International Journal of
Mathematical Modelling & Computations
Vol. 09, No. 04, Fall 2019, 239- 252

Structure of an Adaptive with Memory Method with Efficiency Index 2

V. Torkashvand*

Young Researchers and Elite Club, Shahr-e-Qods Branch, Islamic Azad University, Tehran, Iran.

Abstract. The current research develops derivative-free family with memory methods with 100% improvement in the order of convergence. They have three parameters. The parameters are approximated and increase the convergence order from 4 to $6, 7, 7.5$ and 8, respectively. Additionally, the new self-accelerating parameters are constructed by a new way. They have the properties of simple structures and they are calculated easily. The parameters do not increase the computational cost of the iterative methods. Numerical examples of the new schemes are given to support the theoretical results.

Received: 23 August 2019, Revised: 01 October 2019, Accepted: 15 November 2019.

Keywords: Adaptive method with memory; Accelerator parameter; Newton's interpolatory polynomial; R-order convergence; Nonlinear equation; Simple zeros.

AMS Subject Classification: 65B99, 41A25, 65H05, 34G20.

Index to information contained in this paper

1. Introduction

Finding rapidly and accurately the zeros of nonlinear functions is an interesting and challenging problem in the field of computational mathematics. This study considers iterative methods for solving a nonlinear equation of the form $f(x) = 0$, where $f : I \subseteq \mathbb{R} \to \mathbb{R}$ is a defined scalar function on an open interval I. Since analytical methods for solving such equations are almost non-existent, it is the only possible way to obtain approximate solutions by relying on numerical methods

*Corresponding author. Email: torkashvand1978@gmail.com

based on iterative procedures [1–16, 18–22, 24–26, 28, 29, 31, 33, 35–39, 41–45]. Newton's method [23] ($x_{k+1} = x_k - \frac{f(x_k)}{f'(x_k)}$) is considered the most famous method for solving nonlinear equations. The main weakness of this method is the derivative of the function. Steffensen [37] solved the problem by eliminating the first derivative of the function and approximated it as $(f'(x_k) \approx f[x_k, w_k],\ w_k = x_k + f(x_k))$, where $f[x_k, w_k] = \dfrac{f(x_k) - f(w_k)}{x_k - w_k}$. The Steffensen's without memory method, like the Newton's method, has a degree of convergence of two, with the same efficiency index. In following, Traub developed the first method with memory by applying Steffensen's method. He increased the order of convergence in this method from 2 to 2.41 without using any new information, but just by reusing the information of the previous step:

$$\begin{cases} \gamma_0,\ \gamma_k = -\frac{x_k - x_{k-1}}{f(x_k) - f(x_{k-1})},\ k = 1, 2, 3, \cdots, \\ x_{k+1} = x_k - \frac{\gamma_k f(x_k)^2}{f(x_k + \gamma_k f(x_k)) - f(x_k)},\ k = 0, 1, 2, \cdots. \end{cases} \tag{1}$$

In this study, we focus on finding new two-point derivative free techniques. They have optimal order of convergence with efficiency index 2 according to the hypothesis of Kung and Traub [19] concerning the optimality of multi-point iterations without memory. In this work, we are looking for ways to increase the efficiency index to the highest possible level of 2. In this way, we increased the order of convergence by as much as 100%. Then, we built optimal four-order classes of methods including two steps agreeing Kung and Traub's hypothesis. We next extended one of the new methods for simple zeros and also obtained further accelerations in convergence and computational efficiency index without much more functional evaluation by applying the concept of methods with recursive memory iteration. The discusion is followed by a short literature on derivative-free methods. Then, Section 2 presents the contributions and the structure of the without memory fourth order method. In Section 3, with recursive memory methods are constructed using the two-step methods in Section 2 changing their convergence methods into 6, 7, 7.5 and 8. Actually, here, the idea is to keep all the old information and increase the convergence order by an interpolating polynomial, which its order is growing per cycle. To the best of our knowledge, adaptive methods with memory have been considered in [39]. The first two-step without-memory method is attributed to Ostrowski who presented them in 1960 [24]:

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

In 1973, King studied the following single-parameter method [18]:

$$\begin{cases} y_k = x_k - \frac{f(x_k)}{f'(x_k)},\ k = 0, 1, 2, \cdots, \\ x_{k+1} = y_k - \frac{f(y_k)}{f'(x_k)} \frac{f(x_k) + \beta f(y_k)}{f(x_k) - (\beta - 2) f(y_k)}. \end{cases} \tag{3}$$

Subsequently, in 2005, Chun presented the following two-step method using Adomian decomposition method [6]:

$$\begin{cases} x^*_{k+1} = x_k - \frac{f(x_k)}{f'(x_k)},\ k = 0, 1, 2, \cdots, \\ x_{k+1} = x_k - \frac{f(x_k)}{f'(x_k)} - 2\frac{f(x^*_{k+1})}{f'(x_k)} + \frac{f(x^*_{k+1}) f'(x^*_{k+1})}{f'(x_k)^2}. \end{cases} \tag{4}$$

In 2009, Ren et al. [28] presented the following two-step methods:

$$\begin{cases} y_k = x_k - \frac{f(x_k)}{f[x_k, z_k]}, \; z_k = x_k + f(x_k), \; k = 0, 1, 2, \cdots, \\ x_{k+1} = y_k - \frac{f(y_k)}{f[x_k, y_k] + f[y_k, z_k] - f[x_k, z_k] + \gamma(y_k - x_k)(y_k - z_k)}. \end{cases} \tag{5}$$

Zheng et al. [45], in 2011, presented the following two-step steffensen-like method for solving nonlinear equations:

$$\begin{cases} y_k = x_k - \frac{f(x_k)}{f[x_k, z_k]}, \; z_k = x_k + \gamma f(x_k), \; k = 0, 1, 2, \cdots, \\ x_{k+1} = y_k - \frac{f(y_k)}{f[x_k, y_k] + f[y_k, x_k, z_k](y_k - x_k)}. \end{cases} \tag{6}$$

And in 2013, Cordero et al. [9] proposed the following two-step method, which is the fourth order:

$$\begin{cases} y_k = x_k - \frac{f(x_k)}{f[x_k, z_k]}, \; z_k = x_k + f(x_k), \; k = 0, 1, 2, \cdots, \\ x_{k+1} = y_k - \frac{f(y_k)}{f[x_k, z_k]} \frac{f[x_k, y_k]}{f[y_k, z_k]}, \end{cases} \tag{7}$$

Based on the scheme (7), Jaiswal [15] proposed a derivative free family of two-step methods with memory having improved order and efficiency.

$$\begin{cases} y_k = x_k - \frac{f(x_k)}{f[x_k, z_k] + \delta_k f(z_k)}, \; z_k = x_k + \beta_k f(x_k), \\ x_{k+1} = y_k - \frac{f(y_k)}{f[x_k, z_k]} \frac{f[x_k, y_k]}{f[y_k, z_k]}, \; k = 0, 1, 2, \cdots, \end{cases} \tag{8}$$

We transform this method with partial modification and the entry of three self-referential parameters into a without-memory-optimized method with convergence order (Section 2). Then, by using interpolation Newton's method for parameters, we construct with memory methods with convergence rates of 6,7,7.5,8 (Section 3). In Section 3, we propose a new memory recursive method by imposing a more free parameter. This self-accelerating parameter is applied to improve the order until 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 with and without memory methods but without memory methods were optimal-order. Here some nonlinear equations are presented to test and show applicability and competitiveness of the developed methods.

