applied sciences

MDPI

Article

Q-Analogues of Parallel Numerical Scheme Based on Neural Networks and Their Engineering Applications

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

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

Abstract: Quantum calculus can provide new insights into the nonlinear behaviour of functions and equations, addressing problems that may be difficult to tackle by classical calculus due to high nonlinearity. Iterative methods for solving nonlinear equations can benefit greatly from the mathematical theory and tools provided by quantum calculus, e.g., using the concept of q-derivatives, which extends beyond classical derivatives. In this paper, we develop parallel numerical root-finding algorithms that approximate all distinct roots of nonlinear equations by utilizing q-analogies of the function derivative. Furthermore, we utilize neural networks to accelerate the convergence rate by providing accurate initial guesses for our parallel schemes. The global convergence of the q-parallel numerical techniques is demonstrated using random initial approximations on selected biomedical applications, and the efficiency, stability, and consistency of the proposed hybrid numerical schemes are analyzed.

Keywords: neural network; q-iterative schemes; q-Taylor's series; CPU-Time; convergence rate; biomedical engineering applications

check for updates

Citation: Shams, M.; Carpentieri, B. Q-Analogues of Parallel Numerical Scheme Based on Neural Networks and Their Engineering Applications. *Appl. Sci.* **2024**, *14*, 1540. https://doi.org/10.3390/app14041540

Academic Editor: Marek Krawczuk

Received: 29 December 2023
Revised: 3 February 2024
Accepted: 10 February 2024
Published: 14 February 2024

1. Introduction

Nonlinear equations are widely used in computational science and engineering modeling because of their ability to accurately represent the complexities of real-world phenomena, resulting in more precise predictions, optimizations, and insights into system behaviors in a wide range of scientific and engineering disciplines [1–5], including fluid dynamics [6], quantum mechanics [7], electromagnetism, and computational biology processes [8], to name only a few. They are especially important in chaos theory and complexity, where they can model systems with great sensitivity to initial conditions, e.g., in meteorology, population dynamics, and in the financial sector [9,10]. According to Abel's impossibility theorem [11], there is no algebraic solution to polynomials of degree greater than four expressed in terms of a finite number of additions, subtractions, multiplications, divisions, and root extractions; thus, we need to turn to numerical iterative methods such as Newton's method and fixed-point iteration [12–14] to approximate the roots of a general Equation (1) one at a time. Single root finding methods are highly sensitive to the initial guess values used, though, and their local convergence behavior diverges as $f'(x)$ approaches zero. As a result, in this study we investigate parallel numerical algorithms that exhibit global convergence behavior, are more efficient, and are more stable than single root-finding algorithms [15]. Parallel numerical schemes use simple arithmetic operations to simultaneously and independently update the estimates for each root in each iteration, making them well suited for efficient parallel implementation. These techniques are also well-known for their robust convergence properties [16]. They commonly converge to the roots of nonlinear equations even starting with random initial guesses, which may be very advantageous in a parallel setting [17]. Because of these reasons, a significant amount of work is being devoted to the development of parallel simultaneous root-finding schemes for solving nonlinear

equations. These schemes employ a variety of techniques, each with its own convergence order, e.g., Weierstrass [18], Kerner [19], Dochev [20], Nedzhibov [21], Marcheva et al. [22], Shams et al. [23], Alefeld et al. [24], Nourein [25] and references cited therein.

The design of iterative methods for solving nonlinear equations can benefit greatly from recent developments in quantum calculus theory, a prominent area of mathematics that is constantly evolving [26–29], to tackle problems with high nonlinearity that may be difficult to address by conventional calculus. The concept of q-derivatives, which is introduced in quantum calculus and extends beyond classical derivatives, can provide new insights into the nonlinear behavior of functions and equations.

One way neural networks can help with root finding is by regression or approximation.

The main goal of this research is to develop parallel numerical schemes that approximate all distinct roots of nonlinear Equation (1) by utilizing q-analogies of the function derivative. Neural networks are utilized to accelerate the convergence rate of the new class of parallel numerical schemes introduced in this work. A neural network can be trained to approximate the polynomial's function before using numerical approaches to determine its roots. However, it is crucial to note that, while neural networks excel in approximating complex functions, their ability to precisely locate roots varies according to the nature of the problem and the network's architecture. The hybrid neural network-based q-version of parallel numerical schemes used the neural network's outputs as an initial guess and refined it up to the desired decimal, outperforming classical parallel schemes and speeding up convergence. The numerical scheme that is derived from parallel neural networks (PNNS) and implemented in the q-calculus framework demonstrates global convergence, as illustrated by our numerical experiments.

In this section, we review some fundamental results of q-calculus theory that were utilized in the design of our q-version of parallel numerical schemes for locating all distinct roots of the nonlinear equation

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

Given $q \in (0,1)$, the q-integer is defined as

$$[m;q] = 1 + q + q^2 + q^m + \ldots + q^{m-1} = \frac{1-q^m}{1-q}; for\ m = 1,2,3,\ldots. \tag{2}$$

whereas

$$[m;q] = m;\ \text{for } q = 1. \tag{3}$$

The q-binomial is defined for $0 < j < m$, as

$$\left[\begin{array}{c} m \\ j \end{array};q\right] = \frac{[m;q]!}{[j;q]![m-j;q]!}. \tag{4}$$

Finally, the q-factorial $[m;q]$ is defined as

$$[m;q]! = [m;q][m-1;q]\ldots[3;q][2;q][1;q] \tag{5}$$

and $[0;q] = 1$.

Definition 1 ([30,31]). *The q-derivative of $f(x)$ is defined as*

$$\partial_q(x) = \partial_q^{[1]}(x) = (\partial_q f)(x) = \left[\frac{d}{dx}\right]_q f(x) = \frac{f(qx) - f(x)}{qx - x}, q \neq 1. \tag{6}$$

For $q \to 1$, we have

$$\partial(x) = f'(x) = \left[\frac{d}{dx}\right] f(x), \tag{7}$$

which is the classical derivative. The q-derivative $\left(\partial_q^{[1]} f\right)(x)$ *is known as Jakson derivative [32]. We define higher q-derivatives as*

$$\partial_q^0 f = f, \partial_q^n f = \partial_q\left(\partial_q^{n-1}\right) \text{ for } n = 1, 2, 3, \ldots \tag{8}$$

Definition 2. *Product and quotient of functions* $f(x)$ *and* $g(x)$ *are defined, respectively, as [33]*

$$\begin{aligned} \partial_q^{[1]}(f(x) * g(x)) &= g(x)\partial_q^{[1]} f(x) + f(qx)\partial_q^{[1]} g(x), \\ &= g(qx)\partial_q^{[1]} f(x) + f(x)\partial_q^{[1]} g(x), \end{aligned} \tag{9}$$

and

$$\partial_q^{[1]}\left(\frac{f(x)}{g(x)}\right) = \frac{g(x)\partial_q^{[1]} f(x) - f(x)\partial_q^{[1]} g(x)}{g(qx)g(x)}, g(qx)g(x) \neq 0. \tag{10}$$

Definition 3. *q-Taylor's formula [34] for* $f(x)$ *is given as*

$$\begin{aligned} f(x) &= f(C) + \frac{(x-C)^{[1]}}{[1:q]}\left(\partial_q^{[1]} f\right)(x) + \frac{(x-C)^{[2]}}{[2:q]!}\left(\partial_q^{[2]} f\right)(x) \\ &\quad + \ldots + \frac{(x-C)^{[n]}}{[n:q]!}\left(\partial_q^{[n]} f\right)(x) + R_n, \end{aligned} \tag{11}$$

$$f(x) = \sum_{j=1}^{\infty} \frac{\partial_q^{[j]}(x-C)^{[j]}}{j!} + R_n; (\forall x \in (a,b)), \tag{12}$$

where $(x-C)^0 = 1, (x-C)^i = \prod_{i=0}^{n-1}(x - Cq)^i, i \in N.0 < q < 1$, $R_n = \frac{(x-C)^{[n+1]}}{[n+1:q]!}\left(\partial_q^{[n+1]} f\right)(\zeta)$ *and* $\partial_q^{[1]}, \partial_q^{[2]}, \ldots$ *are all q-derivatives.*

The remainder of the article is structured as follows. In Section 2, we present and analyze a parallel numerical scheme for solving (1) formulated in the framework of quantum calculus. Section 3 discusses the neural network implementation of the proposed q-version of the parallel solver. We address the numerical solution of two nonlinear engineering problems in Section 4. Finally, the paper concludes in Section 5 with some remarks arising from this study.

2. Construction of Parallel Numerical Scheme Using Q-Calculus

In this section, we propose q-analogies of a novel class of single root finding methods of (1). Then, we generalize it into a parallel numerical scheme to find all distinct roots of (1). The starting point of our development is the following numerical scheme:

$$\begin{cases} v^{[\sigma]} = x^{[\sigma]} - \frac{f(x^{[\sigma]})}{\left(\partial_q^{[1]} f\right)(x)}\left[\frac{1}{1 - \frac{\alpha_1 f(x^{[\sigma]})}{1+\alpha_2 f(x^{[\sigma]})}}\right], \\ z^{[\sigma]} = v^{[\sigma]} - \frac{f(v^{[\sigma]})}{\left(\partial_q^{[1]} f\right)(x^{[\sigma]})}, \ \sigma = 0, 1, \ldots \end{cases} \tag{13}$$

where $\left(\partial_q^{[1]} f\right)(x^{[\sigma]}) = \frac{f(qx^{[\sigma]}) - f(x^{[\sigma]})}{qx^{[\sigma]} - x^{[\sigma]}}$ and $q, \alpha_1, \alpha_2 \in \mathbb{R}$.

2.1. Convergence Analysis

The order of convergence of the numerical scheme (13) is established by the following theorem.

Theorem 1. *Let $\zeta \in I$ be a simple root of a sufficiently differential function $f : I \subseteq \mathbb{R} \longrightarrow \mathbb{R}$ in an open interval $I \subset \mathbb{R}$. If $x^{[0]}$ is sufficiently close to ζ, then (13) has $(3:q)$-order convergence (in terms of quantum calculus) with following error equation:*

$$\vartheta^{[\sigma+1]} = \begin{pmatrix} -\alpha_1 \left(\frac{\left(\partial_q^{[2]} f\right)(\zeta)}{\left(\partial_q^{[1]} f\right)(\zeta)} \right) + \\ \left(\frac{\left(\partial_q^{[2]} f\right)(\zeta)}{2\left(\partial_q^{[1]} f\right)(\zeta)} \right)^2 \end{pmatrix} \left(\left[\vartheta^{[\sigma]}\right]^3 ; q \right) + O\left(\left[\vartheta^{[\sigma]}\right]^4 ; q \right). \tag{14}$$

The proof of Theorem 1 can be found in Appendix A.

2.2. Q-Analogies of Parallel Numerical Scheme of Convergence Order $[\epsilon^3; q]$

Next, we introduce and analyze a new family of single-step and two-step q-analogies of parallel numerical schemes for computing all distinct roots of (1) based on numerical scheme (13). We begin with the well-known Weierstrass method [35], which approximates all roots of (1) as

$$x_i^{[\sigma+1]} = x_i^{[\sigma]} - \vartheta_i\left(x_i^{[\sigma]}\right), \tag{15}$$

where $\vartheta_i\left(x_i^{[\sigma]}\right) = \frac{f\left(x_i^{[\sigma]}\right)}{\prod_{\substack{i \neq j \\ j=1}}^{n} \left(x_i^{[\sigma]} - x_j^{[\sigma]}\right)}$ is the so-called Weierstrass correction. The order of convergence of the numerical scheme (15) is 2. By applying the q-analogies of the single root finding method (13) as a correction, specifically substituting $v_j^{[\sigma]}$ for $x_j^{[\sigma]}$ in Equation (15), we consider the following new family of q-analogies of parallel numerical algorithms, denoted as Q-MM$^{\sigma_1}$, which are designed to approximate all distinct roots of Equation (1):

$$x_i^{[\sigma+1]} = x_i^{[\sigma]} - \vartheta_i^*\left(x_i^{[\sigma]}\right), \tag{16}$$

where $\vartheta_i^*\left(x_i^{[\sigma]}\right) = \frac{f\left(x_i^{[\sigma]}\right)}{\prod_{\substack{i \neq j \\ j=1}}^{n} \left(x_i^{[\sigma]} - x_j^{[\sigma]} - \frac{f\left(x_j^{\sigma]}\right)}{\left(\partial_q^{[1]} f\right)\left(x_j^{[\sigma]}\right)} \left[\frac{1}{1 - \frac{\alpha_1 f\left(x_j^{[\sigma]}\right)}{1 + \alpha_2 f\left(x_j^{[\sigma]}\right)}} \right] \right)}$. Hence, method Q-MM$^{\sigma_1}$ can be written as:

$$x_i^{[\sigma+1]} = x_i^{[\sigma]} - \frac{f\left(x_i^{[\sigma]}\right)}{\prod_{\substack{i \neq j \\ j=1}}^{n} \left(x_i^{[\sigma]} - x_j^{[\sigma]} - \frac{f\left(x_j^{\sigma]}\right)}{\left(\partial_q^{[1]} f\right)\left(x_j^{[\sigma]}\right)} \left[\frac{1}{1 - \frac{\alpha_1 f\left(x_j^{[\sigma]}\right)}{1 + \alpha_2 f\left(x_j^{[\sigma]}\right)}} \right] \right)}, \tag{17}$$

where $\alpha_1, \alpha_2 \in \mathbb{R}$.

2.3. Convergence Analysis

In the following theorem we establish the convergence order of Q-MM$^{\sigma_1}$.

Theorem 2. *Let $\zeta_1, \ldots, \zeta_\sigma$ be simple zeros of the nonlinear Equation (1). For initial distinct values $x_1^{[0]}, \ldots, x_n^{[0]}$ that are sufficiently close to the exact roots, the Q-MM$^{\sigma_1}$ has convergence order $[\epsilon^3; q]$.*

Proof. Let $\epsilon_i = x_i^{[\sigma]} - \zeta_i, \epsilon_i' = x_i^{[\sigma+1]} - \zeta_i$ be the errors in $x_i^{[\sigma]}$, and $x_i^{[\sigma+1]}$ respectively. From the first-step of Q-MM$^{\sigma_1}$, we have:

$$y_i^{[\sigma]} - \zeta_i = x_i^{[\sigma]} - \zeta_i - \frac{f\left(x_i^{[\sigma]}\right)}{\prod\limits_{\substack{j=1\\ j\neq i}}^{n}\left(x_i^{[\sigma]} - x_j^{[\sigma]} - \frac{f\left(x_j^{[\sigma]}\right)}{\left(\eth_q^{[1]} f\right)\left(x_j^{[\sigma]}\right)}\left[1 - \frac{\alpha_1 f\left(x_j^{[\sigma]}\right)}{1+\alpha_2 f\left(x_j^{[\sigma]}\right)}\right]^{-1}\right)}, \tag{18}$$

$$\epsilon_i' = \epsilon_i - \vartheta_i^*\left(x_i^{[\sigma]}\right) = \epsilon_i - \epsilon_i \frac{\vartheta_i^*\left(x_i^{[\sigma]}\right)}{\epsilon_i} \tag{19}$$

$$\epsilon_i' = \epsilon_i\left(1 - Q_i^{[*]}\right) \tag{20}$$

where

$$Q_i^{[*]} = \frac{\vartheta_i^*\left(x_i^{[\sigma]}\right)}{\epsilon_i} = \prod_{\substack{j=1\\ j\neq i}}^{n}\left[\frac{x_i^{[\sigma]} - \zeta_j}{x_i^{[\sigma]} - y_j^{[\sigma]}}\right] \tag{21}$$

and $v_j^{[\sigma]} = x_j^{[\sigma]} - \frac{f\left(x_j^{[\sigma]}\right)}{\left(\eth_q^{[1]} f\right)\left(x_j^{[\sigma]}\right)}\left[1 - \frac{\alpha_1 f\left(x_j^{[\sigma]}\right)}{1+\alpha_2 f\left(x_j^{[\sigma]}\right)}\right]^{-1}$. Therefore, we can write

$$\frac{x_i^{[\sigma]} - \zeta_j}{x_i^{[\sigma]} - y_j^{[\sigma]}} = 1 + \frac{v_j^{[\sigma]} - \zeta_j}{x_i^{[\sigma]} - v_j^{[\sigma]}} 1 + O\left(\left|\epsilon_j^2; q\right|\right), \tag{22}$$

and, consequently,

$$\begin{aligned} Q_i^{[*]} &= \prod_{\substack{j=1\\ j\neq i}}^{n}\left[\frac{v_j^{[\sigma]} - \zeta_j}{x_i^{[\sigma]} - v_j^{[\sigma]}}\right] = \left(1 + O\left(\left|\epsilon_j^2\right|\right)\right)^{n-1}, \\ &= 1 + (n-1)O\left(\left|\epsilon_j^2\right|\right) = 1 + O\left(\left|\epsilon_j^2\right|\right), \end{aligned} \tag{23}$$

$$Q_i^{[*]} - 1 = O\left(\left|\epsilon_j^2; q\right|\right). \tag{24}$$

We conclude that

$$\epsilon_i' = \epsilon_i\left(O\left(\left|\epsilon_j^2; q\right|\right)\right).$$

