Int. J. Nonlinear Anal. Appl. 12 (2021) No. 1, 301-316
ISSN: 2008-6822 (electronic)
http://dx.doi.org/10.22075/ijnaa.2021.4799

On the efficient of adaptive methods to solve nonlinear equations

Vali Torkashvand[*,a,b], Reza Ezzati[c]

[a] Young Researchers and Elite Club, Shahr-e-Qods Branch, Islamic Azad University Tehran Iran
[b] Farhangian University, Tehran, Iran
[c] Department of Mathematics, Karaj Branch, Islamic Azad University, Karaj, Iran

(Communicated by Madjid Eshaghi Gordji)

Abstract

The main goal of this work, obtaining a family of Steffensen-type iterative methods adaptive with memory for solving nonlinear equations, which uses three self-accelerating parameters. For this aim, we present a new scheme to construct the self-accelerating parameters and obtain a family of Steffensen-type iterative methods with memory. The self-accelerating parameters have the properties of simple structure and easy calculation, which do not increase the computational cost of the iterative methods. The convergence order of the new iterative methods has increased from 4 to 8. Also, these methods possess very high computational efficiency. Another advantage of the new method is that they remove the severe condition $f'(x)$ in a neighborhood of the required root imposed on Newton's method. Numerical comparisons have made to show the performance of the proposed methods, as shown in the illustrative examples.

Keywords: Nonlinear equations, Newton's interpolatory polynomial, Adaptive method with memory, The order of convergence, Self accelerating parameter.
2010 MSC: 41A25, 65H05, 65B99.

1. Introduction

Solving nonlinear equations is a classical problem that has interesting applications in various branches of science and engineering. To solve nonlinear equations, iterative methods such as Newton's method are usually used. Throughout this paper, we consider iterative methods to find a simple root ξ, i.e., $f(\xi) = 0$ and $f'(\xi) \neq 0$, of a nonlinear equation $f(x) = 0$, where $f : I \subset \mathbb{R} \longrightarrow \mathbb{R}$ for an open

*Corresponding author
Email addresses: torkashvand1978@gmail.com (Vali Torkashvand*,), ezati@kiau.ac.ir (Reza Ezzati)

Received: 18 March 2019 *Revised:* 14 January 2020

interval I. Newton's method(NM) for the calculation of ξ is probably the most widely used iterative scheme defined by

$$x_{k+1} = x_k - \frac{f(x_k)}{f'(x_k)},\ k = 0, 1, 2, \cdots . \tag{1.1}$$

This well-known method is quadratically convergent to compute simple roots [26]. This method is not applicable when the derivative of any function has been defined. Therefore, Steffensen modified Newton's method. He replaced the first derivative $f'(x_n)$ with the forward difference approximation.

$$f'(x_k) = \frac{f(x_k + \beta f(x_k)) - f(x_k)}{\beta f(x_k)}$$

and can obtain the famous Steffensen's method [41]:

$$x_{k+1} = x_k - \frac{\beta f(x_k)^2}{f(x_k + \beta f(x_k)) - f(x_k)},\ k = 0, 1, 2, \cdots , \tag{1.2}$$

where the parameter β to be freely chosen in $\mathbb{R} - \{0\}$ and used to generate a class of Steffensen's methods provided that the denominator is not equal to zero. Newton's and Steffensen's methods are of second-order convergence require two function evaluations per step, but in contrast to Newton's method, Steffensen's method is free from the derivative of function because sometimes the applications of the iterative methods which depend upon derivatives are restricted in engineering. These are two sample schemes of a one-point iteration, i. e. in each iteration step of the evaluations have taken at one point. Multiple-point methods evaluate at several points in each iteration step, and principle allows for a higher convergence order with a lower number of function evaluations. Kung and Traub [19] conjectured that multi-point optimal method without memory with k evaluations could have a convergence order larger than 2^{k-1}. For well-known two-point without-memory methods, one can consult e.g. Jarrat [16], King [17] Ostrowski [27], and Maheshwari[24]. Soleymani et al.[36, 39] developed an optimal three-point iterative method with convergence order 8. Sharma-Arora.[30] used weight functions to construct optimal three-point methods and optimal convergence order eight. Geum and Kim [14] and Sharifi et al.[34]utilizing parametric weight functions. The efficiency index sees [27] gives a measure of the balance between those quantities, according to the formula $p^{1/d}$, where p is the order of convergence of the method and d the number of functional evaluations per step. Some of the people who have worked on increasing the efficiency index of numerical methods for solving nonlinear equations after the Traub (which is leading in with memory methods) and high-efficiency indexing methods are Cordero et al.[7, 8], Dzunic et al.[11, 12], Lotfi et al.[21, 22, 23], Petkovic et al.[28, 29], Soleymani et al.[38, 39], Wang et al.[46, 47, 48]. This paper aims to state a two-point family adaptive with the memory of very high computational efficiency. We start from a family of two-point methods without memory with order 4, derived in [22], and increase the convergence order to $6, 7, 7.22, 7.53, 7.77$ and 8 (depending on the accelerating technique) without additional calculations. In this manner, we have obtained new methods for finding simple roots of nonlinear equations. Computational efficiency is higher than the efficiency of existing methods known in literature in the class of two-point methods and even higher than the efficiency of optimal three-, four-, and five-point methods of optimal order. The main idea has based on the use of suitable two-valued functions and the variation of three free parameters in each iterative step. These parameters have calculated using information from the current and previous iteration so that the developed methods can regard as methods with memory following Traub's classification [45]. An additional motivation for studying adaptive methods with memory arises from a surprising fact that such classes of the methods have been considered in literature very seldom despite their high computational efficiency. If one can increase the order of convergence in a without memory method by reusing the old information, he/she

develops it as a with-memory method. The adaptive with-memory methods reuse the information from all previous steps. Our motivated focus on this problem. Therefore, in this work, we have developed with-memory methods; i.e., that uses the information not only from the last two-steps but also from all previous iterations. The adaptive technique enables theoretically and practically us to achieve the highest efficiency. Indeed, we will develop the adaptive-method with-memory has efficiency index 2, hence, competes with all the existing methods without and with memory in the literature.
This paper has organized as follows: Section 2 deals with modifying the optimal two-points methods with memory introduced by Lotfi et al.[22]. In section 3, the aim of this work has been presented by contributing iterative-adaptive with memory method for solving nonlinear equations, improved order of convergence from 4 to 8 without adding more evaluations, has presented, and has achieved in the maximum performance index. It means that, without any new function calculations, we can improve convergence order by 100%. The numerical study presented in section 4 confirms the theoretical results and the excellent convergence properties of the presented methods in comparison with some optimal iterative methods, without memory and with memory. To show applicability, and competitive of the developed-methods some have tested nonlinear equations have solved.

2. Without memory methods

In this section, we will discuss the convergence analysis of without-memory methods that can build with memory methods in section 3. In 2015, Lotfi et al. proposed a two-step method as following: (LSGAM4) [22]

$$\begin{cases} w_k = x_k + \gamma f(x_k), B_k = \frac{f[x_k,w_k]}{f[y_k,w_k]}, k \in \mathbb{W} \\ y_k = x_k - \frac{f(x_k)}{f[x_k,w_k]}(1 + q\frac{f(w_k)}{f[x_k,w_k]}), \\ x_{k+1} = y_k - \frac{f(y_k)}{f[y_k,x_k]+\lambda(y_k-x_k)(y_k-w_k)}(B_k + (B_k-1)^4), \end{cases} \tag{2.1}$$

where γ, λ and q are arbitrary nonzero real parameters, and $f[x,y] = \frac{f(x)-f(y)}{x-y}$ stands for the divided difference of the first order. This method is an optimal-order without-memory method. In other words, it uses three function evaluations per iteration that it has optimal convergence order 4. The error equation of the method (2.1) is:

$$e_{k+1} = \frac{(1+\gamma f'(\alpha))^2(q-c_2)(-\lambda + f'(\alpha)(q-2c_2)c_2 + c_3)}{f'(\alpha)} e_k^4 + O(e_k^5). \tag{2.2}$$

The next theorem states of the error equation of the method (2.1).

