AUT Journal of Mathematics and Computing

AUT J. Math. Com., 6(1) (2024) 77-89
https://doi.org/10.22060/AJMC.2023.22090.1132

Original Article

Interpolatory four-parametric adaptive method with memory for solving nonlinear equations

Vali Torkashvand*[a]

[a]*Lecturer at Farhangian University, Shahid Beheshti Higher Education Center, Tehran, Iran*

ABSTRACT: The adaptive technique enables us to achieve the highest efficiency index theoretically and practically. The idea of introducing an adaptive self-accelerator (via all the old information for Steffensen-type methods) is new and efficient to obtain the highest efficiency index. In this work, we have used four self-accelerating parameters and have increased the order of convergence from 8 to 16. I.e. any new function evaluations the convergence order improve up to 100%. The numerical results are compared without and with memory methods. It confirms that the proposed methods have more efficiency index..

Review History:

Received:10 January 2023
Revised:14 September 2023
Accepted:20 September 2023
Available Online:01 July 2024

Keywords:

Nonlinear equations
Adaptive method with memory
R-order convergence
Self accelerating parameter

MSC (2020):

1. Introduction

1.1. Literature

Finding the root of a nonlinear equation can be found in many fields of science and engineering. A root-finding algorithm is a numerical method or algorithm to find a value α that is $f(\alpha) = 0$, for a given function f. Such an α is called a root of the function f. Recently, researchers have developed many iterative methods to solve the equation $f(x) = 0$. In general, it can not get an analytical solution for nonlinear problems. Hence, numerical iterative methods are best suited for the purpose. These methods classify one-step and multi-step methods [35]. Also, another division of the iterative method was performed by Kung and Traub: without memory and with memory methods [13]. The multi-step methods have higher convergence order than single-step methods. Other ways to increase the convergence rate are using the weight function and the self-accelerating parameters. The self-accelerating parameters play serious role in providing a without-memory method into the with-memory, and increasing the order of convergence. Researchers approximated the self-accelerating parameters by the method Secant, the Hermite interpolating polynomial, and Newton's interpolation polynomial. In all of them, Newton's method reaches the highest convergence degree.

**Corresponding author.*
E-mail addresses: torkashvand1978@gmail.com

1.2. Motivation and organization

Traub is known as the leader of the memory methods. The with-memory methods have an efficiency index higher than the without-memory methods. Many people worked on these methods. They used self-accelerating parameters without memory methods to find maximum improvement in convergence order. The author has achieved an improved convergence of 100% and has started with three-step four-free parameters optimal order 8, which Cordero et al. [7] proposed. Using information previous arrive at convergence order 16 and the efficiency index 2. This paper is organized as follows: Section 2 deals with modifying the optimal three-point methods with memory introduced by Cordero et al. [7]. Section 3 includes the main contribution and the novel adaptive with memory method. In this section, details of the proposed scheme are discussed. The theoretical results and the excellent convergence properties of the presented methods are compared with some without- and with-memory methods in Section 4. Some nonlinear equations are be solved that show the applicability and competitiveness of the proposed methods.

1.3. Definitions

Definition 1.1. *The efficiency of the iterative methods called efficiency index as follows: $E(p,m) = p^{\frac{1}{m}}$ where p and m are convergence order and functional evaluations, respectively [21]. Therefore, any optimal n-point method without memory has the efficiency index $E[2^m, m+1] = 2^{\frac{m}{m+1}}$. The efficiency index of the optimal multipoint method cannot exceed 2 because $\lim_{m\to\infty} E[2^m, m+1] = 2$. Also, any optimal three-point method without memory has the same efficiency index, say, $E[8,4] = 8^{\frac{1}{4}} \approx 1.681$.*

Definition 1.2. *Multi-point iterative method with memory: In this classification, allow us to define another iteration function ϕ having opinions z_j, where each such argument represents $k+1$ quantities $x_j, w_1(x_j), \ldots, w_n(x_j)$, $(n \geq 1)$. Let the iteration mapping be defined by $x_{k+1} = \phi(z_k; z_{k-1}, \ldots, z_{k-n})$. Then ϕ is named a multipoint IF with memory. In the above-mentioned mapping, a semicolon splits the points at which new information is used from the point at which old information is reused, i.e., at each iterative step we must conserve information of the last n approximations x_j and for each approximation, we must compute n expressions $w_1(x_j), \ldots, w_n(x_j)$.*

1.4. Existing iterative methods

Zheng et al.'s method [38], denoted by ZLHM, has the iterative expression:

$$\begin{cases} z_k = x_k + \gamma f(x_k), \quad y_k = x_k - \dfrac{f(x_k)}{f[x_k, z_k]}, \quad k = 0,1,2,\ldots, \\ u_k = y_k - \dfrac{f(y_k)}{f[y_k, x_k] + f[y_k, x_k, z_k](y_k - x_k)}, \quad \gamma_0 = 1, \\ x_{k+1} = u_k - \dfrac{f(u_k)}{f[u_k, y_k] + f[u_k, y_k, x_k](u_k - y_k) + f[u_k, y_k, x_k, z_k](u_k - y_k)(u_k - x_k)}. \end{cases} \tag{1}$$

Cordero et al.'s method, see [5], denoted by CLMTM, has the iterative expression:

$$\begin{cases} y_k = x_k - \dfrac{f(x_k)}{f'(x_k)}, \ k = 0,1,2,\ldots, \\ u_k = x_k - (1 + \dfrac{f(y_k)}{f(x_k) - 2\beta f(y_k)})\dfrac{f(x_k)}{f'(x_k)}, \quad \beta_0 = 1, \\ x_{k+1} = u_k - \dfrac{f(u_k)}{f[u_k, y_k] + 2(u_k - y_k)f[u_k, y_k, x_k] + f[y_k, x_k, x_k]}. \end{cases} \tag{2}$$

Sharma-Arora's method, see [27], denoted by SAM, has the iterative expression:

$$\begin{cases} y_k = x_k - \dfrac{f(x_k)}{f'(x_k)}, \ k = 0,1,2,\ldots, \\ u_k = x_k - \dfrac{f(y_k)}{2f[y_k, x_k] - f'(x_k)}, \\ x_{k+1} = u_k - \dfrac{f'(x_k) - f[y_k, x_k] + f[u_k, y_k]}{2f[u_k, y_k] - f[u_k, x_k]}\dfrac{f(u_k)}{f'(x_k)}. \end{cases} \tag{3}$$

