mathematics

MDPI

Article

Numerical Solution of the Nonlinear Convection–Diffusion Equation Using the Fifth Order Iterative Method by Newton–Jarratt

Santiago Quinga [1], Wilson Pavon [2,*], Nury Ortiz [1], Héctor Calvopiña [3,*], Gandhy Yépez [1] and Milton Quinga [1]

[1] Departamento de Ciencias Exactas, Universidad de las Fuerzas Armadas ESPE, Sangolqui 171103, Ecuador; sdquinga1@espe.edu.ec (S.Q.); ngortiz1@espe.edu.ec (N.O.); gdyepez@espe.edu.ec (G.Y.); mgquinga@espe.edu.ec (M.Q.)
[2] Facultad de Ciencias de la Ingeniería e Industrias, Universidad UTE, Av. Mariscal Sucre, Quito 170129, Ecuador
[3] Departamento de Ciencias de la Energía y Mecánica, Universidad de las Fuerzas Armadas ESPE, Sangolqui 171103, Ecuador
* Correspondence: wilson.pavon@ute.edu.ec (W.P.); hacalvopina@espe.edu.ec (H.C.)

Abstract: This study presents a novel fifth-order iterative method for solving nonlinear systems derived from a modified combination of Jarratt and Newton schemes, incorporating a frozen derivative of the Jacobian. The method is applied to approximate solutions of the nonlinear convection–diffusion equation. A MATLAB script function was developed to implement the approach in two stages: first, discretizing the equation using the Crank–Nicolson Method, and second, solving the resulting nonlinear systems using Newton's iterative method enhanced by a three-step Jarratt variant. A comprehensive analysis of the results highlights the method's convergence and accuracy, comparing the numerical solution with the exact solution derived from linear parabolic partial differential transformations. This innovative fifth-order method provides an efficient numerical solution to the nonlinear convection–diffusion equation, addressing the problem through a systematic methodology that combines discretization and nonlinear equation solving. The study underscores the importance of advanced numerical techniques in tackling complex problems in physics and mathematics.

Keywords: convection–diffusion; Crank–Nicolson; Newton–Jarrat; iterative method

MSC: 65N12

check for updates

Received: 5 February 2025
Revised: 24 March 2025
Accepted: 27 March 2025
Published: 1 April 2025

Citation: Quinga, S.; Pavon, W.; Ortiz, N.; Calvopiña, H.; Yépez, G.; Quinga, M. Numerical Solution of the Nonlinear Convection–Diffusion Equation Using the Fifth Order Iterative Method by Newton–Jarratt. *Mathematics* **2025**, *13*, 1164. https://doi.org/10.3390/math13071164

Copyright: © 2025 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (https://creativecommons.org/licenses/by/4.0/).

1. Introduction

Several physical phenomena can be described using nonlinear partial differential equations [1–4]. One of these phenomena is fluid dynamics, where analytical solutions are only possible for certain limited cases. Due to this limitation, many numerical techniques have been developed to obtain approximate numerical solutions for convection–diffusion problems [5–10].

The nonlinear convection–diffusion equation (NCDE) plays a critical role in modeling various physical phenomena involving the transport of quantities under the combined effects of nonlinear convection and diffusion [11–14].

In fields, such as aerodynamics and meteorology, advective transport is often more significant than diffusion. This is because the Péclet number, which compares advection and diffusion rates, is high in these contexts, indicating that advection dominates mass or energy

transport [15]. In contrast, diffusion is more relevant in hydrology and oceanography, where the Péclet number is low or moderate. This difference influences modeling phenomena, such as pollutant dispersion and salinity conditions in water bodies [16].

Numerical methods are indispensable for solving the NCDE, particularly when analytical solutions are unattainable. Classical approaches, such as finite difference, finite volume, finite element methods, and boundary element methods, have been widely utilized to discretize the equation [17]. However, these techniques often result in nonlinear systems that demand efficient iterative methods to achieve convergence [18]. The Crank–Nicolson scheme, for example, is a popular choice for time discretization due to its balance of stability and accuracy, but solving the resulting equations reliably remains a nontrivial task [19].

Advancements in iterative solvers, particularly Newton-type methods, have paved the way for tackling these nonlinear systems with improved efficiency. Newton's method is valued for its quadratic convergence but is susceptible to issues, such as divergence in poorly conditioned problems. Consequently, researchers have focused on developing high-order iterative methods that enhance robustness and accelerate convergence [20]. Techniques, such as barycentric interpolation, localized space–time methods, and hybrid schemes, have shown promising results in improving computational efficiency and solution accuracy for NCDEs and related problems [21].

This study introduces a novel fifth-order iterative method, combining elements of the Newton and Jarratt methods with a frozen Jacobian derivative to address the challenges posed by the NCDE. The proposed approach integrates the Crank–Nicolson method for time discretization with the iterative solver to achieve a robust and efficient solution process. By leveraging high-order convergence properties, the method demonstrates the potential to improve accuracy and reduce computational cost, addressing the demands of solving nonlinear systems arising from the NCDE.

The remainder of this paper is structured as follows: Section 2 describes the mathematical formulation of the NCDE and its discretization using the Crank–Nicolson method and the proposed iterative scheme, including its derivation and computational implementation. Numerical results and comparisons are presented in Section 3 to evaluate the method's performance. Finally, Section 4 concludes with a summary of findings and potential directions for future research.

This work contributes to the development of advanced numerical techniques for solving nonlinear convection–diffusion problems, offering insights into applying high-order iterative methods in scientific and engineering contexts.

2. Methodology

The nonlinear convection–diffusion law is an equation that describes the transport of mass or momentum of a fluid through a medium, typically in the presence of a concentration or velocity gradient. This law is applied in physics, chemistry, and engineering to model various phenomena, such as aquifer contamination, the diffusion of chemicals in a liquid, heat flow in a fluid, and others.

