Research Paper
Computational Methods for Differential Equations
http://cmde.tabrizu.ac.ir
Vol. 13, No. 4, 2025, pp. 1289-1305
DOI:10.22034/cmde.2024.59161.2514

A new three-step optimal without memory iterative scheme for solving non-linear equations with basins of attraction

Shahid Abdullah[1,2,*], **Neha Choubey**[1], **and Suresh Dara**[1]

[1]School of Advanced Sciences and Languages, VIT Bhopal University, Kothri-Kalan, Sehore, 466114, MP, India.

[2]Department of Science and Humanities, J.B. Institute of Engineering & Technology, Bhaskar Nagar, Moinabad Mandal, Yekanpally, Hyderabad, Telangana, India.

Abstract

The primary focus of this study is to introduce a new three-step iterative method without memory for root-finding by merging two different existing techniques. Based on the computational cost, the proposed method acquires optimal eight-order convergence with four functional evaluations (three evaluations for the function and one computation of the first derivative). Furthermore, the suggested scheme supports Kung-Traub's Conjecture with an efficiency index of $8^{\frac{1}{4}} = 1.682$. We also established the convergence criteria developed for the root-finding technique and demonstrated the fact that the suggested approach is eighth-order convergent. In order to demonstrate the efficacy as well as application of the constructed root-finding technique, we addressed a few practical engineering models and some non-linear functions. In contrast to several existing approaches, this particular method converges more quickly. Finally, several forms of complex functions are taken into consideration under basins of attraction in order to observe the overall fractal behavior of the proposed technique.

Keywords. Iterative method, Non-linear equation, Gauss quadrature formula, Order of convergence, Basins of attraction.

2010 Mathematics Subject Classification. 65L05, 34K06, 34K28.

1. Introduction

The study of iterative methods for finding the simple roots of non-linear equations rapidly and accurately is one of the challenging and fascinating problems in the modern era. Although such kind of problems not occur only in applied mathematics alone but also appear in various disciplines of science and engineering. After discretization, many integral, partial differential, ordinary differential, and integro-differential equations also result in non-linear equations. Not only this non-linear phenomenon occur in a variety of other domains, including the problems related to economics modeling, transport theory, neuro-physiology, business, social sciences, kinematics, and so on. In general, we can say that by using distinct models, non linear phenomena are developed which contain non-linear equations. Solving such kind of problems is not always possible and when physical systems are mathematically modeled, some nonlinear equations do not have exact solutions. Therefore, relying on numerical methods based on iterative procedures, one can obtain the approximate solution. These iterative methods were developed using various existing techniques such as the homotopy perturbation method [26], Taylor's expansion [24, 25], multi-point iterative methods, Adomian decomposition method [1], variational iteration method, [11, 30] and quadrature formula [2, 19, 32].

Nowadays, there has been a surge of research interest in either developing new multi-point iterative methods or modifying with the existing ones for both with and without memory methods. The aim of developing these methods is to increase the order of convergence with a lesser number of functional evaluations at the same time. In other words, we can say that the primary aim of developing these methods is to achieve the maximum computational efficiency. Among all the methods, Newton's method [20] serves as the basic building block of almost all the higher-order iterative

Received: 10 November 2023; Accepted: 08 September 2024.
∗ Corresponding author. Email: shahidibnabdullah786@gmail.com.

methods for solving nonlinear equations due to its simplicity. In order to boost the order of convergence and efficiency index numerous adjustments have been made to Newton's approach with either increased cost of function evaluations or to the modifications of derivatives or by varying iteration points. Many researchers aim to enhance the efficiency of iterative methods by achieving a high order of convergence while minimizing the number of function evaluations in each iteration to make the iterative methods optimal [9, 28]. We must consider the idea of an efficiency index given by (E) given by $E = r^{\frac{1}{f}}$, variable r represents the convergence order, which relates to the speed at which an iterative method approaches a solution, while f denotes the count of evaluations needed for the functions. However, it is not always possible to develop methods with an optimal order of convergence keeping in view the hypothesis provided by Kung and Traub [15].

In [8], the authors have improved the order of convergence by blending two different methods having convergence orders s_1 and s_2 respectively to form a new method having order of convergence $s_1 s_2$. However, this strategy of course increases the order of convergence at each iteration step but with a larger number of function evaluations. Mir and Zaman [17], combining Ostrowski's approach and Halley's method. They introduced a parameter, leading to the development of families of iterative algorithms that exhibited convergence orders of sixth, seventh, and eighth. However, the efficiency indices were found to be $1.431, 1.383$, and 1.476 respectively. Recently, S. Qureshi et al. [24] merged a third-order Halley's scheme with the modified Newton's third order scheme results in obtaining a ninth order iterative scheme with six functional evaluations per iteration having efficiency index 1.4422. Another method was formed by the composition of Newton's method and fifth-order modified halley's method given by A. Tassaddiq et. al [31] having tenth-order convergence with the evaluations of three functions and three first-order derivatives. The efficiency index was found to be 1.4678. In the same way, Parhi et al. [22] proposed another sixth-order method by combining a third-order method with Newton's method and evaluated the first-order derivatives by linear interpolations having an efficiency index 1.565. In brief, a lot of iterative methods have been established based on the concept of blending different methods but at the same time with the lesser number of function evaluations.

Keeping in view the formation of aforementioned methods. We accomplished to build a new three-step numerical procedure by blending a fourth-order technique [14] with classical Newton's method. The combination produced a new eighth-order iterative approach with five functional evaluations. In order to minimize the number of function evaluations and align with the Kung Traub Conjecture, the utilization of the Gauss quadrature concept is employed for approximating the first derivative in the final stage. In section 2, we provide some of the existing eighth-order optimal iterative methods for comparison purposes. Construction of our proposed iterative method is discussed in section 4. Section 5 involves a graphical analysis that employs basins of attraction to investigate the dynamic characteristics of the proposed methods. In section 6, the implementation of the suggested methods on various nonlinear smooth functions is presented, showcasing their efficiency and improved performance through comparison with existing methods of similar orders. Finally, section 7 provides a summary of concluding remarks.

2. Literature Review

In this section, we will provide a brief review of a few popular numerical techniques that are frequently employed to find approximations of nonlinear model solutions. In its most basic form, $g(u) = 0$ can be used to denote a one-variable nonlinear equation, where the function $g(u)$ is defined on the real interval I and is sufficiently differentiable. Solving these equations and yielding exact solutions is not always possible. Therefore, numerical methods are employed in these situations in order to generate series of approximate solutions that eventually converge to the true solution of the nonlinear problem.

Newton's method [20] is one of the well-known iterative methods immediately comes into mind when one thinks to solve nonlinear equations. The method is optimal having quadratic convergence and additionally requires two evaluations of the functions given by:

$$u_{n+1} = u_n - \frac{g(u_n)}{g'(u_n)}, n = 0, 1, 2, ..., \tag{2.1}$$

where $g'(u_n) \neq 0$, and we denote (2.1) by $NN1$.

J. Dzunic and S. Petkovic [10] constructed a three point optimal iterative method for solving non linear equations by merging a two-step Ostrowski's fourth-order method with the modified Newton's method. As a result, only four function evaluations were needed per iteration. The method is given as:

$$\begin{aligned} v_n &= u_n - \frac{g(u_n)}{g'(u_n)}, \\ w_n &= v_n - \frac{g(u_n)}{g(u_n)-2g(v_n)}\frac{g(v_n)}{g'(u_n)}, \\ v_{n+1} &= w_n - \frac{(1+w_k)(1+2v_k)}{(1-2u_k-u_k^2)}\frac{g(w_n)}{g'(u_n)}, \end{aligned} \tag{2.2}$$

where $u_k = \frac{g(v_n)}{g(u_n)}, v_k = \frac{g(w_n)}{g(u_n)}$ and $w_k = \frac{g(w_n)}{g(v_n)}$. We denote (2.2) by $DP1$.

