fractal and fractional

MDPI

Article

A New Three-Step Root-Finding Numerical Method and Its Fractal Global Behavior

Asifa Tassaddiq [1,†], Sania Qureshi [2,3,*,†], Amanullah Soomro [2,†], Evren Hincal [3,†], Dumitru Baleanu [4,5,†] and Asif Ali Shaikh [2,†]

1 Department of Basic Sciences and Humanities, College of Computer and Information Sciences, Majmaah University, Al-Majmaah 11952, Saudi Arabia; a.tassaddiq@mu.edu.sa
2 Department of Basic Sciences and Related Studies, Mehran University of Engineering and Technology, Jamshoro 76062, Pakistan; soomroamanullah820@gmail.com (A.S.); asif.shaikh@faculty.muet.edu.pk (A.A.S.)
3 Department of Mathematics, Near East University TRCN, Mersin 10, 99138 Nicosia, Turkey; evren.hincal@neu.edu.tr
4 Department of Mathematics, Cankaya University, 06530 Ankara, Turkey; dumitru.baleanu@gmail.com
5 Institute of Space Sciences, Magurele, R76900 Bucharest, Romania
* Correspondence: sania.qureshi@faculty.muet.edu.pk
† These authors contributed equally to this work.

Abstract: There is an increasing demand for numerical methods to obtain accurate approximate solutions for nonlinear models based upon polynomials and transcendental equations under both single and multivariate variables. Keeping in mind the high demand within the scientific literature, we attempt to devise a new nonlinear three-step method with tenth-order convergence while using six functional evaluations (three functions and three first-order derivatives) per iteration. The method has an efficiency index of about 1.4678, which is higher than most optimal methods. Convergence analysis for single and systems of nonlinear equations is also carried out. The same is verified with the approximated computational order of convergence in the absence of an exact solution. To observe the global fractal behavior of the proposed method, different types of complex functions are considered under basins of attraction. When compared with various well-known methods, it is observed that the proposed method achieves prespecified tolerance in the minimum number of iterations while assuming different initial guesses. Nonlinear models include those employed in science and engineering, including chemical, electrical, biochemical, geometrical, and meteorological models.

Keywords: nonlinear models; efficiency index; computational cost; Halley's method; basin of attraction; computational order of convergence

MSC: 65H04; 68W05

Citation: Tassaddiq, A.; Qureshi, S.; Soomro, A.; Hincal, E.; Baleanu, D; Shaikh, A.A. A New Three-Step Root-Finding Numerical Method and Its Fractal Global Behavior. *Fractal Fract.* **2021**, *5*, 204. https://doi.org/10.3390/fractalfract5040204

Academic Editor: Carlo Cattani

Received: 6 September 2021
Accepted: 27 October 2021
Published: 8 November 2021

1. Introduction

The study of iterative methods for solving nonlinear equations and systems appears to be a very important area in theory and practice. Such problems appear not only in applied mathematics but also in many branches of science including engineering (design of an electric circuit), physics (pipe friction), chemistry (greenhouse gases and rainwater), biology (steady-state of Lotka–Volterra system), fluid dynamics (combined conductive–radiative heat transfer), environmental engineering (oxygen level in a river downstream from a sewage discharge), finance (option pricing), and many more. The study of nonlinear models is a vital area of research in numerical analysis. Interesting applications in pure and applied sciences can be studied in the general construction of the nonlinear equations expressed in the form $g(x) = 0$. Due to their significance, several iterative methods have been devised under certain situations since it is near to impossible to obtain exact solutions

of models that are nonlinear in nature. These iterative methods have been constructed using different existing methods such as Taylor expansion, the perturbation method, the homotopy perturbation method, Adomian decomposition method, quadrature formula, multi-point iterative methods, the Steffensen-type methods adapted to multidimensional cases, and the variational iteration method. For detailed information, see [1–8]. Among existing iterative methods, the optimal methods are considered those that satisfy the condition for an order of convergence of 2^{k-1}, where k stands for the number of function evaluations per iteration as suggested in [9]. In this way, Newton's classical method $x_{n+1} = x_n - \frac{g(x_n)}{g'(x_n)}$ is the optimal method with quadratic convergence. Various attempts have been made to improve the efficiency of Newton's classical method in past and recent research, as can be seen in [10–15] and most of the references cited therein.

2. Materials and Methods

In a general form, the uni-variate nonlinear equation can be expressed as $g(x) = 0$, where x is the desired quantity. It is extremely difficult to solve the nonlinear equation to find the value of x. Therefore, we attempt to devise a new, highly convergent iterative method to obtain an accurate approximate x that could yield the smallest possible error in the numerical solution. Before we continue with a discussion and derivation of the proposed method, we present some of the methods that are frequently used in the available literature. Later, we use these methods to compare the results obtained under these methods and the results obtained via the method we plan to propose.

2.1. Some Existing Methods

The iterative method ,called the Newton Rahpson method NR_2 [1,16,17] with quadratic convergence, is shown below and uses two function evaluations: one for the function $g(x)$ itself and 1 for the first derivative $g'(x)$:

$$x_{n+1} = x_n - \frac{g(x_n)}{g'(x_n)}, \; n = 0, 1, 2, \ldots, \tag{1}$$

where $g'(x_n) \neq 0$.

In [2], the authors proposed an iterative method with fifth-order convergence as abbreviated by MHM_5. The method requires four function evaluations per iteration: two for the function itself and two first derivatives. The computational steps for the two-step method MHM_5 is described below:

$$\begin{aligned} y_n &= x_n - \frac{g(x_n)}{g'(x_n)}, \\ x_{n+1} &= y_n - \frac{g'(y_n) + 3g'(x_n)}{5g'(y_n) - g'(x_n)} . \frac{g(y_n)}{g'(x_n)}, \end{aligned} \qquad n = 0, 1, 2, \ldots, \tag{2}$$

where $g'(x_n) \neq 0$ and $5g'(y_n) \neq g'(x_n)$.

An efficient three-step iterative method with sixth-order convergence is proposed in [3]. This method is the combination of two different methods from [1,18] with second and third-order convergence, respectively. The method requires five function evaluations: three evaluations of the function itself and two evaluations of the first-order derivative per iteration. We represent the method as HM_6. The computational steps of the method can be described as follows:

$$\begin{aligned} y_n &= x_n - \frac{g(x_n)}{g'(x_n)}, \\ z_n &= y_n - \frac{g(y_n)}{g'(y_n)}, \\ x_{x+1} &= y_n - \frac{g(y_n) + g(z_n)}{g'(y_n)}. \end{aligned} \qquad n = 0, 1, 2, \ldots, \tag{3}$$

The three-step method with eighth-order convergence as denoted by WO_8 is proposed in [19]. The method requires four function evaluations: three evaluations of the function itself and one evaluation of the first-order derivative per iteration. The computational steps of WO_8 can be described as follows:

$$\begin{aligned} y_n &= x_n - \frac{g(x_n)}{g'(x_n)}, \\ z_n &= x_n - \frac{g(x_n)}{g'(x_n)} \frac{4g(x_n)^2 - 5g(x_n)g(y_n) - g(y_n)^2}{4g(x_n)^2 - 9g(x_n)g(y_n)}, \\ x_{n+1} &= z_n - \frac{g(z_n)}{g'(x_n)} \left[1 + 4\frac{g(z_n)}{g(x_n)}\right] \left[\frac{8g(y_n)}{4g(x_n) - 11g(y_n)} + 1 + \frac{g(z_n)}{g(y_n)}\right]. \end{aligned} \qquad n = 0, 1, 2, \ldots, \quad (4)$$

The iterative method with ninth-order convergence can be seen in [20]. The method requires five function evaluations: three evaluations of function itself and two evaluations of the first-order derivative per iteration. This method is abbreviated as NM_9. The computational steps of the method are described as follows:

$$\begin{aligned} y_n &= x_n - \frac{g(x_n)}{g'(x_n)}, \\ z_n &= y_n - \left[1 + \left(\frac{g(y_n)}{g(x_n)}\right)^2\right] \frac{g(y_n)}{g'(y_n)}, \\ x_{n+1} &= z_n - \left[1 + 2\left(\frac{g(y_n)}{g(x_n)}\right)^2 + 2\frac{g(z_n)}{g(y_n)}\right] \frac{g(z_n)}{g'(y_n)}. \end{aligned} \qquad n = 0, 1, 2, \ldots, \quad (5)$$

The predictor–corrector modified Householder's method with tenth order convergence, as denoted by MH_{10}, is proposed in [21]. The method is free from the second derivative and requires only five function evaluations per iteration: three evaluations of the function itself and two evaluations of the first-order derivative. The computational steps of MH_{10} can be described as

$$\begin{aligned} y_n &= x_n - \frac{g(x_n)}{g'(x_n)}, \\ z_n &= y_n - \frac{g(y_n)}{g'(y_n)} - \frac{g^2(y_n)P(y_n)}{2g'^3(y_n)}, \\ x_{n+1} &= z_n - \frac{g(z_n)}{g[z_n, y_n] + (z_n - y_n)g[z_n, y_n, y_n]}, \end{aligned} \qquad n = 0, 1, 2, \ldots, \quad (6)$$

where

$$\begin{aligned} P(y_n) &= g''(y_n) = \frac{2}{x_n - y_n}\left[3\frac{g(x_n) - g(y_n)}{x_n - y_n} - 2g'(y_n) - g'(x_n)\right], \\ g[z_n, y_n] &= \frac{g(z_n) - g(y_n)}{z_n - y_n}, \\ g[z_n, y_n, y_n] &= \frac{g[z_n, y_n] - g'(y_n)}{z_n - y_n}. \end{aligned} \quad (7)$$

2.2. Proposed Iterative Method

There are many recent research studies wherein researchers have presented modified iterative methods to solve nonlinear models of the form $g(x) = 0$. In some of these methods, modification is based on the idea of combining two existing methods to develop a new method with a better order of convergence. After being motivated by such an idea as observed in [3,22–24], we have developed a new method with tenth-order convergence via the blending of two different iterative methods with second (Newton method) and fifth-order (modified Halley method) convergence as given in [1,2], respectively. We recommended the higher-order convergent method of the convergence order $2 \times 5 = 10$. The choice of the methods in the present work is suitable because the resultant iterative method with tenth order convergence uses only 6 function evaluations (3 function evaluations and

three evaluations of the first-order derivative) per iteration. It may be noted that the choice of blending of methods is extremely important to avoid extra function evaluations that could bring additional computational cost, as can be seen in [25,26]. Hence, the proposed iterative method not only confirms higher-order convergence but also employs fewer function evaluations, as can be described by the following proposed three-step method abbreviated as PM_{10}:

$$\begin{aligned} y_n &= x_n - \frac{g(x_n)}{g'(x_n)}, \\ z_n &= y_n - \frac{g(y_n)}{g'(y_n)}, \\ x_{x+1} &= z_n - \frac{g'(z_n) + 3g'(y_n)}{5g'(z_n) - g'(y_n)} \left(\frac{g(z_n)}{g'(y_n)} \right). \end{aligned} \qquad n = 0, 1, 2, \ldots, \tag{8}$$

The proposed iterative three-step method given in (8) is discussed in the flowchart presented in Figure 1.

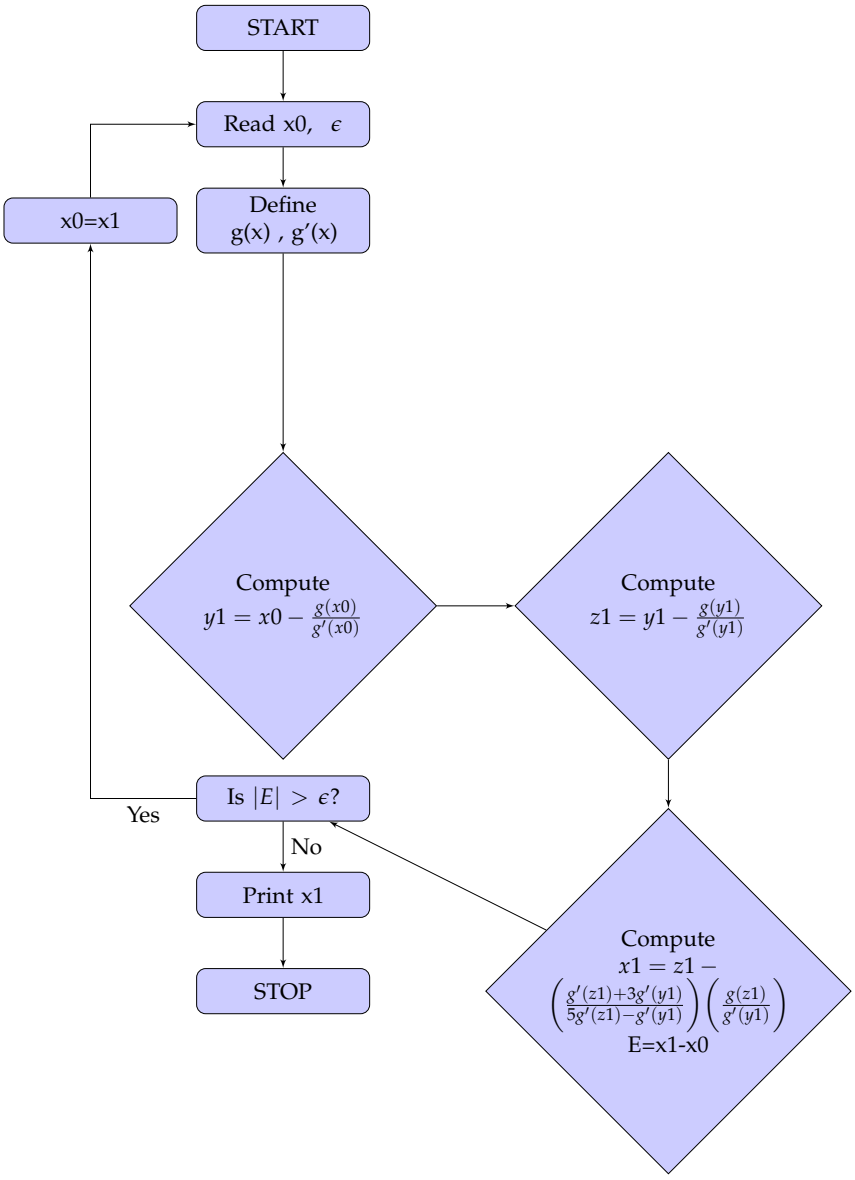

Figure 1. Flowchart of the proposed three-step method given in (8).