2. Modified Steffensen-Like methods

In this section, the first goal is to modify Cordero et al.'s method [9] slightly so that its error equation can provide better form case with memory methods. In fact, we prove that our modified method can generate order of convergence of 8 while theirs has order of convergence of 7 in the case of with memory. Cordero et al.'s method has the iterative expression as follows:

$$e_{k+1} = (1 + f'(\alpha))^2 c_2 (2c_2^2 - c_3) e_k^4 + O(e_k^5). \tag{9}$$

To transform Eq. (8) into a method with memory, with three accelerators:

$$\begin{cases} y_k = x_k - \frac{f(x_k)}{f[x_k, w_k] + \beta f(w_k)}, \; w_k = x_k + \gamma f(x_k), \; k = 0, 1, 2, \cdots, \\ x_{k+1} = y_k - \frac{f(y_k)}{f[x_k, w_k]} \frac{f[x_k, y_k]}{f[y_k, w_k] + \lambda(y_k - x_k)(y_k - w_k)}, \end{cases} \tag{10}$$

where γ, β and λ are arbitrary nonzero parameters. In what follows, we present the error equation of Eq. (10).

Theorem 2.1 *Let $I \subseteq \mathbb{R}$ be an open interval, $f : I \to \mathbb{R}$ be a scalar function which has a simple root α in the open interval I, and also the initail approximation x_0 is sufficiently close the simple zero, then, the two-step iterative method (10) has four-order and satisfies the following error equation:*

$$e_{k+1} = \frac{(1 + f'(\alpha)\gamma)^2(\beta + c_2)(\lambda + f'(\alpha)c_2(\beta + 2c_2) - f'(\alpha)c_3)}{f'(\alpha)} e_k^4 + O(e_k^5). \quad (11)$$

Proof The following code has been written in the computational software package Mathematica by using symbolic computation. The following abbreviations are used. $fla = f'(\alpha)$, $e = x - \alpha$, $ew = w - \alpha$, $ey = y - \alpha$, $e_{k+1} = x_{k+1} - \alpha$, $c_k = \frac{f^k(\alpha)}{k!f'(\alpha)}$.

$In[1] : f[e\_] = fla(e + \sum_{i=2}^{4} c_i e^i);$

$In[2] : ew = e + \gamma Series[f[e], \{e, 0, 2\}]//Fullsimplify$

$Out[2] : (1 + \gamma fla)e + O[e]^2$

$In[3] : f[x\_, y\_] = \frac{f[x]-f[y]}{x-y};$

$In[4] : ey = e - Series[\frac{f[e]}{f[e,ew]+\beta f[ew]}, \{e, 0, 3\}]//Fullsimplify$

$Out[4] : (1 + fla\gamma)(\beta + c_2)e^2 + O(e^3)$

$In[5] : e_{k+1} = ey - Series[\frac{f[ey]}{f[e,ey]} \frac{f[e,ew]}{f[ey,ew]+\lambda(ey-e)(ey-ew)}, \{e, 0, 5\}]//Fullsimplify$

$Out[5] : \frac{(1+fla\gamma)^2(\beta+c_2)(\lambda+flac_2(\beta+2c_2)-flac_3)}{fla} e_k^4 + O(e_k^5)$ ■

3. With memory and adaptive methods

3.1 *With memory methods*

In this section, we first get with memory methods based on the without memory method mentioned in Equation (10). The equation error (11) also can be written as:

$$e_{k+1} = \frac{(1 + f'(\alpha)\gamma)^2(\beta + c_2)(\lambda + f'(\alpha)c_2(\beta + c_2) + f'(\alpha)c_2^2 - f'(\alpha)c_3)}{f'(\alpha)}. \quad (12)$$

We observe from (12) that the order of convergence of the presented methods (10) is 4 when $\gamma \neq \frac{1}{-f'(\alpha)}$, $\beta \neq -c_2$, and, $\lambda \neq \frac{f'''(\alpha)}{6} - \frac{f''(\alpha)^2}{4f'(\alpha)}$. The selection of $\gamma = \frac{1}{-f'(\alpha)}$, $\beta = -c_2 = -\frac{f''(\alpha)}{2f'(\alpha)}$, and, $\lambda = \frac{f'''(\alpha)}{6} - \frac{f''(\alpha)^2}{4f'(\alpha)}$ proves that the order of the presented method would be 7.5. We could approximate the parameters γ, λ, and, β by γ_k, λ_k, and, β_k:

$$\begin{cases} \gamma_k = -\frac{1}{f'(\alpha)} \approx -\frac{1}{N_3'(x_k)}, \\ \beta_k = -\frac{f''(\alpha)}{2f'(\alpha)} \approx -\frac{N_4'(w_k)}{2N_4''(w_k)}, \\ \lambda_k = \frac{f'''(\alpha)}{6} - \frac{f''(\alpha)^2}{4f'(\alpha)} \simeq \frac{N_5'''(y_k)}{6} - \frac{N_5''(y_k)^2}{4N_5'(y_k)}, \end{cases} \quad (13)$$

where $N_3(x_k); N_4(w_k)$ and $N_5(y_k)$ are defined as follows:

$$\begin{cases} N_3(x_k) = N_3(t; x_k, x_{k-1}, w_{k-1}, y_{k-1}), \\ N_4(w_k) = N_4(t; w_k, x_k, x_{k-1}, w_{k-1}, y_{k-1}), \\ N_5(y_k) = N_5(t; y_k, w_k, x_k, x_{k-1}, w_{k-1}, y_{k-1}). \end{cases}$$

We consider three methods with memory following:

(i) If we only interpolate parameter γ_k a six order with memory procedure is obtained:

$$\begin{cases} \gamma_k = -\frac{1}{N_3'(x_k)},\ k = 1, 2, 3, \cdots, \\ y_k = x_k - \frac{f(x_k)}{f[x_k, w_k] + \beta f(w_k)},\ w_k = x_k + \gamma_k f(x_k),\ k = 0, 1, 2, \cdots, \\ x_{k+1} = y_k - \frac{f(y_k)}{f[x_k, w_k]} \frac{f[x_k, y_k]}{f[y_k, w_k] + \lambda(y_k - x_k)(y_k - w_k)}. \end{cases} \tag{14}$$