In 2022, S. Qureshi et al. [25] proposed a sixth-order three-point iterative method by blending Newton's method with the two-step existing. The method requires five function evaluations per iteration having efficiency index 1.4309 and is given as $SQ1$:

$$\begin{aligned} v_n &= u_n - \frac{g(u_n)}{g'(u_n)}, \\ w_n &= v_n - \frac{g(v_n)}{g'(v_n)}, \\ v_{n+1} &= w_n - \left(1 + 2\left(\frac{g(v_n)}{g(u_n)}\right)^2 + \frac{2g(w_n)}{g(v_n)}\right)\frac{g(w_n)}{g'(v_n)}. \end{aligned} \tag{2.3}$$

Sharma and Arora [27] presented an optimal three-step iterative method with eight-order convergence given by $JH1$:

$$\begin{aligned} v_n &= u_n - \frac{g(u_n)}{g'(u_n)}, \\ w_n &= v_n - \left(3 - 2\frac{g[v_n,u_n]}{g'(u_n)}\right)\frac{g(v_n)}{g'(u_n)}, \\ v_{n+1} &= w_n - \frac{g(w_n)}{g'(u_n)}\left(\frac{g'(u_n)-g[v_n,u_n]+g[w_n,v_n]}{2g[w_n,v_n]-g[w_n,u_n]}\right), \end{aligned} \tag{2.4}$$

where $g[.,.]$ is Newton's first order divided difference.

L. Liu and X. Wang [16] derived a three step iterative method for solving non linear equations. The method is optimal having efficiency index 1.682 given by $LX1$:

$$\begin{aligned} v_n &= u_n - \frac{g(u_n)}{g'(u_n)}, \\ w_n &= v_n - \frac{g(u_n)}{g(u_n)-2g(v_n)}\frac{g(v_n)}{g'(u_n)}, \\ v_{n+1} &= w_n - \frac{g(w_n)}{g'(u_n)}\left(\left(\frac{g(u_n)-g(v_n)}{g(u_n)-2g(v_n)}\right)^2 + \frac{g(w_n)}{g(v_n)-\alpha_1 g(w_n)} + \frac{4g(w_n)}{g(u_n)+\alpha_2 g(w_n)}\right), \end{aligned} \tag{2.5}$$

where $\alpha_1, \alpha_2 \in R$.

In 2021, B. Kong-ied [13] proposed an eight-order three-step iterative method without memory. The method involves the evaluation of five functions having an efficiency index of 1.5157 given as $KG1$:

$$\begin{aligned} v_n &= u_n - \frac{g(u_n)}{g'(u_n)}, \\ w_n &= v_n - \frac{(g(u_n))^2 g(v_n)}{(g(u_n))^2 g'(u_n) - 2g(u_n)g'(u_n)g(v_n) + g'(u_n)(g(v_n))^2}, \end{aligned} \tag{2.6}$$

$$v_{n+1} = w_n - \frac{g(w_n)}{g'(w_n)}.$$

N. Choubey and J. P. Jaiswal [7] modified an existing sixth-order method to optimal eight-order iterative method without memory with the help of weight function technique and is given as $NJ1$:

$$\begin{aligned} v_n &= u_n - \frac{g(u_n)}{g'(u_n)}, \\ w_n &= v_n - \frac{g(v_n)}{g'(u_n)} \frac{g(u_n)}{g(u_n) - 2g(v_n)}, \\ v_{n+1} &= w_n - \frac{g(w_n)(w_n - v_n)}{g(w_n) - g(v_n)}(A(t) + B(s) + H(q)), \end{aligned} \tag{2.7}$$

where $A(t), B(s)$, and $H(q)$ are weight functions with $t = \frac{g(w_n)}{g(v_n)}$, $s = \frac{g(w_n)}{g(u_n)}$ and $q = \frac{g(v_n)}{g(u_n)}$.

S. Parimala et al.[23] proposed an efficient three-step optimal-eighth order method for solving non linear equations given as $SP1$:

$$\begin{aligned} v_n &= u_n - \frac{g(u_n)}{g'(u_n)}, \\ w_n &= v_n - \frac{g(u_n)}{g'(u_n)} \frac{g(u_n) - g(v_n)}{g(u_n) - 2g(v_n)}, \\ v_{n+1} &= w_n - \frac{g(w_n)(w_n - v_n)}{g(w_n) - g(v_n)}(1 + 2\tau)\left(1 + \zeta^2 + 2\zeta^3 + \frac{7}{24}\zeta^4)\right), \end{aligned} \tag{2.8}$$

where $\tau = \frac{g(w_n)}{g(u_n)}$ and $\zeta = \frac{g(v_n)}{g(u_n)}$.

Optimal eighth order-method by Kung and Traub [15] $KT1$:

$$\begin{aligned} v_n &= u_n - \frac{g(u_n)}{g'(u_n)} \\ w_n &= v_n - \frac{g(v_n)g(u_n)}{(g(u_n) - g(v_n))^2} \frac{g(u_n)}{g'(u_n)}, \\ v_{n+1} &= w_n - \frac{g(u_n)}{g'(u_n)} \frac{g(u_n)g(v_n)g(w_n)}{(g(u_n) - g(v_n))^2} \frac{(g(u_n))^2 + g(v_n)(g(v_n) - g(w_n))}{(g(u_n) - g(w_n))^2(g(v_n) - g(w_n)}. \end{aligned} \tag{2.9}$$

3. Development and Convergence Analysis of our Proposed Method

In this section, the main purpose is to develop a new multi-point iterative method without memory by blending two existing without memory methods to produce a new scheme of convergence order $s_1 \times s_2$. The idea of combining different existing methods to yield better accuracy and to increase the convergence order by decreasing the number of function evaluations at the same time has been conducted in a number of research articles including [5, 6, 24, 25, 31] and the references cited therein. Being inspired by such recent studies, we proposed a new multi-point iterative method by blending a fourth-order method proposed by Kou et al. [14] with a classic Newton's method as shown below having eighth-order convergence:

$$\begin{aligned} v_n &= u_n - \frac{g(u_n)}{g'(u_n)}, \\ w_n &= v_n - \frac{(g(u_n))^2 + (g(v_n))^2}{g'(u_n)(g(u_n) - g(v_n))}, \\ u_{n+1} &= w_n - \frac{g(w_n)}{g'(w_n)}. \end{aligned} \tag{3.1}$$

The newly constructed scheme $SN1$ (3.1) requires five function evaluations per iteration, three evaluations of the function itself, and two of its first derivatives. The efficiency index is found to be approximately 1.5157 which is better than Newton's method and many other methods available in the literature.

To decrease the count of function evaluations in Eq. (3.1) and to align it with the Kung Traub conjecture. We use the concept of Gauss Quadrature approximation [12] to approximate the value of $g'(w_n)$.

We first consider the Newton's formula to derive the Gauss quadrature approximation as:

$$g'(w_n) = g'(u_n) + \int_{u_n}^{w_n} g''(s)ds. \tag{3.2}$$

By means of weight function the second derivative in (3.2) can be approximated:

$$\int_{u_n}^{w_n} g''(s)ds = c_1 g(u_n) + c_2 g(v_n) + c_3 g(w_n) + c_4 g'(u_n). \tag{3.3}$$

For finding the parameters c_1, c_2, c_3, and c_4. We employ four functions $g(s) = 1, g(s) = s, g(s) = s^2$, and $g(s) = s^3$ in order to obtain a family of four equations as:

$$\begin{aligned} c_1 + c_2 + c_3 &= 0, \\ c_1 u_n + c_2 v_n + c_3 w_n + c_4 &= 0, \\ c_1 u_n^2 + c_2 v_n^2 + c_3 w_n^2 + 2c_4 u_n &= 2(w_n - u_n), \\ c_1 u_n^3 + c_2 v_n^3 + c_3 w_n^3 + 3c_4 u_n^2 &= 3(w_n^2 - u_n^2). \end{aligned} \tag{3.4}$$

The solution of the system of Equations (3.4) is specified by four constants c_1, c_2, c_3 and c_4 and consequently by substituting these values into (3.2), we obatin the value of $g'(w_n)$ as:

$$\begin{aligned} g'(w_n) = &-\frac{(v_n - w_n)(3u_n - 2v_n - w_n)}{(u_n - w_n)(u_n - v_n)^2} g(u_n) + \frac{(u_n - w_n)^2}{(v_n - w_n)(u_n - v_n)^2} g(v_n) \\ &- \frac{(u_n + 2v_n - 3w_n)}{(u_n - w_n)(v_n - w_n)} g(w_n) + \frac{(2u_n + v_n - w_n)}{(u_n - v_n)} g'(u_n), \end{aligned} \tag{3.5}$$

On simplifying the expression (3.5) and substituting it in the last step of (3.1). We achieve the following:

$$\begin{aligned} v_n &= u_n - \frac{g(u_n)}{g'(u_n)}, \\ w_n &= v_n - \frac{(g(u_n))^2 + (g(v_n)^2)}{g'(u_n)(g(u_n) - g(v_n))}, \\ u_{n+1} &= w_n - g(w_n) \frac{(u_n - w_n)(u_n - v_n)^2 (v_n - w_n)}{E g(u_n) + F g(v_n) + G g(w_n) + H g'(u_n)}, \end{aligned} \tag{3.6}$$

where

$$\begin{aligned} E &= -(v_n - w_n)^2 (3u_n - 2v_n - w), \\ F &= (u_n - w_n)^3, \\ G &= -(u_n - v_n)^2 (u_n + 2v_n - 3w), \\ H &= (v_n - w_n)^2 (u_n - w_n)(u_n - v_n). \end{aligned}$$

The iterative method (3.6) is an optimal eight-order iterative method and requires only the four function evaluations, three computations of the function itself and one of its first derivative. The efficiency index of our proposed method ($SN2$) is $8^{\frac{1}{4}} = 1.6818$, Although it would typically be shown as $8^{\frac{1}{n^3+n^2}}$ for $n \geq 1$. To conduct the computation and comparison of all the iterative methods employed in the current research, refer to the following Table 1 and Figure 1.