Furthermore, the efficiency index e is also computed for the proposed iterative method as $10^{1/6} \approx 1.4678$, whereas it would generally be shown as $10^{1/3(n+n^2)}$ for $n \geq 1$. The following Table 1 and the Figure 2 can be consulted for the computation and comparison of all iterative methods taken for comparison in the present research work. Although the function evaluations (FV) of PM_{10} in the Table 1 seem to be more than some of the methods under consideration, to achieve the desired accuracy regarding the performance of the latter under different initial guesses (IG), the number of iterations (N) and CPU time (seconds) are better than most of the methods. This discussion is presented in Section 5.

Table 1. Comparison of efficiency indices for methods under consideration.

Method	Order	FV	EI	New Function Evaluations per Iteration for $n \geq 1$.
PM_{10}	10	6	1.4678×10^0	$3(n+n^2)$
MH_{10}	10	5	1.5849×10^0	$3n+2n^2$
NM_9	9	5	1.5518×10^0	$3n+2n^2$
WO_8	8	4	1.6818×10^0	$3n+n^2$
HM_6	6	5	1.4310×10^0	$3n+2n^2$
MHM_5	5	4	1.4953×10^0	$2(n+n^2)$
NR_2	2	2	1.4142×10^0	$n+n^2$

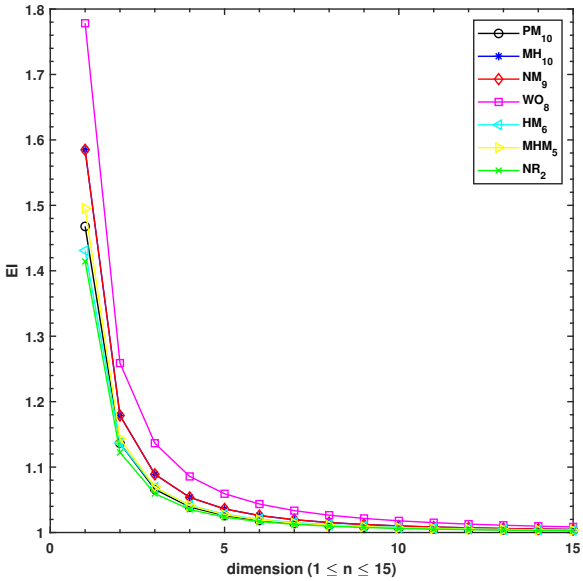

Figure 2. Behavior of efficiency index of various iterative methods for increasing dimensions of the nonlinear problem.

3. Convergence Analysis

This section has been devoted to the proof of local convergence analysis for the proposed tenth-order method under both scalar and vector (systems) cases. Single and multivariate Taylor's series expansion have been used to obtain the required order of local convergence. It is worth noting that the convergence analysis is addressed in a similar manner to many other existing articles, and the main interest in developing higher-order methods is of the academic type. Even if higher-order methods are more complicated, the efficiency can be measured, and this is why we have included the CPU time, as found in Section 5. The theorems stated below are later used for the theoretical analysis of the convergence.

Theorem 1. *(Single real variable Taylor's series expansion): Suppose that $r \geq 1$ is an integer and further suppose that $g : \mathbb{R} \to \mathbb{R}$ is an r-times differentiable function at some finite point $\alpha \in \mathbb{R}$. Then, there exists the following expression:*

$$g(x_n) = g(\alpha) + g'(\alpha)\delta_n + \frac{1}{2!}g''(\alpha)\delta_n^2 + \cdot + \frac{1}{r!}g^{(r)}(\alpha)\delta_n^r + R_r(x_n), \tag{9}$$

where $R_r(x)$ is the remainder term, whose integral form is

$$R_r(x_n) = \int_{\alpha}^{x_n} \frac{1}{(r+1)!}g^{(r+1)}(t)(x_n - t)^r dt. \tag{10}$$

Theorem 2. *(Multivariable Taylor's series expansion): Suppose that $\boldsymbol{G} : P \subseteq \mathbb{R}^n \to \mathbb{R}^n$ is an r-times Frechet differentiable system of functions in a convex set $P \subseteq \mathbb{R}^n$; then, for any $\boldsymbol{x}$ and $k \in \mathbb{R}^n$ the equation given below is true:*

$$\boldsymbol{G}(\boldsymbol{x}+k) = \boldsymbol{G}(\boldsymbol{x}) + k\boldsymbol{G}'(\boldsymbol{x}) + \frac{k^2}{2!}\boldsymbol{G}''(\boldsymbol{x}) + \frac{k^3}{3!}\boldsymbol{G}''(\boldsymbol{x}) + \ldots + \frac{k^{r-1}}{r!}\boldsymbol{G}^{(r-1)}(\boldsymbol{x}) + R_r, \tag{11}$$

where $||R_r|| \leq \frac{1}{r!}\sup_{0<t<1}||\boldsymbol{G}^{(r)}(\boldsymbol{x}+kt)||||k||^r$ and $\boldsymbol{G}^{(q)}(\boldsymbol{x})k^q = (\ldots(\boldsymbol{G}^{(q)}(\boldsymbol{x})k).^{q}.)k \in \mathbb{R}^n$.

3.1. Convergence under Scalar Case

In this subsection, we theoretically prove the local order of convergence for PM_{10}.

Theorem 3. *Suppose that $\alpha \in P$ is the required simple root for a differentiable function $g : P \subseteq \mathbb{R} \to \mathbb{R}$ within an open real interval P. Then, the proposed three-step numerical method (8) possesses tenth-order convergence, and the asymptotic error term is given by*

$$\epsilon_{x_{n+1}} = \frac{g'^7(\alpha)}{4096g'^9(\alpha)}\Big(52g'(\alpha)g'''(\alpha) - 35g''^2(\alpha)\Big)\epsilon_{x_n}^{10} + \mathcal{O}(\epsilon_{x_n}^{11}). \tag{12}$$

Proof. Suppose α is the root of $g(x_n)$, where x_n is the nth approximation for the root by the proposed method (8), and $\epsilon_{x_n} = x_n - \alpha$ is the error term in variable x at the nth iteration step. Employing the single real variable Taylor's series given in the theorem (1) for $g(x_n)$ around α, we obtain

$$g(x_n) = g'(\alpha)\epsilon_{x_n} + \frac{1}{2!}g''(\alpha)\epsilon_{x_n}^2 + \mathcal{O}(\epsilon_{x_n}^3). \tag{13}$$

Similarly, using the Taylor's series for $1/g'(x_n)$ around α, we obtain

$$\frac{1}{g'(x_n)} = \frac{1}{g'(\alpha)} - \frac{g''(\alpha)}{g'^2(\alpha)}\epsilon_{x_n} + \frac{1}{g'(\alpha)}\Big(\frac{g''^2(\alpha)}{g'^2(\alpha)} - \frac{g'''(\alpha)}{2g'(\alpha)}\Big)\epsilon_{x_n}^2 + \mathcal{O}(\epsilon_{x_n}^3). \tag{14}$$

Multiplying (13) and (14), we obtain

$$\frac{g(x_n)}{g'(x_n)} = \epsilon_{x_n} - \frac{g''(\alpha)}{2g'(\alpha)}\epsilon_{x_n}^2 + \Big(\frac{g''^2(\alpha)}{2g'^2(\alpha)} - \frac{g'''(\alpha)}{2g'(\alpha)}\Big)\epsilon_{x_n}^3 + \mathcal{O}(\epsilon_{x_n}^4). \tag{15}$$

Now, substituting (15) in the first step of (8), we obtain

$$\epsilon_{y_n} = \frac{g''(\alpha)}{2g'(\alpha)}\epsilon_{x_n}^2 + \frac{1}{2g'^2(\alpha)}\Big(g'(\alpha)g'''(\alpha) - g''^2(\alpha)\Big)\epsilon_{x_n}^3 + \mathcal{O}(\epsilon_{x_n}^4), \tag{16}$$

where $\epsilon_{y_n} = y_n - \alpha$. Using the Taylor's series (1) for $g(y_n)$ around α, we obtain

$$g(y_n) = g'(\alpha)\epsilon_{y_n} + \frac{1}{2!}g''(\alpha)\epsilon_{y_n}^2 + \mathcal{O}(\epsilon_{y_n}^3). \tag{17}$$

Similarly, using the Taylor's series for $\frac{1}{g'(y_n)}$ around α, we obtain

$$\frac{1}{g'(y_n)} = \frac{1}{g'(\alpha)} - \frac{g''(\alpha)}{g'^2(\alpha)}\epsilon_{y_n} + \frac{1}{g'(\alpha)}\Big(\frac{g'^2(\alpha)}{g'^2(\alpha)} - \frac{g'''(\alpha)}{2g'(\alpha)}\Big)\epsilon_{y_n}^2 + \mathcal{O}(\epsilon_{y_n}^3). \tag{18}$$

Multiplying (17) and (18), we obtain

$$\frac{g(y_n)}{g'(y_n)} = \epsilon_{y_n} - \frac{g''(\alpha)}{2g'(\alpha)}\epsilon_{y_n}^2 + \Big(\frac{g'^2(\alpha)}{2g'^2(\alpha)} - \frac{g'''(\alpha)}{2g'(\alpha)}\Big)\epsilon_{y_n}^3 + \mathcal{O}(\epsilon_{y_n}^4). \tag{19}$$

Now, substituting (19) in the second step of (8), we obtain

$$\epsilon_{z_n} = \frac{g''(\alpha)}{2g'(\alpha)}\epsilon_{y_n}^2 + \frac{1}{2g'^2(\alpha)}\Big(g'(\alpha)g'''(\alpha) - g'^2(\alpha)\Big)\epsilon_{y_n}^3 + \mathcal{O}(\epsilon_{y_n}^4), \tag{20}$$

where $\epsilon_{z_n} = z_n - \alpha$. Using the Taylor's series (1) for $g(z_n)$ around α, we obtain

$$g(z_n) = g'(\alpha)\epsilon_{z_n} + \frac{1}{2}g''(\alpha)\epsilon_{z_n}^2 + \mathcal{O}(\epsilon_{z_n}^3). \tag{21}$$

Using the Taylor's series (1) for $g'(y_n)$ around α, we obtain

$$g'(y_n) = g'(\alpha) + g''(\alpha)\epsilon_{y_n} + \frac{1}{2}g'''(\alpha)\epsilon_{y_n}^2 + \mathcal{O}(\epsilon_{y_n}^3). \tag{22}$$

Using the Taylor's series (1) for $g'(z_n)$ around α, we obtain

$$g'(z_n) = g'(\alpha) + g''(\alpha)\epsilon_{z_n} + \frac{1}{2}g'''(\alpha)\epsilon_{z_n}^2 + +\mathcal{O}(\epsilon_{z_n}^3). \tag{23}$$

Expanding the Taylor series $\frac{1}{5g'(z_n) - g'(y_n)}$ and using Equations (22) and (23), we obtain

$$\begin{aligned}\frac{g'(z_n) + 3g'(y_n)}{5g'(z_n) - g'(y_n)} &= 1 + \frac{g''(\alpha)}{g'(\alpha)}\Big(\epsilon_{y_n} - \epsilon_{z_n}\Big) + \left(\frac{g'''(\alpha)}{2g'(\alpha)} + \frac{g'^2(\alpha)}{4g'^2(\alpha)}\right)\epsilon_{y_n}^2 - \left(\frac{g'''(\alpha)}{2g'(\alpha)} - \frac{5g'^2(\alpha)}{4g'^2(\alpha)}\right)\epsilon_{z_n}^2 \\ &\quad - \frac{3g'^2(\alpha)}{2g'^2(\alpha)}\epsilon_{y_n}\epsilon_{z_n} + \mathcal{O}(\epsilon_{z_n}^3).\end{aligned} \tag{24}$$

Finally, substituting (24) in the third step of (8) and using Equations (16), (18), (20) and (21), we obtain

$$\epsilon_{x_{n+1}} = \frac{g'^7(\alpha)}{4096g'^9(\alpha)}\Big(52g'(\alpha)g'''(\alpha) - 35g'^2(\alpha)\Big)\epsilon_{x_n}^{10} + \mathcal{O}(\epsilon_{x_n}^{11}). \tag{25}$$

Hence, the tenth-order convergence of the proposed method PM_{10} given by (8) for $g(x) = 0$ is proved. □

3.2. Convergence under Vector Case

This subsection extends the proof for the tenth-order convergence of the proposed method PM_{10} given in (8) regarding solving the system of nonlinear functions $\boldsymbol{G}(\boldsymbol{x}) = \boldsymbol{0}$, where $\boldsymbol{G} = [g_1(\boldsymbol{x}), g_2(\boldsymbol{x}), \ldots, g_n(\boldsymbol{x})]'$ from $\boldsymbol{R}^n$ to $\boldsymbol{R}^n$. For the system of nonlinear functions, PM_{10} can be described as follows:

$$\begin{aligned} y_n &= x_n - G'(x_n)^{-1}G(x_n), \\ z_n &= y_n - G'(y_n)^{-1}G(y_n), \\ x_{n+1} &= z_n - \left[5G'(z_n) - G'(y_n)\right]^{-1}\left[G'(z_n) + 3G'(y_n)\right]\left[G'(y_n)^{-1}G(z_n)\right]. \end{aligned} \qquad n = 0, 1, 2, \ldots. \tag{26}$$

We present the following theorem to obtain the asymptotic error term and the order of convergence for PM_{10}.