(ii) To increase the 75% convergence rate, we will have the following method:

$$\begin{cases} \gamma_k = -\frac{1}{N_3'(x_k)},\ \beta_k = -\frac{N_4'(w_k)}{2N_4''(w_k)},\ k = 1, 2, 3, \cdots, \\ y_k = x_k - \frac{f(x_k)}{f[x_k, w_k] + \beta_k f(w_k)},\ w_k = x_k + \gamma_k f(x_k),\ k = 0, 1, 2, \cdots, \\ x_{k+1} = y_k - \frac{f(y_k)}{f[x_k, w_k]} \frac{f[x_k, y_k]}{f[y_k, w_k] + \lambda(y_k - x_k)(y_k - w_k)}. \end{cases} \tag{15}$$

(iii) Replacing the fixed parameters γ, λ and β in the iterative formula (10) by various parameters of γ_k, λ_k and β_k calculated in (13), the following derivative-free two-points scheme with memory is achieved:

$$\begin{cases} \gamma_k = -\frac{1}{N_3'(x_k)},\ \beta_k = -\frac{N_4'(w_k)}{2N_4''(w_k)},\ \lambda_k = \frac{N_5'''(y_k)}{6} - \frac{N_5''(y_k)^2}{4N_5'(y_k)},\ k = 1, 2, 3, \cdots, \\ y_k = x_k - \frac{f(x_k)}{f[x_k, w_k] + \beta_k f(w_k)},\ w_k = x_k + \gamma_k f(x_k),\ k = 0, 1, 2, \cdots, \\ x_{k+1} = y_k - \frac{f(y_k)}{f[x_k, w_k]} \frac{f[x_k, y_k]}{f[y_k, w_k] + \lambda_k(y_k - x_k)(y_k - w_k)}. \end{cases} \tag{16}$$

Lemma 3.1 *If* $\gamma_k = -\frac{1}{N_3'(x_k)}$, *then* $(1 + \gamma_k f'(\alpha)) \sim e_s e_{s,w} e_{s,y}$,

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

Theorem 3.2 *If an initial approximation x_0 is sufficiently close to the zero α of $f(x) = 0$ and the parameter γ_k in the iterative scheme (14) is recursively calculated by (13), then the R-order of convergence is at least 6.*

Proof First we assume that the $R-$order of convergence of sequence x_k, w_k, y_k is at least r, p and q, respectively.
Hence:

$$e_{k+1} \sim e_k^r \sim e_{k-1}^{r^2}, \tag{17}$$

$$e_{k,w} \sim e_k^p \sim e_{k-1}^{rp}, \tag{18}$$

and

$$e_{k,y} \sim e_k^q \sim e_{k-1}^{rq}. \tag{19}$$

By (17), (18), (19), and Lemma 3.1, we obtain

$$1 + \gamma_k f'(\alpha) \sim e_{k-1}^{p+q+1}. \tag{20}$$

On the other hand, we get

$$e_{k,w} \sim (1 + \gamma_k f'(\alpha))e_k, \tag{21}$$

$$e_{k,y} \sim (1 + \gamma_k f'(\alpha))e_k^2, \tag{22}$$

$$e_{k+1} \sim (1 + \gamma_k f'(\alpha))^2 e_k^4. \tag{23}$$

Combining (20)-(21), (20)-(22), and (20)-(23), we conclude

$$e_{k,w} \sim e_{k-1}^{(1+p+q)+r}, \tag{24}$$

$$e_{k,y} \sim e_{k-1}^{(1+p+q)+2r}, \tag{25}$$

and

$$e_{k+1} \sim e_{k-1}^{2(1+p+q)+4r}. \tag{26}$$

Equating the powers of error exponents of e_{k-1} in pair relations (18)-(24), (19)-(25), and (17)-(26), we have

$$\begin{cases} rp - r - (p + q + 1) = 0, \\ rq - 2r - (p + q + 1) = 0, \\ r^2 - 4r - 2(p + q + 1) = 0. \end{cases} \tag{27}$$

This system has the solution $p = 2$, $q = 4$ and $r = 6$ which specifies the $R-$order of convergence of the derivative-free scheme with memory (14). ■

Lemma 3.3 *If* $\gamma_k = -\frac{1}{N_3'(x_k)}$, $\beta_k = -\frac{N_4'(w_k)}{2N_4''(w_k)}$ *and* $\lambda_k = \frac{N_5'''(y_k)}{6} - \frac{N_5''(y_k)^2}{4N_5'(y_k)}$ *then*

$$(1 + \gamma_k f'(\alpha)) \sim e_s e_{s,w} e_{s,y}, \tag{28}$$

$$\beta_k + c_2 \sim e_s e_{s,w} e_{s,y}, \tag{29}$$

$$(\lambda_k + f'(\alpha)c_2(\beta_k + c_2) + f'(\alpha)c_2^2 - f'(\alpha)c_3) \sim e_s e_{s,w} e_{s,y}. \tag{30}$$

Theorem 3.4 *If an initial approximation x_0 is sufficiently close to the zero α of $f(x) = 0$ and the parameters γ_k, β_k and λ_k in the iterative schemes (15) and (16) are recursively calculated by (13), then the R-order of convergences are at least 7 and 7.53.*

Proof Proof of Theorem 3.4 is quite similar to the proof of Theorem 3.2. ■

3.2 *New families of iterative recursive methods with memory*

This section deals with the main contribution of the current study. In other words, it is attempted to introduce a recursive adaptive with memory method so that it has the highest possible efficiency index as proposed about methods with memory in the literature. It is worth mentioning that some special cases of this new method cover the existing methods. Following the same idea in the methods with memory, this issue can be resaved. However, we are going to do it in a more efficient way, that is recursive adaptive method. Let us describe it a little more. 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 increases, and the best updated approximations for computing the self-accelerators γ_k, β_k, and λ_k are obtained. Indeed, we have developed the following recursive adaptive method with memory. Then:

$$\begin{cases} x_0, \gamma_0, \beta_0, \lambda_0, \\ \gamma_k = -\frac{1}{N'_{3k}(x_k)}, \beta_k = -\frac{N''_{3k+1}(w_k)}{2N'_{3k+1}(w_k)}, \lambda_k = \frac{N'''_{3k+2}(y_k)}{6} - \frac{N''_{3k+2}(y_k)^2}{4N'_{3k+2}(y_k)},\ k = 1, 2, 3, \cdots, \\ y_k = x_k - \frac{f(x_k)}{f[x_k, z_k] + \beta_k f(w_k)},\ z_k = x_k + \gamma_k f(x_k),\ k = 0, 1, 2, \cdots, \\ x_{k+1} = y_k - \frac{f(y_k)}{f[x_k, z_k]} \frac{f[x_k, y_k]}{f[y_k, z_k] + \lambda_k (y_k - x_k)(y_k - z_k)}. \end{cases} \tag{31}$$

In what follows, we discuss the general convergence analysis of the recursive adaptive method with memory (31). It should be noted that the convergence order varies as the iterations proceed. First, we need the following lemma.

