can be used to do the analysis in which the graphic planes are shown in a rectangle of $[-3, 3] \times [-3, 3]$ dimension along with a 72-pixel resolution. The dynamic planes illustrate the absorption area for polynomials $p_1(z) = z^3 + 1$ and $p_2(z) = z^4 - 1$. $p_1(z)$ and $p_2(z)$ have the solution of $-1, \frac{1}{2}(1 + i\sqrt{3}), \frac{1}{2}(1 - i\sqrt{3})$ and $-1, 1, -i, i$, respectively. Each solution is allocated a color. The greater the dark areas, the greater the intensity of unstability. In Figures 1, 2 and 3, the stability of single parameter methods (2.6) is shown for $\gamma = 0.1, -0.03093$ and $-0.3333$, in which $p_1(z)$ has been used. Also, in Figures 4, 5 and 6, the stability of single parameter method (2.6) is shown for $\gamma = 0.1, -0.31336$ and $-0.3333$ in which $p_2(z)$ is used. Comparing the figures shows that the stability has been reduced. For the two-parameter method (2.25), in Figure 7, the absorption area and stability are shown for $\gamma$ and $\lambda$ at 0.1 and 0.1 respectively, in Figure 8 for $\gamma = -0.31336$ and $\lambda = 0.99909$ and in Figure 9 for $\gamma = -0.33334$ and $\lambda = 1.00000$, where $p_2(z)$ is used. Finally, the two-parameter iterative method (2.25) has been taken to compare absorption areas, using $p_2(z)$ in Figures 10, 11 and 12, for $\gamma$ and $\lambda$ at 0.1 and 0.1 in Figure 10, -0.21210 and -1.46997 in Figure (11) and -0.25005, -1.49997 in Figure 12, where $p_2(z)$ is used. Comparing the figures, we witness there is a reduction in stability showing that, despite the increase of convergence order in the two-parameter method (2.25), the stability trend is satisfactory. Both developed methods with memory (2.6) and (2.25) show the same instable behavior. Consequently, though developing methods with memory has the advantage from the view of computational complexity, they represent numerical chaos and numerical instability.

5 Conclusion

In this work, we developed two new but very efficient methods with memory to solve a nonlinear equation. We have shown that the methods
could reach the highest possible efficiency indices and can compete with any method with or without memory in the literature. Convergence analysis of the developed methods has been presented, and we have tested some numerical examples to show the practicality of the proposed methods. Though both developed methods have the highest possible efficiency as opposed to any other methods in the literature, we have seen that methods with memory show instability in practice. Therefore, they benefit from computational efficiency and suffer from numerical stability. Finally, we end the conclusion with the following research question: how can a method with memory be developed to show the numerical stability behavior?

References


Mohammad Javad Lalehchini is a PhD candidate in Department of Applied Mathematics, Hamedan Branch, Islamic Azad University, Hamedan, Iran. Her research areas in Numerical analysis is include approximating numerical solutions using iterative methods for nonlinear systems of equations.

Taher Lotfi has got MSc degrees in Applied Mathematics from Kharazmi (Tarbiat Moallem) University, and PhD degree from Science and Research Branch, Islamic Azad University, Tehran, Iran. My main research interest include approximating numerical solutions using iterative methods for nonlinear systems of equations, specially for large systems arising from IE, ODE and PDE. Also, interval analysis, reproducing kernel space methods, soft computing based on wavelets and fuzzy concepts, generalized inverses are my current MSc and PhD students research field.

Katayoun Mahdiani has got MSc and PhD degrees in Applied Mathematics from Islamic Azad University, Karaj, Iran. My main research interest include approximating numerical solutions using iterative methods for nonlinear systems of equations, specially for large systems arising from integral equations.