Theorem 2.1. *Let $I \subseteq \mathbb{R}$ be an open interval, $f : I \to \mathbb{R}$ be a differentiable function, and has a simple zero, say α. If x_0 is an initial guess to α, then the error equation of the method (2.1) is given by*

$$e_{k+1} = \frac{(1+\gamma f'(\alpha))^2(q-c_2)(-\lambda + f'(\alpha)(q-2c_2)c_2 + c_3)}{f'(\alpha)} e_k^4 + O(e_k^5), \tag{2.3}$$

Proof . Let $e_k = x_k - \alpha$, $\tilde{e}_k = w_k - \alpha$, $\hat{e}_k = y_k - \alpha$, and $e_{k+1} = x_{k+1} - \alpha$. Denote $c_k = \frac{f^{(k)}(\alpha)}{k!f'(\alpha)}$ for $k = 2, 3, \cdots$. Using Taylor expansion and taking into account $f(\alpha) = 0$, we have:

$$f(x_k) = f'(\alpha)(e_k + c_2e_k^2 + c_3e_k^3 + c_4e_k^4 + O(e_k^5). \tag{2.4}$$

Then, computing $w_k = x_k + \gamma f(x_k)$, we attain

$$\tilde{e}_k = e_k + e_k\gamma f'(\alpha)(1 + e_k(c_2 + e_k(c_3 + e_kc_4))) + O(e_k^5). \tag{2.5}$$

Considering $f[x,y]=\frac{f(x)-f(y)}{x-y}$ is Newton's first order divided difference. we get

$$\begin{aligned} f[x_k,w_k]=&-1/(e_k\gamma f'(\alpha)(1+e_k(c_2+e_k(c_3+e_kc_4)))))^{-1}(e_kf'(\alpha)(1+e_k(c_2\\ &+e_k(c_3+e_kc_4)))-f'(\alpha)(e_k+e_k\gamma f'(\alpha)(1+e_k(c_2+e_k(c_3+e_kc_4)))\\ &+c_2(1+e_k(c_2+e_k(c_3+e_kc_4))))^2++c_3(1+e_k(c_2+e_k(c_3+e_kc_4))))^3\\ &+c_4(1+e_k(c_2+e_k(c_3+e_kc_4))))^4)) \end{aligned} \tag{2.6}$$

By a simple calculation, we get:

$$\begin{aligned} \frac{f(w_k)}{f[x_k,w_k]}=&-((e_kf'(\alpha)^2\gamma(1+e_k(c_2+e_k(c_3+e_kc_4)))(e_k+e_k\gamma f'(\alpha)(1+e_k(c_2\\ &+e_k(c_3+e_kc_4)))+c_2(e_k+e_k\gamma f'(\alpha)(1+e_k(c_2+e_k(c_3+e_kc_4))))^2\\ &+c_3(e_k+e_k\gamma f'(\alpha)(1+e_k(c_2+e_k(c_3+e_kc_4))))^3+c_4(e_k+e_k\gamma f'(\alpha)\\ &.(1+e_k(c_2+e_k(c_3+e_kc_4))))^4))/(e_kf'(\alpha)(1+e_k(c_2+e_k(c_3+e_kc_4)))\\ &-f'(\alpha)(e_k+e_k\gamma f'(\alpha)(1+e_k(c_2+e_k(c_3+e_kc_4)))+c_2(e_k+e_k\gamma f'(\alpha)\\ &.(1+e_k(c_2+e_k(c_3+e_kc_4))))^2+c_3(e_k+e_k\gamma f'(\alpha)(1+e_k(c_2+e_k(c_3\\ &+e_kc_4))))^3+e_kc_4))))^3+c_4(e_k+e_k\gamma f'(\alpha)(1+e_k(c_2+e_k(c_3+e_kc_4))))^4))) \end{aligned} \tag{2.7}$$

and

$$\begin{aligned} \frac{f(x_k)}{f[x_k,w_k]}=&-((e_k^2f'(\alpha)^2\gamma(1+e_k(c_2+e_k(c_3+e_kc_4)))^2)/(e_kf'(\alpha)(1+e_k(c_2\\ &+e_k(c_3+e_kc_4)))-f'(\alpha)(e_k+e_k\gamma f'(\alpha)(1+e_k(c_2+e_k(c_3+e_kc_4)))\\ &+c_2(e_k+e_k\gamma f'(\alpha)(1+e_k(c_2+e_k(c_3+e_kc_4))))^2+c_3(e_k+e_k\gamma f'(\alpha)\\ &(1+e_k(c_2+e_k(c_3+e_kc_4))))^3+e_kc_4))))^3+c_4(e_k+e_k\gamma f'(\alpha)(1+e_k\\ &(c_2+e_k(c_3+e_kc_4))))^4))). \end{aligned} \tag{2.8}$$

By substituting (2.7) and (2.8) into (2.1), we obtain

$$\begin{aligned} y_k=&\alpha+(1+\gamma f'(\alpha))(q-c_2)e_k^2+((2+\gamma f'(\alpha)(2+\gamma f'(\alpha))(q-c_2)c_2+(1+\gamma f'(\alpha))\\ &(2+\gamma f'(\alpha))c_3e_k^3+(-q(5+\gamma f'(\alpha)(7+5\gamma f'(\alpha)(4+\gamma f'(\alpha))))c_2^2+(4+\gamma f'(\alpha)\\ &(3+\gamma f'(\alpha))))c_2^3+(q4+\gamma f'(\alpha)(7+\gamma f'(\alpha)(5+\gamma f'(\alpha))))c_3-(7+\gamma f'(\alpha)(10+\\ &\gamma f'(\alpha)(7+2\gamma f'(\alpha))))c_2c_3+(1+\gamma f'(\alpha))(3+\gamma f'(\alpha)(3+\gamma f'(\alpha)))c_4e_k^4+O(e_k^5). \end{aligned} \tag{2.9}$$

Using (2.1) and (2.9) we conclude that

$$\begin{aligned} f[y_k,x_k]=&f'(\alpha)+f'(\alpha)c_2e_k+f'(\alpha)(-(1+\gamma f'(\alpha)(q-c_2)(c_2+c_3)e_k^2+f'(\alpha)((2\\ &+\gamma f'(\alpha)(2+\gamma f'(\alpha)))c_2^2+(1+\gamma f'(\alpha))(q+(3+\gamma f'(\alpha))c_2)c_3+c_4)e_k^3\\ &+f'(\alpha)(-q(5+\gamma f'(\alpha)(7+\gamma f'(\alpha)(4+\gamma f'(\alpha))))c_2^3+(4+\gamma)(5+\gamma f'(\alpha)\\ &(3+\gamma f'(\alpha))))c_2^4-(2+\gamma f'(\alpha))(4+\gamma f'(\alpha)(3+2\gamma f'(\alpha)))c_2^2c_3+(1+\gamma f'(\alpha))\\ &(q^2(1+\gamma f'(\alpha))c_3+(2+\gamma f'(\alpha))c_3^2-qc_4)+c_2(q(4+\gamma f'(\alpha)(5+\gamma f'(\alpha)(4\\ &+\gamma f'(\alpha))))c_3+(1+\gamma f'(\alpha))(4+\gamma f'(\alpha)(3+\gamma f'(\alpha)))c_4))e_k^4+O(e_k^5) \end{aligned} \tag{2.10}$$

Using (2.5) and (2.9) we can get

$$\begin{aligned} f[y_k, w_k] =& f'(\alpha) + f'(\alpha)(1+\gamma f'(\alpha)e_k(c_2 + f'(\alpha)(-(q - q\gamma f'(\alpha))c_2 + (1+2\gamma f'(\alpha))c_2^2 \\ & + (1+\gamma f'(\alpha))^2 c_3)e_k^2 + f'(\alpha)(q(2+\gamma f'(\alpha)(2+\gamma f'(\alpha)))c_2^2 - (2+\gamma f'(\alpha) \\ & (2+\gamma f'(\alpha)))c_2^3 + (1+2\gamma f'(\alpha))(3+2\gamma f'(\alpha))c_2c_3(1+\gamma f'(\alpha))^2(-qc_3 + (1 \\ & + \gamma f'(\alpha))c_4))e_k^3 + f'(\alpha)(-q(5+\gamma f'(\alpha)(7+\gamma f'(\alpha)(4+\gamma f'(\alpha))))c_2^3 + (4 \\ & + \gamma f'(\alpha)(5+\gamma f'(\alpha)(3+\gamma f'(\alpha))))c_2^4 + (q + q\gamma f'(\alpha))^2 c_3 - (8+\gamma f'(\alpha) \\ & (11+\gamma f'(\alpha)(7+3\gamma f'(\alpha))))c_2^2 c_3 + (1+\gamma f'(\alpha))(2+\gamma f'(\alpha)(5+\gamma f'(\alpha)))c_3^2 \\ & - q(1+\gamma f'(\alpha))^3 c_4 + c_2(q(4+\gamma f'(\alpha)(6+\gamma f'(\alpha)(5+2\gamma f'(\alpha))))c_3 + (4 \\ & + \gamma f'(\alpha)(13\gamma f'(\alpha)(13+5\gamma f'(\alpha))))c_4))e_k^4 + O(e_k^5) \end{aligned} \tag{2.11}$$

By dividing the relation (2.6) to (2.10) it follows

$$\begin{aligned} B =& \frac{f[x_k, w_k]}{f[y_k, w_k]} = 1 + c_2 e_k + ((1+\gamma f'(\alpha))(q - 2c_2)c_2 + (2+\gamma f'(\alpha))c_3)e_k^2 + (-q \\ & (2+\gamma f'(\alpha)(3+\gamma f'(\alpha)))c_2^2 + (3+\gamma f'(\alpha)(4+3+\gamma f'(\alpha))c_2^3 + q(1+\gamma f'(\alpha))^2 c_3 \\ & - (6+\gamma f'(\alpha)(9+4\gamma f'(\alpha))c_2c_3(3+\gamma f'(\alpha)(3+\gamma f'(\alpha)))c_4 e_k^3 + (q(1+\gamma f'(\alpha) \\ & (2+\gamma f'(\alpha)(1+3\gamma f'(\alpha))c_2^3 + (1+\gamma f'(\alpha))(3+\gamma f'(\alpha))(1+4\gamma f'(\alpha))c_2^4 - (q \\ & + q\gamma f'(\alpha))^2 c_3 - (4+\gamma f'(\alpha)(9+\gamma f'(\alpha)(4+\gamma f'(\alpha))c_3^2 + c_2^2((q + q\gamma f'(\alpha))^2 \\ & (11+\gamma f'(\alpha)(19+3\gamma f'(\alpha)(7+3\gamma f'(\alpha))))c_3 + q(1+\gamma f'(\alpha))^3 c_4 - c_2(q(3 \\ & + \gamma f'(\alpha)(7+\gamma f'(\alpha)(9+4\gamma f'(\alpha))))c_3 + (8+\gamma f'(\alpha)(15+4\gamma f'(\alpha) \\ & (3+\gamma f'(\alpha))))c_4)) + e_k^4 + O(e_k^5). \end{aligned} \tag{2.12}$$