TABLE 1. Comparison of efficiency indices of various iterative methods without memory.

Method	Order	Function Evaluations (FE)	Efficiency index (EI)	New FE per iteration
NN1	2	2	1.4142	$n+n^2$
SQ1	6	5	1.4309	$3n+2n^2$
KG1	8	5	1.5157	$3n+2n^2$
DP1	8	4	1.6817	$3n+n^2$
NJ1	8	4	1.6817	$3n+n^2$
SN1	8	5	1.5157	$3n+2n^2$
SN2	8	4	1.6817	$3n+n^2$

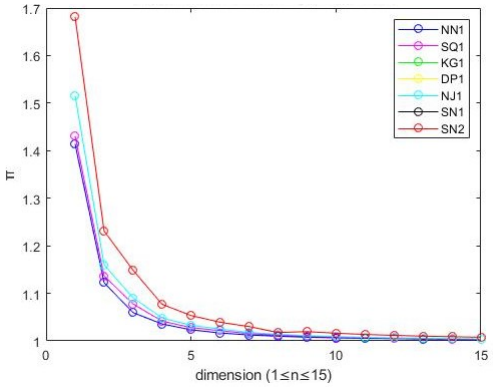

FIGURE 1. Efficiency index behavior of different iterative methods with increasing dimensions.

4. Convergence Analysis

The purpose of this section is to demonstrate the convergence analysis based on the hypothesis that at least an eighth-order technique exists. By using Taylor's series expansion, the required order of convergence has been attained. It is important to note that the convergence analysis is performed similarly to many other publications that have already been published and the development of higher-order procedures is primarily motivated by intellectual curiosity.

Theorem 4.1. *Let $\gamma \in D$ be a simple root of a sufficiently differentiable function $g : D \subseteq R \to R$ within an open interval D. Then, the three-step without memory method (3.1) possesses eighth-order of convergence with four functional evaluations per iteration, and the asymptotic error is given by:*

$$e_{n+1} = d_2^3(-3d_2^2 + d_3)^2 e_n^8 + O(e_n^9), \tag{4.1}$$

Proof. Let $e_n = u_n - \gamma$ be the error term in the $n-th$ iteration. Assuming that $g(\gamma) = 0$ and applying the Taylor's series expansion for function $g(u_n)$ around γ, we obtain:

$$g(u_n) = g'(\gamma)\left[e_n + d_2e_n^2 + d_3e_n^3 + d_4e_n^4 + d_5e_n^5 + d_6e_n^6 + d_7e_n^7 + d_8e_n^8 + O(e_n^9)\right], \tag{4.2}$$

where $d_k = \frac{g^k(\gamma)}{k!g'(\gamma)}$ for $k \in N$. Similarly, applying Taylor's series expansion for the function $g'(u_n)$ around γ, we obtain

$$g'(u_n) = g'(\gamma)\left[1 + 2d_2e_n + 3d_3e_n^2 + 4d_4e_n^3 + 5d_5e_n^4 + 6d_6e_n^5 + 7d_7e_n^6 + 8d_8e_n^7 + 9d_9e_n^7 + O(e_n^9)\right]. \tag{4.3}$$

On dividing (4.2) by (4.3), we obtain

$$\frac{g(u_n)}{g'(u_n)} = e_n - d_2e_n^2 + 2(d_2^2 - d_3)e_n^3 + (-4d_2^3 + 7d_2d_3 - 3d_4)e_n^4 + O(e_n^9). \tag{4.4}$$

Substituting (4.4) in the first step of (3.1), we have

$$v_n = \gamma + d_2e_n^2 + (-2d_2^2 + d_2)e_n^3 + (4d_2^3 - 7d_2d_3 + 3d_4)e_n^4 + O(e_n^9). \tag{4.5}$$

Applying Taylor's series expansion for the function $g(v_n)$ around β, we have

$$g(v_n) = g'(\gamma)\left[d_2e_n^2 + (-2d_2^2 + d_2)e_n^3 + (5d_2^3 - 7d_2d_3 + 3d_4)e_n^4 + O(e_n^9)\right]. \tag{4.6}$$

Also, from the second step of (3.1) with the aid of (4.2) and (4.6) , we have

$$g(u_n) - g(v_n) = e_n + (2d_2^2 - d_3) + (-5d_2^3 + 7d_2d_3 - 2d_4)e_n^4 + O(e_n^5), \tag{4.7}$$

and

$$g(u_n)^2 + g(v_n)^2 = e_n^2 + 2d_2e_n^3 + (2d_2^2 + 2d_3)e_n^4 + O(e_n^5), \tag{4.8}$$

With the help of (4.3), (4.7), (4.8), we obtain

$$\frac{(g(u_n))^2 + (g(v_n))^2}{g'(u_n)(g(u_n) - g(v_n))} = e_n + (-3d_2^3 + d_2d_3)e_n^4 + 2(9d_2^4 - 10d_2^2d_3 + d_2^2 + d_2d_4)e_n^5 + O(e_n^6), \tag{4.9}$$

Substitute (4.9) in the second step of (3.1), we have

$$w_n = \gamma + (3d_2^3 - d_2d_3)e_n^4 - 2(9d_2^4 + 10d_2^2d_3 + d_2^2 + d_2d_4)e_n^5 + O(e_n^6), \tag{4.10}$$

Again, we apply Taylor's series for the function $g(w_n)$ around γ, we obtain

$$g(w_n) = g'(\gamma)\left[(3d_2^3 - d_2d_3)e_n^4 - 2(9d_2^4 + 10d_2^2d_3 + d_2^2 + d_2d_4)e_n^5 + O(e_n^6)\right], \tag{4.11}$$

Taylor's series for the function $g'(w_n)$ around γ, we obtain

$$g(w_n) = g'(\gamma)\left[1 + 2d_2(3d_2^3 - d_2d_3)e_n^4 - 4(d_2(9d_2^4 + 10d_2^2d_3 + d_2^2 + d_2d_4))e_n^5 + O(e_n^6)\right], \tag{4.12}$$

Substituting (4.10) and (4.12) in the last step of (3.1), we have

$$e_{n+1} = d_2^3(-3d_2^2 + d_3)^2e_n^8 + O(e_n^9). \tag{4.13}$$

□

Theorem 4.2. *Let $\gamma \in D$ be a simple root of a sufficiently differentiable function $g : D \subseteq R \to R$ within an open interval D. Then, the three-step without memory method (3.6) possesses optimal eighth order of convergence with four functional evaluations per iteration, and the asymptotic error is given by:*

$$e_{n+1} = c_2^2(3d_2^2 - d_3)(3d_2^3 - d_2d_3 + d_4)e_n^8 + O(e_n^9). \tag{4.14}$$

Proof. Let $e_n = u_n - \gamma$ be the error term in the *nth* iteration. Assuming that $g(\gamma) = 0$ and applying the Taylor's series expansion for function $g(u_n)$ around γ, we obtain:

$$g(u_n) = g'(\gamma)\left[e_n + d_2e_n^2 + d_3e_n^3 + d_4e_n^4 + d_5e_n^5 + d_6e_n^6 + d_7e_n^7 + d_8e_n^8 + O(e_n^9)\right], \tag{4.15}$$

where $d_k = \frac{g^k(\gamma)}{k!g'(\gamma)}$ for $k \in N$. Similarly, applying Taylor's series expansion for the function $g'(u_n)$ around γ, we obtain

$$g'(u_n) = g'(\gamma)\left[1 + 2d_2e_n + 3d_3e_n^2 + 4d_4e_n^3 + 5d_5e_n^4 + 6d_6e_n^5 + 7d_7e_n^6 + 8d_8e_n^7 + 9d_9e_n^7 + O(e_n^9)\right]. \tag{4.16}$$