If $|\epsilon_i|$ and $|\epsilon_j|$ have same order, then we have

$$\epsilon_i' = O\left(\left|\epsilon^3; q\right|\right). \tag{25}$$

This proves the theorem. □

2.4. Q-Analogies of Parallel Numerical Scheme of Convergence Order $[\epsilon^8; q]$

At this stage, we consider the well known two-step Weierstrass method [36] for approximating all roots of Equation (1):

$$x_i^{[\sigma+1]} = y_i^{[\sigma]} - \vartheta_i\left(y_i^{[\sigma]}\right) = y_i^{[\sigma]} - \frac{f\left(y_i^{[\sigma]}\right)}{\prod\limits_{\substack{i\neq j\\ j=1}}^{n}\left(y_i^{[\sigma]} - y_j^{[\sigma]}\right)}, \tag{26}$$

where $y_i^{[\sigma]} = x_i^{[\sigma]} - \vartheta_i\left(x_i^{[\sigma]}\right) = x_i^{[\sigma]} - \frac{f\left(x_i^{[\sigma]}\right)}{\prod\limits_{\substack{i\neq j\\ j=1}}^{n}\left(x_i^{[\sigma]}-x_j^{[\sigma]}\right)}$ and $\vartheta_i\left(x_i^{[\sigma]}\right) = \frac{f\left(x_i^{[\sigma]}\right)}{\prod\limits_{\substack{i\neq j\\ j=1}}^{n}\left(x_i^{[\sigma]}-x_j^{[\sigma]}\right)}$. This numerical scheme has fourth-order convergence. The following novel q-analogies of the double-Weierstrass methods (abbreviated as Q-MM$^{\sigma_2}$) are obtained by substituting $x_j^{[\sigma]}$ with $z_j^{[\sigma]}$:

$$x_i^{[\sigma+1]} = y_i^{[\sigma]} - \vartheta_i\left(y_i^{[\sigma]}\right) = y_i^{[\sigma]} - \frac{f\left(y_i^{[\sigma]}\right)}{\prod\limits_{\substack{i\neq j\\ j=1}}^{n}\left(y_i^{[\sigma]} - y_j^{[\sigma]}\right)}, \tag{27}$$

where $y_i^{[\sigma]} = x_i^{[\sigma]} - \vartheta_i^{**}\left(x_i^{[\sigma]}\right) = x_i^{[\sigma]} - \frac{f\left(x_i^{[\sigma]}\right)}{\prod\limits_{\substack{i\neq j\\ j=1}}^{n}\left(x_i^{[\sigma]}-z_j^{[\sigma]}\right)}$. Method (27) can also be written in the form:

$$\begin{cases} y_i^{[\sigma]} = x_i^{[\sigma]} - \dfrac{f\left(x_i^{[\sigma]}\right)}{\prod\limits_{\substack{i\neq j\\ j=1}}^{n}\left(x_i^{[\sigma]}-v_j^{[\sigma]}+\frac{f\left(x_j^{\sigma]}\right)}{\left(\partial_q^{[1]}f\right)\left(v_j^{[\sigma]}\right)}\right)}, \\ x_i^{[\sigma+1]} = y_i^{[\sigma]} - \dfrac{f\left(y_i^{[\sigma]}\right)}{\prod\limits_{\substack{i\neq j\\ j=1}}^{n}\left(y_i^{[\sigma]}-y_j^{[\sigma]}\right)}, \end{cases} \tag{28}$$

where $v_j^{[\sigma]} = x_j^{[\sigma]} - \frac{f\left(x_j^{\sigma]}\right)}{\left(\partial_q^{[1]}f\right)\left(x_j^{[\sigma]}\right)}\left[\frac{1}{1-\frac{\alpha_1 f\left(x_j^{[\sigma]}\right)}{1+\alpha_2 f\left(x_j^{[\sigma]}\right)}}\right]$, and $\alpha_1, \alpha_2 \in \mathbb{R}$.

2.5. Convergence Analysis

In the following theorem, we prove the convergence order of Q-MM$^{\sigma_2}$.

Theorem 3. *Let $\zeta_1, \ldots, \zeta_\sigma$ be simple zeros of the nonlinear Equation (1). For initial distinct values $x_1^{[0]}, \ldots, x_n^{[0]}$ that are sufficiently close to the exact roots, the Q-MM$^{\sigma_2}$ method has convergence order $\left(\left|(\epsilon)^8; q\right|\right)$.*

Proof. Let $\epsilon_i = x_i^{[\sigma]} - \zeta_i, \epsilon_i' = x_i^{[\sigma+1]} - \zeta_i$ be the errors in $x_i^{[\sigma]}$, and $x_i^{[\sigma+1]}$ respectively. From the first-step of Q-MM$^{\sigma_2}$, we have:

$$y_i^{[\sigma]} - \zeta_i = x_i^{[\sigma]} - \zeta_i - \frac{f\left(x_i^{[\sigma]}\right)}{\prod\limits_{\substack{j=1\\ j\neq i}}^{n}\left(x_i^{[\sigma]} - x_j^{[\sigma]} - \frac{f\left(x_j^{[\sigma]}\right)}{\left(\partial_q^{[1]}f\right)\left(x_j^{[\sigma]}\right)}\left[1 - \frac{\alpha_1 f\left(x_j^{[\sigma]}\right)}{1+\alpha_2 f\left(x_j^{[\sigma]}\right)}\right]^{-1}\right)}, \tag{29}$$

$$\epsilon_i' = \epsilon_i - \vartheta_i^*\left(x_i^{[\sigma]}\right) = \epsilon_i - \epsilon_i\frac{\vartheta_i^*\left(x_i^{[\sigma]}\right)}{\epsilon_i} \tag{30}$$

$$\epsilon_i' = \epsilon_i\left(1 - Q_i^{[**]}\right) \tag{31}$$

where

$$Q_i^{[**]} = \frac{\vartheta_i^*\left(x_i^{[\sigma]}\right)}{\epsilon_i} = \prod_{\substack{j=1\\ j\neq i}}^{n}\left[\frac{x_i^{[\sigma]}-\zeta_j}{x_i^{[\sigma]}-\overset{*}{x}{}_j^{[\sigma]}}\right] \tag{32}$$

and $z_j^{[\sigma]} = v_j^{[\sigma]} - \frac{f\left(v_j^{[\sigma]}\right)}{\left(\partial_q^{[1]}f\right)\left(v_j^{[\sigma]}\right)}$, $v_j^{[\sigma]} = x_j^{[\sigma]} - \frac{f\left(x_j^{[\sigma]}\right)}{\left(\partial_q^{[1]}f\right)\left(x_j^{[\sigma]}\right)}\left[1-\frac{\alpha_1 f\left(x_j^{[\sigma]}\right)}{1+\alpha_2 f\left(x_j^{[\sigma]}\right)}\right]^{-1}$. Therefore, we can write

$$\frac{x_i^{[\sigma]}-\zeta_j}{x_i^{[\sigma]}-z_j^{[\sigma]}} = 1 + \frac{z_j^{[\sigma]}-\zeta_j}{x_i^{[\sigma]}-z_j^{[\sigma]}}1 + O\left(\left|\epsilon_j^3;q\right|\right) \tag{33}$$

and, consequently,

$$\begin{aligned} Q_i^{[**]} &= \prod_{\substack{j=1\\ j\neq i}}^{n}\left[\frac{z_j^{[\sigma]}-\zeta_j}{x_i^{[\sigma]}-z_j^{[\sigma]}}\right] = \left(1+O\left(\left|\epsilon_j^3\right|\right)\right)^{n-1} \\ &= 1+(n-1)O\left(\left|\epsilon^3\right|\right) = 1+O\left(\left|\epsilon_j^3\right|\right). \end{aligned} \tag{34}$$

We conclude that

$$Q_i^{[**]} - 1 = O\left(\left|\epsilon_j^3;q\right|\right). \tag{35}$$

If $|\epsilon_i|$ and $|\epsilon_j|$ have same order, then we have

$$\epsilon_i' = \epsilon_i\left(O\left(\left|\epsilon_j^3;q\right|\right)\right) = O\left(\left|\epsilon^4;q\right|\right). \tag{36}$$

Using the second step of Q-MM$^{\sigma_2}$, we get:

$$x_i^{[\sigma+1]} - \zeta_i = y_i^{[\sigma]} - \zeta_i - \frac{f\left(y_i^{[\sigma]}\right)}{\prod_{\substack{j=1\\ j\neq i}}^{n}\left(y_i^{[\sigma]}-y_j^{[\sigma]}\right)}, \tag{37}$$

$$\epsilon_i'' = \epsilon_i' - \vartheta_i\left(y_i^{[\sigma]}\right) = \epsilon_i' - \epsilon_i'\frac{\vartheta_i\left(y_i^{[\sigma]}\right)}{\epsilon_i'}, \tag{38}$$

$$\epsilon_i'' = \epsilon_i'\left(1-E_i^{[*]}\right), \tag{39}$$

where

$$E_i^{[*]} = \frac{\vartheta_i\left(y_i^{[\sigma]}\right)}{\epsilon_i'} = \prod_{\substack{j=1\\ j\neq i}}^{n}\left[\frac{y_i^{[\sigma]}-\zeta_j}{y_i^{[\sigma]}-y_j^{[\sigma]}}\right]. \tag{40}$$

Let

$$\frac{y_i^{[\sigma]}-\zeta_j}{y_i^{[\sigma]}-y_j^{[\sigma]}} = 1 + \frac{y_j^{[\sigma]}-\zeta_j}{y_i^{[\sigma]}-y_j^{[\sigma]}}1 + O\left(\left|\left(\epsilon_j'\right)\right|;q\right), \tag{41}$$

$$E_i^{[*]} = \prod_{\substack{j=1\\ j\neq i}}^{n}\left[\frac{y_i^{[\sigma]}-\zeta_j}{y_i^{[\sigma]}-y_j^{[\sigma]}}\right] = \left(1+O\left(\left|\epsilon_j'\right|;q\right)\right)^{n-1}, \tag{42}$$

$$= 1+(n-1)O\left(\left|\epsilon_j'\right|\right) = 1+O\left(\left|\epsilon_j'\right|;q\right), \tag{43}$$

$$E_i^{[*]} - 1 = O\left(\left|\left(\epsilon_j'\right)\right|; q\right), \tag{44}$$

$$\epsilon_i'' = \epsilon_i'\left(O\left(\left|\left(\epsilon_j'\right)\right|; q\right)\right). \tag{45}$$

If $|\epsilon_i|$ and $|\epsilon_j|$ have same order, then we have

$$\epsilon_i'' = O\left(\left|(\epsilon')^2; q\right|\right) = O\left(\left|\left(\epsilon^4\right)^2; q\right|\right). \tag{46}$$

$$\epsilon_i'' = O\left(\left|(\epsilon)^8; q\right|\right). \tag{47}$$

This proves that the convergence order of Q-MM$^{\sigma_2}$ is $O\left(\left|(\epsilon)^8; q\right|\right)$. □

3. Neural Network-Based Q-Analogies of the Numerical Scheme

Neural networks have been proven to be extremely versatile computational techniques for the solution of numerous complex engineering problem, due to their ability to represent complex relationships and to learn from data [37–39]. These features make them effective modelling tools for a wide range of applications, including the development of iterative root finding procedures. See, for example, Daws et al. [40], Mourrain et al. [41], Huang et al. [42], Freitas et al. [43], Shams et al. [44] and references cited therein. Utilizing neural network techniques to improve the efficiency and accuracy of the q-analogies of our parallel numerical root finding schemes involves a two-step approach. First, a neural network is trained to approximate the roots of a polynomial using the polynomial's coefficients. The network learns the complex relationship between the input coefficients and the corresponding roots during this phase. Once trained, the neural network produces preliminary estimates for the polynomial roots. These estimates are then used as starting values for the parallel system and refined iteratively until convergence. This hybrid technique has the potential to accelerate the convergence process by leveraging the neural network's ability to capture the intricate mapping between the polynomial's coefficients and roots, resulting in improved initial estimates [45]. However, the success of this strategy depends on the complexity of the polynomial and the accuracy of the neural network's approximation.

To estimate all roots of nonlinear equations using neural network-based q-analogies of our parallel scheme, the following steps were required:

- *Data representation*. Prepare the data sets by feeding them into the neural network with the coefficients of nonlinear functions, particularly higher-degree polynomials. Table in Appendex B, presents the upper edge-data set used in PNN to approximate all roots of (1). The data set contains 5000 archives. Using Symbolic Math Toolbox in Maple, random polynomial co-efficient in the range of [0, 1] were generated (see Appendices B and C). The PNN were trained using 70% of the sample of these data and remaining 30% of the data was utilized to find the generalization capability of the PNN.
- *Architecture*. Two input/output layers were required to develop a neural network architecture capable of approximating the roots of nonlinear equations. The size of the input layer should match the number of polynomial coefficients. The size of the neural network's output layer should correspond to the set of real or complex polynomial roots, as indicated in Figures 1 and 2.
- *Training*. The neural network is trained with three layers (input, two hidden layers, and output layer) to approximating all the roots of nonlinear polynomial equations using well-known Levenberg-Marquardt Algorithm (LMA). The weights of the PNNS con-

nections are modified based on the discrepancy between the predicated and computed values; the error are approximated as

$$\vartheta^{[\sigma+1]} = \vartheta^{[\sigma]} - \left(\left(\hat{\jmath}^{[\sigma]} \right)^{t} \hat{\jmath}^{[\sigma]} + \lambda^{[\sigma]} I \right)^{-1} \left(\left(\hat{\jmath}^{[\sigma]} \right) e^{[\sigma]} \right), \tag{48}$$

where $\hat{\jmath}^{[\sigma]} = \frac{\partial e^{[\sigma]}}{\partial \vartheta^{[\sigma]}}$ and I is the identity matrix. The mean square error (MSE) is computed as

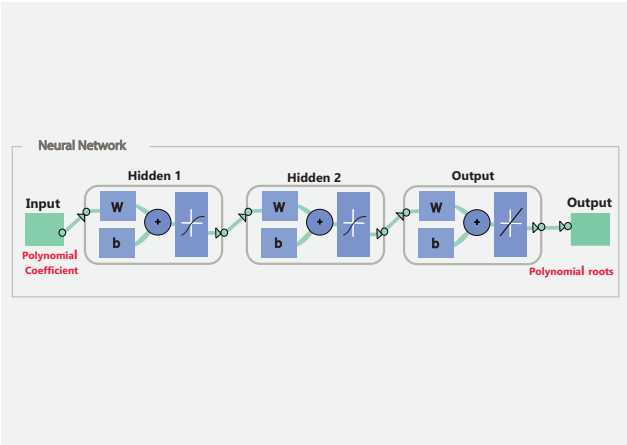


Figure 1. A schematic representation of the process of feeding the coefficients of a polynomial into an Artificial Neural Network (ANN), which then yields an approximation for each root of Equation (1).

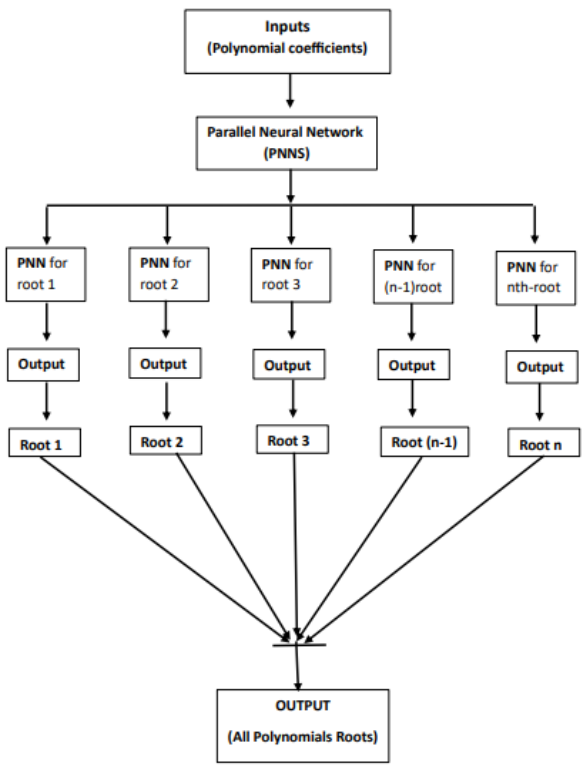


Figure 2. A schematic representation of the process of feeding the coefficients of a polynomial into an Parallel Neural Network (PNNS), which then yields an approximation for each root of Equation (1).

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(\gamma_{11} - \gamma_{12}), \tag{49}$$

where γ_{11} = Exact ith-root in the dat set and γ_{12} = approximate value using $Q - MM^{\sigma_2}$.

- *Enhancement of neural network accuracy*. To increase efficiency and accuracy, we update the neural network's outputs using q-analogies of parallel numerical algorithms.

Algorithm 1 described below utilizes a neural network with q-analogies of the parallel numerical scheme introduced in this paper, to estimate all roots of nonlinear equations in MATLAB.

