Ain Shams Engineering Journal 16 (2025) 103670

Contents lists available at ScienceDirect

Ain Shams Engineering Journal

journal homepage: https://www.sciencedirect.com

Full Length Article

Converging efficiency: Computational and fractal insights into parallel non-linear schemes

Mudassir Shams [a], Nasreen Kausar [a], Ali Akgül [b,c,d,e,f], Tonguç Çağın [g,*]

[a] Department of Mathematics, Faculty of Arts and Science, Balikesir University, Balikesir 10145, Turkey
[b] Department of Electronics and Communication Engineering, Saveetha School of Engineering, SIMATS, Chennai, Tamilnadu, India
[c] Siirt University, Art and Science Faculty, Department of Mathematics, 56100 Siirt, Turkey
[d] Department of Computer Engineering, Biruni University, 34010 Topkapı, Istanbul, Turkey
[e] Near East University, Mathematics Research Center, Department of Mathematics, Near East Boulevard, PC: 99138, Nicosia, Mersin 10, Turkey
[f] Applied Science Research Center, Applied Science Private University, Amman, Jordan
[g] College of Business Administration, American University of the Middle East, Kuwait

ARTICLE INFO

MSC:
65H04
65H05
65H10
65H17

Keywords:
Parallel scheme
Engineering problems
Complex dynamics
Fractal
Residual error

ABSTRACT

This study presents and examines a parallel method for the simultaneous approximation of all roots of nonlinear equations. With the use of a parallel computing architecture, the algorithm aims to enhance computational efficiency. An exhaustive convergence analysis corroborates the finding that the developed scheme converges at sixth order. To optimize parameter values and speed up the convergence rate of the proposed parallel technique, the concepts of dynamical and parametric planes are employed. The computational efficiency percentage demonstrates that the new parallel method is more efficient and involves fewer arithmetic operations compared to the current methods. Randomly chosen initial values are employed to demonstrate the engineering problems have been subjected to comparative analysis, which shows that the suggested parallel schemes surpass traditional methods in residual error, convergence rate, CPU time, memory usage, and computational cost. The findings indicate that the approach holds promise as a means of addressing nonlinear equations in scientific and engineering contexts.

1. Introduction

In engineering and allied domains, nonlinear equations are of utmost importance, forming a core element of research activities. The field of engineering is complex and dynamic, and real-world problems often display nonlinear behaviours. This necessitates the development and examination of nonlinear equations to accurately represent and comprehend their intricate dynamics. From mechanical vibrations and fluid flow patterns to electrical circuits and control systems, nonlinear equations capture the inherent complexities that linear models may not accurately represent [1]. These nonlinear equations are defined as:

$$g(x) = 0. \tag{1}$$

They provide a more realistic representation of the relationships among variables, allowing engineers and researchers to address intricate phenomena, such as nonlinearity of the material [2,3], structural deformations [4], and chaotic behaviour [5]. Moreover, nonlinear equations play a pivotal role in optimization problems, aiding in the design of efficient structures and systems. In engineering and related domains, the investigation and solution of nonlinear equations contribute not only to theoretical advances but also to practical applications, influencing the development of cutting-edge techniques and methodologies. The study of nonlinear phenomena is essential for pushing the boundaries of engineering knowledge, leading to breakthroughs in various disciplines, and paving the way for improved and more accurate engineering solutions [6]. In science and engineering, solving nonlinear equations in general, and polynomial equations in particular, is an ancient problem [7]. Abel's Impossibility Theorem [8] is crucial for highlighting the limitations in finding general algebraic solutions for higher-degree polynomial equations. This theorem influenced the development of mathematical theories, resulted in the development of Galois theory, and has practical implications for numerical approaches in various scientific and

* Corresponding author.
E-mail addresses: mudassir.shams@balikesir.edu.tr (M. Shams), kausar.nasreen57@gmail.com (N. Kausar), aliakgul00727@gmail.com (A. Akgül), tonguc.cagin@aum.edu.kw (T. Çağın).

https://doi.org/10.1016/j.asej.2025.103670
Received 26 January 2025; Received in revised form 1 July 2025; Accepted 2 August 2025

Available online 18 August 2025

engineering disciplines where nonlinear equations or polynomials that simulate these nonlinear equations play a significant role [9,10]. Numerical iterative approaches are particularly important in these situations because exact or analytical methods do not provide a closed-form solution. Consequently, numerical techniques have the following advantages over existing or analytical schemes:

- – Iterative techniques are necessary because many nonlinear equations lack closed-form solutions and provide a scientific and flexible way to approximate the problem. This adaptability extends to the resolution of systems of equations, allowing iterative methods to seamlessly handle multidimensional problems characteristic of engineering research.
- – The flexibility inherent in iterative approaches is pivotal, enabling engineers and researchers to address a diverse array of non-linearity prevalent in real-world scenarios.
- – Numerical stability is crucial in engineering applications, and iterative methods, with their convergence properties, provide a reliable means to navigate uncertainties and numerical errors.
- – The computational efficiency of iterative methods is paramount in engineering, where large-scale problems and real-time applications demand swift and accurate solutions.
- – The global convergence properties of certain iterative techniques enhance their robustness, ensuring convergence from diverse initial conditions.
- – In the realm of optimization, where nonlinear equations often underpin the search for optimal solutions, iterative methods emerge as indispensable tools.

The Newton method, originally devised by Isaac Newton and later refined by Joseph Raphson, is one of the most well-known and traditional iterative techniques. It is still used often because it shows quadratic convergence under optimal circumstances, if the initial estimate is sufficiently close to the exact root [11]. To improve the performance of Newton's method, higher-order techniques and other modifications have been proposed. Jarratt's technique [12] is a third-order formula that adds extra correction terms while maintaining the same number of function evaluations in each step. Similarly, King's technique [13] introduces a new set of two-step fourth-order algorithms using weighting functions. Using a predictor-corrector structure with minimal computing cost, Ostrowski's method [14], another well-known alternative, achieves fourth-order convergence. Other significant advances include the Traub strategy [15], the Chebyshev-Halley family [16], the Erfanifar et al. [17] scheme, and many others, all of which improve the convergence order or resilience by making better use of available derivative information.

Despite these advances, single-root techniques for nonlinear equation solving have intrinsic limitations. One prominent feature is their local convergence behaviour, which means that the effectiveness of the approach is highly influenced by the original guess. Divergence, cycling, and convergence to extraneous roots can result from poor initialisations. Furthermore, these single-root finding algorithms have particularly poor convergence behaviour in the presence of a cluster of roots or multiple roots, whereas Newton's method returns to linear convergence until significantly adjusted (e.g., via multiplicity-based modifications). These methods are also less appropriate for applications where higher-order derivatives are computationally or analytically intractable, as the Halley and Chebyshev methods require. Computing cost, round-off errors, and sensitivity to function behaviour all contribute to the system's inefficiency as it becomes more complicated or nonlinear.

In order to surmount these restrictions, we turn to parallel methods that demonstrate greater stability, consistency, and predictability, and that display global convergence behaviour in the task of finding all roots of nonlinear equations generally and polynomial equations specifically. The iterative strategy of approximating all roots of nonlinear equations—particularly polynomial equations—inspired a wide range of parallel iterative methods. The Weierstrass method [18]—rediscovered by Durand [19] in 1960 and Kerner [20] in 1966—is sometimes known in literature as the Weierstrass-Durand-Kerner (WDK) method, the prominent parallel scheme for finding all roots of nonlinear equations. The Weierstrass technique, while quadratically convergent, is not reliable against multiple or close roots or starting guesses. To accelerate convergence and stability, Newton corrections and repulsion terms of cubic convergence and higher accuracy of root separation are added to the Ehrlich–Aberth scheme [21,22]. In 1985, Nourein devised a quartic-order technique that enhanced accuracy by employing division differences [23]. Other advances include the Nedzhibov inverse Weierstrass approach [24], which converts the original form with reciprocal terms to improve performance in ill-conditioned situations. Petković and Prodanov [25] developed flexible sixth-order root-finding methods using correction functions and weight terms. Neta [26] proposed extended versions of Ehrlich's approach to obtain faster convergence at a low computing cost. The Herceg–Tričković technique [27], which extends the Newton-Weierstrass approach to attain fourth-order accuracy, and the Zeng rational approximation method [28], which offers fifth-order convergence and quick updates, are two more noteworthy contributions in 2005. Yang [29] developed an improved parallel method for cluster roots in 2006. In 2010, Iliev introduced a very efficient sixth-order algorithm that remains robust even for poorly characterized initial estimates [30]. To compute all roots of nonlinear equations in parallel, Proinov et al. [31] in 2014 also presented a family of adaptive Halley-Weierstrass hybrids with parallel implementation and local convergence. The weight function was introduced in 2017 to increase the performance of difficult root distributions in the Ehrlich approach [32]. Psihoyios et al. [33] presented a predictor-corrector method with high efficiency and better convergence in 2021; many others see, for example, [34–36] and the references therein. Despite significant advances and broad adoption of parallel root-finding techniques for nonlinear equations, there are several drawbacks and limitations that affect their performance and applicability in general. These are described as follows:

- Most parallel algorithms are extremely sensitive to first estimations, requiring the use of complex procedures frequently to choose suitable initial values.
- These techniques usually exhibit local convergence but can diverge or halt on roots that are poorly conditioned or extremely crowded.
- Parallel approaches may present numerical instability and round-off errors when used to polynomials with a high degree or clustered/closely spaced roots.
- Despite being simultaneous methods, the total number of arithmetic operations in each iteration can be fairly large, especially in the inverted and corrected variations.
- Certain classical parallel techniques are specifically developed for finding polynomial roots and may not be applicable to other nonlinear functions.
- In high-performance computing configurations, correction or higher-order approaches can be challenging to expand or apply effectively because of their complex formulations.
- Some parallel techniques do not scale well for degree polynomials or very large systems, resulting in higher memory or computation time requirements.

Our primary goal is to create a set of parallel techniques that are efficient, reliable, and stable for the simultaneous determination of all roots of nonlinear equations. This objective is driven by the methods mentioned above and aims to correct the shortcomings of current parallel approaches. This research details a new method that improves convergence behaviour, numerical stability, and computational efficiency. The primary contributions of the research study are outlined below:

- A novel sixth-order parallel iterative approach is developed for approximating all the roots of nonlinear equations simultaneously.

- The method is assisted by a thorough local convergence analysis, which confirms the sixth order of convergence with the least computations.
- Dynamical analysis, including the development of stability areas in the parametric and dynamical planes, is performed to determine the optimal parameter value, which is then used in parallel schemes to improve convergence.
- The newly developed technique is also robust to poor starting estimations, converging even when estimates are far from the exact roots and on random initial guess values.
- The overall computational efficiency of the parallel scheme is demonstrated in comparison to existing techniques.
- Numerical experiments on a variety of engineering-related test problems confirm the recently developed scheme's advantage in terms of accuracy and rate of convergence.
- A comparison with classical methods shows an improvement in the context of error reduction, the number of iterations, and the computer elapsed time.
- The novel technology opens up new opportunities for parallel computing environments in scientific and engineering applications that require fast and reliable root-finding algorithms.

Through this literature review, we confirm that the strategy we propose is one of a kind. This suggests that not much research has been conducted on the development and analysis of efficient parallel algorithms. Specifically, several methods focus on calculating all roots of nonlinear equations simultaneously while achieving higher-order convergence.

The following is a breakdown of the structure of the paper. After the introduction, in Section 2, we develop and analyze the simultaneous method to find all roots of nonlinear equations. Section 3 describes the dynamical analysis of the parallel scheme. The computational features of the recently created simultaneous methods were compared with the existing methods in Section 4. Section 5 includes numerical results, real-world engineering applications, and a comparison of the proposed and existing parallel techniques. Section 6 concludes the manuscript.

2. Construction of iterative methods for multiple roots having order six

When it is impossible to find a closed-form solution, iterative methods are crucial for determining a single root of a nonlinear equation with precision. By approximating nonlinear equations with polynomial representations through the Taylor series expansion, iterative techniques offer systematic and precise root-finding. The mathematical notations, basic concepts, and existing techniques necessary for describing the method are introduced at the beginning of this section.

Notations

$x_i^{[k]}$	Approximation of the i-th root at the k-th iteration in parallel scheme
α_j	multiplicity of the root ζ_j
SM, ZPM, MSP	Existing parallel schemes
NDP	Newly developed parallel schemes
CPU-time	Computational time in seconds
ζ	Exact root
$\psi(.)$	Weight function
$EL(.)$	Computational efficiency index
$Cr_{[i]}$	critical points

This method enhances the rates of convergence and precision, particularly in proximity to the starting estimate. Such implementations play a crucial role in scientific computing, where nonlinear processes are often represented by polynomials, defined as:

$$g(x) = x^n + a_{n-1}x^{n-1} + \ldots + a_0 = \Pi_{j=1}^{m}(x-\xi_j)^{\alpha_j} = \Pi_{j=1}^{m}(x-\xi_j)^{\alpha_j}, \quad (2)$$

with complex or real coefficients. Let $\xi_1, \ldots, \xi_n$ be the exact solution of (10) with multiplicities $\alpha_1, \ldots, \alpha_n$ and $\sum_{j=1}^{m}\alpha_j = n$. These computer algorithms are especially useful when analytical approaches fail to identify the exact solution of a nonlinear equation. The basic idea behind iterative methods is to start with an initial guess and then use algebraic equations to gradually improve the solution until a solution is found with a suitable degree of accuracy. This process of approximating the solution is repeated until all roots are found.

The iterative technique specifies the classical Newton-Raphson method [37] for finding a single root as follows:

$$x^{[k+1]} = x^{[k]} - \frac{g(x^{[k]})}{g'(x^{[k]})}, (k = 0, 1, \ldots), \; g'(x^{[k]}) \neq 0. \quad (3)$$

Method (3) has local quadratic convergence. To develop efficient numerical methods, certain modifications are made to (3). For example, Chicharro et al. [38] modified (3) by applying the weight functions technique and recover some well-known techniques as:

$$x^{[k+1]} = x^{[k]} - \psi\left(\frac{g(x^{[k]})}{g'(x^{[k]})}\right), \quad (4)$$

with $\psi(0) = 0$, $\psi'(0) = 1$, and $\psi''(0) < \infty$.

Choosing $\psi\left(\frac{g(x^{[k]})}{g'(x^{[k]})}\right) = \left(1 + \frac{\phi_1\left(\frac{g(x^{[k]})}{g'(x^{[k]})}\right)}{1+(\phi_1+\phi_2)\left(\frac{g(x^{[k]})}{g'(x^{[k]})}\right)}\right)\left(\frac{\left(\frac{g(x^{[k]})}{g'(x^{[k]})}\right)}{1+\left(\frac{g(x^{[k]})}{g'(x^{[k]})}\right)\phi_1}\right)$, yields the following schemes:

$$x^{[k+1]} = x^{[k]} - \left(1 + \frac{\phi_1\left(\frac{g(x^{[k]})}{g'(x^{[k]})}\right)}{1+(\phi_1+\phi_2)\left(\frac{g(x^{[k]})}{g'(x^{[k]})}\right)}\right)\left(\frac{\left(\frac{g(x^{[k]})}{g'(x^{[k]})}\right)}{1+\left(\frac{g(x^{[k]})}{g'(x^{[k]})}\right)\phi_1}\right), \quad (5)$$

was developed by Kou and Li in [39] and $\phi_1, \phi_2 \in \mathbb{R}$. Choosing $\psi\left(\frac{g(x^{[k]})}{g'(x^{[k]})}\right) = \frac{2\left(\frac{g(x^{[k]})}{g'(x^{[k]})}\right)}{1+\sqrt{1+4\phi_1\left(\frac{g(x^{[k]})}{g'(x^{[k]})}\right)^2}}$, yields the following schemes:

$$x^{[k+1]} = x^{[k]} - \frac{2\left(\frac{g(x^{[k]})}{g'(x^{[k]})}\right)}{1+\sqrt{1+4\phi_1\left(\frac{g(x^{[k]})}{g'(x^{[k]})}\right)^2}}, \quad (6)$$

was developed by Noor et al. in [40]. Using the weight function technique as mentioned, Shams et al. [41] developed the following quadratic convergent scheme (abbreviated as SM):

$$z^{[k]} = x^{[k]} - \left(\frac{g(x^{[k]})}{g'(x^{[k]})}\right)\left(\frac{1}{1+\phi_1\frac{g(x^{[k]})}{1+\phi_2 g(x^{[k]})}}\right), \quad (7)$$

where $\phi_1, \phi_2 \in \mathbb{R}$. Many one-step and multi-step techniques, like Newton's method (see e.g. [42–44], and many others), are available in the literature for determining the simple root of nonlinear equations at a one time. However, these techniques have several drawbacks, including sensitivity to the initial guess, potential convergence problems, and dependence on derivatives, among many others. Therefore, we use a parallel computer numerical scheme to find all distinct as well as multiple solutions of (3) simultaneously.

Compared to single-root finding approaches, simultaneous schemes exhibit global convergence behaviour and can also be used for parallel computing. Due to their global convergence and parallel computer implementation, iterative methods for approximating all roots of (3) have gained a lot of popularity in recent years (see, e.g., Bhalla [45], Ivanov [46], Cordero [47], Petković [48], Mir [49], Farmer [50], Larbaoui [51], Marcheva [52], Cholakov [53] and reference cited therein [54–56]).

Consider the Weierstrass–Dochev's technique [57] as:

$$x_i^{[k+1]} = x_i^{[k]} - w(x_i^{[k]}), \quad (8)$$

where

$$w(x_i^{[k]}) = \frac{g(x_i^{[k]})}{\prod_{\substack{j=1 \\ j\neq i}}^{m}(x_i^{[k]} - x_j^{[k]})}, \quad (i, j = 1, \dots, m),$$

is the Weierstrass correction.