On dividing (4.15) by (4.16), we obtain

$$\frac{g(u_n)}{g'(u_n)} = e_n - d_2e_n^2 + 2(d_2^2 - d_3)e_n^3 + (-4d_2^3 + 7d_2d_3 - 3d_4)e_n^4 + O(e_n^5). \tag{4.17}$$

Substituting (4.17) in the first step of (3.6), we have

$$v_n = \gamma + d_2e_n^2 + (-2d_2^2 + d_2)e_n^3 + (4d_2^3 - 7d_2d_3 + 3d_4)e_n^4 + O(e_n^5). \tag{4.18}$$

Applying Taylor's series expansion for the function $g(v_n)$ around γ, we have

$$g(v_n) = g'(\gamma)\left[d_2e_n^2 + (-2d_2^2 + d_2)e_n^3 + (5d_2^3 - 7d_2d_3 + 3d_4)e_n^4 + O(e_n^5)\right]. \tag{4.19}$$

Also, from the second step of (3.6) with the aid of (4.15) and (4.19) , we have

$$g(u_n) - g(v_n) = e_n + (2d_2^2 - d_3)e_n^3 + (-5d_2^3 + 7d_2d_3 - 2d_4)e_n^4 + O(e_n^5), \tag{4.20}$$

and

$$(g(u_n))^2 + (g(v_n))^2 = e_n^2 + 2d_2e_n^3 + (2d_2^2 + 2d_3)e_n^4 + O(e_n^5), \tag{4.21}$$

with the help of (4.16), (4.20), and (4.21), we obtain

$$\frac{(g(u_n))^2 + (g(v_n))^2}{g'(u_n)(g(u_n) - g(v_n))} = e_n + (-3d_2^3 + d_2d_3)e_n^4 + 2(9d_2^4 - 10d_2^2d_3 + d_2^2 + d_2d_4)e_n^5 + O(e_n^6). \tag{4.22}$$

Substitute (4.22) in the second step of (3.6), we have

$$w_n = \gamma + (3d_2^3 - d_2d_3)e_n^4 - 2(9d_2^4 + 10d_2^2d_3 + d_2^2 + d_2d_4)e_n^5 + O(e_n^6). \tag{4.23}$$

Again, we apply Taylor's series for the function $g(w_n)$ around γ, we obtain

$$g(w_n) = g'(\gamma)\left[(3d_2^3 - d_2d_3)e_n^4 - 2(9d_2^4 + 10d_2^2d_3 + d_2^2 + d_2d_4)e_n^5 + O(e_n^6)\right], \tag{4.24}$$

with the aid of (4.15), (4.16), (4.18), (4.19), (4.23), and (4.24), we obtain the following

$$Eg(u_n) = (-(v-w)^2(3u - 2v - w))g(u_n) = -3d_2^2e_n^5 + O(e_n^6), \tag{4.25}$$

$$Fg(v_n) = (u-w)^3g(v_n) = e_n^3 + (-9d_2^4 + 3d_2d_3)e_n^6 + O(e_n^7), \tag{4.26}$$

$$Gg(w_n) = (u-v)^2(u + 2v - 3w)g(w_n) = e_n^3 + (d_2^2 + 4d_3 + 2(-2d_2^2 + d_2^3))e_n^5 + O(e_n^6), \tag{4.27}$$

$$Hg'(u_n) = (v-w)^2(u-w)(u-v)g'(u_n) = d_2^2e_n^6 + (-d_2^3 + 2d_2 + (-2d_2^2 + 2d_2))e_n^7 + O(e_n^8), \tag{4.28}$$

and

$$(u-w)(u-v)^2(v-w) = d_2e_n^5 + (-4d_2^2 + 2d_3)e_n^6 + O(e_n^7). \tag{4.29}$$

Finally, we employ (4.24)-(4.29) in the last step of the proposed method (3.6), we obtain

$$e_{n+1} = d_2^2(3d_2^2 - d_3)(3d_2^2 - d_2d_3 + d_4)e_n^8. \tag{4.30}$$

Therefore, our proposed method (3.6) is eighth-order optimal iterative method with the evaluation of three functions and one of the first derivative. Note that the efficiency index of our proposed method is $8^{\frac{1}{4}} \approx 1.6818$ which is better than Newton's method and others available in the literature. □

5. Basin of attraction

With the use of a concept known as the basins of attraction, it is possible to determine the stability of solutions (roots) for the nonlinear function $g(z) = 0$ using an iterative method. Basins of attractions serves as phase-planes that indicate iterations utilized by an iterative approach, which can assume multiple choices for the initial estimate. Small alterations to the initial estimate may result in convergence to distinct roots or sometimes no convergence at all. The areas where the iterative method's behaviour varies are indicated by these basin boundaries. Each point is assigned a color based on the root to which the corresponding method converges when starting from that point; if the method diverges, the point is colored white. The dynamical analysis provided below shows that there are several key areas where the novel method performs better than other existing methods that are being used. This property of a rational function associated with an iterative scheme operating on a polynomial provides important information about the numerical characteristics of the method, such as its stability and reliability. For complex functions, we used Mathematica 11 to find the basin of attraction. In this connection, we consider a rectangle $R = [-2, 2] \times [-2, 2] \subset C$ and each point $z_0 \in R$ has a different color allocated to it.

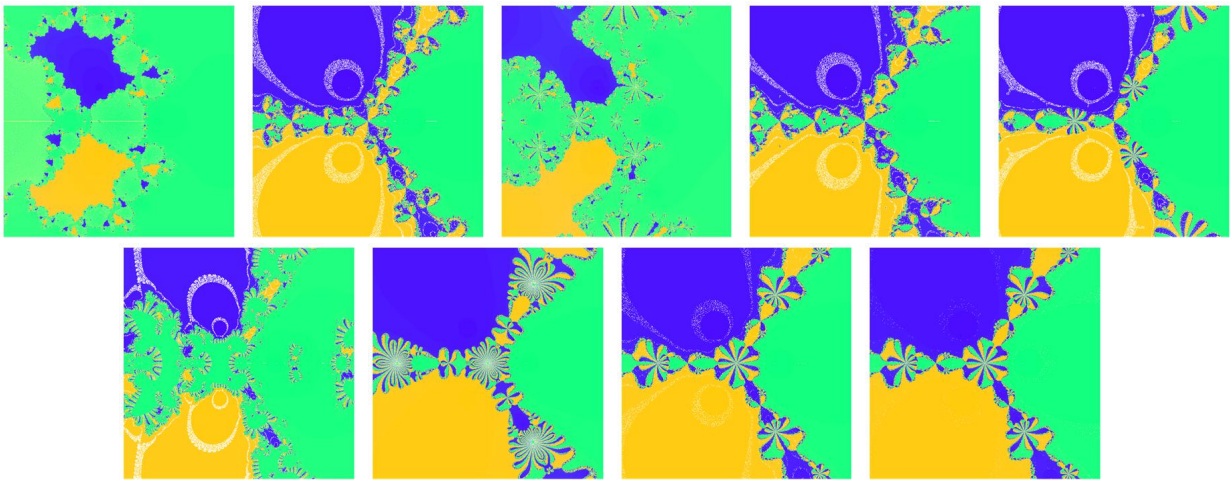

FIGURE 2. Basins of attraction for $JH1$, $KT1$, $LX1$, $CC1$, $DP1$, $NJ1$, $SP1$, $SN1$ and $SN2$ respectively for $p_1(z)$.

