AIMS Mathematics, 10(3): 5421–5443.
DOI: 10.3934/math.2025250
Received: 26 October 2024
Revised: 10 February 2025
Accepted: 14 February 2025
Published: 11 March 2025

https://www.aimspress.com/journal/Math

Research article

Two novel efficient memory-based multi-point iterative methods for solving nonlinear equations

Shubham Kumar Mittal[1], **Sunil Panday**[1,*], **Lorentz Jäntschi**[2] **and Liviu C. Bolunduţ**[2]

[1] Department of Mathematics, National Institute of Technology Manipur, Langol 795004, Manipur, India; shubhammehara2@gmail.com

[2] Department of Physics and Chemistry, Technical University of Cluj-Napoca, Muncii Blvd 103105, Cluj-Napoca 400641, Romania; lorentz.jantschi@chem.utcluj.ro, liviu.bolundut@chem.utcluj.ro

* **Correspondence:** Email: sunilpanday@hotmail.co.in.

Abstract: In order to solve nonlinear equations, we introduce two new three-step with-memory iterative methods in this paper. We have improved the order of convergence of a well-known optimal eighth-order iterative method by extending it into two with-memory methods using one and two self-accelerating parameters, respectively. The self-accelerating parameters that increase the convergence order are computed using the Hermite interpolating polynomial. The newly proposed uni-parametric and bi-parametric with-memory iterative methods (IM) improved the R-order of convergence of the existing eighth-order method from 8 to 10 and 10.7446, respectively. Furthermore, the efficiency index has increased from 1.6818 to 1.7783 and 1.8105, respectively. In addition, this improvement in convergence order and efficiency index can be obtained without using any extra function evaluations. Extensive numerical testing on a wide range of problems demonstrates that the proposed methods are more efficient than some well-known existing methods.

Keywords: nonlinear equation; roots; efficiency index; iterative method; with-memory methods
Mathematics Subject Classification: 41A25, 65D99, 65H05

1. Introduction

In the realms of engineering and mathematical modeling, solving nonlinear equations of the form

$$\varphi(s) = 0$$

has emerged as a fundamental challenge, as finding their roots is critical to addressing a variety of complex problems. Numerous iterative methods have been proposed to tackle this issue (see, [1, 2]),

and they play an essential role in numerical analysis due to their applicability across disciplines in both engineering and applied sciences. Iterative methods approximate solutions with a specified level of accuracy. Our objective is to increase the convergence order of multipoint methods without the evaluation of additional functions while significantly improving the efficiency index and overall performance. The literature contains plenty of one-point and multipoint without-memory methods for solving nonlinear equations (see, [3, 4]). Among the classical approaches, Newton's without-memory method stands out for its quadratic convergence with an efficiency index of 1.414, making it a cornerstone in the field. Newton's method [5] is defined by the iteration:

$$s_{k+1} = s_k - \frac{\varphi(s_k)}{\varphi'(s_k)}, \quad k = 0, 1, 2, \cdots .$$

According to [5], the efficiency index,

$$E = \rho^{1/\gamma}$$

represents the balance between the order of convergence and the number of function evaluations per step. Here, ρ denotes the order of convergence, and γ denotes the number of function and derivative evaluations performed per iteration.

Recently, significant attempts have been made by many researchers in the field of numerical analysis to use self-accelerating parameters to extend without-memory methods to with-memory methods. Memory-based iterative methods use current and past iterations to raise the convergence order and the efficiency index. Wang and Tao [6] upgraded a Newton-type without-memory method into its with-memory variant using one self-accelerating parameter and achieved an improvement in order of convergence from 2 to 2.4142. Torkashvand [7] upgraded a two-step without-memory method into a with-memory method using three self-accelerating parameters and achieved an improvement in the order of convergence from 4 to 7.53. Also, Thangkhenpau et al. [8] used the technique of self-accelerating parameters for multiple roots and upgraded a fifth-order without-memory method into a with-memory method using one parameter and achieved the convergence order 7.2749. In recent years, the development of with-memory iterative methods has gained considerable interest among researchers. Notable contributors to the development of with-memory methods include Lotfi and Assari [9], Panday et al. [10] and Mittal et al. [11].

In this article, we proposed two new three-step, uni-parametric and bi-parametric with-memory iterative techniques by adding two self-accelerating parameters to the first and third step of an existing optimal eighth-order without-memory iterative technique [12], and it improves the R-order of convergence from 8 to 10 and 10.7446, respectively, and the efficiency index is also improved by 1.6818 to 1.7783 and 1.8105, respectively, in Section 2. A complete assessment utilizing numerical tests is offered in Section 3, which provides a comparison of the proposed methods with other well-established methods. A thorough conclusion of the research findings is provided in Section 4.

2. Analysis of convergence for with-memory methods

This section is divided into two subsections. In the first subsection, we will use a parameter α in the first step of the optimal eight-order method proposed by Solaiman and Hashim [12] in 2021, and we will prove the theoretical order of the method using a theorem. Then, in the second subsection, we will use one more parameter, β, in the third step of [12], and we will also prove its theoretical order of convergence using a theorem.

2.1. The uni-parametric with-memory method and its convergence analysis

Here, we introduce the parameter α in the first step of the optimal eighth-order without-memory method, presented in the article [12]:

$$\begin{aligned} v_k &= s_k - \frac{\varphi(s_k)}{\varphi'(s_k) + \alpha\varphi(s_k)}, \\ t_k &= v_k - \frac{\varphi(v_k)}{q(v_k)} - \frac{2(\varphi(v_k))^2 q(v_k) R(s_k, v_k)}{4(q(v_k))^4 - 4\varphi(v_k)(q(v_k))^2 R(s_k, v_k) + (\varphi(v_k))^2 (R(s_k, v_k))^2}, \\ s_{k+1} &= t_k - \frac{\varphi(t_k)}{w'(t_k)}, \end{aligned} \tag{2.1}$$

where $q(v_k)$, $w'(t_k)$ and $R(s_k, v_k)$ are defined in [12] as:

$$\begin{aligned} \varphi'(v_k) &\approx q(v_k) \\ &= 2\varphi[v_k, s_k] - \varphi'(s_k), \\ \varphi'(t_k) &\approx w'(t_k) \\ &= \varphi[t_k, s_k]\left(2 + \frac{s_k - t_k}{v_k - t_k}\right) - \frac{(s_k - t_k)^2}{(s_k - v_k)(v_k - t_k)}\varphi[v_k, s_k] + \varphi'(s_k)\frac{v_k - t_k}{s_k - v_k}, \\ \varphi''(v_k) &\approx R(s_k, v_k) \\ &= \left[3\frac{\varphi(v_k) - \varphi(s_k)}{v_k - s_k} - 2q(v_k) - \varphi'(s_k)\right]\frac{2}{s_k - v_k} \\ &= 2\frac{\varphi'(s_k) - \varphi[v_k, s_k]}{s_k - v_k}, \end{aligned}$$

and, $\varphi[v_k, s_k]$ and $\varphi[t_k, s_k]$ represents divided differences and are calculated as

$$\varphi[v_k, s_k] = \frac{\varphi(v_k) - \varphi(s_k)}{v_k - s_k}$$

and

$$\varphi[t_k, s_k] = \frac{\varphi(t_k) - \varphi(s_k)}{t_k - s_k},$$

respectively.

Using Taylor-series approximation, the expressions for $\varphi(s_k)$ and $\varphi'(s_k)$ can be written as:

$$\varphi(s_k) = A\left(e_k + c_2 e_k^2 + c_3 e_k^3 + c_4 e_k^4 + c_5 e_k^5 + c_6 e_k^6 + c_7 e_k^7 + c_8 e_k^8\right) + O\left(e_k^9\right), \tag{2.2}$$

$$\varphi'(s_k) = A\left(1 + 2c_2 e_k + 3c_3 e_k^2 + 4c_4 e_k^3 + 5c_5 e_k^4 + 6c_6 e_k^5 + 7c_7 e_k^6 + 8c_8 e_k^7 + 9c_9 e_k^8\right) + O\left(e_k^9\right), \tag{2.3}$$

where

$$A = \varphi'(\xi),$$

ξ is the simple root of $\varphi(s)$,

$$e_k = s_k - \xi$$

and

$$c_j = \frac{\varphi^{(j)}(\xi)}{j!\varphi'(\xi)}$$

for $j = 2, 3, \cdots$.

Now, we obtain the error expressions for each sub-step of (2.1) as:

$$\begin{aligned} e_{k,v} &= (\alpha + c_2)\, e_k^2 + \left(-\alpha^2 - 2c_2(\alpha + c_2) + 2c_3\right) e_k^3 \\ &\quad + \left(\alpha^3 + 5\alpha c_2^2 + 4c_2^3 + c_2(3\alpha^2 - 7c_3) - 4\alpha c_3 + 3c_4\right) e_k^4 + O\left(e_k^5\right), \qquad (2.4) \\ e_{k,t} &= -\, c_3\, (\alpha + c_2)\, e_k^4 \\ &\quad + \left(c_3(\alpha^2 + 2c_2(\alpha + c_2) - 2c_3) - 2(\alpha + c_2)c_4\right) e_k^5 + O\left(e_k^6\right) \qquad (2.5) \end{aligned}$$

and

$$\begin{aligned} e_{k+1} =& (\alpha + c_2)^2 c_3(c_2c_3 - c_4)e_k^8 - 2\Big((\alpha + c_2)(2c_2^3c_3^2 + 2c_2^2c_3(\alpha c_3 - 2c_4) + 2c_3^2c_4 \\ &+ \alpha c_4^2 + \alpha c_3(-\alpha c_4 + c_5) + c_2(\alpha^2 c_3^2 - 2c_3^3 + c_4^2 + c_3(-4\alpha c_4 + c_5)))\Big)e_k^9 + O\left(e_k^{10}\right), \qquad (2.6) \end{aligned}$$

where

$$e_{k,v} = v_k - \xi, \quad e_{k,t} = t_k - \xi$$

and $\alpha \in \mathbb{R}$. We obtain the following with-memory iterative scheme by replacing α with a self-accelerating parameter α_k in (2.1), as:

$$\begin{aligned} v_k &= s_k - \frac{\varphi(s_k)}{\varphi'(s_k) + \alpha_k \varphi(s_k)}, \\ t_k &= v_k - \frac{\varphi(v_k)}{q(v_k)} - \frac{2(\varphi(v_k))^2 q(v_k) R(s_k, v_k)}{4(q(v_k))^4 - 4\varphi(v_k)(q(v_k))^2 R(s_k, v_k) + (\varphi(v_k))^2 (R(s_k, v_k))^2}, \\ s_{k+1} &= t_k - \frac{\varphi(t_k)}{w'(t_k)}, \qquad (2.7) \end{aligned}$$

where, the values of $q(v_k)$, $w'(t_k)$, and $R(s_k, v_k)$ are given in Eq (2.1).