The nonlinear convection–diffusion law corresponds to the following mathematical model:

$$\frac{\partial u}{\partial t} + u\frac{\partial u}{\partial x} = \kappa\frac{\partial^2 u}{\partial x^2}, \tag{1}$$

where $t > 0$, κ is a viscous diffusion coefficient that depends on the properties of each fluid and this coefficient is inversely related to the Péclet number, u is the velocity field which is directly related to the Péclet number, $u\frac{\partial u}{\partial x}$ is a nonlinear convection term and $\kappa\frac{\partial^2 u}{\partial x^2}$ is

a diffusion term [11]. Additionally, the following initial condition and border conditions are used:

$$u(x,0) = f(x),\ a < x < b, \tag{2}$$

$$u(a,t) = g_1(t) \tag{3}$$

$$u(a,t) = g_1(t) \tag{4}$$

where $g_1 y g_2$ are time-dependent functions provided as part of the problem.

The exact solution of the equation refers to a solution where the coefficient κ remains constant within the context of the nonlinear convection–diffusion equation. In general, the solution involves applying initial and boundary conditions that may result in analytical expressions, such as Gaussian profiles or other functions that satisfy the equation's form. The specifics depend on the convection velocity and the boundary conditions used. To solve this equation, the Cole Hopf transformation is used, which allows obtaining a linear parabolic partial differential equation [22–24], which is then solved exactly to obtain its solution $u(x,t)$ using Fourier series with coefficients A_0, A_n as shown in Equation (5). This kind of solution is typically used for problems involving heat conduction, diffusion, and similar processes, where the Fourier series provides an effective way to express the solution in terms of its harmonics, especially when solving for time-dependent or spatially varying conditions [25].

$$u(x,t) = 2\pi\kappa \frac{\sum_{n=1}^{\infty} A_n e^{-\kappa n^2 \pi^2 t} \sin(n\pi x)}{A_0 + \sum_{n=1}^{\infty} A_n e^{-\kappa n^2 \pi^2 t} \cos(n\pi x)} \tag{5}$$

By employing forward finite differences, the function's first derivative concerning x and t can be approximated [22]. The formulas for these approximations are derived based on the discrete points in the domain, enabling a computationally efficient way to solve partial differential equations. The mathematical models are as follows:

$$\frac{\partial u(x_i, t_j)}{\partial x} \approx \frac{u_{i+1,j} - u_{i,j}}{h}, \tag{6}$$

$$\frac{\partial u(x_i, t_j)}{\partial t} \approx \frac{u_{i,j+1} - u_{i,j}}{k}, \tag{7}$$

where $u_{i,j}$ represents the value of the function $u(x,t)$ in the node (x_i, t_j), h y k represent the horizontal and vertical steps, respectively, and this type of discretization has a truncation error of order 1.

Using central differences for the second derivative of the function concerning x and t, which has a truncation error of order 2, the following formulas are obtained:

$$\frac{\partial^2 u(x_i, t_j)}{\partial x^2} \approx \frac{u_{i+1,j} - 2u_{i,j} + u_{i-1,j}}{h^2}, \tag{8}$$

$$\frac{\partial^2 u(x_i, t_j)}{\partial t^2} \approx \frac{u_{i,j+1} - 2u_{i,j} + u_{i,j-1}}{k^2}. \tag{9}$$

The finite difference method is used to discretize the domain, thereby generating a system of equations that can be solved using numerical techniques to yield an approximate solution. This method is widely applied in fields, such as fluid dynamics, heat transfer, and wave propagation, where the transformation of continuous problems into discrete analogs facilitates numerical computation and simulation.

Nonlinear equations are typically expressed as $G(x) = 0$, where G is a nonlinear real-valued function that can be represented as a system of equations in the following form:

$$\begin{cases} g_1(x_1, x_2, \ldots x_n) = 0 \\ g_2(x_1, x_2, \ldots x_n) = 0 \\ \vdots \\ g_n(x_1, x_2, \ldots x_n) = 0 \end{cases} \tag{10}$$

where $x_1, x_2, \ldots, x_n$ are the unknowns of the system and $g_1, g_2, \ldots, g_n$ are the nonlinear arbitrary functions related to the unknowns and must be set equal to zero to satisfy the system.

It should also be considered that solving a system of nonlinear equations involves finding the values of x_i that make all the equations equal to zero simultaneously. This requires an iterative approach, where each iteration improves the estimate of the solution based on previous values. The process continues until a solution is found that satisfies the system to a predefined level of accuracy. Nonlinear systems of equations can be solved using a fixed-point method. This method transforms the original system into an equivalent form, which typically involves reformulating the equations so that the unknowns are expressed as functions of the other variables. The system is then solved iteratively by repeatedly applying the transformation until convergence is reached, where the values of the unknowns satisfy the equations within a specified tolerance [26].

The general form of the fixed-point method can be expressed as follows:

$$x = G(x), \tag{11}$$

For a function $G : \mathbb{R}^n \to \mathbb{R}^n$, whose coordinate functions are described by $g_i, i = 1, 2, \ldots n$; if we start with the approximation $x^{(0)} = \left(x_1{}^{(0)}, x_2{}^{(0)}, \ldots x_n{}^{(0)}\right)^T \in \mathbb{R}^n$ the following iterative formula is generated:

$$x^{(k+1)} = G\left(x^{(k)}\right), k = 0, 1, \ldots \tag{12}$$

where $x^{(k+1)}$ is an iterated value that is obtained from the previous iterated value $x^{(k)}$.

If $\left\{x^{(k+1)}\right\} \to \xi$ when $k \to \infty$, then it is said that the process converges. Under certain conditions of G, $\xi = (\xi_1, \xi_2, \ldots \xi_n)^T \in \mathbb{R}^n$ is a solution to the system $x = G(x)$, and ξ is called the fixed point of the function G, and thus, Equation (12) is known as the fixed-point method.