These three methods have the order of convergence eight and are optimal schemes according to Kung-Traub's conjecture, the without memory method. Also, there are four-step methods with the order of convergence 16 listed in relations (4) and (5). Geum-Kim's method, see [11], denoted by GKM, has the iterative expression:

$$\begin{cases} y_k = x_k - \dfrac{f(x_k)}{f'(x_k)},\ u_k = \dfrac{f(y_k)}{f(x_k)},\ m_k = \dfrac{(1+\beta u_k - \frac{\beta}{2})u_k^2}{1+(\beta-2)u_k-(1+\frac{5\beta}{2})u_k^2},\ k=0,1,2,\ldots, \\ z_k = y_k - m_k\dfrac{f(y_k)}{f'(x_k)},\ v_k = \dfrac{f(z_k)}{f(y_k)},\ w_k = \dfrac{f(z_k)}{f(x_k)},\ h_k = \dfrac{1-u_k-\frac{3}{2}v_k-\frac{5}{2}w_k}{1-3u_k-\frac{5}{2}v_k+\frac{3}{2}w_k}, \\ p_k = \dfrac{1-u_k-\frac{3}{2}v_k-3w_k+\frac{3}{2}t_k-\frac{13}{4}v_kw_k+\frac{3}{4}v_k^3-\frac{1}{4}(\beta^2-\beta+8)v_ku_k^4-\frac{3}{2}t_ku_k^2+\lambda u_kw_k^2}{1-3u_k-\frac{5}{2}v_k-3w_k+\frac{1}{2}t_k-\frac{19}{4}v_kw_k-\frac{3}{4}v_k^3-\frac{1}{4}(\beta^2-3\beta+8)v_ku_k^4-\frac{9}{2}t_ku_k^2+(\frac{27}{2}-\lambda)u_kw_k^2} \\ s_k = z_k - h_k\dfrac{f(z_k)}{f'(x_k)},\ t_k = \dfrac{f(s_k)}{f(z_k)},\ x_{k+1} = s_k - p_k\dfrac{f(s_k)}{f'(x_k)}.\ \beta_0=\lambda_0=1, \end{cases} \tag{4}$$

Sharifi et al.'s method, see [26], denoted by SSSLM, has the iterative expression:

$$\begin{cases} y_k = x_k - \dfrac{f(x_k)}{f'(x_k)},\ g_k = t_k^2(5-7t_k)+(2t_k+1)(t_k^3+1)-t_k^4,\ k=0,1,2,\ldots, \\ z_k = y_k - g_k\dfrac{f(y_k)}{f'(x_k)},\ s_k = \dfrac{f(z_k)}{f(y_k)},\ h_k = 1-s_k+(6+u_k^2)(u_k+t_k^2)+2(t_k-u_k), \\ w_k = z_k - h_k\dfrac{f(z_k)}{f'(x_k)},\ p_k = \dfrac{f(w_k)}{f(x_k)},\ q_k = \dfrac{f(w_k)}{f(y_k)},\ V_k = (1+t_k)(2t_k+t_k^2)+t_k^2(3-t_k), \\ J_k = \dfrac{s_k+s_k^2-s_k^3}{1+s_k},\ M_k = \dfrac{1+5u_k}{1+u_k},\ L_k = t_ku_k+6t_k^2u_k+\dfrac{2t_k^3u_k-10t_ku_k^2}{1+t_ku_k},\ r_k = \dfrac{f(w_k)}{f(z_k)}, \\ F_k = 8t_k^2r_k-4s_k^2u_k-2t_k^3r_k+\dfrac{2s_ku_k+2t_kr_k+24t_k^2u_k+2t_ks_ku_k}{1+t_k},\ N_k = 2(p_k+q_k)+\dfrac{6p_k+r_k}{1+p_k}, \\ v_k = \dfrac{f(z_k)}{f(x_k)},\ t_k = \dfrac{f(y_k)}{f(x_k)},\ x_{k+1} = w_k-(V_k+J_k+M_k+-N_k+L_k+F_k)\dfrac{f(w_k)}{f'(x_k)}. \end{cases} \tag{5}$$

The method of Lotfi-Assari's method [15](LAM)

$$\begin{cases} \gamma_k = -\dfrac{1}{N_4'(x_k)},\ q_k = -\dfrac{N_5''(w_k)}{2N_5'(w_k)},\ \lambda_k = \dfrac{N_6'''(y_k)}{6},\ \beta_k = \dfrac{N_7''''(z_k)}{24},\ k=1,2,3,\ldots, \\ w_k = x_k+\gamma_kf(x_k),\ y_k = x_k-\dfrac{f(x_k)}{f[x_k,w_k]+q_kf(w_k)},\ k=0,1,2,\ldots, \\ z_k = y_k-\dfrac{f(y_k)}{f[y_k,x_k]+f[w_k,x_k,y_k](y_k-x_k)+\lambda_k(y_k-x_k)(y_k-w_k)}, \\ x_{k+1} = z_k-\frac{f(z_k)}{f[x_k,z_k]+(f[w_k,x_k,y_k]-f[w_k,x_k,z_k]-f[y_k,x_k,z_k])(x_k-z_k)+\beta_k(z_k-y_k)(z_k-x_k)(z_k-w_k)}. \end{cases} \tag{6}$$

Soleymani et al.'s method [30](SLTKM)

$$\begin{cases} \gamma_k = -\dfrac{1}{N_4'(x_k)},\ k=1,2,3,\ldots, \\ w_k = x_k+\gamma_kf(x_k),\ y_k = x_k-\dfrac{f(x_k)}{f[x_k,w_k]},\ t_k = \dfrac{f(y_k)}{f(x_k)},\ H_k = 1+t_k+t_k^2,\ k=0,1,2,\ldots, \\ z_k = y_k-H_k\dfrac{f(y_k)}{f[y_k,s_k]} = \dfrac{f(y_k)}{f(w_k)},\ V_k = (1+(1+\gamma f[x_k,w_k])s_k^2, \\ v_k = \dfrac{f(z_k)}{f(w_k)},\ J_k = (2+\gamma f[x_k,w_k])v_k,\ x_{k+1} = z_k-(V_k+J_k)\dfrac{f(z_k)}{f[z_k,y_k]}. \end{cases} \tag{7}$$

Jaiswal's method [10](JM)

$$\begin{cases} \gamma_k = -\dfrac{1}{N_4'(x_k)},\ p_k = -\dfrac{N_5''(w_k)}{2N_5'(w_k)}, \quad k = 1, 2, 3, \dots, \\ w_k = x_k + \gamma_k f(x_k),\ y_k = x_k - \dfrac{f(x_k)}{f[x_k, w_k] + p_k f(w_k)},\ b_k = f(u_k),\ k = 0, 1, 2, \dots, \\ u_k = \dfrac{f(y_k) f[x_k, w_k]}{f[x_k, y_k] f[y_k, w_k]},\ c_k = \dfrac{f[y_k, u_k, x_k] - f[y_k, u_k, w_k]}{f[y_k, w_k] - f[y_k, x_k]}, d_k = f[y_k, u_k, w_k] + c_k f[y_k, w_k], \\ s_k = f[y_k, u_k] - d_k(y_k - u_k), + f(y_k)c_k,\ x_{k+1} = u_k - \dfrac{f(u_k)}{s_k - b_k c_k}. \end{cases} \tag{8}$$

2. Derivative methods and analysis of convergence

Recently, Cordero et al. established the following optimal iterative method without memory [7]

$$\begin{cases} y_k = x_k - \dfrac{f(x_k)}{f[x_k, w_k] + \beta f(w_k)},\ k = 0, 1, 2, \dots, \\ z_k = y_k - A(u_k) * g(u_k) * \dfrac{f(y_k)}{f[y_k, w_k] + \beta f(w_k) + \gamma (y_k - x_k)(y_k - w_k)}, \\ x_{k+1} = z_k - \frac{f(z_k)}{f[z_k, y_k] + f[z_k, y_k, x_k](z_k - y_k) + f[z_k, y_k, x_k, w_k](z_k - y_k)(z_k - x_k) + \lambda (z_k - y_k)(z_k - x_k)(z_k - w_k)}. \end{cases} \tag{9}$$

where $w_k = x_k + \theta f(x_k)$, $u_k = \frac{f(y_k)}{f(x_k)}$. This optimal method without memory uses four function evaluations per iteration, and has convergence order 8 when the weight functions satisfy the following conditions.