The new with-memory method (NWM) mentioned in Eq (2.7) is expressed by NWM10. It is now clear from Eq (2.6) that for

$$\alpha \neq -c_2,$$

the convergence order of (2.1) is eight. Then, we can assume

$$\alpha = -c_2 = -\frac{\varphi''(\xi)}{2\varphi'(\xi)}$$

to speed up the order of convergence of (2.7) from eight to ten. However, in reality, the exact values of $\varphi'(\xi)$ and $\varphi''(\xi)$ are not achievable in practice. Thus, we shall choose the parameter α as α_k. Using the available data from the current and previous iterations, the parameter α_k can be calculated and satisfies the condition

$$\lim_{k\to\infty} \alpha_k = -c_2 = -\frac{\varphi''(\xi)}{2\varphi'(\xi)},$$

which means that the error expression (2.6) should have zero values for the eighth and ninth order asymptotic convergence constants. α_k can be calculated using the formula:

$$\alpha_k = -\frac{H_5''(s_k)}{2\varphi'(s_k)}, \qquad (2.8)$$

where

$$\begin{aligned} H_5(s) =&\varphi(s_k) + (s - s_k)\varphi[s_k, s_k] + (s - s_k)^2\varphi[s_k, s_k, t_{k-1}] + (s - s_k)^2(s - t_{k-1})\varphi[s_k, s_k, t_{k-1}, v_{k-1}] \\ &+ (s - s_k)^2(s - t_{k-1})(s - v_{k-1})\varphi[s_k, s_k, t_{k-1}, v_{k-1}, s_{k-1}] \\ &+ (s - s_k)^2(s - t_{k-1})(s - v_{k-1})(s - s_{k-1})\varphi[s_k, s_k, t_{k-1}, v_{k-1}, s_{k-1}, s_{k-1}]. \end{aligned} \tag{2.9}$$

Note: The condition

$$H'_m(s_k) = \varphi'(s_k)$$

is satisfied by the Hermite interpolating polynomial $H_m(s)$ for $m = 5$. So,

$$\alpha_k = -\frac{H''_5(s_k)}{2\varphi'(s_k)}$$

can be expressed as

$$\alpha_k = -\frac{H''_5(s_k)}{2H'_m(s_k)}$$

for $m = 5$.

Theorem 2.1. *Let H_m be the Hermite polynomial of degree m, interpolating the function φ at interpolation nodes $s_k, s_k, t_{k-1}, v_{k-1}, s_{k-1}, s_{k-1}$ within an interval $D \subset \mathbb{R}$, and the derivative $\varphi^{(m+1)}$ is continuous in D with*

$$H_m(s_k) = \varphi(s_k), \quad H'_m(s_k) = \varphi'(s_k).$$

Suppose that all nodes $s_k, s_k, t_{k-1}, v_{k-1}, s_{k-1}, s_{k-1}$ are in the neighborhood of the root ξ. Then,

$$H''_5(s_k) = 2\varphi'(\xi)(c_2 - c_6 e_{k-1,v} e_{k-1,t} e^2_{k-1})$$

and

$$\alpha_k = -\frac{H''_5(s_k)}{2\varphi'(s_k)} \sim -c_2 + c_6 e_{k-1,v} e_{k-1,t} e^2_{k-1}.$$

After simplification, we have

$$\alpha_k + c_2 = c_6 e_{k-1,v} e_{k-1,t} e^2_{k-1}. \tag{2.10}$$

Proof. We can calculate the expression of the fifth-degree Hermite interpolating polynomial as

$$\varphi(s) - H_5(s) = \frac{\varphi^{(6)}(\delta)}{6!}(s - s_k)^2(s - t_{k-1})(s - v_{k-1})(s - s_{k-1})^2, \tag{2.11}$$

where $\delta \in D$. Now, we get the below-mentioned equation after simplifying and differentiating Eq (2.11) two times at the point $s = s_k$,

$$H''_5(s_k) = \varphi''(s_k) - 2\frac{\varphi^{(6)}(\delta)}{6!}(s_k - t_{k-1})(s_k - v_{k-1})(s_k - s_{k-1})^2. \tag{2.12}$$

Next, Taylor's expansion of φ' and φ'' at the point s_k in D about the zero ξ of φ provides:

$$\varphi'(s_k) = \varphi'(\xi)\left(1 + 2c_2 e_k + 3c_3 e_k^2 + O(e_k^3)\right), \tag{2.13}$$

$$\varphi''(s_k) = \varphi'(\xi)\left(2c_2 + 6c_3e_k + O(e_k^2)\right). \tag{2.14}$$

Similarly,

$$\varphi^{(6)}(\delta) = \varphi'(\xi)\left(6!c_6 + 7!c_7e_\delta + O(e_\delta^2)\right), \tag{2.15}$$

where

$$e_\delta = \delta - \xi.$$

Putting Eqs (2.14) and (2.15) in (2.12), we obtain

$$H_5''(s_k) = 2\varphi'(\xi)(c_2 - c_6e_{k-1,v}e_{k-1,t}e_{k-1}^2). \tag{2.16}$$

Now, using Eqs (2.13) and (2.16), we have

$$-\frac{H_5''(s_k)}{2\varphi'(s_k)} \sim -c_2 + c_6e_{k-1,v}e_{k-1,t}e_{k-1}^2.$$

And hence

$$\alpha_k \sim -c_2 + c_6e_{k-1,v}e_{k-1,t}e_{k-1}^2$$

or

$$\alpha_k + c_2 = c_6e_{k-1,v}e_{k-1,t}e_{k-1}^2. \tag{2.17}$$

This completes the proof of Theorem 2.1. □

R-order of convergence. It can be said that a sequence generated by an iterative method (IM) $\{s_k\}$ converges to s^* with an R-order of convergence of at least $\tau > 1$ if there are constants $C \in (0, \infty)$ and $\theta \in (0, 1)$ such that [13]

$$\|s^* - s_k\| \leq C.\theta^{\tau^k}; \quad k = 0, 1, \cdots.$$

Theorem 2.2. *If the errors of approximations*

$$e_k = s_k - \xi$$

obtained in an iterative root-finding method satisfy and non-negative numbers l_j, $0 \leq j \leq m$, s.t.

$$e_{k+1} \sim \prod_{j=0}^{m}(e_{k-j})^{l_j}, k \geq k(\{s_k\}),$$

where, $k(\{s_k\})$ indicates the starting index that satisfies the above approximation. Then the R-order of convergence of the iterative method, denoted with $O_R(IM, \xi)$, satisfies the inequality

$$O_R(IM, \xi) \geq w^*,$$

where w^* is the unique positive solution of the equation [14, 15]

$$w^{m+1} - \sum_{j=0}^{m} l_j w^{m-j} = 0.$$

Presently, for the new iterative scheme with memory (2.7), we can state the subsequent convergence theorem.

Theorem 2.3. *In the iterative method (2.7), let α_k be a varying parameter and calculated by Eq (2.8). If an initial guess s_0 is enough near to a simple zero ξ of $\varphi(s)$, then the R-order of convergence of the iterative method (2.7) with memory is at least 10.*

Proof. Let the iterative method generates the sequence of $\{s_k\}$ which converges to the root ξ of $\varphi(s)$, by means of R-order

$$O_R(IM, \xi) \geq r,$$

we express

$$e_{k+1} \sim D_{k,r} e_k^r$$

and

$$e_k \sim D_{k-1,r} e_{k-1}^r. \tag{2.18}$$

Next, $D_{k,r}$ will tends to the asymptotic error constant D_r of IM by taking $k \to \infty$, then

$$\begin{aligned} e_{k+1} &\sim D_{k,r}(D_{k-1,r} e_{k-1}^r)^r \\ &= D_{k,r} D_{k-1,r}^r e_{k-1}^{r^2}. \end{aligned} \tag{2.19}$$

The resulting error expression of the with-memory scheme (2.7) can be obtained using (2.4)–(2.6) and the varying parameter α_k

$$e_{k,v} = v_k - \xi \sim (\alpha + c_2) e_k^2, \tag{2.20}$$

$$e_{k,t} = t_k - \xi \sim -c_3(\alpha + c_2) e_k^4 \tag{2.21}$$

and

$$e_{k+1} = s_{k+1} - \xi \sim (\alpha + c_2)^2 c_3 (c_2 c_3 - c_4) e_k^8. \tag{2.22}$$

Here, the higher order terms in Eqs (2.20)–(2.22) are excluded.