Theorem 4. *Let the function $G : P^n \subset R^n \to R^n$ be sufficiently differentiable in a convex set P^n containing the zero α of $G(x)$. Let us consider that $G'(x)$ is continuous and nonsingular in α. Then, the solution x obtained by using proposed three-step method PM_{10} converging to α has tenth-order convergence, if an initial guess x_0 is chosen close to α.*

Proof. Suppose that $\epsilon_{xn} = ||x_n - \alpha||$. Now, using the Taylor series described in the Theorem (2) for $G(x_n)$, we obtain

$$G(x_n) = G'(\alpha)\epsilon_{xn} + \frac{1}{2!}G''(\alpha)\epsilon_{x_n}^2 + \mathcal{O}(\epsilon_{x_n}^3). \tag{27}$$

Similarly, using the Taylor's series for $G'(x_n)$ around α, we obtain

$$G'(x_n) = G'(\alpha) + G''(\alpha)\epsilon_{xn} + \frac{1}{2!}G'''(\alpha)\epsilon_{x_n}^2 + \mathcal{O}(\epsilon_{x_n}^3). \tag{28}$$

Employing the Taylor's series for the inverted Jacobian matrix $G'(x_n)^{-1}$ around α, we obtain

$$\begin{aligned} G'(x_n)^{-1} = G'(\alpha)^{-1} - G'^2(\alpha)^{-1}G''(\alpha)\epsilon_{xn} + \\ G'(\alpha)^{-1}\left[G'^2(\alpha)^{-1}G''^2(\alpha) - 2G'(\alpha)^{-1}G'''(\alpha)\right]\epsilon_{x_n}^2 + \mathcal{O}(\epsilon_{x_n}^3). \end{aligned} \tag{29}$$

Multiplying (27) and (29), we obtain

$$G'(x_n)^{-1}G(x_n) = \epsilon_{xn} - 2G'(\alpha)^{-1}G''(\alpha)\epsilon_{x_n}^2 + \left[2G'^2(\alpha)^{-1}G''^2(\alpha) - 2G'(\alpha)^{-1}G'''(\alpha)\right]\epsilon_{x_n}^3 + \mathcal{O}(\epsilon_{x_n}^4). \tag{30}$$

Now, substituting (30) in the first step of (26), we obtain

$$\epsilon_{y_n} = 2G'(\alpha)^{-1}G''(\alpha)\epsilon_{x_n}^2 + 2G'^2(\alpha)^{-1}\left[G'(\alpha)G'''(\alpha) - G''^2(\alpha)\right]\epsilon_{x_n}^3 + \mathcal{O}(\epsilon_{x_n}^4). \tag{31}$$

Similarly, employing the Taylor series for $G(y_n)$ and $G'(y_n)$ about α, and also for the inverted Jacobian matrix $G'(y_n)^{-1}$ in the second step of (26), we obtain

$$\epsilon_{zn} = 2G(\alpha)^{-1}G''(\alpha)\epsilon_{y_n}^2 + 2G'^2(\alpha)^{-1}\left[G'(\alpha)G'''(\alpha) - G''^2(\alpha)\right]\epsilon_{y_n}^3 + \mathcal{O}(\epsilon_{y_n}^4). \tag{32}$$

Using the Taylor series for $G'(z_n)$ and $G'(y_n)$ around α, we obtain the following:

$$G'(z_n) = G'(\alpha) + G''(\alpha)\epsilon_{zn} + \frac{1}{2}G'''(\alpha)\epsilon_{z_n}^2 + \mathcal{O}(\epsilon_{z_n}^3), \tag{33}$$

$$G'(y_n) = G'(\alpha) + G''(\alpha)\epsilon_{y_n} + \frac{1}{2}G'''(\alpha)\epsilon_{y_n}^2 + \mathcal{O}(\epsilon_{y_n}^3). \tag{34}$$

Using the above two equations and the inverted Jacobian matrix for $\left[5G'(z_n) - G'(y_n)\right]^{-1}$, we obtain

$$\begin{aligned} &\left[5G'(z_n) - G'(y_n)\right]^{-1}\left[G'(z_n) + 3G'(y_n)\right] = 1 - G'(\alpha)^{-1}G''(\alpha)\epsilon_{zn} - 2G'(\alpha)^{-1}G'''(\alpha)\epsilon_{z_n}^2 + \\ &G'(\alpha)^{-1}G''(\alpha)\epsilon_{y_n} + \left[2G'(\alpha)^{-1}G'''(\alpha) + 4G'^2(\alpha)^{-1}G''^2(\alpha)\right]\epsilon_{y_n}^2 + 4G'(\alpha)^{-1}5G''^2(\alpha)\epsilon_{z_n}^2 \\ &- 2G'^2(\alpha)^{-1}3G''^2(\alpha)\epsilon_{y_n}\epsilon_{zn} + \mathcal{O}(\epsilon_{z_n}^3). \end{aligned} \tag{35}$$

Finally, substituting (35) and other required expansions in the third step of (26), we obtain

$$\epsilon_{z_{n+1}} = \frac{1}{4096} G'^{9}(\alpha)^{-1} G''^{7}(\alpha) \left[52 G'(\alpha) G'''(\alpha) - 35 G''^{2}(\alpha)\right] \epsilon_{z_n}^{10} + \mathcal{O}(\epsilon_{z_n}^{11}). \quad (36)$$

Hence, the tenth-order convergence of the proposed method PM_{10} given by (26) for a non-linear system of equations $\boldsymbol{G}(\boldsymbol{x}) = \boldsymbol{0}$ is proved. □

3.3. Computational Estimation of the Convergence Order

When a new iterative method is proposed to compute an approximate solution for $g(x) = 0$, in order to verify its theoretical local order of convergence, one needs to use a parameter called the Computational Order of Convergence (COC). However, this is possible only when we have information about the exact root α for $g(x) = 0$. Thus, the following parameters can alternatively be employed under some constraints as described below.

Approximated Computational Order of Convergence [27]:

$$ACOC = \frac{log|\epsilon_n/\epsilon_{n-1}|}{log|\epsilon_{n-1}/\epsilon_{n-2}|}, \quad \epsilon_n = x_n - x_{n-1}, \quad n \geq 4. \quad (37)$$

Computational Order of Convergence [28]:

$$COC = \frac{log|\overline{\epsilon}_n/\overline{\epsilon}_{n-1}|}{log|\overline{\epsilon}_{n-1}/\overline{\epsilon}_{n-2}|}, \quad \overline{\epsilon} = x_n - \alpha, \quad n \geq 3. \quad (38)$$

Extrapolated Computational Order of Convergence [29]:

$$ECOC = \frac{log|\hat{\epsilon}_n/\hat{\epsilon}_{n-1}|}{log|\hat{\epsilon}_{n-1}/\hat{\epsilon}_{n-2}|}, \quad \hat{\epsilon} = x_n - \alpha_n, \quad n \geq 5, \quad (39)$$

where

$$\alpha_n = x_n - \frac{(\rho x_{n-1})^2}{\rho^2 x_{n-1}}, \quad \rho x_n = x_{n+1} - x_n.$$

Petkovic Computational Order of Convergence [30]:

$$PCOC = \frac{log|\widehat{\epsilon}_n|}{log|\widehat{\epsilon}_{n-1}|}, \quad \widehat{\epsilon}_n = \frac{f(x_n)}{f(x_{n-1})}, \quad n \geq 2. \quad (40)$$

All of the above formulas can be used to test the convergence order, but in the present study, ACOC as given by (37) is used since the number of iterations taken by the method PM_{10} (8) is at least four in various numerically tested problems, as discussed in Section 5. Additionally, this is the best-known approach employed in most of the recently conducted research studies to verify the theoretical order of convergence.

4. Basins of Attraction

The stability of solutions (roots) for the nonlinear function $g(z) = 0$ using an iterative method can be analyzed with the help of a concept called the basins of attraction. Basins of attractions are phase-planes that demonstrate iterations employed by the iterative method, which can assume different choices for the initial guess z_0. Such 2D regions are esthetically so beautiful that their applications are not only found in applied mathematics, but also people working in fields such as architecture, arts, and design also use the concept to obtain pleasing designs. Many other fields of applications for these basins can be seen in [31–36] and most of the references cited therein. It may be noted that linear models do not depict such dynamically eye-catching behavior, whereas the non-linearity results in features such as those seen herein under the proposed iterative method PM_{10}. MATLAB's built-in routines, including contour, colormap, and color bar, are utilized to obtain the basins of

attraction in the present study. In this connection, a squared region R on $[-4,4] \times [-4,4]$ containing 2000 by 2000 mesh points is selected for the selected functions in complex form. Some of these complex-valued functions are taken as follows for the illustration of the regions by the proposed method.

Example 1.

$$\begin{aligned} &P_1(z) = z^8 + z^5 - 4, \quad P_2(z) = z^7 - 1, \quad P_3(z) = z^3 - 1, \\ &P_4(z) = cos(z) + cos(2z) + z, \quad P_5(z) = \exp(z) - z, \quad P_6(z) = \cosh(z) - 1. \end{aligned} \tag{41}$$

To maintain diversity, different kinds of functions including polynomials and transcendental functions are used. The regions are achieved with a tolerance of $\epsilon = 10^{-2}$ and a maximum number of iterations allowed of $n = 12$. As can be seen in Figures 3–8, for the Example (1), the maximum number of iterations is needed by initial guesses that reside near boundaries of the regions, whereas if z_0 lies within the neighborhood of the required solution, the proposed method PM_{10} does not require as many iterations as those needed by most of the methods found in the literature. The average time required to produce the dynamical planes shown in Figures 3–8 is stored in the Table 2. As expected, the complex functions that have exponential and hyperbolic components are computationally expensive.

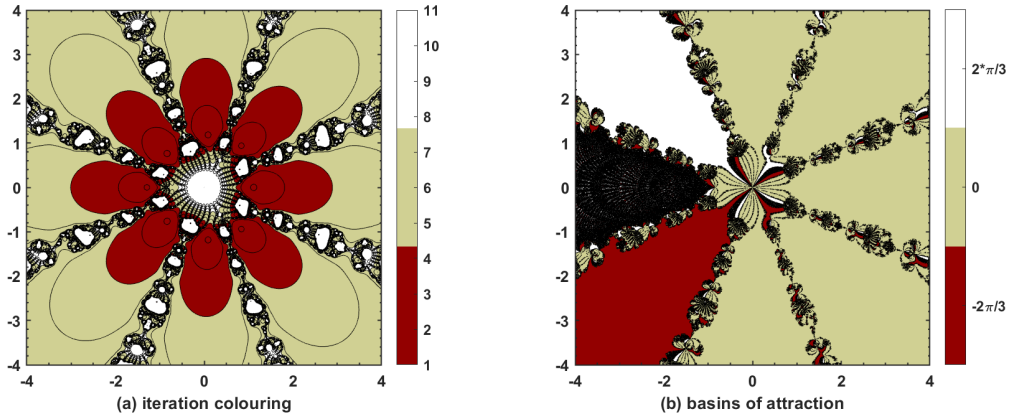

Figure 3. The polynomiographs by the proposed method PM_{10} for $P_1(z)$.

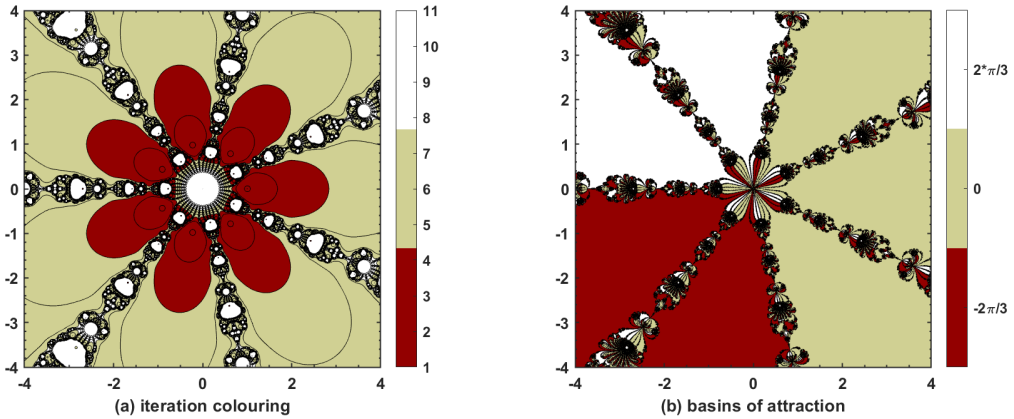

Figure 4. The polynomiographs by the proposed method PM_{10} for $P_2(z)$.

