JOURNAL OF MECHANICS OF CONTINUA AND MATHEMATICAL SCIENCES
www.journalimcms.org

ISSN (Online) : 2454 -7190 , Special Issue, No.-12, August (2025) pp 150-162 ISSN (Print) 0973-8975

AN IMPROVED ITERATIVE SCHEME FOR REAL AND COMPLEX ROOTS DETECTION

Reetu Malhotra[1], Anujeet Siwach[2]

[1,2] Chitkara University Institute of Engineering and Technology, Chitkara University, Punjab, 140417, India

Email: [1]anujeet.siwach@gmail.com [2]reetu.malhotra@chitkarauniversity.edu.in

Corresponding Author: **Anujeet Siwach**

https://doi.org/10.26782/jmcms.spl.12/2025.08.00012

(Received: May 17, 2025; Revised: July 28, 2025; Accepted: August 11, 2025)

Abstract

In this study, researchers propose an innovative numerical approach to solve non-linear equations for real as well as complex roots. The approach, initiated with an initial guess in the complex plane, iteratively converges towards solutions. A notable feature is its ability to accurately identify complex roots even when initialized with a real number. The method demonstrates second-order convergence, with its efficacy evaluated through quantifying the number of iterations needed for convergence. Using Python 3.10.9, experiments were conducted to evaluate its effectiveness across various numerical problems. Results were presented in tabular format, supplemented by graphical representations. Furthermore, the study examines the method's computational efficiency by analyzing CPU time and introducing an efficiency index.

Keywords: High-order transcendental equations, Nonlinear, Complex root-finding algorithms, Convergence, Innovation.

I. Introduction

In both science and engineering, addressing complex challenges often involves solving nonlinear equations related to scalar functions [I - XXVIII]. Iterative techniques, such as those introduced by Newton-Raphson, Halley, and Cauchy, have become the preferred methods due to their reliability and efficiency. Over time, the development of numerical algorithms based on these iterative methods has gained significant traction in modern research. However, many of these algorithms demand considerable computational resources. To address this, numerous researchers have proposed enhancements aimed at improving computational efficiency.

Ostrowski [XVII] proposed two important methods of solving nonlinear equations based on the computation of two functions and one derivative at each iteration that converge to third- and fourth-order. Traub [XXVIII] further advanced the field by introducing an approach that requires evaluating the function alongside two first-order derivatives at each iteration, resulting in third-order convergence. Building on these advancements, Sharma and Guha [XXII] have postulated a six-order convergence three-step method for solving nonlinear equations, represented as g(x) = 0, using a single parameter denoted as ‘a’.

$$p_{n+1} = p_n - \frac{h(p_n)}{h'(p_n)} \tag{1}$$

$$t_{n+1} = w_n - \frac{g(w_n)}{g'(t_n)} \frac{g(t_n)}{g(t_n)-2g(w_n)} \tag{2}$$

$$\widetilde{t_{n+1}} = t_{n+1} - \frac{g(t_{n+1})}{g'(t_n)} \frac{g(t_n)+ag(w_n)}{g(t_n)+(a-2)g(w_n)} \tag{3}$$

Moreover, Melman [XIV] utilized the Newton method involving two steps to find the greatest or smallest root of a polynomial possessing entirely absolute roots. Maheshwari [XI] also utilized a 4^{th}order technique to determine solutions for linear and non-linear equations. This approach requires fewer iterations and functional evaluations, leading to lower absolute error values.

$$w_n = t_n - \frac{h(t_n)}{h'(t_n)} \tag{4}$$

$$t_{n+1} = t_n + \frac{1}{h'(t_n)}\left[\frac{\{h(t_n)\}^2}{h(w_n)-h(t_n)} - \frac{\{h(w_n)\}^2}{h(t_n)}\right] \tag{5}$$

Popovski [IX] explored a three-step method that calculates three function values along with one derivative value per iteration, achieving a 7^{th}-order convergence. Moreover, Singh and Gupta [XXVII] proposed a 4^{th}-order technique designed to obtain a single root of nonlinear equations. Sharma and Bahl [XXV] proposed a 6th-order iterative technique aimed at identifying real roots of non-linear equations, starting with an initial approximation. The effectiveness of their technique was measured by the number of iterations and the evaluated functions.