Lemma 3.5 *If* $\gamma_k = -\frac{1}{N'_{3k}(x_k)}, \beta_k = -\frac{N''_{3k+1}(w_k)}{2N'_{3k+1}(w_k)}$, *and* $\lambda_k = \frac{N'''_{3k+2}(y_k)}{6} - \frac{N''_{3k+2}(y_k)^2}{4N'_{3k+2}(y_k)}$, *then*

$$(1 + \gamma_k f'(\alpha)) \sim \prod_{s=0}^{k-1} e_s e_{s,w} e_{s,y}, \tag{32}$$

$$(c_2 + \beta_k) \sim \prod_{s=0}^{k-1} e_s e_{s,w} e_{s,y}, \tag{33}$$

$$(\lambda_k + f'(\alpha) c_2 (\beta_k + c_2) + f'(\alpha) c_2^2 - f'(\alpha) c_3) \sim \prod_{s=0}^{k-1} e_s e_{s,w} e_{s,y}. \tag{34}$$

Theorem 3.6 *Let* x_0 *be a suitable initial guess to the simple root* α *of* $f(x) = 0$. *Also, suppose the initial values* γ_0, β_0, *and* λ_0 *are chosen appropriately. Then the R-order of the recursive adaptive method with memory (31) can be obtained from the following system of nonlinear equations:*

$$\begin{cases} m^k m_1 - (1 + m_1 + m_2)(1 + m + m^2 + m^3 + \ldots + m^{k-1}) - m^k = 0, \\ m^k m_2 - 2(1 + m_1 + m_2)(1 + m + m^2 + m^3 + \ldots + m^{k-1}) - 2m^k = 0, \\ m^{k+1} - 4(1 + m_1 + m_2)(1 + m + m^2 + m^3 + \ldots + m^{k-1}) - 4m^k = 0, \end{cases} \tag{35}$$

where m, m_1 *and* m_2 *are the order of convergence of the sequences* $\{x_k\}, \{w_k\}$, *and* $\{y_k\}$, *respectively.*

Proof Let $\{x_k\}, \{w_k\}$, and $\{y_k\}$, be convergent with orders m, m_1, and m_2, respectively. Then

$$\begin{cases} e_{k+1} \sim e_k^m \sim e_{k-1}^{m^2} \sim \ldots \sim e_0^{m^{k+1}}, \\ e_{k,w} \sim e_k^{m_1} \sim e_{k-1}^{m_1 m} \sim \ldots \sim e_0^{m_1 m^k}, \\ e_{k,y} \sim e_k^{m_2} \sim e_{k-1}^{m_2 m} \sim \ldots \sim e_0^{m_2 m^k}, \end{cases} \tag{36}$$

where $e_k = x_k - \alpha, e_{k,w} = w_k - \alpha$ and $e_{k,y} = y_k - \alpha$. Now, by Lemma 3.1 and Eq (36), we obtain

$$\begin{aligned} (1+\gamma_k f'(\alpha)) &\sim \prod_{s=0}^{k-1} e_s e_{s,w} e_{s,y} \\ &= (e_0 e_{0,w} e_{0,y}) \ldots (e_{k-1} e_{k-1,w} e_{k-1,y}) \\ &= (e_0 e_0^{m_1} e_0^{m_2})(e_0^m e_0^{m_1 m} e_0^{m_2 m}) \ldots (e_0^{m^{k-1}} e_0^{m^{k-1} m_1} e_0^{m^{k-1} m_2}) \\ &= e_0^{(1+m_1+m_2)+(1+m_1+m_2)m+\ldots+(1+m_1+m_2)m^{k-1}} \\ &= e_0^{(1+m_1+m_2)(1+m+\ldots+m^{k-1})}. \end{aligned} \tag{37}$$

Similarly, we get

$$(\beta_k + c_2) \sim e_0^{(1+m_1+m_2)(1+m+\ldots+m^{k-1})}, \tag{38}$$

and

$$(\lambda_k + f'(\alpha)c_2(\beta_k + c_2) + f'(\alpha)c_2^2 - f'(\alpha)c_3) \sim e_0^{(1+m_1+m_2)(1+m+\ldots+m^{k-1})}. \tag{39}$$

By considering the errors of w_k, y_k, and x_{k+1} in (36), and relations (37)-(39), we conclude

$$e_{k,w} \sim (1+\gamma_k f'(\alpha)) e_k \sim e_0^{(1+m_1+m_2)(1+m+\ldots+m^{k-1})} e_0^{m^k}, \tag{40}$$

$$e_{k,y} \sim (1+\gamma_k f'(\alpha))(\beta_k + c_2) e_k^2 \sim e_0^{((1+m_1+m_2)(1+m+\ldots+m^{k-1}))^2} e_0^{2m^k}, \tag{41}$$

$$\begin{aligned} e_{k+1} &\sim \frac{(1+f'(\alpha)\gamma_k)^2(\beta_k+c_2)(\lambda_k + f'(\alpha)c_2(\beta_k+c_2) + f'(\alpha)c_2^2 - f'(\alpha)c_3)}{f'(\alpha)} e_k^4 \\ &\sim e_0^{((1+m_1+m_2)(1+m+\ldots+m^{k-1}))^4} e_0^{4m^k}. \end{aligned} \tag{42}$$

To obtain the desired result, it is enough to match the right-hand-side of the Eqs. (36), (40), (41), and (42). Then

$$\begin{cases} m^k m_1 - (1+m_1+m_2)(1+m+m^2+m^3+\ldots+m^{k-1}) - m^k = 0, \\ m^k m_2 - 2(1+m_1+m_2)(1+m+m^2+m^3+\ldots+m^{k-1}) - 2m^k = 0, \\ m^{k+1} - 4(1+m_1+m_2)(1+m+m^2+m^3+\ldots+m^{k-1}) - 4m^k = 0. \end{cases}$$

This completes the proof of the theorem. ■

Remark 3.7 For $k = 1$, we use the information from the current and the one previous steps. In this case, the order of convergence of the method with memory can be computed from the following system

$$\begin{cases} mm_1 - (1 + m_1 + m_2) - m = 0, \\ mm_2 - 2(1 + m_1 + m_2) - 2m = 0, \\ m^2 - 4(1 + m_1 + m_2) - 4m = 0. \end{cases} \tag{43}$$

It been shown by TM7. This system of equations has the solution:
$m_1 = \frac{1}{8}(7 + \sqrt{65}) \simeq 1.88$, $m_2 = \frac{1}{4}(7 + \sqrt{65}) \simeq 3.76$ and $m = \frac{1}{2}(7 + \sqrt{65}) \simeq 7.53$.

Also, for $k = 2$, we obtain the order of convergence: $m_1 \simeq 1.98612$, $m_2 \simeq 3.97225$ and $m \simeq 7.94449$.
For $k = 3$, the system of equations(31) has the solution: $m_1 \simeq 1.99829$, $m_2 \simeq 3.99657$ and $m \simeq 7.99315$.
Likewise, for $k = 4$, we obtain the order of convergence:

$$m_1 \simeq 1.99979,\ m_2 \simeq 3.99957 \text{ and } m \simeq 7.99915. \tag{44}$$

(been shown by TM8). In this case the efficiency index is $7.99915^{\frac{1}{3}} = 1.99993 \cong 2$ which shows that our developed method competes all the existing methods with memory.