In [58], Nedzibove et al. proposed an improvement to the classical scheme in Eq. (8), given as:

$$x_i^{[k+1]} = \frac{(x_i^{[k]})^2 \prod_{\substack{j=1 \\ j\neq i}}^{m}(x_i^{[k]} - x_j^{[k]})}{x_i^{[k]} \prod_{\substack{j=1 \\ j\neq i}}^{m}(x_i^{[k]} - x_j^{[k]}) + g(x_i^{[k]})}. \tag{9}$$

The scheme in Eq. (9) is referred to as the inverse Weierstrass technique, abbreviated as WND. The parallel iterative computer approach was used to find all solutions to (3) sequentially. These computer algorithms improve an initial guess for the roots by a series of updates, taking advantage of the link between polynomial coefficients and the corresponding solutions. It is well-known for its reliability and efficiency in converging to the adequate solutions when appropriate starting approximations are utilized. This methodology is especially effective in computational mathematics to solve complicated problems where traditional techniques fail. Considering

$$g(x) = \Pi_{j=1}^{m}(x - \xi_j)^{\alpha_j} = \Pi_{j=1}^{m}(x - \xi_j)^{\alpha_j}. \tag{10}$$

As $g(x^{[k]}) = \Pi_{j=1}^{m}(x^{[k]} - \xi_j)^{\alpha_j}$ and $g(x^{[k]}) = \sum_{j=1}^{m} \prod_{\substack{j=1 \\ j\neq i}}^{m}(x^{[k]} - x_j^{[k]})^{\alpha_j}$. Thus $\vartheta_i^{[k]} = \frac{g(x_i^{[k]})}{g'(x_i^{[k]})}$ (Newtons Corrections) further rewritten as:

$$\frac{1}{\vartheta_i^{[k]}} = \frac{g'(x_i^{[k]})}{g(x_i^{[k]})} = \sum_{\substack{j=1 \\ j\neq i}}^{m} \frac{\alpha_j}{x_i^{[k]} - \xi_j}. \tag{11}$$

This gives

$$\begin{aligned} \frac{1}{\vartheta_i^{[k]}} = \frac{g'(x_i^{[k]})}{g(x_i^{[k]})}, &= \sum_{\substack{j=1 \\ j\neq i}}^{m} \frac{\alpha_j}{x_i^{[k]} - \xi_j}, \\ &= \frac{\alpha_i}{x_i^{[k]} - \xi_i} + \sum_{\substack{j=1 \\ j\neq i}}^{m} \frac{\alpha_j}{x_i^{[k]} - \xi_j}, \end{aligned} \tag{12}$$

$$\frac{\alpha_i}{x_i^{[k]} - \xi_i} = \frac{1}{\vartheta_i^{[k]}} - \sum_{\substack{j=1 \\ j\neq i}}^{m} \frac{\alpha_j}{x_i^{[k]} - \xi_j}, \tag{13}$$

$$\xi_i = x_i^{[k]} - \frac{\alpha_i}{\frac{1}{\vartheta_i^{[k]}} - \sum_{\substack{j=1 \\ j\neq i}}^{m} \frac{\alpha_j}{x_i^{[k]} - \xi_j}}, (i = 1, 2, ..., m). \tag{14}$$

To derive a practical parallel iterative method, we replace the fictional auxiliary variable ξ_j with the current iterative values $x_j^{[k]}$ in (14). This enables the technique to be used without prior knowledge of the exact roots, and the scheme is completely computable. This results in the following updating formula:

$$x_i^{[k+1]} = x_i^{[k]} - \frac{\alpha_i}{\frac{g'(x_i^{[k]})}{g(x_i^{[k]})} - \sum_{\substack{j=1 \\ j\neq i}}^{m} \frac{\alpha_j}{x_i^{[k]} - x_j^{[k]}}}. \tag{15}$$

In 1973, Oliver Aberth created the Aberth-Ehrlich [59] method, a productive iterative approach for concurrently locating all of a polynomial's roots. To enhance convergence, it adds a mutual repulsion term and is based on a correction process to Newton's approach. To improve the rate of convergence of (15) from three to four, Nourein replaced $x_j^{[k]} = x_j^{*[k]} - \vartheta_i^{[k]}$ in (15) as:

$$x_i^{[k+1]} = x_i^{[k]} - \frac{1}{\frac{g'(x_i^{[k]})}{g(x_i^{[k]})} - \sum_{\substack{j=1 \\ j\neq i}}^{m} \frac{1}{x_i^{[k]} - x_j^{*[k]} + \vartheta_j^{[k]}}}. \tag{16}$$

Let $x_1^{[k]}, ..., x_m^{[k]}$ be sufficiently near approximations of the roots $\xi_1, ..., \xi_m$ respectively, of (10), which means that $|\epsilon_i| = \max_{1\leq i\leq m} \left|x_i^{[k]} - \xi_i\right|$ is sufficiently small quantity. Thus, (13) can also be rewritten as:

$$\begin{aligned} \frac{1}{x_i^{[k]} - x_j^{*[k]}} &\cong \frac{1}{x_i^{[k]} - x_j^{[k]} + \alpha_j^{[k]}\vartheta_j^{[k]}}, \\ &= \frac{1}{(x_i^{[k]} - x_j^{[k]})\left(1 + \frac{\alpha_j \vartheta_j^{[k]}}{x_i^{[k]} - x_j^{[k]}}\right)}. \end{aligned} \tag{17}$$

Assuming $|\epsilon_i| << 0$ is small enough to provide $\left|\frac{\alpha_j \vartheta_j^{[k]}}{x_i^{[k]} - x_j^{[k]}}\right| < 1$. Using geometric series for (10), we have

$$\begin{aligned} \frac{1}{x_i^{[k]} - x_j^{*[k]}} &\cong \frac{1}{x_i^{[k]} - x_j^{[k]} + \alpha_j \vartheta_j^{[k]}}, \\ &= \frac{1}{(x_i^{[k]} - x_j^{[k]})}\left(1 - \frac{\alpha_j \vartheta_j^{[k]}}{x_i^{[k]} - x_j^{[k]}} + O\left|\epsilon_i\right|^2\right), \end{aligned} \tag{18}$$

where $x_j^{[k]} = x_j^{[k]} - \vartheta_j^{[k]}$. Thus, by neglecting the higher-order terms in (18), we get:

$$\begin{aligned} \frac{1}{x_i^{[k]} - x_j^{*[k]}} &\cong \frac{1}{x_i^{[k]} - x_j^{[k]} + \vartheta_j^{[k]}}, \\ &= \frac{1}{\left(x_i^{[k]} - x_j^{[k]}\right)}\left(1 - \frac{\alpha_j^{[k]} \vartheta_j^{[k]}}{x_i^{[k]} - x_j^{[k]}}\right). \end{aligned} \tag{19}$$

To improve iteration and accelerate convergence, we use the root derivative approximation in (7) to include second-order corrective terms in (19). This results in a more precise and stable update formula, which takes into account both the mutual influence of roots and the derivative of the function around the current iterate. The derived iterative scheme is:

$$x_i^{[k+1]} = x_i^{[k]} - \frac{\alpha_i}{\frac{g'(x_i^{[k]})}{g(x_i^{[k]})} - \sum_{\substack{j=1 \\ j\neq i}}^{m}\left(\frac{\alpha_j}{x_i^{[k]} - z_j^{[k]}}\right) + \sum_{\substack{j=1 \\ j\neq i}}^{m}\left(\frac{\alpha_j^2 \vartheta_j^{[k]}}{\left(x_i^{[k]} - z_j^{[k]}\right)^2}\right)}, \tag{20}$$

where

$$z_j^{[k]} = x_j^{[k]} - \left(\frac{g(x_j^{[k]})}{g'(x_j^{[k]})}\right)\left(\frac{1}{1 + \phi_1 \frac{g(x_j^{[k]})}{1 + \phi_2 g(x_j^{[k]})}}\right).$$

Thus, we developed a new parallel method (NDP) to find all distinct and multiple roots of (3) simultaneously.

Convergence Analysis: Here, we provide the convergence analysis of the parallel scheme in the form of a theorem to determine the order of convergence.

Theorem 1. *Let $x_1^{[k]}, ..., x_m^{[k]}$ be merely close estimations of the roots, $\xi_1, ..., \xi_m$ respectively, then the order of convergence of the numerical scheme NDP is six.*

Proof. Let $\epsilon_i = x_i^{[k]} - \xi_i$, $\tilde{\epsilon}_i = x_i^{[k+1]} - \xi_i$, then,

$$x_i^{[k+1]} - \xi_i = x_i^{[k]} - \xi_i - \frac{\alpha_i}{\frac{g'(x_i^{[k]})}{g(x_i^{[k]})} - \sum_{j\neq i}\frac{\alpha_j}{x_i^{[k]}-z_j^{[k]}} + \sum_{j\neq i}\frac{\alpha_j^2\vartheta_j^{[k]}}{(x_i^{[k]}-z_j^{[k]})^2}}, \tag{21}$$

$$\tilde{\epsilon}_i = \epsilon_i - \frac{\alpha_i}{\frac{g'(x_i^{[k]})}{g(x_i^{[k]})} - \sum_{j\neq i}\frac{\alpha_j}{x_i^{[k]}-z_j^{[k]}} + \sum_{j\neq i}\frac{\alpha_j^2\vartheta_j^{[k]}}{(x_i^{[k]}-z_j^{[k]})^2}}, \tag{22}$$

$$\tilde{\epsilon}_i = \epsilon_i - \frac{\alpha_i}{\frac{\alpha_i}{\epsilon_i} + \sum_{j\neq i}\frac{\alpha_j}{x_i^{[k]}-\xi_j} - \sum_{j\neq i}\frac{\alpha_j}{x_i^{[k]}-z_j^{[k]}} + \sum_{j\neq i}\frac{\alpha_j^2\vartheta_j^{[k]}}{(x_i^{[k]}-z_j^{[k]})^2}}, \tag{23}$$

$$\tilde{\epsilon}_i = \epsilon_i - \frac{\alpha_i\epsilon_i}{\alpha_i + \epsilon_i\sum_{j\neq i}\alpha_j\left[\frac{1}{x_i^{[k]}-\xi_j} - \frac{1}{x_i^{[k]}-z_j^{[k]}} + \frac{\alpha_j\vartheta_j^{[k]}}{\left(x_i^{[k]}-z_j^{[k]}\right)^2}\right]}. \tag{24}$$

Using Newton's correction in (24), we get

$$\tilde{\epsilon}_i = \epsilon_i - \frac{\alpha_i\epsilon_i}{\alpha_i + \epsilon_i\sum_{j\neq i}\alpha_j\left[\frac{x_i^{[k]}-z_j^{[k]}-\left(x_i^{[k]}-\xi_j\right)}{\left(x_i^{[k]}-\xi_j\right)\left(x_i^{[k]}-z_j^{[k]}\right)} + \frac{\alpha_j\vartheta_j^{[k]}}{\left(x_i^{[k]}-z_j^{[k]}\right)^2}\right]}. \tag{25}$$

This implies

$$\tilde{\epsilon}_i = \epsilon_i - \frac{\alpha_i\epsilon_i}{\alpha_i + \epsilon_i\sum_{j\neq i}\alpha_j\left[\frac{\left(\xi_j-z_j^{[k]}\right)}{\left(x_i^{[k]}-\xi_j\right)\left(x_i^{[k]}-z_j^{[k]}\right)} + \frac{\alpha_j\vartheta_j^{[k]}}{\left(x_i^{[k]}-z_j^{[k]}\right)^2}\right]}, \tag{26}$$

As

$$\vartheta_j^{[k]} = \frac{\epsilon_j^2}{\alpha_j + \epsilon_j^2\sum_{1,i}\alpha_i}. \tag{27}$$

Using (27) in (26), we have

$$\tilde{\epsilon}_i = \epsilon_i - \frac{\alpha_i\epsilon_i}{\alpha_i + \epsilon_i\sum_{j\neq i}\alpha_j\left[\frac{-\epsilon_j^2}{\left(x_i^{[k]}-\xi_j\right)\left(x_i^{[k]}-z_j^{[k]}\right)} + \frac{\alpha_j\epsilon_j^2}{\left(x_i^{[k]}-z_j^{[k]}\right)^2\left(\alpha_j+\epsilon_j^2\sum_{1,i}\alpha_i\right)}\right]}, \tag{28}$$

$$\tilde{\epsilon}_i = \epsilon_i - \frac{\epsilon_i}{1 + \frac{\epsilon_i}{\alpha_i}\sum_{j\neq i}\alpha_j\epsilon_j^2\left[\frac{-\left(x_i^{[k]}-z_j^{[k]}\right)\left(\alpha_j+\epsilon_j^2\sum_{1,i}\alpha_i\right)+\alpha_j\left(x_i^{[k]}-\xi_j\right)}{\left(x_i^{[k]}-\xi_j\right)\left(x_i^{[k]}-z_j^{[k]}\right)^2\left(\alpha_j+\epsilon_j^2\sum_{1,i}\alpha_i\right)}\right]}, \tag{29}$$

$$\tilde{\epsilon}_i = \epsilon_i - \frac{\epsilon_i}{1 + \frac{\epsilon_i}{\alpha_i}\sum_{j\neq i}\alpha_j\epsilon_j^2\left[\frac{-\left(x_i^{[k]}-z_j^{[k]}\right)\epsilon_j^2\sum_{1,i}\alpha_i-\alpha_j\left(x_i^{[k]}-z_j^{[k]}\right)+\alpha_j\left(x_i^{[k]}-\xi_j\right)}{\left(x_i^{[k]}-\xi_j\right)\left(x_i^{[k]}-z_j^{[k]}\right)^2\left(\alpha_j+\epsilon_j^2\sum_{1,i}\alpha_i\right)}\right]}, \tag{30}$$

$$\tilde{\epsilon}_i = \epsilon_i - \frac{\epsilon_i}{1 + \frac{\epsilon_i}{\alpha_i}\sum_{j\neq i}\alpha_j\epsilon_j^2 B_{ij}}, \tag{31}$$

where

$$B_{ij} = \frac{-\left(x_i^{[k]}-z_j^{[k]}\right)\epsilon_j^2\sum_{1,i}\alpha_i + \alpha_j\left(x_i^{[k]}-\xi_j-x_i^{[k]}-z_j^{[k]}\right)}{\left(x_i^{[k]}-\xi_j\right)\left(x_i^{[k]}-z_j^{[k]}\right)^2\left(\alpha_j+\epsilon_j^2\sum_{1,i}\alpha_i\right)},$$

$$= \frac{-\left(x_i^{[k]}-z_j^{[k]}\right)\epsilon_j^2\sum_{1,i}\alpha_i - \alpha_j\epsilon_j^2}{\left(x_i^{[k]}-\xi_j\right)\left(x_i^{[k]}-z_j^{[k]}\right)^2\left(\alpha_j+\epsilon_j^2\sum_{1,i}\alpha_i\right)},$$

$$= \frac{-\left(x_i^{[k]}-z_j^{[k]}\right)\epsilon_j^2\sum_{1,i}\alpha_i - \alpha_j\epsilon_j^2}{\left(x_i^{[k]}-\xi_j\right)\left(x_i^{[k]}-z_j^{[k]}\right)^2\left(\alpha_j+\epsilon_j^2\sum_{1,i}\alpha_i\right)},$$

implies

$$\tilde{\epsilon}_i = \epsilon_i - \frac{\epsilon_i}{1 + \frac{\epsilon_i}{\alpha_i}\sum_{j\neq i}\alpha_j\epsilon_j^4 B_{ij}^*}, \tag{32}$$

$$\tilde{\epsilon}_i = \frac{\frac{\epsilon_i^2}{\alpha_i}\sum_{j\neq i}\alpha_j\epsilon_j^4 B_{ij}}{1 + \frac{\epsilon_i}{\alpha_i}\sum_{j\neq i}\alpha_j\epsilon_j^4 B_{ij}}. \tag{33}$$

Assuming $|\epsilon_i| = |\epsilon_j| = |\epsilon|$. Therefore

$$\tilde{\epsilon}_i = O\left(|\epsilon|^6\right). \tag{34}$$

Hence prove the theorem. □

3. Dynamical analysis

The understanding and use of iterative methods for resolving nonlinear equations in engineering and associated domains are significantly advanced through dynamical analysis. Dynamical analysis, in relation to numerical solutions, consists of looking at the iterative process as a dynamic system and investigating how iterates evolve across successive iterations. It is crucial to comprehend the stability, convergence, and chaotic behaviour of the iterative process to guarantee that the numerical solution is reliable and efficient. In contrast, the fractal analysis [60,61] depicts areas within the solution space where iterative methods converge to a particular root. Investigating basins of attraction provides valuable insight into the influence of initial guesses on convergence patterns, helping to identify the domains where convergence is ensured and the boundaries between regions that may lead to different roots [62]. In this study, dynamical analysis is incorporated to improve the robustness of iterative methods, guiding practitioners to select appropriate initial conditions, optimizing convergence rates, and gaining deeper insights into the behaviour of the numerical solution process. This analytical framework contributes significantly to the development of reliable parallel algorithms for solving complex nonlinear equations, ensuring their applicability and effectiveness in real-world engineering problems. To determine ideal parameter values for the iterative technique SM, parametric planes (Fig. 1(a,b)) are created over the domain $[-2,2]\times[-2,2]\in\mathbb{C}$ using a mesh of 2000×2000 points, using critical points as initial estimations. These graphs indicate convergent (red) and divergent (black) regions, allowing for the selection of stable parameter values from the red zones. The method's stability under these conditions is confirmed by the dynamical planes, i.e., Fig. 3(a-e), that are produced for

$$g(x) = (x-1)(x+1). \tag{35}$$

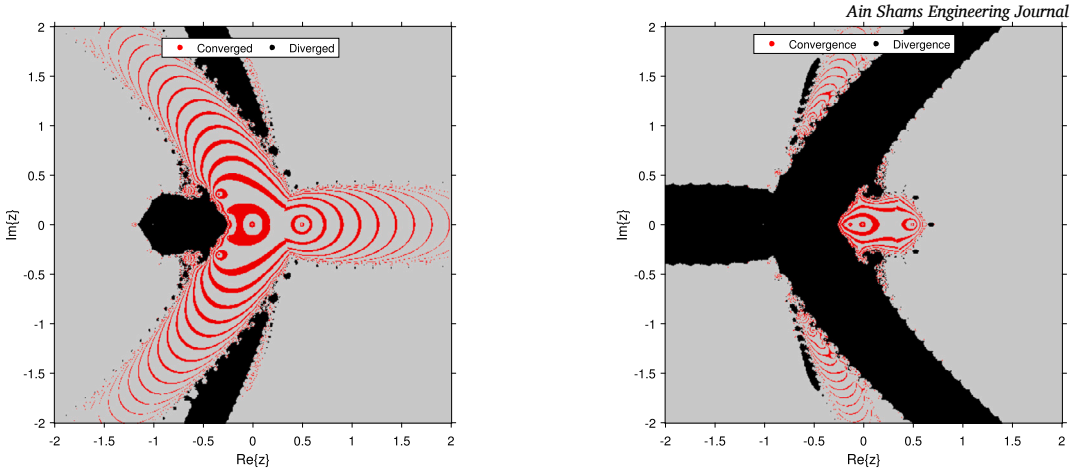
In the complex plane, each color represents the initial guesses that converge to a certain root. In the fractal plots, the convergence to root 1 is shown in blue, to -1 in brown, and the divergence in black. Structured non-overlapping basins show strong convergence behaviour, but parameters beyond the stable zone cause divergence or much lower convergence rates. To generate the dynamical planes, we consider the following iterative map

$$R\left(x,\phi_1,\phi_2\right) = \frac{\left(2\phi_1+\phi_2\right)x^4 + \left(2\phi_1+1\right)x^2 - \phi_2 + 1}{2x\left(\phi_1+\phi_2\right)x^2 - \left(\phi_1+\phi_2-1\right)}, \tag{36}$$

and take a grid 2000×2000 of square $[-2,2]\times[-2,2]\in\mathbb{C}$. Using Möbius transformations of the

$$O\left(x,\phi_1,\phi_2\right) = \frac{x^2\left(x^2 + \left(4\phi_1+4\phi_2-2\right)x + 4 + 1\right)}{1 + \left(4\phi_1+1\right)x^2 + \left(4\phi_1+4\phi_2-1\right)x}. \tag{37}$$

For $\phi_2 = 1$, we obtain the following critical points by putting $R'\left(x,\phi_1,\phi_2\right) = 0$ as:



(a) Parametric palne corresponds to $Cr_{[1]}$. (b) Parametric palne corresponds to $Cr_{[2]}$.