Now, let the R-order convergence of the iterative sequences $\{v_k\}$ and $\{t_k\}$ be p and q, respectively, then

$$e_{k,v} \sim D_{k,p} e_k^p \sim D_{k,p}(D_{k-1,r} e_{k-1}^r)^p = D_{k,p} D_{k-1,r}^p e_{k-1}^{rp} \tag{2.23}$$

and

$$e_{k,t} \sim D_{k,q} e_k^q \sim D_{k,q}(D_{k-1,r} e_{k-1}^r)^q = D_{k,q} D_{k-1,r}^q e_{k-1}^{rq}. \tag{2.24}$$

Now, by Eqs (2.17), (2.18) and (2.20), we obtain

$$\begin{aligned} e_{k,v} \sim (\alpha + c_2) e_k^2 &\sim (c_6 e_{k-1,v} e_{k-1,t} e_{k-1}^2)(D_{k-1,r} e_{k-1}^r)^2 \\ &\sim c_6 D_{k-1,p} e_{k-1}^p D_{k-1,q} e_{k-1}^q e_{k-1}^2 D_{k-1,r}^2 e_{k-1}^{2r} \\ &\sim c_6 D_{k-1,p} D_{k-1,q} D_{k-1,r}^2 e_{k-1}^{p+q+2r+2}. \end{aligned} \tag{2.25}$$

Also, by Eqs (2.17), (2.18) and (2.21), we obtain

$$\begin{aligned} e_{k,t} &\sim -c_3(\alpha + c_2)e_k^4 \sim -c_3(c_6 e_{k-1,v} e_{k-1,t} e_{k-1}^2)(D_{k-1,r} e_{k-1}^r)^4 \\ &\sim -c_3 c_6 D_{k-1,p} e_{k-1}^p D_{k-1,q} e_{k-1}^q e_{k-1}^2 D_{k-1,r}^4 e_{k-1}^{4r} \\ &\sim -c_3 c_6 D_{k-1,p} D_{k-1,q} D_{k-1,r}^4 e_{k-1}^{p+q+4r+2}. \end{aligned} \tag{2.26}$$

Again, by Eqs (2.17), (2.18) and (2.22), we have

$$\begin{aligned} e_{k+1} &\sim (\alpha + c_2)^2 c_3(c_2 c_3 - c_4)e_k^8 \\ &\sim c_3(c_2 c_3 - c_4)(c_6 e_{k-1,v} e_{k-1,t} e_{k-1}^2)^2 (D_{k-1,r} e_{k-1}^r)^8 \\ &\sim c_3(c_2 c_3 - c_4)c_6^2 e_{k-1,v}^2 e_{k-1,t}^2 e_{k-1}^4 D_{k-1,r}^8 e_{k-1}^{8r} \\ &\sim c_3(c_2 c_3 - c_4)c_6^2 (D_{k-1,p} e_{k-1}^p)^2 (D_{k-1,q} e_{k-1}^q)^2 e_{k-1}^4 D_{k-1,r}^8 e_{k-1}^{8r} \\ &\sim c_3(c_2 c_3 - c_4)c_6^2 D_{k-1,p}^2 e_{k-1}^{2p} D_{k-1,q}^2 e_{k-1}^{2q} e_{k-1}^4 D_{k-1,r}^8 e_{k-1}^{8r} \\ &\sim c_3(c_2 c_3 - c_4)c_6^2 D_{k-1,p}^2 D_{k-1,q}^2 D_{k-1,r}^8 e_{k-1}^{2p+2q+8r+4}, \end{aligned} \tag{2.27}$$

since

$$r > q > p.$$

By equating the exponents of e_{k-1} present in the set of (2.23)–(2.25), (2.24)–(2.26), and (2.19)–(2.27), we attain the resulting system of equations:

$$\begin{aligned} rp &= p + q + 2r + 2, \\ rq &= p + q + 4r + 2, \\ r^2 &= 2p + 2q + 8r + 4. \end{aligned} \tag{2.28}$$

The solution of the system of Eq (2.28) is specified by

$$p = 3, \quad q = 5 \quad \text{and} \quad r = 10.$$

As a result, the R-order of convergence of the with-memory iterative method (2.7) is at least 10.

This completes the proof. □

2.2. The bi-parametric with-memory method and its convergence analysis

Now, we will introduce one more parameter β, in the third step of the single parametric with-memory method presented in the Eq (2.7), as:

$$\begin{aligned} v_k &= s_k - \frac{\varphi(s_k)}{\varphi'(s_k) + \alpha_k \varphi(s_k)}, \\ t_k &= v_k - \frac{\varphi(v_k)}{q(v_k)} - \frac{2(\varphi(v_k))^2 q(v_k) R(s_k, v_k)}{4(q(v_k))^4 - 4\varphi(v_k)(q(v_k))^2 R(s_k, v_k) + (\varphi(v_k))^2 (R(s_k, v_k))^2}, \\ s_{k+1} &= t_k - \frac{\varphi(t_k)}{w'(t_k) + \beta \varphi(t_k)}, \end{aligned} \tag{2.29}$$

where, the values of $q(v_k)$, $w'(t_k)$, and $R(s_k, v_k)$ are given in Eq (2.1).

Now, the expressions for $\varphi(s_k)$ and $\varphi'(s_k)$ will be the same as given in the Eqs (2.2) and (2.3), respectively. And, the error expressions for each sub-step of (2.29) will be given by:

$$e_{k,v} = (\alpha + c_2)\, e_k^2 + \left(-\alpha^2 - 2c_2(\alpha + c_2) + 2c_3\right) e_k^3 + \left(\alpha^3 + 5\alpha c_2^2 + 4c_2^3 + c_2(3\alpha^2 - 7c_3) - 4\alpha c_3 + 3c_4\right) e_k^4 + O\left(e_k^5\right), \tag{2.30}$$

$$e_{k,t} = -(\alpha + c_2)\, c_3 e_k^4 + \left(\left(-\frac{1}{2} - 8\alpha\right)c_2^3 - 4c_2^4 - c_2^2\left(\alpha + 6\alpha^2 - 8c_3\right) + 3\alpha^2 c_3 - 2c_3^2 - 2\alpha c_4 - \frac{1}{2}c_2\left(\alpha^2 + 4\alpha^3 - 20\alpha c_3 + 4c_4\right)\right) e_k^5 + O\left(e_k^6\right) \tag{2.31}$$

and

$$\begin{aligned} e_{k+1} = {} & (\alpha + c_2)^2\, c_3\, ((\beta + c_2)\, c_3 - c_4)\, e_k^8 - \frac{1}{2}\Big((\alpha + c_2)\Big(2(1 + 24\alpha + 8\beta)c_2^6 + 16c_2^7 \\ & + 2c_2^5\Big(\beta + \alpha(3 + 28\alpha + 24\beta) - 24c_3\Big) + 2c_2^4\Big(\alpha(3\beta + \alpha(3 + 16\alpha + 28\beta)) \\ & - (1 + 52\alpha + 24\beta)c_3 + 10c_4\Big) + c_2^3\Big(2\alpha^2(\alpha + 4\alpha^2 + 3\beta + 16\alpha\beta) - 2(\beta \\ & + 2\alpha(1 + 19\alpha + 26\beta) - 20c_3)c_3 + c_4 + 8(5\alpha + \beta)c_4 - 4c_5\Big) + 2c_2^2\Big(\alpha^3(1 + 4\alpha)\beta \\ & + 4(6\alpha + 5\beta)c_3^2 + \alpha(1 + 13\alpha + 8\beta)c_4 - c_3(\alpha(\alpha + 10\alpha^2 + 2\beta + 38\alpha\beta) + 16c_4) \\ & - 4\alpha c_5\Big) + c_2\Big(12\alpha(\alpha + 4\beta)c_3^2 - 8c_3^3 + c_4(\alpha^2(1 + 6\alpha + 8\beta) + 4c_4) - 2c_3(\alpha^2(1 \\ & + 10\alpha)\beta + 2(9\alpha + 2\beta)c_4 - 2c_5) - 4\alpha^2 c_5\Big) + 4\Big(-2\beta c_3^3 + \alpha c_4^2 + c_3^2(3\alpha^2\beta + 2c_4) \\ & + \alpha c_3(-2(\alpha + \beta)c_4 + c_5)\Big)\Big)\Big) e_k^9 + \cdots + O\left(e_k^{12}\right), \end{aligned} \tag{2.32}$$

where $\alpha, \beta \in \mathbb{R}$. We obtain the following with-memory iterative scheme by replacing β with β_k in (2.29):

$$\begin{aligned} v_k &= s_k - \frac{\varphi(s_k)}{\varphi'(s_k) + \alpha_k \varphi(s_k)}, \\ t_k &= v_k - \frac{\varphi(v_k)}{q(v_k)} - \frac{2(\varphi(v_k))^2 q(v_k) R(s_k, v_k)}{4(q(v_k))^4 - 4\varphi(v_k)(q(v_k))^2 R(s_k, v_k) + (\varphi(v_k))^2 (R(s_k, v_k))^2}, \\ s_{k+1} &= t_k - \frac{\varphi(t_k)}{w'(t_k) + \beta_k \varphi(t_k)}, \end{aligned} \tag{2.33}$$

where, the values of $q(v_k)$, $w'(t_k)$, and $R(s_k, v_k)$ are given in Eq (2.1).