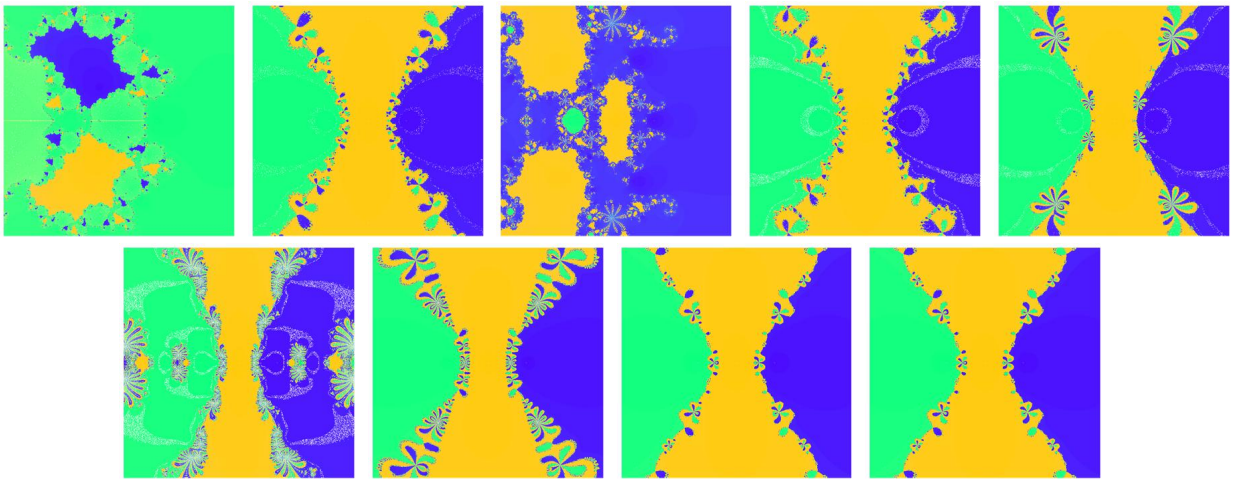

FIGURE 3. Basins of attraction for $JH1$, $KT1$, $LX1$, $CC1$, $DP1$, $NJ1$, $SP1$, $SN1$ and $SN2$ respectively for $p_2(z)$.

TABLE 2. Some non-linear test functions.

function	root
$g_1(u) = \log(u^2 + u + 2) + u - 1$	0.1960..
$g_2(u) = 0.5 - \sin(u)$	0.5235..
$g_3(u) = u^2 \sin(u) - \cos(u)$	0.8952..
$g_4(u) = 4\sin(u) - u + 1$	-0.3421..

Many researchers have utilized basins of attraction to compare their iterative techniques. For instance, Obadah Solaiman [29] investigated various iterative strategies for solving nonlinear equations of different orders. It was concluded that the effectiveness of these methods is influenced by factors beyond just the convergence order.

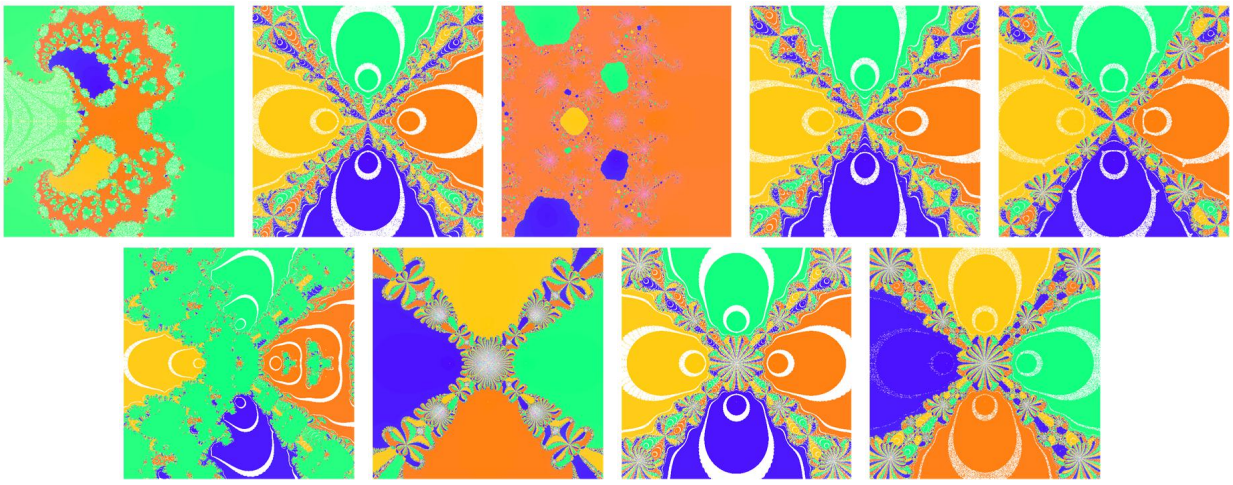

FIGURE 4. Basins of attraction for $JH1$, $KT1$, $LX1$, $CC1$, $DP1$, $NJ1$, $SP1$, $SN1$ and $SN2$ respectively for $p_3(z)$.

For illustrating graphic features in the complex plane, we employed the three complex polynomials listed below together with their roots:

Problem 1 : Let us consider

$$p_1(z) = z^3 - 1, \tag{5.1}$$

having roots as cube roots of unity. The basins are depicted in Figure 2. It can be seen from the figure that our proposed methods $SN1$ and $SN2$ contains fewer number of white points and having much larger basins as compared to other existing methods of same order.

Problem 2 : Let us consider:

$$p_2(z) = z^3 - z, \tag{5.2}$$

having roots $0, 1$ and -1. The basins are depicted in Figure 3. It can be seen from the figure that our proposed methods $SN1$ and $SN2$ contains fewer number of white points and having much larger basins as compared to other existing methods of same order.

Problem 3 : Let us consider:

$$p_3(z) = z^4 - 1, \tag{5.3}$$

having roots $1, -1, \iota$ and $-\iota$. The basins are depicted in Figure 4. It can be seen from the figure that our proposed methods $SN1$ and $SN2$ contains fewer number of white points and having much larger basins as compared to other existing methods of same order.

6. Numerical Implementations

We carried out several numerical simulations with a few existing optimal eighth-order iterative algorithms to show the efficiency of our suggested approach. In Table 2, we consider some nonlinear functions taken from different research articles. Furthermore, we take some application based problems to strengthen the results. Each Table contains the initial guess, the absolute error of functions, the computational order of convergence, and the CPU time. All the numerical results have been tabulated. Furthermore, Algorithms 1 and 2 present the code for both the numerical comparisons and the basins of attraction, respectively, which were utilized in this paper.

Tables 3 and 7 show the numerical comparisons of various three points optimal iterative methods without memory. It is clearly visible that our results are superior and efficient than existing methods. Besides computational order of convergence, CPU time is also one of the best ways to compare the effectiveness of the iterative methods. At this

TABLE 3. Numerical Comparisons of optimal three-point without memory methods for g_1 and g_2.

Function	Method	Guess	$\|u_1-\alpha\|$	$\|u_2-\alpha\|$	$\|u_3-\alpha\|$	COC	CPU
g_1	JH1	2.2	4.1630e-4	5.9730e-31	1.0757e-245	8.0000	0.922
		3.7	1.2345e-2	3.2996e-19	9.3298e-152	8.0000	2.047
	KT1	2.2	1.6648e-3	3.3556e-34	9.1513e-272	8.0000	0.985
		3.7	6.0416e-3	9.7216e-22	4.5414e-172	8.0000	1.313
	LX1	2.2	1.6188e-4	5.3172e-34	7.2116e-270	8.0000	0.921
		3.7	5.4517e-3	8.4943e-22	3.0593e-172	8.0000	1.234
	CC1	2.2	2.9501e-4	6.9325e-34	6.4529e-271	8.0000	0.92
		3.7	8.7916e-3	4.1952e-22	1.1606e-176	8.0000	2.25
	DP1	2.2	1.1200e-5	6.9325e-43	2.1549e-345	8.0000	1.172
		3.7	3.0199e-3	5.9319e-24	1.3341e-189	8.0000	1.485
	NJ1	2.2	3.1267e-1	6.5803e-8	6.3577e-67	8.0000	1.157
		3.7	NC				
	SP1	2.2	1.6501e-3	2.4329e-34	5.4383e-273	8.0000	0.97
		3.7	5.7361e-3	5.0136e-82	1.7689e-174	8.0000	1.298
	SN1	2.2	3.4047e-4	2.6233e-40	3.2597e-321	8.0000	0.969
		3.7	1.3119e-3	1.2631e-27	9.4161e-220	8.0000	1.608
	SN2	2.2	1.7040e-3	1.7805e-34	2.5325e-272	8.0000	0.891
		3.7	6.5262e-3	7.9579e-22	4.0322e-173	8.0000	0.859
g_2	JH1	0.9	2.4175e-4	1.6403e-31	7.3538e-249	8.0000	1.047
		-0.2	9.2502e-6	7.5230e-43	1.4398e-43	8.0000	1.390
	KT1	0.9	5.4730e-5	7.5281e-37	9.6506e-292	8.0000	1.188
		-0.2	5.0354e-7	3.8663e-53	4.6711e-422	8.0000	1.547
	LX1	0.9	1.0095e-4	1.9988e-34	4.7250e-272	8.0000	1.234
		-0.2	1.0263e-6	2.2829e-50	1.3680e-399	8.0000	1.593
	CC1	0.9	4.0020e-4	3.6107e-38	1.5853e-302	8.0000	1.017
		-0.2	2.1096e-6	2.1526e-48	2.5293e-384	8.0000	1.408
	DP1	0.9	2.1378e-5	3.9349e-40	5.1844e-318	8.0000	1.391
		-0.2	8.0959e-6	1.6647e-43	5.3213e-345	8.0000	1.704
	NJ1	0.9	2.0194e-5	2.3278e-40	7.2569e-320	8.0000	1.001
		-0.2	1.5577e-5	2.9181e-41	4.4257e-327	8.0000	1.313
	SP1	0.9	52306e-5	3.8477e-37	3.3006e-294	8.0000	1.233
		-0.2	4.9117e-3	2.3270e-53	5.9065e-424	8.0000	1.298
	SN1	0.9	3.4047e-4	2.6233e-40	3.2597e-321	8.0000	0.969
		-0.2	1.3119e-3	1.2631e-27	9.4161e-220	8.0000	1.608
	SN2	0.9	3.2708e-5	4.3753e-39	4.4874e-310	8.0000	1.015
		-0.2	2.1288e-7	1.4094e-56	5.2015e-450	8.0000	1.312