Fig. 1. (a,b), indicates the parametric planes of the iterative scheme SM illustrating the convergence divergence region associated with critical points $Cr_{[1]}$ and $Cr_{[2]}$.

Table 1
Dynamical analysis of the correction terms for finding best parameter values.

Figure	ϕ_1	ϕ_2	Per-Convergence Elapsed time	
Fig. 2(a)	0.1	1.2	75.324536654%	3.23223434
Fig. 2(b)	0.2	0.2	67.878532665%	5.23434234
Fig. 2(c)	1.7	0.3	15.965756525%	6.45466456
Fig. 2(d)	1.3	0.4	25.767886352%	8.65456645
Fig. 2(e)	2.3+2.3i	0.1	41.668657632%	4.56745675

$$Cr_{[1]} = \frac{-\left(\phi_1\right)^2 + 2\sqrt{\vartheta} + 2\phi_1 + 1}{4\phi_1 + 1},$$
$$Cr_{[2]} = -\frac{\left(\phi_1\right)^2 + 2\sqrt{\vartheta} + 2\phi_1 + 1}{4\phi_1 + 1}, \tag{38}$$

where $\vartheta = 4\left(\phi_1\right)^4 + 4\left(\phi_1\right)^3 - \left(\phi_1\right)^2 + \phi_1$ and $Cr_{[3]} = 0$.

$$\left|R'\left(1, \phi_1, \phi_2\right)\right| = \left|\frac{2\left(\phi_1 + \phi_2\right)}{2\phi_1 + \phi_2}\right|. \tag{39}$$

Using the stability function provided in (39), Fig. 2(a-e) shows the stability zone of the iterative technique SM. The 3D stability surface in Figs. 2(a-e) shows the method's norm behaviour over the complex plane. The smoothness and height of the surface represent the rate of convergence; lower and steeper regions adjacent to singularities indicate slower convergence or instability.

Using the optimal parameter values for which the single-root finding approach is most stable, we accelerate the convergence rate of the newly designed parallel numerical schemes. Table 1 shows the values of the parameters, the corresponding elapsed time, and the percentage of convergence and divergence.

4. Percentage computational efficiency analysis for parallel schemes

In engineering and related areas, the computational efficiency of iterative methods for solving nonlinear equations is a crucial factor that affects the effectiveness of numerical solutions and the overall success of computational algorithms. When it comes to nonlinear equations, for which closed-form solutions are frequently hard to come by, iterative methods like parallel numerical schemes are essential. These methods are considered efficient if they converge quickly to precise solutions while keeping the computational burden low. This is particularly critical in engineering applications, where timely and resource-effective solutions are essential. The convergence rate, stability, and capability of iterative methods directly impact their computational efficiency. Researchers try to optimize these methods to handle large-scale problems efficiently, where the size and complexity of systems require robust and swift convergence. The evaluation and improvement of the computational efficiency of iterative methods contribute significantly to the advancement of numerical techniques in engineering research articles. It ensures that simulations, optimizations, and analysis can be performed with precision and within reasonable computational time frames, thereby fostering innovation and reliability in the application of iterative methods in diverse engineering domains. As presented in [63], the computational efficiency index (EL) is defined as:

$$EL(m) = \frac{\log(r)}{\kappa_{as} AS_n + \kappa_m M_n + \kappa_d D_n}. \tag{40}$$

Here

- r is the convergence order of the parallel scheme.
- AS_n is used to represent the number of addition and subtraction arithmetic operations.
- M_n is utilized to show the number of multiplication and
- D_n illustrates the number of divisions for a polynomial of degree n.

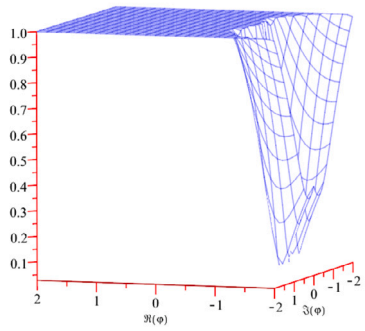
The constants κ_{as}, κ_m, and κ_d are used as weights to compute the relative computational cost of parallel iterative methods to solve nonlinear equations. Applying (40) and the data given in Table 2, we calculate the efficiency ratio $\rho((X),(NDP))$ [64] as:

$$\rho((X),(NDP)) = \left(\frac{EL(X)}{EL(NDP)} - 1\right) \times 100, \tag{41}$$

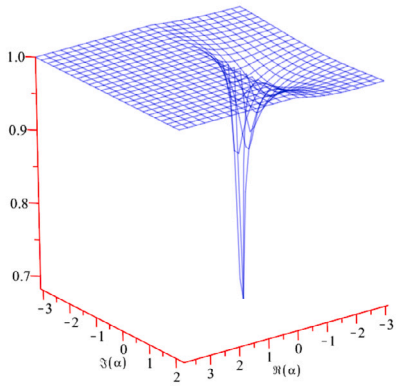
where X is any parallel iterative scheme. Here, we compute and compare the percentage computational efficiency of the newly developed method with the following parallel techniques:

- Petkovic et al. [65] method given as:

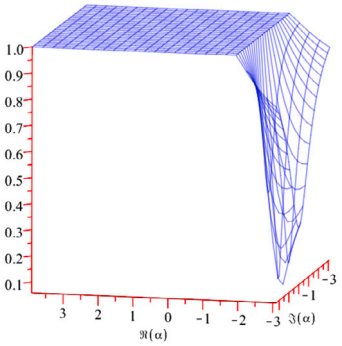
$$x_i^{[k+1]} = x_i^{[k]} - \frac{1}{\frac{1}{N_i(x_i^{[k]})} - \sum_{\substack{j=1 \\ j\neq i}}^{n} \frac{1}{(x_i^{[k]} - Z_j^{[k]})}}, \tag{42}$$



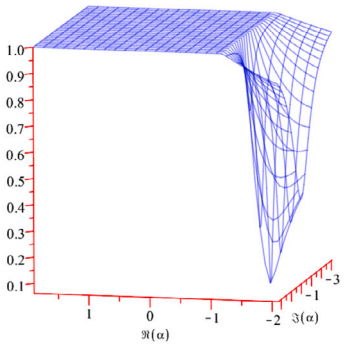
(a) Stability region of SM for $\phi_1 = 0.1$ and $\phi_2 = 1.2$



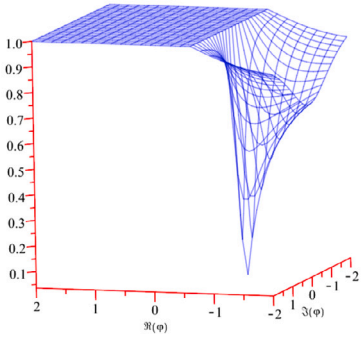
(b) Stability region of SM for $\phi_1 = 0.2$ and $\phi_2 = 0.2$



(c) Stability region of SM for $\phi_1 = 1.7$ and $\phi_2 = 0.3$



(d) Stability region of SM for $\phi_1 = 1.3$ and $\phi_2 = 0.4$



(e) Stability region of SM for $\phi_1 = 2.3 + 2.3i$ and $\phi_2 = 0.1$

Fig. 2. (a-e), indicates the iterative-scheme SM stability regions for various values of parameter ϕ_1 and ϕ_2.

where

$$Z_j^{[k]} = x_j^{[k]} - u(x_j^{[k]})\left(\frac{\beta_j + \gamma_j t(x_j^{[k]})}{1 - k_j t(x_j^{[k]})}\right),$$

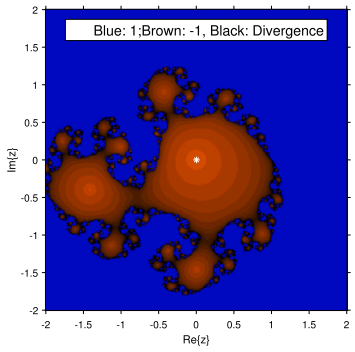
$$t(x_j^{[k]}) = \left(\frac{g(x_j^{[k]} - \theta_j u(x_j^{[k]}))}{g'(x_j^{[k]})}\right), \theta_j = \frac{2\sigma_j}{\sigma_j + 2},$$

$$\beta_j = -\frac{(\sigma_j)^2}{2}, k_j = \left(\frac{\sigma_j + 2}{\sigma_j}\right)^{\sigma_j}$$

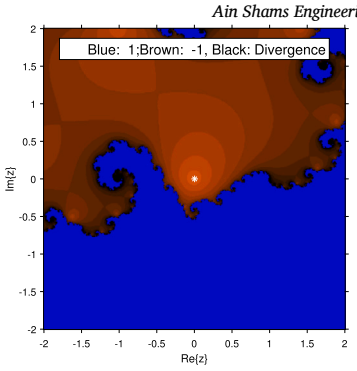
and

$$\gamma_j = \frac{\sigma_j(\sigma_j + 2)}{2} k_j.$$

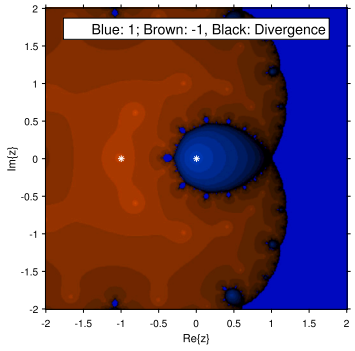
Method (42) is abbreviated as MSP.



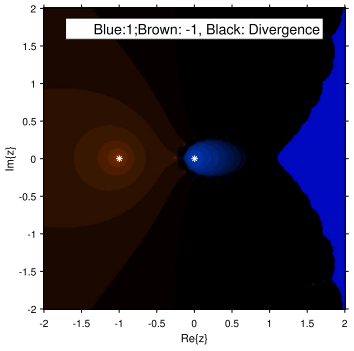
(a) Dynamical plane of SM for$\phi_1 = 0.1$ and $\phi_2 = 1.2$



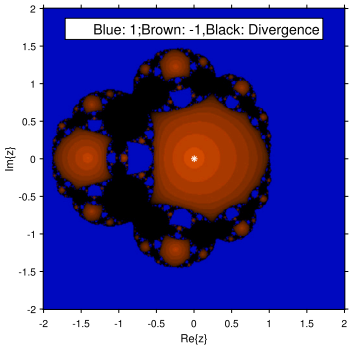
(b) Dynamical plane of SM for$\phi_1 = 0.2$ and $\phi_2 = 0.2$



(c) Dynamical plane of SM for$\phi_1 = 1.7$ and $\phi_2 = 0.3$



(d) Dynamical plane of SM for$\phi_1 = 1.3$ and $\phi_2 = 0.4$



(e) Dynamical plane of SM for$\phi_1 = 2.3 + 2.3i$ and $\phi_2 = 0.1$

Fig. 3. (a-e), indicates the iterative-scheme SM dynamical planes for various values of the parameter ϕ_1 and ϕ_2.

- Zhang et al. method [66] proposed the following fifth-order convergence parallel scheme as

$$x_i^{[k+1]} = x_i^{[k]} - \frac{w(x_i^{[k]})}{1+\rho\left(x_i^{[k]}\right) + \sqrt{\left(1+\rho\left(x_i^{[k]}\right)\right)^2 + 4w(x_i^{[k]})\sum\limits_{\substack{j=1\\ j\neq i}}^{n} \frac{w(x_j^{[k]})}{\left(x_i^{[k]} - x_j^{[k]} - w(x_j^{[k]})\right)}}}, \tag{43}$$

where

$$\rho\left(x_i^{[k]}\right) = \sum_{\substack{j=1\\ j\neq i}}^{n} \frac{w(x_j^{[k]})}{\left(x_i^{[k]} - x_j^{[k]}\right)}.$$

Method (43) is abbreviated as ZPM.

It is evident from Fig. 4 that the NDP-scheme is more consistent and efficient as compared to ZPM and MSP technique.

In Table 2 $\omega_{11} = O(m)$ and Per-Eff represent the percentage of computational efficiency. Percentage computational efficiency is a performance measure of numerical schemes: it quantifies the ratio of accuracy to computational resources: time and memory. It is useful for comparing schemes to decide which one provides acceptable accuracy with mini-

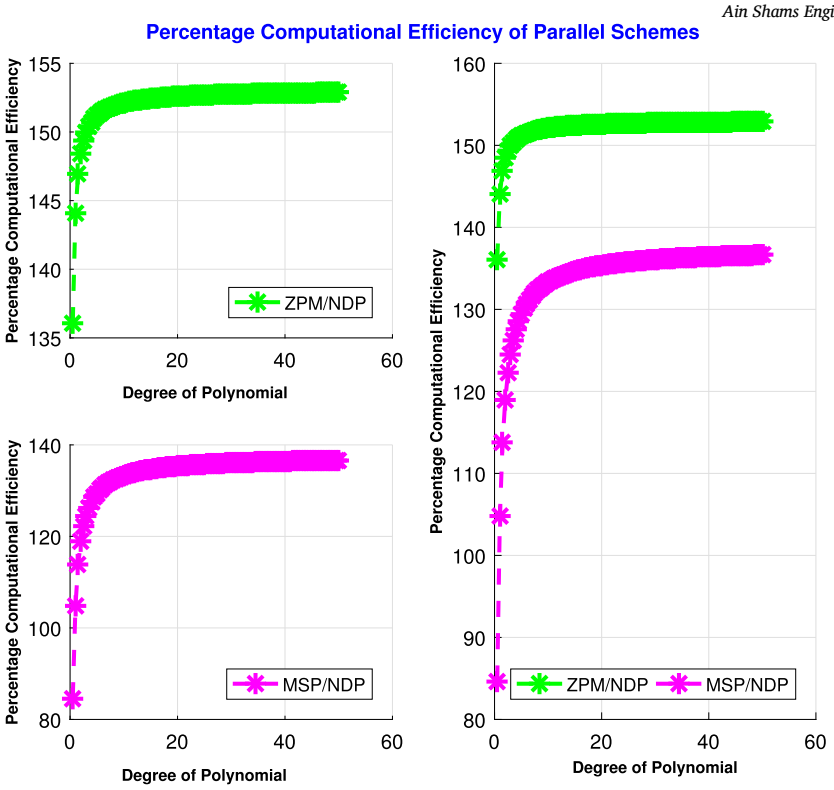


Fig. 4. Illustrate the percentage computational efficiency of the parallel approaches ZPM, MSP and NDP.

Table 2
Operations for finding the efficiency of ZPM, MSP and NDP schemes.

Techniques	$+,-$	$\times$	$\div$	Per-Efficiency
ZPM	$7m^2+\omega_{11}$	$5m^2+\omega_{11}$	$2m^2+\omega_{11}$	35.65%
MSP	$9m^2+\omega_{11}$	$4m^2+\omega_{11}$	$2m^2+\omega_{11}$	41.97%
NDP	$5m^2+\omega_{11}$	$2m^2+\omega_{11}$	$2m^2+\omega_{11}$	65.32%

Table 3
Summary of key attributes of the parallel techaniques.

Methods	Order of Convergence	Computational Cost	$[g(x), g'(x)]$
ZPM	5	$24.56n^2+O(n)$	2
MSP	6	$26.24n^2+O(n)$	3
NDP	6	$13.24n^2+O(n)$	2

mal arithmetic operations. High efficiency indicates a more practical algorithm for large or complex problems.

5. Numerical outcomes

In this section, we conduct a comprehensive comparative analysis between our recently developed parallel method NDP of order six, and Petkovic et al.'s method (MSP) and Zhang's method (ZPM), both exhibiting the local convergence order of six and five, respectively. Several numerical test examples are examined to assess the performance and accuracy of these methods. Using MATLAB R2023a, all calculations were successfully executed. The termination criteria for the computer program were established based on specific conditions, where

$$\epsilon_i^{[k]} = \left\| x_i^{[k+1]} - x_i^{[k]} \right\| < \in = 10^{-30}$$

is the absolute error of consecutive iterations using the norm $\|.\|$. All computational experiments, including numerical simulations, convergence analysis, and dynamical behaviour visualisations, were performed using MATLAB R2023a on a system with the following specifications:

- Processor: Intel(R) Core(TM) i7-12700H CPU @ 2.30GHz
- Operating System: Windows 11 Pro, 64-bit
- RAM: 32 GB DDR4
- Software: MATLAB R2023a with Symbolic Math Toolbox

The entire process for approximating all the roots of a nonlinear equation is described in the Algorithm 1 and the flow digram shown in Fig. 5. These computations were carried out within the computer algebra system CAS-MATLAB R2023a. This detailed comparison aims to provide insight into the efficiency and reliability of our newly developed method, NDP, compared to other methods in the literature. Table 3 illustrates the key attributes of the parallel iterative systems, such as their order of convergence, computational cost, and the number of function and derivative evaluations ($[g(x), g'(x)]$) required.

