mathematics

MDPI

Article

On Highly Efficient Fractional Numerical Method for Solving Nonlinear Engineering Models

Mudassir Shams [1,2] and Bruno Carpentieri [1,*]

[1] Faculty of Engineering, Free University of Bozen-Bolzano (BZ), 39100 Bolzano, Italy; mudassir4shams@gmail.com or mudassir.shams@unibz.it
[2] Department of Mathematics and Statistics, Riphah International University, I-14, Islamabad 44000, Pakistan
* Correspondence: bruno.carpentieri@unibz.it

Abstract: We proposed and analyzed the fractional simultaneous technique for approximating all the roots of nonlinear equations in this research study. The newly developed fractional Caputo-type simultaneous scheme's order of convergence is $3\varsigma + 5$, according to convergence analysis. Engineering-related numerical test problems are taken into consideration to demonstrate the efficiency and stability of fractional numerical schemes when compared to previously published numerical iterative methods. The newly developed fractional simultaneous approach converges on random starting guess values at random times, demonstrating its global convergence behavior. Although the newly developed method shows global convergent behavior when all starting guess values are distinct, the method diverges otherwise. The total computational time, number of iterations, error graphs and maximum residual error all clearly illustrate the stability and consistency of the developed scheme. The rate of convergence increases as the fractional parameter's value rises from 0.1 to 1.0.

Keywords: computational efficiency; error graph; optimal order; simultaneous methods; computer algorithm

MSC: 65H04; 65H05; 65H17

check for updates

Citation: Shams, M.; Carpentieri, B. On Highly Efficient Fractional Numerical Method for Solving Nonlinear Engineering Models. *Mathematics* **2023**, *11*, 4914. https://doi.org/10.3390/math11244914

Academic Editor: Ioannis K. Argyros

Received: 26 October 2023
Revised: 4 December 2023
Accepted: 5 December 2023
Published: 10 December 2023

Copyright: © 2023 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (https://creativecommons.org/licenses/by/4.0/).

1. Introduction

When analytical approaches are not available, iterative schemes are the only viable strategy for numerically approximating the roots of nonlinear equations

$$f(r) = 0, \tag{1}$$

in a stable manner. They start with an initial approximation and iteratively refine the solution using algebraic equations until a satisfactory approximation is obtained. This approximation of the solution is carried out in this manner until every root is identified. There are two types of iterative root-finding schemes: simultaneous techniques, which approximate all roots simultaneously, and methods which approximate one root at a time (see, for example, Traub's method [1], Jarratt's method [2], King's method [3], Ostrowski's method [4], Chun et al.'s method [5], and many others). In recent years, simultaneous techniques have grown in popularity as a result of their global convergence and inherent parallelism (see, for example, the works by Weierstrass [6], Kanno [7], Proinov [8], Mir [9], Farmer [10], Nourein [11], Aberth [12], and Cholakov [13] and the references therein). On the other hand, because of the intrinsic difficulties of these equations, such as the non-linearity and non-locality, standard analytical, semi-analytical, and classical numerical approaches are typically ineffective.

In order to decrease the overall computational time, parallel numerical schemes utilize parallel computing [14] to solve nonlinear equations. This is achieved by decomposing the problem into smaller tasks, which can be executed simultaneously on multiple processors

or cores. Therefore, these schemes are particularly useful when dealing with large-scale or computationally intensive engineering problems [15]. A comprehensive understanding of parallel programming techniques, algorithms, and the specific characteristics of the problem at hand is necessary for the effective implementation of parallel numerical schemes. Furthermore, the selection of a parallel scheme is often influenced by the nature of the nonlinear equations being solved, the hardware at hand, and the size of the problem. An overview of parallel numerical methods for solving nonlinear equations can be found in [16–18].

The performance of simultaneous root-finding algorithms varies depending on the initial guess and the problem at hand, and convergence is not always guaranteed [19–21]. As a result, efforts have been made to develop more robust and efficient procedures. In this research, we propose highly efficient fractional numerical techniques for simultaneously approximating all the roots of nonlinear equations. Fractional simultaneous methods utilize fractional-order derivatives of the function to solve (1). Fractional calculus, which is concerned with non-integer-order derivatives and integrals, is used in many areas, including physics, engineering, and finance [22–24]. A comprehensive analysis of the convergence and of the computational complexity of our method is derived. The performance and global convergence behavior of the algorithm is assessed for solving some practical engineering applications by considering various factors, including CPU time, maximum computational time on random initial guess values, maximum residual error, and local computational order of convergence.

The structure of the paper is outlined as follows. After the introduction, we discuss some basic definitions in Section 2. In Section 3, parallel computing schemes are developed and analyzed to solve (1). Section 4 compares the computational aspects of newly proposed simultaneous techniques to existing methods in the literature. In Section 5, we discuss the numerical results of the newly developed scheme. The conclusion of the paper is in Section 6.

2. Some Preliminaries

In this section, we will go over some fundamental aspects of fractional calculus as well as the fractional iterative approach to solving nonlinear equations using Caputo-type derivatives, even though, apart from the Caputo derivative, all fractional-type derivatives do not fulfill the criteria for a fractional calculus. $[D^{\varsigma}](1) = 0$ if ς is not a natural number.

The gamma function is described as follows [25]:

$$\Gamma(r) = \int_0^{+\infty} u^{r-1} e^{-u} du, \tag{2}$$

where $r > 0$. Gamma is a generalization of the factorial function due to $\Gamma(1) = 1$ and $\Gamma(n+1) = n!$, where $n \in \mathbb{N}$.

Order ς's Caputo fractional derivative [26–29] with $\varsigma > 0, \varsigma_1, \varsigma, r \in \mathbb{R}$ is stated as:

$$\left[{}_C D_{\varsigma_1}^{\varsigma}\right] f(r) = \begin{cases} \frac{1}{\Gamma(m-\varsigma)} \int_0^r \frac{d^m}{dt^m} f(t) \frac{1}{(r-t)^{\varsigma-m+1}} dt, & m-1 < \varsigma \leq m \in \mathbb{N}, \\ \frac{d^m}{dt^m} f(r), & \varsigma = m \in \mathbb{N}, \end{cases} \tag{3}$$

where $\Gamma(r)$ is a gamma function with $r > 0$.

Theorem 1 (Generalized Taylor formula [30,31]). *Suppose* $\left[{}_C D_{\varsigma_1}^{j\varsigma}\right] f(r) \in C([\varsigma_1, \varsigma_2])$ *for* $j = 1, \ldots, n+1$ *where* $\varsigma \in (0, 1]$, *then*

$$f(r) = \sum_{i=0}^{n} \left[{}_C D_{\varsigma_1}^{i\varsigma}\right] f(\varsigma_1) \frac{(r-\varsigma_1)^{i\varsigma}}{\Gamma(i\varsigma+1)} + \left[{}_C D_{\varsigma_1}^{(n+1)\varsigma}\right] f(\xi) \frac{(r-\varsigma_1)^{(n+1)\varsigma}}{\Gamma((n+1)\varsigma+1)}, \tag{4}$$

and $\varsigma_1 \leq \xi \leq r, \forall r \in (\varsigma_1, \varsigma_2]$ *and* $\left[{}_C D_{\varsigma_1}^{n\varsigma}\right] = \left[{}_C D_{\varsigma_1}^{\varsigma}\right].\left[{}_C D_{\varsigma_1}^{\varsigma}\right] \ldots \left[{}_C D_{\varsigma_1}^{\varsigma}\right]$ *(n-times).*

In terms of the Caputo-type Taylor development of $f(r)$ around $\varsigma_1 = \xi$, then

$$f(r) = \frac{\left[{}_C D_{\xi}^{\varsigma}\right] f(\xi)}{\Gamma(\varsigma+1)}(r-\xi)^{\varsigma} + \frac{\left[{}_C D_{\xi}^{2\varsigma}\right] f(\xi)}{\Gamma(2\varsigma+1)}(r-\xi)^{2\varsigma} + O(r-\xi)^{3\varsigma}. \tag{5}$$

Taking the $\frac{\left[{}_C D_{\xi}^{\varsigma}\right] f(\xi)}{\Gamma(\varsigma+1)}$ common, we have:

$$f(r) = \frac{\left[{}_C D_{\xi}^{\varsigma}\right] f(\xi)}{\Gamma(\varsigma+1)}\left[(r-\xi)^{\varsigma} + C_2(r-\xi)^{2\varsigma}\right] + O(r-\xi)^{3\varsigma}, \tag{6}$$

where

$$C_{\gamma} = \frac{\Gamma(\varsigma+1)}{\Gamma(\gamma\varsigma+1)}\frac{\left[{}_C D_{\xi}^{\gamma\varsigma}\right] f(\xi)}{\left[{}_C D_{\xi}^{\varsigma}\right] f(\xi)}, \gamma = 2, 3, \ldots \tag{7}$$

The corresponding derivative of the Caputo type of $f(r)$ around ξ is

$$\left[{}_C D_{\xi}^{\varsigma}\right] f(r) = \frac{\left[{}_C D_{\xi}^{\varsigma}\right] f(\xi)}{\Gamma(\varsigma+1)}\left[\Gamma(\varsigma+1) + \frac{\Gamma(2\varsigma+1)}{\Gamma(\varsigma+1)}C_2(r-\xi)^{\varsigma}\right] + O(r-\xi)^{2\varsigma}. \tag{8}$$

The classic Newton–Raphson technique is the most widely used method for locating a single root:

$$y^{[\vartheta]} = r^{[\vartheta]} - \frac{f(r^{[\vartheta]})}{f'(r^{[\vartheta]})}, (\vartheta = 1, 2, \ldots),\ f'(r^{[\vartheta]}) \neq 0. \tag{9}$$

Akgül et al. [31], Torres-Hernandez et al. [32], Gajori et al. [33] and Kumar et al. [34] discuss the fractional Newton method with different types of fractional derivatives. For the Caputo type of the classical Newton's method (FNN), Candelario et al. [35] propose the following fractional version:

$$z^{[\vartheta]} = r^{[\vartheta]} - \left(\Gamma(\varsigma+1)\frac{f(r^{[\vartheta]})}{\left[{}_C D_{\varsigma_1}^{\varsigma}\right] f(r^{[\vartheta]})}\right)^{1/\varsigma}, \tag{10}$$

where $\left[{}_C D_{\varsigma_1}^{\varsigma}\right] f(r^{[\vartheta]}) \approx \left[{}_C D_{\xi}^{\varsigma}\right] f(\xi)$ for any $\varsigma \in \mathbb{R}$. The order of convergence of the fractional Newton method is $\varsigma + 1$, satisfying the following error equation:

$$e^{[\vartheta+1]} = \left(\frac{\Gamma(2\varsigma+1) - \Gamma^2(\varsigma+1)}{\varsigma\Gamma^2(\varsigma+1)}\right) C_2\left(e^{[\vartheta]}\right)^{\varsigma+1} + O\left(\left(e^{[\vartheta]}\right)^{2\varsigma+1}\right), \tag{11}$$

where $e^{[\vartheta+1]} = z^{[\vartheta]} - \xi$ and $e^{[\vartheta]} = r^{[\vartheta]} - \xi$ and $C_{\gamma} = \left(\frac{\Gamma(\varsigma+1)}{\Gamma(\gamma\varsigma+1)}\right)\left(\frac{\left[{}_C D_{\xi}^{\gamma\varsigma}\right] f(\xi)}{\left[{}_C D_{\xi}^{\varsigma}\right] f(\xi)}\right), \gamma = 2, 3, \ldots$

Candelario et al. [35] also present the following fractional numerical scheme (M^{σ}) for solving simple roots of nonlinear equations:

$$\begin{cases} y^{[\vartheta]} = r^{[\vartheta]} - \left(\Gamma(\varsigma+1)\frac{f(r^{[\vartheta]})}{\left[{}_C D_{\varsigma_1}^{\varsigma}\right] f(r^{[\vartheta]})}\right)^{1/\varsigma}, \\ v^{[\vartheta]} = y^{[\vartheta]} - \left(\Gamma(\varsigma+1)\frac{f(y[\vartheta])}{\left[{}_C D_{\varsigma_1}^{\varsigma}\right] f(r^{[\vartheta]})}\right)^{1/\varsigma}. \end{cases} \tag{12}$$

The order of convergence of M^{σ} is $2\varsigma+1$, and the error equation is given as:

$$e^{[\vartheta+1]} = -\left(\frac{\Gamma(2\varsigma+1)-\Gamma^2(\varsigma+1)}{\varsigma^2\Gamma^2(\varsigma+1)}\right)\Lambda C_2^2\left(e^{[\vartheta]}\right)^{2\varsigma+1} + O\left(\left(e^{[\vartheta]}\right)^{\varsigma^2+2\varsigma+1}\right), \tag{13}$$

where $\Lambda = \frac{\Gamma(2\varsigma+1)-\Gamma^2(\varsigma+1)}{\Gamma^2(\varsigma+1)}$ and $e^{[\vartheta+1]} = v^{[\vartheta]} - \xi$.

3. Construction of Fractional Parallel Computing Scheme for Estimating All Distinct and Multiple Roots

Weierstrass-Dochive [18] presents the following local quadratic convergence scheme:

$$u_i^{[\vartheta]} = r_i^{[\vartheta]} - w(r_i^{[\vartheta]}), \tag{14}$$

where

$$w(r_i^{[\vartheta]}) = \frac{f(r_i^{[\vartheta]})}{\prod\limits_{\substack{j=1\\ j\neq i}}^{n}(r_i^{[\vartheta]} - r_j^{[\vartheta]})}, \quad (i,j=1,2,\ldots,n), \tag{15}$$

is Weierstrass' Correction.

In [19], Nedzibove et al. present two new modifications to (14) as:

$$u_i^{[\vartheta]} = \frac{\left(r_i^{[\vartheta]}\right)^2 * \left(\prod\limits_{\substack{j=1\\ j\neq i}}^{n}(r_i^{[\vartheta]} - r_j^{[\vartheta]})\right)}{r_i^{[\vartheta]} * \prod\limits_{\substack{j=1\\ j\neq i}}^{n}(r_i^{[\vartheta]} - r_j^{[\vartheta]}) + f(r_i^{[\vartheta]})}. \tag{16}$$

In order to construct an iterative process for approximating all the multiple roots of polynomial, let us assume a monic polynomial of degree n with roots ξ_j having known multiplicities α_j such that $\sum\limits_{j=1}^{m}\alpha_j = n$:

$$f(r) = r^n + a_{n-1}r^{n-1} + \ldots\ldots + a_2r^2 + ax + a_0 = \Pi_{j=1}^{m}(r-\xi_j)^{\alpha_j}. \tag{17}$$

Consider the Newton correction $U_i^{[\vartheta]} = \frac{f(r_i^{[\vartheta]})}{f'(r_i^{[\vartheta]})}$ and

$$\breve{U}_i^{[\vartheta]} = \left(\Gamma(\varsigma+1)\frac{f(r^{[\vartheta]})}{[{}_C D_{\varsigma_1}^{\varsigma}]f(r^{[\vartheta]})}\right)^{1/\varsigma}. \tag{18}$$

This implies that

$$\frac{1}{U_i^{[\vartheta]}} = \sum_{\substack{j\neq i\\ j=1}}^{m}\frac{\alpha_j}{r_i^{[\vartheta]} - \xi_j}, \tag{19}$$

where ξ_j is the exact root and $r_j^{[\vartheta]}$ is its approximation. This gives

$$\frac{1}{U_i^{[\vartheta]}} = \frac{f'(r_i^{[\vartheta]})}{f(r_i^{[\vartheta]})} = \sum_{\substack{j\neq i\\ j=1}}^{m}\frac{\alpha_j}{r_i^{[\vartheta]} - \xi_j} = \frac{\alpha_i}{r_i^{[\vartheta]} - \xi_i} + \sum_{\substack{j\neq i\\ j=1}}^{m}\frac{\alpha_j}{r_i^{[\vartheta]} - \xi_j}, \tag{20}$$

$$\frac{\alpha_i}{r_i^{[\vartheta]} - \xi_i} = \frac{1}{U_i^{[\vartheta]}} - \sum_{\substack{j\neq i\\ j=1}}^{m}\frac{\alpha_j}{r_i^{[\vartheta]} - \xi_j}, \tag{21}$$

$$\xi_i = r_i^{[\vartheta]} - \frac{\alpha_i}{\frac{1}{U_i^{[\vartheta]}} - \sum_{\substack{j=1 \\ j\neq i}}^{m} \frac{\alpha_j}{\left(r_i^{[\vartheta]} - \xi_j\right)}}, (i = 1, 2, \ldots, m). \tag{22}$$

Substituting the roots ξ_j by its approximations $r_j^{[\vartheta]} (j \neq i)$ in (19), we obtain the third-order convergent Ehrlich–Aberth method [36] for the roots with multiplicities α_j.

$$r_i^{[\vartheta+1]} = r_i^{[\vartheta]} - \frac{\alpha_i}{\frac{1}{U_i^{[\vartheta]}} - \sum_{\substack{j=1 \\ j\neq i}}^{m} \frac{\alpha_j}{r_i^{[\vartheta]} - r_j^{[\vartheta]}}}, \tag{23}$$

where $r_i^{[\vartheta+1]}$ is new approximation to the root ξ_i. Instead of simple approximation $r_j^{[\vartheta]}$, we can apply some better approximation to ξ_j. The main goal in this accelerating process is to improve convergence. The aim can be achieved by choosing Newton's approximation $r_j^{[\vartheta]} - \alpha_j U_j^{[\vartheta]}$ instead of ξ_j in (22):

$$r_i^{[\vartheta+1]} = r_i^{[\vartheta]} - \frac{\alpha_i}{\frac{1}{U_i^{[\vartheta]}} - \sum_{\substack{j=1 \\ j\neq i}}^{m} \frac{\alpha_j}{r_i^{[\vartheta]} - r_j^{[\vartheta]} + \alpha_j U_j^{[\vartheta]}}}. \tag{24}$$