Algorithm 1: Neural network with q-analogies of the parallel numerical scheme in MATLAB.

```
Function Parallel_Numerical_Scheme (Initial_Gusses)
root=initial gusses=[]
Tolerance=set_Tolerance ()
Max_iteration=set_max_iterations()
Update=neural_network_update(Polynomial _coefficient,root[i]/denominator)+Update
new_root-=evaluate_polynomial(Polynomial _coefficient,root[i])/denominator)+Update
new_roots*approximate(new_root)
Change=max(abc(new_root-old_root)
roots=new_roots
iterations+=1
If iterations==Max_iterations:
Print("Roots maynot have converged within the maximum iterations")
else
print("Root:",roots)
For iterations in range(max_iterations)
previous_roots=roots.copy()
For i in range(len(roots))
denominator=1
For j in range(len(roots))
if j !=i
denominator*=root-root[j]
Correction=polynomial_value(roots)/denominator
root[i]=Correction
if check_convergence(roots,prev_roots,tolerance)
break
returen roots
end do
end do
end do
end do
end do
End do.
```

4. Numerical Results

Some non-linear problems from biomedical engineering and applied sciences are considered to illustrate the performance and efficiency of Q-MM$^{\sigma_1}$ and Q-MM$^{\sigma_2}$. The experiments are performed using CAS Maple 18 with 64 digits floating point arithmetic and the following stopping criterion:

$$e_i^{(\sigma)} = \left| x_i^{(\sigma+1)} - \zeta \right| < \in,$$

where $e_i^{(\sigma)}$ represents the absolute error. We set $\in = 10^{-30}$ as the tolerance used in the stopping criterion. In Tables 1–22, ρ_i represents the local convergence order of our iterative schemes. Here, we compare our newly developed methods Q-MM$^{\sigma_1}$ −Q-MM$^{\sigma_2}$ with Borch-Supan method (abbreviated as BSM^{C_3}) [46] of convergence order 3, defined as:

$$x_i^{(\sigma+1)} = x_i^{(\sigma)} - \frac{\vartheta_i\left(x_i^{[\sigma]}\right)}{1 + \sum\limits_{\substack{j\neq i\\ j=1}}^{n}\left(\frac{\vartheta_i\left(x_i^{[\sigma]}\right)}{\left(x_i^{(\sigma)} - x_j^{(\sigma)}\right)}\right)}, \tag{50}$$

where $\vartheta_i\left(x_i^{[\sigma]}\right) = \left(\frac{f\left(x_i^{[\sigma]}\right)}{\prod\limits_{\substack{i\neq j\\ j=1}}^{n}\left(x_i^{[\sigma]} - x_j^{[\sigma]}\right)}\right)$. Rafiq et al. [47] proposed the following two derivative free simultaneous method (abbreviated as NAM^{C_3}) of convergence order 3 defined as:

$$\begin{cases} y_i^{(\sigma)} = x_i^{(\sigma)} - \left(\vartheta_i\left(x_i^{[\sigma]}\right)\right), \\ x_i^{(\sigma+1)} = x_i^{(\sigma)} - \left(\vartheta_i\left(x_i^{[\sigma]}\right)\right)\left(\frac{f(x_i^{(\sigma)}) + f(y_i^{(\sigma)})}{f(x_i^{(\sigma)})}\right). \end{cases} \tag{51}$$

Nedzibove et al. [48] introduced the following two derivative free simultaneous method (abbreviated as NDM^{C_3}) of convergence order 3 defined as:

$$\begin{cases} y_i^{(\sigma)} = x_i^{(\sigma)} - \left(\vartheta_i\left(x_i^{[\sigma]}\right)\right), \\ x_i^{(\sigma+1)} = x_i^{(\sigma)} - \left(\vartheta_i\left(x_i^{[\sigma]}\right)\right)\left(1 + \frac{f(y_i^{(\sigma)})}{f(x_i^{(\sigma)}) - 2\lambda y_i^{(\sigma)})}\right), \end{cases} \quad \lambda \in \mathbb{R} \tag{52}$$

Mir et al. [49] proposed a method (abbreviated as NAM^{C_8}) of convergence order 8 defined as:

$$\begin{cases} z_i^{[\sigma]} = x_i^{(\sigma)} - \frac{f(x_j^{(\sigma)})}{f'(x_j^{(\sigma)})}, \\ y_i^{(\sigma)} = x_i^{(\sigma)} - \frac{1}{\left(\frac{1}{N_i(x_i^{(\sigma)})}\right) - \sum\limits_{\substack{j=1\\ j\neq i}}^{n}\left(\frac{1}{(x_i^{(\sigma)} - Z_j^{(\sigma)}))}\right) - \alpha}, \\ y_i^{(\sigma)} = x_i^{(\sigma)} - \frac{1}{\left(\frac{1}{N_i(x_i^{(\sigma)})}\right) - \sum\limits_{\substack{j=1\\ j\neq i}}^{n}\left(\frac{1}{(x_i^{(\sigma)} - Z_j^{(\sigma)}))}\right)}, \end{cases}$$

where $N_i(x_i^{(\sigma)}) = \frac{f(x_j^{(\sigma)})}{f'(x_j^{(\sigma)})}$ and $\alpha \in \mathbb{R}$. Thangavel et al. [50] proposed a neutral-type of switched neural networks (abbreviated as TAM^{C_*}) to solve nonlinear problems. We consider all these methods in the comparative performance analysis of our new parallel schemes. To determine all the roots of nonlinear equations, we utilized Algorithms 2 and 3.

Table 1. Error analysis and roots approximation using the Q-MM$^{\sigma_1}$ method with $X_1^{[0]}$ on engineering application 1.

$\mathbf{X_1^{[0]}} = \left[\mathbf{X_{11}^{[0]}}, \mathbf{X_{12}^{[0]}}, \mathbf{X_{13}^{[0]}}, \ldots\right]$									
q	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	1.1
$x_1^{[0]}$	$0.03 - 3.8i$	$-2.0 \times 10^{-5} - 3.7i$	$2.0 - 1.2 \times 10^{-49}i$	$-2.0 \times 10^{-5} - 3.7i$	$3.0 + 2.6 \times 10^{-38}i$	$3.0 - 8.1 \times 10^{-34}$i	$-2.0 - 1.4 \times 10^{-59}i$	$3.0 - 7.5 \times 10^{-96}i$	$3.0 - 7.5 \times 10^{-96}$i
$x_2^{[0]}$	$4.3 - 3.5i$	$3.0 + 4.6 \times 10^{-30}i$	$3.0 + 3.5 \times 10^{-5}i$	$3.0 + 4.6 \times 10^{-30}i$	$2 - 1.8 \times 10^{-35}i$	$1.5 \times 10^{-51} - 3.8i$	$3.0 + 2.6 \times 10^{-56}i$	$2.0 - 1.2 \times 10^{-94}$i	$2.0 - 1.2 \times 10^{-94}i$
$x_3^{[0]}$	$1.9 + 1.6i$	$2.0 - 7.7 \times 10^{-38}i$	$2.1 \times 10^{-63} - 3.8i$	$2.0 - 7.7 \times 10^{-38}i$	$7.8 \times 10^{-50} - 3.8i$	$2.0 + 1.2 \times 10^{-30}i$	$1.3 \times 10^{-62} + 38i$	$-5.2 - 7.7 \times 10^{-38}i$	$-5.2 - 7.7 \times 10^{-38}i$
$x_4^{[0]}$	$-0.04 + 3.8i$	$-4.0 \times 10^{-51} + 3.8i$	$2.4 \times 10^{-64} + 3.8i$	$-4.0 \times 10^{-51} + 3.8i$	$-1.9 \times 10^{-49} + 3.8i$	$6.0 \times 10^{-49} + 3.8i$	$4.0 \times 10^{-53} - 3.8i$	$2.4 \times 10^{-63} + 3.8i$	$2.4 \times 10^{-64} + 3.8i$
$\Lambda_1^{[*]}$	0.11	4.3×10^{-30}	8.8×10^{-25}	4.3×10^{-30}	2.9×10^{-20}	3.3×10^{-18}	6.4×10^{-27}	2.8×10^{-47}	1.8×10^{-49}
$\Lambda_2^{[*]}$	8.16	1.3×10^{-20}	2.0×10^{-26}	1.3×10^{-20}	1.3×10^{-17}	3.8×10^{-33}	8.3×10^{-29}	4.7×10^{-46}	0.7×10^{-47}
$\Lambda_3^{[*]}$	1.66	5.3×10^{-11}	1.0×10^{-37}	5.3×10^{-11}	1.3×10^{-29}	4.7×10^{-15}	1.7×10^{-33}	6.3×10^{-50}	9.0×10^{-50}
$\Lambda_4^{[*]}$	0.08	8.1×10^{-31}	2.6×10^{-37}	8.1×10^{-29}	2.6×10^{-30}	2.0×10^{-32}	8.1×10^{-33}	0.1×10^{-51}	0.1×10^{-55}
E_{max}^{time}	6.10241	4.12154	3.14561	3.51423	4.31414	3.12478	2.01245	1.012451	1.12431

Table 2. Max-Error for the Q-MM$^{\sigma_1}$ method using $X_1^{[0]}$.

$[\text{Q-MM}^{\sigma_1}; \mathbf{q}]$	$e_1^{(3)}$	$e_2^{(3)}$	$e_3^{(3)}$
$[\text{Q-MM}^{\sigma_2}; 1.1]$	4.0×10^{-49}	3.6×10^{-47}	9.9×10^{-50}
$[\text{Q-MM}^{\sigma_2}; 0.9]$	4.33×10^{-47}	6.3×10^{-46}	3.5×10^{-50}
$[\text{Q-MM}^{\sigma_2}; 0.8]$	4.3×10^{-27}	3.6×10^{-29}	4.5×10^{-33}
$[\text{Q-MM}^{\sigma_2}; 0.7]$	5.3×10^{-18}	6.5×10^{-33}	9.9×10^{-15}
$[\text{Q-MM}^{\sigma_2}; 0.6]$	3.1×10^{-20}	6.5×10^{-17}	6.3×10^{-29}
$[\text{Q-MM}^{\sigma_2}; 0.5]$	3.2×10^{-26}	9.3×10^{-36}	5.6×10^{-29}
$[\text{Q-MM}^{\sigma_2}; 0.4]$	4.6×10^{-26}	3.9×10^{-36}	1.4×10^{-35}
$[\text{Q-MM}^{\sigma_2}; 0.3]$	1.2×10^{-30}	6.3×10^{-20}	7.1×10^{-19}
$[\text{Q-MM}^{\sigma_2}; 0.2]$	2.0×10^{-2}	3.1×10^{-2}	1.2×10^{-3}

Table 3. Number of iterations for Q-MM$^{\sigma_1}$ using $X_1^{[0]}$.

$[\text{Q-MM}^{\sigma_1}; \mathbf{q}]$	It-$e_1^{[\sigma]}$	It-$e_2^{[\sigma]}$	It-$e_3^{[\sigma]}$
$[\text{Q-MM}^{\sigma_2}; 1.1]$	16	16	16
$[\text{Q-MM}^{\sigma_2}; 0.9]$	16	16	16
$[\text{Q-MM}^{\sigma_2}; 0.8]$	18	18	18
$[\text{Q-MM}^{\sigma_2}; 0.7]$	20	20	20
$[\text{Q-MM}^{\sigma_2}; 0.6]$	26	26	26
$[\text{Q-MM}^{\sigma_2}; 0.5]$	27	27	27
$[\text{Q-MM}^{\sigma_2}; 0.4]$	28	28	28
$[\text{Q-MM}^{\sigma_2}; 0.3]$	85	85	85
$[\text{Q-MM}^{\sigma_2}; 0.2]$	100	100	100

Table 4. CPU-time for the Q-MM$^{\sigma_1}$ method using $X_1^{[0]}$.

$[\text{Q-MM}^{\sigma_1}; \mathbf{q}]$	CT-$e_1^{(\sigma)}$	CT-$e_2^{(\sigma)}$	CT-$e_3^{(\sigma)}$
$[\text{Q-MM}^{\sigma_2}; 1.1]$	1.94215	1.9415	1.8451
$[\text{Q-MM}^{\sigma_2}; 0.9]$	2.14561	2.54165	2.14554
$[\text{Q-MM}^{\sigma_2}; 0.8]$	3.0124	3.00124	3.01245
$[\text{Q-MM}^{\sigma_2}; 0.7]$	3.12415	3.24156	3.21451
$[\text{Q-MM}^{\sigma_2}; 0.6]$	3.54126	3.84561	3.98451
$[\text{Q-MM}^{\sigma_2}; 0.5]$	4.00121	4.00125	4.00165
$[\text{Q-MM}^{\sigma_2}; 0.4]$	4.01234	4.12013	4.18745
$[\text{Q-MM}^{\sigma_2}; 0.3]$	5.01242	5.12415	5.1421
$[\text{Q-MM}^{\sigma_2}; 0.2]$	5.12455	5.14215	6.1425

Table 5. Error analysis and roots approximation using the Q-MM$^{\sigma_2}$ method with $X_1^{[0]}$ on engineering application 1.

$X_1^{[0]} = \left[X_{11}^{[0]}, X_{12}^{[0]}, X_{13}^{[0]}, \ldots\right]$									
q	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	1.1
$x_1^{[0]}$	$0.03 - 3.8i$	$-2.0 \times 10^{-50} - 3.7i$	$2.0 - 1.2 \times 10^{-49}i$	$-2.0 \times 10^{-50} - 3.7i$	$3.0 + 2.6 \times 10^{-38}i$	$3.0 - 8.1 \times 10^{-34}i$	$-2.0 - 1.4 \times 10^{-59}i$	$3.0 - 7.5 \times 10^{-96}i$	$3.0 - 7.5 \times 10^{-96}i$
$x_2^{[0]}$	$4.3 - 3.5i$	$3.0 + 4.6 \times 10^{-30}i$	$3.0 + 3.5 \times 10^{-50}i$	$3.0 + 4.6 \times 10^{-30}i$	$2 - 1.8 \times 10^{-35}i$	$1.5 \times 10^{-51} - 3.8i$	$3.0 + 2.6 \times 10^{-56}i$	$2.0 - 1.2 \times 10^{-94}i$	$2.0 - 1.2 \times 10^{-94}i$
$x_3^{[0]}$	$1.9 + 1.6i$	$2.0 - 7.7 \times 10^{-38}i$	$2.1 \times 10^{-63} - 3.8i$	$2.0 - 7.7 \times 10^{-38}i$	$7.8 \times 10^{-50} - 3.8i$	$2.0 + 1.2 \times 10^{-30}i$	$1.3 \times 10^{-52} + 38i$	$-5.2 \times 10^{-64} - 7.7i$	$-5.2 \times 10^{-65} - 7.7i$
$x_4^{[0]}$	$-0.04 + 3.8i$	$-4.0 \times 10^{-51} + 3.8i$	$2.4 \times 10^{-64} + 3.8i$	$-4.0 \times 10^{-51} + 3.8i$	$-1.9 \times 10^{-49} + 3.8i$	$6.0 \times 10^{-49} + 3.8i$	$4.0 \times 10^{-53} - 3.8i$	$2.4 \times 10^{-63} + 3.8i$	$2.4 \times 10^{-64} + 3.8i$
$\Lambda_1^{[*]}$	5.1×10^{-55}	1.3×10^{-64}	4.8×10^{-63}	4.3×10^{-52}	2.9×10^{-45}	6.3×10^{-18}	6.4×10^{-55}	4.1×10^{-64}	1.1×10^{-64}
$\Lambda_2^{[*]}$	2.0×10^{-58}	0.3×10^{-64}	0.0	0.7×10^{-53}	6.3×10^{-45}	9.8×10^{-33}	8.3×10^{-55}	2.2×10^{-74}	9.7×10^{-65}
$\Lambda_3^{[*]}$	5.1×10^{-55}	5.9×10^{-86}	1.5×10^{-86}	7.3×10^{-53}	6.0×10^{-46}	9.7×10^{-15}	6.7×10^{-45}	0.3×10^{-64}	9.9×10^{-70}
$\Lambda_4^{[*]}$	2.3×10^{-57}	8.4×10^{-86}	5.6×10^{-86}	7.1×10^{-54}	8.6×10^{-46}	0.1×10^{-32}	6.1×10^{-56}	8.8×10^{-75}	6.0×10^{-71}
E_{max}^{time}	6.10241	4.12154	3.14561	3.51423	4.31414	3.12478	2.01245	1.012451	1.12431

Algorithm 2: For q-Numerical scheme Q-MM$^{\sigma_1}$.

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

Calculate $x_j^{[\sigma]} - \frac{f\left(x_j^{[\sigma]}\right)}{\left(Đ_q^{[1]} f\right)\left(x_j^{[\sigma]}\right)} \left[\frac{1}{1 - \frac{\alpha_1 f\left(x_j^{[\sigma]}\right)}{1+\alpha_2 f\left(x_j^{[\sigma]}\right)}} \right]$

Update $x_i^{[\sigma+1]} = x_i^{[\sigma]} - \vartheta_i^*\left(x_i^{[\sigma]}\right)$.

$x_i^{[\sigma+1]} = x_i^{[\sigma]}(ii = 1, \ldots, n)$.

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

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

End do.

Algorithm 3: For q-Numerical scheme Q-MM$^{\sigma_2}$.

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

Calculate $\begin{cases} v_j^{[\sigma]} = x_j^{[\sigma]} - \frac{f(x_j^{[\sigma]})}{\left(\eth_q^{[1]} f\right)(x_j^{[\sigma]})} \left[\frac{1}{1 - \frac{\alpha_1 f(x_j^{[\sigma]})}{1+\alpha_2 f(x_j^{[\sigma]})}} \right], \\ z_j^{[\sigma]} = v_j^{[\sigma]} - \frac{f(v_j^{[\sigma]})}{\left(\eth_q^{[1]} f\right)(x_j^{[\sigma]})}, \end{cases}$

Update $x_i^{[\sigma+1]} = \begin{cases} y_i^{[\sigma]} = x_i^{[\sigma]} - \frac{f\left(x_i^{[\sigma]}\right)}{\prod_{\substack{i \neq j \\ j=1}}^{n} \left(x_i^{[\sigma]} - v_j^{[\sigma]} + \frac{f\left(v_j^{[\sigma]}\right)}{\left(\eth_q^{[1]} f\right)\left(v_j^{[\sigma]}\right)} \right)}, \\ x_i^{[\sigma+1]} = y_i^{[\sigma]} - \frac{f\left(y_i^{[\sigma]}\right)}{\prod_{\substack{i \neq j \\ j=1}}^{n} \left(y_i^{[\sigma]} - y_j^{[\sigma]} \right)}, \end{cases}$

$x_i^{[\sigma+1]} = x_i^{[\sigma]}(ii = 1, \ldots, n)$.

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

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

End do.

Some biomedical engineering examples [51–54] are shown in this section to assess the effectiveness of the newly developed q-analogies of the parallel method for locating all zeros of (1) simultaneously.

4.1. Engineering Application 1: Osteoporosis in Chinese Women

Wu et al. [52] investigated age-related changes in tibial sound speed and osteoporosis prevalence among Chinese women. The nonlinear relationship between sound speed and age was found to be as follows:

$$f(x) = 0.0039x^3 - 0.78x^2 + 39.9x - 467. \tag{53}$$

The exact solution of (53) up to four decimal places is

$$\zeta_1 = 16.7023, \zeta_2 = 56.5740, \zeta_3 = 126.7235.$$

To analyze global convergence, as in [53], we use the randomly generated initial guesses $X_1^{[0]} = \left[X_{11}^{[0]}, X_{12}^{[0]}, X_{13}^{[0]}, \ldots\right]$ shown in Appendix B, Table A1.