juncture, with the help of *Mathematica*11 the command *Timeused*[] is used to calculate the CPU time. The absolute errors $|u_k-\alpha|$ for the first three iterations along with the computational order of convergence for the presented scheme and the existing schemes. The numerical results were performed with the *Mathematica*11 system running under Windows 10 pro with an installed memory of 10GB having a processor Intel(R) Core(TM) i5 CPU @ 1.80 GHz speed and system type 64-bit operating system. The COC has been calculated by using the usual formula [21]:

$$COC \sim \frac{ln|g(u_{n+1})/g(u_n)|}{ln|g(u_n)/g(u_{n-1})|}, \tag{6.1}$$

to verify the hypothesized convergence rate and assess computational effectiveness.

TABLE 4. Numerical Comparisons of optimal three-point without memory methods for g_3 and g_4.

Function	Method	Guess	$\lvert u_1-\alpha\rvert$	$\lvert u_2-\alpha\rvert$	$\lvert u_3-\alpha\rvert$	COC	CPU
g_3	JH1	1.6	1.8209e-2	2.6774e-14	7.2897e-109	8.0000	1.328
		1.9	1.2345e-2	3.2996e-19	9.3298e-152	8.0000	2.047
	KT1	1.6	22864e-4	8.1366e-30	2.0970e-233	8.0000	1.296
		1.9	1.1846e-3	4.1924e-24	1.0417e-187	8.0000	1.953
	LX1	1.6	4.6523e-4	3.3938e-27	2.7307e-212	8.0000	1.391
		1.9	2.1810e-3	7.8161e-22	2.1615e-169	8.0000	2.078
	CC1	1.6	1.358e-4	1.5105e-22	1.1282e-176	8.0000	1.358
		1.9	2.6908e-3	1.1699e-24	1.4604e-195	8.0000	2.092
	DP1	1.6	5.9232e-4	6.1568e-27	8.4122e-211	8.0000	1.427
		1.9	4.0453e-3	2.9755e-20	2.5034e-157	8.0000	1.328
	NJ1	1.6	NC				
		1.9	NC				
	SP1	1.6	2.1113e-4	3.0038e-30	5.0509e-237	8.0000	1.220
		1.9	1.1563e-3	2.4109e-24	8.6980e-190	8.0000	1.829
	SN1	1.6	3.9605e-4	5.6924e-31	1.0391e-245	8.0000	1.298
		1.9	2.7833e-3	3.3419e-24	1.4661e-191	8.0000	1.97
	SN2	1.6	5.9263e-5	1.1516e-34	2.3425e-272	8.0000	1.188
		1.9	4.8212e-4	2.2002e-27	4.1589e-173	8.0000	1.767
g_4	JH1	-2.5	8.2065e-7	3.7642e-50	7.3759e-397	8.0000	1.016
		-3.6	4.4177e-5	2.6555e-34	4.5245e-286	8.0000	1.422
	KT1	-2.5	7.7616e-7	1.2930e-50	7.6680e-401	8.0000	1.077
		-3.6	7.3784e-8	8.6231e-59	3.0012e-486	8.0000	1.593
	LX1	-2.5	1.2978e-6	1.2130e-48	7.0650e-385	8.0000	0.906
		-3.6	1.2814e-7	1.0957e-56	3.1311e-449	8.0000	1.313
	CC1	-2.5	8.1415e-7	5.0989e-55	1.2068e-440	8.0000	0.999
		-3.6	7.0552e-6	8.2806e-44	2.9825e-347	8.0000	1.467
	DP1	-2.5	6.5141e-7	1.6020e-51	2.1431e-408	8.0000	0.969
		-3.6	2.1251e-5	2.0549e-39	1.5709e-311	8.0000	1.313
	NJ1	-2.5	1.5507e-1	1.1806e-6	5.3658e-48	8.0000	1.001
		-3.6	NC				
	SP1	-2.5	5.5379e-7	6.1002e-52	1.3223e-411	8.0000	1.001
		-3.6	7.3776e-8	6.0521e-59	1.2412e-467	8.0000	1.345
	SN1	-2.5	1.5231e-6	3.7808e-52	5.4494e-417	8.0000	1.063
		-3.6	4.6336e-8	1.4168e-60	1.0825e-480	8.0000	1.469
	SN2	-2.5	3.7495e-7	2.3304e-53	5.1881e-423	8.0000	0.922
		-3.6	3.7007e-8	2.0985e-61	2.2436e-487	8.0000	1.251

Example 6.1. German physicist Max Planck developed the mathematical formula known as Planck's radiation law in 1900 to describe the spectral-energy distribution of radiation emitted by a black body given by [3]:

$$f(\lambda) = \frac{8\pi C h_p \lambda^{-5}}{e^{\frac{Ch_p}{\lambda B_k T}} - 1}, \tag{6.2}$$

where λ = Wavelength of the radiation.
h_p = Planck's constant.
B_k= Boltzmann Constant.
T = Absolute temperature of the Blackbody raditaion.

Algorithm 1

Solve1 = FindRoot[$f(x) == 0$, $x, -1$, WorkingPrecison $\to$ 1000];
digits= The number of digits for high-precision arithmetic;
no=Choose the maximum no of iterations;
Co[0]= initial guess;
For [$i = 1$ to no
$f_x = SetAccuracy[f(x), digits]$;
$f_x^{'} = SetAccuracy[f'(x), digits]$;
$y = SetAccuracy[x - \frac{f_x}{f_x^{'}}, digits]$;
$f_y = SetAccuracy[f(y), digits]$;
$f_y^{'} = SetAccuracy[f'(y), digits]$;
$z = SetAccuracy[x - \frac{f_x^2+f_y^2}{f_x^{'}(f_x-f_y)}, digits]$;
$f_z = SetAccuracy[f(z), digits]$;
$f_z^{'} = SetAccuracy[f'(z), digits]$;
$u = SetAccuracy[z - f_z((x-z)*(x-y)^2*(y-z))/(-(y-z)^2*(3*x-2*y-z)*f_x + (x-z)^3*f_y - (x-y)^2*(x+2*y-3z)*fz + (y-z)^2*(x-z)*(x-y)*f_x^{'}), digits]$;
$x = u$;
$Co[i] = x$; $Print[N[Abs[x-b], 5]];$]
$COC = N[Log[Abs[f[Co[(no)]]/f[Co[(no-1)]]]]/Log[Abs[f[Co[(no-1)]]/f[Co[(no-2)]]]], 5]; TimeUsed[];$