The order of convergence establishes that $\left(x^{(k)}\right)_{k \geq 0}$ is a sequence in $\mathbb{R}^n$ converging to the fixed point ξ, then it is possible to obtain a linear convergence if

$$\exists M, 0 < M < 1 \text{ y } \exists k_0 / \left\| x^{(k+1)} - \xi \right\| \leq \left\| x^{(k)} - \xi \right\|, \forall k \geq k_0 \tag{13}$$

a quadratic convergence if

$$\exists M, M > 0 \text{ y } \exists k_0 / \left\| x^{(k+1)} - \xi \right\| \leq \left\| x^{(k)} - \xi \right\|^2, \forall k \geq k_0 \tag{14}$$

and a convergence of order p if

$$\exists M, M > 0 \text{ y } \exists k_0 / \left\| x^{(k+1)} - \xi \right\| \leq \left\| x^{(k)} - \xi \right\|^p, \forall k \geq k_0 \tag{15}$$

An iterative method involves carrying out operations repeatedly, but this process must end at some point to generate results. For an iterative process to be completed, termination conditions must be defined.

To demonstrate the convergence order, it is necessary to apply Taylor's developments as follows [27]:

If $F : B \subseteq \mathbb{R}^n \to \mathbb{R}^n$ is a function sufficiently Fréchet differentiable around B. Where qth derivative with $q \geq 1$ of F in $u \in \mathbb{R}^n$, is the qlinear function described by $F^{(q)}(u) : \mathbb{R}^n \times \cdots \times \mathbb{R}^n \to \mathbb{R}^n$ such that $F^{(q)}(u)(v_1, \ldots, v_q) \in \mathbb{R}^n$. Then:

(1) $F^{(q)}(u)(v_1, \ldots, v_{q-1}) \in \mathscr{L}(\mathbb{R}^n)$,

(2) $F^{(q)}(u)\left(v_{\sigma(1)}, \ldots, v_{\sigma(q)}\right) = F^{(q)}(u)(v_1, \ldots, v_q)$, or any permutation σ of $\{1, 2, \ldots, q\}$.

From the above, we obtain the following:

(a) $F^{(q)}(u)(v_1, \ldots, v_q) = F^{(q)}(u)v_1 \cdots v_q$,

(b) $F^{(q)}(u)v^{q-1}F^{(p)}v^p = F^{(q)}(u)F^{(p)}(u)v^{q+p-1}$

Applying the Taylor expansion for $\xi + h \in \mathbb{R}^n$ bounded around the solution ξ of $F(x) = 0$ and if the Jacobian matrix $F'(\xi)$ is nonsingular, we obtain the following:

$$F(\xi + h) = F'(\xi)\left[h + \sum_{q=2}^{p} C_q h^q\right] + O\left[h^{p+1}\right], \tag{16}$$

where $C_q = \frac{1}{q!}[F'(\xi)]^{-1}F^{(q)}(\xi)$, for $q \geq 2$. It can be seen that $C_q h^q \in \mathbb{R}^n$ because $F^{(q)}(\xi) \in \mathscr{L}(\mathbb{R}^n \times \cdots \times \mathbb{R}^n, \mathbb{R}^n)$ and $[F'(\xi)]^{-1} \in \mathscr{L}(\mathbb{R}^n)$. Moreover, $F'(\xi + \mathrm{h})$ can be expressed as follows:

$$F'(\xi + h) = F'(\xi)\left[I + \sum_{q=2}^{p-1} qC_q h^{q-1}\right] + O[h^p], \tag{17}$$

where I is the identity matrix and $qC_q h^{q-1} \in \mathscr{L}(\mathbb{R}^n)$.

For the inverse of the Jacobian described in Equation (17), the Taylor expansion is applied, yeilding the following result:

$$\left[F'(\xi + h)\right]^{-1} = \left[I + X_2 h + X_3 h^2 + X_4 h^4 + \cdots\right]\left[F'(\xi)\right]^{-1} + O[h^p] \tag{18}$$

in which

$$\begin{gathered} X_2 = -2C_2 \\ X_3 = 4C_2^2 - 3C_3 \\ X_4 = 6C_3C_2 - 8C_2^3 + 6C_2C_3 - 4C_4 \\ X_5 = 16C_2^4 - 12C_3C_2^2 - 12C_2C_3C_2 + 8C_4C_2 + 9C_3^2 - 12C_2^2C_3 + 8C_2C_4 - 5C_5 \end{gathered} \tag{19}$$

Denoting the error at the kth iteration as follows: $e_k = x^{(k)} - \xi$. Then, the following error equation is obtained:

$$e_{k+1} = Le_k^p + O\left[e_k^{p+1}\right] \tag{20}$$

where p is the order of convergence of the method; L is a function $p - lineal$, and $L \in \mathscr{L}(\mathbb{R}^n \times \cdots \times \mathbb{R}^n, \mathbb{R}^n)$.

2.1. Newton's Method

Newton's method allows us to determine the roots of an equation from a given initial point. It is particularly effective when an accurate initial estimate is available and is noted for its simplicity and speed of convergence, reaching an order of 2 [28].

Newton's method for solving nonlinear systems has the following form:

$$x^{(k+1)} = x^{(k)} - F'\left(x^{(k)}\right)^{-1} F\left(x^{(k)}\right), \tag{21}$$

where $F'(x)$ is the Jacobian matrix, which is associated to the function F.

2.2. Jarratt Method

To achieve more economical computational results and improve the order of convergence, modifications to Newton's method are proposed, giving rise to Jarratt's method, which presents the following iterative structure for a nonlinear equation:

$$\begin{cases} y_n = x_n - \frac{2f(x_n)}{3f'(x_n)} \\ x_{n+1} = x_n - \left(\frac{3f'(y_n)+f'(x_n)}{6f'(y_n)-2f'(x_n)}\right)\frac{f(x_n)}{f'(x_n)} \end{cases} \tag{22}$$

where $f'(x)$ represents the drift of the function f. After taking into account specific considerations, this method can be used to solve systems of nonlinear equations [29,30].