The scheme mentioned in Eq (2.33) is expressed by NWM11. It is now clear from the Eq (2.32) that for

$$\alpha \neq -c_2$$

and

$$\beta \neq \frac{c_4}{c_3} - c_2,$$

the convergence order of (2.29) is 8. Then, we can assume

$$\alpha = -c_2 = -\frac{\varphi''(\xi)}{2\varphi'(\xi)}$$

and

$$\beta = \frac{c_4}{c_3} - c_2 = \frac{\varphi^{(iv)}(\xi)}{4\varphi'''(\xi)} - \frac{\varphi''(\xi)}{2\varphi'(\xi)}$$

to speed up the order of convergence of (2.33) from eight to eleven. However, in reality, the exact values of $\varphi^{(iv)}(\xi), \varphi'''(\xi), \varphi''(\xi)$, and $\varphi'(\xi)$ are not achievable in practice. Thus, we shall choose the parameters α as α_k and β as β_k. Using the available data from the current and previous iterations, the parameters α_k and β_k can be calculated and satisfy the conditions

$$\lim_{k\to\infty} \alpha_k = -c_2 = -\frac{\varphi''(\xi)}{2\varphi'(\xi)}$$

and

$$\lim_{k\to\infty} \beta_k = \frac{c_4}{c_3} - c_2 = \frac{\varphi^{(iv)}(\xi)}{4\varphi'''(\xi)} - \frac{\varphi''(\xi)}{2\varphi'(\xi)},$$

which means that the error expression (2.32) should have zero values for the eighth, ninth, and tenth order asymptotic convergence constants. α_k can be calculated using Eq (2.8), and β_k can be calculated using the formula:

$$\beta_k = \frac{H_7^{(iv)}(t_k)}{4H_6'''(v_k)} - \frac{H_5''(s_k)}{2\varphi'(s_k)}, \tag{2.34}$$

where

$$\begin{aligned}
H_7(s) =& \varphi(t_k) + (s-t_k)\varphi[t_k, v_k] + (s-t_k)(s-v_k)\varphi[t_k, v_k, s_k] \\
&+ (s-t_k)(s-v_k)(s-s_k)\varphi[t_k, v_k, s_k, s_k] + (s-t_k)(s-v_k)(s-s_k)^2\varphi[t_k, v_k, s_k, s_k, t_{k-1}] \\
&+ (s-t_k)(s-v_k)(s-s_k)^2(s-t_{k-1})\varphi[t_k, v_k, s_k, s_k, t_{k-1}, v_{k-1}] \\
&+ (s-t_k)(s-v_k)(s-s_k)^2(s-t_{k-1})(s-v_{k-1})\varphi[t_k, v_k, s_k, s_k, t_{k-1}, v_{k-1}, s_{k-1}] \\
&+ (s-t_k)(s-v_k)(s-s_k)^2(s-t_{k-1})(s-v_{k-1})(s-s_{k-1})\varphi[t_k, v_k, s_k, s_k, t_{k-1}, v_{k-1}, s_{k-1}, s_{k-1}], \\
H_6(s) =& \varphi(v_k) + (s-v_k)\varphi[v_k, s_k] + (s-v_k)(s-s_k)\varphi[v_k, s_k, s_k] \\
&+ (s-v_k)(s-s_k)^2\varphi[v_k, s_k, s_k, t_{k-1}] + (s-v_k)(s-s_k)^2(s-t_{k-1})\varphi[v_k, s_k, s_k, t_{k-1}, v_{k-1}] \\
&+ (s-v_k)(s-s_k)^2(s-t_{k-1})(s-v_{k-1})\varphi[v_k, s_k, s_k, t_{k-1}, v_{k-1}, s_{k-1}] \\
&+ (s-v_k)(s-s_k)^2(s-t_{k-1})(s-v_{k-1})(s-s_{k-1})\varphi[v_k, s_k, s_k, t_{k-1}, v_{k-1}, s_{k-1}, s_{k-1}],
\end{aligned}$$

and the value of $H_5(s)$ can be obtained using Eq (2.9).

Note: The condition

$$H_m'(s_k) = \varphi'(s_k)$$

is satisfied by the Hermite interpolating polynomial $H_m(s)$ for $m = 5, 6, 7$. So,

$$\beta_k = \frac{H_7^{(iv)}(t_k)}{4H_6'''(v_k)} - \frac{H_5''(s_k)}{2\varphi'(s_k)}$$

can be expressed as

$$\beta_k = \frac{H_7^{(iv)}(t_k)}{4H_6'''(v_k)} - \frac{H_5''(s_k)}{2H_m'(s_k)}$$

for $m = 5, 6, 7$.

Theorem 2.4. *Let H_m be the Hermite polynomial of degree m, interpolating the function φ at interpolation nodes $t_k, v_k, s_k, s_k, t_{k-1}, v_{k-1}, s_{k-1}, s_{k-1}$ within an interval $D \subset \mathbb{R}$, and the derivative $\varphi^{(m+1)}$ is continuous in D with*

$$H_m(s_k) = \varphi(s_k), \quad H'_m(s_k) = \varphi'(s_k).$$

Suppose that all nodes $t_k, v_k, s_k, s_k, t_{k-1}, v_{k-1}, s_{k-1}, s_{k-1}$ are in the neighborhood of the root ξ. Then,

$$H_7^{(iv)}(t_k) = 24\varphi'(\xi)(c_4 - c_8 e_{k-1,v} e_{k-1,t} e_{k-1}^2),$$
$$H_6'''(v_k) = 6\varphi'(\xi)(c_3 - c_7 e_{k-1,v} e_{k-1,t} e_{k-1}^2)$$

and

$$H_5''(s_k) = 2\varphi'(\xi)(c_2 - c_6 e_{k-1,v} e_{k-1,t} e_{k-1}^2).$$

Also,

$$\begin{aligned}\beta_k &= \frac{H_7^{(iv)}(t_k)}{4H_6'''(v_k)} - \frac{H_5''(s_k)}{2\varphi'(s_k)} \\ &\sim \frac{c_4}{c_3} - c_2 + \left(\frac{c_4 c_7}{c_3^2} - \frac{c_8}{c_3} + \frac{c_8 c_7}{c_3^2} e_{k-1,v} e_{k-1,t} e_{k-1}^2 + c_6\right) e_{k-1,v} e_{k-1,t} e_{k-1}^2.\end{aligned}$$

After simplification, we have

$$(\beta_k + c_2)c_3 - c_4 \sim \left(\frac{c_4 c_7}{c_3} - c_8 + \frac{c_8 c_7}{c_3} e_{k-1,v} e_{k-1,y} e_{k-1}^2 + c_6 c_3\right) e_{k-1,v} e_{k-1,t} e_{k-1}^2.$$

Proof. We can calculate the expression of the seventh-degree, sixth-degree, and fifth-degree Hermite interpolating polynomial as:

$$\varphi(s) - H_7(s) = \frac{\varphi^{(8)}(\delta)}{8!}(s - t_k)(s - v_k)(s - s_k)^2(s - t_{k-1})(s - v_{k-1})(s - s_{k-1})^2, \tag{2.35}$$

$$\varphi(s) - H_6(s) = \frac{\varphi^{(7)}(\delta)}{7!}(s - v_k)(s - s_k)^2(s - t_{k-1})(s - v_{k-1})(s - s_{k-1})^2, \tag{2.36}$$

$$\varphi(s) - H_5(s) = \frac{\varphi^{(6)}(\delta)}{6!}(s - s_k)^2(s - t_{k-1})(s - v_{k-1})(s - s_{k-1})^2, \tag{2.37}$$

where $\delta \in D$. Now, we obtain the below-mentioned equations after simplifying and differentiating the Eq (2.35) four times at the point

$$s = t_k,$$

Eq (2.36) three times at the point

$$s = v_k$$

and Eq (2.37) two times at the point

$$s = s_k,$$

respectively,

$$H_7^{(iv)}(t_k) = \varphi^{(iv)}(t_k) - 24\frac{\varphi^{(8)}(\delta)}{8!}(t_k - t_{k-1})(t_k - v_{k-1})(t_k - s_{k-1})^2, \tag{2.38}$$

$$H_6'''(v_k) = \varphi'''(v_k) - 6\frac{\varphi^{(7)}(\delta)}{7!}(v_k - t_{k-1})(v_k - v_{k-1})(v_k - s_{k-1})^2, \tag{2.39}$$

$$H_5''(s_k) = \varphi''(s_k) - 2\frac{\varphi^{(6)}(\delta)}{6!}(s_k - t_{k-1})(s_k - v_{k-1})(s_k - s_{k-1})^2. \tag{2.40}$$

Next, Taylor's series expansion of φ', φ'', φ''', and $\varphi^{(iv)}$ at the point s_k in D about the zero ξ of φ provides:

$$\varphi'(s_k) = \varphi'(\xi)\left(1 + 2c_2e_k + 3c_3e_k^2 + O(e_k^3)\right)$$

and

$$\varphi''(s_k) = \varphi'(\xi)\left(2c_2 + 6c_3e_k + O(e_k^2)\right), \tag{2.41}$$

$$\varphi'''(v_k) = \varphi'(\xi)\left(6c_3 + 24c_4e_{k,v} + O(e_{k,v}^2)\right), \tag{2.42}$$

$$\varphi^{(iv)}(t_k) = \varphi'(\xi)\left(24c_4 + 120c_5e_{k,t} + O(e_{k,t}^2)\right). \tag{2.43}$$

Similarly,

$$\varphi^{(6)}(\delta) = \varphi'(\xi)\left(6!c_6 + 7!c_7e_\delta + O(e_\delta^2)\right), \tag{2.44}$$

$$\varphi^{(7)}(\delta) = \varphi'(\xi)\left(7!c_7 + 8!c_8e_\delta + O(e_\delta^2)\right), \tag{2.45}$$

$$\varphi^{(8)}(\delta) = \varphi'(\xi)\left(8!c_8 + 9!c_9e_\delta + O(e_\delta^2)\right), \tag{2.46}$$

where

$$e_\delta = \delta - \xi.$$

Putting Eqs (2.43) and (2.46) in (2.38); (2.42) and (2.45) in (2.39); (2.41) and (2.44) in (2.40), we obtained the following equations:

$$H_7^{(iv)}(t_k) = 24\varphi'(\xi)(c_4 - c_8e_{k-1,v}e_{k-1,t}e_{k-1}^2), \tag{2.47}$$

$$H_6'''(v_k) = 6\varphi'(\xi)(c_3 - c_7e_{k-1,v}e_{k-1,t}e_{k-1}^2) \tag{2.48}$$

and

$$H_5''(s_k) = 2\varphi'(\xi)(c_2 - c_6e_{k-1,v}e_{k-1,t}e_{k-1}^2). \tag{2.49}$$