Roman [XX] developed a new series of methods inspired by Newton-Chebyshev [to solve non-linear equations. Their research focused on quadratic polynomials to investigate fixed and critical points. The above study included an analysis of stable versus unstable behaviors and considerations of the parameterized space. Additionally, the Hansen-Patrick [VII] explored a method distinguished by three orders of convergence, as described by...

$$t_{n+1} = t_n - \left[\frac{\alpha+1}{\alpha \pm (1-(\alpha+1)H^{\frac{1}{2}}}\right]\frac{h(u_n)}{h'(u_n)} \tag{6}$$

where H= $\frac{h''(x_n)h(x_n)}{[h'(x_n)]^2}$, $\alpha \in$ R is a famous technique. The family includes different techniques like Euler's method, Laguerre's method, Newton-Raphson method, and Halley's method. In the above methods, except Newton's method, show three orders of convergence, and Newton's method has 3^{rd} order of convergence. But even though it has better convergence, the application of Newton's method is less because of the heavy cost of computation, requiring estimation based on 2^{nd}-order derivatives. Based on the Hansen-Patrick method, Sharma et al. [XXIII] developed an improved double-parameter scheme which shows a convergence order of three, except with one technique which has a 4th-order convergence. Additionally, Abbasbandy [I] employed a decomposed Adomian modified method to develop an algorithm to find a solution to a system of two non-linear variable equations motivated by Newton's method. Later, Parhi and Gupta [XVIII] created a technique that eliminates the use of 2nd-order derivatives and is used to determine real roots for a non-linear equation. The technique involves computation based on two functions and two 1st-order derivatives per iteration, with six orders of convergence and an efficiency index of 1.565. Importantly, it requires fewer iteration steps than Newton's method. Moreover, Noor and Waseem [XVI] used quadrature formulas to study two new 2-step iterative techniques to solve a sequence of non-linear equations, both of which demonstrate cubic order of convergence. The efficiency index of the technique is $3^{1/(n+4n^2)}$ for $n \geq 2$. Besides the above methods, Sharma and Sharma [XXIV] derived a method to determine the roots of non-linear equations with various collections. The technique requires analysis of three functions and has a 4th-order convergence order.

$$w_n = t_n - \frac{2m}{m+2}\frac{f(t_n)}{f'(t_n)} \tag{7}$$

$$t_{n+1} = w_n - \frac{\frac{1}{2}m(m-2)(\frac{m}{m+2})^{-m}f'(w_n) - \frac{m^2}{2}f'(w_n)}{f'(t_n) - (\frac{m}{m+2})^{-m}f'(w_n)}\frac{f(t_n)}{f'(t_n)} \tag{8}$$

where m - diversity of roots.

Mitlif [XV] suggested a method of 3 steps to determine the roots of a non-linear equation. The order of convergence of the method is 5.

$$w_n = t_n - \frac{h(t_n)}{h'(t_n)} \tag{9}$$

$$u_n = w_n - \frac{2h(w_n)h'(w_n)}{2h'^2(w_n) - h(w_n)h''(w_n)} \tag{10}$$

$$t_{n+1} = w_n - \frac{2[h(w_n) + h(u_n)]\, h'(w_n)}{2h'^2(w_n) - [h(w_n) + h(u_n)]\, h''(w_n)} \tag{11}$$

Fang et al. [V] formulated an algorithm using a modified quasi-Newton method to solve nonlinear equations. Sharma and Kumar [XVIII] introduced a technique achieving 8th-order convergence with 4 evaluations per iteration. Malhotra et al. [XIX] examined the fatigue failure. Gong et al. [XX] showed a survey utilizing Intelligent Optimization Algorithms to identify multiple roots of a nonlinear equation. Al-Obaidi and Darvishi [I] designed a new multi-step frozen Jacobian repetitive method, with a 3rd-order convergence and high efficiency. Bayrak et al. [IV] used the fractional derivative and the fractional expansion of the Taylor series to design a modified Newton-Raphson method, obtaining the 1st and 2nd-order fractional Newton-Raphson methods.