4. Numerical results and comparisons

This section demonstrates the convergence behavior of the recursive methods with memory (31). All computations are performed using the programming package Mathematica with multiple-precision arithmetic. Tables $2-3$ also include, for each test function, the initial estimation values and the last value of the computational order of convergence r_c [26] computed by the expression

$$r_c \approx \frac{\log |f(x_n)/f(x_{n-1})|}{\log |f(x_{n-1})/f(x_{n-2})|}. \tag{45}$$

Iterative methods with and without memory, for comparison with our proposed scheme have been chosen as mentioned below.

(1) Iterative methods one-step with and without memory:
Abbasbandy (AM) [1], Dzunic (DM) [12], Newton (NM) [23], Steffensen (SM) [37], Torkashvand et al. (TLFM) [39] and Zheng et al. (ZLHM) [45].

(2) Iterative methods two-step with and without memory:
Behl et al. (BCMTM) [4], Choubey-Jaiswal (CJM) [5], Chun (CM) [6], Chun-Neta (CNM) [7], Jaiswal (JM) [15], Kansal et al. (KKBM) [16], King (KM) [18], Mohamadi et al. (MLAM) [22], Petkocic et al. (PDPM) [25], Petkovic et al. (PNPDM) [27], Ostrowski (OM) [24], Ren et al. (RWBM) [28], Soleymani et al. (SLTKM) [36], Soleymani et al. (SKJM) [35], Truab (TM) [41], Zaka Ullah et al. (ZK-

FKAM) [44], Wang (WM) [42] and Zheng et al. (ZLHM) [45].

(3) Iterative methods three-step with and without memory:
Choubey-Jaiswal (CJM) [5], Cordero et al. (CTVM) [11], Eftekhari (EM) [13], Jaiswal (JM) [15], Lotfi-Assari (LAM) [20], Khattri-Steihaug (KS) [17], Kung-Truab (KTM) [19], Maroju et al. (MBMM) [21], Petkovic et al. (PNPDM) [27], Sharma-Arora (SAM) [31], Sharma et al. (SGGM) [34], Sharma (SM) [32], Soleymani et al. (SLTKM) [36], Soleymani et al. (SKJM) [35], Thukral-Petkovic (TPM) [38] and Zheng et al. (ZLHM) [45].

(4) Iterative methods four-step without memory:
Geum-Kim (GKM) [14], Maroju et al. (MBMM) [21] and Zheng et al. (ZLHM) [45].

Table 1 lists the exact roots (α) and initial approximations (x_0). Tables $2-4$ show that the proposed methods compete the previous methods. In addition, its efficiency index is much better than the previous works. In other words, TM7 and TM8 have efficiency indices $7.53^{\frac{1}{3}} \simeq 1.96$, and $8^{\frac{1}{3}} = 2$. In order to check the effectiveness of the proposed methods, we have considered 7 test nonlinear functions. The results of comparisons are given in Tables $2-4$. These tables also include, for each test function, the initial estimation values and the last value of the computational order of convergence compared with convergence rate and EI in each method. A comparison between without memory, with memory and adaptive methods in terms of maximum efficiency index along with the number of steps in each cycle are given in Figure 1.

Table 1. Test functions.

Nonlinear function	Zero	Initial guess
$f_1(x) = t\log(1+x\sin(x)) + e^{-1+x^2+x\cos(x)}\sin(\pi x)$	$\alpha = 0$	$x_0 = 0.6$
$f_2(x) = 1 + \frac{1}{x^4} - \frac{1}{x} - x^2$	$\alpha = 1$	$x_0 = 1.4$
$f_3(x) = e^{x^3-x} - \cos(x^2-1) + x^3 + 1$	$\alpha = -1$	$x_0 = -1.65$
$f_4(x) = \frac{-5x^2}{2} + x^4 + x^5 + \frac{1}{1+x^2}$	$\alpha = 1$	$x_0 = 1.5$
$f_5(x) = \log(1+x^2) + e^{-3x+x^2}\sin(x)$	$\alpha = 0$	$x_0 = 0.5$
$f_6(x) = x\log(1-\pi+x^2) - \frac{1+x^2}{1+x^3}\sin(x^2) + \tan(x^2)$	$\alpha = \sqrt{\pi}$	$x_0 = 1.7$
$f_7(x) = x^3 + 4x^2 - 10$	$\alpha = 1$	$x_0 = 1.3652$

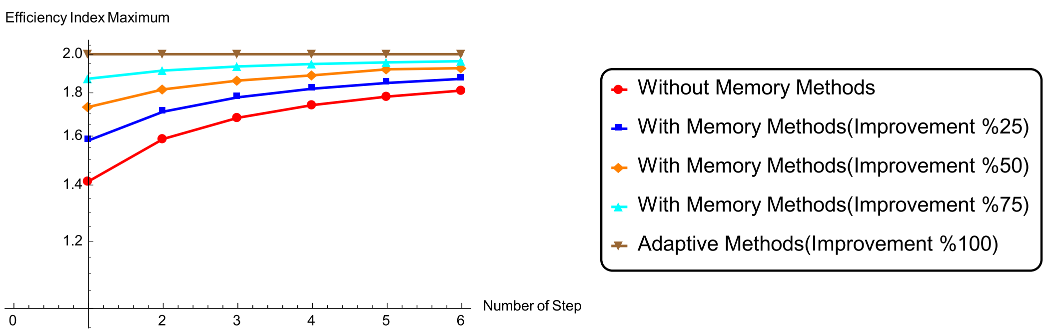

Figure 1. Comparison of methods without memory, with memory and adaptive in terms of the highest possible efficiency index.

Table 2. The numerical results of the proposed methods and other methods of with and without memory for $f_i(x), i = 1, 2, 3, 4$.