In engineering applications, the following acronyms are utilized:

Acronyms

SM, ZPM, MSP	Existing methods
NDP	Newly developed scheme
ČŎ	Convergence order
ČPŬ	Computational time
ň	Number of iterations
P-Con$^{\upsilon^*}$	Percentage convergence
ČPŬ-time$^{\upsilon^*}$	Computational time for parallel schemes with random initial guesses
ζ	Exact root
Average-It$^{\upsilon^*}$	Average iterations for parallel schemes with random initial guesses
Max-Error$^{\upsilon^*}$	Max error for parallel schemes with random initial guesses
$\sigma_i^{[\check{n}-1]}$	Local computational order of convergence
Memory-U$^{\upsilon^*}$	Memory utilized for parallel schemes with random initial guesses in Mbs
$\epsilon^{[*]}$	Residual error

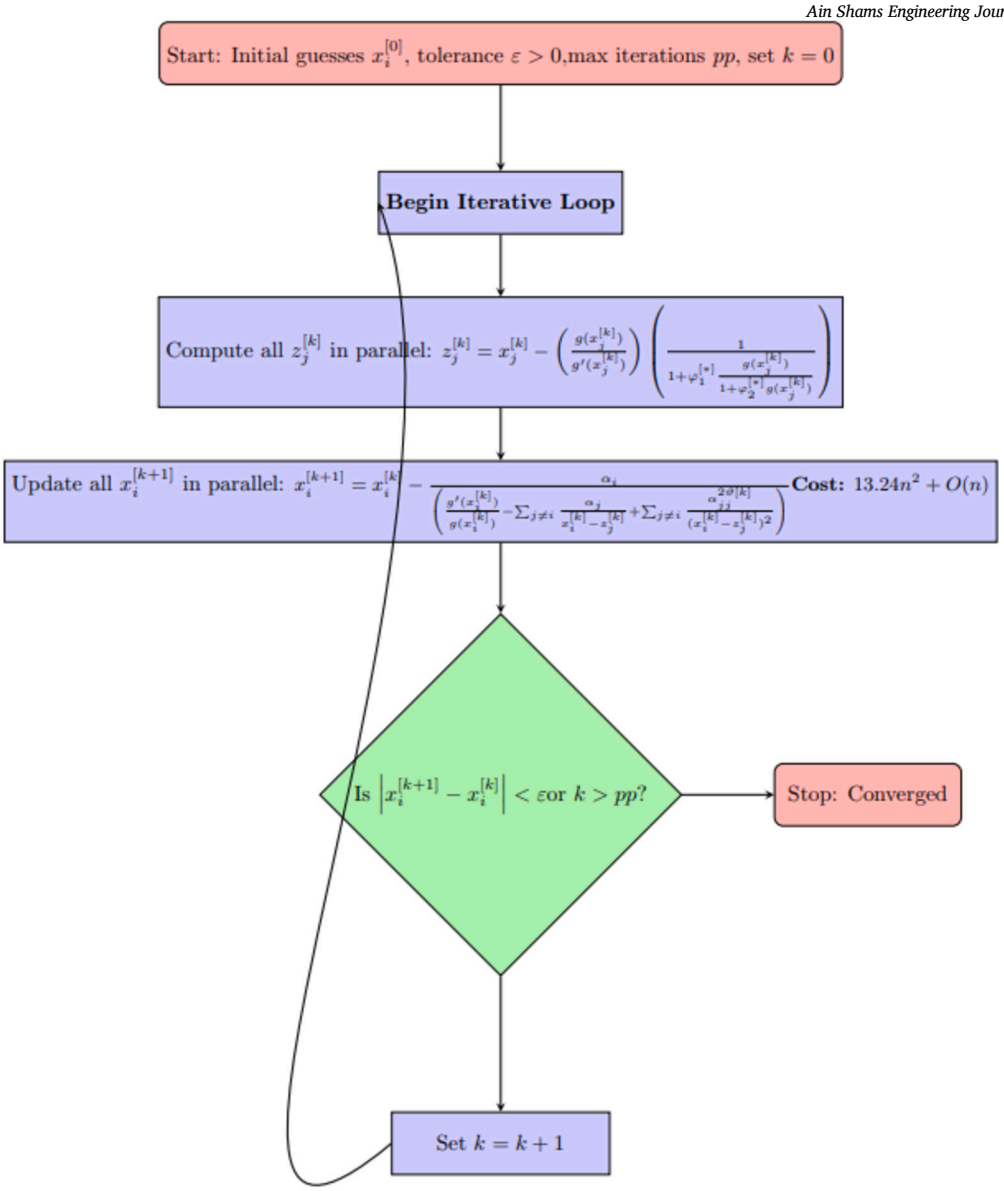


Fig. 5. Flowchart of parallel iterative approach for simultaneous approximation of all roots of nonlinear equations.

Real-World Engineering Applications

This section covers real-world engineering applications that can be approximated using our new and existing parallel techniques.

5.1. Chemical reactor equilibria: an application in chemical engineering [67]

The equilibria of chemical reactors represent the ultimate steady-state positions reached in a reactor This concept plays a crucial role in chemical engineering as it determines the makeup of products and left-over reactants under certain conditions of temperature, pressure, and concentration. The law of mass action governs chemical equilibrium in isothermal batch or continuous stirred-tank reactor (CSTR) conditions, with the equilibrium constant determining the ratio of product to reactant concentrations. Such complex reaction networks, particularly consecutive or simultaneous reversible reaction networks, often create nonlinear algebraic equations based on stoichiometric balances and equilibrium conditions. In these equations, the extent of reaction or species concentrations at equilibrium will be represented by the roots of high-degree polynomials of six. Since positive real roots indicate chemically significant concentrations, they are permitted as long as they are physically reasonable. The system characteristics, such as starting reactant concentrations and equilibrium constants, have a sensitive effect on the quantity and type of these roots. In classical physics, chemical equilibria are states of thermodynamic equilibrium in which the system will minimize its Gibbs free energy and experience no net change other than in response to a disturbance. The accurate solution of these polynomial systems is critical for reactor design, yield optimization, determining conversion efficiency, and guaranteeing safe, cost-effective operation in industrial chemical processes.

We consider a closed isothermal batch reactor involving multiple reversible and nonlinear reactions:

$$\text{Reaction systems:} \begin{cases} \text{(R1):} & \text{A} \rightleftharpoons \text{B (Equilibrium)} \\ \text{(R2):} & 5\text{B} \rightleftharpoons \text{C (Equilibrium)} \\ \text{(R3):} & \text{C} \rightarrow \text{D (Irreversible)} \end{cases}$$

Let

- C_A, C_B, C_C, C_D: Concentrations of species A, B, C, D respectively.
- $K_1^{[*]}$, $K_2^{[*]}$: Equilibrium constants for (R1) and (R2).
- C_{A_0}: Initial concentration of A.

Algorithm 1: Detailed flow of the parallel approach for simultaneously finding all nonlinear equation roots.

Input: Initial guesses $x_i^{[0]}$ for $i = 1, \ldots, N$, tolerance $\epsilon > 0$, maximum iterations pp

Output: Approximated roots $x_i^{[k+1]}$

Set $k = 0$;

while $k \leq pp$ **do**

  for $j = 1$ **to** N **do**

    Compute auxiliary value:

$$z_j^{[k]} = x_j^{[k]} - \left(\frac{g(x_j^{[k]})}{g'(x_j^{[k]})}\right)\left(\frac{1}{1+\phi_1 \frac{g(x_j^{[k]})}{1+\phi_2 g(x_j^{[k]})}}\right)$$

  for $i = 1$ **to** N **do**

    Update:

$$x_i^{[k+1]} = x_i^{[k]} - \frac{\alpha_i}{\left[\frac{g'(x_i^{[k]})}{g(x_i^{[k]})} - \sum_{\substack{j=1 \\ j\neq i}}^{N} \frac{\alpha_j}{x_i^{[k]} - z_j^{[k]}} + \sum_{\substack{j=1 \\ j\neq i}}^{N} \frac{\alpha_{jj}^{2\#[k]}}{(x_i^{[k]} - z_j^{[k]})^2}\right]}$$

  Check convergence: If $|x_i^{[k+1]} - x_i^{[k]}| < \epsilon$ for all i or $k > pp$, then **stop** ;

  Set $k = k + 1$;

- Assume that initially there are no B, C, or D: $C_B(0) = C_C(0) = C_D(0) = 0$.

Defining the reaction stoichiometry and extent of reaction η_1, x for (R1 and R2) as:

$$C_A = C_{A_0} - \eta_1,\ C_B = \eta_1 - 2x, C_C = x \tag{44}$$

and $C_D = \eta_2 \approx 0$ because R3 is slow and negligible in equilibrium. Using the law of mass action at reaction (R1)

$$K_1^{[*]} = \frac{C_B}{C_A} = \frac{\eta_1 - 2x}{C_{A_0} - \eta_1} \tag{45}$$

and at (R2)

$$K_2^{[*]} = \frac{C_c}{C_B^5} = \frac{x}{(\eta_1 - 2x)^5}. \tag{46}$$

Substituting η_1's value from (45) into (46) yields the following:

$$K_2^{[*]} = \frac{x}{\left(\frac{K_1^{[*]} C_{A_0} + 2K_1^{[*]} x}{1+K_1^{[*]}} - 2x\right)^5}. \tag{47}$$

When the stoichiometric and equilibrium conditions are inserted into the law of mass action, a nonlinear algebraic equation is formed, with the roots representing physiologically meaningful equilibrium concentrations. In particular, the fifth-degree polynomial that follows

$$g(x) = 96x^5 - 720x^4 + 2160x^3 - 3240x^2 + 2446x - 729, \tag{48}$$

models the extent of reaction x, which represents the equilibrium concentration of species C in the isothermal batch reactor. Engineers can anticipate reactor composition and maximize product yield by solving for the exact positive roots of this formulation, which corresponds to the fractional conversion of a nitrogen–hydrogen feed at 250 atm and 227 K. The exact roots of (48) are:

$$\zeta_1 = 0.82719, \zeta_{2,3} = 1.1978 \pm 0.6810i, \zeta_{4,5} = 1.1385 \pm 0.5114i \tag{49}$$

and

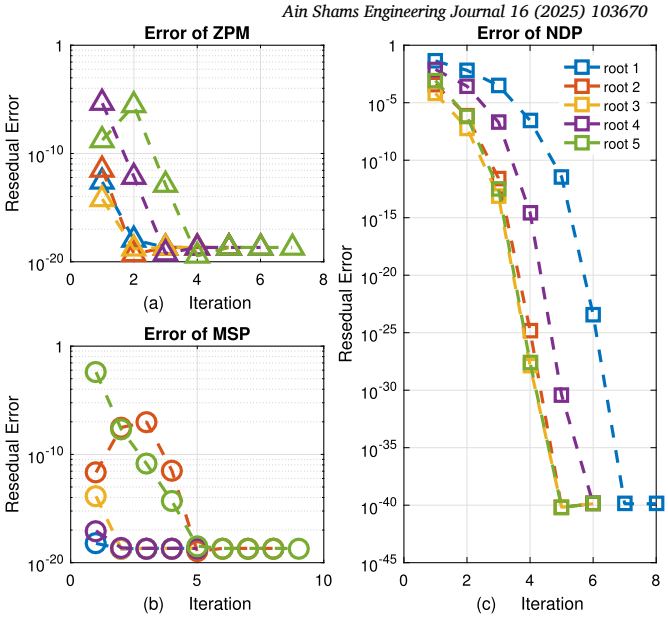


Fig. 6. (a-c), shows the residual error graphs for the parallel schemes—ZPM, MSP, and NDP—that were utilized to solve engineering application 5.1.

$$x_1^{[0]} = 0.9,\ x_{2,3}^{[0]} = 1.1 \pm 0.6i,\ x_{4,5}^{[0]} = 1.1 \pm 0.5i, \tag{50}$$

are chosen as initial guessed values. The results of NDP and ZPM, MSP in terms of numbers to solve (48) are shown in Table 4. In terms of computational convergence order (CO), computational CPU time, residual errors, error graphs and iteration numbers, Table 4 clearly shows that the technique NDP is superior to ZPM, MSP.

The numerical results in Table 4 clearly show that in terms of residual error, the newly developed method NDP outperforms ZPM, MSP, with a high computational order of convergence and a rapid convergence rate. Thus, the parallel technique is sometimes crucial for clustered roots, where convergence analysis is challenging; in order to evaluate convergence behaviour, random initial guesses can be employed. Using some random initial vectors, $v_1^* - v_5^*$, we can determine the global convergence behaviour. On these random initial vectors, NDP takes 17 iterations and achieves a maximum residual error of 7.3e-23 in 0.074 seconds on computers, ZPM takes 35 iterations and achieves a maximum residual error of 0.1e-14 consuming 0.4002 seconds on computer, while MSP takes 35 iterations and achieves a maximum residual error of 2.1e-13 and consuming 0.451 seconds on computers. The results of Table 6 clearly show that NDP outperforms ZPM, MSP and converges more globally. The random initial vectors $v^* = \left[x_1^{[0]}, x_2^{[0]}, x_3^{[0]}, x_4^{[0]}, x_5^{[0]}\right]$ are given in Table 5.

The outcomes of the parallel schemes utilizing the random initial vectors given in Table 5 are presented in Table 6.

In Table 6, Ini-V denotes the set of random initial vectors used to test the global convergence behaviour of the NDP and PP techniques. Based on random initial approximations, Table 6 displays the numerical results of the simultaneous schemes NDP and PP. The Table 6 results clearly show that, in terms of global convergence, the NDP technique performs better than the ZPM, MSP methods.

Using random initial vectors, Table 7 displays the maximum error, computational time in seconds, percentage convergence, average number of iterations, and local computational order of convergence, respectively, as Max-Error$^{v^*}$, ČPŬ-time$^{v^*}$, P-Con$^{v^*}$, Average-It$^{v^*}$, and $\rho_i^{[\hbar-1]}$. Table 7 and Fig. 6(a,c) clearly demonstrate that our technique, NDP, outperforms the existing methods ZPM, MSP in terms of maximum error, computational time, percentage convergence, average number of iterations, and computational order of convergence.

Table 4
Residual error results for closely initialized values to exact roots of (48) using parallel techniques in engineering application 5.1.

Method	ČŎ	ČPŬ	ň	$\epsilon_1^{[\check{n}]}$	$\epsilon_2^{[\check{n}]}$	$\epsilon_3^{[\check{n}]}$	$\epsilon_4^{[\check{n}]}$	$\epsilon_5^{[\check{n}]}$	$\sigma_i^{[\check{n}-1]}$
ZPM	5	0.058	4	3.7e-22	1.5e-26	0.9e-45	9.0e-43	1.3e-40	4.5463
MSP	6	0.047	4	8.1e-62	2.5e-61	1.2e-90	1.2e-90	0.0	5.1434
NDP	6	0.032	4	1.4e-61	0.0	2.0e-146	5.0e-149	0.0	6.1537

Table 5
Random initial guess values were utilised for consistency analysis of parallel schemes.

$\mathbf{v}^{[*]}$	$\overset{[0]}{x_1}$		$\overset{[0]}{x_2}$		$\overset{[0]}{x_3}$		$\overset{[0]}{x_4}$		$\overset{[0]}{x_5}$	
	[Re,	Im]	[Re,	Im]	[Re,	Im]	[Re,	Im]	[Re,	Im]
v_1^*	[0.05,	0.87]	[0.52,	0.23]	[0.72,	0.27]	[0.27,	0.30]	[0.11,	0.20]
v_2^*	[0.03,	0.47]	[0.34,	0.74]	[0.45,	0.24]	[0.94,	0.05]	[0.03,	0.09]
v_3^*	[0.12,	0.52]	[0.73,	0.79]	[0.76,	0.78]	[0.06,	0.45]	[0.07,	0.40]

Table 6
Residual error outcome for engineering application 5.1 on random initial vectors.

Method	Ini-V	ČŎ	ČPŬ	ň	$\epsilon_1^{[\check{n}]}$	$\epsilon_2^{[\check{n}]}$	$\epsilon_3^{[\check{n}]}$	$\epsilon_4^{[\check{n}]}$	$\epsilon_5^{[\check{n}]}$	$\sigma_i^{[\check{n}-1]}$
ZPM	v_1^*	5	0.6902	35	$6.1e-11$	$1.3e-13$	$0.3e-10$	$9.0e-9$	$1.6e-9$	3.76777
MSP	v_1^*	5	0.4002	35	$0.1e-14$	$0.7e-12$	$0.2e-15$	$9.3e-13$	$0.3e-8$	4.23405
NDP	v_1^*	5	0.5015	35	$8.8e-18$	$1.6e-26$	$9.0e-23$	$2.2e-17$	$0.2e-15$	7.00035
ZPM	v_2^*	6	0.4512	35	$6.1e-16$	$2.5e-19$	$6.3e-20$	$8.2e-20$	$1.1e-16$	5.18977
MSP	v_2^*	6	0.4562	35	$0.1e-14$	$2.5e-18$	$0.2e-15$	$9.2e-13$	$7.3e-12$	4.16878
NDP	v_2^*	6	0.4542	35	$8.8e-18$	$2.5e-16$	$5.2e-23$	$1.9e-17$	$8.0e-11$	5.14545
ZPM	v_3^*	6	0.0742	17	$8.4e-31$	$7.3e-21$	$2.0e-36$	$5.0e-49$	$3.2e-39$	6.15464
MSP	v_3^*	6	0.0741	17	$1.8e-26$	$7.3e-29$	$2.0e-37$	$5.0e-29$	$5.4e-29$	6.15546
NDP	v_3^*	6	0.0745	17	$1.5e-23$	$7.3e-28$	$2.0e-28$	$5.0e-34$	$4.6e-45$	6.19424

Table 7
Overall performance of parallel techniques for solving engineering application 5.1.

Method	Max-Error$^{v^*}$	ČPŬ-time$^{v^*}$	P-Con$^{v^*}$	Average-It$^{v^*}$	Memory-U$^{v^*}$	$\sigma_i^{[\check{n}-1]}$
ZPM	$7.4e-11$	0.6453	14.76	6	35.4343	3.45373
MSP	$6.1e-16$	0.4512	17.87	5	25.654	5.18977
NDP	$1.5e-23$	0.0745	45.65	7	19.543	6.19424

5.1.1. Physical behaviour of the problem

The solution to the chemical reactequilibrium problem provides valuable physical findings, in particular:

- The equation has only one distinct positive real root in the physically acceptable range $0 < x < \frac{C_{A_0}}{2}$, implying that concentrations must be positive and satisfy the condition $C_B > 0$, and $C_C > 0$.
- Small adjustments to $K_1^{[*]}$, $K_2^{[*]}$, and C_{A_0} can have a major impact on root behaviour. A high $K_2^{[*]}$, for example, favours C production, causing it x to increase.
- These concentrations describe the equilibrium state of the reactor are $C_A = 3.3711101568$, $C_B = 0.3711015643$, and $C_C = 0.8271921$ for $\eta_1 = 2.025481564$ upto 9 decimal places.
- If R3 becomes significant, it will change equilibrium using Le Chatelier's principle, and we will modify the polynomial accordingly to better understand the reaction behaviour.