Now, we derive a new $3\varsigma + 5$-order method for the determination of all the roots of (1). Let $r_1^{[\vartheta]}, r_2^{[\vartheta]}, \ldots, r_m^{[\vartheta]}$ be reasonably close approximations to the roots $\xi_1, \xi_2, \ldots, \xi_m$, respectively, of polynomial $f(r)$, which means that $|\epsilon_i| = \max_{1\leq i\leq m} \left| r_i^{[\vartheta]} - \xi_i \right|$ is a sufficiently small quantity. Let us return to the relation (19) by replacing ξ_j with $r_j^{[\vartheta]} + +\alpha_j U_j^{[\vartheta]}$. We have:

$$\frac{1}{r_i^{[\vartheta]} - r_j^{[\vartheta]}} \cong \frac{1}{r_i^{[\vartheta]} - r_j^{[\vartheta]} + \alpha_j U_j^{[\vartheta]}} = \frac{1}{(r_i^{[\vartheta]} - r_j^{[\vartheta]})\left(1 + \frac{\alpha_j U_j^{[\vartheta]}}{r_i^{[\vartheta]} - r_j^{[\vartheta]}}\right)}. \tag{25}$$

Assuming that $|\epsilon_i|$ is small enough to provide $\left| \frac{\alpha_j U_j^{[\vartheta]}}{r_i^{[\vartheta]} - r_j^{[\vartheta]}} \right| < 1$, we use the development into geometric series and obtain:

$$\frac{1}{r_i^{[\vartheta]} - r_j^{[\vartheta]}} \cong \frac{1}{\left(r_i^{[\vartheta]} - r_j^{[\vartheta]}\right)} \left(\begin{array}{c} 1 - \frac{\alpha_j U_j^{[\vartheta]}}{r_i^{[\vartheta]} - r_j^{[\vartheta]}} + \\ \left(\frac{\alpha_j U_j^{[\vartheta]}}{r_i^{[\vartheta]} - r_j^{[\vartheta]}} \right)^2 + O|\epsilon_i|^3 \end{array} \right), \tag{26}$$

$$= \frac{1}{\left(r_i^{[\vartheta]} - r_j^{[\vartheta]}\right)} \left(\begin{array}{c} 1 - \frac{\alpha_j U_j^{[\vartheta]}}{r_i^{[\vartheta]} - r_j^{[\vartheta]}} + \\ \left(\frac{\alpha_j U_j^{[\vartheta]}}{r_i^{[\vartheta]} - r_j^{[\vartheta]}} \right)^2 + O|\epsilon_i|^3 \end{array} \right). \tag{27}$$

Neglecting terms of a higher order in the last relations, we obtain:

$$\frac{1}{r_i^{[\vartheta]} - r_j^{[\vartheta]}} \cong \frac{1}{r_i^{[\vartheta]} - r_j^{[\vartheta]} + \alpha_j U_j^{[\vartheta]}}, \tag{28}$$

$$= \frac{1}{\left(r_i^{[\vartheta]} - r_j^{[\vartheta]}\right)} \left(1 - \frac{\alpha_j U_j^{[\vartheta]}}{r_i^{[\vartheta]} - r_j^{[\vartheta]}} + \left(\frac{\alpha_j U_j^{[\vartheta]}}{r_i^{[\vartheta]} - r_j^{[\vartheta]}} \right)^2 \right). \tag{29}$$

Replacing $r_j^{[\vartheta]}$ by $z_j^{[\vartheta]}$, $U_j^{[\vartheta]}$ by $\breve{U}_j^{[\vartheta]}$ in (29) and using in (23), we have

$$r_i^{[\vartheta+1]} = r_i^{[\vartheta]} - \frac{\alpha_i}{\frac{1}{\breve{U}_i^{[\vartheta]}} - \sum_{\substack{j=1 \\ j\neq i}}^{m}\left(\frac{\alpha_j}{r_i^{[\vartheta]}-z_j^{[\vartheta]}}\right) + \begin{pmatrix} \sum_{\substack{j=1 \\ j\neq i}}^{m}\left(\frac{\alpha_j^2\breve{U}_j^{[\vartheta]}}{\left(r_i^{[\vartheta]}-z_j^{[\vartheta]}\right)^2}\right) + \\ \sum_{\substack{j=1 \\ j\neq i}}^{m}\left(\frac{\alpha_j^3\left(\breve{U}_j^{[\vartheta]}\right)^2}{\left(r_i^{[\vartheta]}-z_j^{[\vartheta]}\right)^3}\right) \end{pmatrix}}. \tag{30}$$

We name the method introduced in (30) as the SFM$^{\sigma}$-Method. Now, we calculate the convergence order of the SFM$^{\sigma}$-Method. Firstly, we introduce some notations as:

$$d = \min_{\substack{i,j \\ i\neq j}}\left|\xi_i - \xi_j\right|, \quad g = \frac{2n-1}{d} \text{ and } \sum\sum_{j\neq i} \text{ instead of } \sum_{j=1}^{m}\sum_{j\neq i}^{m} \text{ respectively.} \tag{31}$$

Now, we suppose the condition

$$\epsilon_i < \frac{d}{2n-1} = \frac{1}{g^{\varsigma}}, \tag{32}$$

where $i = (1, 2, \ldots, m)$. The conditions hold for each, where $r_i^{[\vartheta]} - \xi_i = \epsilon_i$.

Convergence Analysis: Here, we prove the following lemma:

Lemma 1. *Let $r_1, r_2, \ldots, r_m$ be reasonably close approximations of roots $\xi_1, \xi_2, \ldots, \xi_m$, respectively. Let $r_i^{[\vartheta]} - \xi_i = \epsilon_i$, $\epsilon_i' = r_i^{[\vartheta+1]} - \xi_i$, where $r_1^{[\vartheta+1]}, r_2^{[\vartheta+1]}, \ldots, r_m^{[\vartheta+1]}$ are the new approximations produced by the iterative SFM$^{\sigma}$. If (32) is satisfied, then the following estimate is also true:*
(i) $\left|\epsilon_i'\right| \leq \frac{g^4}{n-1}\left|\epsilon_i\right|^2\sum_{j\neq i}\left|\epsilon_j\right|^{3\varsigma+3}$,
(ii) $\left|\epsilon_i'\right| < \frac{d}{2n-1} = \frac{1}{g^{\varsigma}}, (i = 1, 2, \ldots, m)$.

Proof. Taking into account (31), we find:

$$\left|r_i^{[\vartheta]} - \xi_j\right| = \left|(\xi_i - \xi_j) + (r_i^{[\vartheta]} - \xi_i)\right| \geq \left|(\xi_i - \xi_j)\right| - \left|(r_i^{[\vartheta]} - \xi_i)\right| > d - \frac{d}{2n-1}, \tag{33}$$

$$= d\left(1 - \frac{1}{2n-1}\right) = d\left(\frac{2n-2}{2n-1}\right) \tag{34}$$

Using (34), we obtain:

$$\left|r_i^{[\vartheta]} - \xi_j\right| = \frac{2n-2}{g} \tag{35}$$

Now, considering (33) and (34), we have:

$$\left|r_i^{[\vartheta]} - z_j^{[\vartheta]}\right| = \left|(r_i^{[\vartheta]} - \xi_j) + (\xi_j - z_j^{[\vartheta]})\right| \geq \left|r_i^{[\vartheta]} - \xi_j\right| - \left|\xi_j - z_j^{[\vartheta]}\right|, \tag{36}$$

$$\left|r_i^{[\vartheta]} - z_j^{[\vartheta]}\right| = \left|r_i^{[\vartheta]} - \xi_j\right| - \left|\xi_j - z_j^{[\vartheta]}\right| > \frac{2n-2}{g} - \frac{1}{g} = \frac{2n-3}{g}. \tag{37}$$

Now, we introduce some new notations:

$$\sum_{1,i} = \sum_{j\neq i}\frac{1}{r_i^{[\vartheta]} - \xi_j}, \tag{38}$$

$$\sum_{1,i}\alpha_j = \sum_{j\neq i}\frac{\alpha_j}{r_i^{[\vartheta]} - \xi_j}, \tag{39}$$

Thus, using (35), we obtain:

$$\left|\sum_{1,i}\alpha_j\right| \leq \sum_{j\neq i}\frac{\alpha_j}{|r_i^{[\vartheta]}-\xi_j|} < \frac{g}{2(n-1)}\sum_{j\neq i}\alpha_j, \tag{40}$$

$$\left|\sum_{1,i}\alpha_j\right| \leq \sum_{j\neq i}\frac{\alpha_j}{|r_i^{[\vartheta]}-\xi_j|} < \frac{g}{2(n-1)}(n-\alpha_j), \tag{41}$$

as $n-\alpha_j < n-1$ for every i, this implies

$$\left|\sum_{1,i}\alpha_j\right| \leq \sum_{j\neq i}\frac{\alpha_j}{|r_i^{[\vartheta]}-\xi_j|} < \frac{g(n-1)}{2(n-1)} = \frac{g}{2}, \tag{42}$$

$$\frac{1}{U_j^{[\vartheta]}} = \frac{f(r_i^{[\vartheta]})}{f(r_i^{[\vartheta]})} = \sum_{1,i}\frac{\alpha_j}{r_i^{[\vartheta]}-\xi_j} = \frac{\alpha_i}{r_i^{[\vartheta]}-\xi_j} + \sum_{1,i}\alpha_j. \tag{43}$$

As $r_i^{[\vartheta]}-\xi_i = \epsilon_i$,

$$\frac{1}{\breve{U}_i^{[\vartheta]}} = \frac{\alpha_i}{\epsilon_i^{\varsigma+1}} + \sum_{1,i}\alpha_j = \frac{\alpha_i+\epsilon_i^{\varsigma+1}\sum_{1,i}\alpha_j}{\epsilon_i^{\varsigma+1}}, \tag{44}$$

$$\breve{U}_i^{[\vartheta]} = \frac{\epsilon_i^{\varsigma+1}}{\alpha_i+\epsilon_i^{\varsigma+1}\sum_{1,i}\alpha_j}. \tag{45}$$

Using (32) and (35) in the above result, we have:

$$\left|\breve{U}_i^{[\vartheta]}\right| = \left|\frac{\epsilon_i^{\varsigma+1}}{\alpha_i+\epsilon_i^{\varsigma+1}\sum_{1,i}\alpha_j}\right| < \frac{|\epsilon_i^{\varsigma+1}|}{|\alpha_i|+\left|\epsilon_i^{\varsigma+1}\right|\left|\sum_{1,i}\alpha_j\right|} < \frac{2}{g^{\varsigma+1}}. \tag{46}$$

As $\alpha_i > 1, |\epsilon_i| = \frac{1}{g^{\varsigma+1}}$

$$\left|\breve{U}_i^{[\vartheta]}\right| < \frac{\frac{1}{g^{\varsigma+1}}}{1-\frac{1}{g}\frac{g}{2}} = \frac{2}{g^{\varsigma+1}}. \tag{47}$$

From (34), we have $\epsilon_i = r_i^{[\vartheta]}-\xi_i$, $\epsilon_i' = r_i^{[\vartheta+1]}-\xi_i$. Therefore,

$$\epsilon_i' = r_i^{[\vartheta+1]}-\xi_i = r_i^{[\vartheta]} - \frac{\alpha_i}{\frac{1}{\breve{U}_i^{[\vartheta]}} - \sum_{j\neq i}\frac{\alpha_j}{r_i^{[\vartheta]}-z_j^{[\vartheta]}} + \sum_{j\neq i}\frac{\alpha_j^2\breve{U}_j^{[\vartheta]}}{\left(r_i^{[\vartheta]}-z_j^{[\vartheta]}\right)^2} - \sum_{j\neq i}\frac{\alpha_j^3\left(\breve{U}_j^{[\vartheta]}\right)^2}{\left(r_i^{[\vartheta]}-z_j^{[\vartheta]}\right)^3}} - \xi_i, \tag{48}$$

$$\epsilon_i' = \epsilon_i - \frac{\alpha_i}{\frac{1}{\breve{U}_i^{[\vartheta]}} - \sum_{j\neq i}\frac{\alpha_j}{r_i^{[\vartheta]}-z_j^{[\vartheta]}} + \sum_{j\neq i}\frac{\alpha_j^2\breve{U}_j^{[\vartheta]}}{\left(r_i^{[\vartheta]}-z_j^{[\vartheta]}\right)^2} - \sum_{j\neq i}\frac{\alpha_j^3\left(\breve{U}_j^{[\vartheta]}\right)^2}{\left(r_i^{[\vartheta]}-z_j^{[\vartheta]}\right)^3}}, \tag{49}$$

and therefore,

$$\epsilon_i' = \epsilon_i - \frac{\alpha_i}{\frac{\alpha_i}{\epsilon_i} + \sum_{j\neq i}\frac{\alpha_j}{r_i^{[\vartheta]}-\xi_j} - \sum_{j\neq i}\frac{\alpha_j}{r_i^{[\vartheta]}-z_j^{[\vartheta]}} + \sum_{j\neq i}\frac{\alpha_j^2\breve{U}_j^{[\vartheta]}}{\left(r_i^{[\vartheta]}-z_j^{[\vartheta]}\right)^2} - \sum_{j\neq i}\frac{\alpha_j^3\left(\breve{U}_j^{[\vartheta]}\right)^2}{\left(r_i^{[\vartheta]}-z_j^{[\vartheta]}\right)^3}}, \tag{50}$$

$$\epsilon_i' = \epsilon_i - \frac{\alpha_i \epsilon_i}{\alpha_i + \epsilon_i \sum_{j \neq i} \alpha_j \left[\frac{1}{r_i^{[\vartheta]} - \xi_j} - \frac{1}{r_i^{[\vartheta]} - z_j^{[\vartheta]}} + \frac{\alpha_j \breve{U}_j^{[\vartheta]}}{\left(r_i^{[\vartheta]} - z_j^{[\vartheta]}\right)^2} - \frac{\alpha_j^2 \left(\breve{U}_j^{[\vartheta]}\right)^2}{\left(r_i^{[\vartheta]} - z_j^{[\vartheta]}\right)^3} \right]}, \tag{51}$$

$$\epsilon_i' = \epsilon_i - \frac{\alpha_i \epsilon_i}{\alpha_i + \epsilon_i \sum_{j \neq i} \alpha_j \left[\frac{r_i^{[\vartheta]} - z_j^{[\vartheta]} - (r_i^{[\vartheta]} - \xi_j)}{\left(r_i^{[\vartheta]} - \xi_j\right)\left(r_i^{[\vartheta]} - z_j^{[\vartheta]}\right)} + \frac{\alpha_j \breve{U}_j^{[\vartheta]}}{\left(r_i^{[\vartheta]} - z_j^{[\vartheta]}\right)^2} - \frac{\alpha_j^2 \left(\breve{U}_j^{[\vartheta]}\right)^2}{\left(r_i^{[\vartheta]} - z_j^{[\vartheta]}\right)^3} \right]}, \tag{52}$$

$$\epsilon_i' = \epsilon_i - \frac{\alpha_i \epsilon_i}{\alpha_i + \epsilon_i \sum_{j \neq i} \alpha_j \left[\frac{r_i^{[\vartheta]} - z_j^{[\vartheta]} - r_i^{[\vartheta]} + \xi_j}{\left(r_i^{[\vartheta]} - \xi_j\right)\left(r_i^{[\vartheta]} - z_j^{[\vartheta]}\right)} + \frac{\alpha_j \breve{U}_j^{[\vartheta]}}{(r_i^{[\vartheta]} - z_j^{[\vartheta]})^2} - \frac{\alpha_j^2 \left(\breve{U}_j^{[\vartheta]}\right)^2}{\left(r_i^{[\vartheta]} - z_j^{[\vartheta]}\right)^3} \right]}, \tag{53}$$

$$\epsilon_i' = \epsilon_i - \frac{\alpha_i \epsilon_i}{\alpha_i + \epsilon_i \sum_{j \neq i} \alpha_j \left[\frac{\xi_j - z_j^{[\vartheta]}}{\left(r_i^{[\vartheta]} - \xi_j\right)\left(r_i^{[\vartheta]} - z_j^{[\vartheta]}\right)} + \frac{\alpha_j \breve{U}_j^{[\vartheta]}}{\left(r_i^{[\vartheta]} - z_j^{[\vartheta]}\right)^2} - \frac{\alpha_j^2 \left(\breve{U}_j^{[\vartheta]}\right)^2}{\left(r_i^{[\vartheta]} - z_j^{[\vartheta]}\right)^3} \right]}, \tag{54}$$

using Newton's correction, we obtain,

$$\breve{U}_j^{[\vartheta]} = \frac{\epsilon_j^{\varsigma+1}}{\alpha_j + \epsilon_j^{\varsigma+1} \sum_{1,i} \alpha_i}, \tag{55}$$

$$\epsilon_i' = \epsilon_i - \frac{\alpha_i \epsilon_i}{\alpha_i + \epsilon_i \sum_{j \neq i} \alpha_j \left[\begin{array}{c} \frac{-\epsilon_j^{\varsigma+1}}{\left(r_i^{[\vartheta]} - \xi_j\right)\left(r_i^{[\vartheta]} - z_j^{[\vartheta]}\right)} + \frac{\alpha_j \epsilon_j^{\varsigma+1}}{(r_i^{[\vartheta]} - z_j^{[\vartheta]})^2 \left(\alpha_j + \epsilon_j^{\varsigma+1} \sum_{1,i} \alpha_i\right)} \\ - \frac{\alpha_j^2 \left(\epsilon_j^{\varsigma+1}\right)^2}{(r_i^{[\vartheta]} - z_j^{[\vartheta]})^3 \left(\alpha_j + \epsilon_j^{\varsigma+1} \sum_{1,i} \alpha_i\right)^2} \end{array} \right]}, \tag{56}$$