2.3. Composition of Iterative Methods

Three-step methods start from a Newton scheme of order p, which allows obtaining composite methods with a frozen Jacobian matrix that reaches an order of convergence of $p+1$. The fifth-order iterative method scheme proposed in this research will be called NJN, and it is a variant of the Jarratt method proposed by Junjua et al. [30] called MSNF-2:

$$\begin{gathered} y^{(k)} = x^{(k)} - \tfrac{2}{3}\left[F'\left(x^{(k)}\right)\right]^{-1} F\left(x^{(k)}\right), \\ x^{(k+1)} = x^{(k)} - \left[I + \tfrac{1}{4}\left(\left[F'\left(x^{(k)}\right)\right]^{-1} F'\left(y^{(k)}\right) - I\right) + \tfrac{3}{8}\left(\left[F'\left(x^{(k)}\right)\right]^{-1} F'\left(y^{(k)}\right) - I\right)^2\right]\left[F'\left(y^{(k)}\right)\right]^{-1} F\left(x^{(k)}\right), \end{gathered} \tag{23}$$

By composing Newton's method by freezing the Jacobian matrix, the following scheme is formulated for the new iterative method proposed in this study called NJN:

$$\begin{gathered} z^{(k)} = x^{(k)} - \tfrac{2}{3}\left[F'\left(x^{(k)}\right)\right]^{-1} F\left(x^{(k)}\right), \\ y^{(k)} = x^{(k)} - \left[I + \tfrac{1}{4}\left(\left[F'\left(x^{(k)}\right)\right]^{-1} F'\left(z^{(k)}\right) - I\right) + \tfrac{3}{8}\left(\left[F'\left(x^{(k)}\right)\right]^{-1} F'\left(z^{(k)}\right) - I\right)^2\right]\left[F'\left(z^{(k)}\right)\right]^{-1} F\left(x^{(k)}\right), \\ x^{(k+1)} = y^{(k)} - \left[F'\left(x^{(k)}\right)\right]^{-1} F\left(y^{(k)}\right) \end{gathered} \tag{24}$$

2.4. High Order Methods

There are high order iterative methods such us SHM5. SHM5 is the name of a fifth order method made by Sharma and Gupta en [31] with the formula:

$$\begin{aligned} y^{(k)} &= x^{(k)} - \tfrac{1}{2}\left[F'\left(x^{(k)}\right)\right]^{-1} F\left(x^{(k)}\right), \\ z^{(k)} &= x^{(k)} - \left[F'\left(y^{(k)}\right)\right]^{-1} F\left(x^{(k)}\right), \\ x^{(k+1)} &= z^{(k)} - \left[2F'\left(y^{(k)}\right) - F'\left(x^{(k)}\right)\right]^{-1} F\left(z^{(k)}\right), \end{aligned} \tag{25}$$

Another high order method is the one called O6, made by Padilla et al. [32], which has a sixth order and the formula:

$$\begin{aligned} y^{(k)} &= x^{(k)} - \tfrac{1}{2}\left[F'\left(x^{(k)}\right)\right]^{-1} F\left(x^{(k)}\right), \\ z^{(k)} &= x^{(k)} - \left[F'\left(y^{(k)}\right)\right]^{-1} F\left(x^{(k)}\right), \\ x^{(k+1)} &= z^{(k)} - \left[2F'\left(y^{(k)}\right) - F'\left(x^{(k)}\right)\right]^{-1} F\left(z^{(k)}\right), \end{aligned} \tag{26}$$

2.5. Convergence Analysis

For the convergence analysis, is assumed that $F : B \subseteq \mathbb{R}^n \to \mathbb{R}^n$ is a sufficiently Fréchet differentiable function on B and ξ is a root of $F(x) = 0$ such that its Jacobian matrix is continuous and nonsingular on B; then, the new method of Equation (24) has convergence order 5.

By performing Taylor series developments for $F\left(x^{(k)}\right)$ y $F'^{(x^{(k)})}$, the following equations are obtained:

$$F\left(x^{(k)}\right) = F'(\xi)\left[e_k + C_2 e_k^2 + C_3 e_k^3 + C_4 e_k^4 + C_5 e_k^5\right] + O\left[e_k^6\right] \tag{27}$$

$$F'\left(x^{(k)}\right) = F'(\xi)\left[I + 2C_2 e_k + 3C_3 e_k^2 + 4C_4 e_k^3 + 5C_5 e_k^4\right] + O\left[e_k^5\right] \tag{28}$$

Using Equation (18), the inverse of the Jacobian matrix is as follows:

$$\begin{aligned} \left[F'\left(x^{(k)}\right)\right]^{-1} = [I \quad & -2C_2 e_k + \left(4C_2^2 - 3C_3\right)e_k^2 + \left(6C_3C_2 - 8C_2^3 + 6C_2C_3 - 4C_4\right)e_k^3 \\ & + \left(16C_2^4 - 12C_3C_2^2 - 12C_2C_3C_2 + 8C_4C_2 + 9C_3^2 - 12C_2^2C_3 + 8C_2C_4 - 5C_5\right)e_k^4]\,[F'(\xi)]^{-1} \\ & + O\left[e_k^5\right] \end{aligned} \tag{29}$$

Using Equations (29) and (27):

$$\begin{aligned} &\left[F'\left(x^{(k)}\right)\right]^{-1} F\left(x^{(k)}\right) \\ &\quad = e_k - C_2 e_k^2 + \left(2C_2^2 - 2C_3\right)e_k^3 + \left(-3C_4 + 4C_2C_3 + 3C_3C_2 - 4C_2^3\right)e_k^4 \\ &\quad + \left(-4C_5 + 6C_2C_4 - 8C_2^2C_3 + 6C_3^2 - 6C_3C_2^2 + 8C_2^4 - 6C_2C_3C_2 + 4C_4C_2\right)e_k^5 + O\left[e_k^6\right] \end{aligned} \tag{30}$$

Replacing $e_k = x^{(k)} - \xi$ and using Equation (30) in the first step of the iterative method described in Equation (24):

$$\begin{aligned} z^{(k)} = \xi + \tfrac{1}{3}e_k + \tfrac{2}{3}C_2 e_k^2 - \tfrac{2}{3}\left(2C_2^2 - 2C_3\right)e_k^3 - \tfrac{2}{3}\left(-3C_4 + 4C_2C_3 + 3C_3C_2 - 4C_2^3\right)e_k^4 \\ - \tfrac{2}{3}\left(-4C_5 + 6C_2C_4 - 8C_2^2C_3 + 6C_3^2 - 6C_3C_2^2 + 8C_2^4 - 6C_2C_3C_2 + 4C_4C_2\right)e_k^5 + O\left[e_k^6\right] \end{aligned} \tag{31}$$