5.2. Fractional conversion [67]

The percentage of reactant that is transformed into products during a chemical reaction is known as fractional conversion. When it yields a proportion in relation to the initial quantity available, it is calculated. Time, temperature, pressure, or the presence of catalysts affect its value. Reactor design optimization, waste reduction, and product yield maximization all depend on fractional conversion monitoring. It frequently has to do with reactor volume and residence time for continuous operations. The economic viability of scaling up chemical reactions is also determined by the expression describe in [68] as:

$$g(x) = x^4 - 7.790750x^3 + 14.7445x^2 + 2.5110x - 1.67400 \tag{51}$$

is the fractional conversion of nitrogen, hydrogen feed at 250 atm. and 227k. The expression (51) shows how conversion efficiency varies with operating conditions and is crucial for optimizing reactor volume, resident time, and overall process yield. The exact roots of (51) are:

$$\zeta_1 = 3.9485 + 0.3161i, \zeta_2 = 3.9485 - 0.3161i, \zeta_3 = -0.3841, \zeta_4 = 0.2778 \tag{52}$$

and

$$\overset{[0]}{x}_1 = 3.5 + 0.3i,\ \overset{[0]}{x}_2 = 3.5 - 0.3i,\ \overset{[0]}{x}_3 = -0.3 + 0.01i,\ \overset{[0]}{x}_4 = 1.8 + 0.01i, \tag{53}$$

are chosen as initial guessed values. The results of NDP, ZPM and MSP in terms of numbers to solve (51) are shown in Table 8. In terms of computational convergence order (CO), computational CPU time, residual errors, error graph (Fig. 7) and iteration numbers, Table 8 clearly shows that the technique NDP is superior to ZPM, MSP.

The numerical results in Table 8 clearly show that in terms of residual error, the newly developed method NDP outperforms ZPM, MSP, with a high computational order of convergence and a rapid convergence rate. For clustered roots, where convergence analysis is challenging, this

Table 8
Residual error results for closely initialized values to exact roots of (51) using parallel techniques in engineering application 5.2.

Method	ČŎ	ČPŬ	ň	$\epsilon_1^{[\check{n}]}$	$\epsilon_2^{[\check{n}]}$	$\epsilon_3^{[\check{n}]}$	$\epsilon_4^{[\check{n}]}$	$\sigma_i^{[\check{n}-1]}$
ZPM	5	0.654	4	0.0	$3.5e-55$	$0.2e-35$	$1.2e-45$	4.655
MSP	6	0.047	4	$8.1e-62$	$2.5e-61$	$1.2e-90$	0.0	5.177
NDP	6	0.032	4	$1.4e-61$	0.0	0.0	$5.0e-149$	6.154

Table 9
Random initial guess values utilised for consistency analysis of parallel schemes.

$\mathbf{v}^{[*]}$	$x_1^{[0]}$		$x_2^{[0]}$		$x_3^{[0]}$		$x_4^{[0]}$	
	[Re,	Im]	[Re,	Im]	[Re,	Im]	[Re,	Im]
v_1^*	[0.01,	0.24]	[0.77,	0.43]	[0.72,	0.43]	[0.42,	0.70]
v_2^*	[0.15,	0.65]	[0.12,	0.70]	[0.67,	0.05]	[0.30,	0.66]
v_3^*	[0.12,	0.65]	[0.03,	0.09]	[0.40,	0.08]	[0.06,	0.40]

Table 10
Residual error outcome for engineering application 5.2 on random initial vectors.

Method	Ini-V	ČŎ	ČPŬ	ň	$\epsilon_1^{[\check{n}]}$	$\epsilon_2^{[\check{n}]}$	$\epsilon_3^{[\check{n}]}$	$\epsilon_4^{[\check{n}]}$	$\sigma_i^{[\check{n}-1]}$
ZPM	v_1^*	5	0.6135	41	$4.0e-10$	$3.3e-13$	$9.3e-5$	$0.2e-7$	4.00554
MSP	v_1^*	5	0.5231	41	$0.1e-11$	$0.5e-5$	$0.1e-11$	$3.1e-3$	4.76566
NDP	v_1^*	5	0.5523	41	$0.2e-7$	$1.1e-8$	$1.3e-2$	$9.0e-7$	5.85674
ZPM	v_2^*	6	0.4512	35	$6.1e-16$	$2.5e-19$	$6.3e-20$	$8.2e-20$	5.18977
MSP	v_2^*	6	0.4562	35	$0.1e-14$	$2.5e-18$	$0.2e-15$	$9.2e-13$	4.16878
NDP	v_2^*	6	0.4542	35	$8.8e-18$	$2.5e-16$	$5.2e-23$	$1.9e-17$	5.14545
ZPM	v_3^*	6	0.0742	17	$8.4e-31$	$7.3e-21$	$2.0e-36$	$5.0e-49$	6.15464
MSP	v_3^*	6	0.0741	17	$1.8e-26$	$7.3e-29$	$2.0e-37$	$5.0e-29$	6.15546
NDP	v_3^*	6	0.0745	17	$1.5e-37$	$7.3e-30$	$2.0e-28$	$5.0e-34$	6.19424

Table 11
Overall performance of parallel techniques for solving engineering application 5.2.

Method	Max-Error$^{v^*}$	ČPŬ-time$^{v^*}$	P-Con$^{v^*}$	Average-It$^{v^*}$	Memory-U$^{v^*}$	$\sigma_i^{[\check{n}-1]}$
ZPM	$0.5e-7$	0.5675	41.543	41	37.653	5.18977
MSP	$6.1e-16$	0.4512	35.764	35	45.654	5.18977
NDP	$1.5e-23$	0.0745	75.654	17	23.875	6.19424

dual approach is essential; hence, initial guesses are made at random to evaluate convergence behaviour. Using some random initial vectors, $v_1^* - v_4^*$, we can determine the global convergence behaviour. On these random initial vectors, NDP takes 17 iterations and achieves a maximum residual error of 7.3e-23 in 0.074 seconds on computers; ZPM takes 41 iterations and achieves a maximum residual error of 0.1e-5 consuming 0.6135 seconds on computer, while PP takes 35 iterations and achieves a maximum residual error of 2.1e-13 and consuming 0.451 seconds on computer. The results of the Table 9 clearly show that NDP outperforms ZPM, MSP and converges more globally. The random initial vectors $v^* = \left[x_1^{[0]}, x_2^{[0]}, x_3^{[0]}, x_4^{[0]}\right]$ are given in Table 9.

The outcomes of the parallel schemes utilizing the random initial vectors given in Table 9 are presented in Table 10.

In Table 10, Ini-V denotes the set of random initial vectors used to test the global convergence behaviour of the NDP, ZPM, MSP techniques. Based on random initial approximations, Table 10 displays the numerical results of the simultaneous schemes NDP and ZPM, MSP. The Table 10 results clearly show that, in terms of global convergence, the NDP technique performs better than the ZPM, MSP method.

Using random initial vectors, Table 11 displays the maximum error, computational time in seconds, percentage convergence, average number of iterations, and local computational order of convergence, respectively, as Max-Error$^{v^*}$, ČPŬ-time$^{v^*}$, P-Con$^{v^*}$, Average-It$^{v^*}$, and $\rho_i^{[\check{n}-1]}$. Table 11 and Fig. 7(a,c) clearly demonstrate that our technique NDP outperforms the existing methods ZPM, MSP in terms of maximum

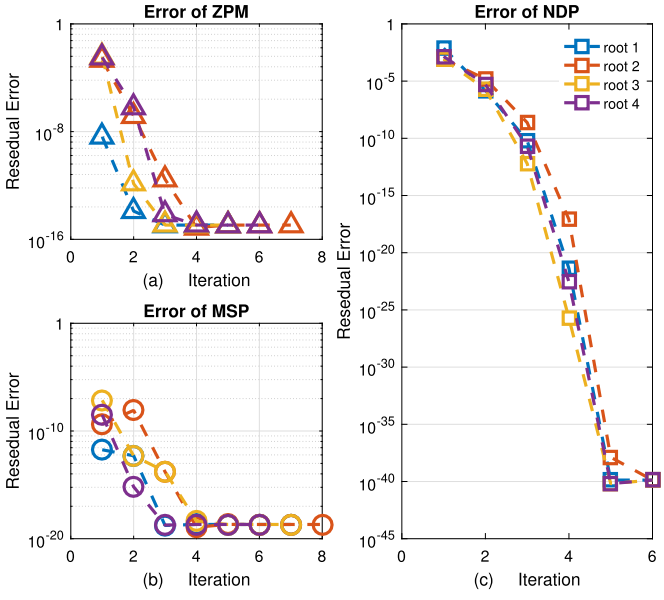


Fig. 7. (a-c), present the residual error graphs for the parallel schemes—ZPM, MSP, and NDP—that were utilized to solve engineering application 5.2.

Table 12
Residual error results for closely initialized values to exact roots of (55) using parallel techniques in engineering application 5.3.

Method	ČŎ	ČPŬ	ň	$\epsilon_1^{[\hbar]}$	$\epsilon_2^{[\hbar]}$	$\epsilon_3^{[\hbar]}$	$\epsilon_4^{[\hbar]}$	$\sigma_i^{[\hbar-1]}$
ZPM	5	0.140	7	$1.7e-3$	$9.1e-5$	$1.0e-4$	$2.8e-24$	4.943
MSP	6	0.078	5	$1.7e-12$	$9.1e-5$	$1.0e-4$	$2.8e-21$	5.342
NDP	6	0.032	5	$1.0e-17$	$1.0e-27$	$3.7e-27$	$1.5e-24$	5.932

Table 13
Random initial guess values utilised for consistency analysis of parallel schemes.

$\mathbf{v}^{[*]}$	$x_1^{[0]}$		$x_2^{[0]}$		$x_3^{[0]}$		$x_4^{[0]}$	
	[Re,	Im]	[Re,	Im]	[Re,	Im]	[Re,	Im]
v_1^*	[0.01,	0.73]	[0.64,	0.70]	[0.77,	0.69]	[0.92,	0.06]
v_2^*	[0.65,	0.54]	[0.64,	0.09]	[0.37,	0.78]	[0.86,	0.89]
v_3^*	[0.64,	0.23]	[0.65,	0.77]	[0.96,	0.40]	[0.78,	0.95]

error, computational time, percentage convergence, average number of iterations, and computational order of convergence.

5.2.1. Physical interpretation of the problem

The solution to the fractional conversion problem provides valuable physical insights, particularly:

- Solving Equation (51) aids in the identification of significant roots associated with conversion maxima or minima values, such as turning points and equilibrium concentrations.
- The equation's multi-root and nonlinear nature makes it a benchmark for testing iterative solvers, including fractional calculus and higher-order scheme-based ones. This allows for advanced numerical techniques.
- Analyzing solution behaviour improves process optimization by minimizing yield, reducing costs, and increasing overall chemical efficiency.

5.3. Chemical rector-chemical engineering problem [69]

Designed to maximize the conversion of reactants into products, a chemical reactor is a system or vessel where chemical reactions occur. In order to increase efficiency, it regulates variables like temperature, pressure, and mixing, which is crucial in sectors like energy, petrochemicals, and medicines. Various reactor types, including plug flow, continuous stirred-tank, and batch reactors, are employed according to the needs of the reaction. Industrial process yield, safety, and economic viability are all directly impacted by reactor design and operation. They are therefore necessary for the large-scale synthesis of chemicals which rise the following nonlinear equations as:

$$\hat{\mathrm{H}}_c \frac{2.98\,(x+2.25)}{(x+1.45)\,(x+2.85)^2\,(x+4.35)} = -1. \tag{54}$$

The expression (54) is derived from the stability condition of a proportional controller that regulates the transfer function of a chemical reactor. The variable x denotes a complex frequency (eigenvalue) and $\hat{H}_c$ is the controller's gain. In order to guarantee stable reactor system operation, the objective is to determine values of $\hat{H}_c$ that guarantee all of the system's poles lie in the left-half complex plane.

Setting $\hat{H}_c = 0$ simplifies the transfer function, and the associated characteristic polynomial takes the following form:

$$g(x) = x^4 + 11.50x^3 + 47.49x^2 + 83.06325x + 51.23266875 = 0. \tag{55}$$

The uncontrolled dynamic behaviour of the reactor system is shown by the nonlinear equations (55), where the roots stand for the natural modes of the system. Analysing these roots enables us to determine whether the unmanaged process will remain stable or require feedback control.

$$\zeta_1 = -1.45, \zeta_2 = -2.85, \zeta_3 = -2.85, \zeta_4 = -4.45. \tag{56}$$

These roots are known as poles of the open-loop transfer function. We take the exact root $\zeta = -4.35$ and

$$x_1^{[0]} = -1.0,\ x_2^{[0]} = -2.1,\ x_3^{[0]} = -1.8,\ x_4^{[0]} = -3.9 \tag{57}$$

are chosen as initial estimates for simultaneous determination of all roots of (55). Results of the NDP, ZPM, and MSP in terms of numbers for solving (55) are shown in Table 12. In terms of computational convergence order (CO), error graph (Fig. 5), computational CPU time, residual errors, and iterations numbers, Table 12 clearly shows that the technique NDP is superior to ZPM, MSP.

The numerical results in Table 12 clearly show that in terms of residual error, the newly developed method outperforms ZPM, MSP, with a high computational order of convergence and a rapid convergence rate. Since clustered roots make convergence analysis hard, this dual technique is essential. To evaluate convergence behaviour, random initial guesses are used. Using some random initial vectors, $v_1^* - v_4^*$, we can determine the global convergence behaviour. On these random initial vectors, NDP takes 13 iterations and achieves a maximum residual error of 7.3e-25 in 0.0714 seconds on computers, ZPM takes 7 iterations and achieves a maximum residual error of 2.1e-3 consuming 0.140 seconds on computer, while MSP takes 31 iterations and achieves a maximum residual error of 2.1e-17 consuming 0.481 seconds on computer. The results of the Table 12 clearly show that NDP outperforms ZPM, MSP and converges more efficiently. The random initial vectors $v^* = [x_1^{[0]}, x_2^{[0]}, x_3^{[0]}, x_4^{[0]}]$ are given in Table 13.

The outcomes of the parallel schemes utilizing the random initial vectors given in Table 13 are presented in Table 14.

In Table 14, Ini-V denotes the set of random initial vectors used to test the global convergence behaviour of the NDP, ZPM, and MSP techniques. The Table 14 results clearly show that, in terms of global convergence, the NDP technique performs better than the ZPM, MSP methods.

Table 15 and Fig. 8(a,c) clearly demonstrate that our technique outperforms the existing method ZPM, MSP in terms of maximum error, computational time, percentage convergence, average number of iterations, and computational order of convergence.

5.3.1. Physical interpretation of the problem

- In a chemical reactor operating in steady-state, where the rates of inward and outward reactions are identical, the equation is based.

Table 14
Residual error outcome for engineering application 5.3 on random initial vectors.

Method	Ini-V	ČÕ	ČPŬ	ň	$\epsilon_1^{[\tilde{n}]}$	$\epsilon_2^{[\tilde{n}]}$	$\epsilon_3^{[\tilde{n}]}$	$\epsilon_4^{[\tilde{n}]}$	$\sigma_i^{[\tilde{n}-1]}$
ZPM	v_1^*	5	0.6005	43	$3.3e-7$	$2.5e-7$	$0.3e-9$	$0.2e-8$	4.11334
MSP	v_1^*	5	0.7845	43	$0.1e-11$	$8.5e-9$	$0.2e-11$	$1.2e-5$	4.53356
NDP	v_1^*	5	0.6057	43	$4.0e-10$	$9.5e-8$	$3.3e-7$	$1.0e-7$	3.99344
ZPM	v_2^*	6	0.4825	31	$7.1e-16$	$3.5e-17$	$6.3e-20$	$5.2e-20$	5.44977
MSP	v_2^*	6	0.4824	31	$4.1e-14$	$3.6e-18$	$0.2e-17$	$9.2e-19$	5.13378
NDP	v_2^*	6	0.4825	31	$8.8e-18$	$7.5e-16$	$5.2e-27$	$1.1e-37$	5.10045
ZPM	v_3^*	6	0.0712	13	$8.4e-31$	$4.5e-22$	$2.9e-38$	$0.1e-49$	6.15464
MSP	v_3^*	6	0.0711	13	$1.8e-26$	$0.3e-32$	$9.0e-37$	$8.0e-39$	6.04546
NDP	v_3^*	6	0.0715	13	$9.5e-35$	$7.3e-34$	$2.0e-38$	$9.0e-39$	6.79424

Table 15
Overall performance of parallel techniques for solving engineering application 5.3.

Method	Max-Error$^{v^*}$	ČPŬ-time$^{v^*}$	P-Con$^{v^*}$	Average-It$^{v^*}$	Memory-U$^{v^*}$	$\sigma_i^{[\tilde{n}-1]}$
ZPM	$7.0e-7$	0.5436	43.5424	43	46.67844	4.18977
MSP	$9.9e-16$	0.4512	35.6353	35	23.45746	5.76498
NDP	$1.5e-23$	0.0745	57.6537	17	17.89553	6.19984

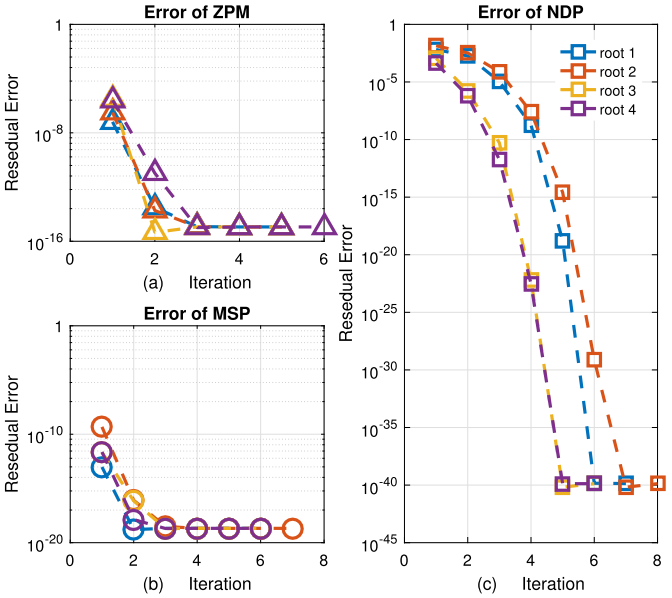


Fig. 8. (a-c), shows the residual error graphs for the parallel schemes—ZPM, MSP, and NDP—that were utilized to solve engineering application 5.3.