Now, using Eqs (2.40) and (2.47)–(2.49), we obtain

$$\frac{H_7^{(iv)}(t_k)}{4H_6'''(v_k)} - \frac{H_5''(s_k)}{2\varphi'(s_k)} \sim \frac{c_4}{c_3} - c_2 + \left(\frac{c_4c_7}{c_3^2} - \frac{c_8}{c_3} + \frac{c_8c_7}{c_3^2}e_{k-1,v}e_{k-1,t}e_{k-1}^2 + c_6\right)e_{k-1,v}e_{k-1,t}e_{k-1}^2.$$

And hence

$$\beta_k \sim \frac{c_4}{c_3} - c_2 + \left(\frac{c_4c_7}{c_3^2} - \frac{c_8}{c_3} + \frac{c_8c_7}{c_3^2}e_{k-1,v}e_{k-1,t}e_{k-1}^2 + c_6\right)e_{k-1,v}e_{k-1,t}e_{k-1}^2$$

or

$$\beta_k - \frac{c_4}{c_3} + c_2 \sim \left(\frac{c_4c_7}{c_3} - c_8 + \frac{c_8c_7}{c_3}e_{k-1,v}e_{k-1,t}e_{k-1}^2 + c_6c_3\right)e_{k-1,v}e_{k-1,t}e_{k-1}^2. \tag{2.50}$$

The above Eq (2.50) can also be written as

$$(\beta_k + c_2)c_3 - c_4 \sim \left(\frac{c_4c_7}{c_3} - c_8 + \frac{c_8c_7}{c_3}e_{k-1,v}e_{k-1,t}e_{k-1}^2 + c_6c_3\right)e_{k-1,v}e_{k-1,t}e_{k-1}^2. \tag{2.51}$$

This completes the proof of Theorem 2.4. □

Presently, for the new iterative scheme with memory (2.33), we can state the subsequent convergence theorem.

Theorem 2.5. *In the iterative method (2.33), let β_k be the varying parameter calculated by Eq (2.34). If an initial guess s_0 is enough near to a simple zero ξ of $\varphi(s)$, then the R-order of convergence of the iterative method (2.33) with memory is at least 10.7446.*

Proof. Let the iterative method generates the sequence of $\{s_k\}$ which converges to the root ξ of $\varphi(s)$, by means of R-order

$$O_R(IM, \xi) \geq r,$$

we express

$$e_{k+1} \sim D_{k,r} e_k^r$$

and

$$e_k \sim D_{k-1,r} e_{k-1}^r. \tag{2.52}$$

Next, $D_{k,r}$ will tends to the asymptotic error constant D_r of IM by taking $k \to \infty$, then

$$e_{k+1} \sim D_{k,r}(D_{k-1,r} e_{k-1}^r)^r = D_{k,r} D_{k-1,r}^r e_{k-1}^{r^2}. \tag{2.53}$$

The resulting error expression of the with-memory scheme (2.33) can be obtained using (2.30)–(2.32) and the varying parameter β_k,

$$e_{k,v} = v_k - \xi \sim (\alpha + c_2)\, e_k^2, \tag{2.54}$$
$$e_{k,t} = t_k - \xi \sim -(\alpha + c_2)\, c_3 e_k^4 \tag{2.55}$$

and

$$e_{k+1} = s_{k+1} - \xi \sim (\alpha + c_2)^2\, c_3\, ((\beta + c_2)\, c_3 - c_4)\, e_k^8. \tag{2.56}$$

Here, the higher-order terms in Eqs (2.54)–(2.56) are excluded.

Now, let the R-order convergence of the iterative sequences $\{v_k\}$ and $\{t_k\}$ be p and q, respectively, then

$$e_{k,v} \sim D_{k,p} e_k^p \sim D_{k,p}(D_{k-1,r} e_{k-1}^r)^p = D_{k,p} D_{k-1,r}^p e_{k-1}^{rp} \tag{2.57}$$

and

$$e_{k,t} \sim D_{k,q} e_k^q \sim D_{k,q}(D_{k-1,r} e_{k-1}^r)^q = D_{k,q} D_{k-1,r}^q e_{k-1}^{rq}. \tag{2.58}$$

Now, by Eqs (2.17), (2.52), and (2.54), we obtain

$$\begin{aligned} e_{k,v} &\sim (\alpha + c_2)\, e_k^2 \\ &\sim c_6 e_{k-1,v} e_{k-1,t} e_{k-1}^2 (D_{k-1,r} e_{k-1}^r)^2 \\ &\sim c_6 (D_{k-1,p} e_{k-1}^p)(D_{k-1,q} e_{k-1}^q) e_{k-1}^2 D_{k-1,r}^2 e_{k-1}^{2r} \\ &\sim c_6 D_{k-1,p} D_{k-1,q} D_{k-1,r}^2 e_{k-1}^{p+q+2r+2}. \end{aligned} \tag{2.59}$$

Also, by Eqs (2.17), (2.52), and (2.55), we obtain

$$\begin{aligned} e_{k,t} &\sim -(\alpha + c_2)\, c_3 e_k^4 \\ &\sim -c_3 c_6 e_{k-1,v} e_{k-1,t} e_{k-1}^2 (D_{k-1,r} e_{k-1}^r)^4 \\ &\sim -c_3 c_6 (D_{k-1,p} e_{k-1}^p)(D_{k-1,q} e_{k-1}^q) e_{k-1}^2 D_{k-1,r}^4 e_{k-1}^{4r} \\ &\sim -c_3 c_6 D_{k-1,p} D_{k-1,q} D_{k-1,r}^4 e_{k-1}^{p+q+4r+2}. \end{aligned} \tag{2.60}$$

Again, by Eqs (2.17), (2.51), (2.52) and (2.56), we have

$$\begin{aligned} e_{k+1} &\sim (\alpha + c_2)^2\, c_3\, ((\beta + c_2)\, c_3 - c_4)\, e_k^8 \\ &\sim c_3 \left(c_6 e_{k-1,v} e_{k-1,t} e_{k-1}^2\right)^2 \left(\frac{c_4 c_7}{c_3} - c_8 + \frac{c_8 c_7}{c_3} e_{k-1,v} e_{k-1,t} e_{k-1}^2 + c_6 c_3\right) e_{k-1,v} e_{k-1,t} e_{k-1}^2 (D_{k-1,r} e_{k-1}^r)^8 \\ &\sim c_3 c_6^2 \left(\frac{c_4 c_7}{c_3} - c_8 + \frac{c_8 c_7}{c_3} e_{k-1,v} e_{k-1,t} e_{k-1}^2 + c_6 c_3\right) e_{k-1,v}^3 e_{k-1,t}^3 e_{k-1}^6 D_{k-1,r}^8 e_{k-1}^{8r} \\ &\sim c_3 c_6^2 \left(\frac{c_4 c_7}{c_3} - c_8 + \frac{c_8 c_7}{c_3} e_{k-1,v} e_{k-1,t} e_{k-1}^2 + c_6 c_3\right) (D_{k-1,p} e_{k-1}^p)^3 (D_{k-1,q} e_{k-1}^q)^3 e_{k-1}^6 D_{k-1,r}^8 e_{k-1}^{8r} \\ &\sim c_3 c_6^2 \left(\frac{c_4 c_7}{c_3} - c_8 + \frac{c_8 c_7}{c_3} e_{k-1,v} e_{k-1,t} e_{k-1}^2 + c_6 c_3\right) D_{k-1,p}^3 e_{k-1}^{3p} D_{k-1,q}^3 e_{k-1}^{3q} e_{k-1}^6 D_{k-1,r}^8 e_{k-1}^{8r} \\ &\sim c_3 c_6^2 \left(\frac{c_4 c_7}{c_3} - c_8 + \frac{c_8 c_7}{c_3} e_{k-1,v} e_{k-1,t} e_{k-1}^2 + c_6 c_3\right) D_{k-1,p}^3 D_{k-1,q}^3 D_{k-1,r}^8 e_{k-1}^{3p+3q+8r+6}, \end{aligned} \tag{2.61}$$

since $r > q > p$.

By equating the exponents of e_{k-1} present in the set of relations (2.57)–(2.59), (2.58)–(2.60), and (2.53)–(2.61), we attain the resulting system of equations:

$$\begin{aligned} rp &= p + q + 2r + 2, \\ rq &= p + q + 4r + 2, \\ r^2 &= 3p + 3q + 8r + 6. \end{aligned} \tag{2.62}$$

The solution of the system of Eq (2.62) is specified by

$$p = 2.9148, \quad q = 4.9148 \quad \text{and} \quad r = 10.7446.$$

As a result, the R-order of convergence of the with-memory iterative method (2.33) is at least 10.7446.

This completes the proof. □

3. Numerical discussion

This section examines the convergence behaviors of the newly proposed with-memory methods NWM10 and NWM11 introduced in (2.7) and (2.33), respectively. Our goal is to evaluate the effectiveness of the recently proposed iterative methods by applying them to a variety of nonlinear problems. The nonlinear test functions, along with their initial guesses and roots for our numerical analysis, are described below:

Example 1. $\varphi_1(s) = s^{10} + 4s^5 - 15s^2 + 2$, $s_0 = 1.2$, $\xi \approx 1.24903$.

Example 2. $\varphi_2(s) = (\cos s)^2 - \sin s + s$, $s_0 = -0.9$, $\xi \approx -1.09775$.

Example 3. $\varphi_3(s) = e^{s-5} - s^3 + \sin s - 1$, $s_0 = -1.1$, $\xi \approx -1.24861$.

Example 4. $\varphi_4(s) = \sin(s^3 + s^2 + 1) + s^2 - 5$, $s_0 = 2.3$, $\xi \approx 2.30388$.

Example 5. $\varphi_5(s) = e^{s^2} - s^3 + s^2 + s - 7$, $s_0 = 1.3$, $\xi \approx 1.35666$.