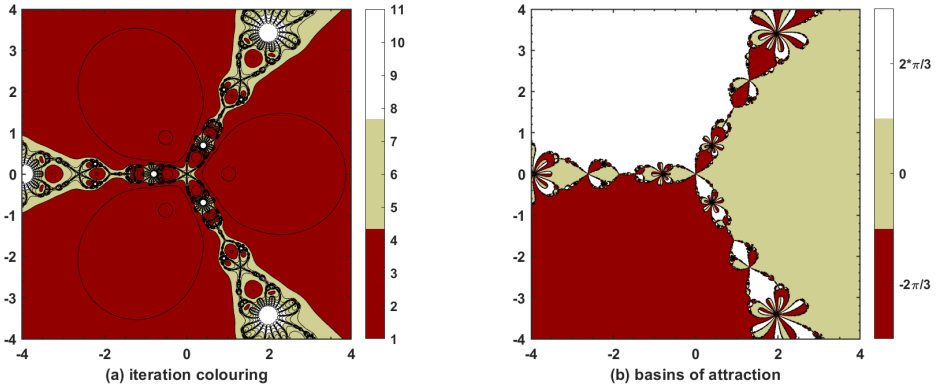

Figure 5. The polynomiographs by the proposed method PM_{10} for $P_3(z)$.

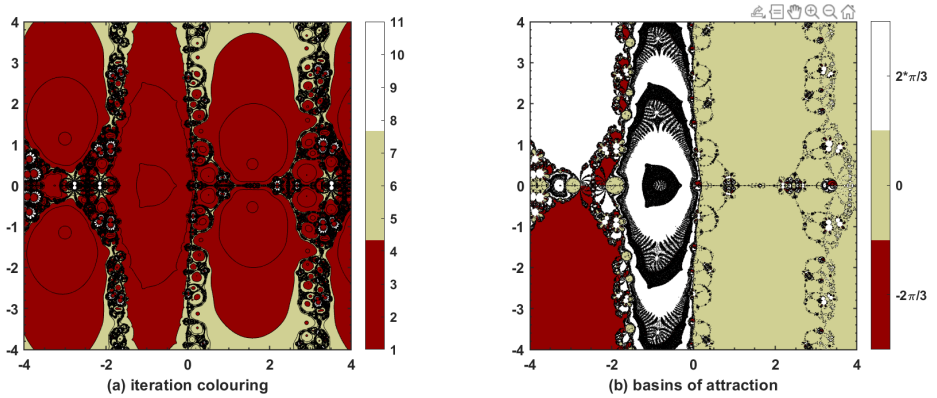

Figure 6. The basins of attractions by the proposed method PM_{10} for $P_4(z)$.

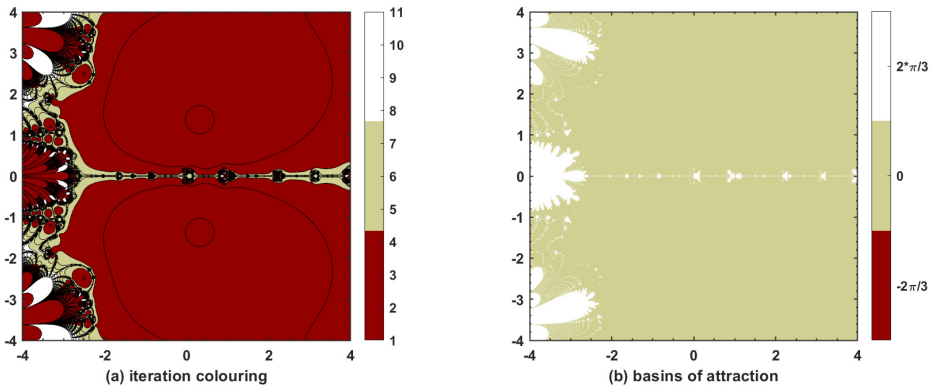

Figure 7. The basins of attractions by the proposed method PM_{10} for $P_5(z)$.

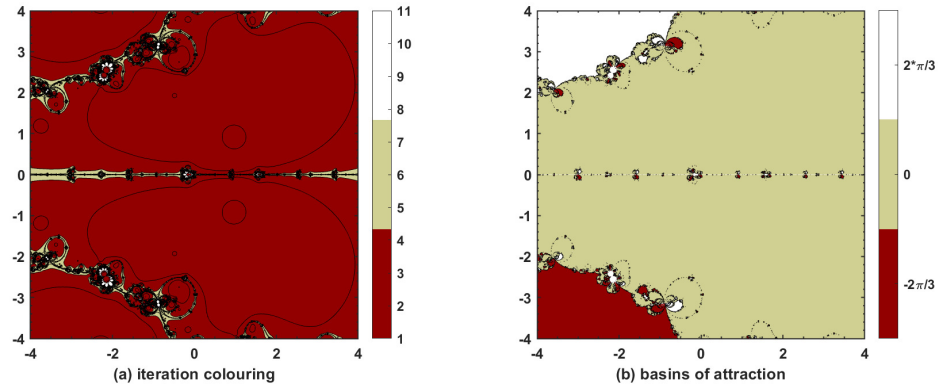

Figure 8. The basins of attractions by the proposed method PM_{10} for $P_6(z)$.

Table 2. Execution time in seconds required by PM_{10} for each $P_i(z)$, $i = 1, 2, \ldots, 6$.

$P_1(z)$	$P_2(z)$	$P_3(z)$	$P_4(z)$	$P_5(z)$	$P_6(z)$
767.413 s	570.103 s	289.626 s	537.531 s	782.588 s	730.300 s

5. Numerical Experiments with Discussion

Various types of test problems are considered from different sources including [3,19,22]. The approximate solutions $x^\star$ up to 50 decimal places are shown against each test function. The error tolerance $|\epsilon_N| = |x_N - x^\star|$ to stop the number of iterations is set to 10^{-200}, whereas the precision is chosen to be as large as 4000, and N shows the total number of iterations taken by the method to achieve the required tolerance. In addition, physically applicable nonlinear models such as the Van der Waals equation, the Shockley ideally diode electric circuit model, the conversion of substances in a batch reactor, and the Lorenz equations in meteorology are taken into consideration to demonstrate the applicability of the proposed method PM_{10}. Obtained numerical results are tabulated for further analysis. Each table contains different initial guesses, numbers of iterations required by a method to achieve the preset error tolerance, function evaluations needed for each method, approximated computational orders of convergence $\left(ACOC = \frac{log|\epsilon_n/\epsilon_{n-1}|}{log|\epsilon_{n-1}/\epsilon_{n-2}|} \right)$ where $\epsilon_n = x_n - x_{n-1}$, absolute errors, absolute values of functions at the last iteration, and the execution (CPU) time in seconds.

For the test problem 2 ($g_1(x)$), two initial guesses are chosen for simulations, as can be seen in Table 3. For the initial guess $x_0 = 1.5$, it is observed that the minimum number of iterations is taken by the methods PM_{10} and WO_8 to achieve the error tolerance; however, the smallest error is given by PM_{10} while consuming an equivalent amount of CPU time. The method NR_2 takes the maximum number of iterations at $x_0 = 1.5$. Under the second initial guess $x_0 = 2.0$, although many methods including PM_{10} take the equal number of iterations to achieve $\epsilon = 10^{-200}$, the smallest absolute error and thus smallest absolute functional value is achieved by PM_{10}. This shows that if an initial guess lying near to the solution of $g(x) = 0$ is passed to PM_{10}, then the method yields the smallest error.

For the test problem 2 ($g_2(x)$), two initial guesses are chosen for simulations as can be seen in Table 4. One of them is taken far away from the approximate solution of the quintic equation $g_2(x)$. For $x_0 = -3.8$, the smallest possible absolute error is yielded by WO_8, but at the cost of the maximum number of iterations and largest amount of CPU time. Next comes HM_6 with an absolute error of 9.2097×10^{-670} and greater time efficiency, but it requires one more iteration when compared with PM_{10}, which achieves an absolute error of 1.9515×10^{-572} with only four iterations while consuming a reasonable amount of CPU time. When an initial guess lying near to the root is chosen, the method NM_9 achieves the smallest error, but it requires one extra iteration when compared with PM_{10} and WO_8. The most expensive methods (in terms of N, FV, and CPU time) for this particular test problem seem to be WO_8 and NR_2.

For the transcendental problem 2 ($g_3(x)$), two initial guesses are chosen for simulations, as can be seen in Table 5. Under both of the initial guesses, the maximum numbers of function evaluations are taken by HM_6 followed by PM_{10} to achieve the desired error tolerance. The smallest absolute functional values are obtained with HM_6 and PM_{10}, where NR_2 seems to be the most expensive method in terms of the number of iterations. Although the method MH_{10} consumes the fewest number of CPU seconds, its ACOC is only eight, contradicting the theoretical order of convergence found in [21].

For the transcendental problem 2 ($g_4(x)$), two initial guesses are chosen for simulations as can be seen in Table 6. One of the guesses is taken far away from the approximate solution of $g_4(x)$. Under both initial guesses, it is observed that the method MHM_5 takes the fewest iterations, fewest function evaluations, and fewest CPU seconds with unsatisfactory absolute errors under $x_0 = -9.5$. The method WO_8 diverges for the second initial guess $x_0 = -9.5$, whereas NR_2 is the most expensive method concerning N and

FV, in particular. The proposed method PM_{10} performs reasonably well under the initial guesses and does not diverge under any situation.

For the test problem 2 ($g_5(x)$), three initial guesses are chosen for simulations as can be seen in Table 7. Two of the guesses lie far away from the approximate root of $g_5(x)$. It is easy to observe that the method PM_{10} performs better than other methods, even when the initial guesses are not near to the approximate root, since the number of iterations to attain the required accuracy is the smallest with PM_{10}. Once again, the most expensive method regarding N and FV proves to be NR_2, whereas WO_8 does not succeed towards the desired root when the initial guesses are assumed to be away from the root. The absolute error achieved by PM_{10} with $x_0 = 2.9$ is the smallest when compared with the results of other methods.

Example 2.

$$\begin{aligned}
&g_1(x) = x^3 - 10,\\
&x^\star \approx 2.154434690031883721759293566519350495259344942192 10,\\
&g_2(x) = x^5 + x - 10000,\\
&x^\star \approx 6.308777129972689094767571771783059113377558058211 10,\\
&g_3(x) = \frac{x}{2} - \sin(x),\\
&x^\star \approx 1.895494267033980947144035738093601691751346627385 40,\\
&g_4(x) = x\exp(x^2) - \sin^2(x) + 3\cos(x) + 5,\\
&x^\star \approx -1.207647827130918927009416758356084097760235818949 50,\\
&g_5(x) = \exp^{\sin(x)} - x + 1,\\
&x^\star \approx 2.630664147927903633975327052350598568584731954733 20.
\end{aligned} \tag{42}$$

Table 3. Numerical results for problem 2: $g_1(x)$.

Method	IG	N	FV	ACOC	$\lvert\epsilon\rvert$	$\lvert f(x_N)\rvert$	CPU
PM_{10}	1.5	4	24	10	4.3384×10^{-427}	1×10^{-3998}	1.6×10^{-2}
	2.0	4	24	10	7.3775×10^{-1117}	$7.7598 \times 10^{-11,164}$	2.1720×10^{0}
MH_{10}	1.5	5	25	8	6.1001×10^{-1501}	6×10^{-3999}	3.2×10^{-2}
	2.0	4	20	8	8.7875×10^{-538}	7.6607×10^{-4298}	1.3391×10^{1}
NM_9	1.5	5	25	9	1.3799×10^{-1487}	6×10^{-3999}	1.6×10^{-2}
	2.0	4	20	9	2.5853×10^{-772}	6×10^{-3999}	1.6×10^{-2}
WO_8	1.5	4	16	7.9999×10^{0}	3.7895×10^{-250}	5.4086×10^{-1999}	1.6×10^{-2}
	2.0	4	16	8	2.2967×10^{-676}	1×10^{-3998}	1.6×10^{-2}
HM_6	1.5	5	25	6	4.6527×10^{-496}	6.0868×10^{-2973}	3.1×10^{-2}
	2.0	4	20	6	2.7077×10^{-230}	2.3643×10^{-1378}	2.66×10^{-1}
MHM_5	1.5	5	20	5	2.5498×10^{-291}	3.4832×10^{-1454}	3.1×10^{-2}
	2.0	5	20	5	2.7042×10^{-743}	4.6736×10^{-3714}	1.2180×10^{0}
NR_2	1.5	10	20	2	4.7719×10^{-221}	1.4717×10^{-440}	3.1×10^{-2}
	2.0	9	18	2	4.5282×10^{-288}	1.3253×10^{-574}	3.1×10^{-2}

Table 4. Data using fixed step-size problem 2: $g_2(x)$.