- The variable x is a measure of the reactor's reaction to changes in process variables and can be concentration, temperature, or rate of conversion, depending on the situation.
- The denominator's cubic and quartic terms indicate that there could be more than one operational point, or root, which could be either a stable or unstable steady state.
- Chemical engineers can use the physical importance encoded in the equation to construct heat exchangers, feedback controls, and catalyst combinations.

5.4. Aerospace optimization–lift-to-drag ratio maximization: mechanical engineering problem [70]

Aerodynamic performance and energy efficiency of flying vehicles are maximised at critical stages of flight such as atmospheric re-entry, cruising, or gliding by minimising the lift-to-drag ratio (L/D) in aerospace engineering. The lift-to-drag ratio measures an aircraft's or spacecraft's ability to convert aerodynamic lift to forward motion while minimising drag-induced resistance. An increased L/D ratio indicates that the vehicle travels a longer horizontal distance per unit of altitude loss, which is especially beneficial for re-entry vehicles, gliders, and long-range aircraft. The optimization problem entails expressing lift coefficient (C_L) as nonlinear polynomials in terms of angle of attack (x), which captures the complex aerodynamics of the vehicle's shape and flight conditions. In general, C_L is an odd-degree polynomial (up to 5th degree), while C_D is an even-degree polynomial. To get the ideal angle of attack for the maximum L/D ratio, the engineers apply the first-order condition of the L/D formula and calculate the resulting rational function, which has a sixth-degree polynomial derivative. The equation's solution yields the optimal aerodynamic angle candidates. Physically, the answer is to find an equilibrium point where the vehicle is in the most efficient aerodynamic attitude, with the least amount of energy loss and thermal loading, and the most range and controllability. For both manned and unmanned space missions, this optimization is critical for maximizing mission success, safety margins, and fuel efficiency.

The lift coefficient C_L and drag coefficient C_D can be fitted as polynomial functions of the angle of attack (x) in radians or degrees.

$$C_L(x) = 0.1x + 0.05x^3, \tag{58}$$

$$C_D(x) = 0.02 + 0.04x^2 + 0.01x^4. \tag{59}$$

These are odd-degree (3, 5) and even-degree (2, 4) polynomials, respectively, due to the underlying physics:

- Lift is an odd function of x, changing sign with negative angles.
- Drag is an even function of x (symmetric around zero).

We intend to maximize:

$$f(x) = \frac{C_L(x)}{C_D(x)}. \tag{60}$$

This represents a nonlinear fractional function $f(x)$. In order to maximize it:

$$\frac{d}{dx}\left(\frac{C_L(x)}{C_D(x)}\right) = 0 \Rightarrow C_D\frac{dC_L}{dx} - C_L\frac{dC_D}{dx} = 0. \tag{61}$$

This leads to:

$$g(x) = -0.0005x^6 - 0.001x^4 - 0.001x^2 + 0.002. \tag{62}$$

Solving (62) yields the best angle(s) of attack x for balancing lift and drag most efficiently. These solutions describe equilibrium aerodynamic attitudes that minimise energy loss, heat stress, and maximise range—all of which are important considerations for both manned and unmanned space missions. The nonlinear character and high degree of this equation highlight the importance of strong numerical root-finding methods,

Table 16
Residual error results for closely initialized values to exact roots of (62) using parallel techniques in engineering application 5.4.

Method	ČÕ	ČPŬ	ň	$\epsilon_1^{[\tilde{n}]}$	$\epsilon_2^{[\tilde{n}]}$	$\epsilon_3^{[\tilde{n}]}$	$\epsilon_4^{[\tilde{n}]}$	$\epsilon_5^{[\tilde{n}]}$	$\epsilon_6^{[\tilde{n}]}$	$\sigma_i^{[\tilde{n}-1]}$
ZPM	5	0.2352	5	$3.0e-85$	$1.8e-16$	0.0	0.0	$1.8e-86$	0.0	4.54
MSP	6	0.043	3	$3.0e-85$	$1.8e-56$	0.0	0.0	$1.8e-86$	0.0	5.14
NDP	6	0.031	3	$3.0e-132$	0.0	0.0	0.0	$1.8e-86$	0.0	6.55

Table 17
Random initial guess values utilised for consistency analysis of parallel schemes.

$\mathbf{v}^{[*]}$	$x_1^{[0]}$		$x_2^{[0]}$		$x_3^{[0]}$		$[\cdots,\cdots]$	$x_6^{[0]}$	
	[Re,	Im]	[Re,	Im]	[Re,	Im]	$[\cdots,\cdots]$	[Re,	Im]
υ_1^*	[0.65,	0.13]	[0.76,	0.43]	[0.42,	0.65]	$[\cdots,\cdots]$	[0.95,	0.09]
υ_2^*	[0.85,	0.23]	[0.34,	0.04]	[0.38,	0.85]	$[\cdots,\cdots]$	[0.36,	0.35]
υ_3^*	[0.24,	0.26]	[0.23,	0.07]	[0.67,	0.64]	$[\cdots,\cdots]$	[0.85,	0.98]

Table 18
Residual error outcome for engineering application 5.4 on random initial vectors.

Method	Ini-V	ČPŬ	$\epsilon_1^{[\tilde{n}]}$	$\epsilon_2^{[\tilde{n}]}$	$\epsilon_3^{[\tilde{n}]}$	$\epsilon_4^{[\tilde{n}]}$	$\epsilon_5^{[\tilde{n}]}$	$\epsilon_6^{[\tilde{n}]}$	$\sigma_i^{[\tilde{n}-1]}$
ZPM	υ_1^*	0.87	$7.1e-6$	$7.1e-6$	$7.1e-10$	$2.5e-7$	$6.9e-2$	$8.9e-11$	6.35
MSP	υ_1^*	0.54	$2.1e-8$	$2.1e-8$	$2.1e-11$	$2.5e-7$	$7.2e-7$	$8.2e-9$	6.06
NDP	υ_1^*	0.35	$8.6e-7$	$8.6e-1$	$8.6e-7$	$2.5e-6$	$5.2e-7$	$1.9e-7$	7.17
ZPM	υ_2^*	0.48	$7.1e-16$	$7.1e-6$	$7.1e-16$	$2.5e-7$	$6.9e-2$	$8.9e-20$	5.14
MSP	υ_2^*	0.05	$2.1e-18$	$2.1e-8$	$2.1e-18$	$2.5e-7$	$7.2e-7$	$8.2e-19$	4.16
NDP	υ_2^*	0.55	$8.6e-17$	$8.6e-1$	$8.6e-17$	$2.5e-6$	$5.2e-2$	$1.9e-37$	6.17
ZPM	υ_3^*	0.07	$1.5e-31$	$1.5e-3$	$1.5e-31$	$7.3e-3$	$2.6e-8$	$5.0e-19$	6.04
MSP	υ_3^*	0.07	$0.7e-29$	$0.7e-9$	$0.7e-29$	$7.3e-7$	$6.0e-8$	$5.0e-29$	6.15
NDP	υ_3^*	0.07	$0.5e-35$	$0.5e-8$	$0.5e-25$	$7.3e-19$	$2.0e-9$	$5.0e-24$	6.16

Table 19
Overall performance of parallel techniques for solving engineering application 5.4.

Method	Max-Error$^{\upsilon^*}$	ČPŬ-time$^{\upsilon^*}$	P-Con$^{\upsilon^*}$	Average-It$^{\upsilon^*}$	Memory-U$^{\upsilon^*}$	$\sigma_i^{[\tilde{n}-1]}$
ZPM	$2.1e-7$	0.4512	35.7646	17	56.87574	5.089
MSP	$0.5e-11$	0.4512	35.7644	11	41.65753	5.179
NDP	$1.5e-23$	0.0745	17.546	7	27.87563	6.104

particularly those suitable for multiple and clustered roots, as explained in the proposed parallel iterative technique. The exact roots of (62) are:

$$\zeta_{1,2} = \pm 0.9387,\ \zeta_{3,4} = -0.5873 \pm 1.3362i,\ \zeta_{5,6} = 0.5873 \pm 1.3362i. \quad (63)$$

The initial estimates have been taken as:

$$x_{1,2}^{[0]} = \pm 0.9, x_{3,4}^{[0]} = -0.5 - 1.3i, x_{5,6}^{[0]} = 0.6 \pm 0.6i. \quad (64)$$

Results of the NDP and MM in terms of numbers for solving (62) are shown in Table 8. In terms of computational convergence order (CO), error computational CPU time, residual errors, and iterations numbers, Table 16 clearly shows that the technique NDP is superior to ZPM, MSP.

The numerical results in Table 16 clearly show that in terms of residual error, the newly developed method outperforms ZPM, MSP, with a high computational order of convergence and a rapid convergence rate. Thus, parallel techniques are sometimes required for clustered roots, where convergence analysis is challenging; therefore, random initial guesses are used to examine convergence behaviour. Using some random initial vectors, $\upsilon_1^* - \upsilon_6^*$, we can determine the global convergence behaviour. On these random initial vectors, NDP takes 19 iterations and achieves a maximum residual error of 7.3e-25 in 0.0714 seconds on computers, ZPM takes 17 iterations and achieves a maximum residual error of 3.0e-15 consuming 0.87 seconds on computer, while MSP takes 37 iterations and achieves a maximum residual error of 2.1e-17 consuming 0.481 seconds on computers. The results of Table 16 clearly show that NDP outperforms ZPM, MSP and converges more consistently. The random initial vectors $\upsilon^* = [x_1^{[0]}, x_2^{[0]}, x_3^{[0]}, x_4^{[0]}, x_5^{[0]}, x_6^{[0]}]$ are given in Table 17.

The outcomes of the parallel schemes utilizing the random initial vectors given in Table 17 are presented in Table 18.

In Table 18, Ini-V denotes the set of random initial vectors used to test the global convergence behaviour of the NDP, ZPM and MSP techniques. The Table 18's results clearly show that, in terms of global convergence, the NDP technique performs better than the ZPM, MSP methods.

Table 19 and Fig. 9(a,c) clearly demonstrate that our technique outperforms the existing methods ZPM, PM, and MSP in terms of maximum error, computational time, percentage convergence, average number of iterations, and computational order of convergence.

5.4.1. Physical interpretation of the problem

- The real root of the polynomial determines the angle of attack ($x = \pm 0.9$) that maximizes the lift-to-drag ratio. The vehicle is orientated in the most aerodynamically efficient manner at this angle.
- The most appropriate point is frequently located within a stable flying envelope, making it both safe and controlled. Roots outside of this range may be capable of causing aerodynamic instability or structural stress.
- At optimum x, the vehicle achieves maximum lift with minimal drag. This results in greater gliding ranges, lower fuel consumption, and improved thermal management during flight.
- The optimal point of L/D in flight reduces energy loss due to drag, allowing for increased cruising or mission endurance.

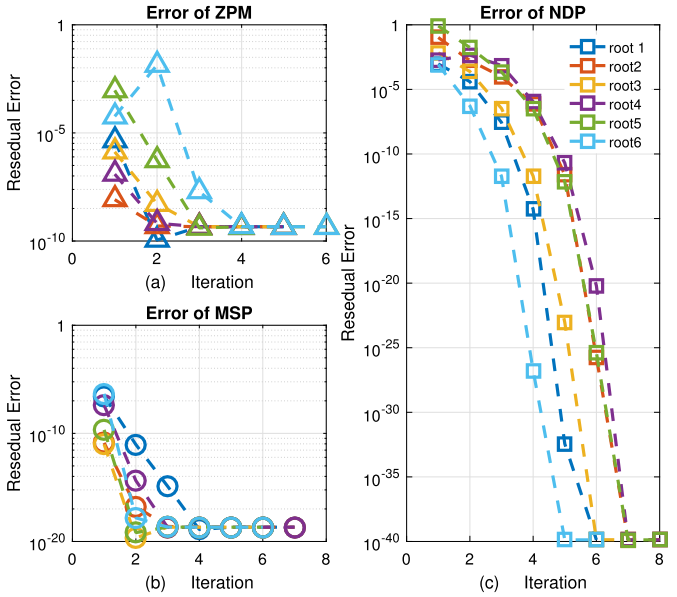


Fig. 9. (a-c), illustrate the residual error graphs for the parallel schemes—ZPM, MSP, and NDP—that were utilized to solve engineering application 5.4.

- Controlled landing at an angle allows for slow heat dissipation while also preventing overheating, protecting the vehicle and components.
- Contributes to the optimization of wing and control surface design in order to maintain high L/D values under varying flight conditions.
- The polynomial can have more than one root; only physically valid real values inside the operating parameters are used. The remaining complex roots could be for non-physical, unstable, or infeasible settings.

5.5. Application in differential equations [71]

Fractional differential equations (FDEs) have the potential to be powerful, particularly in science and engineering applications, because they establish a broad framework that expands classical calculus to include memory and hereditary features. They are very valuable in engineering for simulating viscoelastic materials and control systems, as well as signal processing. FDEs are widely used in science to reveal anomalous diffusion processes caused by biological systems and porous media. This provides them with useful information in long-range interaction systems, such as quantum physics and statistical mechanics. Because of their adaptability, FDEs are indispensable in real-world issues that involve complicated, non-local, and time-dependent phenomena, given as:

$$\begin{cases} \frac{d^{\frac{5.0}{2.0}} g(x)}{dx^{\frac{5.0}{2.0}}} + x \frac{d^{\frac{190}{10.0}} g(x)}{dx^{\frac{19.0}{100}}} + (g(x))^3 = k^{[*]}, \\ g(0) = 0, \\ g'(0) = 0, \\ g''(0) = 0, \end{cases} \tag{65}$$

where $k^{[*]} = 6.770275002x^{0.5} + 5.73347578x^{2.8} + x^3 + x$ and $0 \le x \le 1$. Using the technique defined in [72], the following polynomial is used to simulate (65) as given:

$$g(x) = x^3 + 3x^2 + 4x - 0.00000105. \tag{66}$$

It satisfies the initial conditions and effectively simulates the system's behaviour over the specified region. The exact roots of the nonlinear problem (66) are as follows:

Table 20
Residual error results for closely initialized values to exact roots of (66) using parallel techniques in engineering application 5.5.

Method	ČČ	ČPŮ	ň	$\epsilon_1^{[\hbar]}$	$\epsilon_2^{[\hbar]}$	$\epsilon_3^{[\hbar]}$	$\sigma_i^{[\hbar-1]}$
ZPM	5	0.078	8	$1.7e-32$	0.0	$1.0e-44$	4.924
MSP	6	0.078	5	$1.7e-72$	0.0	$1.0e-44$	5.955
NDP	6	0.032	4	$1.0e-157$	0.0	0.0	6.124

Table 21
Random initial guess values were utilised for consistency analysis of parallel schemes.

$\mathbf{v}^{[*]}$	$x_1^{[0]}$		$x_2^{[0]}$		$x_3^{[0]}$	
	[Re,	Im]	[Re,	Im]	[Re,	Im]
v_1^*	[0.05,	0.06]	[0.10,	0.03]	[0.02,	0.60]
v_2^*	[0.03,	0.03]	[0.04,	0.04]	[0.30,	0.80]
v_3^*	[0.02,	0.02]	[0.03,	0.70]	[0.06,	0.08]

$$\zeta_1 = -0.000002625005, \zeta_2 = -1.49998 \pm 1.32287i. \tag{67}$$

The initial estimates have been taken as:

$$x_1^{[0]} = -1.0, \; x_{2,3}^{[0]} = -1.3 \pm 1.1i, \tag{68}$$

for simultaneous determination of all roots of (66). In terms of computational convergence order (CO), error graph, computational CPU time, residual errors, and iteration numbers, Table 20 clearly shows that the NDP technique is superior to ZPM and MSP.

The numerical results in Table 20 clearly show that in terms of residual error, the newly developed method outperforms ZPM, MSP, with a high computational order of convergence and a rapid convergence rate. To evaluate convergence behaviour, random initial guesses are used. This parallel technique is sometimes crucial for clustered roots, where convergence analysis is hard to achieve. Using some random initial vectors, $v_1^* - v_3^*$, we can determine the global convergence behaviour. On these random initial vectors, NDP takes 13 iterations and achieves a maximum residual error of 7.3e-25 in 0.0714 seconds on computers; ZPM takes 23 iterations and achieves a maximum residual error of 6.3e-4 consuming 0.1755 seconds on computers; while MSP takes 31 iterations and achieves a maximum residual error of 2.1e-17, consuming 0.481 seconds on computers. The results of the Table 20 clearly show that NDP outperforms MSP and ZPM and converges more efficiently. The random initial vectors $v^* = [x_1^{[0]}, x_2^{[0]}, x_3^{[0]}]$ are given in Table 21.

The results of the parallel schemes using the random initial vectors given in Table 21 are presented in Table 22.

In Table 22, Ini-V denotes the set of random initial vectors used to test the global convergence behaviour of the NDP, ZPM, and MSP techniques. The Table 22 results clearly show that, in terms of global convergence, the NDP technique performs better than the ZPM, MSP methods.

Table 23 and Fig. 10(a,c) clearly demonstrate that our technique outperforms the existing methods ZPM and MSP in terms of maximum error, computational time, percentage convergence, average number of iterations, and computational order of convergence.

5.5.1. Physical interpretation of the problem

The solution to the chemical reactor problem provides useful physical insights, more precisely:

- The inclusion of fractional-order derivatives suggests that the system is memory-held. The system's current state is determined not only by its current input but also by its previous states, which is prevalent in viscoelastic, biological, and diffusive systems.
- Chemical engineers can build feedback controls, heat exchangers, and catalyst arrangements based on the physical interpretation of the equation.

Table 22
Residual error outcome for engineering application 5.5 on random initial vectors.

Method	Ini-V	ČČ	ČPŬ	ň	$\epsilon_1^{[\check{n}]}$	$\epsilon_2^{[\check{n}]}$	$\epsilon_3^{[\check{n}]}$	$\sigma_i^{[\check{n}-1]}$
ZPM	υ_1^*	5	0.1755	23	$7.1e-6$	$2.5e-7$	$6.3e-4$	3.24005
MSP	υ_1^*	5	0.4004	22	$4.1e-4$	$2.5e-9$	$0.2e-5$	4.00489
NDP	υ_1^*	5	0.7654	21	$8.8e-8$	$1.5e-10$	$5.2e-7$	3.87574
ZPM	υ_2^*	6	0.2353	23	$7.1e-16$	$2.5e-13$	$6.3e-20$	5.66657
MSP	υ_2^*	6	0.2350	22	$4.1e-14$	$2.5e-11$	$0.2e-17$	4.98678
NDP	υ_2^*	6	0.4025	21	$8.8e-18$	$1.5e-10$	$5.2e-27$	6.15335
ZPM	υ_3^*	6	0.0712	10	$5.4e-31$	$1.3e-42$	$2.9e-38$	6.19865
MSP	υ_3^*	6	0.0711	11	$1.3e-26$	$0.3e-34$	$9.0e-37$	6.10006
NDP	υ_3^*	6	0.0715	10	$6.5e-28$	$1.1e-37$	$2.0e-29$	6.10024

Table 23
Overall performance of parallel techniques for solving engineering application 5.5.