In Tables 1–11, we evaluate the numerical results produced with our parallel numerical scheme for various values of q. In Table 1, we report on the error and approximate solution up to two decimal places whereas the approximated root is computed up to 64-digit floating point arithmetic using the stopping criterion $\Lambda_i^{[*]} = \left| x_i^{[\sigma+1]} - x_i^{[\sigma]} \right|$. The maximum error, number of iterations, and elapsed CPU time for the Q-MM$^{\sigma_1}$ method for various values of q are shown in Tables 2–4. On the other hand, Tables 5–9 displays the numerical results of Q-MM$^{\sigma 2}$ in terms of approximated values, residual errors, maximum errors, number of iterations, and elapsed CPU time for finding all roots of (53) using different values of q. The rate of convergence of Q-MM$^{\sigma_1}$-Q-MM$^{\sigma_2}$ increases significantly as the q values increase from 0.2 to 1.1 on random initial values, demonstrating global convergence behavior.

Table 6. Max-Error for the Q-MM$^{\sigma_2}$ method using $X_1^{[0]}$.

[Q-MM$^{\sigma_2}$; q]	$e_1^{(3)}$	$e_2^{(3)}$	$e_3^{(3)}$
[Q-MM$^{\sigma_2}$; 1.1]	1.4×10^{-19}	1.7×10^{-15}	1.3×10^{-14}
[Q-MM$^{\sigma_2}$; 0.99]	2.2×10^{-21}	1.1×10^{-18}	3.2×10^{-21}
[Q-MM$^{\sigma_2}$; 0.9]	2.1×10^{-15}	2.7×10^{-14}	2.1×10^{-15}
[Q-MM$^{\sigma_2}$; 0.8]	4.2×10^{-25}	3.1×10^{-31}	4.2×10^{-25}
[Q-MM$^{\sigma_2}$; 0.7]	2.1×10^{-15}	2.7×10^{-14}	2.1×10^{-15}
[Q-MM$^{\sigma_2}$; 0.6]	4.2×10^{-25}	3.1×10^{-31}	4.2×10^{-25}
[Q-MM$^{\sigma_2}$; 0.5]	2.1×10^{-15}	2.7×10^{-14}	2.1×10^{-15}
[Q-MM$^{\sigma_2}$; 0.4]	4.2×10^{-25}	3.1×10^{-31}	4.2×10^{-25}
[Q-MM$^{\sigma_2}$; 0.3]	4.2×10^{-25}	3.1×10^{-31}	4.2×10^{-25}

Table 7. Number of iterations of Q-MM$^{\sigma_2}$ using $X_1^{[0]}$.

[Q-MM$^{\sigma_2}$; q]	It-$e_1^{[\sigma]}$	It-$e_2^{[\sigma]}$	It-$e_3^{[\sigma]}$	It-$e_4^{[\sigma]}$
[Q-MM$^{\sigma_2}$; 1.1]	5	5	5	5
[Q-MM$^{\sigma_2}$; 0.9]	5	5	5	5
[Q-MM$^{\sigma_2}$; 0.8]	7	7	7	7
[Q-MM$^{\sigma_2}$; 0.7]	8	8	8	8
[Q-MM$^{\sigma_2}$; 0.6]	8	8	8	8
[Q-MM$^{\sigma_2}$; 0.5]	11	11	11	11
[Q-MM$^{\sigma_2}$; 0.4]	14	14	14	14
[Q-MM$^{\sigma_2}$; 0.3]	17	17	17	17
[Q-MM$^{\sigma_2}$; 0.2]	22	22	22	22

Table 8. CPU-time Q-MM$^{\sigma_2}$ using $X_1^{[0]}$.

[Q-MM$^{\sigma_2}$; q]	CT-$e_1^{(3)}$	CT-$e_2^{(3)}$	CT-$e_3^{(3)}$
[Q-MM$^{\sigma_2}$; 1.1]	0.04215	0.0415	0.0451
[Q-MM$^{\sigma_2}$; 1.0]	0.14061	0.54105	0.14004
[Q-MM$^{\sigma_2}$; 0.9]	2.0124	2.00124	2.01245
[Q-MM$^{\sigma_2}$; 0.8]	2.12415	2.3056	2.21451
[Q-MM$^{\sigma_2}$; 0.7]	3.54126	3.84561	3.98451
[Q-MM$^{\sigma_2}$; 0.6]	3.10111	3.00125	3.4126
[Q-MM$^{\sigma_2}$; 0.5]	4.21234	4.15123	4.18745
[Q-MM$^{\sigma_2}$; 0.4]	4.01242	4.12425	4.64201
[Q-MM$^{\sigma_2}$; 0.3]	4.12455	4.13125	4.14112

Table 9. Error outcomes using neural networks on application 1.

Method	$e_1^{(\sigma)}$	$e_2^{(\sigma)}$	$e_3^{(\sigma)}$	$\rho_i^{[\sigma-1]}$
[Q-MM$^{\sigma_1}$; 1.0]	4.2×10^{-47}	3.1×10^{-46}	4.2×10^{-50}	3.4714
TAM^{C_*}	6.9×10^{-25}	0.4×10^{-25}	8.5×10^{-31}	3.0032
BSM^{C_3}	2.1×10^{-25}	2.7×10^{-24}	2.1×10^{-25}	2.7451
NAM^{C_3}	4.2×10^{-25}	3.1×10^{-13}	4.2×10^{-30}	2.1452
NDM^{C_3}	2.1×10^{-15}	2.7×10^{-14}	2.1×10^{-15}	3.0145
NAM^{C_8}	4.2×10^{-57}	3.1×10^{-56}	4.2×10^{-60}	7.61452
[Q-MM$^{\sigma_2}$; 1.0]	2.1×10^{-65}	2.7×10^{-64}	2.1×10^{-73}	8.0124

Table 10. Improvement in convergence rate using neural network outcomes as input for the parallel root finding scheme.

Method	$e_1^{(3)}$	$e_2^{(3)}$	$e_3^{(3)}$	$\rho_i^{[\sigma-1]}$
[Q-MM$^{\sigma_1}$; 1.0]	1.4×10^{-19}	1.7×10^{-15}	1.3×10^{-14}	3.1024
TAM^{C_*}	6.1×10^{-25}	7.7×10^{-19}	7.8×10^{-17}	3.1332
BSM^{C_3}	2.1×10^{-15}	2.7×10^{-14}	2.1×10^{-15}	3.0612
NAM^{C_3}	4.2×10^{-25}	3.1×10^{-31}	4.2×10^{-25}	3.0071
NDM^{C_3}	2.1×10^{-15}	2.7×10^{-14}	2.1×10^{-15}	2.9981
NAM^{C_8}	4.2×10^{-25}	3.1×10^{-31}	4.2×10^{-25}	7.4751
[Q-MM$^{\sigma_2}$; 1.0]	2.1×10^{-15}	2.7×10^{-14}	2.1×10^{-15}	8.0182

Table 11. Overall results of q-analogies-based neural network outcomes for accurate initial guesses.

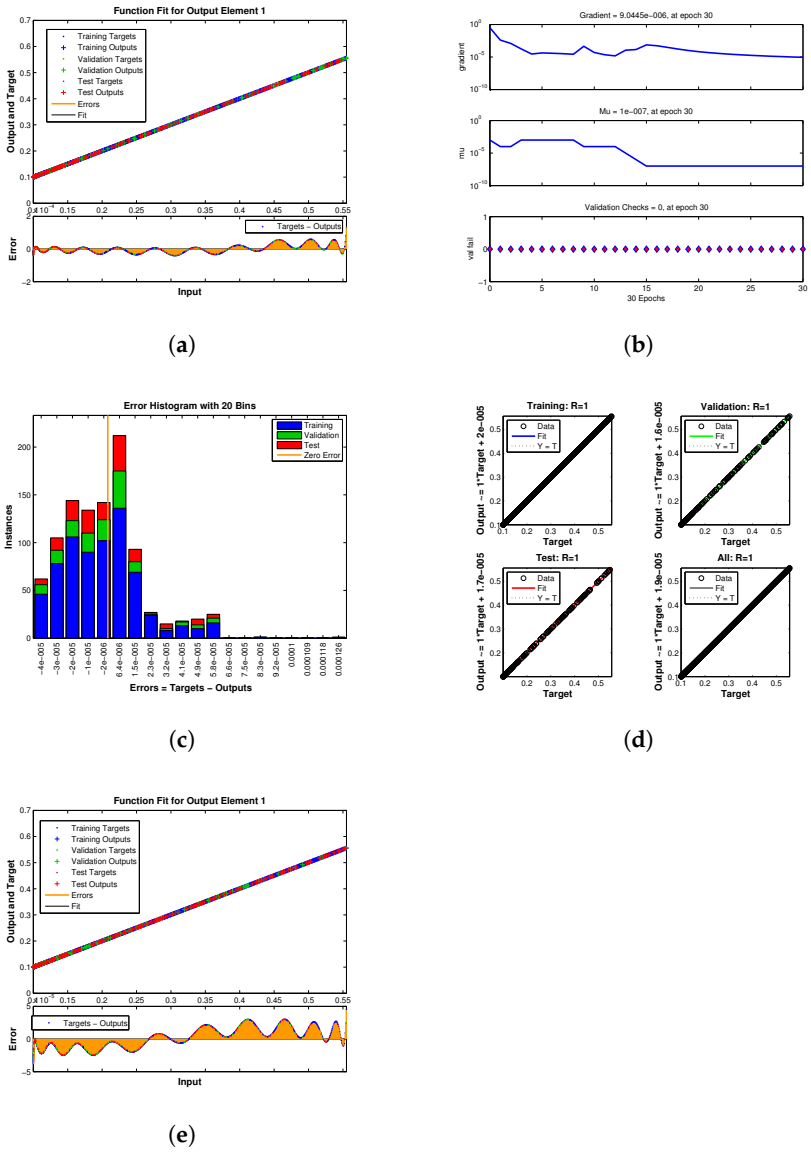
Method	Max-Err$_{\Lambda_1^{[*]}}$	Max-it$_{\Lambda_1^{[*]}}$	Max-CPU$_{\Lambda_1^{[*]}}$	Max-COC$_{\Lambda_1^{[*]}}$	$\rho_i^{[\sigma-1]}$
[Q-MM$^{\sigma_1}$; 1.0]	1.4×10^{-19}	1.3×10^{-14}	2.1×10^{-15}	3.314246	3.4219
TAM^{C_*}	0.7×10^{-27}	0.1×10^{-25}	5.1×10^{-29}	2.013325	3.0106
BSM^{C_3}	2.2×10^{-21}	3.2×10^{-21}	6.1×10^{-30}	2.981454	3.1452
NAM^{C_3}	2.1×10^{-15}	2.1×10^{-15}	2.7×10^{-14}	3.001245	2.9874
NDM^{C_3}	4.2×10^{-25}	4.2×10^{-25}	3.1×10^{-31}	3.012445	3.0145
NAM^{C_8}	2.1×10^{-51}	2.1×10^{-15}	2.7×10^{-14}	7.954154	7.6845
[Q-MM$^{\sigma_2}$; 1.0]	4.2×10^{-25}	4.2×10^{-25}	3.1×10^{-31}	8.012416	8.3541

Figure 3a–e illustrates the neural network implementation and its results using Algorithm 1. The error histogram curve of the neural network demonstrates the consistency of the proposed scheme; the transition statistics curve reflects the effective convergence rate of the neural network; the fitness curve shows accuracy and stability; the regression curve illustrates the linear relationship between the expected and actual outcomes; and the mean square error demonstrates how well the target solution and expected outcomes matched. Figure 3 displays the neural network's outputs in the following ways: (a) the error histogram; (b) the transition statistics curve; (c) the mean square error; (d) the regression curve; and (e) the fitness curve. As illustrated in Figure 3a–e and Table 9, neural network simulations for this engineering application produce reliable and consistent results. Figure 4a,b depicts the residual error and approximate order of convergence, whereas Figure 5a–f depicts the local computational order of convergence of the neural network-based parallel root finding scheme.

The following values are chosen as initial guesses:

$$\overset{(0)}{x_1} = 20.1, \ \overset{(0)}{x_2} = 50.8, \ \overset{(0)}{x_3} = 125.5.$$

As shown in Table 10, the accuracy of the proposed numerical schemes improves when the outputs of the neural networks are used as initial guess values in Q-MM$^{\sigma_1}$, TAM^{C_*}, BSM^{C_3}, NAM^{C_3}, NDM^{C_3}, NAM^{C_8}, Q-MM$^{\sigma_2}$. Table 11 presents the overall convergence behavior of neural network outputs based on q-analogies.



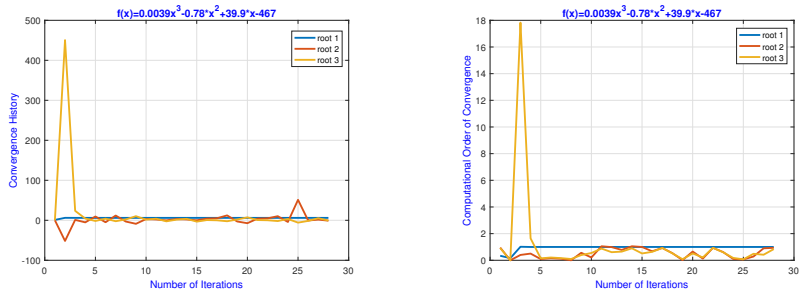
(**a**) (**b**)

(**c**) (**d**)

(**e**)

Figure 3. (**a**–**e**) Error histogram of neural network for engineering application 1 (**b**) Transition statistics curve of neural network for engineering application 1 (**c**) Mean square error curve of neural network for engineering application 1 (**d**) Regression curve of neural network for engineering application 1 (**e**) Fitness curve of neural network for engineering application 1. (**a**) The neural network's error histogram. (**b**) The neural network's transition statistics curve. (**c**) The neural network's mean square error curve. (**d**) The neural network's regression curve. (**e**) The neural network's fitness curve.

In Table 11, Max-Err$_{\Lambda_1^{[*]}}$ represents the maximum error; Max-it$_{\Lambda_1^{[*]}}$ represents the maximum number of iterations; Max-CPU$_{\Lambda_1^{[*]}}$ represents the maximum elapsed CPU time; Max-COC$_{\Lambda_1^{[*]}}$ represents the maximum order of convergence achieved; and $\rho_i^{[\sigma-1]}$ is the local order of convergence of q-analogies based neural network parallel numerical scheme on this application using $\Lambda_i^{[*]} = \left| \overset{[\sigma+1]}{x_i} - \overset{[\sigma]}{x_i} \right|$ as stopping criterion.



(**a**) (**b**)

Figure 4. (**a**,**b**) The convergence path of the approximated roots for engineering application 1 using a neural network, (**b**) the neural network-based computational order of convergence of the approximate roots for engineering application 1. (**a**) The neural network's convergence path. (**b**) The computational order of convergence.

4.2. Engineering Application 2: Blood Rheology Model

Blood is a non-Newtonian fluid modeled as a "Casson Fluid". According to the Casson fluid model, simple fluids such as water and blood will flow through a tube in such a way that the center core of the fluids will travel as a plug with little distortion and a velocity gradient will occur near the wall [52,54]. We used the following non-linear polynomial equation to describe the plug flow of Casson fluids:

$$G = 1 - \frac{16}{7}\sqrt{x} + \frac{4}{3}x - \frac{1}{21}x^4, \tag{54}$$

where G represents the reduction in flow rate. Using $G = 0.40$ in (54), we have:

$$f_4(x) = \frac{1}{441}x^8 - \frac{8}{63}x^5 - 0.05714285714x^4 + \frac{16}{9}x^2 - 3.624489796x + 0.36. \tag{55}$$

The exact roots of (55) are:

$$\begin{aligned}
\zeta_1 &= 0.1046986515, \zeta_2 = 3.822389235, \zeta_3 = 1.553919850 + 0.9404149899i,\\
\zeta_4 &= -1.238769105 + 3.408523568i, \zeta_5 = -2.278694688 + 1.987476450i\\
\zeta_6 &= -2.278694688 - 1.987476450i, \zeta_7 = -1.238769105 - 3.408523568,\\
\zeta_8 &= 1.553919850 - 0.9404149899i.
\end{aligned}$$

To analyze convergence, we use the randomly generated starting guesses $X_2^{[0]} = \left[X_{21}^{[0]}, X_{22}^{[0]}, X_{23}^{[0]}, \ldots\right]$ shown in Appendix B, Table A2. In Tables 12–22, we examine the numerical results of the proposed parallel numerical scheme for this engineering applications at different values of q. In Table 12, the error and estimated solution are given up to two decimal places whereas the approximated root is computed up to 64 digit floating point arithmetic using $\Lambda_i^{[*]} = \left|\overset{[\sigma+1]}{x_i} - \overset{[\sigma]}{x_i}\right|$ as stopping criterion. Tables 13–15 report on the maximum error, number of iterations, and elapsed CPU time for the Q-MM$^{\sigma_1}$ method using different values of q. On the other hand, Tables 16–19 displays the numerical results of Q-MM$^{\sigma 2}$ in terms of approximated value, residual error, maximum error, number of iterations, and elapsed CPU time for finding all roots of Equation (55) using different values of q. The rate of convergence of Q-MM$^{\sigma_1}$-Q-MM$^{\sigma_2}$ increases significantly as the q values increase from 0.2 to 1.1 on random initial values, demonstrating global convergence behavior.

The implementation of the neural network and its results using Algorithm 1 are shown in Figure 6a–e. The neural network's outputs are displayed in Figure 6 as follows: (a) the error histogram; (b) the transition statistics curve;(c) the mean square error; (d) the

regression curve; and (e) the fitness curve. Neural network simulations of this engineering application provide reliable and consistent results, as shown in Figure 6a–e and Table 20. Figure 7a,b represents the residual error and approximate order of convergence whereas the local computational order of convergence of the neural network based on parallel numerical schemes is depicted in Figure 8a–f.