$f_1(x) = x\log(1+x\sin(x)) + e^{-1+x^2+x\cos(x)}\sin(\pi x),\ \alpha = 0,\ x_0 = 0.6, q_0 = \gamma_0 = \lambda_0 = \beta_0 = 0.1$					
Methods	$\lvert x_1-\alpha\rvert$	$\lvert x_2-\alpha\rvert$	$\lvert x_3-\alpha\rvert$	r_c	EI
AM [1]	0.60000(0)	0.44377(0)	0.10028(0)	3.0000	1.44225
DM[12]	0.60000(0)	0.36450(0)	0.54166(−1)	3.4590	1.85984
TM[41]	0.60000(0)	0.47811(0)	0.56230(−1)	2.3950	1.54758
JM[15]	0.60000(0)	0.20180(0)	0.17177(−5)	7.0000	1.91293
SLTKM[36]	0.60000(0)	0.22353(0)	0.30292(−5)	7.1871	1.92982
TPM[38] $(a = b = 0)$	0.60000(0)	0.15946(−1)	0.31506(−13)	8.0000	1.68179
LAM[20]	0.60000(0)	0.19386(−1)	0.12850(−28)	15.5240	1.98496
GKM [14]	0.60000(0)	0.20973(−3)	0.67159(−57)	16.0000	1.74110
SSSLM [29]	0.60000(0)	0.15176(−3)	0.77529(−57)	16.0000	1.74110
TM7 (3.2) k=1	0.60000(0)	0.21352(0)	0.24838(−5)	7.5457	1.96140
TM8 (3.2) k=4	0.21352(0)	0.24838(−5)	0.17810(−44)	8.0000	2.00000
$f_2(x) = 1 + \frac{1}{x^4} - \frac{1}{x} - x^2, \alpha = 1, x_0 = 1.4, q_0 = \gamma_0 = \lambda_0 = \beta_0 = 0.1$					
AM [1]	0.40000(0)	0.69117(−1)	0.84282(−3)	3.0000	1.44225
DM[12]	0.40000(0)	0.46538(−1)	0.12681(−3)	3.5552	1.88552
TM[41]	0.40000(0)	0.60801(−1)	0.28094(−2)	2.4157	1.55425
JM[15]	0.40000(0)	0.48060(−2)	0.18682(−14)	7.0000	1.91293
SLTKM[36]	0.40000(0)	0.37115(−2)	0.28982(−15)	7.2315	1.93379
TPM[38] $(a = b = 0)$	0.40000(0)	0.20584(−3)	0.94711(−27)	8.0000	1.68179
LAM[20]	0.40000(0)	0.33505(−4)	0.80238(−66)	15.5120	1.98457
GKM [14]	0.40000(0)	0.11559(−3)	0.22543(−57)	16.0000	1.74110
SSSLM [29]	0.40000(0)	0.11522(−2)	0.50671(−39)	16.0000	1.74110
TM7 (3.2) k=1	0.40000(0)	0.49798(−2)	0.59138(−16)	7.5333	1.96033
TM8 (3.2) k=4	0.49798(−2)	0.59138(−16)	0.16539(−127)	8.1885	2.01559
$f_3(x) = e^{x^3-x} - \cos(x^2-1) + x^3 + 1, \alpha = -1, x_0 = -1.5, q_0 = \gamma_0 = \lambda_0 = \beta_0 = 0.1$					
AM [1]	0.50000(0)	0.48941(−1)	0.16236(−3)	3.0000	1.44225
DM[12]	0.50000(0)	0.15659(−1)	0.10877(−5)	3.4075	1.84594
TM[41]	0.50000(0)	0.22068(−1)	0.12109(−5)	2.3993	1.54897
JM[15]	0.65000(0)	0.54988(−1)	0.89963(−9)	7.0000	1.91293
SLTKM[36]	0.50000(0)	0.48587(−4)	0.32385(−4)	7.2038	1.93132
TPM[38] $(a = b = 0)$	0.50000(0)	0.22661(−5)	0.12409(−44)	8.0000	1.68179
LAM[20]	0.50000(0)	0.32145(−6)	0.34397(−105)	15.5100	1.98451
SSSLM [29]	0.50000(0)	0.18741(−10)	0.23265(−170)	16.0000	1.74110
GKM [14]	0.50000(0)	0.71640(−9)	0.10752(−146)	16.0000	1.74110
TM7 (3.2) k=1	0.65000(0)	0.53539(−9)	0.11721(−11)	7.5233	1.95946
TM8 (3.2) k=4	0.53539(−1)	0.11721(−11)	0.40657(−89)	8.0047	2.01692
$f_4(x) = \frac{-5x^2}{2} + x^4 + x^5 + \frac{1}{1+x^2}, \alpha = 1, x_0 = 1.5, q_0 = \gamma_0 = \lambda_0 = \beta_0 = 0.1$					
AM [1]	0.50000(0)	0.18311(0)	0.33638(−1)	3.0000	1.44225
DM[12]	0.50000(0)	0.41826(0)	0.71739(−1)	3.5453	1.88290
TM[41]	0.50000(0)	0.41154(0)	0.13304(0)	2.2867	1.51218
JM[15]	0.50000(0)	0.22029(0)	0.15649(−2)	7.0000	1.91293
SLTKM[36]	0.50000(0)	0.31600(0)	0.42113(−2)	7.2133	1.93217
TPM[38] $(a = b = 0)$	0.50000(0)	0.52236(−1)	0.29347(−5)	8.0000	1.68179
LAM[20]	0.50000(0)	0.73848(−1)	0.60939(−12)	15.6030	1.98748
GKM [14]	0.50000(0)	0.10118(−1)	0.29389(−19)	16.0000	1.74110
SSSLM [29]	0.50000(0)	0.25125(−1)	0.16956(−15)	16.0000	1.74110
TM7 (3.2) k=1	0.50000(0)	0.22030(0)	0.65415(−4)	7.5145	1.95869
TM8 (3.2) k=4	0.22030(0)	0.65415(−4)	0.31283(−30)	8.0000	2.00000

5. Conclusion

In this work, Steffensen-Like adaptive with memory methods family was proposed to solve nonlinear equations. By using Newton's interpolation, the parameters of

Table 3. The numerical results of the proposed methods and other methods of with and without memory for $f_i(x), i = 5, 6, 7$.