$$\epsilon_i' = \epsilon_i - \frac{\alpha_i \epsilon_i}{\alpha_i + \epsilon_i \sum_{j \neq i} \alpha_j \epsilon_j^{\varsigma+1} \left[\begin{array}{c} \frac{-1}{\left(r_i^{[\vartheta]} - \xi_j\right)\left(r_i^{[\vartheta]} - z_j^{[\vartheta]}\right)} + \frac{\alpha_j}{(r_i^{[\vartheta]} - z_j^{[\vartheta]})^2 \left(\alpha_j + \epsilon_j^{\varsigma+1} \sum_{1,i} \alpha_i\right)} \\ - \frac{\alpha_j^2 \epsilon_j}{(r_i^{[\vartheta]} - z_j^{[\vartheta]})^3 \left(\alpha_j + \epsilon_j^{\varsigma+1} \sum_{1,i} \alpha_i\right)^2} \end{array} \right]}, \tag{57}$$

$$\epsilon_i' = \epsilon_i - \frac{\epsilon_i}{1 + \frac{\epsilon_i}{\alpha_i} \sum_{j \neq i} \alpha_j \epsilon_j^{\varsigma+1} \left[\begin{array}{c} \frac{-1}{\left(r_i^{[\vartheta]} - \xi_j\right)\left(r_i^{[\vartheta]} - z_j^{[\vartheta]}\right)} + \frac{\alpha_j}{(r_i^{[\vartheta]} - z_j^{[\vartheta]})^2 \left(\alpha_j + \epsilon_j^{\varsigma+1} \sum_{1,i} \alpha_i\right)} \\ - \frac{\alpha_j^2 \epsilon_j}{(r_i^{[\vartheta]} - z_j^{[\vartheta]})^3 \left(\alpha_j + \epsilon_j^{\varsigma+1} \sum_{1,i} \alpha_i\right)^2} \end{array} \right]}, \tag{58}$$

$$\epsilon_i' = \epsilon_i - \frac{\epsilon_i}{1 + \frac{\epsilon_i}{\alpha_i} \sum_{j \neq i} \alpha_j \epsilon_j^{\varsigma+1} \left[\begin{array}{c} \frac{-(r_i^{[\vartheta]} - z_j^{[\vartheta]})(\alpha_j + \epsilon_j^{\varsigma+1} \sum_{1,i} \alpha_i) + \alpha_j \left(r_i^{[\vartheta]} - \xi_j\right)}{\left(r_i^{[\vartheta]} - \xi_j\right)\left(r_i^{[\vartheta]} - z_j^{[\vartheta]}\right)^2 \left(\alpha_j + \epsilon_j^{\varsigma+1} \sum_{1,i} \alpha_i\right)} \\ - \frac{\alpha_j^2 \epsilon_j^{\varsigma+1}}{\left(r_i^{[\vartheta]} - z_j^{[\vartheta]}\right)^3 \left(\alpha_j + \epsilon_j^{\varsigma+1} \sum_{1,i} \alpha_i\right)^2} \end{array} \right]}, \tag{59}$$

$$\epsilon_i' = \epsilon_i - \frac{\epsilon_i}{1 + \frac{\epsilon_i}{\alpha_i} \sum_{j \neq i} \alpha_j \epsilon_j^{\varsigma+1} \left[\begin{array}{c} \frac{-(r_i^{[\vartheta]} - z_j^{[\vartheta]}) \epsilon_j^{\varsigma+1} \sum_{1,i} \alpha_i - \alpha_j (r_i^{[\vartheta]} - z_j^{[\vartheta]}) + \alpha_j \left(r_i^{[\vartheta]} - \xi_j\right)}{\left(r_i^{[\vartheta]} - \xi_j\right)\left(r_i^{[\vartheta]} - z_j^{[\vartheta]}\right)^2 \left(\alpha_j + \epsilon_j \sum_{1,i} \alpha_i\right)} \\ - \frac{\alpha_j^2 \epsilon_j}{\left(r_i^{[\vartheta]} - z_j^{[\vartheta]}\right)^3 \left(\alpha_j + \epsilon_j \sum_{1,i} \alpha_i\right)^2} \end{array} \right]}, \tag{60}$$

$$\epsilon_i' = \epsilon_i - \frac{\epsilon_i}{1 + \frac{\epsilon_i}{\alpha_i}\sum_{j\neq i}\alpha_j\epsilon_j^{\varsigma+1}\left[\begin{array}{c}\frac{-(r_i^{[\vartheta]}-z_j^{[\vartheta]})\epsilon_j\sum_{1,i}\alpha_i+\alpha_j\left(r_i^{[\vartheta]}-\xi_j-r_i^{[\vartheta]}-z_j^{[\vartheta]}\right)}{\left(r_i^{[\vartheta]}-\xi_j\right)\left(r_i^{[\vartheta]}-z_j^{[\vartheta]}\right)^2\left(\alpha_j+\epsilon_j\sum_{1,i}\alpha_i\right)} \\ -\frac{\alpha_j^2\epsilon_j}{\left(r_i^{[\vartheta]}-z_j^{[\vartheta]}\right)^3\left(\alpha_j+\epsilon_j\sum_{1,i}\alpha_i\right)^2}\end{array}\right]}, \tag{61}$$

$$\epsilon_i' = \epsilon_i - \frac{\epsilon_i}{1 + \frac{\epsilon_i}{\alpha_i}\sum_{j\neq i}\alpha_j\epsilon_j^{\varsigma+1}\left[\begin{array}{c}\frac{-\left(r_i^{[\vartheta]}-z_j^{[\vartheta]}\right)\epsilon_j^{\varsigma+1}\sum_{1,i}\alpha_i-\alpha_j\epsilon_j^{\varsigma+1}}{\left(r_i^{[\vartheta]}-\xi_j\right)\left(r_i^{[\vartheta]}-z_j^{[\vartheta]}\right)^2\left(\alpha_j+\epsilon_j^{\varsigma+1}\sum_{1,i}\alpha_i\right)} \\ -\frac{\alpha_j^2\epsilon_j^{\varsigma+1}}{\left(r_i^{[\vartheta]}-z_j^{[\vartheta]}\right)^3\left(\alpha_j+\epsilon_j^{\varsigma+1}\sum_{1,i}\alpha_i\right)^2}\end{array}\right]}, \tag{62}$$

$$\epsilon_i' = \epsilon_i - \frac{\epsilon_i}{1 + \frac{\epsilon_i}{\alpha_i}\sum_{j\neq i}\alpha_j\epsilon_j^{2\varsigma+2}\left[\begin{array}{c}\frac{\alpha_j-\left(r_i^{[\vartheta]}-z_j^{[\vartheta]}\right)\sum_{1,i}\alpha_i}{\left(r_i^{[\vartheta]}-\xi_j\right)\left(r_i^{[\vartheta]}-z_j^{[\vartheta]}\right)^2\left(\alpha_j+\epsilon_j^{\varsigma+1}\sum_{1,i}\alpha_i\right)} \\ -\frac{\alpha_j^2}{\left(r_i^{[\vartheta]}-z_j^{[\vartheta]}\right)^3\left(\alpha_j+\epsilon_j^{\varsigma+1}\sum_{1,i}\alpha_i\right)^2}\end{array}\right]}, \tag{63}$$

$$\epsilon_i' = \epsilon_i - \frac{\epsilon_i}{1 + \frac{\epsilon_i}{\alpha_i}\sum_{j\neq i}\alpha_j\epsilon_j^{2\varsigma+2}B_{ij}^*}, \tag{64}$$

where,

$$B_{ij}^* = \frac{\alpha_j - \left(r_i^{[\vartheta]}-z_j^{[\vartheta]}\right)\sum_{1,i}\alpha_i}{\left(r_i^{[\vartheta]}-\xi_j\right)\left(r_i^{[\vartheta]}-z_j^{[\vartheta]}\right)^2\left(\alpha_j+\epsilon_j^{\varsigma+1}\sum_{1,j}\alpha_i\right)} - \frac{\alpha_j^2}{\left(r_i^{[\vartheta]}-z_j^{[\vartheta]}\right)^3\left(\alpha_j+\epsilon_j^{\varsigma+1}\sum_{1,j}\alpha_i\right)^2},$$

$$B_{ij}^* = \frac{1}{\left(r_i^{[\vartheta]}-z_j^{[\vartheta]}\right)^2\left(\alpha_j+\epsilon_j^{\varsigma+1}\sum_{1,j}\alpha_i\right)} * \left[\frac{\alpha_j-\left(r_i^{[\vartheta]}-z_j^{[\vartheta]}\right)\sum_{1,i}\alpha_i}{\left(r_i^{[\vartheta]}-\xi_j\right)} - \frac{\alpha_j^2}{\left(r_i^{[\vartheta]}-z_j^{[\vartheta]}\right)\left(\alpha_j+\epsilon_j^{\varsigma+1}\sum_{1,j}\alpha_i\right)}\right], \tag{65}$$

$$B_{ij}^* = \frac{1}{\left(r_i^{[\vartheta]}-z_j^{[\vartheta]}\right)^2\left(\alpha_j+\epsilon_j^{\varsigma+1}\sum_{1,j}\alpha_i\right)} * \left[\frac{\left(\begin{array}{c}\alpha_j\left(\alpha_j+\epsilon_j^{\varsigma+1}\sum_{1,j}\alpha_i\right)\left(r_i^{[\vartheta]}-z_j^{[\vartheta]}\right)- \\ \left(r_i^{[\vartheta]}-z_j^{[\vartheta]}\right)^2\sum_{1,j}\alpha_i\left(\alpha_j+\epsilon_j^{\varsigma+1}\sum_{1,j}\alpha_i\right)-\alpha_j^2\left(r_i^{[\vartheta]}-\xi_j\right)\end{array}\right)}{\left(r_i^{[\vartheta]}-\xi_j\right)\left(r_i^{[\vartheta]}-z_j^{[\vartheta]}\right)\left(\alpha_j+\epsilon_j^{\varsigma+1}\sum_{1,j}\alpha_i\right)}\right], \tag{66}$$

$$B_{ij}^* = \frac{\left[\begin{array}{c}\alpha_j^2\left(r_i^{[\vartheta]}-z_j^{[\vartheta]}\right)+\alpha_j\epsilon_j^{\varsigma+1}\sum_{1,j}\alpha_i\left(r_i^{[\vartheta]}-z_j^{[\vartheta]}\right)- \\ \left(r_i^{[\vartheta]}-z_j^{[\vartheta]}\right)^2\sum_{1,j}\alpha_i\left(\alpha_j+\epsilon_j^{\varsigma+1}\sum_{1,j}\alpha_i\right)-\alpha_j^2\left(r_i^{[\vartheta]}-\xi_j\right)\end{array}\right]}{\left(r_i^{[\vartheta]}-\xi_j\right)\left(r_i^{[\vartheta]}-z_j^{[\vartheta]}\right)^3\left(\alpha_j+\epsilon_j^{\varsigma+1}\sum_{1,j}\alpha_i\right)^2}, \tag{67}$$

$$B_{ij}^{*}=\frac{\left[\begin{array}{c}-\alpha_j^2\epsilon_j^{\varsigma+1}+\alpha_j\epsilon_j^{\varsigma+1}\sum_{1,j}\alpha_i\left(r_i^{[\vartheta]}-z_j^{[\vartheta]}\right)-\\ \left(r_i^{[\vartheta]}-z_j^{[\vartheta]}\right)^2\sum_{1,j}\alpha_i\left(\alpha_j+\epsilon_j^{\varsigma+1}\sum_{1,j}\alpha_i\right)\end{array}\right]}{\left(r_i^{[\vartheta]}-\xi_j\right)\left(r_i^{[\vartheta]}-z_j^{[\vartheta]}\right)^3\left(\alpha_j+\epsilon_j^{\varsigma+1}\sum_{1,j}\alpha_i\right)^2}. \tag{68}$$

From (55), we get:

$$\left|\breve{U}_j^{[\vartheta]}\right|=\left|\frac{\epsilon_j^{\varsigma+1}}{\alpha_j+\epsilon_j^{\varsigma+1}\sum_{1,j}\alpha_i}\right|<\frac{2}{g^{\varsigma+1}}, \tag{69}$$

$$\left|\breve{U}_j^{[\vartheta]}\right|=\left|\frac{1}{\alpha_j+\epsilon_j^{\varsigma+1}\sum_{1,j}\alpha_i}\right|<\frac{2}{g^{\varsigma+1}|\epsilon_j|^{\varsigma+1}}, \tag{70}$$

Now, applying Equations (31), (32) and (39) in the above relation in (70), we obtain:

$$B_{ij}^{*}\leq\frac{\left[\begin{array}{c}|\alpha_j|^2|\epsilon_j^{\varsigma+1}|+|\alpha_j||\epsilon_j^{\varsigma+1}||\sum_{1,j}\alpha_i||r_i^{[\vartheta]}-z_j^{[\vartheta]}|\\ +|r_i^{[\vartheta]}-z_j^{[\vartheta]}|^2|\sum_{1,j}\alpha_i||\alpha_j+\epsilon_j^{\varsigma+1}\sum_{1,j}\alpha_i|\end{array}\right]}{|r_i^{[\vartheta]}-\xi_j||r_i^{[\vartheta]}-z_j^{[\vartheta]}|^3|\alpha_j+\epsilon_j^{\varsigma+1}\sum_{1,j}\alpha_i|^2}, \tag{71}$$

$$B_{ij}^{*}\leq\frac{n^2|\epsilon_j^{\varsigma+1}|+n|\epsilon_j^{\varsigma+1}|\frac{g}{2}\left(\frac{2n-3}{g}\right)+\left(\frac{2n-3}{g}\right)^2\frac{g}{2}\left(\frac{g}{2}\left|\epsilon_j^{\varsigma+1}\right|\right)}{\left(\frac{2n-3}{g}\right)\left(\frac{2n-3}{g}\right)^3\left(\frac{g}{2\left|\epsilon_j^{\varsigma+1}\right|}\right)^2}, \tag{72}$$

$$B_{ij}^{*}\leq\frac{|\epsilon_j^{\varsigma+1}|\left[n^2+\frac{n(2n-3)}{2}+\frac{(2n-3)^2}{4}\right]}{2\left(\frac{n-1}{4g^2}\right)(2n-3)^3|\epsilon_j^{\varsigma+1}|^2}, \tag{73}$$

Therefore, $B_{ij}^{*}\leq B_{ij}*|\epsilon_j^{\varsigma+1}|$, where

$$B_{ij}\leq\frac{\left[n^2+\frac{n(2n-3)}{2}+\frac{(2n-3)^2}{4}\right]}{2\left(\frac{n-1}{4g^2}\right)(2n-3)^3|\epsilon_j^{\varsigma+1}|^2}, \tag{74}$$

and therefore,

$$\epsilon_i'=\epsilon_i-\frac{\epsilon_i}{1+\frac{\epsilon_i}{\alpha_i}\sum_{j\neq i}\alpha_j\epsilon_j^{2\varsigma+2}B_{ij}*|\epsilon_j^{\varsigma+1}|}, \tag{75}$$

$$\epsilon_i'=\epsilon_i-\frac{\epsilon_i}{1+\frac{\epsilon_i}{\alpha_i}\sum_{j\neq i}\alpha_j\epsilon_j^{3\varsigma+3}B_{ij}}, \tag{76}$$

$$\epsilon_i'=\frac{\epsilon_i+\frac{\epsilon_i^2}{\alpha_i}\sum_{j\neq i}\alpha_j\epsilon_j^{3\varsigma+3}B_{ij}-\epsilon_i}{1+\frac{\epsilon_i}{\alpha_i}\sum_{j\neq i}\alpha_j\epsilon_j^{3\varsigma+3}B_{ij}}=\frac{\frac{\epsilon_i^2}{\alpha_i}\sum_{j\neq i}\alpha_j\epsilon_j^{3\varsigma+3}B_{ij}}{1+\frac{\epsilon_i}{\alpha_i}\sum_{j\neq i}\alpha_j\epsilon_j^{3\varsigma+3}B_{ij}}, \tag{77}$$

$$B_{ij}\leq\frac{\left[\frac{4n^2+2n(2n-3)+(2n-3)^2}{4}\right]}{2(\frac{n-1}{4q^2})(2n-3)^3|\epsilon_j^{\varsigma+1}|^2}, \tag{78}$$

$$B_{ij}\leq\frac{\left[\frac{4n^2+4n^2-6n+4n^2+9-12n}{4}\right]}{2(\frac{n-1}{4q^2})(2n-3)^3|\epsilon_j^{\varsigma+1}|^2}=\frac{g^2(12n^2-18n+9)}{2(n-1)|\epsilon_j^{\varsigma+1}|^2(2n-3)^3}, \tag{79}$$

$$B_{ij} \leq \frac{g^2}{2(n-1)|\epsilon_j|}.\frac{12n^2-18n+9}{(2n-3)^3}. \tag{80}$$

Since $\frac{(12n^2-18n+9)}{(2n-3)^3}, n \geq 3$ is a monotonically decreasing sequence, let us estimate from the above the absolute values of B_{ij}. We have:

$$B_{ij} < \frac{g^2}{2(n-1)|\epsilon_j^{\varsigma+1}|^2}, \tag{81}$$

$$\left|\frac{\epsilon_i}{\alpha_i}\sum_{j\neq i}\alpha_j\epsilon_j^{3\varsigma+3}B_{ij}\right| \leq |\frac{\epsilon_i}{\alpha_i}|\sum_{j\neq i}\alpha_j|\epsilon_j|^{3\varsigma+3}B_{ij}. \tag{82}$$