By substituting $(2.9), (2.5)$ and (2.10) into (2.1), we find

$$\begin{aligned} & \frac{f(y_k)}{f[x_k, y_k] + \lambda(y_k - x_k)(y_k - w_k)} = -(1+\gamma f'(\alpha))(q - c_2)e_k^2 + ((3+\gamma f'(\alpha)(3+\gamma \\ & f'(\alpha)))(q - c_2)c_2 + (1+\gamma f'(\alpha))(2+\gamma f'(\alpha))c_3)e_k^3 + (f'(\alpha))^{-1}(-qf'(\alpha)(2+\gamma f'(\alpha)) \\ & (4+\gamma)f'(\alpha)(3+\gamma f'(\alpha)))c_2^2 + f'(\alpha)(7+\gamma f'(\alpha)(8+\gamma f'(\alpha)(4+\gamma f'(\alpha))))c_2^3 + qf'(\alpha) \\ & (5+\gamma f'(\alpha)(8+\gamma f'(\alpha)(5+\gamma f'(\alpha))))c_3c_2(-\lambda(1+\gamma f'(\alpha))^2 - 2f'(\alpha)(5+\gamma f'(\alpha) \\ & (7+\gamma f'(\alpha)(4+\gamma f'(\alpha)))c_3) + (1+\gamma f'(\alpha))(q\lambda + (1+\gamma f'(\alpha)) + f'(\alpha)(3+\gamma f'(\alpha) \\ & (3+\gamma f'(\alpha)))c_4)) + e_k^4 + O(e_k^5). \end{aligned} \tag{2.13}$$

Substituting equations (2.4)-(2.13) into equation (2.1), we obtain

$$e_{k+1} = \frac{(1+\gamma f'(\alpha))^2(q - c_2)(-\lambda + f'(\alpha)(q - 2c_2)c_2 + c_3)}{f'(\alpha)} e_k^4 + O(e_k^5). \tag{2.14}$$

This reveals that the proposed scheme (2.1) reaches fourth-order convergence. □

3. Family of two-point methods with memory

By considering (2.3) it is clear that there are some possibilities to vanish the coefficient of e_k^4. For example, if $(1+\gamma f'(\alpha))=0$, $(q-c_2)=0$, or $(-\lambda+f'(\alpha)((q-2c_2)c_2+c_3))=0$, then the coeffcient of e_k^4 vanishes at once. We propose Steffensen-type methods with memory as follows :

1. *The new with-memory methods order 6.*(TEM6)
If $(1+\gamma f'(\alpha))=0$, it can be seen that this relation leads to $\gamma=\frac{-1}{f'(\alpha)}$, since α is unknown, it is impossible to compute $f'(\alpha)$. If we assume that α is known, computing $f'(\alpha)$ has been not suggested since it increases these function evaluations. Fortunately, during the iterative process (2.1), finder approximations to α are generated by the sequence x_k, and therefore we try to obtain a good approximate for $f'(\alpha)$. Each iteration, $x_k,\ w_k,\ y_k$, and x_{k+1}, are accessible, except at the initial step. Hence, we can interpolate $f'(\alpha)$ using these nodes. Now, we consider Newton interpolating polynomial as follows:

$$\begin{cases} N_3'(x_k)=[\frac{d}{dt}N_3(t;x_{k-1},w_{k-1},y_{k-1},x_k)]_{t=x_k}=[\frac{d}{dt}(f(x_k)+f[x_k,y_{k-1}] \\ .(t-x_k)+f[x_k,y_{k-1},w_{k-1}](t-x_k)(t-y_{k-1})+f[x_k,y_{k-1},w_{k-1},x_{k-1}] \\ .(t-x_k)(t-y_{k-1})(t-w_{k-1})]_{t=x_k}=f[x_k,y_{k-1}]+f[x_k,y_{k-1},w_{k-1}] \\ .(x_k-y_{k-1})+f[x_k,y_{k-1},w_{k-1},x_{k-1}](x_k-y_{k-1})(x_k-w_{k-1}). \end{cases} \tag{3.1}$$

If $\gamma=\frac{-1}{f'(\alpha)}=\frac{-1}{N_3'(x_k)}$, we obtain the following with-memory method order 6:

$$\begin{cases} \gamma_k=-\frac{1}{N_3'(x_k)},\ k\in\mathbb{N}, \\ w_k=x_k+\gamma_k f(x_k),\ k\in\mathbb{W}, \\ y_k=x_k-\frac{f(x_k)}{f[x_k,w_k]}(1+q\frac{f(w_k)}{f[x_k,w_k]}),\ B_k=\frac{f[x_k,w_k]}{f[y_k,w_k]}, \\ x_{k+1}=y_k-\frac{f(y_k)}{f[y_k,x_k]+\lambda(y_k-x_k)(y_k-w_k)}(B_k+(B_k-1)^4). \end{cases} \tag{3.2}$$

2. *The new with-memory methods order 7.*(TEM7)
If $(1+\gamma f'(\alpha))=0$ and $(q-c_2)=0$ it can be seen that these relations lead to $\gamma=\frac{-1}{f'(\alpha)}=\frac{-1}{N_3'(x_k)}$ and $q=c_2=\frac{f''(\alpha)}{2f'(\alpha)}$. Each iteration, $x_k,\ w_k,\ y_k,\ x_{k+1}$ and w_{k+1}, are accessible, except at the initial step. Hence, we can interpolate $f'(\alpha)$ using these nodes, and as a result, we consider Newton interpolating polynomial as follows:

$$\begin{cases} N_4'(w_k)=[\frac{d}{dt}N_4(t;x_{k-1},w_{k-1},y_{k-1},x_k,w_k)]_{t=w_k}=[\frac{d}{dt}(f(w_k)+f[w_k,x_k] \\ .(t-w_k)+f[w_k,x_k,y_{k-1}](t-w_k)(t-x_k)+f[w_k,x_k,y_{k-1},w_{k-1}](t-w_k) \\ .(t-x_k)(t-y_{k-1})+f[w_k,x_k,y_{k-1},w_{k-1},x_{k-1}](t-w_k)(t-x_k)(t-y_{k-1}) \\ (t-w_{k-1})]_{t=w_k}=f[w_k,x_k]+f[w_k,x_k,y_{k-1}](w_k-x_k)+f[w_k,x_k,y_{k-1}, \\ w_{k-1}](w_k-x_k)(w_k-y_{k-1})+f[w_k,x_k,y_{k-1},w_{k-1},x_{k-1}] \\ .(w_k-x_k)(w_k-y_{k-1})(w_k-w_{k-1}). \end{cases} \tag{3.3}$$

Now if $\gamma=\frac{-1}{f'(\alpha)}=\frac{-1}{N_3'(x_k)}$ and $q=c_2=\frac{f''(\alpha)}{2f'(\alpha)}=\frac{N_4''(w_k)}{2N_4'(w_k)}$ then, we have a new method with memory as follows order 7:

$$\begin{cases} \gamma_k=-\frac{1}{N_3'(x_k)},\ q_k=\frac{N_4''(w_k)}{2N_4'(w_k)},\ k\in\mathbb{N}, \\ w_k=x_k+\gamma_k f(x_k),\ k\in\mathbb{W}, \\ y_k=x_k-\frac{f(x_k)}{f[x_k,w_k]}(1+q_k\frac{f(w_k)}{f[x_k,w_k]}),\ B_k=\frac{f[x_k,w_k]}{f[y_k,w_k]}, \\ x_{k+1}=y_k-\frac{f(y_k)}{f[y_k,x_k]+\lambda(y_k-x_k)(y_k-w_k)}(B_k+(B_k-1)^4). \end{cases} \tag{3.4}$$

3. *The new with-memory methods order 7.23*(LSGAM7.2)
If $(1+\gamma f'(\alpha))=0$, $(q-c_2)=0$ and $(-\lambda+f'(\alpha)((q-2c_2)c_2+c_3))=0$ it can be seen that these relations lead to $\gamma=\frac{-1}{f'(\alpha)}=\frac{-1}{N_3'(x_k)}$, $q=c_2=\frac{f''(\alpha)}{2f'(\alpha)}=\frac{N_4''(w_k)}{2N_4'(w_k)}$ and $\lambda=f'(\alpha)((q-2c_2)c_2+c_3)=\frac{f''^2(\alpha)}{-4f'(\alpha)}+\frac{f'''(\alpha)}{6}=\frac{N_5''^2(w_k)}{-4N_5'(w_k)}+\frac{N_5'''(w_k)}{6}$, then, we earn the new-family with-memory method following order 7.23:
$$\begin{cases}\gamma_k=-\frac{1}{N_3'(x_k)},\ q_k=\frac{N_4''(w_k)}{2N_4'(w_k)},\ \lambda_k=\frac{N_5''^2(w_k)}{-4N_5'(w_k)}+\frac{N_5'''(w_k)}{6},\ k\in\mathbb{N},\\ w_k=x_k+\gamma_k f(x_k),\ y_k=x_k-\frac{f(x_k)}{f[x_k,w_k]}(1+q_k\frac{f(w_k)}{f[x_k,w_k]}),\ k\in\mathbb{W},\\ B_k=\frac{f[x_k,w_k]}{f[y_k,w_k]},\ x_{k+1}=y_k-\frac{f(y_k)}{f[y_k,x_k]+\lambda_k(y_k-x_k)(y_k-w_k)}(B_k+(B_k-1)^4).\end{cases}\tag{3.5}$$