Method	Max-Error$^{\upsilon^*}$	ČPŬ-time$^{\upsilon^*}$	P-Con$^{\upsilon^*}$	Average-It$^{\upsilon^*}$	Memory-U$^{\upsilon^*}$	$\sigma_i^{[\check{n}-1]}$
ZPM	$0.5e-7$	0.76474	45.763	23	23.657	4.76474
MSP	$9.9e-16$	0.46765	65.764	10	17.653	5.18977
NDP	$1.5e-23$	0.07454	77.857	7	7.8746	6.99424

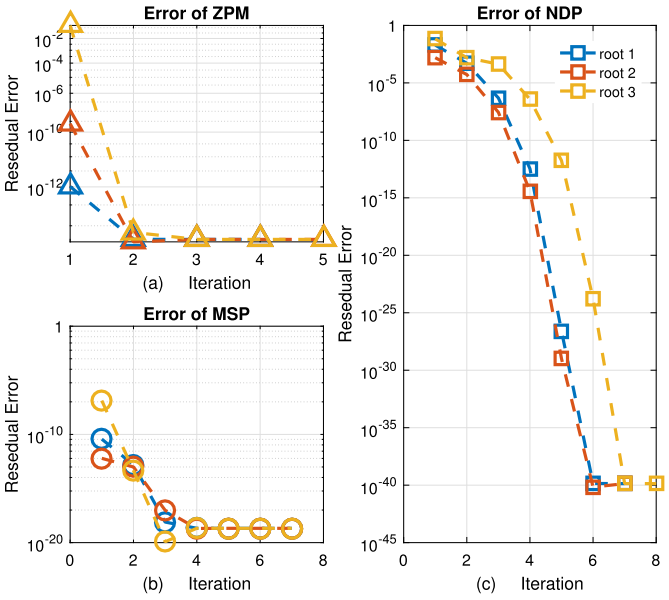


Fig. 10. (a-c), present the residual error graphs of parallel systems (ZPM, MSP, and NDP) utilised to solve engineering application 5.5.

- Chemical engineers can build feedback controls, heat exchangers, and catalyst arrangements based on the physical interpretation of the equation.
- Hereditary attributes are captured via fractional-order terms, which are crucial in biological tissues and slow-reaction materials.
- The model's ability to capture behaviour within a finite physical or temporal range, representative of spatially bounded diffusion, reaction, or signal spread problems, is indicated by the $x \in [0,1]$ domain.

5.6. Nonlinear Schrödinger partial differential equations [73]

Nonlinear Schrödinger Equations (NLSE) are a type of engineering equation that describes wave propagation in nonlinear dispersive media. In models, these are significantly involved in determining optical solitons, which may be critical to the successful development of long-distance fibre optic communication networks. NLSE aids in the description of Bose-Einstein condensate applications in quantum engineering, as well as the dynamics of waves modelling water in the context of developing resistant coastal and offshore constructions. Furthermore, NLSE reveals characteristics of nonlinear acoustics that aid in noise control and ultrasonic imaging technology. By describing wave interactions in plasma physics, they aid in the creation of fusion energy systems. New technologies like optical switches and high-power lasers are made possible by NLSE, which are a component of nonlinear optics. In addition, they contribute to the study of erratic waves, which is extremely beneficial to maritime safety and structural strength. Simulations of nonlinear wave phenomena can be used by NLSE to optimize engineering systems. Due to its versatility, the NLSE is commonly used to simulate nonlinear wave phenomena and optimize complicated engineering systems. The NLSE can be expressed in the following general form:

$$\begin{cases} i\frac{\partial \phi}{\partial t} + \nabla^2 \phi + |\phi|^2 \phi = 0. \end{cases} \tag{69}$$

Assume $\phi(x,t) = g(x)\exp(-i\varpi t)$, then gives

$$\begin{cases} i\frac{d^2 g}{dx^2} + + |g|^2 g = 0. \end{cases} \tag{70}$$

For certain boundary condition, we simulate (70) by the following nonlinear polynomial as:

$$g(x) = A_1^{[*]}x^3 + A_2^{[*]}x^2 + A_3^{[*]}. \tag{71}$$

Using $A_1^{[*]} = 1$, $A_2^{[*]} = -3$ and $A_3^{[*]} = 4$, we have

$$g(x) = x^3 - 3x + 4, \tag{72}$$

or

$$g(x) = (x-2)^2(x+1). \tag{73}$$

Using a complex structure with multiple roots, the nonlinear model (73) provides a realistic test case to evaluate the precision and stability of root-finding algorithms such as the proposed NDP approach. Such examples bridge the gap between theoretical advances and real-world engineering applications. The exact roots of the nonlinear problem (73) are as follows

$$\zeta_{1,2} = 2, 2, \zeta_3 = -1, \tag{74}$$

with multiplicity 2 and 1, respectively.

$$x_1^{[0]} = 3.0,\ x_2^{[0]} = 2.3,\ x_3^{[0]} = 5.1, \tag{75}$$

are chosen as initial estimates for simultaneous determination of all roots of (72). Results of the NDP, ZPM, and MSP in terms of numbers for solving (72) are shown in Tables 15-16. In terms of computational convergence order (CO), error graph (Fig. 11), computational CPU time,

Table 24
Residual error results for closely initialized values to exact roots of (72) using parallel techniques in engineering application 5.6.

Method	ČŎ	ČPŬ	ň	$\epsilon_1^{[\check{n}]}$	$\epsilon_2^{[\check{n}]}$	$\epsilon_3^{[\check{n}]}$	$\sigma_i^{[\check{n}-1]}$
ZPM	5	0.657	7	0.0	$1.5e-35$	$1.0e-12$	3.953
MSP	6	0.185	5	$1.7e-64$	0.0	$1.0e-63$	4.953
NDP	6	0.007	5	$7.8e-99$	0.0	0.0	6.162

Table 25
Results of the residual error on closed initial values to exact roots for the parallel techniques in engineering application 5.6.

Method	ČŎ	ČPŬ	ň	$\epsilon_1^{[\check{n}]}$	$\epsilon_2^{[\check{n}]}$	$\epsilon_3^{[\check{n}]}$	$\sigma_i^{[\check{n}-1]}$
ZPM	5	0.934	8	0.0	$1.5e-45$	$1.0e-12$	4.987
MSP	6	0.265	7	$0.7e-33$	$1.5e-25$	$1.0e-75$	5.765
NDP	6	0.007	4	$7.8e-127$	0.0	0.0	6.162

Table 26
Random initial guess values utilised for consistency analysis of parallel schemes.

$\mathbf{v}^{[*]}$	$x_1^{[0]}$		$x_2^{[0]}$		$x_3^{[0]}$	
	[Re,	Im]	[Re,	Im]	[Re,	Im]
v_1^*	[0.11,	0.06]	[0.98,	0.57]	[0.12,	0.03]
v_2^*	[0.03,	0.11]	[0.90,	0.04]	[0.37,	0.05]
v_3^*	[0.56,	0.52]	[0.24,	0.70]	[0.66,	0.48]

residual errors, and iteration numbers, Tables 24-25 clearly shows that the technique NDP is superior to ZPM, MSP.

The numerical outcomes for multiple roots are presented in Table 24.

The numerical results in Table 24-25 clearly show that in terms of residual error, the newly developed method outperforms ZPM, MSP, with a high computational order of convergence and a rapid convergence rate. To evaluate convergence behaviour, random initial guesses are used. This parallel technique is sometimes crucial for clustered roots, where convergence analysis is hard to achieve. Using some random initial vectors, $v_1^* - v_3^*$, we can determine the global convergence behaviour. On these random initial vectors, NDP takes 13 iterations and achieves a maximum residual error of 7.3e-25 in 0.0714 seconds on computers, ZPM takes 28 iterations and achieves a maximum residual error of 0.1e-10 consuming 0.6551 seconds on computer, while MSP takes 31 iterations and achieves a maximum residual error of 2.1e-17 consuming 0.481 seconds on computer. The results of the Table 24-25 clearly show that NDP outperforms ZPM, MSP and converges more globally. The random initial vectors $v^* = [x_1^{[0]}, x_2^{[0]}, x_3^{[0]}]$ are given in Table 26.

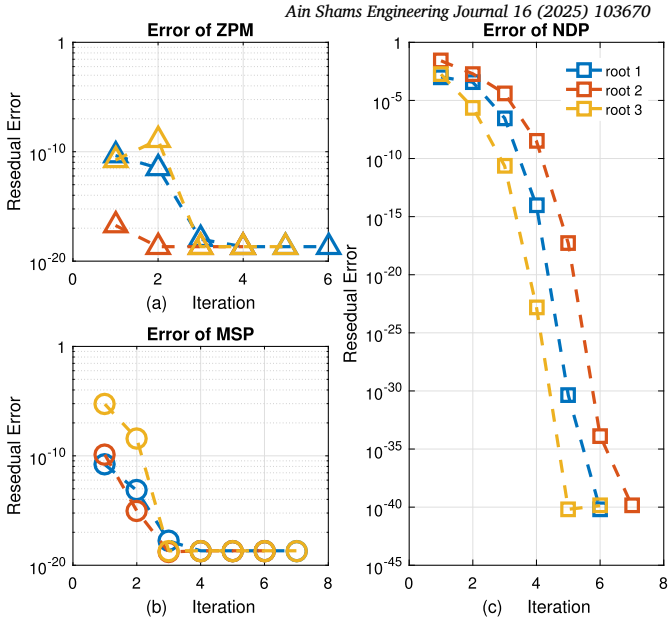


Fig. 11. (a-c), shows the residual error graphs for the parallel schemes—ZPM, MSP, and NDP—that were utilized to solve engineering application 5.6.

The outcomes of the parallel schemes utilizing the random initial vectors given in Table 26 are presented in Table 27.

In Table 27, Ini-V denotes the set of random initial vectors used to test the global convergence behaviour of the NDP, ZPM, and MSP techniques. The Table 27 results clearly show that, in terms of global convergence, the NDP technique performs better than the ZPM, MSP methods.

Table 28 and Fig. 11(a,c) clearly demonstrate that our technique outperforms the existing method PP in terms of maximum error, computational time, percentage convergence, average number of iterations, and computational order of convergence.

5.6.1. *Physical interpretation of the problem*

The solution to the nonlinear Schrödinger partial differential equation provides useful physical insights, particularly in the following aspects:

Table 27
Residual error outcome for engineering application 5.6 on random initial vectors.

Method	Ini-V	ČŎ	ČPŬ	ň	$\epsilon_1^{[\check{n}]}$	$\epsilon_2^{[\check{n}]}$	$\epsilon_3^{[\check{n}]}$	$\sigma_i^{[\check{n}-1]}$
ZPM	v_1^*	5	0.6546	28	$8.1e-19$	$7.9e-6$	$5.3e-13$	5.46777
MSP	v_1^*	5	0.7658	27	$0.1e-10$	$5.7e-11$	$5.9e-9$	4.18678
NDP	v_1^*	5	0.8758	27	$5.0e-18$	$7.9e-10$	$7.9e-11$	6.18678
ZPM	v_2^*	6	0.6435	23	$7.1e-16$	$2.5e-16$	$6.3e-26$	5.46777
MSP	v_2^*	6	0.7534	22	$4.1e-14$	$2.8e-11$	$0.2e-19$	4.18678
NDP	v_2^*	6	0.6545	21	$8.8e-18$	$1.8e-10$	$5.2e-21$	6.18678
ZPM	v_3^*	6	0.5323	10	$5.4e-31$	$1.0e-47$	0.0	6.17656
MSP	v_3^*	6	0.2532	11	$1.3e-26$	0.0	$9.0e-36$	6.54354
NDP	v_3^*	6	0.2453	10	$6.5e-29$	$1.0e-37$	$2.0e-39$	5.93364

Table 28
Overall performance of parallel techniques for solving engineering application 5.6.

Method	Max-Error$^{v^*}$	ČPŬ-time$^{v^*}$	P-Con$^{v^*}$	Average-It$^{v^*}$	Memory-U$^{v^*}$	$\sigma_i^{[\check{n}-1]}$
ZPM	$0.8e-6$	0.543853	35.8757	11	36.764	4.74778
MSP	$7.4e-26$	0.065468	45.657	28	29.875	5.10035
NDP	$3.0e-33$	0.074545	17.9858	22	21.764	6.99424

Table 29
Residual error results for closely initialized values to exact roots of (76) using parallel techniques in engineering application 5.7.

Method	ČÖ	ČPŬ	ň	$\epsilon_1^{[\check{n}]}$	$\epsilon_2^{[\check{n}]}$	$\epsilon_3^{[\check{n}]}$	$\epsilon_4^{[\check{n}]}$	$\sigma_i^{[\check{n}-1]}$
ZPM	5	0.054	7	$4.8e-55$	$1.8e-36$	$0.3e-16$	$3.7e-26$	5.15
MSP	6	0.043	3	$8.0e-85$	$1.8e-56$	0.0	0.0	6.13
NDP	6	0.031	3	$1.0e-132$	$3.4e-131$	0.0	0.0	6.54

Table 30
Random initial guess values utilised for consistency analysis of parallel schemes.

$\mathbf{v}^{[*]}$	$x_1^{[0]}$		$x_2^{[0]}$		$x_3^{[0]}$		$x_4^{[0]}$	
	[Re,	Im]	[Re,	Im]	[Re,	Im]	[Re,	Im]
υ_1^*	[0.05,	0.56]	[0.17,	0.20]	[0.42,	0.63]	[0.50,	0.16]
υ_2^*	[0.03,	0.93]	[0.504,	0.04]	[0.07,	0.05]	[0.07,	0.05]
υ_3^*	[0.12,	0.52]	[0.10,	0.70]	[0.66,	0.08]	[0.36,	0.05]

Table 31
Residual error outcome for engineering application 5.7 on random initial vectors.

Method	Ini-V	ČÖ	ČPŬ	ň	$\epsilon_1^{[\check{n}]}$	$\epsilon_2^{[\check{n}]}$	$\epsilon_3^{[\check{n}]}$	$\epsilon_4^{[\check{n}]}$	$\sigma_i^{[\check{n}-1]}$
ZPM	υ_1^*	5	0.6874	47	$7.1e-16$	$9.5e-7$	$6.9e-2$	$8.9e-20$	5.09765
MSP	υ_1^*	5	0.6546	47	$0.1e-18$	$1.5e-7$	$7.2e-1$	$8.2e-19$	4.09766
NDP	υ_1^*	5	0.5463	47	$8.6e-17$	$3.5e-6$	$5.2e-2$	$1.9e-37$	6.10015
ZPM	υ_2^*	6	0.4825	37	$7.1e-16$	$5.5e-7$	$6.9e-2$	$8.9e-20$	5.16554
MSP	υ_2^*	6	0.4824	37	$2.1e-18$	$0.5e-7$	$7.2e-7$	$8.2e-19$	4.14548
NDP	υ_2^*	6	0.4825	37	$8.6e-17$	$3.5e-6$	$5.2e-7$	$1.9e-37$	6.10015
ZPM	υ_3^*	6	0.0712	19	$1.0e-31$	$2.3e-9$	$2.6e-8$	$5.0e-39$	6.0096
MSP	υ_3^*	6	0.0711	19	$0.8e-29$	$2.3e-7$	$6.0e-8$	$5.0e-49$	6.14876
NDP	υ_3^*	6	0.0715	19	$0.5e-25$	$5.3e-9$	$2.0e-9$	$5.0e-34$	6.14464

- The inflection points and curvature of the polynomial reveal areas of defocusing and focusing, which are crucial to wave stability in nonlinear systems.
- Although it is not an exact soliton, this cubic approximation simulates a solitary wave shape to investigate nonlinearity-induced behaviours such as peak generation and steepening.
- Symbolic analysis of wave behaviour and simpler numerical simulation are made possible by these polynomial approximations, which eliminate the necessity for numerically solving intricate PDEs.

5.7. Higher degree polynomials with multiple roots [70]

Consider

$$g(x) = (x+1)^{300}(x-2)^{600}(x-1-i)^{700}(x-1+i)^{400}, \tag{76}$$

with exact roots:

$$\zeta_1 = -1,\ \zeta_2 = 2,\ \zeta_3 = 1+i,\ \zeta_4 = 1-i, \tag{77}$$

having the following multiplicity

$$\alpha_1 = 300, \alpha_1 = 600, \alpha_1 = 700, \alpha_1 = 400, \tag{78}$$

and $\sum_{j=1}^{4} \alpha_j = 2000$ degree of (76). The initial estimates have been taken as:

$$x_1^{[0]} = -1.1 + 0.2i, x_2^{[0]} = 2.1 - 0.2i, x_3^{[0]} = 0.8 + 1.2i, x_4^{[0]} = 0.9 - 1.2i. \tag{79}$$

Results of the NDP, ZPM, and MSP in terms of numbers for solving (76) are shown in Table 29. In terms of computational convergence order (CO), computational CPU time, residual errors, and iterations numbers, Table 29 clearly shows that the technique NDP is superior to ZPM, MSP.

The numerical results in Table 29 clearly show that in terms of residual error, the newly developed method outperforms ZPM, MSP, with a high computational order of convergence and a rapid convergence rate. To evaluate convergence behaviour, random initial guesses are used. This parallel technique is sometimes crucial for clustered roots, where convergence analysis is hard to achieve. Using some random initial vectors, $\upsilon_1^* - \upsilon_3^*$, we can determine the global convergence behaviour. On these random initial vectors, NDP takes 19 iterations and achieves a maximum residual error of 7.3e-25 in 0.0714 seconds on computers, ZPM takes 47 iterations and achieves a maximum residual error of 1.5e-7 consuming 0.6875 seconds on computer, while MSP takes 37 iterations and achieves a maximum residual error of 2.1e-17 consuming 0.481 seconds on computer. The results of the Table 30 clearly show that NDP outperforms ZPM, MSP and converges are more stable. The random initial vectors $\upsilon^* = [x_1^{[0]}, x_2^{[0]}, x_3^{[0]}, x_4^{[0]}]$ are given in Table 30.

The outcomes of the parallel schemes utilizing the random initial vectors given in Table 30 are presented in Table 31.