Since $\alpha_i \geq 1 \Rightarrow \frac{1}{\alpha_i} \leq 1$ and $\frac{|\epsilon_i|}{\alpha_i} \leq \frac{1}{g}$ for all i,

$$\begin{aligned}
|\frac{\epsilon_i}{\alpha_i}|\sum_{j\neq i}\alpha_j|\epsilon_j|^{3\varsigma+3}B_{ij} &< \frac{|\epsilon_i|}{\alpha_i}\sum_{j\neq i}\alpha_j|\epsilon_j|^{3\varsigma+3}g^2\frac{g^2}{2(n-1)|\epsilon_j^{\varsigma+1}|^2}, \\
&= \frac{|\epsilon_i|}{\alpha_i}\sum_{j\neq i}\alpha_j|\epsilon_j|^{\varsigma+1}\frac{g^2}{2(n-1)} \\
&< \frac{1}{g^{\varsigma+1}}\sum_{j\neq i}\alpha_j\frac{1}{g^{\varsigma+1}}.\frac{g^2}{2(n-1)}, \\
&< \frac{1}{2(n-1)}.
\end{aligned} \tag{83}$$

Using $\sum_{j\neq i}\alpha_j = n-\alpha_i < n-1$ for all i, we obtain:

$$\left|\frac{\epsilon_i}{\alpha_i}\sum_{j\neq i}\alpha_j\epsilon_j^{3\varsigma+3}B_{ij}\right| \leq \frac{1}{2(n-1)}(n-1) = \frac{1}{2}, \tag{84}$$

also from (84)

$$\left|1+\frac{\epsilon_i}{\alpha_i}\sum_{j\neq i}\alpha_j\epsilon_j^{3\varsigma+3}B_{ij}\right| \geq \left|1-\frac{\epsilon_i}{\alpha_i}\sum_{j\neq i}\alpha_j\epsilon_j^{3\varsigma+3}B_{ij}\right| > \frac{1}{2}. \tag{85}$$

Using Equations (81) and (85) in Equation (77), we obtain

$$|\epsilon_i'| < \frac{|\epsilon_i|^2\sum_{j\neq i}\alpha_j|\epsilon_j|^{3\varsigma+3}\frac{g^2}{2(n-1)|\epsilon_j|^{\varsigma+1}}}{\frac{1}{2}} = \frac{g^2}{n-1}|\epsilon_i|^2\sum_{j\neq i}\alpha_j|\epsilon_j|^{2\varsigma+2}. \tag{86}$$

Hence, we have the proof of Lemma 1 (i). Now, from Equation (86), we have $\tilde{\epsilon_i} < \frac{g^2}{n-1}\frac{1}{g^2}\sum_{j\neq i}\alpha_j\frac{1}{g^\varsigma} = \frac{1}{(n-1)g^\varsigma}(n-\alpha_i) = \frac{1}{(n-1)g^\varsigma}(n-1) = \frac{1}{g^\varsigma}$,

$$\epsilon_i' < \frac{d}{2n-1} = \frac{1}{g^\varsigma} \Rightarrow |\epsilon_i'| < \frac{1}{g^\varsigma}, \tag{87}$$

which completes the proof of Lemma 1 (ii). □

Let $r_1^{[0]},\ldots,r_n^{[0]}$ be the good initial guesses to roots $\xi_1,\xi_2,\ldots,\xi_n$ of an algebraic polynomial f and suppose $\epsilon_i^{[\vartheta]} = r_i^{[\vartheta]} - \xi_i$, where $r_1^{[\vartheta]},\ldots,r_n^{[\vartheta]}$ approximations are obtained in the ϑ^{th} iterative step by the simultaneous SFM$^\sigma$-method. Using the conditions of Lemma 1, now we state the main convergence theorem of our SFM$^\sigma$-Method.

Theorem 2. *According to the following assumptions*

$$\epsilon_i^{[0]} = \left| r^{[0]} - \alpha_i \right| < \frac{d}{2n-1} = \frac{1}{g^{\varsigma}} (i = 1, 2, \ldots, m), \tag{88}$$

the iterative formula SFM$^{\sigma}$ is convergent, having convergent order $3\varsigma + 5$.

Proof. In Lemma 1 (i), we develop the results (86) under the assumptions (32). Using the same arguments under condition (88) of theorem 1, we have from (86):

$$|\epsilon_i^{[1]}| \leq \frac{g^4}{n-1} |\epsilon_i^{[0]}|^2 \sum_{j \neq i} \alpha_j |\epsilon_i^{[0]}|^{3\varsigma+3} < \frac{1}{g^{\varsigma}} (i, \ldots, m). \tag{89}$$

So according to Lemma 1 (ii), we have:

$$|\epsilon_i^{[0]}| < \frac{d}{2n-1} = \frac{1}{g} \Rightarrow |\epsilon_i^{[1]}| < \frac{d}{2n-1} = \frac{1}{g^{\varsigma}} (i, \ldots, m). \tag{90}$$

We prove the theorem by mathematical induction; condition (88) implies

$$|\epsilon_i^{[\vartheta+1]}| \leq \frac{g^4}{n-1} |\epsilon_i^{[\vartheta]}|^2 \sum_{j \neq i} \alpha_j |\epsilon_i^{[\vartheta]}|^{3\varsigma+3} < \frac{1}{g^{\varsigma}} (i, \ldots, m), \tag{91}$$

for every , $\vartheta = 0, 1, 2, \ldots$ and $i = 1, 2, \ldots, m$. Using $|\epsilon_i^{[\vartheta]}| = \frac{t_i^{[\vartheta]}}{g^{\varsigma}}$, (91) becomes

$$|t_i^{[\vartheta+1]}| \leq \frac{g^4}{n-1} \frac{|t_i^{[\vartheta]}|^2}{g^2} \sum_{j \neq i} \frac{|t_j^{[\vartheta]}|^3}{g^{\varsigma+3}} < \frac{1}{g^{\varsigma}} (i, \ldots, m), \tag{92}$$

$$|t_i^{[\vartheta+1]}| \leq \frac{|t_i^{[\vartheta]}|^2}{n-1} \sum_{j \neq i} |t_j^{[\vartheta]}|^{3\varsigma+3} < \frac{1}{g^{\varsigma}} (i, \ldots, m). \tag{93}$$

Let $t^{[\vartheta]} = \max_{1 \leq i \leq m} |t_i^{[\vartheta]}|$, then from assumptions (93), it follows that $g^{\varsigma} \left| \epsilon_i^{[0]} \right| = t_i^{[0]} \leq t^{[0]} < 1$. For all $i = 1, 2, \ldots, m$ and from (93), we obtain $t_i^{[\vartheta]} < 1$, for each $\vartheta = 0, 1, 2, \ldots$ and $i = 1, 2, \ldots, m$. Therefore, from (93), we obtain:

$$|t_i^{[\vartheta+1]}| \leq \frac{|t^{[\vartheta]}|^2}{n-1} (n - \alpha_j) |t_j^{[\vartheta]}|^{3\varsigma+3} < \frac{|t^{[\vartheta]}|^2}{n-1} (n-1) |t_j^{[\vartheta]}|^{3\varsigma+3} \leq |t^{[\vartheta]}|^{3\varsigma+5}, \tag{94}$$

which shows that the proposition $\{t_i^{[\vartheta]} : i = 1, 2, \ldots, m\}$ converges to zero. Consequently, the sequence $\{|t_i^{[\vartheta]}|\}$ also converges to zero. That is, $r_i^{[\vartheta]} \to \xi_i$, for all i as ϑ increases. Finally, from (94), it can be concluded that the method (SFM$^{\sigma}$-Method) has convergence order $3\varsigma + 5$. □

4. Computational Analysis of Simultaneous Methods

Global convergence behavior dominates the computing complexity of the simultaneous technique as compared to a simple roots-finding computer algorithm. This implies that the overall complexity of the parallel computer technique for (1) is $O(n^2)$. As presented in [37], the computational efficiency of an iterative method can be estimated using the efficiency index given by

$$EL(m) = \frac{\log \mathbf{r}}{w_{as} AS_m + w_m M_m + w_d D_m}, \tag{95}$$

Applying (95) and the data given in Table 1, we compute the efficiency ratio $\varrho^*(SFM^{\sigma_1} - SFM^{\sigma_4}, SFM^{\sigma})$ [37] as:

$$\varrho^*(SFM^{\sigma_1} - SFM^{\sigma_4}, SFM^{\sigma}) = \left(\frac{SFM^{\sigma_1} - SFM^{\sigma_4}}{SFM^{\sigma}} - 1\right) \times 100. \tag{96}$$

Figure 1a–e graphically illustrate these percentage ratios.

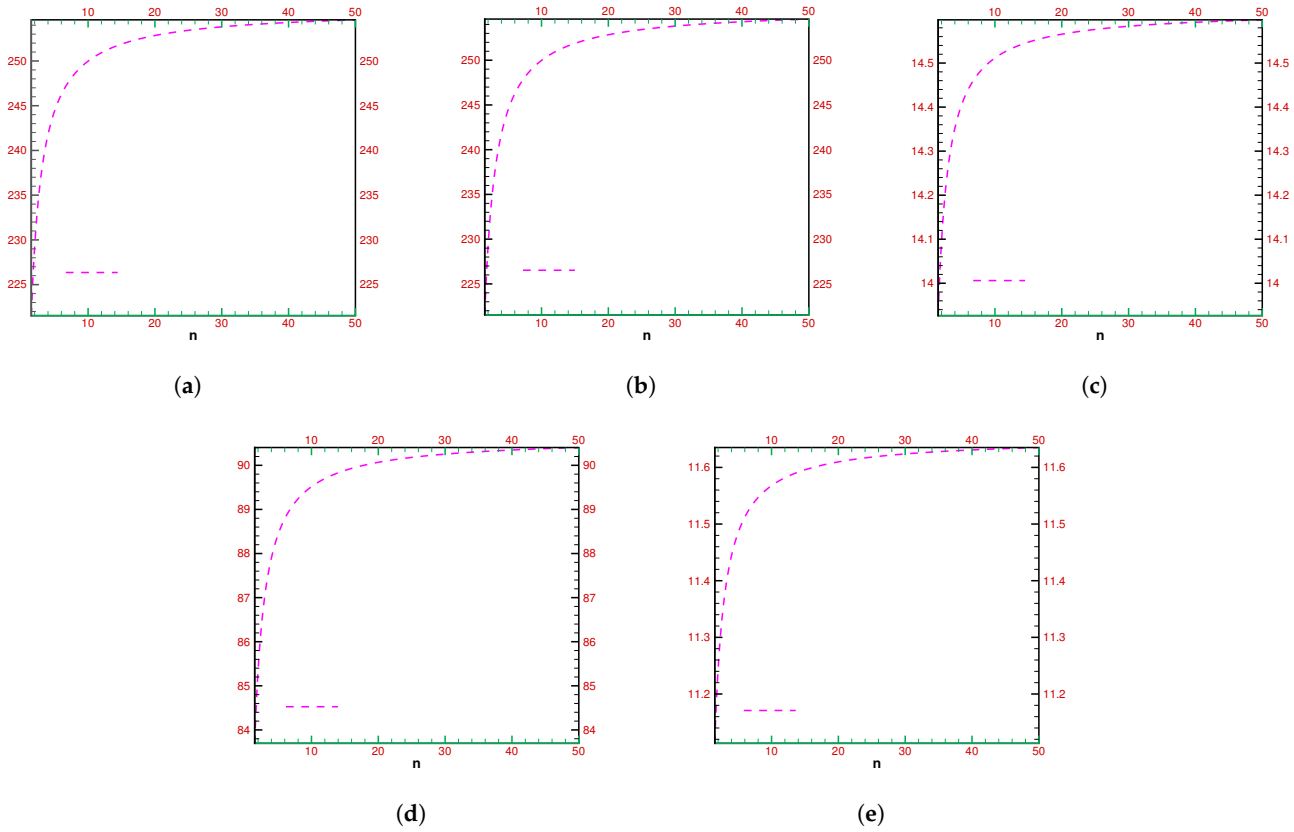

Figure 1. (**a**–**e**) The computational efficiency ratios of fractional simultaneous schemes with respect to each other for different fractional parameter values. (**a**) Computational efficiency ratio of SFM^{σ_1} with respect to SFM^{σ}. (**b**) Computational efficiency ratio of SFM^{σ_2} with respect to SFM^{σ}. (**c**) Computational efficiency ratio of SFM^{σ_3} with respect to SFM^{σ}. (**d**) Computational efficiency ratio of SFM^{σ_4} with respect to SFM^{σ}. (**e**) Computational efficiency ratio of SFM^{σ} with respect to SFM^{σ_1}.

Table 1. Operations per cycle.

Methods	Addition and Subtraction	Multiplications	Divisions
SFM^{σ_1}–SFM^{σ_4}	$5\,m^2 + \Lambda_{11}$	$4\,m^2 + \Lambda_{11}$	$2\,m^2 + \Lambda_{11}$
SFM^{σ}	$5\,m^2 + \Lambda_{11}$	$7\,m^2 + \Lambda_{11}$	$2\,m^2 + \Lambda_{11}$

Here, $\Lambda_{11} = O(m)$, and SFM^{σ} is a simultaneous method for the fractional parameter value equals one.

5. Numerical Outcomes

To compare our recently developed simultaneous methods SFM^{σ_1}–SFM^{σ_4} of order $3\varsigma + 5$ to SFM^{σ}, we look at a few numerical test examples in this section. With Maple 18's 64 digits floating point arithmetic, all calculations were completed. The parallel computer algorithm was terminated based on the following conditions:

$$e_i^{[\vartheta]} = \left| \left(r_i^{[\vartheta+1]} - r_i^{[\vartheta]} \right) \right| < \in = 10^{-30},$$

where $e_i^{[\vartheta]}$ denotes the absolute error of consecutive iterations. In Table, 2-21, the numerical schemes for various fractional parameter values, i.e., 0.1, 0.3, 0.5, 0.8, 1.0, are represented by SFM$^{\sigma_1}$ −SFM$^{\sigma_4}$, SFM$^{\sigma}$ respectively, and B** denotes digits floating point arithmetic. In all tables, we use the following computer terminating criteria (Algorithm 1).

Algorithm 1 For the fractional numerical scheme SFM$^{\sigma}$

For initial estimates $r_i^{[0]}(ii = 1, .., N)$, tolerance $\in > 0$ and set $kk = 0$ for iterations pp

Calculate $z_j^{[\vartheta]} = r_j^{[\vartheta]} - \left(\Gamma(\varsigma+1) \frac{f(r_j^{[\vartheta]})}{[{}_C D_{\varsigma_1}^{\varsigma}] f(r_j^{[\vartheta]})} \right)^{1/\varsigma}$.

Update $r_i^{[\vartheta+1]} = r_i^{[\vartheta]} - \dfrac{\alpha_i}{\left[\begin{array}{l} \frac{1}{\breve{U}_i^{[\vartheta]}} - \sum_{\substack{j=1 \\ j\neq i}}^{m} \left(\frac{\alpha_j}{r_i^{[\vartheta]} - z_j^{[\vartheta]}} \right) + \\ \sum_{\substack{j=1 \\ j\neq i}}^{m} \left(\frac{\alpha_j^2 \breve{U}_j^{[\vartheta]}}{\left(r_i^{[\vartheta]} - z_j^{[\vartheta]} \right)^2} \right) + \sum_{\substack{j=1 \\ j\neq i}}^{m} \left(\frac{\alpha_j^3 \left(\breve{U}_j^{[\vartheta]} \right)^2}{\left(r_i^{[\vartheta]} - z_j^{[\vartheta]} \right)^3} \right) \end{array} \right]}$

$r_i^{[\vartheta+1]} = r_i^{[\vartheta]} (ii = 1, ..., n)$.

if $e_i^{[\vartheta]} = \left| \left(r_i^{[\vartheta+1]} - r_i^{[\vartheta]} \right) \right| < \in = 10^{-30}$ or $\sigma > pp$, then stop.

Set $kk = kk + 1$and go to step 2.

End do.

Engineering Applications

This section presents many problems in engineering whose solutions are approximated by our newly created parallel approaches SFM$^{\sigma_1}$ −SFM$^{\sigma_4}$ and SFM$^{\sigma}$.