$$\begin{cases} A(0) = 1, \quad A'(0) = 2, \quad A''(0) < \infty, \\ g(0) = 1, \quad g'(0) = -1, \quad g''(0) < \infty, \end{cases} \tag{10}$$

Among the weight functions that apply to these conditions are:

$$\begin{cases} A_1(u_k) = 1 + 2u_k, \quad A_2(u_k) = \dfrac{1}{1 - 2u_k}, \quad A_3(u_k) = 1 + 2\sin(u_k), \dots, \\ g_1(u_k) = 1 - u_k, \quad g_2(u_k) = e^{-u_k}, \quad g_3(u_k) = \cos(u_k) - u_k, \dots \end{cases} \tag{11}$$

The error equation under the conditions of the weight functions $A_2(u_k)$ and $g_1(u_k)$ is as follows.

Theorem 2.1 ([7])**.** *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 (9) is given by:*

$$\begin{aligned} e_{k+1} = & (1 + \theta f'(\alpha))^4 (\beta + c_2)^2 \Big(-\gamma + \beta^2 f'(\alpha)(1 + \theta f'(\alpha)) + f'(\alpha)\big(2 f'(\alpha)\theta\beta c_2 + (-1 + \theta f'(\alpha))c_2^2 + c_3\big) \Big) \\ & \times \Big(\lambda + c_2(-\gamma + f'(\alpha)\beta^2(1 + \theta f'(\alpha)) + f'(\alpha)(2 f'(\alpha)\beta\theta c_2 + (-1 + \theta f'(\alpha))c_2^2 + c_3)) - f'(\alpha)c_4 \Big) \\ & \times f'(\alpha)^{-2} e_k^8 + O(e_k^9), \end{aligned} \tag{12}$$

where γ, β, λ and θ are nonzero arbitrary parameters.

Proof. In the reference [7], the proof is fully described. □

They increased the order of convergence by using relationships

$$\begin{cases} (1 + \theta f'(\alpha)) = 0, \\ (\beta + c_2) = 0, \\ (-\gamma + \beta^2 f'(\alpha)(1 + \theta f'(\alpha)) + f'(\alpha)(2 + \theta\beta c_2 f'(\alpha) + (-1 + f'(\alpha))c_2^2 + c_3)) = 0, \\ \Big(\lambda + c_2(-\gamma + \beta^2 f'(\alpha)(1 + \theta + f'(\alpha)) + f'(\alpha)(2\theta\beta c_2 f'(\alpha) + (-1 + \theta f'(\alpha))c_2^2 + c_3)) - f'(\alpha)c_4 \Big) = 0, \end{cases} \tag{13}$$

and arrived at a memory method. By interpolating the free quadruple(γ, β, λ and θ), they eventually arrived at the following with memory a convergence rate of 15.5.

$$\begin{cases} \theta_k = -\dfrac{1}{N_4'(x_k)},\ \beta_k = -\dfrac{N_5''(w_k)}{2N_5'(w_k)}, \gamma_k = \dfrac{N_6'''(y_k)}{6},\ \lambda_k = \dfrac{N_7''''(z_k)}{24},\ k = 1, 2, 3, \ldots, \\ w_k = x_k + \theta_k f(x_k),\ y_k = x_k - \dfrac{f(x_k)}{f[x_k, w_k] + \beta_k f(w_k)},\ k = 0, 1, 2, \ldots, \\ z_k = y_k - A_2(u_k).g_1(u_k)\dfrac{f(y_k)}{f[y_k, w_k] + \beta_k f(w_k) + \gamma_k(y_k - x_k)(y_k - w_k)}, \\ x_{k+1} = z_k - \frac{f(z_k)}{f[z_k,y_k]+f[z_k,y_k,x_k](z_k-y_k)+f[z_k,y_k,x_k,w_k](z_k-y_k)(z_k-x_k)+\lambda_k(z_k-y_k)(z_k-x_k)(z_k-w_k)}. \end{cases} \tag{14}$$

Motivated by this paper, one natural question raised in our mind. Is it possible to improve more efficiency index using the same number of function evaluations? Author emphasizes that although this method is with memory, it is not adaptive yet. It uses information from the last two iterations. To extend it to an adaptive with memory method, they will be updated $\theta_k, \beta_k, \gamma_k$, and λ_k based on all the available information from the first iteration to the current iteration. In the next section, It will be introduced a new adaptive method with memory.

3. Further accelerations via with memory concept

This section introduces a new efficient adaptive method with memory. Author continues as before and develop a three-step with memory method with the best efficiency index. Indeed, it will be achieved the efficiency index 2. Hence

$$\begin{cases} \theta_k = -\dfrac{1}{N_4'(x_k)} \simeq \dfrac{-1}{f'(\alpha)}, \\ \beta_k = -\dfrac{N_5''(w_k)}{2N_5'(w_k)} \simeq -\dfrac{f''(\alpha)}{2f'(\alpha)}, \\ \gamma_k = \dfrac{N_6'''(y_k)}{6} \simeq f'(\alpha)c_3 = \dfrac{f'''(\alpha)}{6}, \\ \lambda_k = \dfrac{N_7'''(z_k)}{24} \simeq f'(\alpha)c_4 = \dfrac{f''''(\alpha)}{24}. \end{cases} \tag{15}$$

where $N_4'(x_k), N_5''(w_k), N_6'''(y_k)$ and $N_7''''(z_k)$ are Newton's interpolation polynomials obtained from the nodes. To construct a recursive adaptive method with memory, I have used 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-accelerator γ_k, q_k, λ_k and β_k are obtained. I have developed the following recursive adaptive method with memory. Let $w_0 = x_0 + \gamma_0 f(x_0), x_0, \gamma_0, q_0, \lambda_0$ and β_0 be given suitably. Then:

$$\begin{cases} \theta_k = -\dfrac{1}{N_{4k}'(x_k)}, \beta_k = -\dfrac{N_{4k+1}''(w_k)}{2N_{4k+1}'(w_k)}, \gamma_k = \dfrac{N_{4k+2}'''(y_k)}{6}, \lambda_k = \dfrac{N_{4k+3}''''(z_k)}{24}, k = 1, 2, 3, \ldots, \\ w_k = x_k + \theta_k f(x_k),\ y_k = x_k - \dfrac{f(x_k)}{f[x_k, w_k] + \beta_k f(w_k)},\ k = 0, 1, 2, \ldots, \\ z_k = y_k - A_2(u_k).g_1(u_k)\dfrac{f(y_k)}{f[y_k, w_k] + \beta_k f(w_k) + \gamma_k(y_k - x_k)(y_k - w_k)}, \\ x_{k+1} = z_k - \frac{f(z_k)}{f[z_k,y_k]+f[z_k,y_k,x_k](z_k-y_k)+f[z_k,y_k,x_k,w_k](z_k-y_k)(z_k-x_k)+\lambda_k(z_k-y_k)(z_k-x_k)(z_k-w_k)}. \end{cases} \tag{16}$$

In what follows, It is discussed the general convergence analysis of the recursive adaptive method with memory (16). It should be noted that the convergence order varies as the iteration proceeds. The following lemma is needed.