Table 12. Error analysis and roots approximation using the Q-MM$^{\sigma_1}$ method with $X_2^{[0]}$ on engineering application 2.

$X_2^{[0]} = \left[X_{21}^{[0]}, X_{22}^{[0]}, X_{23}^{[0]}, \ldots\right]$									
q	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	1.1
$x_1^{[\sigma]}$	$0.1 - 1.0 \times 10^{-64}i$	$0.1 - 1.0 \times 10^{-64}i$	$0.1 - 1.0 \times 10^{-64}i$	$0.1 - 1.0 \times 10^{-64}i$	$0.1 - 1.0 \times 10^{-64}i$	$0.1 - 1.0 \times 10^{-64}i$	$0.1 - 1.0 \times 10^{-64}i$	$0.1 - 1.0 \times 10^{-64}i$	$0.1 - 1.0 \times 10^{-64}i$
$x_2^{[\sigma]}$	$3.8 - 0.5 \times 10^{-63}i$	$3.8 - 0.5 \times 10^{-63}i$	$3.8 - 0.5 \times 10^{-63}i$	$3.8 - 0.5 \times 10^{-63}i$	$3.8 - 0.5 \times 10^{-63}i$	$3.8 - 0.5 \times 10^{-63}i$	$3.8 - 0.5 \times 10^{-63}i$	$3.8 - 0.5 \times 10^{-63}i$	$3.8 - 0.5 \times 10^{-63}i$
$x_3^{[\sigma]}$	$1.531 + 0.124i$	$1.531 + 0.124i$	$1.531 + 0.124i$	$1.531 + 0.124i$	$1.531 + 0.124i$	$1.531 + 0.124i$	$1.531 + 0.124i$	$1.531 + 0.124i$	$1.531 + 0.124i$
$x_4^{[\sigma]}$	$-1.2 + 3.41i$	$-1.2 + 3.41i$	$-1.2 + 3.41i$	$-1.2 + 3.41i$	$-1.2 + 3.41i$	$-1.2 + 3.41i$	$-1.2 + 3.41i$	$-1.2 + 3.41i$	$-1.2 + 3.41i$
$x_5^{[\sigma]}$	$-2.2 + 1.94i$	$-2.2 + 1.94i$	$-2.2 + 1.94i$	$-2.2 + 1.94i$	$-2.2 + 1.94i$	$-2.2 + 1.94i$	$-2.2 + 1.94i$	$-2.2 + 1.94i$	$-2.2 + 1.94i$
$x_6^{[\sigma]}$	−2.2-1.94i	$-2.2 - 1.94i$	$-2.2 - 1.94i$	$-2.2 - 1.94i$	$-2.2 - 1.94i$	$-2.2 - 1.94i$	$-2.2 - 1.94i$	$-2.2 - 1.94i$	$-2.2 - 1.94i$
$x_7^{[\sigma]}$	$-1.2 - 3.40i$	$-1.2 - 3.40i$	$-1.2 - 3.40i$	$-1.2 - 3.40i$	$-1.2 - 3.40i$	$-1.2 - 3.40i$	$-1.2 - 3.40i$	$-1.2 - 3.40i$	$-1.2 - 3.40i$
$x_8^{[\sigma]}$	$1.5 - 0.94i$	$1.5 - 0.94i$	$1.5 - 0.94i$	$1.5 - 0.94i$	$1.5 - 0.94i$	$1.5 - 0.94i$	$1.5 - 0.94i$	$1.5 - 0.94i$	$1.5 - 0.94i$
$\Lambda_1^{[*]}$	0.11	4.3×10^{-30}	8.8×10^{-25}	4.3×10^{-30}	1.3×10^{-29}	4.7×10^{-15}	6.4×10^{-27}	2.8×10^{-47}	1.8×10^{-49}
$\Lambda_2^{[*]}$	8.16	1.3×10^{-20}	2.0×10^{-26}	1.3×10^{-20}	2.6×10^{-30}	2.0×10^{-32}	8.3×10^{-29}	4.7×10^{-46}	0.7×10^{-47}
$\Lambda_3^{[*]}$	1.66	5.3×10^{-11}	1.0×10^{-37}	5.3×10^{-11}	1.3×10^{-29}	4.7×10^{-15}	1.7×10^{-33}	6.3×10^{-50}	9.0×10^{-50}
$\Lambda_4^{[*]}$	0.08	8.1×10^{-31}	2.6×10^{-37}	8.1×10^{-29}	2.6×10^{-30}	2.0×10^{-32}	8.1×10^{-33}	0.1×10^{-51}	0.1×10^{-55}
$\Lambda_5^{[*]}$	0.11	4.3×10^{-30}	8.8×10^{-25}	4.3×10^{-30}	2.9×10^{-20}	3.3×10^{-18}	6.4×10^{-27}	2.8×10^{-47}	1.8×10^{-49}
$\Lambda_6^{[*]}$	8.16	1.3×10^{-20}	2.0×10^{-26}	1.3×10^{-20}	1.3×10^{-17}	3.8×10^{-33}	8.3×10^{-29}	4.7×10^{-46}	0.7×10^{-27}
$\Lambda_7^{[*]}$	1.66	5.3×10^{-11}	1.0×10^{-37}	5.3×10^{-11}	1.3×10^{-29}	4.7×10^{-15}	1.7×10^{-33}	6.3×10^{-50}	9.0×10^{-50}
$\Lambda_8^{[*]}$	0.08	8.1×10^{-31}	2.6×10^{-37}	8.1×10^{-29}	2.6×10^{-30}	2.0×10^{-32}	8.1×10^{-33}	0.1×10^{-51}	0.1×10^{-55}
E_{max}^{time}	8.84512	9.1241	9.2145	10.254	9.4151	9.1245	9.1245	7.1425	6.3251

Table 13. Max-Error for the Q-MM$^{\sigma_1}$ method using $X_1^{[0]}$ on engineering application 2.

[Q-MM$^{\sigma_1}$; q]	$e_1^{(\sigma)}$	$e_2^{(\sigma)}$	$e_3^{(\sigma)}$	$e_4^{(\sigma)}$	$e_5^{(\sigma)}$	$e_6^{(\sigma)}$	$e_7^{(\sigma)}$	$e_8^{(\sigma)}$
[Q-MM$^{\sigma_2}$; 1.1]	0.2×10^{-29}	0.2×10^{-65}	3.5×10^{-64}	88×10^{-30}	0.3×10^{-35}	1.0×10^{-35}	9.7×10^{-45}	5.7×10^{-41}
[Q-MM$^{\sigma_2}$; 1.0]	2.2×10^{-29}	1.1×10^{-65}	3.2×10^{-64}	6.1×10^{-30}	7.3×10^{-35}	3.5×10^{-35}	4.2×10^{-45}	3.1×10^{-41}
[Q-MM$^{\sigma_2}$; 0.9]	2.1×10^{-25}	2.7×10^{-24}	2.1×10^{-25}	2.7×10^{-24}	2.1×10^{-25}	2.7×10^{-24}	2.1×10^{-25}	2.7×10^{-20}
[Q-MM$^{\sigma_2}$; 0.8]	4.2×10^{-25}	3.1×10^{-20}	4.2×10^{-22}	3.1×10^{-31}	4.2×10^{-25}	3.1×10^{-21}	4.2×10^{-25}	3.1×10^{-18}
[Q-MM$^{\sigma_2}$; 0.7]	2.1×10^{-15}	2.7×10^{-14}	2.1×10^{-15}	2.7×10^{-14}	2.1×10^{-15}	2.7×10^{-14}	2.1×10^{-15}	2.7×10^{-14}
[Q-MM$^{\sigma_2}$; 0.6]	4.2×10^{-20}	8.1×10^{-10}	9.2×10^{-15}	3.1×10^{-11}	4.2×10^{-15}	3.1×10^{-11}	4.2×10^{-15}	3.1×10^{-10}
[Q-MM$^{\sigma_2}$; 0.5]	2.1×10^{-5}	6.7×10^{-4}	2.1×10^{-5}	2.1×10^{-4}	2.1×10^{-5}	2.7×10^{-4}	2.1×10^{-5}	2.7×10^{-4}
[Q-MM$^{\sigma_2}$; 0.4]	0.2×10^{-5}	1.1×10^{-3}	4.2×10^{-5}	3.1×10^{-3}	4.1×10^{-2}	3.0×10^{-3}	3.2×10^{-5}	3.1×10^{-4}
[Q-MM$^{\sigma_2}$; 0.3]	1.2×10^{-3}	0.1×10^{-3}	4.2×10^{-2}	3.1×10^{-3}	4.2×10^{-2}	3.1×10^{-3}	4.2×10^{-2}	3.1×10^{-3}

Table 14. Number of iterations for the Q-MM$^{\sigma_1}$ method using $X_2^{[0]}$.

[MM$^{\alpha_1}$; q]	It-$e_1^{[\sigma]}$	It-$e_2^{[\sigma]}$	It-$e_3^{[\sigma]}$	It-$e_4^{[\sigma]}$	It-$e_5^{[\sigma]}$	It-$e_6^{[\sigma]}$	It-$e_7^{[\sigma]}$	It-$e_8^{[\sigma]}$
[Q-MM$^{\sigma_2}$; 1.1]	63	63	63	63	63	63	63	63
[Q-MM$^{\sigma_2}$; 0.9]	63	63	63	63	63	63	63	63
[Q-MM$^{\sigma_2}$; 0.8]	77	77	77	77	77	77	77	77
[Q-MM$^{\sigma_2}$; 0.7]	79	79	79	79	79	79	79	79
[Q-MM$^{\sigma_2}$; 0.6]	85	85	85	85	85	85	85	85
[Q-MM$^{\sigma_2}$; 0.5]	87	87	87	87	87	87	87	87
[Q-MM$^{\sigma_2}$; 0.4]	91	91	91	91	91	91	91	91
[Q-MM$^{\sigma_2}$; 0.3]	97	97	97	97	97	97	97	97
[Q-MM$^{\sigma_2}$; 0.2]	99	99	99	99	99	99	99	99

Table 15. CPU-time for the Q-MM$^{\sigma_1}$ method using $X_2^{[0]}$.

[Q-MM$^{\sigma_1}$; q]	CT-$e_1^{(3)}$	CT-$e_2^{(3)}$	CT-$e_3^{(3)}$	CT-$e_4^{(3)}$	CT-$e_5^{(3)}$	CT-$e_6^{(3)}$	CT-$e_7^{(3)}$	CT-$e_8^{(3)}$
[Q-MM$^{\sigma_2}$; 1.1]	6.3325	6.1424	7.1241	7.2148	6.2145	6.3251	6.2145	6.2525
[Q-MM$^{\sigma_2}$; 0.9]	6.3214	6.2145	6.9856	6.3298	6.8547	63214	7.2145	6.3214
[Q-MM$^{\sigma_2}$; 0.8]	7.3652	7.1456	7.32145	7.3214	7.1452	7.3652	7.1245	7.1426
[Q-MM$^{\sigma_2}$; 0.7]	7.2145	7.3652	7.1245	7.3652	7.1254	7.6521	7.3256	7.1254
[Q-MM$^{\sigma_2}$; 0.6]	7.36541	7.98654	7.6935	7.96523	7.3214	7.69321	7.8546	8.2156
[Q-MM$^{\sigma_2}$; 0.5]	8.36521	8.96542	8.32154	8.36542	8.36954	8.21453	8.6352	8.9658
[Q-MM$^{\sigma_2}$; 0.4]	8.65418	8.6954	8.6598	8.74566	8.3654	9.6532	8.3214	9.3654
[Q-MM$^{\sigma_2}$; 0.3]	9.45178	9.1024	10.2541	8.7454	9.6542	9.8745	9.4157	9.6542
[Q-MM$^{\sigma_2}$; 0.2]	9.1241	9.84512	9.41524	8.5412	9.35412	9.8754	9.4157	9.4571

Table 16. Error analysis and roots approximation using the Q-MM$^{\sigma_2}$ method with $X_2^{[0]}$ on engineering application 2.

$X_2^{[0]} = \left[X_{21}^{[0]}, X_{22}^{[0]}, X_{23}^{[0]}, \ldots\right]$									
q	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	1.1
$x_1^{[\sigma]}$	$0.1 - 1.0 \times 10^{-64}i$	$0.1 - 1.0 \times 10^{-64}i$	$0.1 - 1.0 \times 10^{-64}i$	$0.1 - 1.0 \times 10^{-64}i$	$0.1 - 1.0 \times 10^{-64}i$	$0.1 - 1.0 \times 10^{-64}i$	$0.1 - 1.0 \times 10^{-64}i$	$0.1 - 1.0 \times 10^{-64}i$	$0.1 - 1.0 \times 10^{-64}i$
$x_2^{[\sigma]}$	$3.8 - 0.5 \times 10^{-63}i$	$3.8 - 0.5 \times 10^{-63}i$	$3.8 - 0.5 \times 10^{-63}i$	$3.8 - 0.5 \times 10^{-63}i$	$3.8 - 0.5 \times 10^{-63}i$	$3.8 - 0.5 \times 10^{-63}i$	$3.8 - 0.5 \times 10^{-63}i$	$3.8 - 0.5 \times 10^{-63}i$	$3.8 - 0.5 \times 10^{-63}i$
$x_3^{[\sigma]}$	$1.531 + 0.124i$	$1.531 + 0.124i$	$1.531 + 0.124i$	$1.531 + 0.124i$	$1.531 + 0.124i$	$1.531 + 0.124i$	$1.531 + 0.124i$	$1.531 + 0.124i$	$1.531 + 0.124i$
$x_4^{[\sigma]}$	$-1.2 + 3.41i$	$-1.2 + 3.41i$	$-1.2 + 3.41i$	$-1.2 + 3.41i$	$-1.2 + 3.41i$	$-1.2 + 3.41i$	$-1.2 + 3.41i$	$-1.2 + 3.41i$	$-1.2 + 3.41i$
$x_5^{[\sigma]}$	$-2.2 + 1.94i$	$-2.2 + 1.94i$	$-2.2 + 1.94i$	$-2.2 + 1.94i$	$-2.2 + 1.94i$	$-2.2 + 1.94i$	$-2.2 + 1.94i$	$-2.2 + 1.94i$	$-2.2 + 1.94i$
$x_6^{[\sigma]}$	$-2.2 - 1.94i$	$-2.2 - 1.94i$	$-2.2 - 1.94i$	$-2.2 - 1.94i$	$-2.2 - 1.94i$	$-2.2 - 1.94i$	$-2.2 - 1.94i$	$-2.2 - 1.94i$	$-2.2 - 1.94i$
$x_7^{[\sigma]}$	$-1.2 - 3.40i$	$-1.2 - 3.40i$	$-1.2 - 3.40i$	$-1.2 - 3.40i$	$-1.2 - 3.40i$	$-1.2 - 3.40i$	$-1.2 - 3.40i$	$-1.2 - 3.40i$	$-1.2 - 3.40i$
$x_8^{[\sigma]}$	$1.5 - 0.94i$	$1.5 - 0.94i$	$1.5 - 0.94i$	$1.5 - 0.94i$	$1.5 - 0.94i$	$1.5 - 0.94i$	$1.5 - 0.94i$	$1.5 - 0.94i$	$1.5 - 0.94i$
$\Lambda_1^{[*]}$	0.11	4.3×10^{-30}	8.8×10^{-25}	4.3×10^{-30}	1.3×10^{-29}	4.7×10^{-15}	6.4×10^{-27}	2.8×10^{-47}	1.8×10^{-49}
$\Lambda_2^{[*]}$	8.16	1.3×10^{-20}	2.0×10^{-26}	1.3×10^{-20}	2.6×10^{-30}	2.0×10^{-32}	8.3×10^{-29}	4.7×10^{-46}	0.7×10^{-47}
$\Lambda_3^{[*]}$	1.66	5.3×10^{-11}	1.0×10^{-37}	5.3×10^{-11}	1.3×10^{-29}	4.7×10^{-15}	1.7×10^{-33}	6.3×10^{-50}	9.0×10^{-50}
$\Lambda_4^{[*]}$	0.08	8.1×10^{-31}	2.6×10^{-37}	8.1×10^{-29}	2.6×10^{-30}	2.0×10^{-32}	8.1×10^{-33}	0.1×10^{-51}	0.1×10^{-55}
$\Lambda_5^{[*]}$	0.11	4.3×10^{-30}	8.8×10^{-25}	4.3×10^{-30}	2.9×10^{-20}	3.3×10^{-18}	6.4×10^{-27}	2.8×10^{-47}	1.8×10^{-49}
$\Lambda_6^{[*]}$	8.16	1.3×10^{-20}	2.0×10^{-26}	1.3×10^{-20}	1.3×10^{-17}	3.8×10^{-33}	8.3×10^{-29}	4.7×10^{-46}	0.7×10^{-47}
$\Lambda_7^{[*]}$	1.66	5.3×10^{-11}	1.0×10^{-37}	5.3×10^{-11}	1.3×10^{-29}	4.7×10^{-15}	1.7×10^{-33}	6.3×10^{-50}	9.0×10^{-50}
$\Lambda_8^{[*]}$	0.08	8.1×10^{-31}	2.6×10^{-37}	8.1×10^{-29}	2.6×10^{-30}	2.0×10^{-32}	8.1×10^{-33}	0.1×10^{-51}	0.1×10^{-55}
E_{max}^{time}	8.84512	9.1241	9.2145	10.254	9.4151	9.1245	9.1245	7.1425	6.3251

Table 17. Max-Error for the Q-MM$^{\sigma_2}$ using $X_2^{[0]}$ on engineering application 2.