Engineering Application 1: Emden–Fowler equation

The Emden–Fowler second-order nonlinear differential equation arises in various fields of physics and engineering, fluid dynamics, heat transfer, and astrophysics, in particular, to model the structure of self-gravitating, spherically symmetric objects, such as stars. The equation is named in honor of Ralph H. Fowler and Robert Emden, two German astrophysicists who made significant contributions to its formulation. The general form of the Emden–Fowler equation is given by [38,39]:

$$\begin{cases} \frac{d^2y}{dr^2} + g_1(r)\frac{dy}{dr} + g_2(r)y(r) + g(r) = y^n, \\ y(0) = 0, y'(0) = 0. \end{cases} \tag{97}$$

Because of its nonlinearity, solving the Emden–Fowler equation is often difficult, and closed-form solutions exists only in specific cases. Choosing $n = 2$, $g_1(r) = \frac{5}{r}$, $g_2(r) = \frac{3}{r^2}$, and $g(r) = r^4 - 15$ in (97), we obtain the following nonlinear initial value problem:

$$\begin{cases} \frac{d^2y}{dr^2} + \frac{5}{r}\frac{dy}{dr} + \frac{3}{r^2}y(r) + g(r) = y^2, \\ y(0) = 0, y'(0) = 0. \end{cases} \tag{98}$$

Using the procedure described in [40], the numerical solution of (98) can be performed by solving the following polynomial:

$$f(r) = r^2 - \frac{79}{28991745} r^{14} + \frac{1306}{57747104415} r^{18}. \tag{99}$$

The Caputo-type derivative of (99) is given as:

$$[{}_C D_{\varsigma_1}^{\varsigma}] f(r) = \frac{\Gamma(3)}{\Gamma(3-\varsigma)} r^{2-\varsigma} - \frac{79}{28991745} \frac{\Gamma(15)}{\Gamma(15-\varsigma)} r^{14-\varsigma} + \frac{1306}{57747104415} \left(\frac{\Gamma(19)}{\Gamma(19-\varsigma)} \right) r^{18-\varsigma}. \tag{100}$$

The exact solution of (99) up to four decimal places is:

$$\begin{aligned}
\zeta_1 &= -3.1792 - 0.2571i, \zeta_2 = -3.1792 + 0.2575i, \zeta_3 = -2.4135 - 1.4797i \\
\zeta_4 &= -2.4135 + 1.4797i, \zeta_5 = -1.4797 - 2.4135i, \zeta_6 = -1.4797 + 2.4135i \\
\zeta_7 &= -0.25758 - 3.1792i, \zeta_8 = -0.2575 + 3.17923i, \zeta_{9,10} = 0.0, \\
\zeta_{11} &= 0.2575 - 3.17923i, \zeta_{12} = 0.2575 + 3.1792i, \zeta_{13} = 1.4797 - 2.4135i \\
\zeta_{14} &= 1.4797 + 2.4135i, \zeta_{15} = 2.41359 - 1.47975i, \zeta_{16} = 2.4135 + 1.4797i \\
\zeta_{17} &= 3.1792 - 0.25758i, \zeta_{18} = 3.1792 + 0.2575i.
\end{aligned}$$

In order to determine the global convergence behavior of the parallel scheme, we generate a random initial guess ranging from $r_{1*}^{[0]}$ to $r_{5*}^{[0]}$ using Matlab as explained in Appendix A Table A1. According to the results presented in Table 2, when an arbitrary starting value is used, SFM$^{\sigma_1}$ −SFM$^{\sigma_4}$, SFM$^{\sigma}$ converges to exact zeros after 19, 17, 13, 10, and 10 iterations for fraction parameters 0.1, 0.3, 0.5, 0.8, and 1.0, respectively. The corresponding CPU times are 2.1254, 1.0874, 1.0078, 0.0784 and 0.0078 as shown in Table 3. The acceleration of the convergence rate of SFM$^{\sigma_1}$ −SFM$^{\sigma_4}$, SFM$^{\sigma}$ as the fractional parameter value increases from 0.1 to 1.0 can be clearly seen in Table 4. Global convergence is demonstrated by the fact that the newly developed method converges to exact roots for randomly generated initial guess values.

Table 2. Experiments using random initial approximation for finding all polynomial roots simultaneously.

$f(r) = r^2 - \frac{79}{28991745}r^{14} + \frac{1306}{57747104415}r^{18}$					
Ini-V	**Number of Iterations**				
$r^{[0]}$	SFM$^{\sigma_1}$	SFM$^{\sigma_2}$	SFM$^{\sigma_3}$	SFM$^{\sigma_4}$	SFM$^{\sigma}$
$r_{1*}^{[0]}$	19.0	17.0	13.0	10.0	10.0
$r_{2*}^{[0]}$	19.0	17.0	13.0	10.0	10.0
$r_{3*}^{[0]}$	19.0	17.0	13.0	10.0	10.0
$r_{4*}^{[0]}$	19.0	17.0	13.0	10.0	10.0
$r_{5*}^{[0]}$	19.0	17.0	13.0	10.0	10.0
Iteration are computed by using 64 D**					

Table 3. CPU-Time using random initial approximation for finding all polynomial roots.

$f(r) = r^2 - \frac{79}{28991745}r^{14} + \frac{1306}{57747104415}r^{18}$					
R-Initial	**Computational CPU-Time in Seconds**				
$r^{[0]}$	SFM$^{\sigma_1}$	SFM$^{\sigma_2}$	SFM$^{\sigma_3}$	SFM$^{\sigma_4}$	SFM$^{\sigma}$
$r_{1*}^{[0]}$	1.0124	1.0012	1.0014	0.0417	0.0045
$r_{2*}^{[0]}$	2.0125	1.0415	1.0045	0.0784	0.0048
$r_{3*}^{[0]}$	2.1254	1.0148	1.0047	0.0745	0.0078
$r_{4*}^{[0]}$	1.5241	1.0874	1.0078	0.0451	0.0071
$r_{5*}^{[0]}$	1.7451	1.0741	1.0078	0.0874	0.0069
Maximum CPU-Time is equal to 2.1254 using 64 D**					

Table 4. Local computational order of convergence using random initial approximation.

$f(r) = r^2 - \frac{79}{28991745}r^{14} + \frac{1306}{57747104415}r^{18}$					
Ini-V	**Local Computational Order of Convergence**				
$r^{[0]}$	SFM$^{\sigma_1}$	SFM$^{\sigma_2}$	SFM$^{\sigma_3}$	SFM$^{\sigma_4}$	SFM$^{\sigma}$
$r^{[0]}_{1*}$	6.0019	6.7215	7.7124	7.9115	7.9414
$r^{[0]}_{2*}$	7.0128	6.4515	7.4236	8.0365	7.5210
$r^{[0]}_{3*}$	4.9917	6.6454	7.0148	7.9914	7.8456
$r^{[0]}_{4*}$	5.8748	5.9874	7.5154	7.7214	8.0147
$r^{[0]}_{5*}$	6.1427	6.8748	7.3174	7.6148	7.4878
Maximum LCOC is equal to 8.0147 using 64 D**					

Table 2 shows the number of iterations of the fractional simultaneous scheme SFM$^{\sigma_1}$−SFM$^{\sigma_4}$, SFM$^{\sigma}$ for different choices of the random initial vector given in Appendix A, Table A1. Table 2 clearly shows that the number of iterations decreased as the fractional parameter values increased from 0.1 to 1.0.

Table 5 shows the maximum error (Max-Err) computed by the fractional simultaneous scheme SFM$^{\sigma_1}$−SFM$^{\sigma_4}$, SFM$^{\sigma}$ for different selections of the random initial vector given in Appendix A Table A1 to approximate all roots of the polynomial equations used in application 1. Table 5 clearly demonstrates that as the fractional parameter values increased from 0.1 to 1.0, the accuracy computed by the simultaneous scheme increased significantly (Figure 2).

Table 5. Maximum error using random initial approximation for finding all polynomial roots.

$f(r) = -2949.604r^4 + 14748.02r^3 - 62295.63648r^2 + 2.229900624 \times 10^5 r - 2.675880749 \times 10^5$					
R-Initial	**Maximum Error**				
$r^{[0]}$	SFM$^{\sigma_1}$	SFM$^{\sigma_2}$	SFM$^{\sigma_3}$	SFM$^{\sigma_4}$	SFM$^{\sigma}$
$r^{[0]}_{1*}$	3.1×10^{-15}	0.1×10^{-18}	6.3×10^{-20}	9.5×10^{-22}	0.2×10^{-29}
$r^{[0]}_{2*}$	0.01×10^{-15}	1.2×10^{-18}	6.5×10^{-24}	8.8×10^{-27}	3.5×10^{-20}
$r^{[0]}_{3*}$	0.1×10^{-14}	0.1×10^{-18}	0.1×10^{-22}	3.8×10^{-25}	1.5×10^{-25}
$r^{[0]}_{4*}$	0.4×10^{-13}	9.5×10^{-18}	9.0×10^{-21}	3.2×10^{-20}	1.2×10^{-25}
$r^{[0]}_{5*}$	7.7×10^{-15}	7.8×10^{-18}	3.9×10^{-21}	0.2×10^{-31}	0.5×10^{-31}
Residual errors are equal to 10^{-30} using 64 D**					

Table 4 shows the approximate local computational order of convergence. The approximate local computational order of convergence increases as the fractional parameter values increase from 0.1 to 1.0.

Table 3 shows the computational CPU time in seconds to approximate all roots of the polynomial equation used in application 1 employing the fractional simultaneous scheme.

The rate of convergence increases as the initial guess values are chosen to be sufficiently close to the exact root of (99) as:

$$\begin{aligned}
r_1^{[0]} &= -3.1 - 0.2i, r_2^{[0]} = -3.1 + 0.2i, r_3^{[0]} = -2.4 - 1.4i \\
r_4^{[0]} &= -2.4 + 1.4i, r_5^{[0]} = -1.4 - 2.4i, r_6^{[0]} = -1.4 + 2.4i \\
r_7^{[0]} &= -0.2 - 3.1i, r_8^{[0]} = -0.2 + 3.1i, r_9^{[0]} = 0.02, r_{10}^{[0]} = 0.01, \\
r_{11}^{[0]} &= 0.2 - 3.1i, r_{12}^{[0]} = 0.2 + 3.1i, r_{13}^{[0]} = 1.4 - 2.4i \\
r_{14}^{[0]} &= 1.4 + 2.4i, r_{15}^{[0]} = 2.4 - 1.4i, r_{16}^{[0]} = 2.4 + 1.4i \\
r_{17}^{[0]} &= 3.1 - 0.2i, r_{18}^{[0]} = 3.1 + 0.2i.
\end{aligned}$$

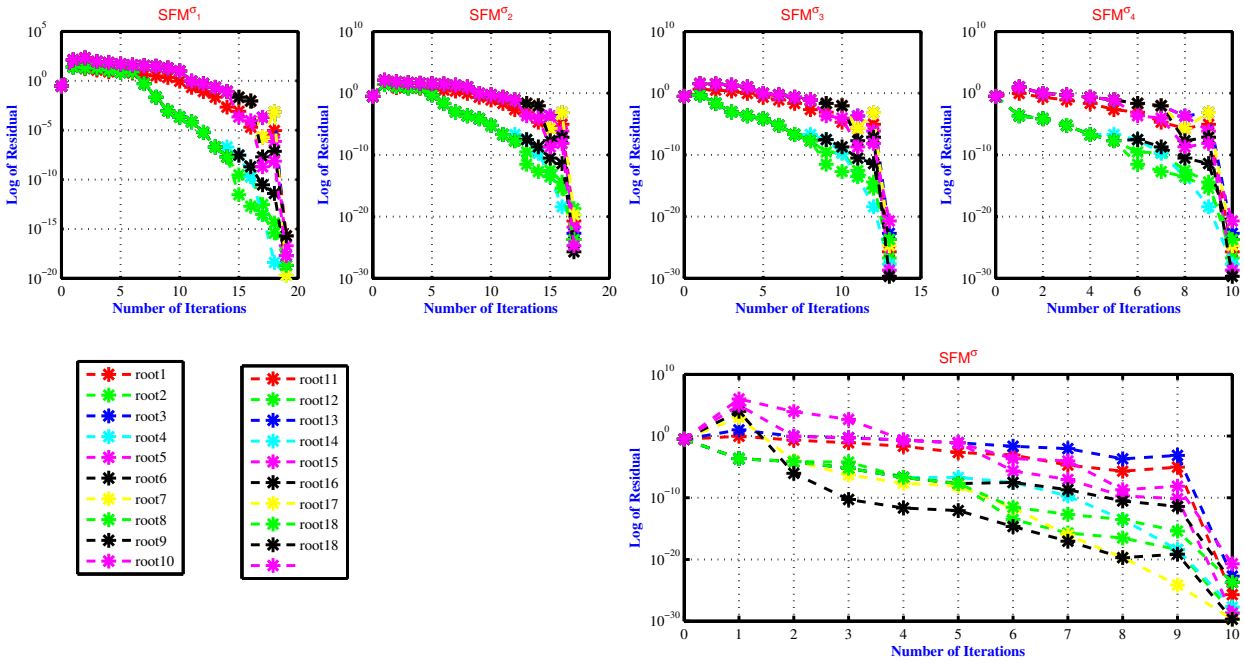

Figure 2. Residual error of the SFM$^{\sigma}$ for approximating all polynomial equation roots used in engineering application 1 for various fractional parameter values, namely $\varsigma = 0.1, 0.3, 0.7, 0.8, 1.0$.

If we start with initial guessed values that are close to the exact root, Table 6 demonstrates that the fractional simultaneous scheme's accuracy and convergence order improve. As the fractional parameter value was increased from 0.1 to 1.0, the residual error calculated using numerical methods also increased.

Engineering Application 2: Under Conservative Force—Mass Spring System

Let us now examine an external force acting on a vibrating mass on a spring. A driving force that causes the spring support to oscillate vertically, for instance, could be represented by $f(r)$. If the mechanical system is conservative, the following nonlinear equation arises [41,42]:

$$\begin{cases} f''(r) + (f'(r))^2 + 32 - 160r = 0, \\ f(0) = 0, f'(0) = 0. \end{cases} \tag{101}$$

Table 6. Computation of all polynomial equation roots.

Methods	**SFM**$^{\sigma_1}$	**SFM**$^{\sigma_2}$	**SFM**$^{\sigma_3}$	**SFM**$^{\sigma_4}$	**SFM**$^{\sigma}$
Error it	$\vartheta = 09$	$\vartheta = 09$	$\vartheta = 07$	$\vartheta = 06$	$\vartheta = 06$
CPU	0.0141	0.016	0.054	0.067	0.065
$e_1^{[\vartheta]}$	2.1×10^{-3}	0.7×10^{-3}	0.3×10^{-12}	0.7×10^{-32}	0.3×10^{-42}
$e_2^{[\vartheta]}$	5.2×10^{-2}	2.6×10^{-4}	9.1×10^{-16}	2.1×10^{-26}	2.8×10^{-64}
$e_3^{[\vartheta]}$	2.1×10^{-2}	3.2×10^{-3}	3.2×10^{-16}	3.2×10^{-36}	3.5×10^{-46}
$e_4^{[\vartheta]}$	3.1×10^{-2}	4.1×10^{-6}	4.1×10^{-16}	4.1×10^{-26}	4.1×10^{-46}
$e_5^{[\vartheta]}$	2.1×10^{-3}	0.7×10^{-3}	0.3×10^{-12}	0.7×10^{-32}	0.3×10^{-42}
$e_6^{[\vartheta]}$	2.0×10^{-3}	0.7×10^{-3}	0.3×10^{-12}	0.7×10^{-32}	0.3×10^{-42}
$e_7^{[\vartheta]}$	0.2×10^{-2}	0.1×10^{-4}	9.1×10^{-16}	4.1×10^{-26}	2.7×10^{-43}
$e_8^{[\vartheta]}$	5.1×10^{-2}	1.2×10^{-5}	1.1×10^{-14}	0.2×10^{-31}	3.0×10^{-42}
$e_9^{[\vartheta]}$	0.1×10^{-3}	0.6×10^{-6}	0.1×10^{-15}	4.1×10^{-26}	6.1×10^{-36}
$e_{10}^{[\vartheta]}$	5.2×10^{-2}	2.9×10^{-4}	9.1×10^{-16}	4.1×10^{-26}	2.7×10^{-43}
$e_{11}^{[\vartheta]}$	0.1×10^{-1}	1.2×10^{-5}	1.1×10^{-14}	0.2×10^{-31}	3.0×10^{-42}
$e_{12}^{[\vartheta]}$	0.2×10^{-1}	3.6×10^{-2}	9.8×10^{-16}	8.1×10^{-26}	2.6×10^{-42}
$e_{13}^{[\vartheta]}$	2.8×10^{-2}	7.2×10^{-3}	3.2×10^{-16}	0.7×10^{-36}	3.3×10^{-45}
$e_{14}^{[\vartheta]}$	1.1×10^{-2}	1.2×10^{-5}	4.1×10^{-14}	4.2×10^{-31}	3.0×10^{-42}
$e_{15}^{[\vartheta]}$	0.1×10^{-3}	0.7×10^{-3}	7.3×10^{-12}	0.7×10^{-32}	1.3×10^{-41}
$e_{16}^{[\vartheta]}$	0.2×10^{-2}	3.6×10^{-4}	9.8×10^{-16}	8.1×10^{-26}	2.6×10^{-42}
$e_{17}^{[\vartheta]}$	2.8×10^{-2}	7.2×10^{-3}	3.2×10^{-16}	0.0×10^{-36}	3.3×10^{-45}
$e_{18}^{[\vartheta]}$	8.1×10^{-2}	0.1×10^{-6}	4.1×10^{-16}	0.1×10^{-26}	4.1×10^{-46}
$\rho_i^{[\vartheta-1]}$	2.1212	2.2312	2.0451	2.5112	3.14212

Using the method described in [40], the following polynomial is used to simulate (101) as:

$$\begin{aligned} f(r) &= -\frac{43359567872}{189}r^{10} + \frac{425993216}{405}r^9 - \frac{278592512}{63}r^8 + \\ &\frac{1024000}{63}r^7 - \frac{4096384}{45}r^6 + 256r^5 \\ &-\frac{6400}{3}r^4 + \frac{16}{3}r^3 - 80r^2. \end{aligned} \tag{102}$$

The Caputo-type derivative of (102) is given as:

$$\begin{aligned} [{}_C D_{\varsigma_1}^{\varsigma}] f(r) &= -\frac{43359567872}{189}\frac{\Gamma(11)}{\Gamma(11-\varsigma)}r^{10-\varsigma} + \frac{425993216}{405}\frac{\Gamma(10)}{\Gamma(10-\varsigma)}r^{9-\varsigma} - \\ &\frac{278592512}{63}\frac{\Gamma(9)}{\Gamma(9-\varsigma)}r^{8-\varsigma} + \frac{1024000}{63}\frac{\Gamma(8)}{\Gamma(8-\varsigma)}r^{7-\varsigma} - \\ &\frac{4096384}{45}\frac{\Gamma(7)}{\Gamma(7-\varsigma)}r^{6-\varsigma} + 256\frac{\Gamma(6)}{\Gamma(6-\varsigma)}r^{5-\varsigma} - \\ &\frac{6400}{3}\frac{\Gamma(5)}{\Gamma(5-\varsigma)}r^{4-\varsigma} + \frac{16}{3}\frac{\Gamma(4)}{\Gamma(4-\varsigma)}r^{3-\varsigma} - 80\frac{\Gamma(3)}{\Gamma(3-\varsigma)}r^{2-\varsigma} \end{aligned} \tag{103}$$

The exact solution up to four decimal places is written as:

$$\begin{aligned} \zeta_1 &= -0.12909 - 0.0824i, \zeta_2 = -0.1290 + 0.0824i, \zeta_3 = -0.0513 - 0.1495i, \\ \zeta_4 &= -0.0513 + 0.1495i, \zeta_{5,6} = 0.0, \zeta_7 = 0.0525 - 0.1493i, \zeta_8 = 0.0525 + 0.1493i, \\ \zeta_9 &= 0.1301 - 0.0822i, \zeta_{10} = 0.13010 + 0.0822i. \end{aligned}$$

To determine the global convergence component of the parallel scheme, use Matlab to generate a random initial guess value ranging from $r_{1*}^{[0]} - r_{5*}^{[0]}$ as specified in Appendix A Table A2. With an arbitrary starting value, SFM$^{\sigma_1}$ −SFM$^{\sigma_4}$, SFM$^{\sigma}$ converges to exact zeros after 19, 16, 14, 10 and 10 iterations as indicated in Table 7 for fraction parameters values 0.1, 0.3, 0.5, 0.8, and 1.0, respectively. As described in Table 8, the corresponding CPU times are 3.1254, 1.0729, 1.0137, 0.0881 and 0.0141, respectively. Table 9 clearly illustrates how the rate of convergence of SFM$^{\sigma_1}$ −SFM$^{\sigma_4}$,SFM$^{\sigma}$ accelerates as the value of the fractional parameter increases from 0.1 to 1.0. The newly developed method converges to exact roots for randomly generated initial guess values, demonstrating its global convergence.