Method	IG	N	FV	ACOC	$\|\epsilon\|$	$\|f(x_N)\|$	CPU
PM_{10}	−3.8	4	24	10	1.9515×10^{-572}	0	3.1×10^{-2}
	8.8	4	24	9.9999×10^{0}	1.7260×10^{-277}	6.0030×10^{-2769}	3.1×10^{-2}
MH_{10}	−3.8	4	20	8	4.4858×10^{-276}	1.2527×10^{-2202}	3.2×10^{-2}
	8.8	5	25	8	1.1988×10^{-910}	0	3.1×10^{-2}
NM_9	−3.8	4	20	9	1.3626×10^{-350}	5.8823×10^{-3149}	3.1×10^{-2}
	8.8	5	25	9	3.4812×10^{-1362}	0	1.5×10^{-2}
WO_8	−3.8	11	44	8	2.4211×10^{-933}	0	9.4×10^{-2}
	8.8	4	16	8.0239×10^{0}	1.7219×10^{-242}	1.1652×10^{-1938}	3.2×10^{-2}
HM_6	−3.8	5	25	6	9.2097×10^{-670}	0	3.1×10^{-2}
	8.8	5	25	6	5.0406×10^{-303}	8.3149×10^{-1813}	3.1×10^{-2}
MHM_5	−3.8	5	20	5	8.1401×10^{-297}	2.1439×10^{-1479}	3.2×10^{-2}
	8.8	5	20	5	1.3486×10^{-221}	2.6761×10^{-1103}	3.1×10^{-2}
NR_2	−3.8	10	20	2	2.1403×10^{-292}	1.1503×10^{-580}	3.2×10^{-2}
	8.8	11	22	2	6.8822×10^{-280}	1.1893×10^{-555}	1.6×10^{-2}

Table 5. Numerical results for problem 2: $g_3(x)$.

Method	IG	N	FV	ACOC	$\|\epsilon\|$	$\|f(x_N)\|$	CPU
PM_{10}	3.5	4	24	10	1.3985×10^{-540}	3×10^{-4000}	1.41×10^{-1}
	2.5	4	24	10	9.4449×10^{-603}	3×10^{-4000}	1.41×10^{-1}
MH_{10}	3.5	4	20	8	4.4067×10^{-244}	1.1093×10^{-1948}	9.4×10^{-2}
	2.5	4	20	8	7.5796×10^{-277}	8.4971×10^{-2211}	9.4×10^{-2}
NM_9	3.5	4	20	9	1.9811×10^{-308}	4.6535×10^{-2771}	2.81×10^{-1}
	2.5	4	20	9	3.3116×10^{-366}	4.7431×10^{-3291}	1.4×10^{-1}
WO_8	3.5	5	20	8	5.0436×10^{-1292}	3×10^{-4000}	2.66×10^{-1}
	2.5	4	16	8	6.6531×10^{-270}	1.4143×10^{-2155}	2.18×10^{-1}
HM_6	3.5	5	25	6	3.9257×10^{-663}	3.8877×10^{-3976}	1.41×10^{-1}
	2.5	5	25	6	7.7054×10^{-742}	5×10^{-4000}	1.56×10^{-1}
MHM_5	3.5	5	20	5	9.1683×10^{-320}	1.7296×10^{-1597}	1.25×10^{-1}
	2.5	5	20	5	1.4187×10^{-412}	1.5343×10^{-2061}	1.25×10^{-1}
NR_2	3.5	10	20	2	5.1202×10^{-289}	1.2423×10^{-577}	9.3×10^{-2}
	2.5	10	20	2	4.5680×10^{-321}	9.8883×10^{-642}	1.09×10^{-1}

Table 6. Numerical results for problem 2: $g_4(x)$ with * showing the divergence of the method.

Method	IG	N	FV	ACOC	$\|\epsilon\|$	$\|f(x_N)\|$	CPU
PM_{10}	−4.5	10	60	10	4.1220×10^{-954}	8×10^{-3999}	4.38×10^{-1}
	−9.5	30	180	10	1.0834×10^{-353}	1.0796×10^{-3527}	1.39×10^{0}
MH_{10}	−4.5	12	60	8	5.5553×10^{-552}	8×10^{-3999}	5.62×10^{-1}
	−9.5	39	195	8	1.0488×10^{-636}	8×10^{-3999}	1.8590×10^{0}
NM_9	−4.5	12	60	9	4.4565×10^{-1467}	8×10^{-3999}	9.69×10^{-1}
	−9.5	37	185	9.0001×10^{0}	6.8×10^{-223}	4.0517×10^{-1997}	3.3590×10^{0}
WO_8	−4.5	17	68	8	2.3107×10^{-1507}	8×10^{-3999}	1.5940×10^{0}
	−9.5	200 *	800	9.5356×10^{-1}	2.0883×10^{-2}	$9.0918 \times 10^{+72}$	1.9266×10^{1}
HM_6	−4.5	13	65	6	1.0498×10^{-243}	4.1588×10^{-1456}	5.47×10^{-1}
	−9.5	43	215	6	3.9815×10^{-227}	1.2374×10^{-1356}	1.8910×10^{0}
MHM_5	−4.5	7	28	5	2.1606×10^{-978}	8×10^{-3999}	2.34×10^{-1}
	−9.5	17	68	5	3.1372×10^{-238}	4.1824×10^{-1186}	5.62×10^{-1}
NR_2	−4.5	30	60	2	1.4030×10^{-243}	6.0043×10^{-485}	4.84×10^{-1}
	−9.5	101	202	2	6.9221×10^{-226}	1.4616×10^{-449}	1.5620×10^{0}

Table 7. Numerical results for problem 2: $g_5(x)$.

Method	IG	N	FV	ACOC	$\lvert\epsilon\rvert$	$\lvert f(x_N)\rvert$	CPU
PM_{10}	2.9	4	24	10	3.0523×10^{-1325}	0	1.57×10^{-1}
	−3.7	5	30	10	3.1294×10^{-1405}	0	2.19×10^{-1}
	7.4	6	36	10	1.2713×10^{-454}	0	2.5×10^{-1}
MH_{10}	2.9	4	20	8	3.3060×10^{-632}	0	9.4×10^{-2}
	−3.7	9	45	8	7.1797×10^{-378}	4.0342×10^{-3023}	2.97×10^{-1}
	7.4	22	110	8	5.3293×10^{-202}	3.7179×10^{-1616}	7.81×10^{-1}
NM_9	2.9	4	20	9	3.6544×10^{-815}	0	2.5×10^{-1}
	−3.7	5	25	9	2.2115×10^{-563}	0	3.13×10^{-1}
	7.4		failed	–	–	–	–
WO_8	2.9	4	16	8	8.7376×10^{-540}	0	1.87×10^{-1}
	−3.7	failed	–	–	–	–	–
	7.4	failed	–	–	–	–	–
HM_6	2.9	4	20	6	2.1574×10^{-305}	3.1362×10^{-1833}	1.41×10^{-1}
	−3.7	6	30	6	3.8865×10^{-904}	0	1.72×10^{-1}
	7.4	12	60	6	3.3038×10^{-311}	4.0442×10^{-1868}	4.22×10^{-1}
MHM_5	2.9	5	20	5	4.0088×10^{-716}	5.2603×10^{-3580}	1.1×10^{-1}
	−3.7	8	32	5	7.1662×10^{-833}	0	2.97×10^{-1}
	7.4	8	32	5	2.16×10^{-326}	2.3890×10^{-1631}	2.19×10^{-1}
NR_2	2.9	9	18	2	4.2361×10^{-377}	3.9782×10^{-754}	9.4×10^{-2}
	−3.7	11	22	2	4.9340×10^{-262}	5.3970×10^{-524}	3.13×10^{-1}
	7.4	16	32	2	1.0369×10^{-370}	2.3835×10^{-741}	2.66×10^{-1}

Example 3. *Volume from Van der Waals' Equation [37].*
The Van der Waals equation is represented by the following model:

$$(P + \frac{an^2}{V^2})(V - bn) = nRT. \tag{43}$$

After some simplifications, one obtains the following polynomial of nonlinear form:

$$g(V) = PV^3 - n(RT + bP)V^2 + n^2aV - n^3ab. \tag{44}$$

The Van der Waals equation of state was formulated in 1873, with two constants a and b (Van der Waals constants) determined from the behavior of a substance at the critical point. The equation is based on two effects, which are the molecular size and attractive force between the molecules. The model (43) is a modified version of the ideal gas equation $V = RT/nP$, where n shows the number of moles, R stands for the universal gas constant (0.0820578), T is the absolute temperature, V shows the volume, and P denotes the absolute pressure. If $V = 1.4$ moles of benzene vapor form under $P = 40$ atm with $a = 18$ and $b = 0.1154$, then one has

$$g_1(V) = 40V^3 - 95.26535116V^2 + 35.28V - 5.6998368. \tag{45}$$

The approximate solution up to 50 dp is given as:

$$V^* = 1.97078421940702941144713037208685635986181216035 38.$$

Being cubic, the Equation (45) certainly possesses one real root. Here, we aim to show the performance of PM_{10} on this model. Therefore, the model is numerically solved by PM_{10} and the other six methods chosen for comparison. It can be observed in Table 8 that PM_{10} achieves the smallest possible error ϵ along with functional values nearest to 0 in a reasonable amount of time, irrespective of the initial guesses.

Table 8. Numerical results for problem 3.

Method	IG	N	FV	ACOC	$\|\epsilon\|$	$\|f(x_N)\|$	CPU
PM_{10}	2.0	4	24	10	4.6111×10^{-1485}	1.3×10^{-3997}	4.7×10^{-2}
	10.3	6	36	10	1.6261×10^{-1641}	1.1×10^{-3997}	6.2×10^{-2}
MH_{10}	2.0	4	20	8	1.5202×10^{-726}	1.1×10^{-3997}	1.6×10^{-2}
	10.3	6	30	8	6.2361×10^{-305}	2.2532×10^{-2431}	1.6×10^{-2}
NM_9	2.0	4	20	9	1.0867×10^{-1015}	1.3×10^{-3997}	1.6×10^{-2}
	10.3	6	30	9	9.6289×10^{-479}	1.3×10^{-3997}	3.2×10^{-2}
WO_8	2	4	16	8	3.9765×10^{-833}	1.1×10^{-3997}	1.6×10^{-2}
	10.3	6	24	8	4.8401×10^{-690}	1.1×10^{-3997}	4.7×10^{-2}
HM_6	2.0	4	20	6	9.3310×10^{-311}	2.9554×10^{-1858}	1.6×10^{-2}
	10.3	7	35	6	1.3806×10^{-532}	3.1008×10^{-3189}	1.6×10^{-2}
MHM_5	2.0	4	16	5	1.6172×10^{-201}	8.3557×10^{-1003}	0
	10.3	7	28	5	7.9826×10^{-804}	1.1×10^{-3997}	0
NR_2	2.0	9	18	2	1.7818×10^{-383}	4.4839×10^{-764}	0
	10.3	15	30	2	1.1910×10^{-271}	2.0034×10^{-540}	1.6×10^{-2}

Example 4. *The Shockley Ideally Diode Electric Circuit Model.*

The Shockley diode model giving the voltage going through the diode V_D is represented by the following equation:

$$J = J_S\Big(\exp(V_D/nV_T) - 1\Big),$$

where J_S stands for the saturation current, n is the emission coefficient, V_T is the thermal voltage, and J is the diode current. Using the Kirchhoff's second law ($V_R + V_D = V_S$) and Ohm's law ($V = JR$), a root-finding model can be found. The final structure for the model would be as follows:

$$V_S = JR + nV_T ln\Big(\frac{J}{J_S} + 1\Big). \tag{46}$$

Assuming values of parameters V_S, R, n, V_T, J_S from [38], we obtain the following equation that is nonlinear in the variable J:

$$g(J) = 1.4ln(J + 1) + 0.1J - 0.5. \tag{47}$$

The approximate solution of the above equation correct to 50 dp is as follows:

$$J^\star = 0.389977198390077586586453532646341189968369462436 62.$$

Table 9 presents numerical simulations for (47) with two different initial conditions of 0.5 and 1.8, under which the smallest absolute error seems to lie under the column of the proposed method PM_{10}. Further analysis can easily be conducted by the careful examination of the results tabulated therein.

Example 5. *Conversion of Species within a Chemical Reactor [39].*

The following nonlinear equation arises in chemical engineering during the conversion of species in a chemical reactor:

$$g(x) = \frac{x}{1-x} - 5\ln\Big(\frac{0.4(1-x)}{0.4 - 0.5x}\Big) + 4.45977, \tag{48}$$

where x stands for the fractional conversion of the species; thus, it must lie in (0,1). The approximate solution of the above equation correct to 50 dp is as follows:

$$x^\star \approx 0.75739624625375387945964129792914529342795578042081.$$

Table 9. Numerical results for problem 4.

Method	IG	N	FV	ACOC	$\|\epsilon\|$	$\|f(x_N)\|$	CPU
PM_{10}	0.5	4	24	10	7.2498×10^{-1614}	2×10^{-4000}	7.8×10^{-2}
	1.8	4	24	10	1.1065×10^{-661}	2×10^{-4000}	7.8×10^{-2}
MH_{10}	0.5	4	20	8	5.7732×10^{-741}	2×10^{-4000}	3.1×10^{-2}
	1.8	4	20	8.0001×10^{0}	1.3446×10^{-210}	5.0780×10^{-1683}	3.1×10^{-2}
NM_9	0.5	4	20	9	1.7171×10^{-1088}	2×10^{-4000}	4.7×10^{-2}
	1.8	4	20	9	5.9947×10^{-330}	8.3091×10^{-2968}	4.7×10^{-2}
WO_8	0.5	4	16	8	3.2491×10^{-712}	2×10^{-4000}	6.2×10^{-2}
	1.8	4	16	8.0001×10^{0}	8.3705×10^{-212}	3.1312×10^{-1692}	6.3×10^{-2}
HM_6	0.5	4	20	6	1.1747×10^{-300}	2.1830×10^{-1802}	3.1×10^{-2}
	1.8	5	25	6	6.2470×10^{-444}	4.9378×10^{-2662}	4.7×10^{-2}
MHM_5	0.5	5	20	5	2.9184×10^{-855}	2×10^{-4000}	1.6×10^{-2}
	1.8	5	20	5	3.7051×10^{-207}	1.9490×10^{-1034}	1.6×10^{-2}
NR_2	0.5	9	18	2	1.3×10^{-371}	6.1228×10^{-743}	1.5×10^{-2}
	1.8	10	20	2×10^{0}	1.1395×10^{-201}	4.7044×10^{-403}	3.2×10^{-2}

Numerical results can be found in Table 10, where it is shown that PM_{10} has outperformed all other methods in terms of the absolute errors under consideration under two different initial conditions.