In Table 31, Ini-V denotes the set of random initial vectors used to test the global convergence behaviour of the NDP, ZPM, and MSP techniques. The Table 31's results clearly show that, in terms of global convergence, the NDP technique performs better than the ZPM, MSP method.

Table 32 and Fig. 12(a,c) clearly demonstrate that our technique outperforms the existing ZPM and MSP methods in terms of maximum error, computational time, percentage convergence, average number of iterations, and computational order of convergence.

5.7.1. Performance insights and limitations

The proposed parallel algorithm addresses a wide range of nonlinear engineering problems with improved precision, stability, and convergence. In terms of residual error, percentage convergence, and computational efficiency, the technique exceeds existing methods. Despite these advantages, several numerical and computational limitations that can be identified in specific situations are as follows:

Table 32
Overall performance of parallel techniques for solving engineering application 5.7.

Method	Max-Error$^{v^*}$	ČPŮ-time$^{v^*}$	P-Con$^{v^*}$	Average-It$^{v^*}$	Memory-U$^{v^*}$	$\sigma_i^{[\hat{n}-1]}$
ZPM	$9.8e-7$	0.5437	25.6563	47	76.764	4.76464
MSP	$4.6e-7$	0.4512	35.7647	37	56.653	5.18977
NDP	$1.1e-9$	0.0745	67.8736	19	45.875	6.19424

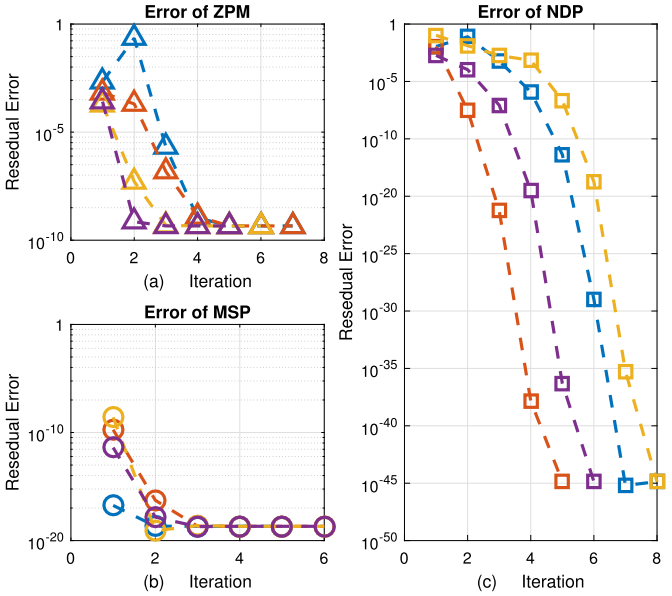


Fig. 12. (a-c), illustrate the residual error graphs for the parallel schemes—ZPM, MSP, and NDP—that were utilized to solve engineering application 5.7.

- In parallel configurations, particularly when root clusters or degenerate roots exist, the computational workload might be inconsistent across threads or nodes, resulting in empty cores and low hardware utilization.
- Inter-process communication for evaluating correction terms across all root estimations can become a bottleneck as polynomiality increases, especially on distributed-memory systems.
- High-precision data and correction terms require large amounts of memory, which could affect system performance when memory is insufficient.
- Ensuring numerical consistency and stability in the presence of several parallel threads necessitates regular synchronization, which can increase latency and degrade parallel performance.
- Parallel speedup is architecture-dependent (e.g., CPU, GPU, hybrid). Some settings lack optimized libraries for the advanced arithmetic necessary in high-order iteration techniques.

Higher-order correction terms, adaptive precision, and deflation methods may contribute to improve convergence. Robustness is further enhanced by using machine learning to make better initial guesses and symbolic reprocessing. Hybrid CPU-GPU implementations and dynamic load balancing boost parallel performance. Together, these strategies hasten convergence, particularly for clustered or multiple complex roots.

6. Conclusion and future work

A new parallel iterative method, called NDP, for approximating the roots of nonlinear equations was introduced in this study. This method converges at a rate of six and was developed to be computationally efficient for use in science and engineering. The main accomplishments of this research can be summarised as follows:

- Stability analysis, which incorporates parametric and dynamical planes (Fig. 1- 3), are used to determine the best parameter values to accelerate the convergence rate of NDP parallel scheme.
- The developed parallel technique NDP performs better than ZPM and MSP methods in terms of memory usage, computing time (Tables 4- 32), residual error in both close and random initial guess values (Figs. 6- 12), and computational efficiency (e.g., Table 2 and Fig. 4).
- The NDP technique outperformed existing methods ZPM and MSP in terms of residual error, iteration count, and computational consistency (see Tables 2-32 and Figs. 4-12).
- These findings indicate that the NDP technique is a feasible and efficient alternative to solving nonlinear equations, and it should be further investigated for use in larger scientific and engineering contexts [74,75].

Despite the promising results, there are some limitations to the suggested approach for solving nonlinear problems. The performance of the parallel techniques may be influenced by the structure and distribution of the initial guesses, especially in the case of closely clustered or near-multiple roots. In addition, while parallelism improves computational efficiency, performance improvements can be limited on systems with limited memory bandwidth or fewer processing cores. The parallel methods require analytical formulations for derivatives, which may be challenging for data-driven models.
Future research could focus on the following cutting-edge advances:

- Interval arithmetic and spectral radius analysis are used to get verified convergence bounds, particularly for clustered or multiple roots.
- Machine learning, such as neural networks, will be used to improve convergence reliability and predict the best initial estimates.
- Using distributed-memory models (e.g., MPI and OpenMP) to develop adjustable, task-parallel implementations that enable simultaneous root processing.

CRediT authorship contribution statement

Mudassir Shams: Writing – review & editing, Writing – original draft, Visualization, Validation, Software, Methodology, Investigation, Formal analysis, Data curation, Conceptualization. **Nasreen Kausar:** Writing – original draft, Validation, Supervision, Software, Project administration, Methodology, Investigation, Funding acquisition, Formal analysis, Conceptualization. **Ali Akgül:** Software, Project administration, Methodology, Investigation, Formal analysis, Data curation, Conceptualization. **Tonguç Çağın:** Writing – review & editing, Writing – original draft, Validation, Supervision, Project administration, Funding acquisition, Data curation.

Declaration of competing interest

All authors confirm that they have no conflict of interest regarding the manuscript.

References

[1] Rodríguez-Belenguer P, March-Vila E, Pastor M, Mangas-Sanjuan V, Soria-Olivas E. Usage of model combination in computational toxicology. Toxicol Lett 2023;389:34–44.

[2] Wang P, Yin F, ur Rahman M, Khan MA, Baleanu D. Unveiling complexity: exploring chaos and solitons in modified nonlinear Schrödinger equation. Results Phys 2024;56:107268.
[3] Fazly M. Stable solutions of symmetric systems involving hypoelliptic operators. J Funct Anal 2018;274(12):3470–502.
[4] Kempner ES. Movable lobes and flexible loops in proteins structural deformations that control biochemical activity. FEBS Lett 1993;326(1–3):4–10.
[5] Rössler OE. Chaotic behavior in simple reaction systems. Z Naturforsch A 1976 Apr 1;31(3–4):259–64.
[6] Lapidus L, Pinder GF. Numerical solution of partial differential equations in science and engineering. John Wiley & Sons; 1999 Jul 8.
[7] Pan VY. Solving Polynomials with Computers: Speedy computer algorithms offer new answers to a mathematical problem as ancient as Babylon: finding the zeros, or roots, of high-degree polynomials. Am Sci 1998;86(1):62–9.
[8] Griffiths PA. Variations on a theorem of Abel . Invent Math 1976;35(1):321–90.
[9] Vogelsberger M, Marinacci F, Torrey P, Puchwein E. Cosmological simulations of galaxy formation. Nat Rev Phys 2020;2(1):42–66.
[10] Haggerty CC, Caprioli D. Kinetic simulations of cosmic-ray-modified shocks. I. Hydrodynamics. Astrophys J 2020;905(1):1.
[11] Ortega JM, Rheinboldt WC. Iterative solution of nonlinear equations in several variables. Soc Ind Appl Math 2000 Jan 1.
[12] Jarratt P. Some fourth order multipoint iterative methods for solving equations. Math Comput 1966;20(95):434–7.
[13] King RF. A family of fourth order methods for nonlinear equations. SIAM J Numer Anal 1973;10(5):876–9.
[14] Wang X, Sun M. A new family of fourth-order Ostrowski-type iterative methods for solving nonlinear systems. AIMS Math 2024;9(4):10255–66.
[15] Zhao H. A reduced order model based on machine learning for numerical analysis: an application to geomechanics. Eng Appl Artif Intell 2021;100:104194.
[16] Wang X, Guo S. A class of fifth-order Chebyshev–Halley-type iterative methods and its stability analysis. Fractal Fract 2024;8(3):150.
[17] Erfanifar R, Hajarian M, Sayevand K. A family of iterative methods to solve nonlinear problems with applications in fractional differential equations. Math Methods Appl Sci 2024;47(4):2099–119.
[18] Weierstrass K, des Satzes NB. Dass jede ganze rationale Function einer Verän derlichen dargestellt werden kann als ein Product aus linearen Functionen derselben Verän derlichen. Sitzungsberichte der Königlich Preussischen Akademie der Wissenschaften zu Berlin; 1891.
[19] Durand É. Solutions numeriques des equations algebriques Tom 1: equation du tupe $F(X) = 0$. Racines d'um Polnome Masson; 1960.
[20] Kerner IO. Ein gesamtschrittverfahren zur berechnung der nullstellen von polynomen. Numer Math 1966;8(3):290–4.
[21] Aberth O. Iteration methods for finding all zeros of a polynomial simultaneously. Math Comput 1973;27(122):339–44.
[22] Marcheva PI, Ivanov IK, Ivanov SI. On the Q-convergence and dynamics of a modified Weierstrass method for the simultaneous extraction of polynomial zeros. Algorithms 2025;18(4):205.
[23] Anourein AW. An improvement on two iteration methods for simultaneous determination of the zeros of a polynomial. Int J Comput Math 1977;6(3):241–52.
[24] Nedzhibov GH. Iterative methods for simultaneous computing arbitrary number of multiple zeros of nonlinear equations. Int J Comput Math 2013;90(5):994–1007.
[25] Petkovic LJ, Trickovic SL. On the construction of simultaneous methods for multiple zeros. Nonlinear Anal 1997;30(2):669–76.
[26] Pan VY. Old and new nearly optimal polynomial root-finders. In: Computer algebra in scientific computing: 21st international workshop. Springer International Publishing; 2019. p. 393–411.
[27] Herceg D, Tričković S, Petković M. On the fourth order methods of Weierstrass' type. Nonlinear Anal, Theory Methods Appl 1997;30:83–8.
[28] Zeng Z. Computing multiple roots of inexact polynomials. Math Comput 2005;74(250):869–903.
[29] Yang Z, Hon BY. An improved modified extended tanh-function method. Z Naturforsch A 2006;61(3–4):103–15.
[30] Iliev A, Kyurkchiev N. Nontrivial methods in numerical analysis: selected topics in numerical analysis. LAP Lambert Academic Publishing; 2010.
[31] Proinov PD, Petkova MD. A new semilocal convergence theorem for the Weierstrass method for finding zeros of a polynomial simultaneously. J Complex 2014;30(3):366–80.
[32] Kyncheva VK, Yotov VV, Ivanov SI. On the convergence of Schröder's method for the simultaneous computation of polynomial zeros of unknown multiplicity. Calcolo 2017;54(4):1199–212.
[33] Psihoyios G, Simos TE. The numerical solution of the radial Schrödinger equation via a trigonometrically fitted family of seventh algebraic order predictor–corrector methods. J Math Chem 2006;40:269–93.
[34] Larbaoui Y. New theorems and formulas to solve fourth degree polynomial equation in general forms by calculating the four roots nearly simultaneously. Am J Appl Math 2023;11(6):95–105.
[35] Nedzhibov G. Dynamic mode decomposition via polynomial root-finding methods. Mathematics 2025;13(5):709.
[36] Pan VY. Nearly optimal black box polynomial root-finders. In: Proceedings of the 2024 annual ACM-SIAM symposium on discrete algorithms (SODA). Society for Industrial and Applied Mathematics; 2024. p. 3860–900.
[37] Meza JC. Newton's method. Wiley Interdiscip Rev Comput Stat 2011;3(1):75–8.
[38] Chicharro FI, Cordero A, Garrido N, Torregrosa JR. Generating root-finder iterative methods of second order: convergence and stability. Axioms 2019;8(2):55.
[39] Kou J, Li Y. A family of new Newton-like methods. Appl Math Comput 2007;192(1):162–7.
[40] Noor MA, Noor KI, Khan WA, Ahmad F. On iterative methods for nonlinear equations. Appl Math Comput 2006;183(1):128–33.
[41] Shams M, Kausar N, Araci S, Kong L. On the stability analysis of numerical schemes for solving non-linear polynomials arises in engineering problems. AIMS Math 2024;9(4):8885–903.
[42] Aurentz JL, Mach T, Vandebril R, Watkins DS. Fast and backward stable computation of roots of polynomials. SIAM J Matrix Anal Appl 2015;36(3):942–73.
[43] Abdullah S, Choubey N, Dara S. Two novel with and without memory multipoint iterative methods for solving non-linear equations. Commun Math Appl 2024;15(1):9.
[44] Ogbereyivwe O, Izevbizua O. A three-free-parameter class of power series based iterative method for approximation of nonlinear equations solution. Iran J Numer Anal Optim 2023;13(2):157–69.
[45] Bhalla S, Panwar M, Behl R, Chun C. Simultaneous root approximation: a high-convergence iterative approach. Math Methods Appl Sci 2025;48(8):9098–107.
[46] Ivanov SI. Families of high-order simultaneous methods with several corrections. Numer Algorithms 2024;97(2):945–58.
[47] Cordero A, Garrido N, Torregrosa JR, Triguero-Navarro P. Iterative schemes for finding all roots simultaneously of nonlinear equations. Appl Math Lett 2022;134:108325.
[48] Petković MS, Petković LD, Džunić J. On an efficient simultaneous method for finding polynomial zeros. Appl Math Lett 2014:60–5.
[49] Mir NA, Muneer R, Jabeen I. Some families of two-step simultaneous methods for determining zeros of nonlinear equations. Int Sch Res Not 2011;2011(1):817174.
[50] Farmer MR. Computing the zeros of polynomials using the divide and conquer approach. London, UK: Department of Computer Science and Information Systems, Birkbeck, University of London; 2014.
[51] Larbaoui Y. New theorems solving fifth degree polynomial equation in complete forms by proposing new five roots composed of radical expressions. Am J Appl Math 2024;12(1):9–23.
[52] Marcheva PI, Ivanov SI. Convergence analysis of a modified Weierstrass method for the simultaneous determination of polynomial zeros. Symmetry 2020;12(9):1408.
[53] Cholakov SI. Local and semilocal convergence of Wang-Zheng's method for simultaneous finding polynomial zeros. Symmetry 2019;11(6):736.
[54] Sendov B, Andreev A, Kjurkchiev N. Numerical solution of polynomial equations. Handb Numer Anal 1994;3:625–778.
[55] Petković MS, Petković LD, Džunić J. On an efficient simultaneous method for finding polynomial zeros. Appl Math Lett 2014;28:60–5.
[56] Consnard M, Fraigniaud P. Finding the roots of a polynomial on an MIMD multicomputer. Parallel Comput 1990;15(1–3):75–85.
[57] Proinov PD. On the local convergence of Gargantini-Farmer-Loizou method for simultaneous approximation of multiple polynomial zeros. J Nonlinear Sci Appl 2018;11(9):1045–55.
[58] Nedzhibov GH. Inverse Weierstrass-Durand-Kerner iterative method. Int J Appl Math 2013;28(2):1258–64.
[59] Pan VY. Solving a polynomial equation: some history and recent progress. SIAM Rev 1997;39(2):187–220.
[60] Agathiyan A, Fataf NA, Gowrisankar A. Explicit relation between Fourier transform and fractal dimension of fractal interpolation functions. Eur Phys J Spec Top 2023;232(7):1077–91.
[61] Agathiyan A, Gowrisankar A, Fataf NA. On the integral transform of fractal interpolation functions. Math Comput Simul 2024;222:209–24.
[62] Agathiyan A, Gowrisankar A, Fataf NA, Cao J. Remarks on the integral transform of non-linear fractal interpolation functions. Chaos Solitons Fractals 2023;173:113749.
[63] Shams M, Kausar N, Samaniego C, Agarwal P, Ahmed SF, Momani S. On efficient fractional Caputo-type simultaneous scheme for finding all roots of polynomial equations with biomedical engineering applications. Fractals 2023;31(04):2340075.
[64] Dong C. A family of multipoint iterative functions for finding multiple roots of equations. Int J Comput Math 1987;21(3–4):363–7.
[65] Petković MS, Petković LD, Džunić J. On an efficient method for the simultaneous approximation of polynomial multiple roots. Appl Anal Discrete Math 2014:73–94.
[66] Zhang X, Peng H, Hu G. A high order iteration formula for the simultaneous inclusion of polynomial zeros. Appl Math Comput 2006;179(2):545–52.
[67] Kumar S. A new fractional modeling arising in engineering sciences and its analytical approximate solution. Alex Eng J 2013;52(4):813–9.
[68] Kumar P, Agrawal OP. An approximate method for numerical solution of fractional differential equations. Signal Process 2006;86(10):2602–10.
[69] Martín MM, editor. Introduction to software for chemical engineers. CRC Press; 2025 Mar 24.
[70] Akram S, Akram F, Junjua MU, Arshad M, Afzal T. A family of optimal eighth order iteration functions for multiple roots and its dynamics. J Math 2021;2021(1):5597186.
[71] Hattaf K. On the stability and numerical scheme of fractional differential equations with application to biology. Computation 2022;10(6):97.

[72] Jassim MH, Al-Hayani WM. Combining Laplace transform and variational iteration method for solving singular IVPs and BVPs of Lane–Emden type equation. Wasit J Comput Math Sci 2024;3(2):8–18.

[73] Pavkov TM, Kabadzhov VG, Ivanov IK, Ivanov SI. Local convergence analysis of a one parameter family of simultaneous methods with applications to real-world problems. Algorithms 2023;16(2):103.

[74] Khuri S, Assadi R. An extended variational iteration method for fractional BVPs encountered in engineering applications. Int J Numer Methods Heat Fluid Flow 2023;33(7):2671–81.

[75] Fazly M, Shahgholian H. Monotonicity formulas for coupled elliptic gradient systems with applications. Adv Nonlinear Anal 2019;9(1):479–95.