Table 7. Iteration numbers using random initial approximation for finding all polynomial roots.

$f(r) = -\frac{43359567872}{189}r^{10} + \frac{425993216}{405}r^{9} - \frac{278592512}{63}r^{8} + \frac{1024000}{63}r^{7} - \frac{4096384}{45}r^{6} + 256r^{5} - \frac{6400}{3}r^{4} + \frac{16}{3}r^{3} - 80r^{2}.$

Ini-V	**Number of Iterations**				
$r^{[0]}$	SFM$^{\sigma_1}$	SFM$^{\sigma_2}$	SFM$^{\sigma_3}$	SFM$^{\sigma_4}$	SFM$^{\sigma}$
$r_{1*}^{[0]}$	19.0	16.0	14.0	10.0	10.0
$r_{2*}^{[0]}$	19.0	16.0	14.0	10.0	10.0
$r_{3*}^{[0]}$	19.0	16.0	14.0	10.0	10.0
$r_{4*}^{[0]}$	19.0	16.0	14.0	10.0	10.0
$r_{5*}^{[0]}$	19.0	16.0	14.0	10.0	10.0

Residual errors are equal to 10^{-30} using 64 D**

Table 8. CPU-Time using random initial approximation for finding all polynomial roots.

$f(r) = -\frac{43359567872}{189}r^{10} + \frac{425993216}{405}r^{9} - \frac{278592512}{63}r^{8} + \frac{1024000}{63}r^{7} - \frac{4096384}{45}r^{6} + 256r^{5} - \frac{6400}{3}r^{4} + \frac{16}{3}r^{3} - 80r^{2}.$

R-Initial	**Computational CPU-Time in Seconds**				
$r^{[0]}$	SFM$^{\sigma_1}$	SFM$^{\sigma_2}$	SFM$^{\sigma_3}$	SFM$^{\sigma_4}$	SFM$^{\sigma}$
$r_{1*}^{[0]}$	3.0114	1.0012	0.0015	0.0417	0.0031
$r_{2*}^{[0]}$	2.1105	1.0317	1.0080	0.0881	0.0141
$r_{3*}^{[0]}$	3.1254	1.0159	1.0137	0.0765	0.0039
$r_{4*}^{[0]}$	1.4201	1.0678	1.0061	0.0430	0.0067
$r_{5*}^{[0]}$	1.7465	1.0729	1.0070	0.09143	0.0073

Residual errors are equal to 10^{-30} using 64 D**

Table 7 shows the number of iterations of fractional simultaneous scheme SFM$^{\sigma_1}$ −SFM$^{\sigma_4}$, SFM$^{\sigma}$ for different random initial vectors given in Appendix A Table A2. Table 7 clearly shows that the number of iterations decreased as the fractional parameter values increased from 0.1 to 1.0.

Table 9 shows the maximum error (Max-Err) computed by fractional simultaneous scheme SFM$^{\sigma_1}$ −SFM$^{\sigma_4}$, SFM$^{\sigma}$ for different random initial vectors given in Appendix A Table A2 to approximate all the roots of polynomial equations used in application 2. Table 9 clearly demonstrates that as the fractional parameter values increased from 0.1 to 1.0, the accuracy computed by simultaneous scheme increased significantly (Figure 3).

Table 9. Maximum error using random initial approximation for finding all polynomial roots.

$f(r) = -\frac{43359567872}{189}r^{10} + \frac{425993216}{405}r^9 - \frac{278592512}{63}r^8 + \frac{1024000}{63}r^7 - \frac{4096384}{45}r^6 + 256r^5 - \frac{6400}{3}r^4 + \frac{16}{3}r^3 - 80r^2.$					
R-Initial	**Maximum Error**				
$r^{[0]}$	SFM$^{\sigma_1}$	SFM$^{\sigma_2}$	SFM$^{\sigma_3}$	SFM$^{\sigma_4}$	SFM$^{\sigma}$
$r^{[0]}_{1*}$	0.1×10^{-16}	1.1×10^{-18}	6.3×10^{-20}	4.5×10^{-23}	6.1×10^{-25}
$r^{[0]}_{2*}$	2.1×10^{-16}	6.2×10^{-19}	1.5×10^{-24}	7.8×10^{-26}	3.0×10^{-26}
$r^{[0]}_{3*}$	0.1×10^{-15}	6.1×10^{-18}	0.4×10^{-24}	7.8×10^{-26}	2.5×10^{-27}
$r^{[0]}_{4*}$	3.4×10^{-14}	6.5×10^{-18}	0.1×10^{-20}	3.2×10^{-28}	6.2×10^{-25}
$r^{[0]}_{5*}$	9.7×10^{-17}	1.8×10^{-19}	3.2×10^{-21}	9.2×10^{-33}	3.5×10^{-32}
Residual errors are equal to 10^{-30} using 64 D**					

The approximate local computational order of convergence is shown in Table 10. From 0.1 to 1.0, as the fractional parameter values increase, the approximate local computational order of convergence increases.

Table 10. Local computational order of convergence using random initial approximation.

$f(r) = -\frac{43359567872}{189}r^{10} + \frac{425993216}{405}r^9 - \frac{278592512}{63}r^8 + \frac{1024000}{63}r^7 - \frac{4096384}{45}r^6 + 256r^5 - \frac{6400}{3}r^4 + \frac{16}{3}r^3 - 80r^2.$					
Ini-V	**Local Computational Order of Conergence**				
$r^{[0]}$	SFM$^{\sigma_1}$	SFM$^{\sigma_2}$	SFM$^{\sigma_3}$	SFM$^{\sigma_4}$	SFM$^{\sigma}$
$r^{[0]}_{1*}$	5.0119	6.7005	7.7354	7.9195	7.9654
$r^{[0]}_{2*}$	7.0128	6.4515	7.4236	7.4365	8.1210
$r^{[0]}_{3*}$	4.9917	6.6454	7.0148	7.9914	7.8456
$r^{[0]}_{4*}$	5.1708	4.9104	7.5154	7.7114	8.0101
$r^{[0]}_{5*}$	5.1127	5.8018	7.0074	7.0048	7.9018
Residual errors are equal to 10^{-30} using 64 D**					

According to the fractional simultaneous scheme, Table 10 displays the local computational order of convergence needed for the approximation of all roots of the polynomial equation used in application 2. Convergence rates increase as the following initial estimations are sufficiently adjusted to the exact roots of the engineering application 2:

$$\begin{aligned} r_1^{[0]} &= -0.1 - 0.01i, r_2^{[0]} = -0.1 + 0.01i, r_3^{[0]} = -0.01 - 0.1i, \\ r_4^{[0]} &= -0.01 + 0.1i,\ r_5^{[0]} = 0.01,\ r_6^{[0]} - 0.1,\ r_7^{[0]} = 0.05 - 0.14i, \\ r_8^{[0]} &= 0.05 + 0.14i,\ r_9^{[0]} = 0.1 - 0.08i, r_{10}^{[0]} = 0.1 + 0.08i. \end{aligned}$$

are chosen as initial guessed values.

Table 11 shows that the convergence order and accuracy of the fractional simultaneous scheme are increased if we take the initial guessed values close to the exact root. The residual error computed by numerical methods also increased as we increased the fractional parameter value from 0.1 to 1.0.

Table 11. Determination of all polynomial equation roots.

Methods	**SFM$^{\sigma_1}$**	**SFM$^{\sigma_2}$**	**SFM$^{\sigma_3}$**	**SFM$^{\sigma_4}$**	**SFM$^{\sigma}$**
Error it	$\vartheta = 09$	$\vartheta = 09$	$\vartheta = 08$	$\vartheta = 08$	$\vartheta = 05$
CPU	0.0141	0.016	0.054	0.067	0.065
$e_1^{[\vartheta]}$	0.2×10^{-3}	1.7×10^{-4}	1.3×10^{-11}	5.7×10^{-33}	0.3×10^{-42}
$e_2^{[\vartheta]}$	7.2×10^{-2}	2.8×10^{-5}	9.9×10^{-17}	5.5×10^{-27}	2.8×10^{-46}
$e_3^{[\vartheta]}$	2.5×10^{-3}	3.2×10^{-3}	3.2×10^{-16}	1.2×10^{-36}	3.0×10^{-47}
$e_4^{[\vartheta]}$	3.1×10^{-2}	9.9×10^{-7}	4.8×10^{-16}	4.9×10^{-23}	4.1×10^{-46}
$e_5^{[\vartheta]}$	7.7×10^{-3}	6.7×10^{-6}	6.5×10^{-16}	6.5×10^{-26}	6.5×10^{-46}
$e_6^{[\vartheta]}$	2.5×10^{-2}	4.3×10^{-7}	4.8×10^{-17}	4.0×10^{-27}	4.0×10^{-47}
$e_7^{[\vartheta]}$	1.6×10^{-2}	3.0×10^{-6}	3.0×10^{-16}	7.7×10^{-36}	3.0×10^{-46}
$e_8^{[\vartheta]}$	7.4×10^{-3}	4.4×10^{-7}	2.9×10^{-15}	2.9×10^{-40}	2.4×10^{-46}
$e_9^{[\vartheta]}$	3.0×10^{-4}	2.8×10^{-6}	2.0×10^{-16}	2.7×10^{-35}	1.1×10^{-49}
$e_{10}^{[\vartheta]}$	3.5×10^{-2}	2.1×10^{-7}	4.5×10^{-16}	2.0×10^{-39}	2.7×10^{-41}
$\rho_i^{[\vartheta-1]}$	2.1012	2.2452	2.0491	3.5112	3.18742

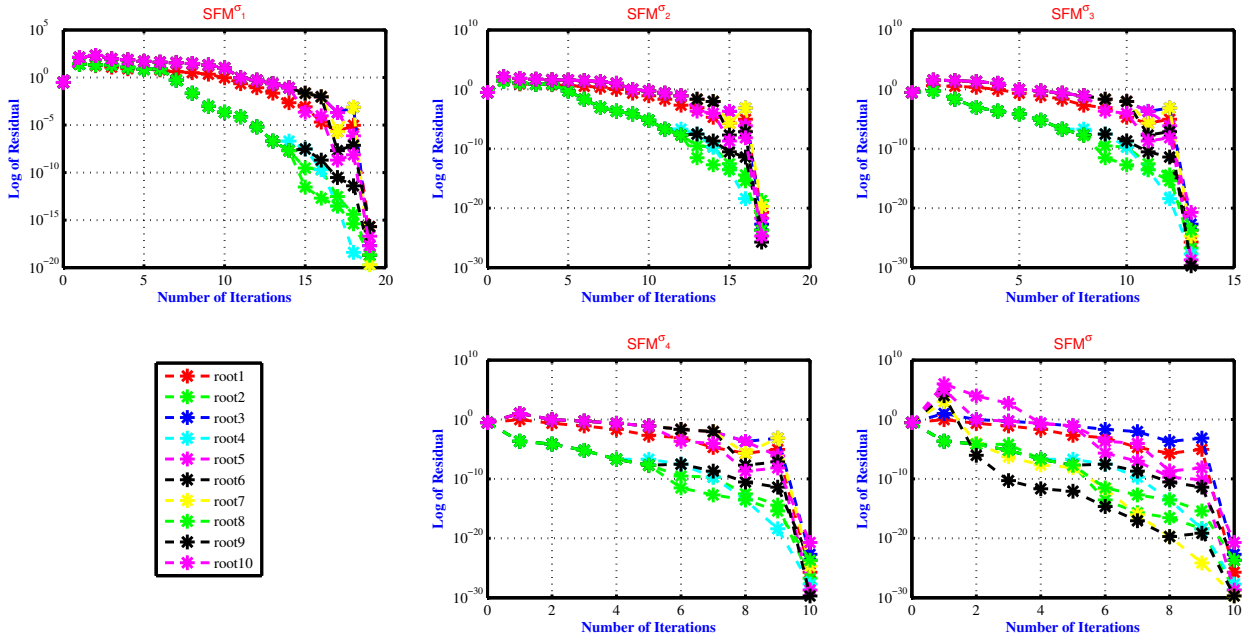

Figure 3. The residual error of SFM$^{\sigma}$ for approximating all polynomial equation roots used in engineering application 2 for various fractional parameter values, namely $\varsigma = 0.1, 0.3, 0.7, 0.8, 1.0$.

Engineering Application 3: Series Circuit Analogue

Consider a flexible spring that is stretched vertically from a rigid support and has a mass m attached to its free end. Naturally, the mass will determine how much the spring elongates or stretches; different weight masses will result in different ways that the spring will stretch. Hooke's law states that the spring itself generates a restoring force F that is opposed to the direction of elongation and proportional to the amount of elongation s. In short, a proportionality constant is defined as $F = ks$, where k is the spring constant. In an undamped spring/mass system, the differential equation represents $F(x)$, which is mathematically modeled as [40,42]:

$$\begin{cases} \frac{d^2y}{dr^2} + y^3 = 0, \\ y(0) = 1, y'(0) = 1. \end{cases} \tag{104}$$

Using the method described in [40], the following polynomial is used to simulate (104) as:

$$f(r) = -0.5r^3 - 0.5r^2 + r + 1. \tag{105}$$

The Caputo-type derivative of (105) is given as:

$$[{}_C D^{\varsigma}_{\varsigma_1}] f(r) = -0.5\frac{\Gamma(4)}{\Gamma(4-\varsigma)} r^{3-\varsigma} - 0.5\frac{\Gamma(3)}{\Gamma(3-\varsigma)} r^{2-\varsigma} + \frac{\Gamma(2)}{\Gamma(2-\varsigma)} r^{1-\varsigma} + \frac{1}{\Gamma(1-\varsigma)} r^{-\varsigma}, \tag{106}$$

The exact solution of (105) up to the decimal places is written as follows:

$$\zeta_1 = -1, \zeta_2 = 1.414213562, \zeta_3 = -1.414213562.$$

To determine the global convergence component of the parallel scheme, use Matlab to generate a random initial guess value ranging from $r^{[0]}_{1*} - r^{[0]}_{5*}$ as specified in Appendix A Table A3. With an arbitrary starting value, SFM$^{\sigma_1}$−SFM$^{\sigma_4}$, SFM$^{\sigma}$ converges to exact zeros after 19, 16, 13, 8, and 8 iterations as indicated in Table 12 for various fractional parameters, i.e., 0.1, 0.3, 0.5, 0.8, and 1.0. As described in Table 13 , the corresponding CPU times are 3.1364, 1.0701, 1.0078, 0.0874 and 0.0975, respectively. Table 14 clearly illustrates how the rate of convergence of SFM$^{\sigma_1}$−SFM$^{\sigma_4}$, SFM$^{\sigma}$ accelerates as the value of the fractional parameter increases from 0.1 to 1.0. The newly developed method converges to exact roots for randomly generated initial guess values, demonstrating its global convergence.

Table 12. Using random initial approximation for finding all polynomial roots simultaneously.

$f(r) = -0.5r^3 - 0.5r^2 + r + 1$					
Ini-V	**Number of Iterations**				
$r^{[0]}$	SFM$^{\sigma_1}$	SFM$^{\sigma_2}$	SFM$^{\sigma_3}$	SFM$^{\sigma_4}$	SFM$^{\sigma}$
$r^{[0]}_{1*}$	19.0	16.0	13.0	8.0	8.0
$r^{[0]}_{2*}$	19.0	16.0	13.0	8.0	8.0
$r^{[0]}_{3*}$	19.0	16.0	13.0	8.0	8.0
$r^{[0]}_{4*}$	19.0	16.0	13.0	8.0	8.0
$r^{[0]}_{5*}$	19.0	16.0	13.0	8.0	8.0
Residual errors are equal to 10^{-30} using 64 D**					

Table 13. CPU-Time using random initial values for finding all polynomial roots.