Table 10. Numerical results for problem 5.

Method	IG	N	FV	ACOC	$\|\epsilon\|$	$\|f(x_N)\|$	CPU
PM_{10}	0.71	5	30	10	2.5434×10^{-1635}	4×10^{-3999}	1.25×10^{-1}
	0.76	4	24	10	5.0617×10^{-1424}	1×10^{-3998}	9.4×10^{-2}
MH_{10}	0.71	5	25	8	5.1147×10^{-640}	4×10^{-3999}	6.2×10^{-2}
	0.76	4	20	8	6.3350×10^{-690}	4×10^{-3999}	4.7×10^{-2}
NM_9	0.71	5	25	9	3.8480×10^{-559}	1×10^{-3998}	7.8×10^{-2}
	0.76	4	20	9	4.0774×10^{-981}	4×10^{-3999}	4.7×10^{-2}
WO_8	0.71	5	20	8	3.3478×10^{-584}	1.3×10^{-3998}	1.56×10^{-1}
	0.76	4	16	8	1.3297×10^{-696}	2×10^{-3999}	6.3×10^{-2}
HM_6	0.71	6	30	6	5.4957×10^{-618}	4.4077×10^{-3696}	6.3×10^{-2}
	0.76	4	20	6	9.3504×10^{-288}	1.0692×10^{-1714}	4.7×10^{-2}
MHM_5	0.71	6	24	5	2.0592×10^{-216}	2.9538×10^{-1072}	4.7×10^{-2}
	0.76	5	20	5	4.2638×10^{-834}	4×10^{-3999}	3.1×10^{-2}
NR_2	0.71	12	24	2	5.5571×10^{-216}	3.9060×10^{-428}	4.7×10^{-2}
	0.76	9	18	2	5.3583×10^{-356}	3.6316×10^{-708}	3.2×10^{-2}

Example 6. *The Two-Dimensional Bratu Model [40].*

The two-dimension Bratu system is given by the following partial differential equation:

$$\frac{\partial^2 U}{\partial x^2} + \frac{\partial^2 U}{\partial y^2} + \lambda \exp(U) = 0, \ x, y \in D = [0,1] \times [0,1], \tag{49}$$

subject to the following boundary conditions

$$U(x,y) = 0 \ x, y \in D. \tag{50}$$

The two-dimensional Bratu system has two bifurcated exact solutions for $\lambda < \lambda_c$, a unique solution for $\lambda = \lambda_c$, and no solutions for $\lambda > \lambda_c$. The exact solution to (49) is determined as follows:

$$U(x,y) = 2\ln\left[\frac{cosh(\frac{\theta}{4})\cosh((x-\frac{1}{2})(y-\frac{1}{2})\theta)}{\cosh((x-\frac{1}{2})\frac{\theta}{2})\cosh((y-\frac{1}{2})\frac{\theta}{2})}\right], \tag{51}$$

where θ is an undetermined constant satisfying the boundary conditions and assumed to be the solution of (49). Using the procedure described in [41], one obtains

$$\theta^2 = \lambda\cosh^2\left(\frac{\theta}{4}\right). \tag{52}$$

Differentiating (52) with respect to θ and setting $\frac{d\lambda}{d\theta} = 0$, the critical value λ_c satisfies

$$\theta = \frac{1}{4}\lambda_c\cosh\left(\frac{\theta}{4}\right)\sinh\left(\frac{\theta}{4}\right). \tag{53}$$

By eliminating λ from (52) and (53), we have the value of θ_c for the critical λ_c satisfying

$$\frac{\theta_c}{4} = \coth\left(\frac{\theta_c}{4}\right), \tag{54}$$

and $\theta_c = 4.798714561$. We then obtain $\lambda_c = 7.027661438$ from (53). Numerical simulations performed in Table 11 show that the proposed three-step method takes a smaller number of iterations and produces considerably smaller absolute errors with a reasonable amount of CPU time.

Table 11. Numerical results for problem 6.

Method	IG	N	FV	ACOC	$\|\epsilon\|$	$\|f(x_N)\|$	CPU
PM_{10}	4.0	4	24	10	3.9×10^{-1086}	0	1.41×10^{-1}
	15.5	4	24	10	1.9074×10^{-553}	0	2.19×10^{-1}
MH_{10}	4.0	4	20	8	5.2357×10^{-556}	0	1.09×10^{-1}
	15.5	4	20	8	3.6499×10^{-277}	6.3171×10^{-2220}	1.09×10^{-1}
NM_9	4.0	4	20	9	4.3486×10^{-847}	0	6.3×10^{-2}
	15.5	4	20	9	2.3344×10^{-332}	1.9566×10^{-2994}	6.3×10^{-2}
WO_8	4.0	4	16	7.9999×10^{0}	1.4375×10^{-487}	1.6570×10^{-3902}	1.09×10^{-1}
	15.5	5	20	8	5.7605×10^{-1311}	0	9.4×10^{-2}
HM_6	4.0	4	20	6	1.4310×10^{-221}	3.9699×10^{-1331}	1.25×10^{-1}
	15.5	5	25	6	2.0883×10^{-638}	3.8348×10^{-3832}	1.25×10^{-1}
MHM_5	4.0	5	20	5	8.6954×10^{-585}	4.7414×10^{-2925}	9.4×10^{-2}
	15.5	5	20	5	5.0264×10^{-265}	3.06×10^{-1326}	9.4×10^{-2}
NR_2	4.0	9	18	2	5.7111×10^{-278}	1.0742×10^{-556}	7.8×10^{-2}
	15.5	10	20	2	1.2645×10^{-281}	5.2661×10^{-564}	1.09×10^{-1}

Now, we consider five different kinds of nonlinear multidimensional equations and numerically solve them with PM_{10}, HM_6, MH_5, and NR_2 since the methods MH_{10}, NM_9, and WO_8 were either divergent or not applicable on systems of nonlinear equations. For the systems considered, various types of initial guesses are used, and for comparison purposes, the approximate solution and the normed error $\boldsymbol{\epsilon} = ||\boldsymbol{x_{n+1}} - \boldsymbol{x_n}||_\infty$ having the same tolerance 10^{-200} and CPU time are taken into consideration. It can be observed in Tables 12–16 that the smallest possible absolute error is achieved only with the proposed method—that is, PM_{10}—in a reasonably affordable time period.

Example 7. *The nonlinear system of two equations from [3,41] is given as:*

$$\begin{aligned} x_1 + exp(x_2) - cos(x_2) &= 0, \\ 3x_1 - x_2 - sin(x_1) &= 0. \end{aligned} \tag{55}$$

The exact solution of the system (55) is $\boldsymbol{x} = [0, 0]'$. *The numerical results for this system are shown in Table 12 under the proposed* PM_{10} *and other three methods.*

Table 12. Numerical results for problem 7.

N	$[x_{1,0}, x_{2,0}]^T$	$[x_1, x_2]^T$	$\lvert\epsilon\rvert$	CPU
PM_{10}	$-1.0, 1.0$	$5.0151 \times 10^{-4003}, 1.0030 \times 10^{-4002}$	7.7834×10^{-3318}	0
HM_6	–	$1.2648 \times 10^{-2287}, 2.5296 \times 10^{-2287}$	9.9330×10^{-382}	1.4×10^{-1}
MH_5	–	$4.6720 \times 10^{-968}, 9.8372 \times 10^{-968}$	5.2359×10^{-194}	1.5×10^{-2}
NR_2	–	$1.2203 \times 10^{-11}, 2.4406 \times 10^{-11}$	6.0505×10^{-06}	4.7×10^{-2}
PM_{10}	$-1.9, 1.8$	$5.0151 \times 10^{-4003}, 1.0030 \times 10^{-4002}$	7.0614×10^{-1865}	3.2×10^{-2}
HM_6	–	$5.4937 \times 10^{-1225}, 1.0987 \times 10^{-1224}$	1.2688×10^{-204}	6.2×10^{-2}
MH_5	–	$2.1088 \times 10^{-559}, 4.4402 \times 10^{-559}$	2.8177×10^{-112}	6.2×10^{-2}
NR_2	–	$6.9588 \times 10^{-07}, 1.3916 \times 10^{-06}$	1.4445×10^{-03}	4.6×10^{-2}
PM_{10}	$2.5, -2.3$	$1.9407 \times 10^{-4002}, 3.8815 \times 10^{-4002}$	4.3934×10^{-3428}	1.6×10^{-2}
HM_6	–	$1.2918 \times 10^{-4000}, 2.5836 \times 10^{-4000}$	1.8909×10^{-691}	6.3×10^{-2}
MH_5	–	$2.8427 \times 10^{-695}, 5.9858 \times 10^{-695}$	1.8873×10^{-139}	6.2×10^{-2}
NR_2	–	$8.2443 \times 10^{-15}, 1.6489 \times 10^{-14}$	1.5727×10^{-07}	4.7×10^{-2}
PM_{10}	8.9, 5.5	$3.9204 \times 10^{-1989}, 8.3019 \times 10^{-1989}$	2.1886×10^{-199}	7.8×10^{-2}
HM_6	–	$3.6432 \times 10^{-93}, 7.2864 \times 10^{-93}$	5.4995×10^{-16}	9.3×10^{-2}
MH_5	–	$5.6511 \times 10^{-70}, 1.1901 \times 10^{-69}$	2.1654×10^{-14}	1.41×10^{-1}
NR_2	–	$2.7716 \times 10^{-1}, 5.1674 \times 10^{-1}$	7.9930×10^{-1}	4.7×10^{-2}
PM_{10}	1.9, 6.5	$1.5819 \times 10^{-1006}, 3.3499 \times 10^{-1006}$	3.9880×10^{-101}	7.8×10^{-2}
HM_6	–	$5.9575 \times 10^{-44}, 1.1915 \times 10^{-43}$	8.7617×10^{-08}	9.4×10^{-2}
MH_5	–	$9.8552 \times 10^{-36}, 2.0743 \times 10^{-35}$	1.5268×10^{-07}	1.8700×10^{-1}
NR_2	–	$6.7641 \times 10^{-1}, 1.3228 \times 10^{0}$	1.0621×10^{0}	4.6×10^{-2}
PM_{10}	0.1, 0.1	$9.2473 \times 10^{-4002}, 1.8495 \times 10^{-4001}$	0	0
HM_6	–	$4.9611 \times 10^{-4002}, 9.9222 \times 10^{-4002}$	2.4509×10^{-1479}	7.8×10^{-2}
MH_5	–	$1.8099 \times 10^{-3702}, 3.8109 \times 10^{-3702}$	6.8648×10^{-741}	4.7×10^{-2}
NR_2	–	$2.8142 \times 10^{-39}, 5.6283 \times 10^{-39}$	9.1883×10^{-20}	4.7×10^{-2}

Example 8. *Another nonlinear system of three equations taken from [42] is shown below:*

$$\begin{aligned} 3x_1 - \cos(x_2 x_3) - 1/2 &= 0, \\ x_1^2 - 81(x_2 + 0.1)^2 + \sin(x_3) + 1.06 &= 0, \\ \exp(-x_1 x_2) + 20x_3 + (10\pi/3 - 1) &= 0, \end{aligned} \tag{56}$$

where its approximate solution up to 50 dp is shown below:

$$\boldsymbol{x} \approx \begin{bmatrix} 0.49814468458949119126228211413809456132099782481239 \\ 0 \\ -0.52882597757338745562224205210357569604547206124467 \end{bmatrix}. \tag{57}$$

The numerical results for the system (56) are shown in Table 13.

Table 13. Numerical results for problem 8.