$f_5(x) = \log(1+x^2) + e^{-3x+x^2}\sin(x), \alpha = 0, x_0 = 0.5, q_0 = \gamma_0 = \lambda_0 = \beta_0 = 0.1$					
Methods	$\lvert x_1-\alpha\rvert$	$\lvert x_2-\alpha\rvert$	$\lvert x_3-\alpha\rvert$	r_c	EI
AM [1]	0.50000(0)	0.88441(−4)	0.45840(−12)	3.0000	1.44225
DM[12]	0.50000(0)	0.64108(−1)	0.12721(−2)	3.5500	1.88414
TM[41]	0.50000(0)	0.42599(−1)	0.11207(−2)	2.4134	1.55351
JM[15]	0.50000(0)	0.28016(−1)	0.23124(−10)	7.0000	1.91293
SLTKM[36]	0.50000(0)	0.22780(−1)	0.13805(−10)	7.2390	1.93446
TPM[38] $(a = b = 0)$	0.50000(0)	0.54581(−2)	0.75773(−15)	8.0000	1.68179
LAM[20]	0.50000(0)	0.55075(−3)	0.30763(−48)	15.5080	1.98444
GKM [14]	0.50000(0)	0.46311(−6)	0.28713(−94)	16.0000	1.74110
SSSLM [29]	0.50000(0)	0.55428(−3)	0.10344(−42)	16.0000	1.74110
TM7 (3.2) k=1	0.50000(0)	0.25396(−1)	0.11699(−10)	7.5429	1.96116
TM8 (3.2) k=4	0.25396(−1)	0.11699(−10)	0.13293(−84)	8.0000	2.00000
$f_6(x) = x\log(1-\pi+x^2) - \frac{1+x^2}{1+x^3}\sin(x^2) + \tan(x^2), \alpha = \sqrt{\pi}, x_0 = 1.7, q_0 = \gamma_0 = \lambda_0 = \beta_0 = 0.1$					
Methods	$\lvert x_1-\alpha\rvert$	$\lvert x_2-\alpha\rvert$	$\lvert x_3-\alpha\rvert$	r_c	EI
AM [1]	0.72454(−1)	0.58079(−3)	0.33518(−9)	3.0000	1.44225
DM[12]	0.72454(−1)	0.10543(−1)	0.44438(−6)	3.5005	1.87096
TM[41]	0.72454(−1)	0.11486(−1)	0.28170(−5)	2.4090	1.55210
JM[15]	0.72454(−1)	0.19317(−3)	0.71968(−23)	7.0000	1.91293
SLTKM[36]	0.72454(−1)	0.19245(−3)	0.18418(−25)	7.5321	1.93205
TPM[38] $(a = b = 0)$	0.72454(−1)	0.21456(−8)	0.94711(−69)	8.0000	1.68179
GKM [14]	0.72454(−1)	0.43034(−19)	0.25205(−310)	16.0000	1.74110
LAM[20]	0.72454(−1)	0.27167(−6)	0.15849(−97)	15.5120	1.98457
SSSLM [29]	0.72454(−1)	0.11233(−14)	0.10052(−237)	16.0000	1.74110
TM7 (3.2) k=1	0.72454(−1)	0.70222(−5)	0.25789(−31)	7.4793	1.95563
TM8 (3.2) k=4	0.19245(−3)	0.18418(−25)	0.94704(−202)	8.0000	2.00000
$f_7(x) = x^3 + 4x^2 - 10, \alpha = 1.3652, x_0 = 1, q_0 = \gamma_0 = \lambda_0 = \beta_0 = 0.1$					
AM [1]	0.36520(0)	0.47568(−1)	0.22845(−4)	3.0000	1.44225
DM[12]	0.36520(0)	0.36340(0)	0.34044(−3)	3.7222	1.92930
TM[41]	0.36520(0)	0.27996(0)	0.64692(−2)	2.4053	1.55090
JM[15]	0.36520(0)	0.10860(0)	0.30014(−4)	7.0000	1.91293
SLTKM[36]	0.36520(0)	0.61026(0)	0.23768(−3)	8.0000	2.00000
TPM[38] $(a = b = 0)$	0.36520(0)	0.69070(−3)	0.30013(−4)	8.0000	1.68179
LAM[20]	0.36520(0)	0.75262(−3)	0.30013(−4)	16.0000	2.00000
GKM [14]	0.36520(0)	0.84783(−3)	0.30013(−4)	16.0013	1.74392
SSSLM [29]	0.36520(0)	0.19417(−1)	0.30013(−4)	16.0000	1.74110
TM7 (3.2) k=1	0.36500(0)	0.11034(0)	0.30013(−4)	8.0000	2.00000
TM8 (3.2) k=4	0.11034(0)	0.30013(−4)	0.30014(−4)	8.0000	2.00000

self-evaluation are interpolated. The numerical results show that proposed method is very useful to find an acceptable approximation of the exact solution for nonlinear equations, specially if the function is non-differentiable. The proposed adaptive method (31) was compared with the optimal n-point methods without memory. Therefore, we have developed a family iterative methods which have efficiency index 2. The efficiency index of the proposed method is $8^{\frac{1}{3}} = 2$ which is much better than one-,..., five-point optimal methods without memory and all the methods mentioned in the references[1–22, 24–38, 40–45], also with existing methods with- and without memory.

Table 4. Comparison efficiency index of proposed method by with and without memory methods.

without memory methods	EF	r_c	EI	with memory methods	EF	r_c	EI
AM[1]	3	3.000	1.442	CJM[5]	3	4.561	1.658
BCMTM[4]	3	4.000	1.587	CJM[5]	3	4.791	1.685
BAMM[3]	4	7.000	1.627	CJM[5]	3	5.000	1.710
BCMTM[3]	3	8.000	1.682	CJM[5]	4	9.582	1.759
CJM[5]	3	4.000	1.587	CJM[5]	4	9.795	1.769
CM[6]	3	4.000	1.587	CJM[5]	4	10.000	1.778
CNM[7]	4	8.000	1.682	DM[12]	2	3.550	1.884
CHMTM[10]	4	8.000	1.682	DM[12]	3	7.000	1.913
CHMTM[10]	3	4.000	1.587	JM[15]	3	7.000	1.913
CHMTM[9]	3	4.000	1.587	JM[15]	4	14.000	1.934
CHMTM[9]	4	8.000	1.682	KKBM[16]	3	7.000	1.913
CTVM[11]	4	8.000	1.682	PNPDM[27]	3	4.561	1.658
EM[13]	4	8.000	1.682	EM[13]	4	12.000	1.861
GKM[14]	5	16.000	1.741	LAM[20]	4	15.000	1.968
KM[18]	3	4.000	1.587	LAM[20]	4	15.500	1.984
KTM[19]	4	8.000	1.682	MLAM[22]	3	5.700	1.786
KTM[19]	3	4.000	1.587	MLAM[22]	3	5.950	1.812
MBMM[21]	4	8.000	1.682	PDPM[25]	3	4.450	1.645
MBMM[21]	5	16.000	1.741	PDPM[25]	3	5.000	1.710
NM[23]	2	2.000	1.414	PDPM[25]	3	5.370	1.751
OM[24]	3	4.000	1.587	PDPM[25]	3	6.000	1.817
KSM[17]	4	8.000	1.682	PNPDM[27]	4	10.131	1.784
SAM[31]	4	8.000	1.682	SGGM[34]	4	12.000	1.861
SKJM[35]	4	8.000	1.682	SLTKM[36]	3	7.230	1.934
RWBM[28]	3	4.000	1.587	SLTKM[36]	4	12.000	1.861
SSSLM[29]	5	16.000	1.741	TLFM[39]	2	4.000	2.000
SM[37]	3	2.000	1.414	TM[41]	2	2.410	1.482
TPM[38]	3	8.000	1.682	ZKFKAM[44]	3	7.944	1.995
SM[32]	3	8.000	1.682	WM[42]	3	4.230	1.617
ZLHM[45]	2	2.000	1.414	TM(3.2), k=1	3	7.530	1.960
ZLHM[45]	3	4.000	1.587	TM(3.2), k=2	3	7.944	1.995
ZLHM[45]	4	8.000	1.682	TM(3.2), k=3	3	7.993	1.999
ZLHM[45]	5	16.000	1.741	TM(3.2), k=4	3	8.000	2.000

References

[1] S. Abbasbandy, Modified homotopy perturbation method for nonlinear equations and comparison with Adomian decomposition method, Applied Mathematics and Computation, **172** (2006) 431-438.

[2] S. Amat, J. A. Ezquerro and M. A. Hernandez-Veron, On a Steffensen-like method for solving nonlinear equations, Calcolo, **53 (2)** (2016) 171-188.

[3] R. Behl, I. K. Argyros and S. S. Motsa, Improved Chebyshev-Halley family of methods with seventh and eighth order of convergence for simple roots, SeMA Journal, **74 (4)** (2017) 643-665.