4. *The new with-memory methods order 7.53*(LSGAM7.5)
If $(1+\gamma f'(\alpha))=0$, $(q-c_2)=0$ and $(-\lambda+f'(\alpha)((q-2c_2)c_2+c_3))=0$ it can be seen that these relations lead to $\gamma=\frac{-1}{f'(\alpha)}=\frac{-1}{N_3'(x_k)}$ and $q=c_2=\frac{f''(\alpha)}{2f'(\alpha)}=\frac{N_4''(w_k)}{2N_4'(w_k)}$. Each iteration, x_k, w_k, y_k, x_{k+1}, w_{k+1} and y_{k+1}, are accessible, except at the initial step. Hence, we can interpolate $f'(\alpha)$ using these nodes, and as a result, we consider Newton interpolating polynomial as follows:
$$\begin{cases}N_5'(y_k)=[\frac{d}{dt}N_5(t;x_{k-1},w_{k-1},y_{k-1},x_k,w_k,y_k)]_{t=y_k}=[\frac{d}{dt}(f(y_k)+f[y_k,w_k]\\ .(t-y_k)+f[y_k,w_k,x_k](t-y_k)(t-w_k)+f[y_k,w_k,x_k,y_{k-1}](t-y_k)\\ .(t-w_k)(t-x_k)+f[y_k,w_k,x_k,y_{k-1},w_{k-1}](t-y_k)(t-w_k)(t-x_k)(t-y_{k-1})\\ +f[y_k,w_k,x_k,y_{k-1},w_{k-1},x_{k-1}](t-y_k)(t-w_k)(t-x_k)(t-y_{k-1})(t-w_{k-1})\\ ]_{t=y_k}=f[y_k,w_k]+f[y_k,w_k,x_k](y_k-w_k)+f[y_k,w_k,x_k,y_{k-1}](y_k-w_k)\\ .(y_k-x_k)+f[y_k,w_k,x_k,y_{k-1},w_{k-1}](y_k-w_k)(y_k-x_k)(y_k-y_{k-1})\\ +f[y_k,w_k,x_k,y_{k-1},w_{k-1},x_{k-1}](y_k-w_k)(y_k-x_k)(y_k-y_{k-1})(y_k-w_{k-1}).\end{cases}\tag{3.6}$$

Now if $\gamma=\frac{-1}{f'(\alpha)}=\frac{-1}{N_3'(x_k)}$, $q=c_2=\frac{f''(\alpha)}{2f'(\alpha)}=\frac{N_4''(w_k)}{2N_4'(w_k)}$ and $\lambda=f'(\alpha)((q-2c_2)c_2+c_3)=\frac{f''^2(\alpha)}{-4f'(\alpha)}+\frac{f'''(\alpha)}{6}=\frac{N_5''^2(y_k)}{-4N_5'(y_k)}+\frac{N_5'''(y_k)}{6}$, then, we earn the new-family with-memory method following order 7.53:
$$\begin{cases}\gamma_k=-\frac{1}{N_3'(x_k)},\ q_k=\frac{N_4''(w_k)}{2N_4'(w_k)},\ \lambda_k=\frac{N_5''^2(y_k)}{-4N_5'(y_k)}+\frac{N_5'''(y_k)}{6},\ k\in\mathbb{N},\\ w_k=x_k+\gamma_k f(x_k),\ y_k=x_k-\frac{f(x_k)}{f[x_k,w_k]}(1+q_k\frac{f(w_k)}{f[x_k,w_k]}),\ k\in\mathbb{W},\\ B_k=\frac{f[x_k,w_k]}{f[y_k,w_k]},\ x_{k+1}=y_k-\frac{f(y_k)}{f[y_k,x_k]+\lambda_k(y_k-x_k)(y_k-w_k)}(B_k+(B_k-1)^4).\end{cases}\tag{3.7}$$

5. *The new with-memory methods order 7.77*(LSGAM7.7)
If $\gamma=\frac{-1}{N_4'(x_k)}$, $q=\frac{N_5''(w_k)}{2N_5'(w_k)}$ and $\lambda=\frac{N_6''^2(y_k)}{-4N_6'(y_k)}+\frac{N_6'''(w_y)}{6}$, then we have a new method with memory following with order 7.77:
$$\begin{cases}\gamma_k=-\frac{1}{N_4'(x_k)},\ q_k=\frac{N_5''(w_k)}{2N_5'(w_k)},\ \lambda_k=\frac{N_6''^2(y_k)}{-4N_6'(y_k)}+\frac{N_6'''(y_k)}{6},\ k\in\mathbb{N},\\ w_k=x_k+\gamma_k f(x_k),\ y_k=x_k-\frac{f(x_k)}{f[x_k,w_k]}(1+q_k\frac{f(w_k)}{f[x_k,w_k]}),\ k\in\mathbb{W},\\ B_k=\frac{f[x_k,w_k]}{f[y_k,w_k]},\ x_{k+1}=y_k-\frac{f(y_k)}{f[y_k,x_k]+\lambda_k(y_k-x_k)(y_k-w_k)}(B_k+(B_k-1)^4).\end{cases}\tag{3.8}$$

The proof of the order of convergence of the methods mentioned in relations 3.2, 3.4, 3.5, 3.7 and 3.8 is similar. The order of convergence has been obtained with memory methods 3.5, 3.7, and 3.8 in [22].

4. Family of adaptive methods with memory

To get the best result, we suggest that all these relations hold simultaneously. The following equations hold:

$$\begin{cases} \gamma_k = \frac{-1}{f'(\alpha)}, \\ q_k = \frac{f''(\alpha)}{2f'(\alpha)}, \\ \lambda_k = \frac{f'''(\alpha)}{6} - \dfrac{f''^2(\alpha)}{4f'(\alpha)}, \end{cases} \tag{4.1}$$

Since α is unknown, it is impossible to compute $f'(\alpha), f''(\alpha)$, and $f'''(\alpha)$. Even worse, if we assume that α is known, computing $f'(\alpha)$, $f''(\alpha)$, and $f'''(\alpha)$ are not suggested since it increases function evaluations and it spoils that optimality of the method (2.1). Following the same idea in the with memory methods, this issue can be resaved. However, we are going to do it more efficiently, say recursive adaptively. Let us describe it a little more. The accelerators are updated using the information from all previous iterations in such a way that the highest efficiency indices obtain. Hence

$$\begin{cases} \gamma_k = \frac{-1}{f'(\alpha)} \simeq -\frac{1}{N_3'(x_k)}, \\ q_k = \frac{f''(\alpha)}{2f'(\alpha)} \simeq \frac{N_4''(w_k)}{2N_4'(w_k)}, \\ \lambda_k = \frac{f'''(\alpha)}{6} - \dfrac{f''^2(\alpha)}{4f'(\alpha)} \simeq \frac{N_5''^2(y_k)}{-4N_5'(y_k)} + \frac{N_5'''(y_k)}{6}, \end{cases} \tag{4.2}$$

where $N_3'(x_k), N_4''(w_k)$ and $N_5'''(y_k)$ are Newton's interpolation polynomials for the nodes $\{x_k, x_{k-1}, w_{k-1}, y_{k-1}\}, \{w_k, x_k, x_{k-1}, w_{k-1}, y_{k-1}\}$ and $\{y_k, w_k, x_k, x_{k-1}, w_{k-1}, y_{k-1}\}$, respectively. To construct a recursive adaptive method with memory, we use the information not only in the current and its previous iterations but also in all the previous iterations, i.e., from the beginning to the current iteration. Thus, as iterations proceed, the degree of interpolation polynomials increase, and the best-updated approximations for computing the self-accelerator γ_k, q_k, and λ_k are obtained. Indeed, we have developed the following recursive adaptive method with memory. Let x_0, γ_0, q_0, and λ_0 are given suitably. Then:

$$\begin{cases} \gamma_k = -\frac{1}{N_{3k}'(x_k)},\ q_k = \frac{N_{3k+1}''(w_k)}{2N_{3k+1}'(w_k)},\ \lambda_k = \frac{N_{3k+2}''^2(y_k)}{-4N_{3k+2}'(y_k)} + \frac{N_{3k+2}'''(y_k)}{6},\ k \in \mathbb{N}, \\ w_k = x_k + \gamma_k f(x_k),\ y_k = x_k - \frac{f(x_k)}{f[x_k,w_k]}(1 + q_k \frac{f(w_k)}{f[x_k,w_k]}),\ k \in \mathbb{W}, \\ B_k = \frac{f[x_k,w_k]}{f[y_k,w_k]},\ x_{k+1} = y_k - \frac{f(y_k)}{f[y_k,x_k]+\lambda_k(y_k-x_k)(y_k-w_k)}(B_k + (B_k-1)^4). \end{cases} \tag{4.3}$$

In what follows, we discuss the general convergence analysis of the recursive adaptive method with memory (4.15). It should be noted that the convergence order varies as the iteration go ahead. First, we need the following lemma:

Lemma 4.1. *If* $\gamma_k = -\frac{1}{N_{3k}'(x_k)}, q_k = -\frac{N_{3k+1}''(w_k)}{2N_{3k+1}'(w_k)}$, *and* $\lambda_k = \frac{N_{3k+2}''^2(y_k)}{-4N_{3k+2}'(y_k)} + \frac{N_{3k+2}'''(y_k)}{6}$, *then*

$$(1 + \gamma_k f'(\alpha)) \sim \prod_{s=0}^{k-1} e_s e_{s,w} e_{s,y}, \tag{4.4}$$

$$(q_k - c_2) \sim \prod_{s=0}^{k-1} e_s e_{s,w} e_{s,y}, \tag{4.5}$$

$$(-\lambda_k + f'(\alpha)((q_k - 2c_2)c_2 + c_3)) \sim \prod_{s=0}^{k-1} e_s e_{s,w} e_{s,y}, \tag{4.6}$$

where $e_s = x_s - \alpha, e_{s,w} = w_s - \alpha, e_{s,y} = y_s - \alpha$.