From Equation (31), $z^{(k)} = \xi + e_{z^{(k)}}$, then

$$\begin{aligned} e_{z^{(k)}} = \tfrac{1}{3}e_k + \tfrac{2}{3}C_2 e_k^2 - \tfrac{2}{3}\left(2C_2^2 - 2C_3\right)e_k^3 - \tfrac{2}{3}\left(-3C_4 + 4C_2C_3 + 3C_3C_2 - 4C_2^3\right)e_k^4 \\ - \tfrac{2}{3}\left(-4C_5 + 6C_2C_4 - 8C_2^2C_3 + 6C_3^2 - 6C_3C_2^2 + 8C_2^4 - 6C_2C_3C_2 + 4C_4C_2\right)e_k^5 + O\left[e_k^6\right] \end{aligned} \tag{32}$$

Thus, the following development is obtained for the Jacobian matrix $F'\left(z^{(k)}\right)$:

$$\begin{aligned} F'\left(z^{(k)}\right) = F'(\xi)\Big[I + \tfrac{2}{3}C_2 e_k + \left(\tfrac{4}{3}C_2^2 + \tfrac{1}{3}C_3\right)e_k^2 + \left(-\tfrac{8}{3}C_2^3 + \tfrac{8}{3}C_2C_3 + \tfrac{4}{3}C_3C_2 + \tfrac{4}{27}C_4\right)e_k^3 \\ + \left(4C_2C_4 - \tfrac{16}{3}C_2^2C_3 - 4C_2C_3C_2 + \tfrac{16}{3}C_2^4 - \tfrac{4}{3}C_3C_2^2 + \tfrac{8}{3}C_2^3 + \tfrac{8}{9}C_4C_2\right)e_k^4\Big] + O\left[e_k^5\right] \end{aligned} \tag{33}$$

The inverse of the expression above is as follows:

$$\begin{aligned} \left[F'\left(z^{(k)}\right)\right]^{-1} = \Big[I - \tfrac{2}{3}C_2 e_k + \left(-\tfrac{8}{9}C_2^2 - \tfrac{1}{3}C_3\right)e_k^2 + \left(\tfrac{112}{27}C_2^3 - \tfrac{22}{9}C_2C_3 - \tfrac{10}{9}C_3C_2 - \tfrac{4}{27}C_4\right)e_k^3 \\ + \Big(-\tfrac{316}{81}C_2C_4 + \tfrac{236}{27}C_2^2C_3 + \tfrac{140}{27}C_2C_3C_2 - \tfrac{416}{81}C_2^4 - \tfrac{32}{9}C_2^3 + \tfrac{68}{27}C_3C_2^2 - \tfrac{23}{9}C_2^3 - \tfrac{64}{81}C_4C_2 \\ - \tfrac{5}{81}C_5\Big)e_k^4\Big][F'(\xi)]^{-1} + O\left[e_k^5\right] \end{aligned} \tag{34}$$

Using Equations (29) and (33):

$$\begin{aligned}\left[F'\left(x^{(k)}\right)\right]^{-1}F'\left(z^{(k)}\right)&\\ =I-\tfrac{4}{3}C_2e_k+\left(4C_2^2-\tfrac{8}{3}C_3\right)e_k^2+\left(-\tfrac{32}{3}C_2^3+8C_2C_3+\tfrac{16}{3}C_3C_2-\tfrac{104}{27}C_4\right)e_k^3&\\ +\left(\tfrac{316}{27}C_2C_4-\tfrac{64}{3}C_2^2C_3-\tfrac{44}{3}C_2C_3C_2+\tfrac{64}{3}C_2^4-\tfrac{40}{3}C_3C_2^2+\tfrac{32}{3}C_3^2+\tfrac{56}{9}C_4C_2+\tfrac{16}{3}C_2^3-5C_5\right)e_k^4&\\ +O\left[e_k^5\right]&\end{aligned}\tag{35}$$

Using Equations (34) and (27):

$$\begin{aligned}\left[F'\left(z^{(k)}\right)\right]^{-1}F\left(x^{(k)}\right)&\\ =e_k+\tfrac{1}{3}C_2e_k^2+\left(\tfrac{2}{3}C_3-\tfrac{14}{9}C_2^2\right)e_k^3+\left(\tfrac{23}{27}C_4-\tfrac{28}{9}C_2C_3-\tfrac{13}{9}C_3C_2+\tfrac{88}{27}C_2^3\right)e_k^4&\\ +\left(\tfrac{76}{81}C_5-\tfrac{370}{81}C_2C_4+\tfrac{212}{27}C_2^2C_3-\tfrac{26}{9}C_3^2-\tfrac{80}{81}C_2^4+\tfrac{74}{27}C_2C_3C_2+\tfrac{38}{27}C_3C_2^2-\tfrac{76}{81}C_4C_2&\\ -\tfrac{32}{9}C_2^3\right)e_k^5+O\left[e_k^6\right]&\end{aligned}\tag{36}$$

Replacing $e_k = x^{(k)} - \xi$ again and using the above results in the second step of the iterative method described in Equation (24):

$$\begin{aligned}y^{(k)}=\xi+\left(\tfrac{1}{9}C_4-C_3C_2+\tfrac{7}{3}C_2^3\right)e_k^4&\\ +\left(\tfrac{329}{81}C_5-\tfrac{590}{459}C_2^2C_3-2C_3^2-\tfrac{478}{27}C_2^4+6C_2C_3C_2+\tfrac{20}{3}C_3C_2^2-\tfrac{20}{9}C_4C_2+\tfrac{130}{27}C_2^3\right)e_k^5+O\left[e_k^6\right]&\end{aligned}\tag{37}$$