[Q-MM$^{\sigma_2}$; q]	$e_1^{(3)}$	$e_2^{(3)}$	$e_3^{(3)}$	$e_4^{(3)}$	$e_5^{(3)}$	$e_6^{(3)}$	$e_7^{(3)}$	$e_8^{(3)}$
[Q-MM$^{\sigma_2}$; 1.1]	1.4×10^{-19}	1.7×10^{-15}	1.3×10^{-14}	2.1×10^{-15}	2.0×10^{-13}	4.3×10^{-14}	2.1×10^{-15}	2.7×10^{-14}
[Q-MM$^{\sigma_2}$; 1.0]	2.2×10^{-21}	1.1×10^{-18}	3.2×10^{-21}	6.1×10^{-30}	7.3×10^{-22}	3.5×10^{-25}	4.2×10^{-25}	3.1×10^{-31}
[Q-MM$^{\sigma_2}$; 0.9]	2.1×10^{-13}	2.7×10^{-14}	2.1×10^{-15}	2.7×10^{-14}	2.1×10^{-15}	2.7×10^{-14}	2.1×10^{-15}	2.7×10^{-14}
[Q-MM$^{\sigma_2}$; 0.8]	4.2×10^{-25}	3.1×10^{-31}	4.2×10^{-25}	3.1×10^{-31}	4.2×10^{-25}	3.1×10^{-31}	4.2×10^{-25}	3.1×10^{-31}
[Q-MM$^{\sigma_2}$; 0.7]	2.1×10^{-15}	2.7×10^{-14}	2.1×10^{-15}	2.7×10^{-14}	2.1×10^{-15}	2.7×10^{-14}	2.1×10^{-15}	2.7×10^{-14}
[Q-MM$^{\sigma_2}$; 0.6]	4.2×10^{-25}	3.1×10^{-31}	4.2×10^{-25}	3.1×10^{-31}	4.2×10^{-25}	3.1×10^{-31}	4.2×10^{-25}	3.1×10^{-31}
[Q-MM$^{\sigma_2}$; 0.5]	2.1×10^{-15}	2.7×10^{-14}	2.1×10^{-15}	2.7×10^{-14}	2.1×10^{-15}	2.7×10^{-14}	2.1×10^{-15}	2.7×10^{-14}
[Q-MM$^{\sigma_2}$; 0.4]	4.2×10^{-25}	3.1×10^{-31}	4.2×10^{-25}	3.1×10^{-31}	4.2×10^{-25}	3.1×10^{-31}	4.2×10^{-25}	3.1×10^{-31}
[Q-MM$^{\sigma_2}$; 0.3]	4.2×10^{-25}	3.1×10^{-31}	4.2×10^{-25}	3.1×10^{-31}	4.2×10^{-25}	3.1×10^{-31}	4.2×10^{-25}	3.1×10^{-31}

Table 18. Number of iterations for the Q-MM$^{\sigma_2}$ method using $X_2^{[0]}$.

[Q-MM$^{\sigma_2}$; q]	It-$e_1^{[\sigma]}$	It-$e_2^{[\sigma]}$	It-$e_3^{[\sigma]}$	It-$e_4^{[\sigma]}$	It-$e_5^{[\sigma]}$	It-$e_6^{[\sigma]}$	It-$e_7^{[\sigma]}$	It-$e_8^{[\sigma]}$
[Q-MM$^{\sigma_2}$; 1.1]	24	24	24	24	24	24	24	24
[Q-MM$^{\sigma_2}$; 1.0]	23	23	23	23	23	23	23	23
[Q-MM$^{\sigma_2}$; 0.9]	30	30	30	30	30	30	30	30
[Q-MM$^{\sigma_2}$; 0.8]	31	31	31	31	31	31	31	31
[Q-MM$^{\sigma_2}$; 0.7]	36	36	36	36	36	36	36	36
[Q-MM$^{\sigma_2}$; 0.6]	39	39	39	39	39	39	39	39
[Q-MM$^{\sigma_2}$; 0.5]	46	46	46	46	46	46	46	46
[Q-MM$^{\sigma_2}$; 0.4]	47	47	47	47	47	47	47	47
[Q-MM$^{\sigma_2}$; 0.3]	50	50	50	50	50	50	50	50

Table 19. CPU-time for the Q-MM$^{\sigma_2}$ method using $X_2^{[0]}$.

[Q-MM$^{\sigma_2}$; q]	CT-$e_1^{(3)}$	CT-$e_2^{(3)}$	CT-$e_3^{(3)}$	CT-$e_4^{(3)}$	CT-$e_5^{(3)}$	CT-$e_6^{(3)}$	CT-$e_7^{(3)}$	CT-$e_8^{(3)}$
[Q-MM$^{\sigma_2}$; 1.1]	4.3325	4.1424	4.1241	5.2148	5.2145	5.3251	4.2145	6.2525
[Q-MM$^{\sigma_2}$; 1.0]	6.3214	5.2145	5.9856	4.3298	4.8547	4.3214	5.2145	5.3214
[Q-MM$^{\sigma_2}$; 0.9]	5.3652	5.1456	5.32145	5.3214	5.1452	7.3652	5.1245	5.1426
[Q-MM$^{\sigma_2}$; 0.8]	6.2145	6.3652	6.1245	6.3652	6.1254	6.6521	6.3256	7.1254
[Q-MM$^{\sigma_2}$; 0.7]	6.36541	6.98654	7.6935	7.96523	7.3214	7.69321	7.8546	8.2156
[Q-MM$^{\sigma_2}$; 0.6]	8.36521	7.96542	8.32154	8.36542	7.36954	8.21453	8.6352	8.9658
[Q-MM$^{\sigma_2}$; 0.5]	8.65418	8.6954	7.6598	7.74566	7.3654	9.6532	8.3214	9.3654
[Q-MM$^{\sigma_2}$; 0.4]	9.45178	9.1024	8.2541	8.7454	8.6542	8.8745	8.4157	9.6542
[Q-MM$^{\sigma_2}$; 0.3]	8.1241	8.84512	8.41524	8.5412	8.35412	8.8754	8.4157	9.4571

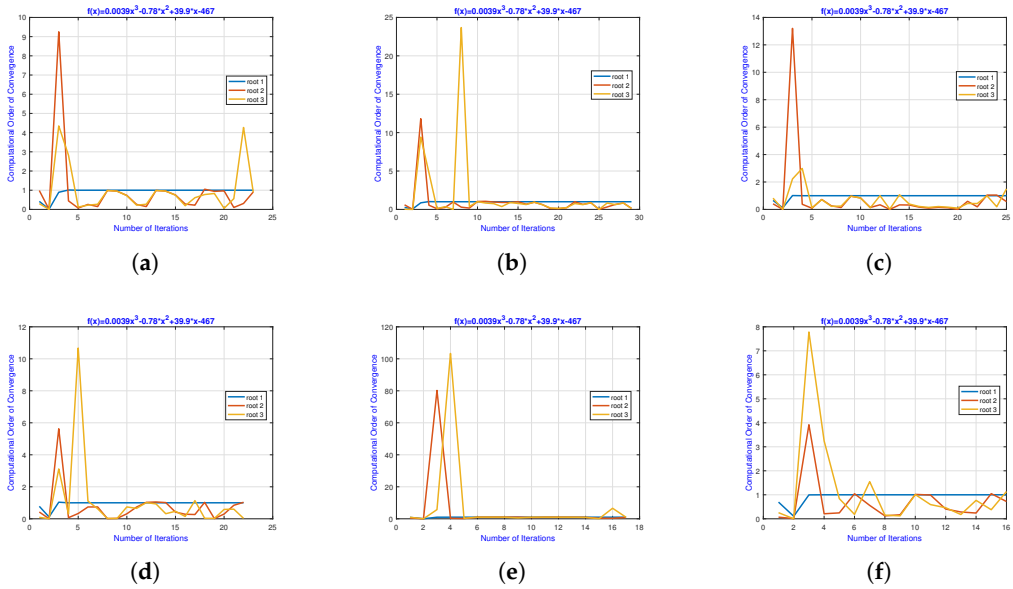


Figure 5. (**a**–**f**) Using the neural network's output as input, the local computational order of convergence of Q-MM$^{\sigma_1}$ required 24 iterations to converge (**b**) using the neural network's output as input, the local computational order of convergence of BSM^{C_3} required 30 iterations (**c**) using the neural network's output as input, the local computational order of convergence of NAM^{C_3} required 25 iterations (**d**) using the neural network's output as input, the local computational order of convergence of NDM^{C_3} required 22 iterations (**e**) using the neural network's output as input, the local computational order of convergence of NAM^{C_8} required 19 iterations (**e**) using the neural network's output as input, the local computational order of convergence of Q-MM$^{\sigma_2}$ required 16 iterations to converge. (**a**) Local computing order of Q-MM$^{\sigma_1}$ convergence. (**b**) Local computing order of BSM^{C_3} convergence. (**c**) Local computing order of NAM^{C_3} convergence. (**d**) Local computing order of NDM^{C_3} convergence. (**e**) Local computing order of NAM^{C_8} convergence. (**f**) Local computing order of Q-MM$^{\sigma_2}$ convergence, for solving biomedical engineering application 1.

Table 20. Error outcomes using neural networks on application 2.

Method	$e_1^{(3)}$	$e_2^{(3)}$	$e_3^{(3)}$	$e_4^{(3)}$	$e_5^{(3)}$	$e_6^{(3)}$	$e_7^{(3)}$	$e_8^{(3)}$	$\rho_i^{[\sigma-1]}$
[Q-MM$^{\sigma_1}$; 1.0]	1.4×10^{-19}	1.7×10^{-15}	1.3×10^{-14}	2.1×10^{-15}	2.0×10^{-13}	4.3×10^{-14}	2.1×10^{-15}	2.7×10^{-14}	3.014
TAM^{C_*}	8.7×10^{-18}	0.6×10^{-21}	6.1×10^{-25}	3.3×10^{-11}	0.1×10^{-9}	9.7×10^{-24}	5.1×10^{-25}	4.7×10^{-19}	2.742
BSM^{C_3}	2.1×10^{-15}	2.7×10^{-14}	2.1×10^{-15}	2.7×10^{-14}	2.1×10^{-15}	2.7×10^{-14}	2.1×10^{-15}	2.7×10^{-14}	3.142
NAM^{C_3}	4.2×10^{-25}	3.1×10^{-31}	4.2×10^{-25}	3.1×10^{-31}	4.2×10^{-25}	3.1×10^{-31}	4.2×10^{-25}	3.1×10^{-31}	2.941
NDM^{C_3}	2.1×10^{-15}	2.7×10^{-14}	2.1×10^{-15}	2.7×10^{-14}	2.1×10^{-15}	2.7×10^{-14}	2.1×10^{-15}	2.7×10^{-14}	3.124
NAM^{C_8}	4.2×10^{-25}	3.1×10^{-31}	4.2×10^{-25}	3.1×10^{-31}	4.2×10^{-25}	3.1×10^{-31}	4.2×10^{-25}	3.1×10^{-31}	7.984
[Q-MM$^{\sigma_2}$; 1.0]	2.1×10^{-15}	2.7×10^{-14}	2.1×10^{-15}	2.7×10^{-14}	2.1×10^{-15}	2.7×10^{-14}	2.1×10^{-15}	2.7×10^{-14}	8.014

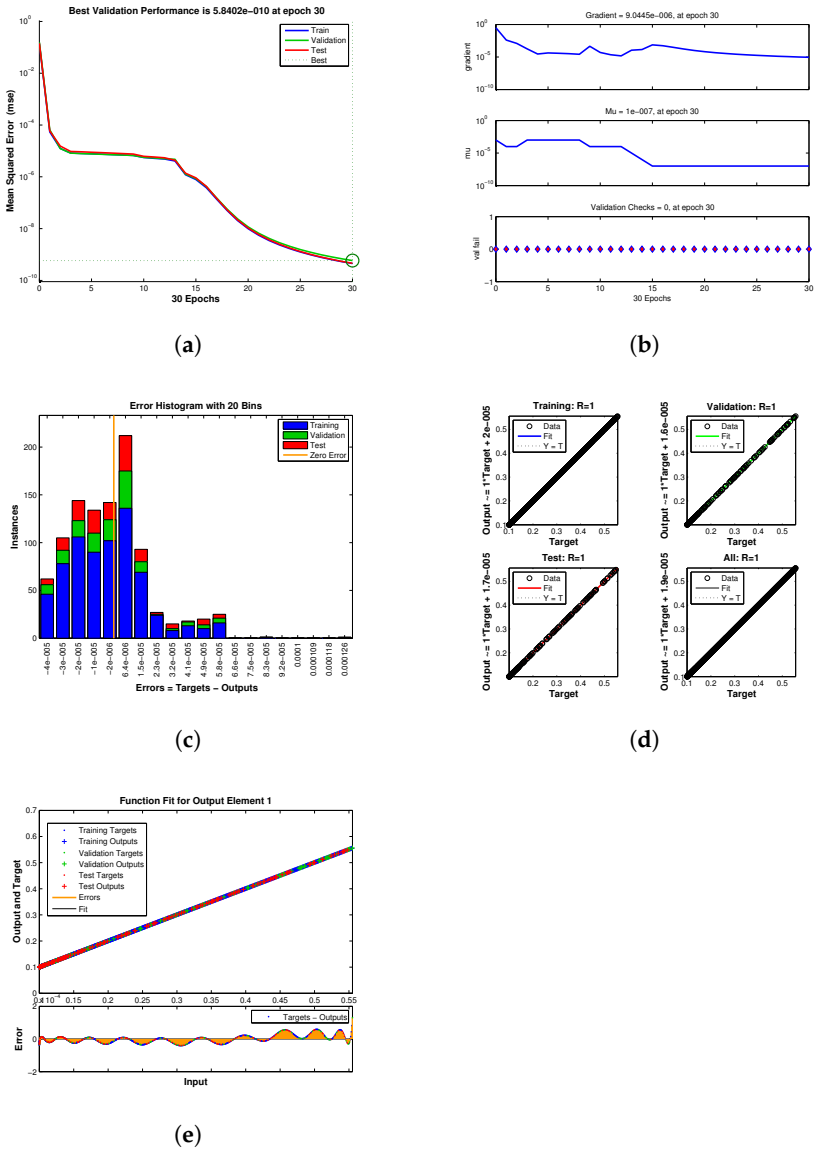
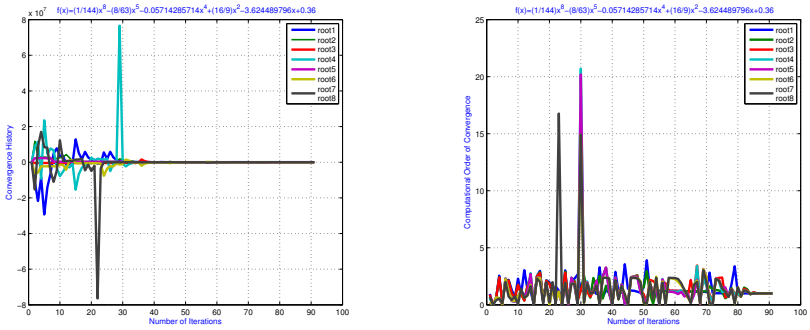


Figure 6. (**a**–**e**) Error histogram of neural network for engineering application 2 (**b**) Transition statistics curve of neural network for engineering application 2 (**c**) Mean square error curve of neural network for engineering application 2 (**d**) Regression curve of neural network for engineering application 2 (**e**) Fitness curve of neural network for engineering application 2. (**a**) The neural network's error histogram. (**b**) The neural network's transition statistics curve. (**c**) The neural network's mean square error curve. (**d**) The neural network's regression curve. (**e**) The neural network's fitness curve.

The following values are chosen as initial guesses:

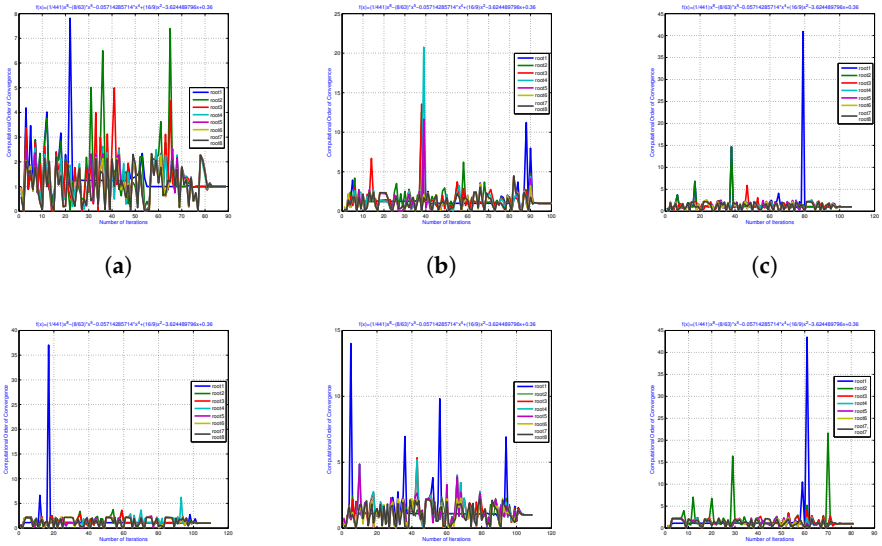
$$\begin{aligned} \overset{(0)}{x_1} &= 0.1, \ \overset{(0)}{x_2} = 3.8, \ \overset{(0)}{x_3} = 1.5 + 0.9i, \ \overset{(0)}{x_4} = 1.2 + 3.4i. \\ \overset{(0)}{x_5} &= -2.2 + 1.9i, \ \overset{(0)}{x_6} = -2.2 - 1.9i \ \overset{(0)}{x_7} = -1.2 - 3.4i, \ \overset{(0)}{x_8} = 1.5 + 0.9i. \end{aligned}$$

As shown in Table 21, the accuracy of the proposed numerical schemes improves when the outputs of the neural networks (see Appendix C) are used as initial guess values in Q-MM$^{\sigma_1}$, TAM^{C_*}, BSM^{C_3}, NAM^{C_3}, NDM^{C_3}, NAM^{C_8}, Q-MM$^{\sigma_2}$. Table 22 presents the overall convergence behavior of neural network outputs based on q-analogies.