Proof: The proof is similar to Lemma 1 in [44, 47]. $\square$

Theorem 4.2. *Let x_0 be a suitable initial guess to the simple root α of $f(x) = 0$. Also, suppose the initial values $\gamma_0, q_0,$ and λ_0 are chosen appropriately. Then the R-order of the recursive adaptive method with memory (4.3) can be obtained from the following system of nonlinear equations:*

$$\begin{cases} r^k r_1 - (1 + r_1 + r_2)(1 + r + r^2 + r^3 + \ldots + r^{k-1}) - r^k = 0, \\ r^k r_2 - 2(1 + r_1 + r_2)(1 + r + r^2 + r^3 + \ldots + r^{k-1}) - 2r^k = 0, \\ r^{k+1} - 4(1 + r_1 + r_2)(1 + r + r^2 + r^3 + \ldots + r^{k-1}) - 4r^k = 0, \end{cases} \tag{4.7}$$

where r, r_1 and r_2 are the order of convergence of the sequences $\{x_k\}, \{w_k\},$ and $\{y_k\}$, respectively. Also, k, indicates the number of iterations.

Proof . Let $\{x_k\}, \{w_k\},$ and $\{y_k\},$ are convergent with orders r, $r_1,$ and $r_2,$ respectively. Then:

$$\begin{cases} e_{k+1} \sim e_k^r \sim e_{k-1}^{r^2} \sim \ldots \sim e_0^{r^{k+1}}, \\ e_{k,w} \sim e_k^{r_1} \sim e_{k-1}^{r_1 r} \sim \ldots \sim e_0^{r_1 r^k}, \\ e_{k,y} \sim e_k^{r_2} \sim e_{k-1}^{r_2 r} \sim \ldots \sim e_0^{r_2 r^k}, \end{cases} \tag{4.8}$$

where $e_k = x_k - \alpha, e_{k,w} = w_k - \alpha$ and $e_{k,y} = y_k - \alpha$. Now, by using Lemma (4.1) and Equation (4.8), we obtain

$$\begin{aligned} (1 + \gamma_k f'(\alpha)) \sim \prod_{s=0}^{k-1} e_s e_{s,w} e_{s,y} &= (e_0 e_{0,w} e_{0,y}) \ldots (e_{k-1} e_{k-1,w} e_{k-1,y}) \\ &= (e_0 e_0^{r_1} e_0^{r_2})(e_0^r e_0^{r_1 r} e_0^{r_2 r}) \ldots (e_0^{r^{k-1}} e_0^{r^{k-1} r_1} e_0^{r^{k-1} r_2}) \\ &= e_0^{(1+r_1+r_2)+(1+r_1+r_2)r+\ldots+(1+r_1+r_2)r^{k-1}} \\ &= e_0^{(1+r_1+r_2)(1+r+\ldots+r^{k-1})}. \end{aligned} \tag{4.9}$$

Similarly, we get :

$$(q_k - c_2) \sim e_0^{(1+r_1+r_2)(1+r+\ldots+r^{k-1})}, \tag{4.10}$$

and

$$(-\lambda_k + f'(\alpha)((q_k - 2c_2)c_2 + c_3)) \sim e_0^{(1+r_1+r_2)(1+r+\ldots+r^{k-1})}. \tag{4.11}$$

By considering the errors of $w_k, y_k,$ and x_{k+1} in Equation (4.8), and Equations (2.12)-(4.4), we conclude that

$$e_{k,w} \sim (1 + \gamma_k f'(\alpha))e_k \sim e_0^{(1+r_1+r_2)(1+r+\ldots+r^{k-1})} e_0^{r^k}, \tag{4.12}$$

$$e_{k,y} \sim -(1 + \gamma_k f'(\alpha))(q_k + c_2)e_k^2 \sim e_0^{((1+r_1+r_2)(1+r+\ldots+r^{k-1}))^2} e_0^{2r^k}, \tag{4.13}$$

$$\begin{aligned} e_{k+1} &\sim (1 + \gamma_k f'(\alpha))^2 (q_k - c_2)(-\lambda_k + f'(\alpha)((q_k - 2c_2)c_2 + c_3))e_k^4 \\ &\sim e_0^{((1+r_1+r_2)(1+r+\ldots+r^{k-1}))^4} e_0^{4r^k}. \end{aligned} \tag{4.14}$$

Equating the powers of error exponents of e_{k-1} in pairs of relations in Equations (4.8)-(4.12), (4.8)-(4.13), and (4.8)-(4.14), we have:

$$\begin{cases} r^k r_1 - (1+r_1+r_2)(1+r+r^2+r^3+\ldots+r^{k-1}) - r^k = 0, \ k=1,2,\ldots, \\ r^k r_2 - 2(1+r_1+r_2)(1+r+r^2+r^3+\ldots+r^{k-1}) - 2r^k = 0, \\ r^{k+1} - 4(1+r_1+r_2)(1+r+r^2+r^3+\ldots+r^{k-1}) - 4r^k = 0. \end{cases} \tag{4.15}$$

□

Remark 4.3. *For, $k = 1$, we use the information from the current and the one previous steps. In this case, the order of convergence of the with-memory method can compute from the following system*

$$\begin{cases} rr_1 - (1+r_1+r_2) - r = 0, \\ rr_2 - 2(1+r_1+r_2) - 2r = 0, \\ r^2 - 4(1+r_1+r_2) - 4r = 0. \end{cases} \tag{4.16}$$

This case special gives the solutions result of Lotfi et al. [22]. This system of equations has the following solution:
$r_1 = \frac{1}{8}(7+\sqrt{65}) \simeq 1.88, r_2 = \frac{1}{4}(7+\sqrt{65}) \simeq 3.76$ *and* $r = \frac{1}{2}(7+\sqrt{65}) \simeq 7.53$. *The same order as in 3.7.*

Remark 4.4. *For, $k = 2$, we obtain the order of convergence as follows:*
$r_1 \simeq 1.98612,\ r_2 \simeq 3.97225$, *and* $r \simeq 7.94449$. *Also, for $k = 3$, the system of equations(4.3) has the solution:* $r_1 \simeq 1.99829, r_2 \simeq 3.99657$, *and* $r \simeq 7.99315$. *(regarding TEM)*

Remark 4.5. *Likewise, for $k = 4$, we obtain the order of convergence:* $r_1 \simeq 1.99979, r_2 \simeq 3.99957$ *and* $r \simeq 7.99915$ *(regarding TEM8). In this case the efficiency index is* $7.99915^{\frac{1}{3}} = 1.99993 \cong 2$, *which shows that our developed method competes with all the existing with memory methods.*

Remark 4.6. *Can easily see that the improvement of the order of convergence from 4 to 8 (100% improvement) has attained without any additional functional evaluations. Therefore, the efficiency index of the proposed method (4.3) is* $EI = 8^{1/3} = 2$.