Example 6. $\varphi_6(s) = s^7 - 4s^4 + s - 1$, $s_0 = 1.5$, $\xi \approx 1.57494$.

Example 7. $\varphi_7(s) = e^{s^3+\cos s+1} - s^2 + s + 1$, $s_0 = -1.2$, $\xi \approx -1.07875$.

Example 8. $\varphi_8(s) = e^{s^2} + \sin s - \cos s - 1$, $s_0 = 0.4$, $\xi \approx 0.54177$.

We will compare our newly proposed methods NWM10 (2.7) and NWM11 (2.33) to various well-established methods published in the literature, including SH6 (3.1), SH8 (3.2), HS10 (3.3), CJ10 (3.4), WZ10 (3.5), and CCJT10 (3.6), which are discussed below:

In 2016, Solaiman and Hashim (SH6) [16] developed a sixth-order iterative method, which is defined as:

$$\begin{aligned} v_k &= s_k - \frac{\varphi(s_k)}{\varphi'(s_k)}, \\ s_{k+1} &= v_k - \frac{\varphi(v_k)}{\varphi'(v_k)} - \frac{2\left(\varphi(v_k)\right)^2 \varphi'(v_k) Q(s_k, v_k)}{4\left(\varphi'(v_k)\right)^4 - 4\varphi(v_k)\left(\varphi'(v_k)\right)^2 Q(s_k, v_k) + \left(\varphi(v_k)\right)^2 \left(Q(s_k, v_k)\right)^2}, \end{aligned} \tag{3.1}$$

where,

$$Q(s_k, v_k) = \frac{2}{s_k - v_k}\left(3\frac{\varphi(s_k) - \varphi(v_k)}{s_k - v_k} - 2\varphi'(v_k) - \varphi'(s_k)\right) = \varphi''(v_k).$$

In 2020, Solaiman and Hashim (SH8) [12] developed an optimal eighth-order iterative method, which is defined as:

$$\begin{aligned} v_k &= s_k - \frac{\varphi(s_k)}{\varphi'(s_k)}, \\ t_k &= v_k - \frac{\varphi(v_k)}{\varphi'(v_k)} - \frac{2(\varphi(v_k))^2 \varphi'(v_k) R(s_k, v_k)}{4(\varphi'(v_k))^4 - 4\varphi(v_k)(\varphi'(v_k))^2 R(s_k, v_k) + (\varphi(v_k))^2 (R(s_k, v_k))^2}, \\ s_{k+1} &= t_k - \frac{\varphi(t_k)}{\varphi'(t_k)}, \end{aligned} \tag{3.2}$$

where, the approximate values of $\varphi'(v_k)$, $\varphi'(t_k)$ and $R(s_k, v_k)$ are given in Eq (2.1).

In 2017, Husayni and Subaihi (HS10) [17] developed a tenth-order iterative method, which is defined as:

$$\begin{aligned} v_k &= s_k - \frac{\varphi(s_k)}{\varphi[G_k, s_k]}, \\ t_k &= v_k - \left(\frac{\varphi(s_k)\varphi(G_k)}{\varphi(v_k) - \varphi(s_k)}\right)\left(\frac{1}{\varphi[G_k, s_k]} - \frac{1}{\varphi[G_k, v_k]}\right), \\ s_{k+1} &= t_k - \frac{\varphi(t_k)}{2\varphi[v_k, s_k] - \varphi[G_k, s_k]} - w(t_k)\frac{\varphi\left(t_k - \frac{\varphi(t_k)}{2\varphi[v_k, s_k] - \varphi[G_k, s_k]}\right)}{2\varphi[v_k, s_k] - \varphi[G_k, s_k]}, \end{aligned} \tag{3.3}$$

where

$$G_k = s_k + \varphi(s_k)^3$$

and

$$t_k = \frac{\varphi\left(t_k - \frac{\varphi(t_k)}{2\varphi[v_k,s_k]-\varphi[G_k,s_k]}\right)}{\varphi(t_k)}.$$

In 2016, Choubey and Jaiswal (CJ10) [18] developed a bi-parametric with-memory iterative method with a tenth order of convergence, which is defined as:

$$\begin{aligned} v_k &= s_k - \frac{\varphi(s_k)}{\varphi'(s_k) - T\varphi(s_k)}, \\ t_k &= v_k - \left(\frac{\varphi(v_k)(\varphi(s_k) + \gamma\varphi(v_k))}{(\varphi'(s_k) - 2t_k\varphi(s_k))(\varphi(s_k) + (\gamma-2)\varphi(v_k))}\right), \\ s_{k+1} &= t_k - \frac{\varphi(t_k)}{\varphi[t_k, v_k] + \varphi[t_k, v_k, s_k](t_k - v_k) + \varphi[t_k, v_k, s_k, s_k](t_k - v_k)(t_k - s_k)}, \end{aligned} \tag{3.4}$$

where $T, \gamma \in \mathrm{R}$, and T is calculated as

$$T = \frac{H_5''(s_k)}{2\varphi'(s_k)}.$$

In 2013, Wang and Zhang (WZ10) [19] developed a tenth-order family of three-step with-memory iterative schemes using one self-accelerating parameter, which is defined as:

$$\begin{aligned} v_k &= s_k - \frac{\varphi(s_k)}{\varphi'(s_k) - t_k\varphi(s_k)}, \\ t_k &= v_k - \left(\frac{\varphi(v_k)}{2\varphi[s_k, v_k] - \varphi'(s_k) + t_k\varphi(v_k)}\right), \\ s_{k+1} &= t_k - [G(s_k) + H(t_k)]\left(\frac{(\alpha + w)\varphi(t_k)}{2w\varphi[v_k, t_k] + (\alpha - w)(\varphi'(s_k) + L\varphi(t_k))}\right), \end{aligned} \tag{3.5}$$

where

$$s_k = \frac{\varphi(t_k)}{\varphi(s_k)}, \quad t_k = \frac{\varphi(v_k)}{\varphi(s_k)}, \quad \alpha = v_k - s_k, \quad w = t_k - s_k$$

and $L \in R$. Also, t_k is calculated as

$$t_k = -\frac{H_5''(s_k)}{2\varphi'(s_k)}.$$

In 2018, Choubey et al. (CCJT10) [20] proposed a tenth-order with-memory iterative method using two self-accelerating parameters, which is defined as:

$$\begin{aligned} v_k &= s_k - \frac{\varphi(s_k)}{\varphi'(s_k) - \gamma_k\varphi(s_k)}, \\ t_k &= v_k - \left(\frac{\varphi(v_k)}{-\varphi'(s_k) + 2((\varphi(v_k) - \varphi(s_k))/(v_k - s_k))}\right), \\ s_{k+1} &= t_k - \varphi(t_k)\left(-\frac{\varphi(v_k) - \varphi(s_k)}{v_k - s_k} + \frac{\varphi(t_k) - \varphi(v_k)}{t_k - v_k} + \frac{\varphi(t_k) - \varphi(s_k)}{t_k - s_k} + \lambda_k(t_k - s_k)(t_k - v_k)\right), \end{aligned} \tag{3.6}$$

where γ, $\lambda \in R$ and are calculated as

$$\gamma_k = \frac{H_5''(s_k)}{2\varphi'(s_k)}$$

and

$$\lambda_k = \frac{H_6'''(v_k)}{6}.$$

The comparative outcomes of every strategy are summarized in Tables 1–8. The absolute differences ($|s_k - s_{k-1}|$) between the last two iterations and the absolute residual error ($|\varphi(s_k)|$) of up to three iterations for each function are shown in these tables, along with the COC for the proposed methods in comparison to some well-known existing methods. The COC [21] is found using the following equation:

$$COC = \frac{log|\varphi(s_k)/\varphi(s_{k-1})|}{log|\varphi(s_{k-1})/\varphi(s_{k-2})|}.$$

All numerical computations were performed using the programming software Mathematica 12.2. To begin the initial iterations for our newly proposed with-memory methods NWM10 and NWM11, we set the parameter values as

$$\alpha_0 = 0.01$$

and

$$\beta_0 = 0.00001.$$

Table 1. Comparison results of the with- and without-memory methods after three iterations for $\varphi_1(s)$.

Method	$\lvert(s_1 - s_0)\rvert$	$\lvert(s_2 - s_1)\rvert$	$\lvert(s_3 - s_2)\rvert$	$\lvert\varphi_1(s_3)\rvert$	COC
SH6	0.04903	5.4760×10^{-6}	7.6713×10^{-30}	4.9394×10^{-171}	6.0000
SH8	0.04903	5.4670×10^{-8}	1.6829×10^{-55}	1.1557×10^{-433}	8.0000
HS10	*div.*	*div.*	*div.*	*div.*	*div.*
CJ10	0.04904	1.2344×10^{-5}	6.6291×10^{-46}	1.0292×10^{-446}	10.0000
WZ10	0.04903	5.5393×10^{-6}	7.1079×10^{-49}	7.1557×10^{-476}	10.0000
CCJT10	0.04903	1.1312×10^{-6}	5.5989×10^{-56}	3.9938×10^{-547}	10.0000
NWM10	0.04903	8.5737×10^{-8}	1.5694×10^{-67}	5.6911×10^{-663}	10.0000
NWM11	0.04903	8.5738×10^{-8}	4.3360×10^{-74}	7.5373×10^{-786}	10.7430

Table 2. Comparison results of the with- and without-memory methods after three iterations for $\varphi_2(s)$.