(**a**) (**b**)

Figure 7. (**a**,**b**) The convergence path of the approximated roots for engineering application 2 using a neural network (**b**) the neural network-based computational order of convergence of the approximate roots for engineering application 2. (**a**) The neural network's convergence path. (**b**) The computational order of convergence.



(**a**) (**b**) (**c**)

(**d**) (**e**) (**f**)

Figure 8. (**a**–**f**) Using the neural network's output as input, the local computational order of convergence of Q-MM$^{\sigma_1}$ required 80 iterations to converge (**b**) Using the neural network's output as input, the local computational order of convergence of BSM^{C_3} required 90 iterations (**c**) using the neural network's output as input, the local computational order of convergence of NAM^{C_3} required 119 iterations (**d**) using the neural network's output as input, the local computational order of convergence of NDM^{C_3} required 19 iterations (**e**) Using the neural network's output as input, the local computational order of convergence of NAM^{C_8} required 115 iterations (**e**) using the neural network's output as input, the local computational order of convergence of Q-MM$^{\sigma_2}$ required 80 iterations to converge. (**a**) Local computing order of Q-MM$^{\sigma_1}$ convergence. (**b**) Local computing order of BSM^{C_3} convergence. (**c**) Local computing order of NAM^{C_3} convergence. (**d**) Local computing order of NDM^{C_3} convergence. (**e**) Local computing order of NAM^{C_8} convergence. (**f**) Local computing order of Q-MM$^{\sigma_2}$ convergence.

Table 21. Improvement in convergence rate using neural network outcomes as input.

Method	$e_1^{(3)}$	$e_2^{(3)}$	$e_3^{(3)}$	$e_4^{(3)}$	$e_5^{(3)}$	$e_6^{(3)}$	$e_7^{(3)}$	$e_8^{(3)}$	$\rho_i^{[\sigma-1]}$
[Q-MM$^{\sigma_1}$; 1.0]	0.4×10^{-39}	1.7×10^{-35}	1.3×10^{-34}	2.1×10^{-45}	2.0×10^{-43}	4.3×10^{-34}	2.1×10^{-35}	2.7×10^{-34}	3.525
TAM^{C_*}	9.1×10^{-26}	0.8×10^{-31}	2.0×10^{-55}	6.0×10^{-19}	1.1×10^{-21}	8.5×10^{-11}	9.1×10^{-25}	4.4×10^{-24}	2.924
BSM^{C_3}	0.1×10^{-35}	2.7×10^{-34}	2.0×10^{-55}	2.7×10^{-14}	2.1×10^{-25}	2.7×10^{-14}	2.1×10^{-15}	2.7×10^{-14}	3.124
NAM^{C_3}	4.2×10^{-65}	3.1×10^{-31}	5.2×10^{-25}	3.1×10^{-31}	4.3×10^{-35}	3.1×10^{-31}	0.2×10^{-25}	3.2×10^{-33}	3.145
NDM^{C_3}	0.1×10^{-65}	2.7×10^{-14}	2.1×10^{-25}	3.7×10^{-24}	2.1×10^{-35}	2.7×10^{-34}	2.1×10^{-35}	2.7×10^{-34}	2.965
NAM^{C_8}	3.2×10^{-75}	3.1×10^{-61}	4.2×10^{-55}	3.1×10^{-11}	4.2×10^{-45}	3.1×10^{-61}	4.2×10^{-55}	3.1×10^{-51}	7.891
[Q-MM$^{\sigma_2}$; 1.0]	2.1×10^{-75}	2.7×10^{-94}	2.1×10^{-85}	2.7×10^{-80}	2.1×10^{-75}	2.7×10^{-74}	0.0	2.7×10^{-71}	8.012

Table 22. Overall results of q-analogies-based neural network outcomes for accurate initial guesses.

Method	Max-Err$_{\Lambda_1^{[*]}}$	Max-it$_{\Lambda_1^{[*]}}$	Max-CPU$_{\Lambda_1^{[*]}}$	Max-COC$_{\Lambda_1^{[*]}}$	$\rho_i^{[\sigma-1]}$
[Q-MM$^{\sigma_1}$; 1.0]	1.4×10^{-19}	1.3×10^{-14}	2.1×10^{-15}	3.314246	3.5412
TAM^{C_*}	6.2×10^{-25}	4.7×10^{-16}	9.1×10^{-26}	3.905654	3.3229
BSM^{C_3}	2.2×10^{-21}	3.2×10^{-21}	6.1×10^{-30}	2.981454	3.3412
NAM^{C_3}	2.1×10^{-15}	2.1×10^{-15}	2.7×10^{-14}	3.001245	3.0125
NDM^{C_3}	4.2×10^{-25}	4.2×10^{-25}	3.1×10^{-31}	3.012445	3.4125
NAM^{C_8}	2.1×10^{-15}	2.1×10^{-15}	2.7×10^{-14}	7.954154	8.0145
[Q-MM$^{\sigma_2}$; 1.0]	4.2×10^{-25}	4.2×10^{-25}	3.1×10^{-31}	8.012416	8.3214

In Table 22, Max-Err$_{\Lambda_1^{[*]}}$ represents the maximum error; Max-it$_{\Lambda_1^{[*]}}$ represents the maximum number of iterations; Max-CPU$_{\Lambda_1^{[*]}}$ represents the maximum CPU time consumed; Max-COC$_{\Lambda_1^{[*]}}$ represents the maximum order of convergence achieved; and $\rho_i^{[\sigma-1]}$ is the local order of convergence of q-analogies based neural network parallel numerical scheme for solving this engineering application using $\Lambda_i^{[*]} = \left| \overset{[\sigma+1]}{x_i} - \overset{[\sigma]}{x_i} \right|$ as stopping criterion.

5. Conclusions

In this paper, we introduced two new families of q-type parallel numerical methods for finding all distinct roots of nonlinear Equation (1). Furthermore, a new hybrid numerical scheme combining neural network techniques and q-analogies are thoroughly examined. In order to analyze the global convergence behavior of the proposed numerical methods, random starting values for the iterations are used. The numerical outcomes of the Q-MM$^{\sigma_1}$-Q-MM$^{\sigma_2}$ on random initial approximations clearly demonstrate that the newly created methods are more efficient than the existing methods, as shown in Tables 1–8 and 12–19 and Figures 5a–f and 8a–f. Analyzing the overall convergence behaviour in Tables 11 and 22 reveals that the newly created techniques Q-MM$^{\sigma_1}$-Q-MM$^{\sigma_2}$ are more stable and consistent than the existing methods BSM^{C_3}, TAM^{C_*}, NAM^{C_3}, NDM^{C_3}, and NAM^{C_8}. The results of our experiments illustrate that convergence is significantly improved when the neural network outcomes are utilized as input for the q-analogies of our parallel schemes, as shown in Tables 1–22 and Figures 2–7. The numerical experiments on two biomedical engineering applications reveal that the new Q-MM$^{\sigma_1}$-Q-MM$^{\sigma_2}$ methods can outperform the BSM^{C_3}, TAM^{C_*}, NAM^{C_3}, NDM^{C_3}, and NAM^{C_8} methods in terms of errors using random initial guesses, CPU time, residual error, computational and local-computational order of convergence, and number of iterations. The next step of this research will involve the development and analysis of higher-order q-analogues of inverse numerical schemes based on neural network [55–57].

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.; supervi-

sion, 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.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Data are contained within the article.

Acknowledgments: The work is supported by Provincia autonoma di Bolzano/Alto Adige-Ripartizione Innovazione, Ricerca, Università e Musei (contract nr. 19/34). The work of Bruno Carpentieri is also supported by the Free University of Bozen-Bolzano (IN200Z SmartPrint). Bruno Carpentieri is a member of the *Gruppo Nazionale per il Calcolo Scientifico* (GNCS) of the Istituto Nazionale di Alta Matematica (INdAM) and this work was partially supported by INdAM-GNCS under Progetti di Ricerca 2022.

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

Appendix A

Proof of Theorem 1. Let ζ be a simple root of f, and $x^{[\sigma]} = \zeta + \vartheta^{[\sigma]}$ and $z^{[\sigma]} = \zeta + \vartheta^{[\sigma+1]}$. By q-Taylor's series expansion of $\left(\partial_q^{[1]} f\right)(x^{[\sigma]})$ around $x^{[\sigma]} = \zeta$, and taking $f(\zeta) = 0$, we get

$$\begin{aligned} f(x^{[\sigma]}) &= \left(\partial_q^{[1]} f\right)(\zeta)\left(\left[\vartheta^{[\sigma]}\right]^1; q\right) + \left(\partial_q^{[2]} f\right)(\zeta)\left(\left[\vartheta^{[\sigma]}\right]^2; q\right) + \\ &\quad \left(\partial_q^{[3]} f\right)(\zeta)\left(\left[\vartheta^{[\sigma]}\right]^3; q\right) + \dots \end{aligned} \tag{A1}$$

$$f(x^{[\sigma]}) = \left(\left[\vartheta^{[\sigma]}\right]^1; q\right) + \frac{\left(\partial_q^{[2]} f\right)(\zeta)}{\left(\partial_q^{[1]} f\right)(\zeta)}\left(\left[\vartheta^{[\sigma]}\right]^2; q\right) + \frac{\left(\partial_q^{[3]} f\right)}{\left(\partial_q^{[1]} f\right)(\zeta)}(\zeta)\left(\left[\vartheta^{[\sigma]}\right]^3; q\right) + \dots \tag{A2}$$

and

$$\left(\partial_q^{[1]} f\right)(x^{[\sigma]}) = \left(\partial_q^{[1]} f\right)(\zeta)\left\{\begin{array}{c} 1 + 2\frac{\left(\partial_q^{[2]} f\right)(\zeta)}{[2!]\left(\partial_q^{[1]} f\right)(\zeta)}\left(\left[\vartheta^{[\sigma]}\right]; q\right) \\ +3\frac{\left(\partial_q^{[3]} f\right)(\zeta)}{[3!]\left(\partial_q f\right)(\zeta)}\left(\left[\vartheta^{[\sigma]}\right]^2; q\right) + \dots \end{array}\right\}. \tag{A3}$$

When (A1) is divided by (A3), we have:

$$\begin{aligned} \frac{f(x^{[\sigma]})}{\left(\partial_q^{[1]} f\right)(x^{[\sigma]})} &= \left(\vartheta^{[\sigma]}; q\right) - \left(\frac{\left(\partial_q^{[2]} f\right)(\zeta)}{[2!]\left(\partial_q f\right)(\zeta)}\right)\left(\left[\vartheta^{[\sigma]}\right]^2; q\right) - \\ &\quad \left(\begin{array}{c} 2\left(\frac{\left(\partial_q^{[3]} f\right)(\zeta)}{[3!]\left(\partial_q f\right)(\zeta)}\right) - \\ 2\left(\frac{\left(\partial_q^{[2]} f\right)(\zeta)}{[2!]\left(\partial_q f\right)(\zeta)}\right)^2 \end{array}\right)\left(\left[\vartheta^{[\sigma]}\right]^3; q\right) + \dots \end{aligned} \tag{A4}$$

$$\begin{aligned} 1+\alpha_2 f(x^{[\sigma]}) &= 1+\alpha_2\left(\vartheta^{[\sigma]};q\right)+\alpha_2\frac{\left(\eth_q^{[2]}f\right)(\zeta)}{\left(\eth_q^{[1]}f\right)(\zeta)}\left(\left[\vartheta^{[\sigma]}\right]^2;q\right) \\ &\quad +\alpha_2\frac{\left(\eth_q^{[3]}f\right)(\zeta)}{\left(\eth_q^{[1]}f\right)(\zeta)}\left(\left[\vartheta^{[\sigma]}\right]^3;q\right)+\ldots \end{aligned} \tag{A5}$$

$$\frac{\alpha_1 f(x^{[\sigma]})}{1+\alpha_2 f(x^{[\sigma]})}=\alpha_1\left(\vartheta^{[\sigma]};q\right)+\left(\alpha\frac{\left(\eth_q^{[2]}f\right)(\zeta)}{\left(\eth_q^{[1]}f\right)(\zeta)}-\alpha_1\alpha_2\right)\left(\left[\vartheta^{[\sigma]}\right]^2;q\right)+O\left(\left[\vartheta^{[\sigma]}\right]^3;q\right). \tag{A6}$$

By using this expression in the first-step of (13), we have:

$$1-\frac{\alpha_1 f(x^{[\sigma]})}{1+\alpha_2 f(x^{[\sigma]})}=1-\alpha_1\left(\vartheta^{[\sigma]};q\right)-\left(\begin{array}{c}\alpha\frac{\left(\eth_q^{[2]}f\right)(\zeta)}{\left(\eth_q^{[1]}f\right)(\zeta)}- \\ \alpha_1\alpha_2\end{array}\right)\left(\left[\vartheta^{[\sigma]}\right]^2;q\right)+O\left(\left[\vartheta^{[\sigma]}\right]^3;q\right). \tag{A7}$$

We can take the inverse of (A7):

$$\left[1-\frac{\alpha_1 f(x^{[\sigma]})}{1+\alpha_2 f(x^{[\sigma]})}\right]^{-1}=1+\alpha_1\left(\vartheta^{[\sigma]};q\right)+\left(\begin{array}{c}\alpha_1\frac{\left(\eth_q^{[2]}f\right)(\zeta)}{\left(\eth_q^{[1]}f\right)(\zeta)} \\ -\alpha_1\alpha_2\end{array}\right)\left(\left[\vartheta^{[\sigma]}\right]^2;q\right)+\ldots \tag{A8}$$

Thus,

$$\begin{aligned} \frac{f(x^{[\sigma]})}{\left(\eth_q^{[1]}f\right)(x^{[\sigma]})}\left[\frac{1}{1-\frac{\alpha_1 f(x^{[\sigma]})}{1+\alpha_2 f(x^{[\sigma]})}}\right] &= \left(\vartheta^{[\sigma]};q\right)+\left(\alpha_1-\frac{\left(\eth_q^{[2]}f\right)(\zeta)}{\left(\eth_q^{[1]}f\right)(\zeta)}\right)\left(\left[\vartheta^{[\sigma]}\right]^2;q\right)+ \\ &\quad \left(\begin{array}{c}\alpha_1^2-\alpha_1\alpha_2 \\ +\left(\frac{\left(\eth_q^{[2]}f\right)(\zeta)}{\left(\eth_q^{[1]}f\right)(\zeta)}\right)^2 \\ -2\left(\frac{\left(\eth_q^{[3]}f\right)(\zeta)}{\left(\eth_q^{[1]}f\right)(\zeta)}\right)\end{array}\right)\left(\left[\vartheta^{[\sigma]}\right]^3;q\right)+\ldots \end{aligned} \tag{A9}$$

At this stage, we can use $\frac{f(x^{[\sigma]})}{\left(\eth_q^{[1]}f\right)(x^{[\sigma]})}\left[\frac{1}{1-\frac{\alpha_1 f(x^{[\sigma]})}{1+\alpha_2 f(x^{[\sigma]})}}\right]$ in the first-step of (13):

$$v^{[\sigma]}-\zeta = x^{[\sigma]}-\zeta-\frac{f(x^{[\sigma]})}{\left(\eth_q^{[1]}f\right)(x)}\left[\frac{1}{1-\frac{\alpha_1 f(x^{[\sigma]})}{1+\alpha_2 f(x^{[\sigma]})}}\right], \tag{A10}$$

$$v^{[\sigma]}-\zeta = \left(-\alpha_1+\frac{\left(\eth_q^{[2]}f\right)(\zeta)}{2!\left(\eth_q^{[1]}f\right)(\zeta)}\right)\left(\left[\vartheta^{[\sigma]}\right]^2;q\right)+\left(\left[\vartheta^{[\sigma]}\right]^3;q\right). \tag{A11}$$

Using the Taylor series to expand $f(v^{[\sigma]})$, we get

$$\begin{aligned} f(v^{[\sigma]}) \quad = \quad & \left(-\alpha_1 + \frac{\left(\eth_q^{[2]} f\right)(\zeta)}{2!\left(\eth_q^{[1]} f\right)(\zeta)}\right)\left(\left[\vartheta^{[\sigma]}\right]^2; q\right) + \\ & \left(\begin{array}{c} -\alpha_1^2 + \alpha_1\alpha_2 - 2\left(\frac{\left(\eth_q^{[2]} f\right)(\zeta)}{2!\left(\eth_q^{[1]} f\right)(\zeta)}\right)^2 \\ +2\left(\frac{\left(\eth_q^{[3]} f\right)(\zeta)}{3!\left(\eth_q^{[1]} f\right)(\zeta)}\right) \end{array}\right)\left(\left[\vartheta^{[\sigma]}\right]^3; q\right) + \\ & \left(\begin{array}{c} -\alpha_1^3 + 2\alpha_1^2\alpha_2 - \alpha_1^2\alpha_2^2\left(\frac{\left(\eth_q^{[2]} f\right)(\zeta)}{2!\left(\eth_q^{[1]} f\right)(\zeta)}\right) - \\ 3\alpha_1\left(\frac{\left(\eth_q^{[2]} f\right)(\zeta)}{2!\left(\eth_q^{[1]} f\right)(\zeta)}\right)^2 + \\ 5\left(\frac{\left(\eth_q^{[2]} f\right)(\zeta)}{2!\left(\eth_q^{[1]} f\right)(\zeta)}\right)^3 + \\ \alpha_1\left(\frac{\left(\eth_q^{[3]} f\right)(\zeta)}{3!\left(\eth_q^{[1]} f\right)(\zeta)}\right) - \\ 7\left(\frac{\left(\eth_q^{[2]} f\right)(\zeta)}{2!\left(\eth_q^{[1]} f\right)(\zeta)}\right)\left(\frac{\left(\eth_q^{[3]} f\right)(\zeta)}{3!\left(\eth_q^{[1]} f\right)(\zeta)}\right) \end{array}\right)\left(\left[\vartheta^{[\sigma]}\right]^4; q\right), \end{aligned} \tag{A12}$$