Lemma 3.1. *If* $\theta_k = -\frac{1}{N_{4k}'(x_k)}$, $\beta_k = -\frac{N_{4k+1}''(w_k)}{2N_{4k+1}'(w_k)}$, $\gamma_k = \frac{N_{4k+2}'''(y_k)}{6}$ *and* $\lambda_k = \frac{N_{4k+3}''''(z_k)}{24}$, *then*

$$(1 + \theta_k f'(\alpha)) \sim \prod_{s=0}^{k-1} e_s e_{s,w} e_{s,y} e_{s,z}, \tag{17}$$

$$(\beta_k + c_2) \sim \prod_{s=0}^{k-1} e_s e_{s,w} e_{s,y} e_{s,z}, \tag{18}$$

$$\big(-\gamma_k + \beta_k^2 f'(\alpha)(1+\theta_k f'(\alpha))\big) + f'(\alpha)\big(2 + \theta_k \beta_k c_2 f'(\alpha) + (-1 + f'(\alpha))c_2^2 + c_3)\big) \sim \prod_{s=0}^{k-1} e_s e_{s,w} e_{s,y} e_{s,z} \tag{19}$$

$$\Big(\lambda_k + c_2\big(-\gamma_k + \beta_k^2 f'(\alpha)\big(1+\theta_k + f'(\alpha)\big) + f'(\alpha)(2\theta_k\beta_k c_2 f'(\alpha) + (-1+\theta_k f'(\alpha))c_2^2 + c_3)\big) - f'(\alpha)c_4\Big)$$
$$\sim \prod_{s=0}^{k-1} e_s e_{s,w} e_{s,y} e_{s,z}, \tag{20}$$

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

Proof. The proof is similar to Lemma 3.1 in [33]. □

Theorem 3.2. *Let x_0 be a suitable initial guess to the simple root α of $f(x) = 0$. Also, suppose the initial values $\theta_0, beta_0, \gamma_0,$ and λ_0 are chosen appropriately. Then the R-order of the recursive adaptive method with memory (16) can be obtained from the following system of nonlinear equations:*

$$\begin{cases} r^k r_1 - (1 + r_1 + r_2 + r_3) \sum\limits_{i=0}^{k-1} r^i - r^k = 0, \\ r^k r_2 - 2(1 + r_1 + r_2 + r_3) \sum\limits_{i=0}^{k-1} r^i - 2r^k = 0, \\ r^k r_3 - 4(1 + r_1 + r_2 + r_3) \sum\limits_{i=0}^{k-1} r^i - 4r^k = 0, \\ r^{k+1} - 8(1 + r_1 + r_2 + r_3) \sum\limits_{i=0}^{k-1} r^i - 8r^k = 0, \end{cases} \tag{21}$$

where r, r_1, r_2 and r_3 are the order of convergence of the sequences $\{x_k\}, \{w_k\}, \{y_k\}$, and $\{z_k\}$, respectively. Also, k indicates the number of iterations.

Proof. Let $\{x_k\}, \{w_k\}, \{y_k\}$ and $\{z_k\}$ be convergent with orders r, r_1, r_2 and r_3 respectively. Then:

$$\begin{cases} e_{k+1} \sim e_k^r \sim e_{k-1}^{r^2} \sim \dots \sim e_0^{r^{k+1}}, \\ e_{k,w} \sim e_k^{r_1} \sim e_{k-1}^{rr_1} \sim \dots \sim e_0^{r_1 r^k}, \\ e_{k,y} \sim e_k^{r_2} \sim e_{k-1}^{rr_2} \sim \dots \sim e_0^{r_2 r^k}, \\ e_{k,z} \sim e_k^{r_3} \sim e_{k-1}^{rr_3} \sim \dots \sim e_0^{r_3 r^k}, \end{cases} \tag{22}$$

where $e_k = x_k - \alpha$, $e_{k,w} = w_k - \alpha$, $e_{k,y} = y_k - \alpha$ and $e_{k,z} = z_k - \alpha$. Now, by Lemma 3.1 and Eq. (22), obtained

$$\begin{aligned} (1 + \theta_k f'(\alpha)) \sim \prod_{s=0}^{k-1} e_s e_{s,w} e_{s,y} e_{s,z} &= (e_0 e_{0,w} e_{0,y} e_{0,z}) \dots (e_{k-1} e_{k-1,w} e_{k-1,y} e_{k-1,z}) \\ &= (e_0 e_0^{r_1} e_0^{r_2} e_0^{r_3})(e_0^{r} e_0^{r_1 r} e_0^{r_2 r} e_0^{r_3 r}) \dots (e_0^{r^{k-1}} e_0^{r^{k-1} r_1} e_0^{r^{k-1} r_2} e_0^{r^{k-1} r_3}) \\ &= e_0^{(1+r_1+r_2+r_3) + (1+r_1+r_2+r_3)r + \dots + (1+r_1+r_2+r_3)r^{k-1}} \\ &= e_0^{(1+r_1+r_2+r_3)\sum_{i=0}^{k-1} r^i}. \end{aligned} \tag{23}$$

$\sum\limits_{i=1}^{N} c_i e_i$ Similarly, it will be get

$$(\beta_k + c_2) \sim e_0^{(1+r_1+r_2+r_3)\sum_{i=0}^{k-1} r^i}, \tag{24}$$

and

$$(-\gamma_k + \beta_k^2 f'(\alpha)(1 + \theta_k f'(\alpha)) + f'(\alpha)(2 + \theta_k \beta_k c_2 f'(\alpha) + (-1 + f'(\alpha))c_2^2 + c_3)) \sim e_0^{(1+r_1+r_2+r_3)\sum_{i=0}^{k-1} r^i}, \tag{25}$$

$$\Big(\lambda_k + c_2\big(-\gamma_k + \beta_k^2 f'(\alpha)(1+\theta_k+f'(\alpha)) + f'(\alpha)(2\theta_k\beta_k c_2 f'(\alpha) + (-1+\theta_k f'(\alpha))c_2^2 + c_3)\big) - f'(\alpha)c_4\Big)$$
$$\sim e_0^{(1+r_1+r_2+r_3)\sum_{i=0}^{k-1} r^i}. \tag{26}$$