Method	$\lvert(s_1 - s_0)\rvert$	$\lvert(s_2 - s_1)\rvert$	$\lvert(s_3 - s_2)\rvert$	$\lvert\varphi_2(s_3)\rvert$	COC
SH6	0.19775	1.1574×10^{-8}	2.0722×10^{-51}	9.2574×10^{-308}	6.0000
SH8	0.19775	7.4544×10^{-11}	2.8192×10^{-85}	1.5995×10^{-680}	8.0000
HS10	0.19775	2.3574×10^{-10}	6.4022×10^{-101}	1.3974×10^{-1006}	10.0000
CJ10	0.19775	7.8166×10^{-11}	1.6451×10^{-107}	9.0133×10^{-1072}	10.0000
WZ10	0.19775	3.3270×10^{-11}	2.2829×10^{-108}	4.4311×10^{-1080}	10.0000
CCJT10	0.19775	2.6059×10^{-9}	1.0304×10^{-88}	2.7635×10^{-882}	10.0000
NWM10	0.19775	4.3937×10^{-11}	1.2093×10^{-109}	4.1537×10^{-1093}	10.0000
NWM11	0.19775	4.3933×10^{-11}	2.8779×10^{-119}	1.7784×10^{-1278}	10.7430

Table 3. Comparison results of the with- and without-memory methods after three iterations for $\varphi_3(s)$.

Method	$\lvert(s_1 - s_0)\rvert$	$\lvert(s_2 - s_1)\rvert$	$\lvert(s_3 - s_2)\rvert$	$\lvert\varphi_3(s_3)\rvert$	COC
SH6	0.14860	4.1118×10^{-6}	1.1010×10^{-33}	1.7692×10^{-198}	6.0000
SH8	0.14861	1.2389×10^{-8}	3.0525×10^{-65}	1.8075×10^{-517}	8.0000
HS10	0.14861	1.0345×10^{-6}	2.6888×10^{-65}	3.7848×10^{-651}	10.0000
CJ10	0.14862	6.4305×10^{-6}	3.4403×10^{-54}	2.8696×10^{-536}	10.0000
WZ10	0.14861	3.8388×10^{-6}	4.8325×10^{-56}	2.1019×10^{-554}	10.0000
CCJT10	0.14861	6.7601×10^{-7}	5.2500×10^{-64}	1.8179×10^{-634}	10.0000
NWM10	0.14861	2.0994×10^{-8}	4.7148×10^{-79}	6.7060×10^{-785}	10.0000
NWM11	0.14861	2.0993×10^{-8}	1.1393×10^{-86}	1.5623×10^{-932}	10.7390

Table 4. Comparison results of the with- and without-memory methods after three iterations for $\varphi_4(s)$.

Method	$\lvert(s_1 - s_0)\rvert$	$\lvert(s_2 - s_1)\rvert$	$\lvert(s_3 - s_2)\rvert$	$\lvert\varphi_4(s_3)\rvert$	COC
SH6	0.00388	9.8385×10^{-13}	5.8561×10^{-71}	6.2869×10^{-419}	6.0000
SH8	0.00388	1.5938×10^{-16}	1.4716×10^{-123}	1.8763×10^{-978}	8.0000
HS10	0.00388	1.6421×10^{-13}	5.0425×10^{-133}	3.7588×10^{-1328}	10.0000
CJ10	0.00388	6.4472×10^{-15}	4.4549×10^{-136}	9.1833×10^{-1345}	10.0000
WZ10	0.00388	9.4199×10^{-16}	7.5075×10^{-143}	1.1651×10^{-1412}	10.0000
CCJT10	0.00388	5.8641×10^{-13}	3.4760×10^{-115}	6.6135×10^{-1136}	10.0000
NWM10	0.00388	1.6069×10^{-16}	2.2318×10^{-150}	9.1434×10^{-1488}	10.0000
NWM11	0.00188	2.2997×10^{-16}	3.1977×10^{-163}	2.6116×10^{-1738}	10.7350

Table 5. Comparison results of the with- and without-memory methods after three iterations for $\varphi_5(s)$.

Method	$\lvert(s_1 - s_0)\rvert$	$\lvert(s_2 - s_1)\rvert$	$\lvert(s_3 - s_2)\rvert$	$\lvert\varphi_5(s_3)\rvert$	COC
SH6	0.05665	1.4425×10^{-7}	4.1391×10^{-41}	3.5304×10^{-241}	6.0000
SH8	0.05665	9.1131×10^{-10}	5.2988×10^{-72}	1.0583×10^{-568}	8.0000
HS10	0.05665	4.9575×10^{-2}	2.4135×10^{-32}	7.4443×10^{-325}	10.0000
CJ10	0.05665	3.9801×10^{-8}	1.7264×10^{-74}	7.1328×10^{-737}	10.0000
WZ10	0.05665	6.9055×10^{-9}	1.2500×10^{-81}	2.5734×10^{-808}	10.0000
CCJT10	0.05665	7.0088×10^{-9}	1.7572×10^{-81}	5.1691×10^{-807}	10.0000
NWM10	0.05665	9.9658×10^{-10}	2.4061×10^{-90}	1.9718×10^{-895}	10.0000
NWM11	0.05665	9.9659×10^{-10}	9.3351×10^{-99}	1.7551×10^{-1051}	10.7150

Table 6. Comparison results of the with- and without-memory methods after three iterations for $\varphi_6(s)$.

Method	$\lvert(s_1 - s_0)\rvert$	$\lvert(s_2 - s_1)\rvert$	$\lvert(s_3 - s_2)\rvert$	$\lvert\varphi_6(s_3)\rvert$	COC
SH6	0.07491	2.8459×10^{-5}	5.5745×10^{-26}	1.4266×10^{-148}	6.0000
SH8	0.07494	2.6904×10^{-7}	1.2077×10^{-50}	9.0275×10^{-396}	8.0000
HS10	11.61089	1.2106	8.3766×10^{-2}	2.6580×10^{-16}	10.3150
CJ10	0.07513	1.8605×10^{-4}	7.4813×10^{-35}	3.7413×10^{-337}	10.0000
WZ10	0.07484	9.8996×10^{-5}	4.0035×10^{-37}	2.1170×10^{-359}	10.0000
CCJT10	0.07495	1.0136×10^{-5}	2.7132×10^{-47}	2.3084×10^{-461}	10.0000
NWM10	0.07494	6.8053×10^{-7}	3.1990×10^{-59}	7.6445×10^{-581}	10.0000
NWM11	0.07494	6.8053×10^{-7}	7.0208×10^{-65}	2.0005×10^{-691}	10.8340

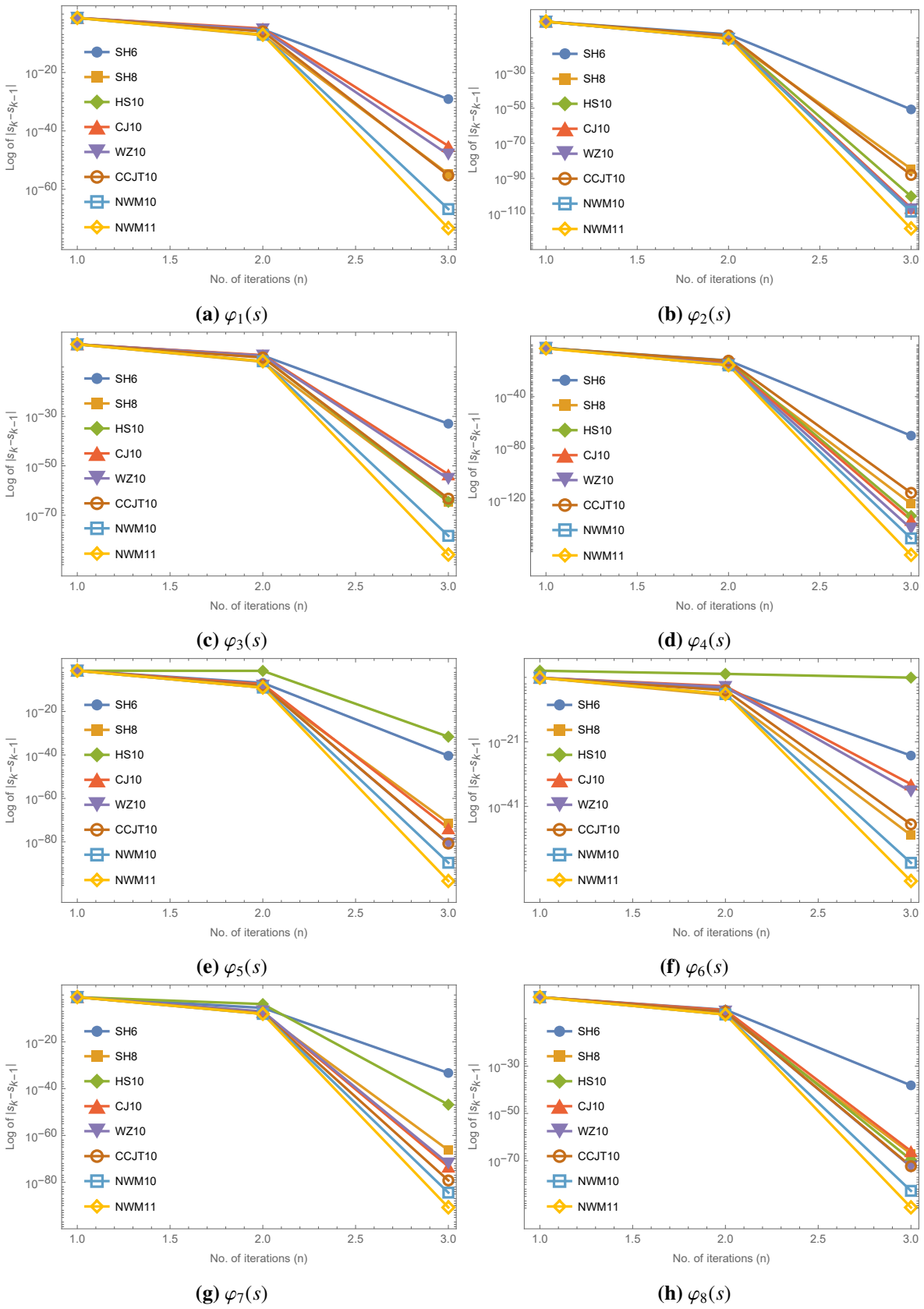
Table 7. Comparison results of the with- and without-memory methods after three iterations for $\varphi_7(s)$.