From Equation (37), $y^{(k)} = \xi + e_{y^{(k)}}$, then

$$\begin{aligned}e_{y^{(k)}}=\left(\tfrac{1}{9}C_4-C_3C_2+\tfrac{7}{3}C_2^3\right)e_k^4&\\ +\left(\tfrac{329}{81}C_5-\tfrac{590}{459}C_2^2C_3-2C_3^2-\tfrac{478}{27}C_2^4+6C_2C_3C_2+\tfrac{20}{3}C_3C_2^2-\tfrac{20}{9}C_4C_2+\tfrac{130}{27}C_2^3\right)e_k^5+O\left[e_k^6\right]&\end{aligned}\tag{38}$$

The following development for the matrix $F\left(y^{(k)}\right)$ is obtained:

$$\begin{aligned}F\left(y^{(k)}\right)=F'(\xi)\Big[\left(\tfrac{1}{9}C_4-C_3C_2+\tfrac{7}{3}C_2^3\right)e_k^4&\\ ++\left(\tfrac{329}{81}C_5-\tfrac{590}{459}C_2^2C_3-2C_3^2-\tfrac{478}{27}C_2^4+6C_2C_3C_2+\tfrac{20}{3}C_3C_2^2-\tfrac{20}{9}C_4C_2+\tfrac{130}{27}C_2^3\right)e_k^5\Big]&\\ +O\left[e_k^6\right]&\end{aligned}\tag{39}$$

Applying Equations (29) and (39):

$$\begin{aligned}\left[F'\left(x^{(k)}\right)\right]^{-1}F\left(y^{(k)}\right)&\\ =\left(\tfrac{1}{9}C_4-C_3C_2+\tfrac{7}{3}C_2^3\right)e_k^4&\\ +\left(\tfrac{329}{81}C_5-\tfrac{590}{459}C_2^2C_3-2C_3^2-\tfrac{604}{27}C_2^4+8C_2C_3C_2+\tfrac{20}{3}C_3C_2^2-\tfrac{20}{9}C_4C_2-\tfrac{2}{9}C_2C_4\right)e_k^5+O\left[e_k^6\right]&\end{aligned}\tag{40}$$

Using Equations (37) and (40) in the last step of the iterative method described in Equation (24):

$$x^{(k+1)}=\xi+\left(\frac{14}{3}C_2^4-2C_2C_3C_2+\frac{2}{9}C_2C_4\right)e_k^5+O\left[e_k^6\right].\tag{41}$$

From Equation (41), the following error equation is obtained:

$$e^{(k+1)}=\left(\frac{14}{3}C_2^4-2C_2C_3C_2+\frac{2}{9}C_2C_4\right)e_k^5+O\left[e_k^6\right]\tag{42}$$

Therefore, the order of convergence is 5.

2.6. Discretization

For the discretization of the nonlinear convection–diffusion equation, a Crank–Nicolson-type method is used together with a regular rectangular mesh. The nodes of the mesh are $(x_i, t_j) = (a + ih, 0 + jk)$, $i = 0, 1, \ldots nx$ and $j = 0, 1, \ldots nt$, for the calculation of the spatial step; $h = \frac{b-a}{nx}$ where nx is the number of spatial sub-intervals. For the time step calculation $k = \frac{T_{max}}{nt}$, where T_{max} is the maximum simulation time and nt represents the number of time sub-intervals.

For the first time instant, $\frac{\partial u}{\partial t}$ is approximated by a progressive difference and central finite differences for $\frac{\partial u}{\partial x}$ and $\frac{\partial^2 u}{\partial x^2}$, using Equations (8) and (9), respectively, resulting in the following:

$$\frac{u_{i,j+1} - u_{i,j}}{k} - \not{k}\frac{u_{i+1,j} - 2u_{i,j} + u_{i-1,j}}{h^2} + u_{i,j}\frac{u_{i+1,j} - u_{i-1,j}}{2h} = 0, \tag{43}$$

Backward differences are used to approximate $\frac{\partial u}{\partial t}$ and central differences are used in $\frac{\partial u}{\partial x}$ y $\frac{\partial^2 u}{\partial x^2}$ for the second time instant t_{j+1}, then from Equation (1):

$$\frac{u_{i,j+1} - u_{i,j}}{k} - \not{k}\frac{u_{i+1,j+1} - 2u_{i,j+1} + u_{i-1,j+1}}{h^2} + u_{i,j+1}\frac{u_{i+1,j+1} - u_{i-1,j+1}}{2h} = 0. \tag{44}$$

After calculating the arithmetic mean between Equations (43) and (44), and simplifying, the resulting expression is as follows:

$$\begin{aligned} \tfrac{u_{i,j+1}}{k} - \tfrac{\not{k}}{2h^2}u_{i+1,j+1} + \tfrac{\not{k}}{h^2}u_{i,j+1} - \tfrac{\not{k}}{2h^2}u_{i-1,j+1} + \tfrac{\not{k}}{4h}u_{i,j+1}u_{i+1,j+1} \\ - \tfrac{\not{k}}{4h}u_{i,j+1}u_{i-1,j+1} = \left(\tfrac{1}{k} - \tfrac{\not{k}}{h^2}\right)u_{i,j} + \tfrac{\not{k}}{2h^2}u_{i+1,j} \\ + \tfrac{\not{k}}{2h^2}u_{i-1,j} - \tfrac{1}{4h}u_{i,j}u_{i+1,j} + \tfrac{1}{4h}u_{i,j}u_{i-1,j} \end{aligned} \tag{45}$$