Method	$[x_{1,0}, x_{2,0}, x_{3,0}]^T$	$[x_1, x_2, x_3]^T$	$\|\epsilon\|$	CPU
PM_{10}	1.1, 1.1, −1.1	5×10^{-1}, $-1.4199 \times 10^{-4001}$, -5.2360×10^{-1}	3.6961×10^{-541}	4.6×10^{-2}
HM_6	–	5×10^{-1}, 5.9801×10^{-445}, -5.2360×10^{-1}	2.1352×10^{-75}	4.7×10^{-2}
MH_5	–	5×10^{-1}, 3.9207×10^{-162}, -5.2360×10^{-1}	2.4120×10^{-40}	7.8×10^{-2}
NR_2	–	5.0001×10^{-1}, 9.5530×10^{-04}, -5.2357×10^{-1}	1.3842×10^{-2}	1.5×10^{-2}
PM_{10}	3.3, 2.1, −2.1	5×10^{-1}, $-4.5590 \times 10^{-2904}$, -5.2360×10^{-1}	2.9346×10^{-362}	6.3×10^{-2}
HM_6	–	5×10^{-1}, 2.1651×10^{-257}, -5.2360×10^{-1}	3.8835×10^{-44}	4.7×10^{-2}
MH_5	–	5×10^{-1}, -7.2199×10^{-131}, -5.2360×10^{-1}	5.4152×10^{-33}	9.3×10^{-2}
NR_2	–	5.0007×10^{-1}, 7.6756×10^{-03}, -5.2340×10^{-1}	3.9888×10^{-2}	3.1×10^{-2}
PM_{10}	−1.3, 1.1, −0.1	5×10^{-1}, $-8.2172 \times 10^{-4001}$, -5.2360×10^{-1}	1.1159×10^{-551}	6.3×10^{-2}
HM_6	–	5×10^{-1}, 1.2373×10^{-456}, -5.2360×10^{-1}	2.4102×10^{-77}	4.6×10^{-2}
MH_5	–	5×10^{-1}, 3.9887×10^{-168}, -5.2360×10^{-1}	7.6082×10^{-42}	4.7×10^{-2}
NR_2	–	5.0001×10^{-1}, 8.3811×10^{-04}, -5.2358×10^{-1}	1.2961×10^{-2}	1.6×10^{-2}
PM_{10}	1.9, 4.1, 0.1	5×10^{-1}, $-4.6794 \times 10^{-1318}$, -5.2360×10^{-1}	4.8484×10^{-164}	6.3×10^{-2}
HM_6	–	5×10^{-1}, 3.4949×10^{-117}, -5.2360×10^{-1}	9.0619×10^{-21}	4.7×10^{-2}
MH_5	–	5×10^{-1}, $-5.3226e-56$, -5.2360×10^{-1}	1.0093×10^{-12}	4.7×10^{-2}
NR_2	–	5.0048×10^{-1}, 5.5684×10^{-2}, -5.2215×10^{-1}	1.1923×10^{-1}	3.1×10^{-2}
PM_{10}	6.5, 2.2, −3.3	5×10^{-1}, $-7.0328e-2659$, -5.2360×10^{-1}	1.3210×10^{-331}	6.3×10^{-2}
HM_6	–	5×10^{-1}, 6.7653×10^{-236}, -5.2360×10^{-1}	1.4849×10^{-40}	4.7×10^{-2}
MH_5	–	5×10^{-1}, -3.4190×10^{-98}, -5.2360×10^{-1}	3.6270×10^{-24}	1.1×10^{-1}
NR_2	–	5.0009×10^{-1}, 1.0418×10^{-2}, -5.2333×10^{-1}	4.6778×10^{-2}	3.2×10^{-2}
PM_{10}	3.5, 3.7, −2.3	5×10^{-1}, $-9.2310 \times 10^{-1487}$, -5.2360×10^{-1}	3.8829×10^{-185}	6.3×10^{-2}
HM_6	–	5×10^{-1}, 2.4238×10^{-131}, -5.2360×10^{-1}	3.9573×10^{-23}	6.3×10^{-2}
MH_5	–	5×10^{-1}, -1.2128×10^{-61}, -5.2360×10^{-1}	3.6152×10^{-14}	1.41×10^{-1}
NR_2	–	5.0039×10^{-1}, 4.4303×10^{-2}, -5.2244×10^{-1}	1.0395×10^{-1}	3.1×10^{-2}

Example 9. *A three-dimensional nonlinear system is taken from [3] as given below:*

$$\begin{aligned} x_1^2 + x_2^2 + x_3^2 - 1 &= 0, \\ 2x_1^2 + x_2^2 - 4x_3 &= 0, \\ 3x_1^2 - 4x_2^2 + x_3^2 &= 0, \end{aligned} \tag{58}$$

where its approximate solution up to 50 dp is as follows:

$$\boldsymbol{x} \approx \begin{bmatrix} 0.69828860997151390091867421225192307770469334334732 \\ 0.62852429796021380638277617781675123954652671431496 \\ 0.34256418968956943776230136116401106884202074401616 \end{bmatrix}. \tag{59}$$

The numerical results for the system (58) are shown in Table 14.

Table 14. Numerical results for problem 9.

Method	$[x_{1,0}, x_{2,0}, x_{3,0}]^T$	$[x_1, x_2, x_3]^T$	$\|\epsilon\|$	CPU
PM_{10}	0.5, 0.5, 0.5	$6.9829 \times 10^{-1}, 6.2852 \times 10^{-1}, 3.4256 \times 10^{-1}$	1×10^{-4000}	0
HM_6	–	$6.9829 \times 10^{-1}, 6.2852 \times 10^{-1}, 3.4256 \times 10^{-1}$	3.8739×10^{-864}	0
MH_5	–	$6.9829 \times 10^{-1}, 6.2852 \times 10^{-1}, 3.4256 \times 10^{-1}$	6.0191×10^{-527}	1.5×10^{-2}
NR_2	–	$6.9829 \times 10^{-1}, 6.2852 \times 10^{-1}, 3.4256 \times 10^{-1}$	3.8598×10^{-12}	1.5×10^{-2}
PM_{10}	1.0, 1.0, 1.0	$6.9829 \times 10^{-1}, 6.2852 \times 10^{-1}, 3.4256 \times 10^{-1}$	1×10^{-4000}	0
HM_6	–	$6.9829 \times 10^{-1}, 6.2852 \times 10^{-1}, 3.4256 \times 10^{-1},$	4.1436×10^{-596}	1.6×10^{-2}
MH_5	–	$6.9829 \times 10^{-1}, 6.2852 \times 10^{-1}, 3.4256 \times 10^{-1},$	3.5513×10^{-269}	0
NR_2	–	$6.9829 \times 10^{-1}, 6.2852 \times 10^{-1}, 3.4256 \times 10^{-1},$	1.1136×10^{-08}	0
PM_{10}	2.8, 3.2, 6.1	$6.9829 \times 10^{-1}, 6.2852 \times 10^{-1}, 3.4256 \times 10^{-1}$	7.0153×10^{-894}	0
HM_6	–	$6.9829 \times 10^{-1}, 6.2852 \times 10^{-1}, 3.4256 \times 10^{-1}$	1.4280×10^{-103}	0
MH_5	–	$6.9829 \times 10^{-1}, 6.2852 \times 10^{-1}, 3.4256 \times 10^{-1}$	6.9270×10^{-53}	0
NR_2	–	$6.9929 \times 10^{-1}, 6.2876 \times 10^{-1}, 3.4257 \times 10^{-1}$	3.7312×10^{-2}	0
PM_{10}	5.1, 4.2, 1.1	$6.9829 \times 10^{-1}, 6.2852 \times 10^{-1}, 3.4256 \times 10^{-1}$	4.8091×10^{-1119}	0
HM_6	–	$6.9829 \times 10^{-1}, 6.2852 \times 10^{-1}, 3.4256 \times 10^{-1}$	8.9729×10^{-126}	0
MH_5	–	$6.9829 \times 10^{-1}, 6.2852 \times 10^{-1}, 3.4256 \times 10^{-1}$	6.3425×10^{-68}	0
NR_2	–	$6.9851 \times 10^{-1}, 6.2861 \times 10^{-1}, 3.4256 \times 10^{-1}$	1.7731×10^{-2}	0
PM_{10}	5.1, 4.2, −1.1	$6.9829 \times 10^{-1}, 6.2852 \times 10^{-1}, 3.4256 \times 10^{-1}$	1.1175×10^{-174}	0
HM_6	–	$6.9829 \times 10^{-1}, 6.2852 \times 10^{-1}, 3.4256 \times 10^{-1}$	2.7993×10^{-19}	0
MH_5	–	$4.3060 \times 10^{0}, -9.1902 \times 10^{-1}, -2.7461 \times 10^{0}$	2.4533×10^{0}	0
NR_2	–	$1.0879 \times 10^{0}, 8.0413 \times 10^{-1}, 5.3209 \times 10^{-1}$	7.7974×10^{-1}	0
PM_{10}	10.2, 14.7, 11.1	$6.9829 \times 10^{-1}, 6.2852 \times 10^{-1}, 3.4256 \times 10^{-1}$	2.9425×10^{-319}	0
HM_6	–	$6.9829 \times 10^{-1}, 6.2852 \times 10^{-1}, 3.4256 \times 10^{-1}$	2.8736×10^{-34}	0
MH_5	–	$6.9829 \times 10^{-1}, 6.2852 \times 10^{-1}, 3.4256 \times 10^{-1}$	1.1693×10^{-19}	1.6×10^{-2}
NR_2	–	$7.5545 \times 10^{-1}, 7.3493 \times 10^{-1}, 3.4367 \times 10^{-1}$	3.7993×10^{-1}	0

Example 10. *(Catenary curve and the ellipse ([43], p. 83)):*

Given below is a nonlinear system of two equations that describe trajectories for the catenary and the ellipse, respectively. We are interested in finding their intersection point that lies in the first quadrant of the cartesian plane. The system has been solved under the proposed PM_{10} method and other methods under consideration. The performance of each method is shown in Table 15, whereas an approximate solution up to 50 dp of the system (60) is shown in comparison to the system.

$$\begin{aligned} &x_2 - \frac{1}{2}\Big(\exp(x_1/2) + \exp(-x_1/2)\Big) = 0, \\ &9x_1^2 + 25x_2^2 - 225 = 0. \end{aligned} \tag{60}$$

Approximate solution:

$$\boldsymbol{x} \approx \begin{bmatrix} 3.0311553917189839536524964478460650851937092065081 \\ 2.3858656535628857281228809627652263081419323345176 \end{bmatrix}. \tag{61}$$

Table 15. Numerical results for problem 10 with * showing the divergence of the method.

Method	$[x_{1,0}, x_{2,0}]^T$	$[x_1, x_2]^T$	$\|\epsilon\|$	CPU
PM_{10}	9.3, 8.6	$3.0312 \times 10^0, 2.3859 \times 10^0$	8.4435×10^{-523}	3.1×10^{-2}
HM_6	–	–	3.1353×10^{-78}	4.6×10^{-2}
MH_5	–	–	1.4228×10^{-40}	3.1×10^{-2}
NR_2	–	–	2.1386×10^{-1}	1.6×10^{-2}
PM_{10}	11.6, 13.1	$3.0312 \times 10^0, 2.3859 \times 10^0$	1.4738×10^{-275}	4.7×10^{-2}
HM_6	–	–	8.2367×10^{-31}	1.6×10^{-2}
MH_5	–	–	1.8199×10^{-17}	3.1×10^{-2}
NR_2	–	–	1.0020×10^0	1.6×10^{-2}
PM_{10}	4.6, 3.6	$3.0312 \times 10^0, 2.3859 \times 10^0$	1.7497×10^{-2436}	3.1×10^{-2}
HM_6	–	–	2.4449×10^{-528}	3.1×10^{-2}
MH_5	–	–	1.3516×10^{-167}	1.6×10^{-2}
NR_2	–	–	2.0398×10^{-07}	1.6×10^{-2}
PM_{10}	16.6, 14.5	$3.0312 \times 10^0, 2.3859 \times 10^0$	8.0467×10^{-58}	3.1×10^{-2}
HM_6	–	–	3.8892×10^{-04}	3.2×10^{-2}
MH_5	–	–	1.1246×10^{-05}	3.1×10^{-2}
NR_2	–	$6.6073 \times 10^0, -1.1486 \times 10^0$	2.5508×10^0 *	1.6×10^{-2}
PM_{10}	2.9, 1.9	$3.0312 \times 10^0, 2.3859 \times 10^0$	0	3.1×10^{-2}
HM_6	–	–	1.1004×10^{-1156}	3.1×10^{-2}
MH_5	–	–	5.7581×10^{-307}	3.1×10^{-2}
NR_2	–	–	3.1421×10^{-15}	0
PM_{10}	10.3, 11.7	$3.0312 \times 10^0, 2.3859 \times 10^0$	5.3262×10^{-397}	9.4×10^{-2}
HM_6	–	–	7.9396×10^{-54}	9.3×10^{-2}
MH_5	–	–	1.6306×10^{-28}	6.2×10^{-2}
NR_2	–	–	5.0395×10^{-1}	3.1×10^{-2}

Example 11. *Steady-State Lorenz Equations ([44], p. 816).*

In this problem, we consider a system developed by Edward Lorenz, who was an American meteorologist studying atmospheric convection around the Earth's surface. Lorenz's nonlinear system is a set of three ordinary differential equations, as given below:

$$\begin{aligned} \dot{x_1}(t) &= a(x_2 - x_1), \\ \dot{x_2}(t) &= x_1(b - x_3) - x_2, \\ \dot{x_3}(t) &= x_1 x_2 - c x_3. \end{aligned} \tag{62}$$

In order to study the steady-state behavior of the system (62), we take $\dot{x_1}(t) = \dot{x_2}(t) = \dot{x_3}(t) = 0$ and $a = -1, b = 2, c = 3$ to obtain the following nonlinear algebraic system:

$$\begin{aligned} x_1 - x_2 &= 0, \\ 2x_1 - x_1 x_3 - x_2 &= 0, \\ x_1 x_2 - 3x_3 &= 0. \end{aligned} \tag{63}$$

The approximate solution for the system (63) correct to 50 dp is given as

$$x \approx \begin{bmatrix} 1.7320508075688772935274463415058723669428052538104 \\ 1.7320508075688772935274463415058723669428052538104 \\ 1 \end{bmatrix}. \tag{64}$$

The nonlinear steady-state system (63) has been numerically solved in Table 16.

Table 16. Numerical results for problem 11 with * showing the divergence of the method.