By considering the errors of w_k, y_k, z_k and x_{k+1} in Eq. (22) and Eqs. (23)-(26).
It will be concluded:

$$e_{k,w} \sim (1+\theta_k f'(\alpha))e_k \sim e_0^{(1+r_1+r_2+r_3)\sum_{i=0}^{k-1} r^i} e_0^{r^k}, \tag{27}$$

$$e_{k,y} \sim (1+\theta_k f'(\alpha))(\beta_k + c_2)e_k^2 \sim e_0^{((1+r_1+r_2+r_3)\sum_{i=0}^{k-1} r^i)^2} e_0^{2r^k}, \tag{28}$$

$$\begin{aligned} e_{k,z} &\sim f'(\alpha)^{-1}(1+\theta_k f'(\alpha))^2(\beta_k+c_2)(-\gamma_k + \beta_k^2 f'(\alpha)(1+\theta_k f'(\alpha)) \\ &\quad + f'(\alpha)(2\beta_k\theta_k c_2 f'(\alpha) + (-1+\theta_k f'(\alpha))c_2^2 + c_3)e_k^4 \sim e_0^{((1+r_1+r_2+r_3)\sum_{i=0}^{k-1} r^i)^4} e_0^{4r^k}, \end{aligned} \tag{29}$$

$$\begin{aligned} e_{k+1} &\sim f'(\alpha)^{-2}\big(1+\theta_k f'(\alpha)\big)^4\Big(\beta_k + c_2\Big)^2\Big(-\gamma_k + \beta_k^2(1+\theta_k f'(\alpha))\big(2 + \beta_k\theta_k c_2 f'(\alpha) + (-1+\theta_k f'(\alpha))c_2^2 + c_3\big)\Big) \\ &\quad \times \Big(\lambda_k + c_2(-\gamma_k + \beta_k^2 f'(\alpha)(1+\theta_k f'(\alpha)) + f'(\alpha)(2+\beta_k\theta_k c_2 f'(\alpha) + (-1+\theta_k f'(\alpha))c_2^2 + c_3)) - f'(\alpha)c_4\Big)e_k^8 \\ &\sim e_0^{((1+r_1+r_2+r_3)\sum_{i=0}^{k-1} r^i)^8} e_0^{8r^k}. \end{aligned} \tag{30}$$

To obtain the desired result, it is enough to match the right-hand-side of the Eqs. (22), (27), (28), (29) and (30). Then:

$$\begin{cases} r^k r_1 - (1+r_1+r_2+r_3)\sum_{i=0}^{k-1} r^i - r^k = 0, \\ r^k r_2 - 2(1+r_1+r_2+r_3)\sum_{i=0}^{k-1} r^i - 2r^k = 0, \\ r^k r_3 - 4(1+r_1+r_2+r_3)\sum_{i=0}^{k-1} r^i - 4r^k = 0, \\ r^{k+1} - 8(1+r_1+r_2+r_3)\sum_{i=0}^{k-1} r^i - 8r^k = 0. \end{cases}$$

This completes the proof of the theorem. □

Corollary 3.3. *For $k = 1$, it will be used the information from the current and the one previous steps In this case, the order of convergence for with memory method can be computed from the following system*

$$\begin{cases} rr_1 - (1+r_1+r_2+r_3) - r = 0, \\ rr_2 - 2(1+r_1+r_2+r_3) - 2r = 0, \\ rr_3 - 4(1+r_1+r_2+r_3) - 4r = 0, \\ r^2 - 8(1+r_1+r_2+r_3) - 8r = 0. \end{cases} \tag{31}$$

This system of equations has the solution:

$$r_1 = \frac{1}{16}(15+\sqrt{257}) \simeq 1.93945, \quad r_2 = \frac{1}{8}(15+\sqrt{257}) \simeq 3.8789, \quad r_3 = \frac{1}{4}(15+\sqrt{257}) \simeq 7.7578$$

and

$$r = \frac{1}{2}(15+\sqrt{257}) \simeq 15.5156.$$

This special case gives the given result by Cordero et al. (shown by CJTYZM) [7]. For $k = 2$, it will be obtained the order of convergence: $r_1 \simeq 1.99632$, $r_2 \simeq 3.99265, r_3 \simeq 7.9853$ and $r \simeq 15.9706$. And for $k = 3$, the system of equations (21) has the solution: $r_1 \simeq 1.99977$, $r_2 \simeq 3.99954$, $r_3 \simeq 7.99908$ and $r \simeq 15.9982$.

Corollary 3.4. *For $k = 4$, the system (21) has the solution: (shown by TM16) $r_1 = 2$, $r_2 = 4$, $r_3 = 8$ and $r = 16$. In this case, the efficiency index is $16^{\frac{1}{4}} = 2$ which shows that our developed method competes all the existing methods.*

Corollary 3.5. *As can easily be seen, the order of the convergence from 8 to 16 (100% of an improvement) is attained without any additional functional evaluations that this work presents the high computational efficiency of the proposed method. The efficiency index of the proposed method (16) is $EI = 16^{1/4} = 2$.*

4. Numerical results and comparisons

To show the efficiency of the described methods, I will compare their features with the ones obtained by optimal eighth-order derivative-free methods without memory and with memory numerically. The errors $|x_k - \alpha|$ of approximations to the sought roots, produced by the different methods at the first four iterations, are given in Table 2, where $Ae - h$ stands for $A \times 10^{-h}$. The test functions and their exact root α are displayed with only five decimal digits in Table 1.

Table 1: The test functions [34]

Nonlinear function	Root	Initial guess
$f_1(x) = x\log(1 + x\sin(x)) + e^{-1+x^2+x\cos(x)}\sin(\pi x)$	$\alpha = 0$	$x_0 = 0.6$
$f_2(x) = \frac{-5x^2}{2} + x^4 + x^5 + \frac{1}{1+x^2}$	$\alpha = 1$	$x_0 = 1.4$
$f_3(x) = x\log(1 + x^2 - \pi) + \tan(x^2) - \frac{1+x^2}{1+x^3}\sin(t^2)$	$\alpha = \sqrt{\pi}$	$x_0 = 1.7$
$f_4(x) = \log(1 + x^2) + e^{-3x+x^2}\sin(x)$	$\alpha = 0$	$x_0 = 0.35$
$f_5(x) = 1 + \frac{1}{x^4} - \frac{1}{x} - x^2$	$\alpha = 1$	$x_0 = 1.4$
$f_6(x) = \frac{8}{7} - \sqrt{6} + \frac{x^3}{1+x^4} + \sqrt{8 + x^4}\sin\frac{\pi}{2+x^2}$	$\alpha = -2$	$x_0 = -2.3$