[4] R. Behl, A. Cordero, S. S. Motsa and J. R. Torregrosa Sanchez, Construction of fourth-order optimal families of iterative methods and their dynamics, Applied Mathematics and Computation, **271** (2015) 89-101.

[5] N. Choubey and J. P. Jaiswal, Two- and three-moint with memory methods for solving nonlinear equations, Numerical Analysis and Applications, **10 (1)** (2017) 74-89.

[6] C. Chun, Iterative methods improving Newton's method by the decomposition method, Computers & Mathematics with Applications, **50** (2005) 1559-1568.

[7] C. Chun and B. Neta, An analysis of a new family of eighth-order optimal methods, Applied Mathematics and Computation, **245** (2014) 86-107.

[8] 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.

[9] A. Cordero, J. L. Hueso, E. Martinez and J. R. Torregrosa, A new technique to obtain derivative-free optimal iterative methods for solving nonlinear equations, Journal of Computational and Applied Mathematics, **252** (2013) 95-102.

[10] A. Cordero, J. L. Hueso, E. Martinez and J. R. Torregrosa, New modifications of Potra-Ptak's method with optimal fourth and eighth orders of convergence, Journal of Computational and Applied Mathematics, **234 (10)** (2010) 2969-2976.

[11] A. Cordero, J. R. Torregrosa and M. P. Vassileva, A family of modified Ostrowski's methods with optimal eighth order of convergence, Applied Mathematics Letters, **24** (2011) 2082-2086.

[12] J. Dzunic, On efficient two-parameter methods for solving nonlinear equations, Numerical Algorithms, **63** (2013) 549-569.

[13] T. Eftekhari, An efficient class of multipoint root-solvers with and without memory for nonlinear equations, Acta Mathematica Vietnamica, **41 (2)** (2016) 299-311.

[14] Y. H. Geum and Y. I. Kim, A biparametric family optimally convergent sixteenth-order multipoint methods with their fourth-step weighting function as a sum of a rational and a generic two-variable function, Journal of Computational and Applied Mathematics, **235** (2011) 3178-3188.

[15] J. P. Jaiswal, Two efficient bi-parametric derivative free with memory methods for finding simple roots nonlinear equations, Journal of Advances in Applied Mathematics, **1 (4)** (2016) 203-210.

[16] M. Kansal, V. Kanwar and S. Bhatia, Efficient derivative-free variants of Hansen-Patrick's family with memory for solving nonlinear equations, Numerical Algorithms, **73** (2016) 1017-1036.

[17] S. K. Khattri and T. Steihaug, Algorithm for forming derivative-free optimal methods, Numerical Algorithms, **65 (4)** (2013) 809-824.

[18] R. F. King, A family of fourth order methods for nonlinear equations, SIAM Journal on Numerical Analysis, **10 (5)** (1973) 876-879.

[19] H. T. Kung and J. F. Traub, Optimal order of one-point and multipoint iteration, Journal of the Association for Computing Machinery, **21 (4)** (1974) 643-651.

[20] T. Lotfi and P. Assari, New three-and four-parametric iterative with memory methods with efficiency index near 2, Applied Mathematics and Computation, **270** (2015) 1004-1010.

[21] P. Maroju, R. Behl and S. S. Motsa, Some novel and optimal families of King's method with eighth and sixteenth-order of convergence, Journal of Computational and Applied Mathematics, **318** (2017) 136-148.

[22] M. Mohamadi Zadeh, T. Lotfi and M. Amirfakhrian, Developing two efficient adaptive Newton-type methods with memory, Mathematical Methods in the Applied Sciences, **42** (2019) 5687-5695.

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

[24] A. M. Ostrowski, Solution of Equations and Systems of Equations, Academic Press, New York, (1960).

[25] M. S. Petkovic, J. Dzunic and L. D. Petkovic, A family of two-point with memory for solving nonlinear equations, Applicable Analysis and Discrete Mathematics, **5** (2011) 298-317.

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

[27] M. S. Petkovic, B. Neta, L. D. Petkovic and J. Dzunic, Multipoint methods for solving nonlinear equations: A survey, Applied Mathematics and Computation, **226** (2014) 635-660.

[28] H. Ren, Q. Wu and W. Bi, A class of two-step Steffensen type methods with fourth-order convergence, Applied Mathematics and Computation, **209** (2009) 206-210.

[29] 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, Mathematics and Computers in Simulation, **119 (c)** (2016) 69-90.

[30] S. Sharifi, S. Siegmund and M.Salimi, Solving nonlinear equations by a derivative-free form of the Kings family with memory, Calcolo, **53 (2)** (2016) 201-215.

[31] J. Raj. Sharma and H. Arora, An efficient family of weighted-Newton methods with optimal eighth order convergence, Applied Mathematics Letters, **29** (2014) 1-6.

[32] J. R. Sharma, Improved Chebyshev-Halley methods with sixth and eighth order convergence, Applied Mathematics and Computation, **256** (2015) 119-124.

[33] J. R. Sharma, H. Arora, Efficient Jarratt-like methods for solving systems of nonlinear equations, Calcolo, **51 (1)** (2014) 193-210.

[34] J. R. Sharma, R. K. Guha and P. Gupta, Some efficient derivative free methods with memory for solving nonlinear equations, Applied Mathematics and Computation, **219** (2012) 699-707.

[35] F. Soleymani, S. Karimi Vanani and M. Jamali Paghaleh, A class of three-step derivative-free root solvers with optimal convergence order, Journal of Applied Mathematics, **2012** (2012) 1-15.

[36] F. Soleymani, T. Lotfi, E. Tavakoli and F. Khaksar Haghani, Several iterative methods with memory using self-accelerators, Applied Mathematics and Computation, **254** (2015) 452-458.

[37] J. F. Steffensen, Remarks on iteration, Scandinavian Aktuarietidskr, **16** (1933) 64-72.

[38] R. Thukral and M. S. Petkovic, A family of three-point methods of optimal order for solving nonlinear equations, Journal of Computational and Applied Mathematics, **233** (2010) 2278-2284.

[39] V. Torkashvand, T. Lotfi and M. A. Fariborzi Araghi, A new family of adaptive methods with memory for solving nonlinear equations, Mathematical Sciences, **13** (2019) 1-20.

[40] V. Torkashvand, T. Lotfi and M. A. Fariborzi Araghi, Efficient iterative methods with memory for solving nonlinear equations, 49^{th} Annual Iranian Mathematics Conference, Tehran, Iran, August 2018, 1-6.

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

[42] X. Wang, An Ostrowski-type method with memory using a novel self-accelerating parameter, Journal of Computational and Applied Mathematics, **330** (2017) 1-18.

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

[44] M. Zaka Ullah, S. Kosari, F. Soleymani, F. Khaksar Haghani and A. S. Al-Fhaid, A super-fast tri-parametric iterative method with memory, Applied Mathematics and Computation, **289** (2016) 486-491.

[45] Q. Zheng, J. Li and F. Huang, An optimal Steffensen-type family for solving nonlinear equations, Applied Mathematics and Computation, **217** (2011) 9592-9597.