Method	$[x_{1,0}, x_{2,0}, x_{3,0}]^T$	$[x_1, x_2, x_3]^T$	$\lvert\epsilon\rvert$	CPU
PM_{10}	$-2.5, -3.5, -1.5$	1.7321, 1.7321, 1	3.1099×10^{-36}	1.5×10^{-2}
HM_6	–	1.7321, 1.7321, 1	3.8304×10^{-2}	1.6×10^{-2}
MH_5	–	$-1.7321, -1.7321, 1$	8.8956×10^{-08} *	0
NR_2	–	–	1.1129×10^{-1} *	0
PM_{10}	$-1.0, -1.0, 2.0$	$-1.7321, -1.7321, 1$	2.2477×10^{-1164}	0
HM_6	–	–	2.5855×10^{-138}	1.6×10^{-2}
MH_5	–	–	2.6805×10^{-77}	0
NR_2	–	–	2.8563×10^{-04}	0
PM_{10}	$-3.9, -3.3, -6.2$	1.7321, 1.7321, 1	5.2666×10^{-84}	1.6×10^{-2}
HM_6	–	–	5.8363×10^{-07}	0
MH_5	–	–	1.0692×10^{-05}	1.5×10^{-2}
NR_2	–	1.9410, 1.9410, 1.1534	5.5426×10^{-1} *	0
PM_{10}	1, 1, 2	1.7321, 1.7321, 1	$2.247\,7 \times 10^{-1164}$	1.6×10^{-2}
HM_6	–	–	2.5855×10^{-138}	0
MH_5	–	–	2.6805×10^{-77}	1.6×10^{-2}
NR_2	–	–	2.8563×10^{-04}	0
PM_{10}	5.9, 3.3, 6.2	1.7321, 1.7321, 1	5.9770×10^{-811}	1.6×10^{-2}
HM_6	–	–	4.0516×10^{-136}	1.5×10^{-2}
MH_5	–	–	5.5239×10^{-62}	0
NR_2	–	–	1.4635×10^{-2}	0
PM_{10}	2.4, 3.0, 1.0	1.7321, 1.7321, 1	0	0
HM_6	–	–	1.8162×10^{-811}	0
MH_5	–	–	5.2175×10^{-433}	1.6×10^{-2}
NR_2	–	–	4.2917×10^{-11}	0

6. Concluding Remarks

This research study is based on devising a new, highly effective three-step iterative method with tenth-order convergence. The convergence is proved theoretically via Taylor's series expansion for single and multi-variable nonlinear equations, and the approximate computational order of convergence confirms such findings. Thus, the proposed method PM_{10} is applicable not only to single nonlinear equations but also to nonlinear systems. Moreover, dynamical aspects of PM_{10} are also explored with basins of attraction that show quite esthetic phase plane diagrams when applied to complex-valued functions, thereby proving the stability of the method when initial guesses are taken within the vicinity of the underlying nonlinear model. Finally, different types of nonlinear equations and systems, including those used in physical and natural sciences, are chosen to be tested with PM_{10} and with various well-known optimal and non-optimal methods in the sense of King–Traub. In most of the cases, PM_{10} is found to have better results, particularly when it comes to the number of iterations N to achieve required accuracy, ACOC, absolute error, and absolute functional value. It is also worthwhile to note that the proposed method always converges, irrespective of whether the initial guess passed to it lies near to or away from the approximate solution. Hence, PM_{10} is a competitive iterative method with tenth-order convergence for solving nonlinear equations and systems.

We understand that methods of very high order are only of academic interest since approximations to the solutions of very high accuracy are not needed in practice. On the other hand, such methods are, to some extent, complicated and do not offer much of an increase in computational efficiency. Moreover, the method proposed in this article lies in the family of methods without memory, requiring the evaluation of three Jacobian matrices, and thereby becomes computationally expensive. To avoid computational complexity, we will propose, in future studies, a modification of the proposed method by replacing

the first-order derivative with a suitable finite-difference approximation. In addition, the proposed method will also be analyzed for its semi-local convergence.

Author Contributions: Conceptualization, A.T., S.Q. and A.S.; methodology, S.Q. and D.B.; validation, A.S. and E.H.; formal analysis, S.Q., A.T. and A.A.S.; investigation, S.Q., D.B. and E.H.; data curation, S.Q. and A.A.S.; writing—original draft preparation, A.T., S.Q., A.S. and A.A.S.; writing—review and editing, A.T., S.Q., A.S., E.H. and D.B. All authors have read and agreed to the published version of the manuscript.

Funding: This research did not receive any specific external funding.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Not applicable.

Acknowledgments: The authors extend their appreciation to the deputyship for Research & Innovation, Ministry of Education in Saudi Arabia for funding this research work through the project number (IFP-2020-64).

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Ortega, J.M. *Numerical Analysis: A Second Course*; Society for Industrial and Applied Mathematics: Philadelphia, PA, USA, 1990.
2. Ham, Y.; Chun, C. A fifth-order iterative method for solving nonlinear equations. *Appl. Math. Comput.* **2007**, *194*, 287–290. [CrossRef]
3. Abro, H.A.; Shaikh, M.M. A new time-efficient and convergent nonlinear solver. *Appl. Math. Comput.* **2019**, *355*, 516–536. [CrossRef]
4. Cordero, A.; Torregrosa, J.R.; Vassileva, M.P. Design, Analysis, and Applications of Iterative Methods for Solving Nonlinear Systems. In *Nonlinear Systems-Design, Analysis, Estimation and Control*; Lee, D., Burg, T., Volos, C., Eds.; IntechOpen: Rijeka, Croatia, 2016. [CrossRef]
5. Hafiz, M.A.; Bahgat, M.S. An efficient two-step iterative method for solving system of nonlinear equations. *J. Math. Res.* **2012**, *4*, 28.
6. Noor, M.A.; Khan, W.A.; Hussain, A. A new modified Halley method without second derivatives for nonlinear equation. *Appl. Math. Comput.* **2007**, *189*, 1268–1273. [CrossRef]
7. Sharifi, M.; Babajee, D.K.R.; Soleymani, F. Finding the solution of nonlinear equations by a class of optimal methods. *Comput. Math. Appl.* **2012**, *63*, 764–774. [CrossRef]
8. Noor, M.A. Some iterative methods for solving nonlinear equations using homotopy perturbation method. *Int. J. Comput. Math.* **2010**, *87*, 141–149. [CrossRef]
9. Kung, H.T.; Traub, J.F. Optimal order of one-point and multipoint iteration. *J. ACM (JACM)* **1974**, *21*, 643–651. [CrossRef]
10. Householder, A.S. *The Numerical Treatment of a Single Nonlinear Equation*; McGraw-Hill: New York, NY, USA, 1970.
11. Bahgat, M.S.; Hafiz, M.A. Three-step iterative method with eighteenth order convergence for solving nonlinear equations. *Int. J. Pure Appl. Math.* **2014**, *93*, 85–94. [CrossRef]
12. Chun, C. Some fourth-order iterative methods for solving nonlinear equations. *Appl. Math. Comput.* **2008**, *195*, 454–459. [CrossRef]
13. Sharma, R.; Bahl, A. An optimal fourth order iterative method for solving nonlinear equations and its dynamics. *J. Complex Anal.* **2015**, *2015*, 259167. [CrossRef]
14. Geum, Y.H.; Kim, Y.I.; Neta, B. Constructing a family of optimal eighth-order modified Newton-type multiple-zero finders along with the dynamics behind their purely imaginary extraneous fixed points. *J. Comput. Appl. Math.* **2018**, *333*, 131–156. [CrossRef]
15. Qureshi, S.; Ramos, H.; Soomro, A.K. A New Nonlinear Ninth-Order Root-Finding Method with Error Analysis and Basins of Attraction. *Mathematics* **2021**, *9*, 1996. [CrossRef]
16. Ramos, H.; Monteiro, M.T.T. A new approach based on the Newton's method to solve systems of nonlinear equations. *J. Comput. Appl. Math.* **2017**, *318*, 3–13. [CrossRef]
17. Ramos, H.; Vigo-Aguiar, J. The application of Newton's method in vector form for solving nonlinear scalar equations where the classical Newton method fails. *J. Comput. Appl. Math.* **2015**, *275*, 228–237. [CrossRef]
18. Darvishi, M.T.; Barati, A. A third-order Newton-type method to solve systems of nonlinear equations. *Appl. Math. Comput.* **2007**, *187*, 630–635. [CrossRef]
19. Wang, X.; Liu, L. New eighth-order iterative methods for solving nonlinear equations. *J. Comput. Appl. Math.* **2010**, *234*, 1611–1620. [CrossRef]
20. Hu, Z.; Guocai, L.; Tian, L. An iterative method with ninth-order convergence for solving nonlinear equations. *Int. J. Contemp. Math. Sci.* **2011**, *6*, 17–23.

21. Hafiz, M.A.; Al-Goria, S.M. Solving nonlinear equations using a new tenth-and seventh-order methods free from second derivative. *Int. J. Differ. Equ. Appl.* **2013**, *12(4)*.
22. Cordero, A.; Hueso, J.L.; Martínez, E.; Torregrosa, J.R. New modifications of Potra–Pták's method with optimal fourth and eighth orders of convergence. *J. Comput. Appl. Math.* **2010**, *234*, 2969–2976. [CrossRef]
23. Lotfi, T.; Bakhtiari, P.; Cordero, A.; Mahdiani, K.; Torregrosa, J.R. Some new efficient multipoint iterative methods for solving nonlinear systems of equations. *Int. J. Comput. Math.* **2015**, 92, 1921–1934. [CrossRef]
24. Cordero, A.; Hueso, J.L.; Martínez, E.; Torregrosa, J.R. A modified Newton-Jarratt's composition. *Numer. Algorithms* **2010**, *55*, 87–99. [CrossRef]
25. Waseem, M.; Noor, M.A.; Noor, K.I. Efficient method for solving a system of nonlinear equations. *Appl. Math. Comput.* **2016**, *275*, 134–146. [CrossRef]
26. Noor, M.A.; Noor, K.I.; Al-Said, E.; Waseem, M. Some new iterative methods for nonlinear equations. *Math. Probl. Eng.* **2010**, *2010*, 198943. [CrossRef]
27. Hueso, J.L.; Martínez, E.; Torregrosa, J.R. Third and fourth order iterative methods free from second derivative for nonlinear systems. *Appl. Math. Comput.* **2009**, *211*, 190–197. [CrossRef]
28. Weerakoon, S.; Fernando, T.G.I. A variant of Newton's method with accelerated third-order convergence. *Appl. Math. Lett.* **2000**, *13*, 87–93. [CrossRef]
29. Grau-Sánchez, M.; Gutiérrez, J.M. Zero-finder methods derived from Obreshkov's techniques. *Appl. Math. Comput.* **2009**, *215*, 2992–3001. [CrossRef]
30. Petkovi´c, M.S. Remarks on "On a general class of multipoint root-finding methods of high computational efficiency". *SIAM J. Numer. Anal.* **2011**, *49*, 1317–1319. [CrossRef]
31. Scott, M.; Neta, B.; Chun, C. Basin attractors for various methods. *Appl. Math. Comput.* **2011**, *218*, 2584–2599. [CrossRef]
32. Stewart, B.D. *Attractor Basins of Various Root-Finding Methods*; Naval Postgraduate School: Monterey CA, USA, 2001.
33. Halley, E. A new, exact, and easy method of finding the roots of any equations generally, and that without any previous reduction. *Philos. Trans. R. Soc. Lond.* **1694**, *18*, 136–145.
34. Chen, H.; Zheng, X. Improved Newton Iterative Algorithm for Fractal Art Graphic Design. *Complexity* **2020**, *2020*, 6623049. [CrossRef]
35. Susanto, H.; Karjanto, N. Newton's method's basins of attraction revisited. *Appl. Math. Comput.* **2009**, *215*, 1084–1090. [CrossRef]
36. Tao, Y.; Madhu, K. Optimal Fourth, Eighth and Sixteenth Order Methods by Using Divided Difference Techniques and Their Basins of Attraction and Its Application. *Mathematics* **2019**, *7*, 322. [CrossRef]
37. Said Solaiman, O.; Hashim, I. Efficacy of optimal methods for nonlinear equations with chemical engineering applications. *Math. Probl. Eng.* **2019**, *2019*, 1728965. [CrossRef]
38. Khoury, R.; Harder, D.W. *Numerical Methods and Modelling for Engineering*; Springer: Berlin/Heidelberg, Germany, 2016; pp. 120–124.
39. Shacham, M. Numerical solution of constrained non-linear algebraic equations. *Int. J. Numer. Methods Eng.* **1986**, *23*, 1455–1481. [CrossRef]
40. Madhu, K.; Babajee, D.K.R.; Jayaraman, J. An improvement to double-step Newton method and its multi-step version for solving system of nonlinear equations and its applications. *Numer. Algorithms* **2017**, *74*, 593–607. [CrossRef]
41. Madhu, K.; Elango, A.; Landry, R., Jr.; Al-arydah, M.T. New multi-step iterative methods for solving systems of nonlinear equations and their application on GNSS pseudorange equations. *Sensors* **2020**, *20*, 5976. [CrossRef] [PubMed]
42. Burden, R.L.; Faires, J.D. *Numerical Analysis*; Brooks/Cole: Boston, MA, USA, 2010; Volume 7.
43. Amos, G.; Subramaniam, V. *Numerical Methods for Engineers and Scientists: An Introduction with Applications Using MATLAB*; Department of Mechanical Engineering, The Ohio State University: Columbus, OH, USA, 2014.
44. Chapra, S.C.; Canale, R.P. *Numerical Methods for Engineers*; McGraw-Hill Higher Education: Boston, MA, USA, 2010.