Table 2: Comparison of the absolute error and COC of various iterative methods

	Methods	$\lvert x_1 - \alpha\rvert$	$\lvert x_2 - \alpha\rvert$	$\lvert x_3 - \alpha\rvert$	$\lvert x_4 - \alpha\rvert$	COC
$f_1(x)$	CJTYZM (14)	0.18818(−1)	0.19386(−28)	0.16031(−443)	0.38910(−6886)	15.5150
	ZLHM (1), $\gamma = -1$	0.60000(0)	0.23448(−3)	0.10417(−32)	0.15929(−267)	8.0000
	CLMTM (2)	0.60000(0)	0.23182(−2)	0.15923(−21)	0.80726(−175)	8.0000
	SAM (3)	0.60000(0)	0.86179(−1)	0.26208(−8)	0.11062(−68)	8.0000
	GKM (4)	0.45315(0)	0.94119(−6)	0.16748(−94)	0.16932(−1514)	16.0000
	SSSLM (5)	0.60000(0)	0.15176(−3)	0.77529(−57)	0.16479(−909)	16.0000
	LAM (6)	0.19386(−1)	0.12850(−28)	0.53611(−445)	0.83625(−6909)	15.5150
	SLTKM (7)	0.17769(−1)	0.39931(−19)	0.32046(−233)	0.11287(−2801)	12.0000
	JM (8)	0.14347(0)	0.232531(−12)	0.17059(−177)	0.13255(−2490)	14.0000
	TM16 (16), $k = 4$	0.18818(−1)	0.19386(−28)	0.45328(−460)	0.11484(−7366)	16.0010
$f_2(x)$	CJTYZM (14)	0.21644(0)	0.16864(−5)	0.55365(−85)	0.72154(−1341)	15.5020
	ZLHM (1)	0.40000(0)	0.91433(−1)	0.44901(−3)	0.54498(−20)	8.0000
	CLMTM (2)	0.40000(0)	0.21239(−3)	0.34356(−25)	0.16204(−199)	8.0000
	SAM (3)	0.40000(0)	0.86179(−1)	0.26208(−8)	0.11062(−68)	8.0000
	GKM (4)	0.29415(0)	0.37965(0)	0.37965(0)	0.37965(0)	0.0000
	SSSLM (5)	0.11206(−1)	0.67326(−21)	0.14048(−326)	0.18127(−5217)	16.0000
	LAM (6)	0.36503(−1)	0.16814(−16)	0.10277(−261)	0.17558(−4078)	15.5130
	SLTKM (7)	0.58190(−1)	0.28652(−8)	0.36402(−95)	0.64385(−1138)	12.0000
	JM (8)	0.43787(−1)	0.64404(−13)	0.90234(−178)	0.43616(−2486)	−1.0769
	TM16 (16), $k = 4$	0.21644(0)	0.16864(−5)	0.56862(−85)	0.15873(−1356)	16.0000

In order to test our proposed methods (16) and also to compare them with the methods (14), (1), (2), (3), (4), (5), (6) and (7), I compute the error, the computational order of convergence (COC) by the approximate formula [37]

$$COC \approx \frac{\ln|(x_{n+1} - \alpha)/(x_n - \alpha)|}{\ln|(x_n - \alpha)/(x_{n-1} - \alpha)|}. \tag{32}$$

Table 2: (Continued)

	Methods	$\lvert x_1-\alpha\rvert$	$\lvert x_2-\alpha\rvert$	$\lvert x_3-\alpha\rvert$	$\lvert x_4-\alpha\rvert$	COC
$f_3(x)$	CJTYZM (14)	0.38877(−6)	0.73864(−95)	0.11636(−1474)	0.28873(−22878)	15.5160
	ZLHM (1)	0.72454(−1)	0.19884(−1)	0.76133(−8)	0.16532(−60)	8.0000
	CLMTM (2)	0.72454(−1)	0.33570(−8)	0.81114(−68)	0.94253(−545)	8.0000
	SAM (3)	0.72454(−1)	0.23872(−8)	0.26806(−26)	0.67766(−584)	8.0000
	GKM (4)	0.60313(−15)	0.11751(−241)	0.50666(−3869)	0.72266(−61907)	16.0000
	SSSLM (5)	0.11233(−14)	0.10052(−237)	0.17002(−3806)	0.76333(−60907)	16.0000
	LAM (6)	0.27167(−6)	0.15849(−97)	0.86129(−1516)	0.20432(−23516)	15.5160
	SLTKM (7)	0.14078(−6)	0.98648(−80)	0.67330(−956)	0.68808(−11470)	12.0000
	JM (8)	0.8537(−7)	0.12125(−91)	0.13471(−1284)	0.58849(−17986)	−1.0769
	TM16 (16), $k=4$	0.38877(−6)	0.73864(−95)	0.72246(−1515)	0.39231(−24235)	16.0000
$f_4(x)$	CJTYZM (14)	0.14841(−8)	0.74322(−131)	0.22054(−2035)	0.11199(−31580)	15.5160
	ZLHM (1)	0.35000(0)	0.17236(−4)	0.32121(−35)	0.46744(−281)	8.0000
	CLMTM (2)	0.35000(0)	0.36365(−6)	0.78834(−50)	0.38457(−399)	8.0000
	SAM (3)	0.35000(0)	0.31864(−3)	0.22968(−26)	0.16752(−211)	8.0000
	GKM (4)	0.31473(−5)	0.37550(−80)	0.63308(−1279)	0.26976(−20459)	16.0000
	SSSLM (5)	0.75661(−5)	0.14428(−72)	0.44141(−1156)	0.26002(−18492)	16.0000
	LAM (6)	0.14806(−8)	0.98452(−132)	0.34191(−2047)	0.17398(−31763)	15.5160
	SLTKM (7)	0.47200(−4)	0.89870(−48)	0.16770(−572)	0.29881(−6869)	12.0000
	JM (8)	0.30950(−6)	0.46687(−87)	0.38533(−1218)	0.26235(−17053)	0.0000
	TM16 (16), $k=4$	0.14841(−8)	0.74322(−131)	0.17768(−2092)	0.21810(−33478)	16.0000
$f_5(x)$	CJTYZM (14)	0.36373(−4)	0.71417(−64)	0.20331(−994)	0.15521(−15428)	15.5160
	ZLHM (1)	0.40000(0)	0.49444(−1)	0.13420(−6)	0.41166(−51)	8.0000
	CLMTM (2)	0.40000(0)	0.13970(−3)	0.22159(−29)	0.88653(−236)	8.0000
	SAM (3)	0.40000(0)	0.12877(−3)	0.58002(−31)	0.98006(−250)	8.0000
	GKM (4)	0.14326(−5)	0.77018(−87)	0.37509(−1387)	0.37576(−22192)	16.0000
	SSSLM (5)	0.11522(−2)	0.50671(−39)	0.10465(−620)	0.11469(−9927)	16.0000
	LAM (6)	0.33505(−4)	0.80238(−66)	0.15298(−1024)	0.47261(−15896)	15.5160
	SLTKM (7)	0.13208(−3)	0.52727(−43)	0.15341(−515)	0.56439(−6186)	12.0000
	JM (8)	0.10412(−3)	0.232531(−52)	0.30173(−735)	0.17664(−10295)	−1.0769
	TM16 (16), $k=4$	0.36373(−4)	0.71417(−64)	0.26681(−1022)	0.0000(−100000)	16.0000
$f_6(x)$	CJTYZM (14)	0.68180(−10)	0.21784(−160)	0.18000(−2500)	0.15521(−15428)	15.5160
	ZLHM (1)	0.30000(−10)	0.52074(−7)	0.81302(−60)	0.27803(−482)	8.0000
	CLMTM (2)	0.30000(−0)	0.14898(−6)	0.18319(−55)	0.95738(−447)	8.0000
	SAM (3)	0.30000(0)	0.43304(−6)	0.12618(−51)	0.65548(−416)	8.0000
	GKM (4)	0.10733(−10)	0.21023(−175)	0.98647(−2811)	0.54491(−44976)	16.0000
	SSSLM (5)	0.57945(−11)	0.23054(−179)	0.90848(−2874)	0.30726(−45984)	16.0000
	LAM (6)	0.60942(−10)	0.35824(−161)	0.20767(−2512)	0.58386(−38986)	15.5160
	SLTKM (7)	0.22656(−6)	0.30908(−81)	0.74024(−980)	0.26359(−11763)	12.0000
	JM (8)	0.99381(−8)	0.492531(−115)	0.92223(−1621)	0.60042(−22701)	−1.0769
	TM16 (16), $k=4$	0.68180(−10)	0.21784(−160)	0.30138(−2580)	0.15289(−41290)	16.0000