5. Numerical results and comparisons

The errors $|x_k - \alpha|$ of approximations to the sought zeros, produced by the different methods at the first three iterations have given in Table 2 where $m(-n)$ stands for $m \times 10^{-n}$. Tables $1-3$ also include, for each test function, the initial estimation values and the last value of the computational order of convergence COC [15] and order convergence p [50] computed by the expressions
(if it is stable)

$$COC = \frac{\log|f(x_n)/f(x_{n-1})|}{\log|f(x_{n-1})/f(x_{n-2})|} \approx p. \tag{5.1}$$

The package Mathematica 10, with 2000 arbitrary precision arithmetic, has been used in our computations. The following test functions have used:
$f_1(x) = \frac{-5x^2}{2} + x^4 + x^5 + \frac{1}{1+x^2},\ \alpha = 1,\ x_0 = 1.4.$
$f_2(x) = x\log(1+x\sin(x)) + e^{-1+x^2+x\cos(x)}\sin(\pi x),\ \alpha = 0,\ x_0 = 0.6,$
$f_3(x) = e^{x^2-3x}\sin(x) + \log(x^2+1),\ \alpha = 0,\ x_0 = 0.35,$

$f_4(x) = e^{-x^2+x+2} + e^{-1+x^2+x\cos(x)}\sin(\pi x) + 1,\ \alpha = 1.55031,\ x_0 = 1.3,$

Here, we compare the performance of the proposed method (2.1), (3.1), (3.3), (3.4), (3.6), (3.7), (4.15) and Abbasbandy's method order 3 (AM)[1], Artidiello et al.'s method order 8 (ACTVM)[2], Babajee et al.'s method order 8 (BCSTM)[3], Chun's method order 4 (CM)[5], Chun-Neta's method order 8 (CNM)[6], two-step with memory derivative-free Cordero et al. order 7 (CLTAM)[8], two-step with memory Cordero et al. order 6 (CLKTM)[7], two-step without memory Cordero et al. order 8 (CFGTM)[9], Choubey-Jaiswal's method order 8 (CJM) [10], Dzunic's method order 7 (DM) [11], Dzunic-Petkovic's method order 3 (DPM)[12], two-step with-derivative Kou's et al. order 4 (KLWM)[18], Kung-Traub's order 4 and 8 (KTM)[19], two-step without memory Lee-Kim order 4 (LKM)[20], three-step with memory Lotfi-Assari order 15.5 (LAM15)[21], three-step with memory Lotfi et al. order 12 (LSGAM) [22], three-step without memory Fardi et al. order 7 (FDGM) [13], Maheshwari's method order 4(MM)[24], Neta-Scott's method order 8(NSM)[25], Newton's method (NM)[27], Salimi et al.'s method order 8 (SLSSM)[32], Sharma-Arora's method order 8 (SAM)[30], Sharifi et al's methods by order 16 [33], order 8 [34] and order 12 [35], Soleymani's method order 10 (SM)[37], three-step with memory Soleymani et al. order 12 (SLTKM)[38], three-step without memory Soleymani et al. order 8 (SSMM)[39], two-step without memory Soleymani et al. order 4 (SKKM)[40], Steffensen's method with order 2 (SM) [41], Thukral-Petkovic's method order 8 (TPM)[42], one-step adaptive method Torkashvand et al.[43] Wang's method order 4.23 (WM)[46], two-step with memory Wang et al. order 7.53 (WZQM)[48], three-step with memory Wang et al. order 10, 11, 11.66 and 12 (WDZM)[49], two-step without memory Zheng et al. order 4 (ZLHM4) and 8 (ZLHM8)[51], two-step with memory Zheng et al. order 4.2361 (ZZLM) and 4.74483 (ZZLM)[52]. The results of comparison of the test functions are summarized in Tables 1-3. From Tables 1 and 3, we observe that the new scheme is superior than some existing methods.

6. Conclusion

In this work, the general the general Steffensen-like with-memory adaptive-family method proposed for solving nonlinear equations. To this end, Newton's interpolatory polynomial with different degrees has applied. The numerical results show that the proposed method is more useful to find an acceptable approximation of the exact solution of nonlinear equations, especially when the function is non-differentiable. In Tables $1-3$, we have examined some methods with different kinds of convergence order. Table 1 compares iterative methods with and without memory and the proposed method on functions $f_1(t),\ f_2(t),\ f_3(t),\ f_4(t)$. It has observed that these methods support their theoretical aspects. The fourth column of Table 1 shows the computational order of convergence by COC and the last column of the tables show the efficiency index defined by $EI = COC^{1/n}$, which is asymptotically 2. In other words, the proposed adaptive methods with memory (4.3) show behavior as optimal n-point methods without memory. Therefore, we have developed an adaptive-family with-memory method that has efficiency index 2. Moreover, the developed methods (4.3) do not need any derivatives and can be used even for non-smooth functions. Table 3 shows the convergence rate of adaptive-methods in comparison with the corresponding with-memory methods. This table results from the fact that we have reached the maximum improvement of the order of 100%. Figure 1 shows a comparison of without-memory methods, with memory and adaptive (%25, %50, %75, and %100 of improvements) in terms of the highest possible convergence order. Figure 2 shows a comparison of methods without memory, with memory and adaptive (%25, %50, %75, and %100 of improvements) in terms of the highest possible efficiency index. Studying basins of attraction of the methods can be considered for future researches. These methods are under development for the general case. In other words, the efficiency index of the proposed adaptive family with memory is

$8^{\frac{1}{3}} = 2$, which is much better than the optimal one-,..., five-point optimal methods without memory having efficiency indexes $2^{1/2} \simeq 1.41421$, $4^{1/3} \simeq 1.58740$, $8^{1/4} \simeq 1.68179$, $16^{1/5} \simeq 1.74110$, $32^{1/6} \simeq 1.78180$, respectively, also, $7.77^{1/3} \simeq 1.98065$. Adaptive methods with memory have minimum evaluation function, not evaluation derivative, and most efficiency index, hence competes with existing methods with- and without memory.

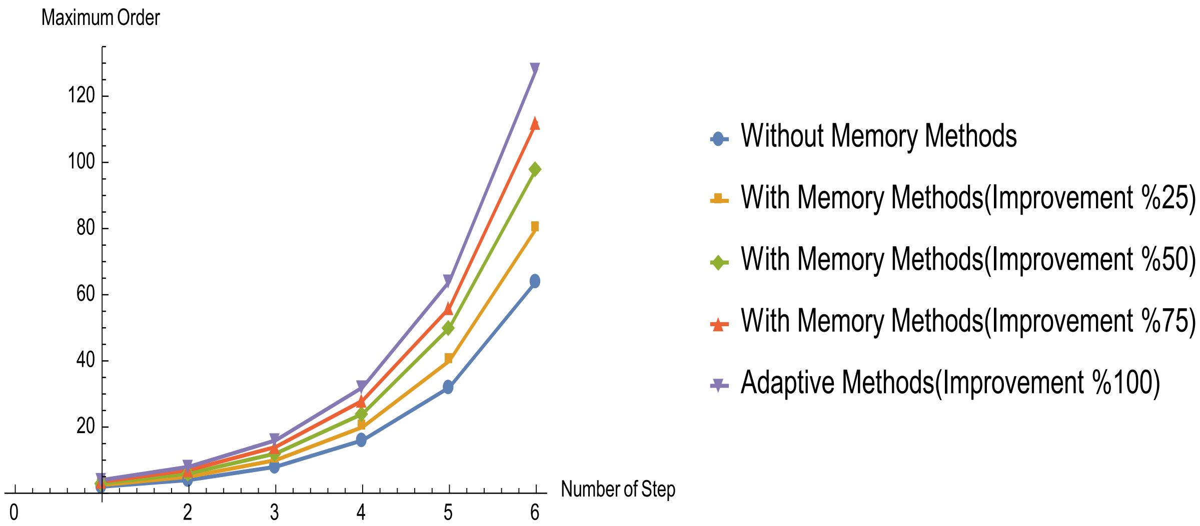

Figure 1: Comparison of methods without memory, with memory and adaptive (%25, %50, and %75 of improvements) in terms of highest possible convergence order.

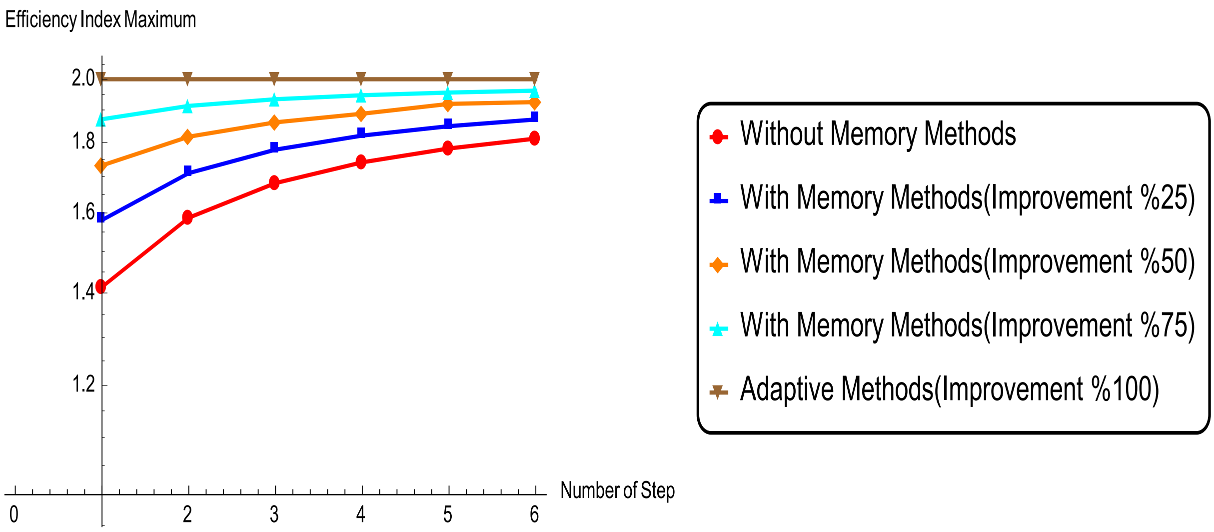

Figure 2: Comparison of methods without memory, with memory and adaptive (%25, %50, %75, and %100 of improvements) in terms of highest possible efficiency index.