For $i = 1, 2, \ldots nx - 1$; and $j = 1, 2, \ldots nt - 1$. The Equation (45) has a convergence order of two. By analyzing arbitrary values of j, the following system of nonlinear equations of size $(nx - 1) \times (nx - 1)$ is obtained:

$$\begin{cases} \tfrac{u_{1,j+1}}{k} - \tfrac{\not{k}}{2h^2}\,u_{2,j+1} + \tfrac{\not{k}}{h^2}\,u_{1,j+1} \\ \quad - \tfrac{\not{k}}{2h^2}\,u_{0,j+1} + \tfrac{\not{k}}{4h}\,u_{1,j+1}u_{2,j+1} \\ \quad - \tfrac{\not{k}}{4h}\,u_{1,j+1}u_{i-1,j+1} = \left(\tfrac{1}{k} - \tfrac{\not{k}}{h^2}\right)u_{1,j} \\ + \tfrac{\not{k}}{2h^2}\,u_{2,j} + \tfrac{\not{k}}{2h^2}\,u_{0,j} - \tfrac{1}{4h}\,u_{1,j}u_{2,j} + \tfrac{1}{4h}\,u_{1,j} \\ \qquad \vdots \\ \tfrac{u_{nx-1,j+1}}{k} - \tfrac{\not{k}}{2h^2}\,u_{nx,j+1} + \tfrac{\not{k}}{h^2}\,u_{nx-1,j+1} \\ \quad - \tfrac{\not{k}}{2h^2}\,u_{nx-2,j+1} + \tfrac{\not{k}}{4h}\,u_{nx-1,j+1}u_{nx,j+1} \\ - \tfrac{\not{k}}{4h}\,u_{nx-1,j+1}u_{nx-2,j+1} = \left(\tfrac{1}{k} - \tfrac{\not{k}}{h^2}\right)u_{nx-1,j} \\ \quad + \tfrac{\not{k}}{2h^2}\,u_{nx,j} + \tfrac{\not{k}}{2h^2}\,u_{nx-2,j} \\ \quad - \tfrac{1}{4h}\,u_{nx-1,j}u_{nx,j} + \tfrac{1}{4h}\,u_{nx-1,j}u_{nx-2,j} \end{cases} \tag{46}$$

2.7. MATLAB Function

A MATLAB function was developed to present a solution to the nonlinear convection–diffusion problem using the Crank–Nicolson time discretization method in combination with the NJN iterative scheme. The key steps of the implemented algorithm are as follows:

Definition of Variables and Initial Parameters: The spatial domain was uniformly discretized with a step size h, which depends on the number of spatial points, while the temporal domain was discretized with a step size k, determined by the number of time

steps. The computed solutions were stored in matrices corresponding to each spatial and temporal point.

1. Initial and Boundary Conditions: Boundary conditions were imposed at the extremities of the spatial domain, and the initial condition was specified at all spatial points at the initial time. These conditions play a fundamental role in ensuring the correct temporal evolution of the solution.
2. Computation of Discretization Coefficients: The coefficients required for the finite difference discretization of the convection–diffusion equation were defined. These coefficients were computed as functions of the system parameters, including the spatial and temporal step sizes and the diffusion parameter $\mathscr{k}$.
3. Initialization of the Iterative Method: An initial approximation of the solution was provided to initiate the iterative process. The NJN iterative scheme was then employed to solve the nonlinear system arising from the discretization of the convection–diffusion equation.
4. Solution of the Nonlinear System and Solution Update: At each time step, the nonlinear system was solved using the NJN iterative method. This procedure was repeated until the predefined tolerance, or the maximum number of iterations was reached.

3. Results

To check the efficiency of the method, the performance was analyzed in Matlab R2017b using variable precision arithmetic with 2000 digits of mantissa. The PC used is an 11th Gen Intel(R) Core (TM) i5-1135G7 with 2.42 GHz.

Table 1 shows the number of iterations, the residual errors $\left\|x^{(k+1)}-x^{(k)}\right\|$, $\left\|F\left(x^{(k+1)}\right)\right\|$ of the last iteration, the stopping criterium $\left\|x^{(k+1)}-x^{(k)}\right\|<10^{-300}$ or $\left\|F\left(x^{(k+1)}\right)\right\|<10^{-300}$ the CPU time, which is the mean of 10 executions of the program measured in seconds, the convergency order, which is an approximate of the computational order of convergence called ACOC using the formula:

$$p \approx ACOC = \frac{ln\left(\left\|x^{(k+1)}-x^{(k)}\right\| / \left\|x^{(k)}-x^{(k-1)}\right\|\right)}{ln\left(\left\|x^{(k)}-x^{(k-1)}\right\| / \left\|x^{(k-1)}-x^{(k-2)}\right\|\right)}, \tag{47}$$

Table 1. Comparison of various iterative methods for solving systems of nonlinear equations.

Example	Method	Iterations	$\left\|x^{(k+1)}-x^{(k)}\right\|$	$\left\|F\left(x^{(k+1)}\right)\right\|$	*ACOC*	CPU Time
E1	Newton	9	1.1412×10^{-397}	4.8016×10^{-795}	2	0.36055
	MSNF-2	6	1.4497×10^{-1006}	1.079×10^{-2007}	4	0.46249
	NJN	5	2.22×10^{-639}	1.079×10^{-2007}	5	0.57177
	SHM5	5	7.67×10^{-455}	1.079×10^{-2007}	5	0.47859
	O6	5	1.9666×10^{-1596}	1.079×10^{-2007}	6	0.55804
E2	Newton	10	5.1105×10^{-317}	1.8081×10^{-633}	2	0.27985
	MSNF-2	10	6.8426×10^{-320}	1.1372×10^{-457}	1	0.50723
	NJN	7	2.6144×10^{-585}	1.4581×10^{-1204}	2	0.44107
	SHM5	6	1.0218×10^{-929}	0.0	4	0.43012
	O6	6	2.6253×10^{-1065}	1.2005×10^{-3123}	4	0.45489
E3	Newton	11	2.5778×10^{-497}	6.7394×10^{-994}	2	0.75281
	MSNF-2	6	1.8185×10^{-382}	4.5723×10^{-1527}	4	0.84646
	NJN	6	4.1971×10^{-969}	0.0	5	1.10354
	SHM5	5	2.1877×10^{-426}	5.395×10^{-2008}	5	0.87297
	O6	6	6.9962×10^{-1696}	6.0138×10^{-2008}	6	1.12233

In Equation (47), $\| \cdot \|$ refers to the norm of the difference between iterates. The nonlinear systems of equations presented in Table 2 are solved using the designed NJN method, comparing it with the Newton, MSNF-2, SHM5, and O6 methods.

Table 2. Nonlinear equation systems, roots, and starting vector.