In addition, Ahmad and Singh [II] developed a 4-step repetitive technique amalgamating Newton, Householder, Halley, and Steffensen techniques, achieving a convergence order of 36. Jin et al. [XXIV] used the uncertain barrier swaption model by incorporating a fractional differential operator. Cao et al. [XXV] outlined an algorithm to forecast water quality using the dendritic neuron model. Malhotra et al. [XII, XIII] delved into a reliability-based model using Markovian processes. Additionally, Kumar et al. [XXVII] employed techniques based on machine learning for learner-centric training, with methods exhibiting $5^{th}$and 7^{th}-order convergence. In addition, Singh & Sharma [XXVIII] described two multi-step iterative methods requiring two Jacobian matrices only and a single matrix inversion per iteration. Sharma et al. [XXI] presented a 4^{th}-order iterative method for the solution of nonlinear equations in science and engineering. It provides better convergence, even when derivatives tend to root, widening its scope to treat the critical points. Gorashiya and Shah [XXX] gave a new iteration-based algorithm to find algebraic as well as transcendental equations, based on a fixed point and initial guess values on the x-axis, by employing the slope of a line and employing the Taylor series to compute the derivation. With 2^{nd}-order convergence and computations of two functions per step, it sets the computational efficiency index equal to 1.414 and informational efficiency equal to 1, proven by example solutions and comparisons with Newton's method. Siwach and Malhotra [XXVI] emphasized the widespread use of iterative methods like Newton-Raphson, Halley's, and Cauchy's for solving nonlinear scalar equations in scientific and engineering domains. They highlighted the challenges posed by the high computational costs of these traditional approaches. Their study underscores the ongoing pursuit of more efficient and accurate numerical methods to improve convergence and reduce computational burden in nonlinear problem-solving.

Finding solutions to nonlinear equations is far-reaching in science and engineering. Newton-Raphson and Halley are iterative methods that may be quite costly to carry out because of the evaluation of the derivatives. In a bid to enhance efficiency [VIII, IX, X], there are a variety of methods developed of higher order. The strengths of existing methods usually fail at complex roots, and some get weak with close branch points or non-analytic singularities. This technique overcomes these difficulties by giving a derivative-free update to the higher-order derivatives and keeping the form of the first derivative used as in the Newton method. The paper introduces a better version of an iterative technique in solving nonlinear equations with an eye on both computational efficiency and the capability of finding complex roots, and the starting point is real or complex.

The authors present an iterative technique for solving nonlinear equations without the need for second or higher-order derivatives, minimizing computational costs. The technique seeks complete solutions, allowing one to spot complex roots. Advantages include efficient problem-solving, fewer iterations needed, cost-effectiveness by avoiding higher-order derivatives, and the capability to use complex or real initial guesses for identifying all roots. Validation across numerical problems demonstrates its effectiveness.

II. Development of Method

Let a nonlinear equation $h(y) = 0$.

where $h(y)$is a differential function in an interval D, which is a subset of $R \subset C$. Consider a parabolic equation with a shifted origin.

$$h(y) = c_0 + c_1 y^2 \tag{12}$$

Let y_n represent the n^{th} approximation obtained from Equation (12). By substituting y_n into Equation (12), we establish a relationship for our approximation. Upon differentiation with respect to y, which we further evaluated at $y = y_n$. Considering y_{n+1} as an exact root of Equation (12). This delineates the iterative process and the relationship between successive approximations and exact roots. At last, all the equations were solved to find the values of c_0 and c_1.

$$c_0 = \frac{2\,h(y_n) - y_n h'(y_n)}{2} \tag{13}$$

$$c_1 = \frac{h'(y_n)}{2\,y_n} \tag{14}$$

Substituting these values

$$y_{n+1} = \sqrt{y_n\,(y_n - \frac{2\,h(y_n)}{h'(y_n)})} \tag{15}$$

Equation (15) represents our proposed method.

In addition, the authors applied the method for analytic functions also, if h(y) is an analytic function in C., the formula reduces to

$$y_{n+1} = y_n - \frac{h(y_n)}{2(y_n)} \tag{16}$$

This function requires only 1 iteration and 1 derivative computation per iteration.

The Newton-Raphson algorithm updates with a step size whose choice depends directly on the derivative of the function at the current point. The suggested procedure involves the denominator, and this has a damping term that is proportional to the current iterate instead of the derivative.

Damping and Stability: The above approach has an implicit damping of the iterative step, as it is possible that it could increase at a higher rate than compared to regions of steep gradient, giving smaller update steps. This dampening is useful when Newton-Raphson risked overshoots, particularly with initial estimates very far away from the root. Truncation Error: The Newton-Raphson approach estimates the function as linear in all steps, and this, therefore, can create more truncation errors with functions that are more nonlinear. In some cases, the proposed approach may address curvature issues better via a parabolic approximation, leading to a smaller local truncation error.