When (A12) is divided by (A3), we have:

$$\begin{aligned} \frac{f(y^{[\sigma]})}{\left(Đ_q^{[1]} f\right)(x^{[\sigma]})} \quad = \quad & \left(-\alpha_1 + \frac{\left(\eth_q^{[2]} f\right)(\zeta)}{2!\left(\eth_q^{[1]} f\right)(\zeta)}\right)\left(\left[\vartheta^{[\sigma]}\right]^2; q\right) + \\ & \left(\begin{array}{c} -\alpha_1^2 + \alpha_1\alpha_2 + \\ 2\alpha_1\frac{\left(\eth_q^{[2]} f\right)(\zeta)}{2!\left(\eth_q^{[1]} f\right)(\zeta)} - \\ 4\left(\frac{\left(\eth_q^{[2]} f\right)(\zeta)}{2!\left(\eth_q^{[1]} f\right)(\zeta)}\right)^2 + 2\frac{\left(\eth_q^{[3]} f\right)(\zeta)}{3!\left(\eth_q^{[1]} f\right)(\zeta)} \end{array}\right)\left(\left[\vartheta^{[\sigma]}\right]^3; q\right) + \\ & \left(\begin{array}{c} -\alpha_1^3 + 2\alpha_1^2\alpha_2 + \\ \alpha_1^2\frac{\left(\eth_q^{[2]} f\right)(\zeta)}{\left(\eth_q^{[1]} f\right)(\zeta)} - \\ \alpha_1^2\alpha_2^2 - \alpha_1\alpha_2\frac{\left(\eth_q^{[2]} f\right)(\zeta)}{2!\left(\eth_q^{[1]} f\right)(\zeta)} \\ -7\alpha_1\left(\frac{\left(\eth_q^{[2]} f\right)(\zeta)}{2!\left(\eth_q^{[1]} f\right)(\zeta)}\right)^2 + \\ 13\left(\frac{\left(\eth_q^{[2]} f\right)(\zeta)}{2\left(\eth_q^{[1]} f\right)(\zeta)}\right)^3 + \\ 4\alpha_1\left(\frac{\left(\eth_q^{[3]} f\right)(\zeta)}{3!\left(\eth_q^{[1]} f\right)(\zeta)}\right) - \\ 14\left(\frac{\left(\eth_q^{[2]} f\right)(\zeta)}{2!\left(\eth_q^{[1]} f\right)(\zeta)}\right)\left(\frac{\left(\eth_q^{[3]} f\right)(\zeta)}{3!\left(\eth_q^{[1]} f\right)(\zeta)}\right) + \\ 3\left(\frac{\left(\eth_q^{[4]} f\right)(\zeta)}{4!\left(\eth_q^{[1]} f\right)(\zeta)}\right) \end{array}\right)\left(\left[\vartheta^{[\sigma]}\right]^4; q\right). \end{aligned} \tag{A13}$$

When (A13) is used in the second-step of (13), we have:

$$z^{[\sigma]} - \zeta = v^{[\sigma]} - \zeta - \frac{f(v^{[\sigma]})}{\left(\eth_q^{[1]} f\right)(x^{[\sigma]})}, \tag{A14}$$

$$\vartheta^{[\sigma+1]} = \left(\begin{array}{c} -\alpha_1 \left(\dfrac{\left(\partial_q^{[2]} f\right)(\zeta)}{\left(\partial_q^{[1]} f\right)(\zeta)} \right) + \\ \left(\dfrac{\left(\partial_q^{[2]} f\right)(\zeta)}{2\left(\partial_q^{[1]} f\right)(\zeta)} \right)^2 \end{array} \right) \left(\left[\vartheta^{[\sigma]}\right]^3 ; q \right) + O\left(\left[\vartheta^{[\sigma]}\right]^4 ; q \right). \tag{A15}$$

This proves the third-order convergence of our numerical scheme (13). □

Appendix B

Table A1: In biomedical engineering application 1, the ANN is trained on the head of the input data set to find all roots of (53).

Table A1. Randomly generated initial guesses $X_1^{[0]}$.

$X^{[0]}$	$\overset{(0)}{x_1}$	$\overset{(0)}{x_2}$	$\overset{(0)}{x_3}$	$\overset{(0)}{x_4}$
$X_{11}^{[0]}$	0.101	−0.070	0.503	0.137
$X_{12}^{[0]}$	0.654	0.645	0.552	0.721
$X_{13}^{[0]}$	0.709	−0.111	0.652	0.345
⋮	⋮	⋮	⋮	⋮

Table A2: In biomedical engineering application 2, the ANN is trained on the head of the input data set to find all roots of (54).

Table A2. Randomly generated initial guesses $X_2^{[0]}$.

$X_2^{[0]}$	$\overset{(0)}{x_1}$	$\overset{(0)}{x_2}$	$\overset{(0)}{x_3}$	$\overset{(0)}{x_4}$	$\overset{(0)}{x_5}$	$\overset{(0)}{x_6}$	$\overset{(0)}{x_7}$	$\overset{(0)}{x_8}$
$X_{21}^{[0]}$	1.000	−0.631	0.994	0.967	0.111	0.909	0.643	0.967
$X_{22}^{[0]}$	0.007	−0.013	0.011	0.850	0.615	0.003	0.398	0.190
$X_{23}^{[0]}$	0.433	−0.882	0.544	0.167	0.076	0.113	0.943	0.700
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮

Appendix C

Using Algorithm 2 in CAS-MATLAB@2012b, the ANN is trained on the head of the output data set to find the real and imaginary roots of polynomial equations in engineering applications 1 to 2.

Table A3. The head of the output data set to find the real and imaginary roots of biomedical engineering applications 1 to 2.

ζ_0		ζ_1		ζ_2		ζ_3		ζ_4	
$\mathrm{Re}(\zeta_0)$	$\mathrm{Im}(\zeta_0)$	$\mathrm{Re}(\zeta_1)$	$\mathrm{Im}(\zeta_1)$	$\mathrm{Re}(\zeta_2)$	$\mathrm{Im}(\zeta_2)$	$\mathrm{Re}(\zeta_3)$	$\mathrm{Im}(\zeta_3)$	$\mathrm{Re}(\zeta_4)$	$\mathrm{Im}(\zeta_4)$
0.004	0.327	0.5343	0.3143	4.925	0.156	8.343	6.655	9.313	8.443
0.075	4.187	4.232	4.315	0.014	0.015	8.779	9.115	5.343	9.876
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮
					⋮				
ζ_6		ζ_7		ζ_8					
$\mathrm{Re}(\zeta_6)$	$\mathrm{Im}(\zeta_6)$	$\mathrm{Re}(\zeta_7)$	$\mathrm{Im}(\zeta_7)$	$\mathrm{Re}(\zeta_8)$	$\mathrm{Im}(\zeta_8)$				
0.3213	0.110	0.547	2.098	9.232	8.5443				
0.0045	0.543	4.098	9.112	0.995	91458				
⋮	⋮	⋮	⋮	⋮	⋮				

References

1. Akbari, M.R.; Ganji, D.D.; Nimafar, M.; Ahmadi, A.R. Significant progress in solution of nonlinear equations at displacement of structure and heat transfer extended surface by new AGM approach. *Front. Mech. Eng.* **2014**, *9*, 390–401. [CrossRef]
2. Akbari, M.R. Akbari-Ganjis method AGM to chemical reactor design for non-isothermal and non-adiabatic of mixed flow reactors. *J. Chem. Eng. Mater. Sci.* **2020**, *11*, 1–9.
3. Von-Karman, T. The engineer grapples with nonlinear problems. *Bull. Am. Math. Soc.* **1940**, *46*, 615–683. [CrossRef]
4. Cordero, A.; Garrido, N.; Torregrosa, J.R.; Triguero-Navarro, P. Iterative schemes for finding all roots simultaneously of nonlinear equations. *Appl. Math. Lett.* **2022**, *134*, 108325. [CrossRef]
5. Fredlund, D.G. Unsaturated soil mechanics in engineering practice. *J. Geotech. Geoenviron. Eng.-ASCE* **2006**, *132* , 286–321. [CrossRef]
6. Baranovskii, E.S.; Artemov, M.A. Optimal control for a nonlocal model of non-Newtonian fluid flows. *Mathematics* **2021**, *9*, 275. [CrossRef]
7. Marchildon, L. *Quantum Mechanics: From Basic Principles to Numerical Methods and Applications*; Springer Science & Business Media: Berlin/Heidelberg, Germany, 2013.
8. Johnson, C.R. Computational and numerical methods for bioelectric field problems. *Crit. Rev. Biomed. Eng.* **1997**, *25*, 1–10. [CrossRef]
9. Mao, X.; Szpruch, L. Strong convergence and stability of implicit numerical methods for stochastic differential equations with non-globally Lipschitz continuous coefficients. *J. Comput. Appl. Math.* **2013**, *238*, 14–28. [CrossRef]
10. Lux, T. Estimation of an agent-based model of investor sentiment formation in financial markets. *J. Econ. Dyn. Cont.* **2012**, *36*, 1284–1302. [CrossRef]
11. Alekseev, V.B. *Abel's Theorem in Problems and Solutions: Based on the Lectures of Professor VI Arnold*; Springer: Dordrecht, The Netherlands, 2004.
12. Cordero, A.; Neta, B.; Torregrosa, J.R. Memorizing Schrö der's method as an efficient strategy for estimating roots of unknown multiplicity. *Mathematics* **2021**, *9*, 2570. [CrossRef]
13. Akram, S.; Akram, F.; Junjua, M.U.D.; Arshad, M.; Afzal, T. A family of optimal eighth order iteration functions for multiple roots and its dynamics. *J. Math.* **2021**, *2021*, 5597186. [CrossRef]
14. Erfanifar, R.; Hajarian, M. A new multi-step method for solving nonlinear systems with high efficiency indices. *Numer. Algorithms* **2024**, 1–26. [CrossRef]
15. Sugiura, H.; Hasegawa, T. On the global convergence of Schröder's iterative formulae for real roots of algebraic equations. *J. Comput. Appl. Math.* **2018**, *344*, 313–322. [CrossRef]
16. Proinov, P.D.;Vasileva, M.T. Local and semilocal convergence of Nourein's iterative method for finding all zeros of a polynomial simultaneously. *Symmetry* **2020**, *12*, 1801. [CrossRef]
17. Ivanov, S.I. A unified semilocal convergence analysis of a family of iterative algorithms for computing all zeros of a polynomial simultaneously. *Numer. Algorithms* **2017**, *75*, 1193–1204. [CrossRef]
18. 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.
19. Kerner, I.O. Ein gesamtschrittverfahren zur berechnung der nullstellen von polynomen. *Numer. Math.* **1966**, *8*, 290–294. [CrossRef]
20. Dochev, M. Modified Newton method for the simultaneous computation of all roots of a given algebraic equation. *Phys. Math. J. Bulg. Acad. Sci.* **1962**, *5*, 136–139.
21. Nedzhibov, G.H. Improved local convergence analysis of the Inverse Weierstrass method for simultaneous approximation of polynomial zeros. In Proceedings of the MATTEX 2018 Conference, Targovishte, Bulgaria, 16–17 October 2018; Volume 1, pp. 66–73.
22. Marcheva, P.I.; Ivanov, S.I. Convergence analysis of a modified Weierstrass method for the simultaneous determination of polynomial zeros. *Symmetry* **2020**, *12*, 1408. [CrossRef]
23. 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]
24. Alefeld, G.; Herzberger, J. On the convergence speed of some algorithms for the simultaneous approximation of polynomial roots. *SIAM J. Numer. Anal.* **1974**, *11*, 237–243. [CrossRef]
25. 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]
26. Nemri, A.; Soltani, F. Analytical approximation formulas in quantum calculus. *Math. Mech. Solid.* **2017**, *22*, 2075–2090. [CrossRef]
27. Noeiaghdam, Z.; Rahmani, M.; Allahviranloo, T. Introduction of the numerical methods in quantum calculus with uncertainty. *J. Math. Model.* **2021**, *9*, 303–322.
28. Al-Salih, R. Dynamic network flows in quantum calculus. *J. Analy. Appl.* **2020**, *18*, 53–66.
29. Sinha, A.K.; Panda, S. Shehu Transform in Quantum Calculus and Its Applications. *Int. J. Appl.Comput. Math.* **2022**, *8*, 1–19. [CrossRef]
30. Alhindi, K.R. Convex Families of q-Derivative Meromorphic Functions Involving the Polylogarithm Function. *Symmetry* **2023**, *15*, 1388. [CrossRef]

31. Akça, H.; Benbourenane, J.; Eleuch, H. The q-derivative and differential equation. *J. Phys. Conf. Ser.* **2019**, *1411*, 12002. [CrossRef]
32. Sana, G.; Mohammed, P.O.; Shin, D.Y.; Noor, M.A.; Oudat, M.S. On iterative methods for solving nonlinear equations in quantum calculus. *Fractal Fract.* **2021**, *5*, 60. [CrossRef]
33. Georgiev, S.G.; Tikare, S. Taylor's formula for general quantum calculus. *J. Math. Model.* **2023**, *11*, 491–505.
34. Sheng, Y.; Zhang, T. Some Results on the q-Calculus and Fractional q-Differential Equations. *Mathematics* **2022**, *10*, 64. [CrossRef]
35. Proinov, P.D. General convergence theorems for iterative processes and applications to the Weierstrass root-finding method. *J. Complex.* **2016**, *33*, 118–144. [CrossRef]
36. Proinov, P.D.; Petkova, M.D. Convergence of the two-point Weierstrass root-finding method. *Jpn. J. Ind. Appl. Math.* **2014**, *31*, 279–292. [CrossRef]
37. Huang, D.S.; Chi, Z. Finding complex roots of polynomials by feed forward neural networks. In Proceedings of the IJCNN'01, International Joint Conference on Neural Networks, Cat. No. 01CH37222, Washington, DC, USA, 15–19 July 2001; Volumes 15–19, p. A13.
38. Huang, D.S.; Chi, Z. Neural networks with problem decomposition for finding real roots of polynomials. In Proceedings of the IJCNN'01, International Joint Conference on Neural Networks, 01CH37222 , Washington, DC, USA, 15–19 July 2001; p. A25.
39. Huang, D.S.; Ip, H.H.; Chi, Z.; Wong, H.S. Dilation method for finding close roots of polynomials based on constrained learning neural networks. *Phys. Lett. A* **2003**, *309*, 443–451. [CrossRef]
40. Daws, J., Jr.; Webster, C G. A polynomial-based approach for architectural design and learning with deep neural networks. *arXiv* **2019**, arXiv:1905.10457.
41. Mourrain, B.; Pavlidis, N.G.; Tasoulis, D.K.; Vrahatis, M.N. Determining the number of real roots of polynomials through neural networks. *Comput. Math. Appl.* **2006**, *51*, 527–536. [CrossRef]
42. Huang, D.; Chi, Z. Finding roots of arbitrary high order polynomials based on neural network recursive partitioning method. *Sci. China Inf. Sci.* **2004**, *47*, 232–245. [CrossRef]
43. Freitas, D.; Guerreiro Lopes, L.; Morgado-Dias, F. A Neural network-based approach for approximating arbitrary roots of polynomials. *Mathematics* **2021**, *9*, 317. [CrossRef]
44. Shams, M.; Carpentieri, B. On highly efficient fractional numerical method for solving nonlinear engineering models. *Mathematics* **2023**, *11*, 4914. [CrossRef]
45. Huang, D.S. A constructive approach for finding arbitrary roots of polynomials by neural networks. *IEEE Trans. Neural Netw.* **2004**, *15*, 477–491. [CrossRef] [PubMed]
46. Börsch-Supan, W. Residuenabschätzung für Polynom-Nullstellen mittels Lagrange-Interpolation. *Numer. Math.* **1970**, *14*, 287–296. [CrossRef]
47. Rafiq, N.; Mir, N.A.; Yasmin, N. Some two-step simultaneous methods for determining all the roots of a non-linear equation. *Life Sci. J.* **2013**, *10*, 54–59.
48. Nedzhibov, G.H.; Petkov, M.G. On a family of iterative methods for simultaneous extraction of all roots of algebric polynomial, *Appl. Math. Comput.* **2005**, *162*, 427–433.
49. Mir, N.A.; Shams, M.; Rafiq, N.; Akram, S.; Ahmed, R. On family of simultaneous method for finding distinct as well as multiple roots of non-linear equation. *Punjab Univ. J. Math.* **2020**, *52*, 1–10.
50. Saravanakumar, T.; Nirmala, V.J.; Raja, R.; Cao, J.; Lu, G. Finite-time reliable dissipative control of neutral-type switched artificial neural networks with non-linear fault inputs and randomly occurring uncertainties. *Asian J. Cont.* **2020**, *22*, 2487–2499. [CrossRef]
51. Polyanin, A.D.; Manzhirov, A.V. *Handbook of Mathematics for Engineers and Scientists*; CRC Press: Boca Raton, FL, USA, 2006.
52. Fournier, R.L. *Basic Transport Phenomena in Biomedical Engineering*; Taylor Franics: New York, NY, USA, 2007.
53. Petković, M.S.; Tričković, S.; Herceg, D. On Euler-like methods for the simultaneous approximation of polynomial zeros. *Jpn. J. Indust. Appl. Math.* **1998**, *15*, 295–315. [CrossRef]
54. Neumaier, A.; Schäfer, A. Divided differences, shift transformations and Larkin's root finding method. *Math. Comput.* **1985**, *45*, 181–196.
55. Argyros, I.K.; Magreñán, Á.A.; Orcos, L. Local convergence and a chemical application of derivative free root finding methods with one parameter based on interpolation. *J. Math. Chem.* **2016**, *54*, 1404–1416. [CrossRef]
56. Wu, X.; Shao, R.; Zhu, Y. Jacobi-free and complex-free method for finding simultaneously all zeros of polynomials having only real zeros. *Comput. Math. Appl.* **2003**, *46*, 1387–1395. [CrossRef]
57. Proinov, P.D. On the local convergence of Ehrlich method for numerical computation of polynomial zeros. *Calcolo* **2016**, *53*, 413–426. [CrossRef]