Table 3: Comparison of TNE and EI for different derivative-free methods

	Methods	$\lvert f(x_k)\rvert$	TNE	IT	$ACOC$	r_c	EI
$f_1(x)$	CJTYZM (14)	0.61154(−106846)	4	4	15.5210	15.5150	1.9847
	ZLHM (1)	0.35149(−17176)	4	4	8.0000	8.0000	1.6818
	CLMTM (2)	0.40724(−1403)	4	6	8.0000	8.0000	1.6818
	SAM (3)	0.12878(−551)	4	5	8.0000	8.0000	1.6818
	GKM (4)	0.90653(−2640)	4	4	16.0000	16.0000	1.7411
	SSSLM (5)	0.00000(0)	4	5	16.0000	16.0000	1.7411
	LAM (6)	0.64267(−107197)	4	5	15.5240	15.5120	1.9847
	SLTKM (7)	0.47531(−33623)	4	5	11.9970	12.0000	1.8612
	JM (8)	0.47531(−33623)	4	5	14.0000	10.8970	1.8612
	TM16 (16), $k = 4$	0.00000(−60000)	4	4	15.9940	16.0010	2.0000
$f_2(x)$	CJTYZM (14)	0.24390(−20809)	4	4	15.8010	15.5020	1.9849
	ZLHM (1)	0.12696(−9899)	4	4	8.0000	8.0000	1.6818
	CLMTM (2)	0.26465(−712)	4	6	8.0000	8.0000	1.6818
	SAM (3)	0.13891(−1595)	4	5	8.0000	8.0000	1.6818
	GKM (4)	0.90653(−2640)	4	4	16.0000	16.0000	0.0000
	SSSLM (5)	0.63444(−5217)	4	5	16.0000	16.0000	1.7411
	LAM (6)	0.32073(−63286)	4	5	15.5650	15.5130	1.9846
	SLTKM (7)	0.21122(−13650)	4	5	12.0000	12.0000	1.8612
	JM (8)	0.57955(−34802)	4	5	−1.0771	−1.0769	*div*
	TM16 (16), $k = 4$	0.75500(−21701)	4	4	15.5570	16.0000	2.0000
$f_3(x)$	CJTYZM (14)	0.10996(−354969)	4	4	15.5120	15.5160	1.9849
	ZLHM (1)	0.35133(−3853)	4	7	8.0000	8.0000	1.6818
	CLMTM (2)	0.37787(−4361)	4	6	8.0000	8.0000	1.6818
	SAM (3)	0.13639(−4383)	4	5	8.0000	8.0000	1.6818
	GKM (4)	0.87176(−61907)	4	4	16.0000	16.0000	1.7411
	SSSLM (5)	0.92083(−60906)	4	5	16.0000	16.0000	1.7411
	LAM (6)	0.79483(−364871)	4	5	15.5120	15.5160	1.9847
	SLTKM (7)	0.10771(−137636)	4	5	12.0000	12.0000	1.8612
	JM (8)	0.65447(−251804)	4	5	−1.0771	0.0000	*div*
	TM16 (16), $k = 4$	0.33744(−387758)	4	4	16.0050	16.0000	2.0000
$f_4(x)$	CJTYZM (14)	0.25891(−489996)	4	4	15.5130	15.5160	1.9847
	ZLHM (1)	0.94013(−2250)	4	6	8.0000	8.0000	1.6818
	CLMTM (2)	0.12331(−3195)	4	6	8.0000	8.0000	1.6818
	SAM (3)	0.13416(−1692)	4	5	8.0000	8.0000	1.6818
	GKM (4)	0.26976(−20459)	4	4	16.0000	16.0000	1.7411
	SSSLM (5)	0.26002(−18942)	4	5	16.0000	16.0000	1.7411
	LAM (6)	0.14535(−492831)	4	5	15.5140	15.5160	1.9847
	SLTKM (7)	0.30605(−82430)	4	5	12.0000	12.0000	1.8612
	JM (8)	0.12061(−238745)	4	5	14.0000	0.0000	*div*
	TM16 (16), $k = 4$	0.57958(−535653)	4	4	16.0390	16.0000	2.0000

Table 3: (Continued)

	Methods	$\lvert f(x_k)\rvert$	TNE	IT	$ACOC$	r_c	EI
$f_5(x)$	CJTYZM (14)	0.58201(−239384)	4	4	15.5110	15.5160	1.9847
	ZLHM (1)	0.23065(−3257)	4	4	8.0000	8.0000	1.6818
	CLMTM (2)	0.29164(−1887)	4	6	8.0000	8.0000	1.6818
	SAM (3)	0.32558(−1999)	4	5	8.0000	8.0000	1.6818
	GKM (4)	0.18788(−22192)	4	4	16.0000	16.0000	1.7411
	SSSLM (5)	0.57345(−9927)	4	5	16.0000	16.0000	1.7411
	LAM (6)	0.91761(−246638)	4	5	15.5120	15.5160	1.9847
	SLTKM (7)	0.17353(−74230)	4	5	12.0000	12.0000	1.8612
	JM (8)	0.49051(−144138)	4	5	−1.0769	−12.665	*div*
	TM16 (16), $k = 4$	0.0000(−100000)	4	4	16.0520	16.0000	2.0000
$f_6(x)$	CJTYZM (14)	0.58201(−239384)	4	5	15.5110	15.5160	1.9847
	ZLHM (1)	0.22995(−3864)	4	4	8.0000	8.0000	1.6818
	CLMTM (2)	0.17689(−3577)	4	6	8.0000	8.0000	1.6818
	SAM (3)	0.11542(−3330)	4	5	8.0000	8.0000	1.6818
	GKM (4)	0.18788(−22191)	4	4	16.0000	16.0000	1.7411
	SSSLM (5)	0.10200(−45984)	4	5	16.0000	16.0000	1.7411
	LAM (6)	0.13397(−604899)	4	5	15.5120	15.5160	1.9847
	SLTKM (7)	0.36375(−141165)	4	5	12.0000	12.0000	1.8612
	JM (8)	0.48999(−317824)	4	5	−1.0769	−1.0769	*div*
	TM16 (16), $k = 4$	0.0000(−200000)	4	4	16.0790	16.0000	2.0000

and the approximated computational order of convergence, (ACOC) by the formula [6]

$$ACOC \approx \frac{\ln|(x_{n+1}-x_n)/(x_n-x_{n-1})|}{\ln|(x_n-x_{n-1})/(x_{n-1}-x_{n-2})|}. \tag{33}$$

Also,

$$r_c \approx \frac{\ln|f(x_{n+1})/f(x_n)|}{\ln|f(x_n)/f(x_{n-1})|}, \tag{34}$$

the calculated value r_c approximates well with the theoretical order of convergence [23]. The package Mathematica 10, with 5000 arbitrary precision arithmetic, has been used in our computations. Symbols div, In, TNE, EI, and IT are the abbreviated form of divergence, Infinity, the total number of evaluations, Efficiency Index, and the number iterations, respectively. Tables 2 and 3 show that the proposed methods compete with the previous methods. In addition, its efficiency index is much better than previous works.