III. Convergence Analysis

The order of convergence of an iterative method is a measure of how fast an iterative method converges to the root. The order of convergence of an iterative method is defined by the largest positive real number p for which certain conditions hold. These conditions typically relate to the rate at which the error diminishes as the iteration advances.

$$|e_{n+1}| \le K\,|e_n|^p \tag{17}$$

for some constant $K \neq 0$, where $e_n = w_n - \alpha$ and $e_{n+1} = w_{n+1} - \alpha$ are the errors in the n^{th} and $(n+1)^{th}$ approximation, respectively. K is referred to as the asymptotic error constant.

Theorem Let d belong to D, be a simple root of a function f: D → R which is sufficiently differentiable in an open interval D. The 2nd-order method is defined by equation (18).

Proof: Assume d is simple root of h(y)=0 and after replacing $y_n = b_n + d$ in equation (18), we obtain

$$b_{n+1} + d = \sqrt{(b_n + d)\left\{(b_n + d) - \frac{2\,h(b_n+d)}{h'(b_n+d)}\right\}} \quad (19)$$

Note: h(d) = 0 as d is a simple root, and expand h (b_n + d) and h′ (b_n + d) about the point 'd' using Taylor's series and replacing the values of h (b_n + d) and h′ (b_n + d), we get

$$b_{n+1} = \frac{1}{2d}\left(d\,\frac{h''(d)}{h'(d)} - 1\right) b_n^2 + O\left(b_n^3\right) \quad (20)$$

Omitting the terms of b_n having power greater than or equal to 3, we get

$b_{n+1} = C\, b_n^2$, where $C = \frac{1}{2a}\left(a\,\frac{h''(\alpha)}{h'(\alpha)} - 1\right)$

Hence, the order of convergence is 2. The 2nd-order convergence of the technique implies that it converges faster than linear convergence methods, such as the secant method and, bisection method. This characteristic enhances the efficiency and speed of the algorithm, making it a valuable tool for solving nonlinear equations with greater accuracy.

The authors expanded the convergence analysis to a function of a complex domain.

For $h: C \to C$, assuming the analyticity near the root, the proposed method retains quadratic convergence as long as h is holomorphic and there are no singularities or branch points near the root. Proceeding as above,

Behavior at Branch Points and Non-Analytic Singularities: Near branch points, or non-analytic singularities in the complex plane, the method may become unstable, owing to the failure of the assumption of Taylor expansion. On these points, the derivative may not lie in the range of real finite numbers or lie in the range of infinite numbers; and in that case, the denominator in the iteration expression will tend towards zero, and the iteration will be divergent or oscillatory. Practically, the method can still find roots in areas not close to singularities; however, it can not be guaranteed that it will converge near these critical points, and careful choice of initial guesses or modification of the method is usually mandatory.

IV. Graphical Illustration

This section provides a visual representation of the convergence behaviour and root trajectories corresponding to the examples discussed above. Each graph plots the underlying function curve f(w), highlights the starting point $w_0$with a distinct marker, and traces the successive iterates as they march toward the final root α.

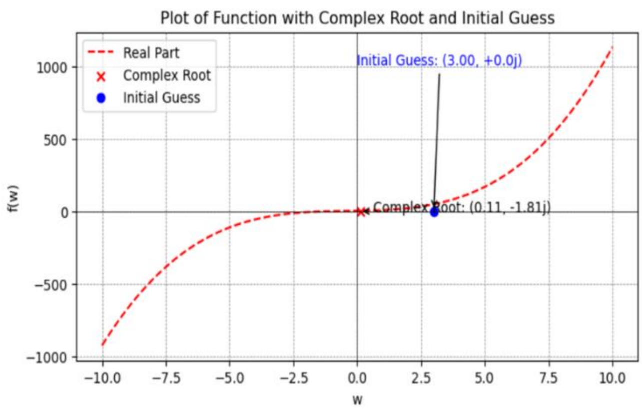

Fig. 1. Graph of f(w) = $w^3 + w^2 + 3w + 4$ with Initial Guess and Complex Root

Figure 1 represents the function f(w) = $w^3 + w^2 + 3w + 4$, where the variable w is along the x-axis and f(w) on the y-axis. The function has been plotted over a range of values for w.