Table 1: Comparison evaluation function and efficiency index of proposed method by with and without memory methods

without memory methods	EF	p	EI	with memory methods	EF	p	EI
AM[1]	3	3.0000	1.4423	CLKTM[7]	3	6.0000	1.8171
CM[5]	3	4.0000	1.4142	CLTAM[8]	3	7.0000	1.9129
KM[17]	3	4.0000	1.5874	DM[11]	3	7.0000	1.9129
GKM[14]	5	16.0000	1.7411	DPM[12]	2	3.0000	1.7321
JM[16]	3	4.0000	1.5874	DM[11]	2	3.5000	1.8708
KLWM[18]	3	4.0000	1.5874	PDP[28]	3	6.0000	1.8171
OM[27]	3	4.0000	1.5874	TM[19]	2	2.4100	1.5524
MM[24]	3	4.0000	1.5874	LAM15[21]	4	15.5000	1.9842
SAM[30]	4	8.0000	1.6818	LSGAM[22]	3	7.7700	1.9807
RWBM[31]	3	4.0000	1.5874	LSNKKM[23]	3	6.0000	1.8171
SSSLM[33]	5	16.0000	1.7411	LSNKKM[23]	4	12.0000	1.8612
TPM[42]	4	8.0000	1.6818	SLTKM[38]	4	12.0000	1.8612
SM[41]	2	2.0000	1.4142	SLTKM[38]	3	7.2200	1.9328
SSMM[39]	3	4.0000	1.5874	SM[37]	4	10.0000	1.7783
SKKM[40]	3	4.0000	1.5874	WZM[47]	3	5.0000	1.7100
KTM[19]	3	4.0000	1.5874	WZQM[48]	3	7.5300	1.9600
LKM[20]	4	4.0000	1.5874	WM[48]	3	7.0174	1.9145
SLSSM[32]	4	8.0000	1.6818	WM[46]	3	4.2361	1.6180
ACTVM[2]	4	8.0000	1.6818	WM[46]	3	4.4495	1.6448
SM[36]	4	8.0000	1.6818	WM[46]	3	4.3028	1.6265
ZLHM2[51]	2	2.0000	1.4142	SSSM[35]	4	12.0000	1.8612
NSM[25]	3	3.0000	1.4422	WDZM[49]	4	10.0000	1.7783
SFSSM[34]	4	8.0000	1.6818	WDZM[49]	4	11.0000	1.8212
BCSTM[3]	4	8.0000	1.6818	WDZM[49]	4	11.6600	1.8479
CNM[6]	4	8.0000	1.6818	WDZM[49]	4	12.0000	1.8612
CJM[10]	4	8.0000	1.6818	ZZLM[52]	3	4.2316	1.6175
CFGTM[9]	4	8.0000	1.6818	ZZLM[52]	3	4.7448	1.6804
ZLHM16[51]	5	16.0000	1.7411	TLFM[43]	3	8.0000	2.0000
WFM[50]	3	3.0000	1.4423	TEM(4.15), k=2	3	7.9400	1.9950
FDGM[13]	4	7.0000	1.6266	TEM(4.15), k=3	3	7.9932	1.9994
ZLHM8[51]	4	8.0000	1.6818	TEM8(4.15), k=4	3	8.0000	2.0000

Table 2: Comparison of the absolute errors and COC of proposed methods

$f_1(x)=\frac{-5x^2}{2}+x^4+x^5+\frac{1}{1+x^2}$, $\alpha=1$, $x_0=1.4$, $\beta_0=0.1$, $p_0=-1$, $q_0=1$					
Methods	$\lvert x_1-\alpha\rvert$	$\lvert x_2-\alpha\rvert$	$\lvert x_3-\alpha\rvert$	COC	EI
LSGAM4 (2.1)	0.11555(0)	0.55867(−2)	0.10920(−6)	3.9850	1.58541
TEM6 (2.14)	0.17922(0)	0.11180(−2)	0.23559(−14)	6.0000	1.81712
TEM7 (3.3)	0.17922(0)	0.24723(−4)	0.59492(−29)	7.0063	1.91350
LSGAM7.2 (3.4)	0.17922(0)	0.52733(−3)	0.11285(−22)	7.4873	1.95633
LSGAM7.5 (3.6)	0.17922(0)	0.59291(−3)	0.28780(−22)	7.5350	1.96047
LSGAM7.7 (3.7)	0.17922(0)	0.59291(−3)	0.28857(−22)	7.7862	1.98202
TEM8 (4.15), $k=4$	0.17922(0)	0.59291(−3)	0.28857(−22)	8.0000	2.00000
$f_2(x)=x\log(1+x\sin(x))+e^{-1+x^2+x\cos(x)}\sin(\pi x)$, $\alpha=0$, $x_0=0.6$, $\beta_0=0.1$, $p_0=-1$, $q_0=1$					
LSGAM4 (2.1)	0.14680(1)	0.11181(1)	0.11213(1)	4.0037	1.58789
TEM6 (2.14)	0.14680(1)	0.11167(1)	0.11213(1)	6.0006	1.81718
TEM7 (3.3)	0.14680(1)	0.11192(1)	0.11213(1)	7.0002	1.91295
LSGAM7.2 (3.4)	0.14680(1)	0.11219(1)	0.11213(1)	7.5076	1.95809
LSGAM7.5 (3.6)	0.14680(1)	0.11219(1)	0.11213(1)	7.5308	1.96011
LSGAM7.7 (3.7)	0.14680(1)	0.11219(1)	0.11213(1)	7.5671	1.96325
TEM8 (4.15), $k=4$	0.14680(1)	0.11219(1)	0.11213(1)	8.0000	2.00000
$f_3(x)=e^{x^2-3x}\sin(x)+\log(x^2+1)$, $\alpha=0$, $x_0=0.35$, $\alpha=0$, $x_0=0.6$, $\beta_0=0.1$, $p_0=-1$, $q_0=1$					
LSGAM4 (2.1)	0.93406(−1)	0.21278(−3)	0.27326(−13)	4.0000	1.58740
TEM6 (2.14)	0.93406(−1)	0.19600(−5)	0.10280(−31)	6.0000	1.81712
TEM7 (3.3)	0.93406(−1)	0.33401(−6)	0.17714(−44)	7.0000	1.91293
LSGAM7.2 (3.4)	0.93406(−1)	0.31832(−7)	0.82387(−54)	7.5313	1.96015
LSGAM7.5 (3.6)	0.93406(−1)	0.19724(−7)	0.10531(−54)	7.5654	1.96311
LSGAM7.7 (3.7)	0.93406(−1)	0.19724(−7)	0.54476(−59)	7.7355	1.97771
TEM8 (4.15), $k=4$	0.93406(−1)	0.19724(−7)	0.54476(−59)	8.0000	2.00000
$f_4(x)=e^{-x^2+x+2}+e^{-1+x^2+x\cos(x)}\sin(\pi x)+1$, $\alpha=1.55031$, $x_0=1.3$, $\beta_0=0.1$, $p_0=-1$, $q_0=1$					
LSGAM4 (2.1)	0.98065(−1)	0.22402(−3)	0.44953(−5)	4.0712	1.59676
TEM6 (2.14)	0.98065(−1)	0.17038(−4)	0.44953(−5)	6.0000	1.81712
TEM7 (3.3)	0.98065(−1)	0.50709(−5)	0.44953(−5)	7.0000	1.91293
LSGAM7.2 (3.4)	0.98065(−1)	0.43638(−5)	0.44953(−5)	7.5266	1.95975
LSGAM7.5 (3.6)	0.98065(−1)	0.43638(−5)	0.44953(−5)	7.5908	1.96047
LSGAM7.7 (3.7)	0.98065(−1)	0.43068(−5)	0.44953(−5)	7.7345	1.97763
TEM8 (4.15), $k=4$	0.98065(−1)	0.43068(−5)	0.44953(−5)	8.0000	2.00000

Table 3: Comparison of the percentage improvement of the convergence rate

with memory methods	number of steps	optimal order	p	percentage increase
CLKTM[7]	2	4.0000	6.0000	%50
CLTAM[8]	2	4.0000	7.0000	%75
DM[11]	2	4.0000	7.0000	%75
DPM[12]	1	2.0000	3.0000	%50
LA15M[21]	3	8.0000	15.5000	%93.75
LSGAM[22]	2	4.0000	7.7700	%94.25
LSNKKM[23]	3	8.0000	12.0000	%50
PDPM[28]	2	4.0000	6.0000	%50
SSSM[35]	3	8.0000	12.0000	%50
WM[46]	2	4.0000	4.4400	%11
WZM[47]	2	4.0000	5.5700	%39.25
WZQM[48]	2	4.0000	7.5300	%88.25
TEM6(2.14)	2	4.0000	6.0000	%50
TEM7(3.2)	2	4.0000	7.0000	%75
LSGAM(3.4)	2	4.0000	7.2300	%80.5
TEM(4.15), $k = 2$	2	4.0000	7.9449	%98.62
TLAM[43]	2	2.0000	4.0000	%100
TEM8(4.15), $k = 4$	2	4.0000	8.0000	%100

References

[1] S. Abbasbandy, *Modified homotopy perturbation method for nonlinear equations and comparison with Adomian decomposition method*, Appl. Math. Comput. 172 (2006) 431–438.

[2] S. Artidiello, A. Cordero, J.R. Torregrosa and M.P. Vassileva, *Two weighted eight-order classes of iterative root-finding methods*, Int. J. Comput. Math. 92(9) (2015) 1790–1805.