$f(r) = -0.5r^3 - 0.5r^2 + r + 1$					
Ini-V	**Computational CPU-Time in Seconds**				
$r^{[0]}$	SFM$^{\sigma_1}$	SFM$^{\sigma_2}$	SFM$^{\sigma_3}$	SFM$^{\sigma_4}$	SFM$^{\sigma}$
$r^{[0]}_{1*}$	0.0159	0.0491	1.4514	0.0319	0.0455
$r^{[0]}_{2*}$	2.0136	1.0464	1.0045	0.0694	0.0107
$r^{[0]}_{3*}$	3.1364	1.0139	1.0047	0.0646	0.0059
$r^{[0]}_{4*}$	1.5209	0.0874	1.0078	0.0456	0.0975
$r^{[0]}_{5*}$	1.7451	1.0701	1.0038	0.0874	0.0048
Maximum CPU-Time is equale to 2.1254 using 64 D**					

Table 12 shows the number of iterations of fractional simultaneous scheme SFM$^{\sigma_1}$−SFM$^{\sigma_4}$, SFM$^{\sigma}$ for different random initial vectors given in Appendix A Table A3. Table 12 clearly shows that the number of iterations decreased as the fractional parameter values increased from 0.1 to 1.0.

Table 14. Maximum error using random initial approximation for finding all polynomial roots.

$f(r) = -0.5r^3 - 0.5r^2 + r + 1$					
R-Initial	**Maximum Error**				
$r^{[0]}$	SFM$^{\sigma_1}$	SFM$^{\sigma_2}$	SFM$^{\sigma_3}$	SFM$^{\sigma_4}$	SFM$^{\sigma}$
$r^{[0]}_{1*}$	3.2×10^{-19}	9.1×10^{-17}	0.3×10^{-21}	9.5×10^{-22}	9.1×10^{-23}
$r^{[0]}_{2*}$	2.4×10^{-16}	1.2×10^{-17}	6.1×10^{-25}	8.1×10^{-29}	3.5×10^{-27}
$r^{[0]}_{3*}$	0.1×10^{-14}	0.5×10^{-19}	0.4×10^{-23}	0.8×10^{-20}	9.5×10^{-28}
$r^{[0]}_{4*}$	6.4×10^{-15}	6.5×10^{-18}	0.1×10^{-20}	3.2×10^{-28}	6.2×10^{-25}
$r^{[0]}_{5*}$	7.7×10^{-15}	7.8×10^{-18}	7.2×10^{-23}	6.6×10^{-36}	6.7×10^{-31}
Residual errors are equal to 10^{-30} using 64 D**					

Table 14 shows the maximum error (Max-Err) computed by fractional simultaneous scheme SFM$^{\sigma_1}$−SFM$^{\sigma_3}$, SFM$^{\sigma}$ for different random initial vectors given in Appendix A Table A3 to approximate all roots of the polynomial equations used in application 3. Table 14 clearly demonstrates that as the fractional parameter values increased from 0.1 to 1.0, the accuracy computed by simultaneous scheme increased significantly (Figure 4).

The approximate local computational order of convergence is shown in Table 15. As the fractional parameter values increase from 0.1 to 1.0, the approximate local computational order of convergence increases.

Table 15. Local computational order of convergence using random initial approximation.

$f(r) = -0.5r^3 - 0.5r^2 + r + 1$					
Ini-V	**Local Computational Order of Convergence**				
$r^{[0]}$	SFM$^{\sigma_1}$	SFM$^{\sigma_2}$	SFM$^{\sigma_3}$	SFM$^{\sigma_4}$	SFM$^{\sigma}$
$r^{[0]}_{1*}$	5.0429	6.4013	7.1234	7.9011	7.9012
$r^{[0]}_{2*}$	7.0128	6.4515	7.4416	8.1301	7.9210
$r^{[0]}_{3*}$	5.5927	6.6114	6.9141	7.0914	8.8151
$r^{[0]}_{4*}$	5.4501	5.9174	7.5124	7.7014	8.0140
$r^{[0]}_{5*}$	5.1121	6.5701	7.1104	7.8108	8.0038
Residual errors are equal to 10^{-30} using 64 D**					

Table 13 displays the computational CPU time in seconds required to approximate all roots of the polynomial equation used in application 3 using the fractional simultaneous scheme.

Convergence rates increase as the following initial estimations are sufficiently adjusted to the exact root of engineering application 3:

$$r^{[0]}_1 = -0.1,\ r^{[0]}_2 = 1.4, r^{[0]}_3 = -1.4,$$

are chosen as the initial guessed values.

Table 16 shows how the convergence order and accuracy of the fractional simultaneous scheme increase when we use initial guessed values close to the exact root. The residual error computed by numerical schemes increased as we increased the fractional parameter value from 0.1 to 1.0.

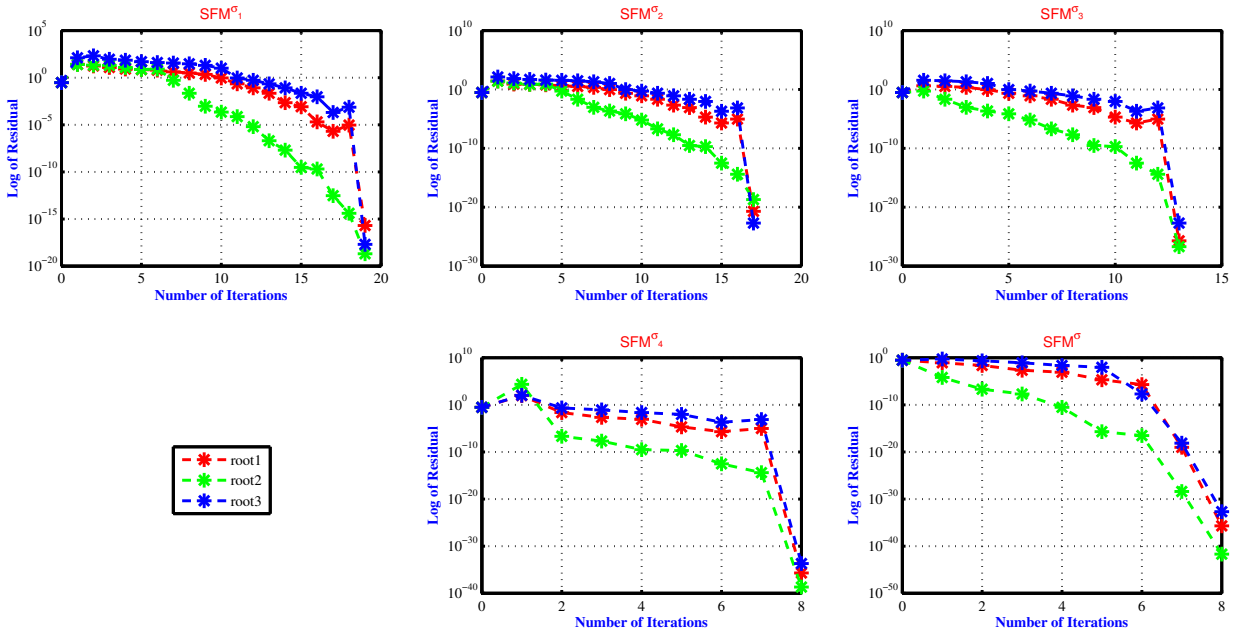

Figure 4. The residual error of the SFM$^{\sigma}$ for approximating all polynomial equation roots used in engineering application 3 for various fractional parameter values, namely $\varsigma = 0.1, 0.3, 0.7, 0.8, 1.0$.

Table 16. Determination of all polynomial equation roots.

Methods	**SFM$^{\sigma_1}$**	**SFM$^{\sigma_2}$**	**SFM$^{\sigma_3}$**	**SFM$^{\sigma_4}$**	**SFM$^{\sigma}$**
Error it	$\vartheta = 09$	$\vartheta = 09$	$\vartheta = 08$	$\vartheta = 08$	$\vartheta = 05$
CPU	0.0124	0.017	0.039	0.077	0.015
$e_1^{[\vartheta]}$	9.9×10^{-4}	0.1×10^{-4}	5.5×10^{-16}	9.1×10^{-35}	0.3×10^{-42}
$e_2^{[\vartheta]}$	5.2×10^{-2}	0.6×10^{-5}	9.1×10^{-16}	5.0×10^{-29}	2.8×10^{-46}
$e_3^{[\vartheta]}$	7.3×10^{-3}	8.4×10^{-2}	6.6×10^{-19}	6.2×10^{-39}	3.5×10^{-46}
$\rho_i^{[\vartheta-1]}$	1.1009	3.2546	2.0451	2.55412	6.78921

Application 4: Hanging Object

A chain attached to an object on the ground is pulled vertically upward by constant forces against gravity, causing the following nonlinear initial value problem:

$$\begin{cases} \frac{d^2y}{dr^2} - \left(\frac{dy}{dr}\right)^2 + 13r + 1 = 0, \\ y(0) = 0, y'(0) = \frac{e^2-1}{e^2+1}. \end{cases} \tag{107}$$

Using the method described in [40], the following polynomial is used to simulate (107) as:

$$f(r) = -0.02590111180r^4 - 0.1066166681r^3 - 0.2099871708r^2 + 0.7615941560r. \tag{108}$$

The Caputo-type derivative of (108) is given as:

$$\begin{aligned} \left[{}_C D_{\varsigma_1}^{\varsigma}\right] f(r) \quad = \quad & -0.02590111180\frac{\Gamma(5)}{\Gamma(5-\varsigma)}r^{4-\varsigma} - 0.1066166681\frac{\Gamma(4)}{\Gamma(4-\varsigma)}r^{3-\varsigma} \\ & -0.2099871708\frac{\Gamma(3)}{\Gamma(3-\varsigma)}r^{2-\varsigma} + 0.7615941560\frac{\Gamma(2)}{\Gamma(2-\varsigma)}r^{1-\varsigma}, \end{aligned} \tag{109}$$

The exact solution of (109) up to 4 decimal places is written as follows:

$$\zeta_1 = 0, \zeta_2 = 1.6609, \zeta_3 = -2.8886 + 3.0592i, \zeta_4 = -2.8886 - 3.0592i.$$

To determine the global convergence component of the parallel scheme, use Matlab to generate a random initial guess value ranging from $r_{1*}^{[0]} - r_{5*}^{[0]}$ as specified in Appendix A Table A4. With an arbitrary starting value, SFM$^{\sigma_1}$−SFM$^{\sigma_4}$, SFM$^{\sigma}$ converges to exact zeros after 19, 16, 14, 8 and 8 iterations as indicated in Table 17 for various fractional parameters, i.e., 0.1, 0.3, 0.5, 0.8, and 1.0. The newly developed method converges to exact roots for randomly generated initial guess values, demonstrating its global convergence.

Table 17 shows the number of iterations of fractional simultaneous scheme SFM$^{\sigma_1}$−SFM$^{\sigma_4}$, SFM$^{\sigma}$ for different random initial vectors given in Appendix A Table A4. Table 18 shows the maximum error (Max-Err) computed by fractional simultaneous scheme SFM$^{\sigma_1}$−SFM$^{\sigma_4}$, SFM$^{\sigma}$ for different random initial vectors given in Appendix A Table A4 to approximate all roots of the polynomial equations used in application 4. Table 18 clearly demonstrates that as the fractional parameter values increased from 0.1 to 1.0, the accuracy computed by the simultaneous scheme increased significantly (Figure 5). This indicates the behavior of our recently developed simultaneous scheme in terms of global convergence. Table 19 clearly illustrates how the computational order of convergence of SFM$^{\sigma_1}$−SFM$^{\sigma_4}$, SFM$^{\sigma}$ increase as the value of the fractional parameter increases from 0.1 to 1.0. As described in Table 20, the corresponding CPU times are 2.1254, 1.0874, 1.0078, 0.0874, and 0.0078, are consumed respectively.

Table 17. Using random initial approximation for finding all polynomial roots simultaneously.

$f(r) = -0.02590111180r^4 - 0.1066166681r^3 - 0.2099871708r^2 + 0.7615941560r$					
Ini-V	**Number of Iterations**				
$r^{[0]}$	SFM$^{\sigma_1}$	SFM$^{\sigma_2}$	SFM$^{\sigma_3}$	SFM$^{\sigma_4}$	SFM$^{\sigma}$
$r_{1*}^{[0]}$	19.0	16.0	14.0	8.0	8.0
$r_{2*}^{[0]}$	19.0	16.0	14.0	8.0	8.0
$r_{3*}^{[0]}$	19.0	16.0	14.0	8.0	8.0
$r_{4*}^{[0]}$	19.0	16.0	14.0	8.0	8.0
$r_{5*}^{[0]}$	19.0	16.0	14.0	8.0	8.0
Residual errors are equal to 10^{-30} using 64 D**					

Table 18. Maximum error using random initial approximation for finding all polynomial roots.

$f(r) = -0.02590111180r^4 - 0.1066166681r^3 - 0.2099871708r^2 + 0.7615941560r$					
R-Initial	**Maximum Error**				
$r^{[0]}$	SFM$^{\sigma_1}$	SFM$^{\sigma_2}$	SFM$^{\sigma_3}$	SFM$^{\sigma_4}$	SFM$^{\sigma}$
$r_{1*}^{[0]}$	3.1×10^{-15}	6.1×10^{-19}	6.9×10^{-21}	9.9×10^{-23}	0.2×10^{-25}
$r_{2*}^{[0]}$	2.6×10^{-13}	0.2×10^{-17}	6.5×10^{-24}	0.8×10^{-29}	5.5×10^{-27}
$r_{3*}^{[0]}$	0.1×10^{-14}	5.1×10^{-18}	0.4×10^{-26}	8.8×10^{-20}	2.5×10^{-26}
$r_{4*}^{[0]}$	3.4×10^{-13}	8.5×10^{-19}	2.1×10^{-20}	5.2×10^{-29}	6.2×10^{-25}
$r_{5*}^{[0]}$	7.7×10^{-15}	7.7×10^{-19}	3.2×10^{-22}	0.2×10^{-37}	9.5×10^{-37}
Residual errors are equal to 10^{-30} using 64 D**					

The approximate local computational order of convergence is shown in Table 19. As the fractional parameter values increase from 0.1 to 1.0, the approximate local computational order of convergence increases.

Table 19. Local computational order of convergence using random initial approximation.

$f(r) = -0.02590111180r^4 - 0.1066166681r^3 - 0.2099871708r^2 + 0.7615941560r$					
Ini-V	**Local Computational Order of Convergence**				
$r^{[0]}$	SFM$^{\sigma_1}$	SFM$^{\sigma_2}$	SFM$^{\sigma_3}$	SFM$^{\sigma_4}$	SFM$^{\sigma}$
$r^{[0]}_{1*}$	5.1813	5.9243	7.7124	7.0515	7.7165
$r^{[0]}_{2*}$	7.0878	6.4985	7.4236	8.1364	7.5243
$r^{[0]}_{3*}$	4.9917	6.9404	7.0148	7.3911	7.8656
$r^{[0]}_{4*}$	5.8748	5.9874	7.5154	7.6212	8.5447
$r^{[0]}_{5*}$	6.5421	6.8748	7.7177	7.0148	8.4873
Residual errors are equal to 10^{-30} using 64 D**					

Table 20 displays the computational CPU time in seconds required to approximate all roots of the polynomial equation used in application 4 using the fractional simultaneous scheme.

Table 20. CPU-Time using random initial values for finding all polynomial roots.

$f(r) = -0.02590111180r^4 - 0.1066166681r^3 - 0.2099871708r^2 + 0.7615941560r$					
Ini-V	**Computational CPU-Time in Seconds**				
$r^{[0]}$	SFM$^{\sigma_1}$	SFM$^{\sigma_2}$	SFM$^{\sigma_3}$	SFM$^{\sigma_4}$	SFM$^{\sigma}$
$r^{[0]}_{1*}$	3.0194	2.0514	1.0454	0.0403	0.0049
$r^{[0]}_{2*}$	2.0120	1.0314	1.0028	0.0644	0.0070
$r^{[0]}_{3*}$	2.1114	1.0141	2.0061	0.0749	0.0031
$r^{[0]}_{4*}$	1.3231	1.0804	1.0078	0.0409	0.0090
$r^{[0]}_{5*}$	1.7300	1.0761	1.0078	0.0741	0.0093
Maximum CPU-Time is equal to 2.1254 using 64 D**					

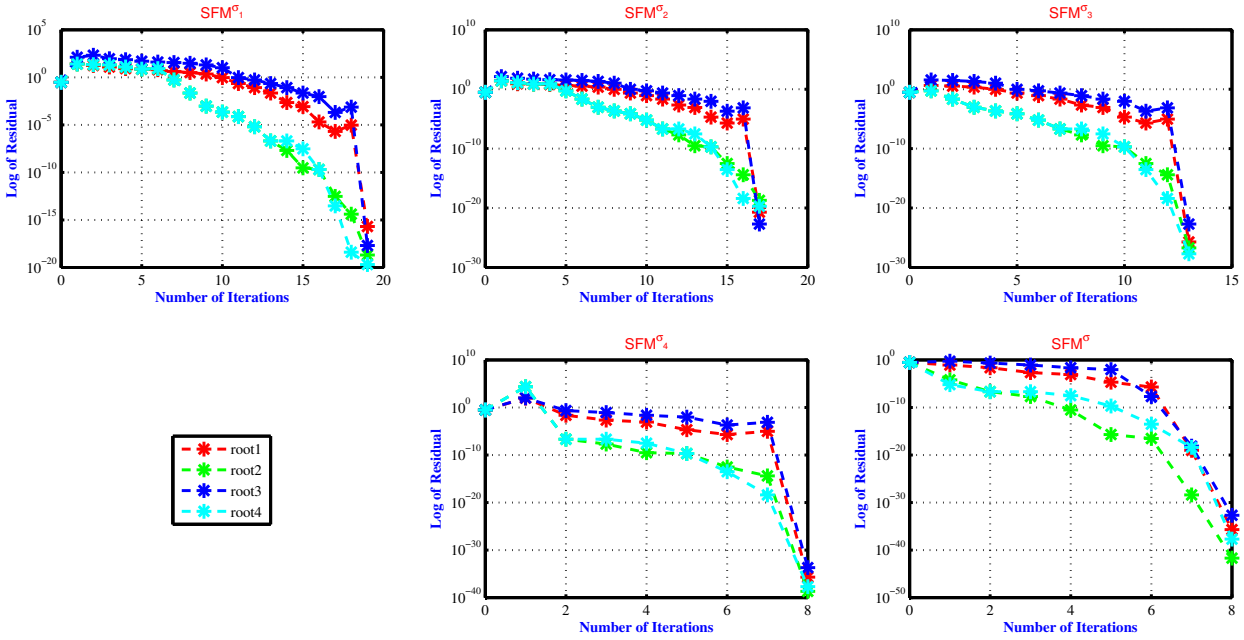

Figure 5. The residual error of the SFM$^{\sigma}$ for approximating all polynomial equation roots used in engineering application 4 for various fractional parameter values, namely $\varsigma = 0.1, 0.3, 0.7, 0.8, 1.0$.