One of the roots of the function is located at approximately w=0.11 –1.81i, indicating that it is a complex root. The initial guess for finding this root is w=3, which is a real value.

Table 1 shows the absolute errors and residuals at each iteration are given in the following table:

Table 1: Absolute errors and Residuals at each iteration

iteration	Absolute error $\|y_n-r\|$	Residual
0	3.000	64.0000
1	2.4870	18.5121
2	0.6305	2.5430
3	0.1124	0.2112
4	0.0142	0.0123

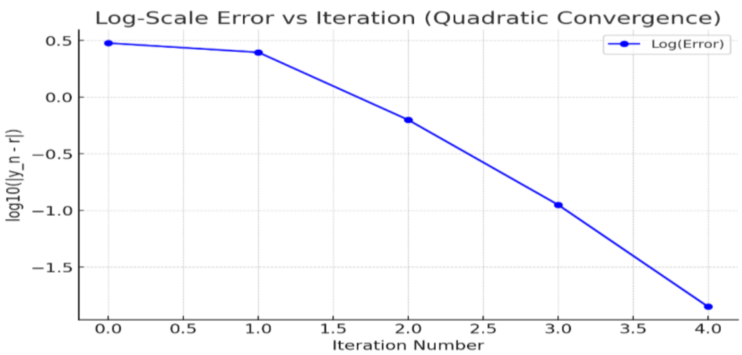

Fig. 2. Graph of Log-Scale Error vs Iteration

Figure 2 shows the graph between logarithmic plot of the absolute error and number of iterations is a straight line with a slope nearer to 2 (second order convergence)

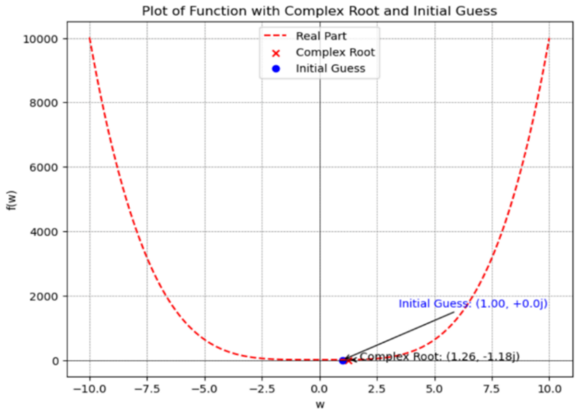

Fig. 3. Graph of f(w) = $w^4 - w + 10$ with Initial Guess and Complex Root

Figure 3 depicts the function f(w) = $w^4 - w + 10$, with w *a*s the independent variable plotted along the x-axis, and the corresponding values of f(w)on the y-axis. The function is analyzed over a range of w values.

A root of the function is situated approximately at w=1.26−1.18i, signifying a complex root. The initial estimate used to find this root is w = 1, a real value.

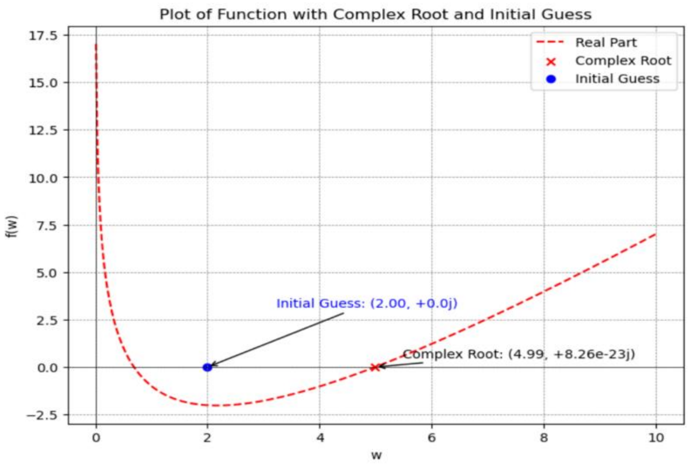

Fig. 4. Graph of f(w) = 2w – 10 log(w) – 3 with Initial Guess and Complex Root

Figure 4 illustrates the function f(w) = 2w – 10 log(w) – 3, where w varies along the x-axis and the values of f(w) are shown on the y-axis. It examines the behavior of the function across different w values.
One of the zeros of the function is approximately at w = 4.99 + 8.26 e^{-23}i, indicating a complex root. The initial estimation utilized to determine this root is w = 2.