5. Conclusion

It has solved the nonlinear equations using a family of the with-memory methods. It is worth mentioning out that the member of the family can be used with any function. Tables 2 and 3 indicate that the absolute error of proposed methods, Steffensen-Ostrwoski's type, is lower than those with and without memory. Tables 4 and 5 show that the efficiency index of the proposed method is equal to $16^{\frac{1}{4}} = 2$. And the better efficiency index of all three and four-step without-memory and with-memory method has been investigated so far. Also referenced in this article [1, 2, 3, 4, 8, 9, 16, 18, 19, 17, 14, 20, 12, 21, 22, 24, 25, 28, 29, 31, 32, 36].

References

[1] S. Abbasbandy, *Improving Newton-Raphson method for nonlinear equations by modified Adomian decomposition method*, Appl. Math. Comput., 145 (2003), pp. 887–893.

[2] P. Bakhtiari, T. Lotfi, K. Mahdiani, and F. Soleymani, *Interval Ostrowski-type methods with guaranteed convergence*, Ann. Univ. Ferrara Sez. VII Sci. Mat., 59 (2013), pp. 221–234.

[3] W. Bi, H. Ren, and Q. Wu, *Three-step iterative methods with eighth-order convergence for solving nonlinear equations*, J. Comput. Appl. Math., 225 (2009), pp. 105–112.

[4] N. Choubeĭ and D. P. Dzhaisval, *Two- and three-point methods with memory for solving nonlinear equations*, Sib. Zh. Vychisl. Mat., 20 (2017), pp. 91–106.

[5] A. Cordero, T. Lotfi, K. Mahdiani, and J. R. Torregrosa, *A stable family with high order of convergence for solving nonlinear equations*, Appl. Math. Comput., 254 (2015), pp. 240–251.

[6] A. Cordero and J. R. Torregrosa, *Variants of Newton's method using fifth-order quadrature formulas*, Appl. Math. Comput., 190 (2007), pp. 686–698.

[7] A. Cordero, M. ud Din Junjua, J. R. Torregrosa, N. Yasmin, and F. Zafar, *Efficient Four-Parametric with-and-without-memory iterative methods possessing high efficiency indices*, Math. Probl. Eng., 2018 (2018), pp. 1–12.

[8] J. Džunić, M. S. Petković, and L. D. Petković, *Three-point methods with and without memory for solving nonlinear equations*, Appl. Math. Comput., 218 (2012), pp. 4917–4927.

[9] T. Eftekhari, *A new family of four-step fifteenth-order root-finding methods with high efficiency index*, Comput. Methods Differ. Equ., 3 (2015), pp. 51–58.

[10] ——, *Two efficient bi-parametric derivative free with memory methods for finding simple roots nonlinear equations*, J. Adv. Appl. Math., 1 (2016), pp. 203–210.

[11] 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), pp. 1336–1342.

[12] M.-U.-D. Junjua, S. Akram, T. Afzal, and A. Ali, *Construction of optimal derivative free iterative methods for nonlinear equations using Lagrange interpolation*, J. Prime Res. Math., 16 (2020), pp. 30–45.

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

[14] M. J. Lalehchini, T. Lotfi, and K. Mahdiani, *On developing an adaptive free-derivative Kung and Traub's method with memory*, J. Math. Ext., 14 (2020), pp. 171–188.

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

[16] T. Lotfi, A. Cordero, J. R. Torregrosa, M. A. Abadi, and M. Mohammadi-Zadeh, *On generalization based on bi et al. iterative methods with eighth-order convergence for solving nonlinear equations*, Sci. World. J., 2014 (2014), pp. 1–8.

[17] T. Lotfi, K. Mahdiani, P. Bakhtiari, and F. Soleymani, *Constructing two-step iterative methods with and without memory*, Comput. Math. Math. Phys., 55 (2015), pp. 183–193.

[18] T. Lotfi, S. Sharifi, M. Salimi, and S. Siegmund, *A new class of three-point methods with optimal convergence order eight and its dynamics*, Numer. Algor., 68 (2015), pp. 261–288.

[19] T. Lotfi and E. Tavakoli, *On a new efficient steffensen-like iterative class by applying a suitable self-accelerator parameter*, Sci. World. J., 2014 (2014), pp. 1–9.

[20] G. Matthies, M. Salimi, S. Sharifi, and J. L. Varona, *An optimal three-point eighth-order iterative method without memory for solving nonlinear equations with its dynamics*, Jpn. J. Ind. Appl. Math., 33 (2016), pp. 751–766.

[21] A. M. Ostrowski, *Solution of equations and systems of equations*, Pure and Applied Mathematics, Vol. 9, Academic Press, New York-London, second ed., 1966.

[22] M. S. Petković, J. Džunić, and B. Neta, *Interpolatory multipoint methods with memory for solving nonlinear equations*, Appl. Math. Comput., 218 (2011), pp. 2533–2541.

[23] M. S. Petković, B. Neta, L. D. Petković, and J. Džunić, *Multipoint methods for solving nonlinear equations*, Elsevier/Academic Press, Amsterdam, 2013.

[24] ——, *Multipoint methods for solving nonlinear equations: a survey*, Appl. Math. Comput., 226 (2014), pp. 635–660.

[25] M. Salimi, N. M. A. Nik Long, S. Sharifi, and B. A. Pansera, *A multi-point iterative method for solving nonlinear equations with optimal order of convergence*, Jpn. J. Ind. Appl. Math., 35 (2018), pp. 497–509.

[26] 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), pp. 201–215.

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

[28] F. Soleymani, *Regarding the accuracy of optimal eighth-order methods*, Math. Comput. Modelling, 53 (2011), pp. 1351–1357.

[29] ——, *Some optimal iterative methods and their with memory variants*, J. Egyptian Math. Soc., 21 (2013), pp. 133–141.

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

[31] V. Torkashvand, *A general class of one-parametric with memory method for solving nonlinear equations*, Casp. J. Math. Sci., 10 (2021), pp. 309–335.

[32] ——, *Transforming Ostrowski's method into a derivative-free method and its dynamics*, Comput. Math. Comput. Model. Appl., 2 (2023), pp. 1–10.

[33] V. Torkashvand and M. Kazemi, *On an efficient family with memory with high order of convergence for solving nonlinear equations*, Int. J. Ind. Math., 12 (2020), pp. 209–224.

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

[35] J. F. Traub, *Iterative methods for the solution of equations*, Prentice-Hall Series in Automatic Computation, Prentice-Hall, Inc., Englewood Cliffs, NJ, 1964.

[36] X. Wang and L. Liu, *Modified Ostrowski's method with eighth-order convergence and high efficiency index*, Appl. Math. Lett., 23 (2010), pp. 549–554.

[37] S. Weerakoon and T. G. I. Fernando, *A variant of Newton's method with accelerated third-order convergence*, Appl. Math. Lett., 13 (2000), pp. 87–93.

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

Please cite this article using:

Vali Torkashvand, Interpolatory four-parametric adaptive method with memory for solving nonlinear equations, AUT J. Math. Com., 6(1) (2024) 77-89
https://doi.org/10.22060/AJMC.2023.22090.1132