Convergence rates increase as the following initial estimations are sufficiently adjusted to the exact answer of engineering application 4:

$$r_1^{[0]} = 0.2,\ r_2^{[0]} = 1.4, r_3^{[0]} = -2.8 + 3.0i, r_4^{[0]} = -2.8 - 3.0i.$$

are chosen as initial guessed values.

Table 21 shows how the convergence order and accuracy of the fractional simultaneous scheme increase when we use initial guessed values close to the exact root. The residual error computed by the numerical schemes increased as we increased the fractional parameter value from 0.1 to 1.0.

Table 21. Determination of all polynomial equation roots.

Methods	**SFM$^{\sigma_1}$**	**SFM$^{\sigma_2}$**	**SFM$^{\sigma_3}$**	**SFM$^{\sigma_4}$**	**SFM$^{\sigma}$**
Error it	$\vartheta = 09$	$\vartheta = 09$	$\vartheta = 08$	$\vartheta = 08$	$\vartheta = 05$
CPU	0.0141	0.717	0.262	0.092	0.073
$e_1^{[\vartheta]}$	9.1×10^{-4}	0.5×10^{-4}	0.1×10^{-12}	5.7×10^{-39}	9.3×10^{-45}
$e_2^{[\vartheta]}$	1.0×10^{-4}	2.6×10^{-4}	9.1×10^{-16}	5.1×10^{-25}	0.3×10^{-41}
$e_3^{[\vartheta]}$	6.2×10^{-4}	2.6×10^{-4}	9.1×10^{-16}	4.5×10^{-20}	8.8×10^{-48}
$e_4^{[\vartheta]}$	2.1×10^{-2}	0.2×10^{-2}	9.9×10^{-16}	7.2×10^{-37}	0.5×10^{-42}
$\rho_i^{[\vartheta-1]}$	2.1864	2.5002	4.0481	5.5872	7.19212

6. Conclusions

- In order to approximate all roots of nonlinear equations, a new fractional parallel approach with convergence orders of $3\varsigma + 5$ is presented. The global convergence behavior of the fractional parallel schemes is demonstrated using a variety of random starting estimates of SFM$^{\sigma_1} -$SFM$^{\sigma_4}$, SFM$^{\sigma}$.
- The numerical results of the engineering applications from Tables 1–21 and Figures 1–5 clearly show the efficiency of the newly developed methods in terms of CPU-time, computational error, maximum residual error, and local computational order of convergence (LCOC). The acceleration of the convergence rate is observed when the initial approximations close to the exact roots are selected as shown in Tables 6, 11, 16 and 21.
- In the future, higher-order parallel iterative approaches for solving (1) will be developed to handle more difficult engineering problems using fractional derivatives of Riemann–Liouville and Grunwald–Letnikov types.

Author Contributions: Conceptualization, M.S. and B.C.; methodology, M.S.; software, M.S.; validation, M.S.; formal analysis, B.C.; investigation, M.S.; resources, B.C.; writing—original draft preparation, M.S. and B.C.; writing—review and editing, B.C.; visualization, M.S. and B.C.; supervision, B.C.; project administration, B.C.; funding acquisition, B.C. All authors have read and agreed to the published version of the manuscript.

Funding: The work is supported by Provincia autonoma di Bolzano/Alto Adigeâ euro " Ripartizione Innovazione, Ricerca, Universitá e Musei (contract nr. 19/34). Bruno Carpentieri is a member of the Gruppo Nazionale per it Calcolo Scientifico (GNCS) of the Istituto Nazionale di Alta Matematia (INdAM) and this work was partially supported by INdAM-GNCS under Progetti di Ricerca 2022.

Data Availability Statement: Data are contained within the article.

Conflicts of Interest: The authors declare that there are no conflict of interest regarding the publication of this article.

Abbreviations

The following abbreviations are utilized in this study's article:

$SFM^{\sigma_1} - SFM^{\sigma_3}$, SFM^{σ}	Fractional parallel scheme
Error it	Iteration number
Ex-Time	Computer CPU-Time in seconds
$\rho_{\varsigma i}^{(\sigma-1)}$	Computational local order of convergence
Per-E	Percentage effectiveness
Ini-V	Initial vector
D**	Digits floating point arithmetic
CPU-Time	Computational time in seconds

Appendix A

Table A1. Initial random vectors used in fractional simultaneous schemes for approximating all polynomial roots used in engineering application 1.

$\boldsymbol{r^{[0]}}$	$[\boldsymbol{r_1^{[0]}, r_2^{[0]}, r_3^{[0]}, r_4^{[0]}, r_5^{[0]}, r_6^{[0]}, r_7^{[0]}, r_8^{[0]}, r_9^{[0]}, r_{10}^{[0]}, r_{11}^{[0]}, r_{12}^{[0]}, r_{13}^{[0]}, r_{14}^{[0]}, r_{15}^{[0]}, r_{16}^{[0]}, r_{17}^{[0]}, r_{18}^{[0]},}]$
$r_{1*}^{[0]}$	[−0.160, 0.643, 0.967, 0.085, 0.967, 0.881, 0.760, 0.643, 0.874, 0.475, 0.876, −0.153, 0.392, 0.615, 0.171, 0.743, 0.643, 0.967]
$r_{2*}^{[0]}$	[0.743, 0.392, 0.655, 0.171, 0.743, 0.392, 0.855, 0.071, 0.145, 0.874, 0.775, 0.076, 0.643, 0.967, 0.085, 0.967, 0.881, 0.076]
$r_{3*}^{[0]}$	[−0.145, 0.874, 0.475, 0.876, −0.153, 0.392, 0.615, 0.171, 0.743, 0.775, 0.076, 0.3456, 0.74125, 0.643, 0.967, 0.874, 0.473, 0.145]
⋮	[⋮, ⋮, ⋮, ⋮, ⋮, ⋮, ⋮, ⋮, ⋮, ⋮, ⋮, ⋮, ⋮, ⋮, ⋮, ⋮, ⋮, ⋮]

Table A2. Initial random vectors used in fractional simultaneous schemes for approximating all polynomial roots used in engineering application 2.

$\boldsymbol{r^{[0]}}$	$[\boldsymbol{r_1^{[0]}, r_2^{[0]}, r_3^{[0]}, r_4^{[0]}, r_5^{[0]}, r_6^{[0]}, r_7^{[0]}, r_8^{[0]}, r_9^{[0]}, r_{10}^{[0]}}]$
$r_{1*}^{[0]}$	[−0.760, 0.643,0.967, 0.881, 0.760, 0.643, 0.967, 0.085, 0.01451, 0.1452]
$r_{2*}^{[0]}$	[−0.153, 0.392, 0.615, 0.171, 0.743, 0.392, 0.855, 0.071, 0.4512, 0.5641]
$r_{3*}^{[0]}$	[−0.905, 0.874, 0.473, 0.076, 0.145, 0.874, 0.775, 0.076, 0.3456, 0.74125]
⋮	[⋮, ⋮, ⋮, ⋮, ⋮, ⋮, ⋮, ⋮, ⋮, ⋮]

Table A3. Initial random vectors used in fractional simultaneous schemes for approximating all polynomial roots used in engineering application 3.

$\boldsymbol{r^{[0]}}$	$[\boldsymbol{r_1^{[0]}, r_2^{[0]}, r_3^{[0]}}]$
$r_{1*}^{[0]}$	[−0.760, 0.643,0.967]
$r_{2*}^{[0]}$	[0.743, 0.392, 0.855]
$r_{3*}^{[0]}$	[0.076, 0.145, 0.874]
⋮	[⋮, ⋮ , ⋮]

Table A4. Initial random vectors used in fractional simultaneous schemes for approximating all polynomial roots used in engineering application 4.

$\boldsymbol{r^{[0]}}$	$[\boldsymbol{r_1^{[0]}, r_2^{[0]}, r_3^{[0]}, r_4^{[0]}}]$
$r_{1*}^{[0]}$	[−0.160, 0.643, 0.967, 0.085]
$r_{2*}^{[0]}$	[0.743, 0.392, 0.655, 0.171]
$r_{3*}^{[0]}$	[−0.145, 0.874, 0.475, 0.876]
⋮	[⋮, ⋮, ⋮, ⋮]

References

1. Traub, J.F. *Iterative Methods for the Solution of Equations*; Prentice-Hall: Englewood Cliffs, NJ, USA, 1964.
2. Jarratt, P. Some efficient fourth order multiple methods for solving equations. *BIT* **1969**, *9*, 119–124. [CrossRef]
3. King, R. A family of fourth order methods for nonlinear equations. *SIAM J. Numer. Anal.* **1973**, *10*, 876–879. [CrossRef]
4. Ostrowski, A.M. *Solution of Equation in Euclidean and Banach Space*, 3rd ed.; Academic Press: New York, NY, USA, 1973.
5. Chun, C. Some fourth-order iterative methods for solving nonlinear equations. *Appl. Math. Lett.* **2008**, *195*, 454–456. [CrossRef]
6. Weierstrass, K. Neuer Beweis des Satzes, dass jede ganze rationale Function einer Verän derlichen dargestellt werden kann als ein Product aus linearen Functionen derselben Verän derlichen. Sitzungsberichte KöNiglich Preuss. *Akad. Der Wiss. Berl.* **1981**, *2*, 1085–1101.
7. Kanno, S.; Kjurkchiev, N.V.; Yamamoto, T. On some methods for the simultaneous determination of polynomial zeros. *Japan J. Appl. Math.* **1995**, *13*, 267–288. [CrossRef]
8. Proinov, P.D.; Cholakov, S.I. Semilocal convergence of Chebyshev-like root-finding method for simultaneous approximation of polynomial zeros. *Appl. Math. Comput.* **2014**, *236*, 669–682. [CrossRef]
9. Mir, N.A.; Muneer, R.; Jabeen, I. Some families of two-step simultaneous methods for determining zeros of nonlinear equations. *ISRN Appl. Math.* **2011**, *2011*, 817174. [CrossRef]
10. Farmer, M.R. *Computing the Zeros of Polynomials Using the Divide and Conquer Approach*; Department of Computer Science and Information Systems; Birkbeck: London, UK, 2014.
11. Nourein, A.W. An improvement on Nourein's method for the simultaneous determination of the zeroes of a polynomial (an algorithm). *J. Comput. Appl. Math.* **1977**, *3*, 109–112 [CrossRef]
12. Aberth, O. Iteration methods for finding all zeros of a polynomial simultaneously. *Math. Comput*. **1973**, *27*, 339–344. [CrossRef]
13. Cholakov, S.I.; Vasileva, M.T. A convergence analysis of a fourth-order method for computing all zeros of a polynomial simultaneously. *J. Comput. Appl. Math.* **2017**, *321*, 270–283. [CrossRef]
14. Consnard, M.; Fraigniaud, P. Finding the roots of a polynomial on an MIMD multicomputer. *Parall. Comput.* **1990**, *15*, 75–85. [CrossRef]
15. Petković, M.S.; Petković, L.D.; Džunić, J. On an efficient method for the simultaneous approximation of polynomial multiple roots. *Appl. Anal. Disc. Math.* **2014**, *8*, 73–94. [CrossRef]
16. Rafiq, N.; Shams, M.; Mir, N.A.; Gaba, Y.U. A highly efficient computer method for solving polynomial equations appearing in Engineering Problems. *Math. Probl. Eng.* **2023**, *2021*, 9826693. [CrossRef]
17. Shams, M.; Rafiq, N.; Kausar, N.; Agarwal, P.; Park, C.; Mir, N.A. On iterative techniques for estimating all roots of nonlinear equation and its system with application in differential equation. *Adv. Differ. Equ*. **2021**, *2021*, 480. [CrossRef]
18. Kyncheva, V.K.; Yotov, V.V.; Ivanov, S.I. Convergence of Newton, Halley and Chebyshev iterative methods as methods for simultaneous determination of multiple polynomial zeros. *Appl. Numer. Math.* **2017**, *112*, 146–154. [CrossRef]
19. Nedzhibov, H. Iterative methods for simultaneous computing arbitrary number of multiple zeros of nonlinear equations. *Int. J. Comp. Math.* **2013**, *90*, 994–1007. [CrossRef]
20. Sendov, B.L.; Andreev, A.; Kjurkchiev, N. Numerical solution of polynomial equations. *Handb. Numer. Anal.* **1994**, *3*, 625–778.
21. Kyurkchiev, N.; Iliev, A. A general approach to methods with a sparse Jacobian for solving nonlinear systems of equations. *Serdica Math. J.* **2007**, *33*, 433–448.
22. Shams, M.; Kausar, N.; Agarwal, P.; Oros, G.I. On Efficient Fractional Caputo-type Simultaneous Scheme for Finding all Roots of polynomial equations. *Fractals* **2023**, *6*, 2340075. [CrossRef]
23. Dimitrov, Y.; Georgiev, S.; Todorov, V. Approximation of Caputo Fractional Derivative and Numerical Solutions of Fractional Differential Equations. *Fractal Fract.* **2023**, *7*, 750. [CrossRef]
24. Shams, M.; Kausar, N.; Agarwal, P.; Shah, M.A. On family of Caputo-Type fractional numerical scheme for solving polynomial. *Appl. Math. Sci. Eng.* **2023**, *31*, 2181959. [CrossRef]
25. Oliveira, D.E.C.; Tenreiro Machado, J.A. A review of definitions for fractional derivatives and integral. *Math. Probl. Eng.* **2014**, *2014*, 238459. [CrossRef]
26. Oldham, K.; Spanier, J. *The Fractional Calculus Theory and Applications of Differentiation and Integration to Arbitrary Order*; Elsevier: Amsterdam, The Netherlands, 1974.
27. Kukushkin, M.V. Abstract fractional calculus for m-accretive operators. *arXiv* **2019**, arXiv:1901.06118.
28. Samko, S.G.; Kilbas, A.A.; Marichev, O.I. *Fractional Ntegrals and Derivatives: Theory and Applications*; Gordon and Breach Science Publishers: Philadelphia, PA, USA, 1993.
29. Shams, M.; Carpentieri, B. Efficient Inverse Fractional Neural Network-Based Simultaneous Schemes for Nonlinear Engineering Applications. *Fractal. Fract.* **2023**, *7*, 849. [CrossRef]
30. Odibat, Z.M.; Shawagfeh, N.T. Generalized Taylor's formula. *Appl. Math. Comput.* **2007**, *186*, 286–293. [CrossRef]
31. Akgül, A.; Cordero, A.; Torregrosa, J.R. A fractional Newton method with 2th-order of convergence and its stability. *Appl. Math. Lett.* **2019**, *98*, 344–351. [CrossRef]
32. Torres-Hernandez, A.; Brambila-Paz, F. Sets of fractional operators and numerical estimation of the order of convergence of a family of fractional fixed-point methods. *Fractal Fract.* **2021**, *4*, 240. [CrossRef]
33. Cajori, F. Historical note on the Newton-Raphson method of approximation. *Am. Math. Mon.* **1911**, *18*, 29–32. [CrossRef]

34. Kumar, P.; Agrawal, O.P. An approximate method for numerical solution of fractional differential equations. *Signal Process.* **2006**, *86*, 2602–2610. [CrossRef]
35. Candelario, G.; Cordero, A.; Torregrosa, J.R. Multipoint fractional iterative methods with (2 + 1) th-order of convergence for solving nonlinear problems. *Mathematics* **2020**, *8*, 452. [CrossRef]
36. Proinov, P.D.; Vasileva, M.T. On the convergence of high-order Ehrlich-type iterative methods for approximating all zeros of a polynomial simultaneously. *J. Ineq. Appl.* **2015**, *2015*, 336. [CrossRef]
37. Chu, Y.; Rafiq, N.; Shams, M.; Akram, S.; Mir, N.A.; Kalsoom, H. Computer methodologies for the comparison of some efficient derivative free simultaneous iterative methods for finding roots of non-linear equations. *Comput. Mater. Cont.* **2020**, *66*, 275–290. [CrossRef]
38. Naseem, A.; Rehman, M.A.; Abdeljawad, T. Computational methods for non-linear equations with some real-world applications and their graphical analysis. *Intell. Autom. Soft Comput.* **2021**, *30*, 1–14. [CrossRef]
39. Akin-Bohnera, E.; Hoackerb, J. Oscillation properties of an Emden-Fowler type equation on discrete time scales. *J. Diff. Equ. Appl.* **2003**, *9*, 603612. [CrossRef]
40. Shams, M.; Kausar, N.; Yaqoob, N.; Arif, N.; Addis, G.M. Techniques for finding analytical solution of generalized fuzzy differential equations with applications. *Complexity* **2023**, *2023*, 3000653. [CrossRef]
41. Zill, D.G. *Differential Equations with Boundary-Value Problems*; Cengage Learning: Boston, MA, USA, 2016.
42. Chapra, S. *EBOOK: Applied Numerical Methods with MATLAB for Engineers and Scientists*; McGraw Hill: New York, NY, USA, 2011.

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.