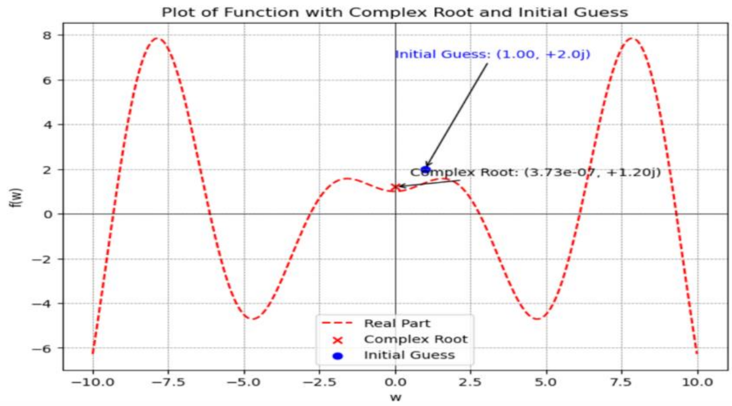

Fig. 5. Graph of f(w) = w sin(w) + cos(w)with Initial Guess and Complex Root

Figure 5 depicts the function, where w varies along the x-axis and the resulting values of f(w) are displayed on the y-axis. It examines how the function behaves for different w values.

A root of the function is found approximately at w = 3.73 e^{-7} + 1.20 i To initiate the root-finding procedure, the initial value chosen is w = 1 + 2i.

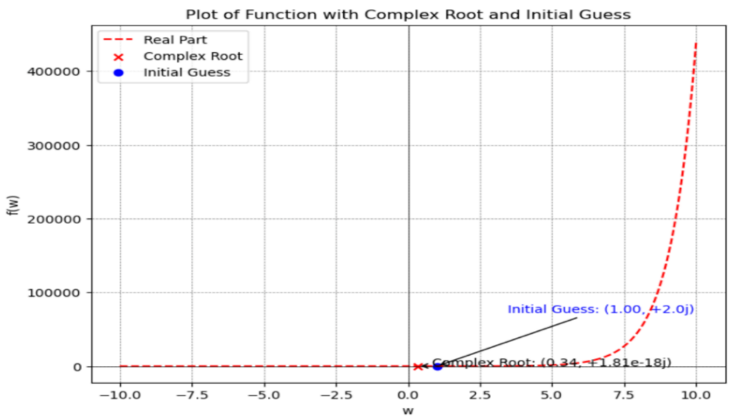

Fig. 6. Graph of f(w) = - cos(w) + 2w e^w with Initial Guess and Complex Root

Figure 6 portrays the function f(w) = - cos(w) + 2w e^w, where w is depicted along the x-axis and the values of f(w) are shown on the y-axis. It explores the behaviour of the function across different w values.

One of the roots of the function is approximately located at w = 0.34 + 1.81 e^{-18}i. To commence the root-finding process, the initial guess used is w = 1+ 2i.

V. Comparison of Computational Efficiency

The efficiency index (EI) is generally used to measure the computational efficiency of an iterative method. The efficiency index (EI) is given by EI = $p^{\frac{1}{c}}$, where p represents the order of convergence and c represents the number of function evaluations.

The convergence order is 2 for the proposed method, and only two functions are computed, resulting in an EI of about $2^{\frac{1}{2}} \approx 1.41421$.

For a complete comparison of the computational efficiency of different iterative techniques for the solution of non-linear equations, we performed experiments to compare the CPU time taken by four different methods: Newton-Raphson, Laguerre's method, Halley's method, and our proposed approach.

We measured CPU times by applying each technique to a selection of non-linear equations with different levels of complexity. Our goal was to derive meaningful insights into how these methods perform in real-world scenarios.