Method	$\lvert(s_1 - s_0)\rvert$	$\lvert(s_2 - s_1)\rvert$	$\lvert(s_3 - s_2)\rvert$	$\lvert\varphi_7(s_3)\rvert$	COC
SH6	0.12125	2.9952×10^{-6}	4.8045×10^{-34}	7.0299×10^{-200}	6.0000
SH8	0.12125	7.7131×10^{-9}	4.7932×10^{-67}	9.1585×10^{-532}	8.0000
HS10	0.12125	1.3062×10^{-4}	1.4202×10^{-47}	3.3274×10^{-477}	10.0000
CJ10	0.12125	5.3825×10^{-8}	4.7085×10^{-74}	1.6972×10^{-735}	10.0000
WZ10	0.12125	9.5115×10^{-8}	9.1359×10^{-73}	1.8214×10^{-722}	10.0000
CCJT10	0.12125	1.6076×10^{-8}	5.6170×10^{-80}	1.0041×10^{-792}	10.0010
NWM10	0.12125	8.4570×10^{-9}	3.5933×10^{-85}	1.1373×10^{-846}	10.0000
NWM11	0.12125	8.4571×10^{-9}	1.6632×10^{-91}	1.3380×10^{-979}	10.7490

Table 8. Comparison results of the with- and without-memory methods after three iterations for $\varphi_8(s)$.

Method	$\lvert(s_1 - s_0)\rvert$	$\lvert(s_2 - s_1)\rvert$	$\lvert(s_3 - s_2)\rvert$	$\lvert\varphi_8(s_3)\rvert$	COC
SH6	0.14177	6.8830×10^{-7}	7.3009×10^{-39}	2.9384×10^{-230}	6.0000
SH8	0.14177	6.3516×10^{-9}	9.7114×10^{-68}	8.1960×10^{-538}	8.0000
HS10	0.14177	2.9779×10^{-7}	2.7815×10^{-70}	1.4059×10^{-700}	10.0000
CJ10	0.14177	4.5598×10^{-7}	1.1993×10^{-66}	6.0622×10^{-660}	10.0000
WZ10	0.14177	1.0462×10^{-7}	2.8607×10^{-73}	3.1584×10^{-727}	10.0000
CCJT10	0.14177	8.8001×10^{-8}	5.1277×10^{-73}	2.1772×10^{-725}	9.9995
NWM10	0.14177	6.2949×10^{-9}	1.9362×10^{-83}	7.2933×10^{-828}	10.0000
NWM11	0.14177	6.2947×10^{-9}	1.8530×10^{-90}	1.6124×10^{-965}	10.7380

Comparing the newly proposed with-memory methods NWM10 and NWM11 to the other existing methods, the numerical results in Tables 1–8 and Figure 1 demonstrate that they are highly competitive and have fast convergence towards the roots with minimal absolute residual error and a minimum error value in consecutive iterations. Additionally, the numerical results demonstrate that the computational order of convergence in the test functions is consistent with the theoretical convergence order of the newly proposed methods.



(a) $\varphi_1(s)$ **(b)** $\varphi_2(s)$

(c) $\varphi_3(s)$ **(d)** $\varphi_4(s)$

(e) $\varphi_5(s)$ **(f)** $\varphi_6(s)$

(g) $\varphi_7(s)$ **(h)** $\varphi_8(s)$

Figure 1. Comparison of the methods after the first three iterations based on the error in successive iterations, $|s_k - s_{k-1}|$.

4. Conclusions

This article presents three-step with-memory iterative techniques that use single and double self-accelerating parameters to solve nonlinear equations. The main objective is to improve the optimal eighth-order method's convergence order without involving more calculations. In order to do this, the eighth-order technique incorporates self-acceleration parameters and their estimates. The Hermite interpolating polynomial is used to get the estimations of these self-accelerating parameters. When parameters are included, the with-memory methods NWM10 and NWM11's R-order of convergence rise from 8 to 10 and 10.7446, respectively. The findings demonstrate that compared to other existing methods, the proposed NWM10 and NWM11 techniques have smaller asymptotic constant values and faster convergence. Additionally, the newly introduced techniques perform exceptionally well overall and have a fast rate of convergence, making them a viable substitute for solving nonlinear equations.

The presented techniques can be used by the interested researchers to improve efficiency for single or multivariate functions by extending well-known higher optimal order without-memory iterative methods to with-memory algorithms along with single or multi self-accelerating parameters.

Author contributions

S. K. Mittal: conceptualization, methodology, software, validation, formal analysis, resources, writing–original draft preparation, writing–review and editing, visualization; S. Panday: methodology, software, validation, formal analysis, writing–original draft preparation, writing–review and editing, visualization, supervision; L. Jäntschi: software, formal analysis, writing–review and editing; L. C. Bolunduţ: formal analysis, writing–review and editing. All authors have read and agreed to the published version of the manuscript.

Use of Generative-AI tools declaration

The authors declare they have not used Artificial Intelligence (AI) tools in the creation of this article.

Conflict of interest

The authors declare no conflicts of interest.

References

1. A. Cordero, J. R. Torregrosa, On the design of optimal iterative methods for solving nonlinear equations, In: S. Amat, S. Busquier, *Advances in iterative methods for nonlinear equations*, 2016, 79–111. https://doi.org/10.1007/978-3-319-39228-8_5
2. R. Sharma, A. Bahl, An optimal fourth order iterative method for solving nonlinear equations and its dynamics, *J. Complex Anal.*, 2015, 259167. https://doi.org/10.1155/2015/259167
3. J. L. Varona, An optimal thirty-second-order iterative method for solving nonlinear equations and a conjecture, *Qual. Theory Dyn. Syst.*, **21** (2022), 39. https://doi.org/10.1007/s12346-022-00572-3

4. G. Zhang, Y. Zhang, H. Ding, New family of eighth-order methods for nonlinear equation, *Int. J. Comput. Math. Electr. Electron. Eng.*, **28** (2009), 1418–1427. https://doi.org/10.1108/03321640910991985

5. J. F. Traub, Iterative methods for the solution of equations, *Amer. Math. Soc.*, **312** (1982), 151. https://doi.org/10.2307/2004117

6. X. Wang, Y. Tao, A new Newton method with memory for solving nonlinear equations, *Mathematics*, **8** (2020), 108. https://doi.org/10.3390/math8010108

7. V. Torkashvand, A two-step method adaptive with memory with eighth-order for solving nonlinear equations and its dynamic, *Comput. Methods Differ. Equations*, **10** (2022), 1007–1026. https://doi.org/10.22034/cmde.2022.46651.1961

8. G. Thangkhenpau, S. Panday, S. K. Mittal, L. Jäntschi, Novel parametric families of with and without memory iterative methods for multiple roots of nonlinear equations, *Mathematics*, **11** (2023), 2036. https://doi.org/10.3390/math11092036

9. T. Lotfi, P. Assari, A new two step class of methods with memory for solving nonlinear equations with high efficiency index, *Int. J. Math. Modell. Comput.*, **4** (2014), 277–288.

10. S. Panday, S. K. Mittal, C. E. Stoenoiu, L. Jäntschi, A new adaptive eleventh-order memory algorithm for solving nonlinear equations, *Mathematics*, **12** (2024), 1809. https://doi.org/10.3390/math12121809

11. S. K. Mittal, S. Panday, L. Jäntschi, Enhanced ninth-order memory-based iterative technique for efficiently solving nonlinear equations, *Mathematics*, **12** (2024), 3490. https://doi.org/10.3390/math12223490

12. O. S. Solaiman, I. Hashim, Optimal eighth-order solver for nonlinear equations with applications in chemical engineering, *Intell. Autom. Soft Comput.*, **27** (2021), 379–390. https://doi.org/10.32604/iasc.2021.015285

13. J. M. Ortega, W. C. Rheinboldt, *Iterative solution of nonlinear equations in several variables*, Society for Industrial and Applied Mathematics, 2000.

14. G. Alefeld, J. Herzberger, *Introduction to interval computation*, Academic Press, 2012.

15. N. Kumar, A. Cordero, J. P. Jaiswal, J. R. Torregrosa, An efficient extension of three-step optimal iterative scheme into with memory and its stability, *J. Anal.*, 2024. https://doi.org/10.1007/s41478-024-00833-1

16. O. S. Solaiman, I. Hashim, Two new efficient sixth order iterative methods for solving nonlinear equations, *J. King Saud Univ. Sci.*, **31** (2019), 701–705. https://doi.org/10.1016/j.jksus.2018.03.021

17. Y. Y. Al-Husayni, I. A. Al-Subaihi, Tenth-order iterative methods without derivatives for solving nonlinear equations, *J. Res. Appl. Math.*, **3** (2017), 13–18.

18. N. Choubey, J. P. Jaiswal, Two-and three-point with memory methods for solving nonlinear equations, *Numer. Anal. Appl.*, **10** (2017), 74–89. https://doi.org/10.1134/S1995423917010086

19. X. Wang, T. Zhang, Some Newton-type iterative methods with and without memory for solving nonlinear equations, *Int. J. Comput. Methods*, **11** (2014), 1350078. https://doi.org/10.1142/S0219876213500783

20. N. Choubey, A. Cordero, J. P. Jaiswal, J. R. Torregrosa, Dynamical techniques for analyzing iterative schemes with memory, *Complexity*, 2018. https://doi.org/10.1155/2018/1232341

21. S. Weerakoon, T. G. I. Fernando, A variant of Newton's method with accelerated third-order convergence, *Appl. Math. Lett.*, **13** (2000), 87–93. https://doi.org/10.1016/S0893-9659(00)00100-2

AIMS Press