Algorithm 2

$p[z] = (z^3 - 1)$;
$h[z] := p[z]/p'[z]$;
$theRoots = z/.NSolve[h[z] == 0, z]$;
$y = Simplify[z - p[z]/p'[z]]$;
$w = Simplify[z - ((p[z]^2 + p[y]^2)/(p'[z] * (p[z] - p[y])))]$;
$Simplify[w - (p[w]/p'[w])]cp = Compile[z, Complex, Evaluate[h[z]]]$;
$n = Compile[z, Complex, Evaluate[Simplify[w - (p[w] * (((z-w)*(z-y)^2*(y-w))/((-(y-w)^2*(3*z-2*y-w)*p[z]) + ((z-w)^3*p[y]) - ((z-y)^2*(z+2*y-3*w)*p[w]) + ((y-w)^2*(z-w)*(z-y)*p'[z])))))]$
$bail = 300$; $orbitData = Table[NestWhileList[n, x + Iy, Abs[cp[]] > 0.001, 1, bail], y, -3, 3, 0.01$
$, x, -3, 3, 0.01]$
$numRoots = Length[Union[theRoots]]$;
$sameRootFunc = Compile[z,_C omplex, Evaluate[Abs[3h[z]/h'[z]]]]$;
$whichRoot[orbit_] := Module[i, z, z = Last[orbit]; i = 1; Scan[If[Abs[z] < sameRootFunc[z],$
$If[i <= numRoots, i, Length[orbit], None]]$;
$rootData = Map[whichRoot, orbitData, 2]$;
$colorList = CMYKColor[cc, 0, 0.5], CMYKColor[0.7, cc, 0], CMYKColor[0, 0.2, cc], CMYKColor[0, 0.5, cc]$;
$cols = rootData//\ k_Inteqer, lInteger(colorList[[k]]\ //.\ cc- > (1 - l/(bail + 1))^8)$,
$None- > CMYKColor[0, 0, 0]$;
$Rasterize[Show[Graphics[RasterArray[cols]], AspectRatio- > Automatic], ImageResolution- > 72]$

C = Speed of light.
We are interested in determining the wavelength that fits the highest energy density. For that purpose using (6.2), we obtain

$$f'(\lambda) = \left(\frac{8\pi C h_p \lambda^{-6}}{e^{Ch_p/\lambda B_k T} - 1}\right)\left(\frac{(Ch_p/\lambda B_k T)e^{Ch_p/\lambda B_k T}}{e^{Ch_p/\lambda B_k T} - 1} - 5\right) = R\dot{S}. \tag{6.3}$$

TABLE 5. Numerical Comparisons of optimal three-point without memory methods for g_5.

Function	Method	Guess	$\|u_1-\alpha\|$	$\|u_2-\alpha\|$	$\|u_3-\alpha\|$	COC	CPU
g_5	JH1	2.2	2.3547e-1	9.0106e-8	6.9729e-61	8.0000	1.080
		2.5	1.1211e-2	4.6576e-20	3.5540e-159	8.0000	1.486
	KT1	2.2	5.1383e-2	1.1010e-14	9.8100e-116	8.0000	1.296
		2.5	4.8976e-3	1.4055e-22	6.9193e-179	8.0000	1.953
	LX1	2.2	2.6755e-1	5.4938e-10	5.1603e-78	8.0000	1.453
		2.5	1.3274e-2	4.9454e-19	2.2249e-150	8.0000	1.860
	CC1	2.2	2.9299e-1	2.5831e-6	9.8236e-49	8.0000	1.391
		2.5	3.7784e-3	2.1645e-23	2.3878e-185	8.0000	1.813
	DP1	2.2	3.8378e-2	2.4955e-15	1.3083e-120	8.0000	1.032
		2.5	3.4315e-3	1.5983e-23	3.7039e-186	8.0000	1.438
	NJ1	2.2	NC				
		2.5	NC				
	SP1	2.2	9.9011e-2	1.0596e-12	6.5437e-100	8.0000	1.391
		2.5	6.2505e-3	8.8131e-22	1.4985e-190	8.0000	1.875
	SN1	2.2	4.4671e-3	4.3121e-24	3.4684e-192	8.0000	1.390
		2.5	6.0113e-4	4.9042e-31	9.7081e-248	8.0000	1.749
	SN2	2.2	4.6159e-3	7.2491e-23	2.8535e-181	8.0000	1.079
		2.5	2.1466e-3	1.6391e-25	1.9498e-202	8.0000	1.455

It is clear that when $S=0$, there is a maximum for f, that is, when

$$\left(\frac{(Ch_p/\lambda B_k T)e^{Ch_p/\lambda B_k T}}{e^{Ch_p/\lambda B_k T}-1}-5\right)=0. \tag{6.4}$$

Let us take $u=\frac{Ch_p}{\lambda B_k T}$, then equation (51) becomes:

$$g_5(u)=e^{-u}+\frac{u}{5}-1. \tag{6.5}$$

As mentioned in [24] the approximate root of the Eq. (6.5) is $x=4.96511423174427$. Therefore, the wavelength of radiation where the energy density is maximum can be approximately determined using the formula below:

$$\lambda\approx\frac{Ch_p}{4.965114B_kT}. \tag{6.6}$$

Example 6.2. The following equation describes the path of an electron traverses in the region between two parallel plates by considering the multi factor effect:

$$r(t)=r_0+\left(v_0+q_0\frac{E}{m\phi}\sin\phi t_0(t-t_0)+\beta\right)(t-t_0)+q_0\frac{E_0}{m\phi^2}(\cos(\phi t+\beta)+\sin(\phi t+\beta)), \tag{6.7}$$

where r_0 and v_0 are the position and velocity of the electron at time t_0, q_0 and m are the charge and mass of an electron at rest and $E_0\sin\phi(t)+\beta$ is the RF electric field between the plates. If we choose the specific parameters, then Eq. (6.7) can be expressed as:

$$g_6(u)=\frac{\pi}{4}-\frac{cosu}{2}+u=0. \tag{6.8}$$

The desired root of Eq. (6.8) is $\zeta=-0.3090932715417949$.

Example 6.3. Civil engineering beams [4] used in a mathematical model of a beam are horizontal building components used to sustain loads and cover voids such as the topmost section is either made of stone or brick (the beam is referred to as lintel in that case). The term "floor joist" or "roof joist" is used to describe a beam depending on whether it is supporting a floor or a roof. While the stringers support the lesser weights over the bridge deck, the floor beams

TABLE 6. Numerical Comparisons of optimal three-point without memory methods for g_6.

Function	Method	Guess	$\|u_1-\alpha\|$	$\|u_2-\alpha\|$	$\|u_3-\alpha\|$	COC	CPU
g_6	JH1	0.7	1.9951e-4	9.3186e-33	2.1092e-259	8.0000	0.97
		1.2	1.4724e-3	8.2299e-26	7.8063e-204	8.0000	1.001
	KT1	0.7	1.0293e-4	5.7698e-34	5.6251e-286	8.0000	1.046
		1.2	7.78606e-4	6.1835e-29	9.7885e-230	8.0000	1.405
	LX1	0.7	1.8590e-4	7.3515e-34	4.3961e-269	8.0000	1.156
		1.2	1.5589e-3	1.7980e-26	5.6285e-210	8.0000	1.220
	CC1	0.7	4.5297e-6	1.5721e-46	3.3102e-370	8.0000	1.141
		1.2	3.6135e-4	2.5759e-31	1.7191e-248	8.0000	1.610
	DP1	0.7	7.9877e-5	2.6434e-38	3.8082e-306	8.0000	1.141
		1.2	1.0402e-3	2.1468e-29	7.2085e-235	8.0000	1.391
	NJ1	0.7	3.0788e-4	4.4155e-32	7.8990e-255	8.0000	1.094
		1.2	5.0015e-2	1.9364e-14	1.0809e-113	8.0000	1.532
	SP1	0.7	6.2505e-3	2.6660e-37	9.0403e-297	8.0000	1.078
		1.2	6.2152e-4	7.8826e-30	5.2800e-237	8.0000	1.469
	SN1	0.7	9.4815e-5	6.1863e-36	2.0323e-285	8.0000	1.064
		1.2	6.1701e-4	1.9878e-29	2.3092e-233	8.0000	1.156
	SN2	0.7	2.7963e-5	2.1140e-40	2.2557e-321	8.0000	0.954
		1.2	1.1700e-4	1.9849e-35	1.3626e-281	8.0000	1.267

TABLE 7. Numerical Comparisons of optimal three-point without memory methods for g_7.