Table 2: Performance estimation of Root-Finding Techniques for Various Functions

Function (f(w))	Root (α)	CPU Time (seconds)			
		Newton's Method	Halley's Method	Laguerre's Method	Proposed Method
$w^3 + w^2 + 3w + 4$	0.11 – i 1.81	0.0006	0.0050	0.0005	0.00008
w sin(w) + cos(w)	$3.73\ e^{-7}$ + i 1.20	0.0005	0.0044	0.0030	0.0001
2w – 10 log(w) – 3	4.99 + i 8.26 e^{-23}	0.00025	0.0017	0.0005	0.00009
$w^4 - w + 10$	1.26 – i 1.18	0.0004	0.0042	0.0003	0.00007
- cos(w) + 2w e^w	0.34 + i 1.81 e^{-18}	0.0005	0.0035	0.0006	0.0003

Table 2 above provides a comparative analysis of the root-finding methods applied to a range of functions, including polynomials, trigonometric, and exponential expressions. Each function was tried with the four methods—Newton's, Halley's, Laguerre's, the Proposed Method, and their corresponding CPU times were measured. Lower CPU times indicate quicker convergence and, consequently, higher computational efficiency. The proposed method consistently exhibited superior performance across various test cases, often outpacing traditional approaches. However, while these results suggest the potential advantages of our method, further evaluation is necessary to fully assess its robustness and accuracy. Overall, the table serves as a useful resource for comparing root-finding algorithms and evaluating their effectiveness across diverse mathematical problems.

VI. Conclusion

Our research paper presents a novel numerical technique that is designed to find both complex as well as real roots of nonlinear equations efficiently. The method, which demonstrates second-order convergence, is particularly notable for its ability to accurately determine complex roots, even when starting with a real initial guess. If the function is holomorphic and has no branch points or singularities around the root, quadratic convergence is maintained.

Extensive testing on a wide range of numerical problems highlights the method's superiority, requiring fewer iterations to solve problems, thereby improving computational efficiency. The authors implemented their method using Python 3.10.9 and conducted CPU time analysis to emphasize its practical benefits in terms of computational cost-effectiveness. By eliminating the requirement to evaluate derivatives of higher order, this technique provides a flexible and efficient solution for solving nonlinear equations, ensuring both accuracy and efficiency across different applications.

Conflict of Interest:

The authors declare no conflicting interests.

References

I. Abbasbandy, Saeid. "Extended Newton's Method for a System of Nonlinear Equations by Modified Adomian Decomposition Method." *Applied Mathematics and Computation*, vol. 170, no. 1, 2005, pp. 648–656. 10.1016/j.amc.2004.11.061.

II. Ahmad, N., and Singh, V. P. "A New Efficient Four-Step Iterative Method with 36th-Order Convergence for Solving Nonlinear Equations." *Journal of Nonlinear Modeling and Analysis*, vol. 5, no. 2, 2023, pp. 1-8.

III. Al-Obaidi, R. H., and M. T. Darvishi. "Constructing a Class of Frozen Jacobian Multi-Step Iterative Solvers for Systems of Nonlinear Equations." *Mathematics*, vol. 10, no. 16, 2022, pp. 2952–2965. 10.3390/math10162952.

IV. Bayrak, M. A., A. Demir, and E. Ozbilge. "On Fractional Newton-Type Method for Nonlinear Problems." *Journal of Mathematics*, 2022. 10.1155/2022/3768720.

V. Fang, X., Q. Ni, and M. Zeng. "A Modified Quasi-Newton Method for Nonlinear Equations." *Journal of Computational and Applied Mathematics*, vol. 328, 2018, pp. 44–58. 10.1016/j.cam.2017.07.012.

VI. Gong, W., Z. Liao, X. Mi, L. Wang, and Y. Guo. "Nonlinear Equations Solving with Intelligent Optimization Algorithms: A Survey." *Complex System Modeling and Simulation*, vol. 1, no. 1, 2021, pp. 15–32.

VII. Hansen, E., and M. Patrick. *A Family of Root Finding Methods*. Springer-Verlag, 1977, pp. 257–269. 10.1007/BFb0062109.

VIII. Kaur, H., and Malhotra, R. "Comparative Analysis of a Redundant System Subject to Inspection of a Manufacturing Plant." *Journal of Mechanics of Continua and Mathematical Sciences*, 2024, pp. 85–105. 10.26782/jmcms.spl.11/2024.05.00007

IX. Kaur, H., and Malhotra, R. "Profit Analysis of a System of Non-Identical Units with Varying Demand." *Palestine Journal of Mathematics*, vol. 14, Special Issue I, 2025, pp. 183–197.