[3] D.K.R. Babajee, A. Cordero, F. Soleymani and J.R. Torregrosa, *On improved three-step schemes with high efficiency index and their dynamics*, Numer. Algor. 65 (2014) 153–169.

[4] R. Behl and S.S. Motsa, *Geometric construction of eighth-order optimal families of Ostrowski's method*, Sci. World J. 2015 (2015) 1–11.

[5] C. Chun, *Cunstruction of Newton-like iteration methods for solving nonlinear equations*, Numer. Math. 104 (2006) 297–315.

[6] C. Chun and B. Neta, *An analysis of a new family of eighth-order optimal methods*, Appl. Math. Comput. 245 (2014) 86–107.

[7] A. Cordero, T. Lotfi, A. Khoshandi and J.R. Torregrosa, *An efficient Steffensen-like iterative method with memry*, Bull. Math. Soc. Sci. Math. Roum, Tome 58(106)(1) (2015) 49–58.

[8] A. Cordero, T. Lotfi, J.R. Torregrosa, P. Assari and K. Mahdiani, *Some new bi-accelarator two-point methods for solving nonlinear equations*, Comput. Appl. Math. 35 (2016) 251–267.

[9] A. Cordero, M. Fardi, M. Ghasemi and J.R. Torregrosa, *Accelerated iterative methods for finding solutions of nonlinear equations and their dynamical behavior*, Calcolo 51(1) (2014) 17–30.

[10] N. Choubey and J.P. Jaiswal, *An improved optimal eighth-order iterative scheme with its dynamical behaviour*, Int. J. Comput. Science. Math. 7(4) (2016) 361–370.

[11] J. Dzunic, *On efficient two-parameter methods for solving nonlinear equations*, Numer. Algor. 63 (2013) 549–569.

[12] J. Dzunic and M.S. Petkovic, *A cubicaly convergent Steffensen-like method for solving nonlinear equations*, Appl. Math. Lett. 25 (2012) 1881–1886.

[13] M. Fardi, M. Ghasemi and A. Davari, *New iterative methods with seventh-order convergence for solving nonlinear equations*, Int. J. Nonlinear Anal. Appl. 3(2) (2012) 31–37.

[14] Y. H. Geum and Y.I. Kim, *A biparametric family of four-step sixteenth-order root-finding methods with the optimal efficiency index*, Appl. Math. Lett. 24 (2011) 1336–1342.

[15] L.O. Jay, *A note on Q-order of convergence*, BIT. 41(2) (2001) 422–429.

[16] P. Jarratt, *Some efficient fourth-order multipoint methods for solving equations*, BIT. 9(2) (1969) 119–124.

[17] R.F. King, *A family of fourth order methods for nonlinear equations*, Siam. J. Numer. Anal. 10(5) (1973) 876–879.

[18] J. Kou, Y. Li and X. Wang, *A family of fourth-order methods for solving non-linear equations*, Appl. Math. Comput. 118(1) (2007) 1031–1036.

[19] H.T. Kung and J.F. Traub, *Optimal order of one-point and multipoint iteration*, J. Assoc. Comput. Mach. 21(4) (1974) 643–651.

[20] M.Y. Lee and Y.I. Kim, *A family of fast derivative-free fourth-order multipoint optimal methods for nonlinear equations*, Int. J. Comput. Math. 89(15) (2012) 2081–2093.

[21] T. Lotfi and P. Assari, *New three-and four-parametric iterative with memory methods with efficiency index near 2*, Appl. Math. Comput. 270 (2015) 1004–1010.

[22] T. Lotfi, F. Soleymani, M. Ghorbanzadeh and P. Assari, *On the cunstruction of some tri-parametric iterative methods with memory*, Numer. Algor. 70(4) (2015) 835–845.

[23] T. Lotfi, F. Soleymani, Z. Noori, A. Kilicman and F. Khaksar Haghani, *Efficient iterative methods with and without memory possessing high efficiency indices*, Dis. Dyn. Nat. Soc. 2014 (2014) 1–9.

[24] A.K. Maheshwari, *A fourth order iterative method for solving nonlinear equations*, Appl. Math. Comput. 211 (2009) 383–391.

[25] B. Neta and M. Scott, *On a family of Halley-like methods to find simple roots of nonlinear equations*, Appl. Math. Comput. 219 (2013) 7940–7944.

[26] J.M. Ortega and W.G. Rheinboldt, *Iterative Solutions of Nonlinear Equations in Several Variables*, Academic Press, New York, 1970.

[27] A.M. Ostrowski, *Solution of Equations and Systems of Equations*, Academic press, New York, 1960.

[28] M.S. Petkovic, J. Dzunic and L.D. Petkovic, *A family of two-point with memory for solving nonlinear equations*, Appl. Anal. Disc. Math. 5 (2011) 298–317.

[29] M.S. Petkovic, B. Neta, L.D. Petkovic and J. Dzunic, *Multipoint Methods for Solving Nonlinear Equations*, Elsevier, Amsterdam, 2013.

[30] J.Raj. Sharma and H. Arora, *An efficient family of weighted-Newton methods with optimal eighth order convergence*, Appl. Math. Lett. 29 (2014) 1–6.

[31] H. Ren, Q. Wu and W. Bi, *A class of two-step Steffensen type methods with fourth-order convergence*, Appl. Math. Comput. 209 (2009) 206–210.

[32] M. Salimi, T. Lotfi, S. Sharifi and S. Siegmund, *Optimal Newton–Secant like methods without memory for solving nonlinear equations with its dynamics*, Int. J. Comput. Math. 94(9) (2017) 1759–1777.

[33] S. Sharifi, M. Salimi, S. Siegmund and T. Lotfi, *A new class of optimal four-point methods with convergence order 16 for solving nonlinear equations*, Math. Comput. Simul. 119 (c) (2016) 69-90.

[34] S. Sharifi, M. Ferrara, M. Salimi and S. Siegmund, *New modification of Maheshwari's method with optimal eighth order of convergence for solving nonlinear equations*, Open Math. 14 (2016) 443–451.

[35] S. Sharifi, S. Siegmund and M. Salimi, *Solving nonlinear equations by a derivative-free form of the King's family with memory*, Calcolo 53 (2016) 201–215.

[36] F. Soleymani, *On a bi-parametric classes of optimal eighth-order derivative-free methods*, Int. J. Pure Appl. Math. 72 (1) (2011) 27–37.

[37] F. Soleymani, *Some optimal iterative methods and their with memory variants*, J. Egyp. Math. Soc. (2013) 1–9.

[38] F. Soleymani, T. Lotfi, E. Tavakoli and F. Khaksar Haghani, *Several iterative methods with memory using self-accelerators*, Appl. Math. Comput. 254 (2015) 452–458.

[39] F. Soleymani, M. Sharifi and B. S. Mousavi, *An improvement of Ostrowski's and King's techniques with optimal convergence order eight*, J. Optim. Theory. Appl. 153 (2012) 225–236.

[40] F. Soleymani, S. K. Khattri and S. Karimi Vanani, *Two new classes of optimal Jarratt-type fourth-order methods*, Appl. Math. Lett. 25 (2012) 847–853.

[41] J.F. Steffensen, *Remarks on iteration*, Scand. Aktuar. 16 (1933) 64–72.

[42] R. Thukral and M.S. Petkovic, *A family of three-point methods of optimal order for solving nonlinear equations*, J. Comput. Appl. Math. 233 (2010) 2278–2284.

[43] V. Torkashvand, T. Lotfi and M.A. Fariborzi Araghi, *A new family of adaptive methods with memory for solving nonlinear equations*, Math. Sci. 13 (2019) 1–20.

[44] V. Torkashvand and M. Kazemi, *On an Efficient Family with Memory with High Order of Convergence for Solving Nonlinear Equations*, Int. J. Industrial Mathematics, 12(2) (2020) 209–224.

[45] J.F. Traub, *Iterative Methods for the Solution of Equations*, Prentice Hall, New York, USA, 1964.

[46] X. Wang, *An Ostrowski-type method with memory using a novel self-accelerating parameter*, J. Comput. Appl. Math. 330 (2017) 1–18.

[47] X. Wang and T. Zhang, *A new family of Newton-type iterative methods with and without memory for solving nonlinear equations*, Calcolo 51 (2014) 1–15.

[48] X. Wang, T. Zhang and Y. Qin, *Efficient two-step derivative-free iterative methods with memory and their dynamics*, Int. J. Comput. Math. 93(8) (2015) 1–27.

[49] X. Wang, J. Dzunic and T. Zhang, *On an efficient family of derivative free three-point methods for solving nonlinear equations*, Appl. Math. Comput. 219 (2012) 1749–1760

[50] S. Weerakoon and T.G.I. Fernando, *A variant of Newtons method with accelerated third-order convergence*, J. Appl. Math. Comput. 13(8) (2000) 87–93.

[51] Q. Zheng, J. Li and F. Huang, *An optimal Steffensen-type family for solving nonlinear equations*, Appl. Math. Comput. 217 (2011) 9592–9597.

[52] Q. Zheng, X. Zhao and Y. Liu, *An optimal biparametric multipoint family and its self-acceleration with memory for solving nonlinear equations*, Algorithms 8 (4) (2015) 1111–1120.