Function	Method	Guess	$\|u_1-\alpha\|$	$\|u_2-\alpha\|$	$\|u_3-\alpha\|$	COC	CPU
g_7	JH1	-2.9	2.6559e-2	3.6314e-13	3.4115e-100	8.0000	0.75
		-3.5	1.2787e-1	3.0892e-7	9.3560e-53	8.0000	0.859
	KT1	-2.9	5.1207e-2	1.3452e-11	4.1465e-88	8.0000	0.703
		-3.5	6.6020e-2	9.4130e-11	2.3831e-81	8.0000	0.828
	LX1	-2.9	6.7377e-2	1.2647e-10	2.8804e-80	8.0000	0.781
		-3.5	8.9289e-2	1.0627e-9	7.1621e-73	8.0000	0.906
	CC1	-2.9	4.1678e-2	1.1673e-12	7.0429e-97	8.0000	0.797
		-3.5	6.4198e-2	2.8589e-11	9.1169e-86	8.0000	0.922
	DP1	-2.9	5.8271e-2	1.0298e-11	1.2198e-89	8.0000	0.812
		-3.5	9.3462e-2	3.9431e-10	5.6342e-77	8.0000	0.953
	NJ2	-2.9	NC				
		-3.5	NC				
	SP1	-2.9	4.6526e-2	4.1823e-12	2.4164e-92	8.0000	0.812
		-3.5	6.3953e-2	4.7738e-11	6.9626e-84	8.0000	0.922
	SN1	-2.9	3.9432e-2	1.7254e-12	3.0454e-95	8.0000	0.719
		-3.5	5.1127e-2	1.2734e-11	2.6804e-88	8.0000	0.844
	SN2	-2.9	3.1471e-2	2.9056e-13	1.9127e-101	8.0000	0.641
		-3.5	3.8318e-2	1.3387e-12	3.8831e-96	8.0000	0.751

are the heavier transverse components. Often referred to as girders, large beams are used to support the terminal ends of smaller, perpendicular beams. Single rolled pieces of metal can be utilized, or I-shaped girders can be built by joining plates and angles using rivets or welding to increase rigidity and lengthen spans. Moreover, concrete girders are widely used. In this context, many nonlinear equation-based mathematical models have been developed to describe

the precise beam location. The model below is an example which was taken from [18]:

$$g_7(u) = u^4 + 4u^3 - 24u^2 + 16u + 16 = 0. \tag{6.9}$$

The roots of the above fourth order polynomial are $2, 2$ and $-4 \pm 2\sqrt{3}$.

7. Conclusion

In this study, we provide additional contributions to the theory of iteration processes and suggest a new family of eighth-order technique that is optimal for solving nonlinear equations quickly. The proposed method acquires eighth-order convergence with only four functional evaluations. The order of convergence and asymptotic error of the suggested three-step hybrid approach have been theoretically stated using the Taylor series. In terms of absolute errors and CPU time in seconds computed during the final iteration, the proposed three-step composite technique surpasses other known numerical algorithms. Polynomiography is used to demonstrate the proposed method's fast convergence when it is implemented on a number of complex-valued polynomials. Further research can be conducted to explore the conversion of the proposed method into a derivative free method by making suitable approximations. Also, it is further possible to add different self-accelerated parameters to convert it into a derivative free with memory method.

Acknowledgment

The authors express their gratitude to the referees for their comments and suggestions.

References

[1] S. Abbasbandy, *Improving newton–raphson method for nonlinear equations by modified adomian decomposition method*, Applied mathematics and computation, *145*(2-3) (2003), 887–893.
[2] S. Abdullah, N. Choubey, and S. Dara, *Two novel with and without memory multipoint iterative methods for solving non-linear equations*, Communications in Mathematics and Applications, *3* (2024), 9-31.
[3] S. Abdullah, N. Choubey, and S. Dara, *An efficient two-point iterative method with memory for solving non-linear equations and its dynamics*, Journal of Applied Mathematics and Computing, *70*(1) (2024), 285–315.
[4] S. Abdullah, N. Choubey, and S. Dara, *Optimal fourth-and eighth-order iterative methods for solving nonlinear equations with basins of attraction*, Journal of Applied Mathematics and Computing, *70* (2024), 3477–3507.
[5] S. Abdullah, N. Choubey, and S. Dara, *Dynamical analysis of optimal iterative methods for solving nonlinear equations with applications*, Journal of Applied Analysis and Computation, *14*(6) (2024), 3349–3376.
[6] H. A. Abro and M. M. Shaikh, *A new time-efficient and convergent nonlinear solver* Applied Mathematics and Computation, *355* (2019), 516–536.
[7] N. Choubey and J. P. Jaiswal, *An improved optimal eighth-order iterative scheme with its dynamical behaviour*, International Journal of Computing Science and Mathematics, *7*(4) (2016), 361–370.
[8] A. Cordero, J. L. Hueso, E. Martínez, and J. R. Torregrosa, *A modified newtonjarratt's composition*, Numerical Algorithms, *55* (2010) , 87–99.
[9] A. Cordero, T. Lotfi, K. Mahdiani, and J. R. Torregrosa, *Two optimal general classes of iterative methods with eighth-order*, Acta applicandae mathematicae, *134*(1) (2014), 61–74.
[10] J. Džunić and M. S. Petković, *A family of three-point methods of ostrowski's type for solving nonlinear equations*, Journal of Applied Mathematics, *2012*(2) (2012).
[11] J. H. He and X. H. Wu, *Variational iteration method: new development and applications*, Computers and Mathematics with Applications, *54*(7-8) (2007), 881–894.
[12] S. K. Khattri, *Quadrature based optimal iterative methods with applications in high-precision computin*, Numerical Mathematics: Theory, Methods and Applications, *5*(4) (2012), 592–601.
[13] B. Kong-ied, *Two new eighth and twelfth order iterative methods for solving nonlinear equations*, International Journal of Mathematics and Computer Science, *16* (2021), 333–344.
[14] J. Kou, Y. Li, and X. Wang, *A composite fourth-order iterative method for solving non-linear equations*, Applied Mathematics and Computation, *184*(2) (2007), 471–475.

[15] H. Kung and J. F. Traub, *Optimal order of one-point and multipoint iteration*, Journal of the ACM (JACM), *21*(4) (1974), 643–651.
[16] L. Liu and X. Wang, *Eighth-order methods with high efficiency index for solving nonlinear equations*, Applied Mathematics and Computation, *215*(9) (2010), 3449–3454.
[17] N. A. Mir and T. Zaman, *Some quadrature based three-step iterative methods for non-linear equations*, Applied Mathematics and Computation *193*(2) (2007), 366–373.
[18] A. Naseem, M. Rehman, S. Qureshi, and N. A. D. Ide, *Graphical and numerical study of a newly developed root-finding algorithm and its engineering applications*, IEEE Access, *11* (2023), 2375–2383.
[19] M. A. Noor and M. Waseem, *Some iterative methods for solving a system of nonlinear equations. Computers and Mathematics with Applications*, *57*(1) (2009), 101–106.
[20] J. M. Ortega, *Numerical Analysis*, A Second Course. SIAM, (1990).
[21] A. Y. Ozban, *Some new variants of newton's method*, Applied Mathematics Letters, *17*(6) (2004), 677–682.
[22] S. Parhi and D. K. Gupta, *A sixth order method for nonlinear equations*, Applied Mathematics and Computation, *203*(1) (2008), 50–55.
[23] S. Parimala, K. Madhu, and J. Jayaraman, *A new class of optimal eighth order method with two weight functions for solving nonlinear equation*, J. Nonlinear Anal. Appl, (2018), 83–94.
[24] S. Qureshi, H. Ramos, and A. K. Soomro, *A new nonlinear ninth-order root-finding method with error analysis and basins of attraction*, Mathematics, *9*(16) (2021).
[25] S. Qureshi, A. Soomro, A. A. Shaikh, E. Hinca,l and N. Gokbulut, *A novel multistep iterative technique for models in medical sciences with complex dynamics*, Computational and Mathematical Methods in Medicine, (2022).
[26] M. Rafiullah, *A fifth-order iterative method for solving nonlinear equations*, Numerical Analysis and Applications, *4*(3) (2011), 239.
[27] J. R. Sharma and H. Arora, *An efficient family of weighted-newton methods with optimal eighth order convergence*, Applied Mathematics Letters, *29* (2014), 1–6.
[28] A. Singh and J. P. Jaiswal, *An efficient family of optimal eighth-order iterative methods for solving nonlinear equations and its dynamics*, Journal of Mathematics, (2014).
[29] O. Solaiman and A. IshakHashim, *The attraction basins of several root finding methods, with a note about optimal methods*, Proceedings The 6th International Arab Conference on Mathematics and Computations, (2019), 68.
[30] H. Tari, D. Ganji, and H. Babazadeh, *The application of he's variational iteration method to nonlinear equations arising in heat transfer*, Physics Letters A, *363*(3) (2007), 213–217.
[31] A. Tassaddiq, S. Qureshi, A. Soomro, E. Hincal, D. Baleanu, and A. A Shaikh, *A new three-step root-finding numerical method and its fractal global behavior*, Fractal and Fractional, *5*(4) (2021), 204.
[32] F. Zafar and N. A. Mir, *A generalized family of quadrature based iterative methods*, General Math, *18*(4) (2010), 43–51.