X. Kaur, H., and Malhotra, R. "Stochastic Analysis of a Two-Unit Standby Autoclave System with Inspection and Varying Demand." *Journal of Mechanics of Continua and Mathematical Sciences*, 2024, pp. 147–163. 10.26782/jmcms.spl.11/2024.05.00011

XI. Maheshwari, A. K. "A Fourth-Order Iterative Method for Solving Nonlinear Equations." *Applied Mathematics and Computation*, vol. 211, no. 2, 2009, pp. 383–391. 10.1016/j.amc.2009.02.048.

XII. Malhotra, R. and Taneja, G. "Comparative Analysis of Two Stochastic Models Subjected to Inspection and Scheduled Maintenance." *International Journal of Software Engineering and Its Applications*, vol. 9, no. 10, 2015, pp. 179–188.

XIII. Malhotra, R. and Taneja, G. "Comparative Analysis of Two Stochastic Models with Varying Demand." *International Journal of Applied Engineering Research*, vol. 10, no. 17, 2015, pp. 37453–37460.

XIV. Melman, A. "The Double-Step Newton Method for Polynomials with All Real Zeros." *Applied Mathematics Letters*, vol. 20, no. 6, 2007, pp. 671–675. 10.1016/j.aml.2006.07.007.

XV. Mitlif, R. J. "New Iterative Method for Solving Nonlinear Equations." *Baghdad Science Journal*, vol. 11, no. 4, 2014, pp. 1649–1654.

XVI. Noor, M. A., and M. Waseem. "Some Iterative Methods for Solving a System of Nonlinear Equations." *Computers and Mathematics with Applications*, vol. 57, no. 1, 2009, pp. 101–106. 10.1016/j.camwa.2008.08.007.

XVII. Ostrowski, A. M. *Solution of Equations and Systems of Equations*. Academic Press, 1966.

XVIII. Parhi, S. K., and D. K. Gupta. "A Sixth-Order Method for Nonlinear Equations." *Applied Mathematics and Computation*, vol. 203, no. 1, 2008, pp. 50–55. 10.1016/j.amc.2008.04.033.

XIX. Popovski, D. B. "A Note on Neta's Family of Sixth-Order Methods for Solving Equations." *International Journal of Computer Mathematics*, vol. 10, no. 1, 1981, pp. 91–93. 10.1080/00207168108803241.

XX. Roman, C. "A Family of Newton-Chebyshev-Type Methods to Find Simple Roots of Nonlinear Equations and Their Dynamics." *Communications in Numerical Analysis*, vol. 2017, no. 2, 2017, pp. 172–185.

XXI. Sharma, H., R. Behl, M. Kansal, and H. Ramos. "A Robust Iterative Family for Multiple Roots of Nonlinear Equations: Enhancing Accuracy and Handling Critical Points." *Journal of Computational and Applied Mathematics*, vol. 444, 2024, p. 115795. 10.1016/j.cam.2023.115795.

XXII. Sharma, J. R., and R. K. Guha. "A Family of Modified Ostrowski Methods with Accelerated Sixth-Order Convergence." *Applied Mathematics and Computation*, vol. 190, no. 1, 2007, pp. 111–115. 10.1016/j.amc.2007.01.083.

XXIII. Sharma, J. R., R. K. Guha, and R. Sharma. "Some Variants of Hansen-Patrick Method with Third and Fourth Order Convergence." *Applied Mathematics and Computation*, vol. 214, no. 1, 2009, pp. 171–177. 10.1016/j.amc.2009.03.063.

XXIV. Sharma, J. R., and R. Sharma. "Modified Jarratt Method for Computing Multiple Roots." *Applied Mathematics and Computation*, vol. 217, no. 2, 2010, pp. 878–881. 10.1016/j.amc.2010.06.021.

XXV. Sharma, R., and A. Bahl. "An Optimal Fourth-Order Iterative Method for Solving Nonlinear Equations and Their Dynamics." *Journal of Complex Analysis*, 2015, Article ID 935723. 10.1155/2015/935723.

XXVI. Siwach, A., and R. Malhotra. "Complex Roots-Finding Method of Nonlinear Equations." *Contemporary Mathematics*, vol. 5, no. 3, 2024, pp. 2848–2857.

XXVII. Singh, S., and D. K. Gupta. "A New Sixth-Order Method for Nonlinear Equations in R." *The Scientific World Journal*, vol. 2014, Article ID 509579, 2014. 10.1155/2014/509579.

XXVIII. Traub, J. F. *Iterative Methods for the Solution of Equations*. Prentice Hall, 1964.