	Example	Exact Root	Starting Vector
E1	$f = e^{x_1}e^{x_2} + x_1 cos(x_2)$ $g = x_1 + x_2 - 1$	$\xi \approx \begin{pmatrix} 3.47063096 \\ -2.47163096 \end{pmatrix}$	$x^{(0)} = (3, -2)^T$
E2	$f = x_1 + e^{x_2} - cos(x_2)$ $g = 3x_1 - x_2 - sin(x_2)$	$\xi \approx \begin{pmatrix} 0 \\ 0 \end{pmatrix}$	$x^{(0)} = (0.5, 0.5)^T$
E3	$f = cos(x_2) - sin(x_1)$ $g = {x_3}^{x_1} - 1/\ x_2$ $h = e^{x_1} - {x_3}^2$	$\xi \approx \begin{pmatrix} 0.90956949 \\ 0.66122683 \\ 0.15783414 \end{pmatrix}$	$x^{(0)} = (0.8, 0.5, 1.4)^T$

In Table 1, the theoretical order of convergence matches the computational order of convergence (ACOC), for example, E2 and E3. When comparing our method with lower-order methods, such as Newton and MSNF-2, we see that fewer iterations are required with lower residual error values. However, the computational time is slightly higher.

When comparing NJN with another method of the same order of convergence, such as SHM5, more iterations are necessary, but the results show lower residual error values and slightly higher computational time. Finally, when comparing our method with a higher-order convergence method, we see that our method requires the same number of iterations, as observed in examples E1 and E3. The results show higher residual error values and less computational time.

In example E2, the computational order of convergence of the MSNF-2 method and our method do not reach the theoretical order of convergence, possibly due to the initial approximation value chosen.

Figure 1 shows the convergence curves obtained for the NJN method applied to the systems of equations in Table 2.

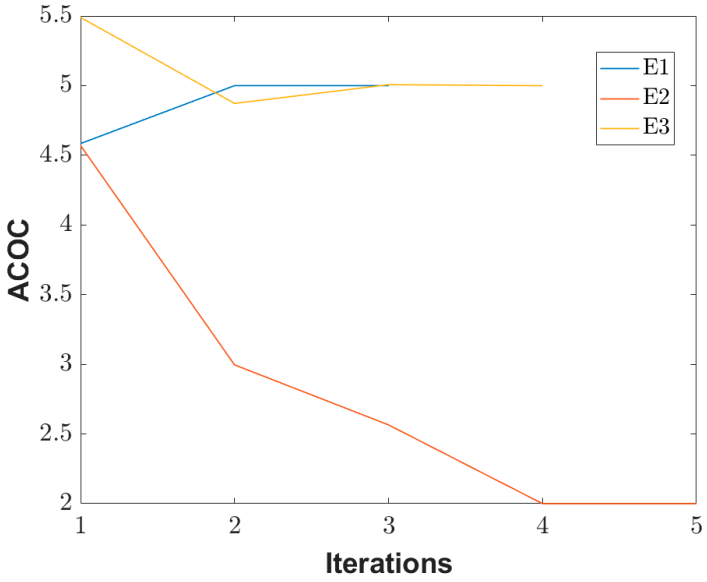


Figure 1. Convergence curves are obtained from systems in Table 2.

The absence of oscillations in convergence, as observed in Figure 1, along with the regularity in ACOC and the controlled decrease in error, indicates that NJN is not sensitive

to rounding errors or perturbations in the calculations. Consequently, the method is numerically stable and maintains its expected behavior across different scenarios.

A script was created in Matlab R2017b with a function that allows the approximate solution of the nonlinear convection–diffusion equation to be obtained once it has been discretized. The program solves nt nonlinear type systems of size $(nx-1)\times(nx-1)$ using the method developed in this research.

The Cole–Hopf transformation allows obtaining the exact solution of the nonlinear convection–diffusion equation by means of the expression described in Equation (5) and using 100 Fourier coefficients and $\kappa = 10$ with the following initial condition:

$$u(x,0) = \sin(\pi x),\ 0 < x < 1, \tag{48}$$

and Homogeneous Dirichlet-type boundary conditions:

$$u(0,t) = 0,\ u(1,t) = 0,\ t > 0. \tag{49}$$

Using the new Newton–Jarratt-modified iterative method and the exact solution, variations are made to different parameters of both the nonlinear convection–diffusion equation and those used in the discretization.

Table 3 shows the difference between the exact and approximate solution for various values of the spatial variable for a time of $t = 0.1$ s. An equal number of spatial and temporal nodes are used; in this case, 1000 nodes are chosen and the coefficient $\kappa = 10$.

Table 3. Comparison of solutions using the Newton–Jarratt-modified iterative method for $\kappa = 10$, $t = 0.1$ s, $nt = 1000$, and $nx = 1000$.

x	Exact Solution	Approximate Solution	Error
0	0.00000	0.00000	0.0000
0.05	8.0910×10^{-6}	8.0264×10^{-6}	6.4592×10^{-8}
0.10	1.5983×10^{-5}	1.5855×10^{-5}	1.2759×10^{-7}
0.15	2.3481×10^{-5}	2.3294×10^{-5}	1.8745×10^{-7}
0.20	3.0401×10^{-5}	3.0158×10^{-5}	2.4270×10^{-7}
0.25	3.6573×10^{-5}	3.6281×10^{-5}	2.9197×10^{-7}
0.30	4.1844×10^{-5}	4.1510×10^{-5}	3.3405×10^{-7}
0.35	4.6084×10^{-5}	4.5716×10^{-5}	3.6790×10^{-7}
0.40	4.9190×10^{-5}	4.8797×10^{-5}	3.9270×10^{-7}
0.45	5.1085×10^{-5}	5.0677×10^{-5}	4.0782×10^{-7}
0.50	5.1722×10^{-5}	5.1309×10^{-5}	4.1291×10^{-7}
0.55	5.1085×10^{-5}	5.0677×10^{-5}	4.0782×10^{-7}
0.60	4.9190×10^{-5}	4.8797×10^{-5}	3.9270×10^{-7}
0.65	4.6084×10^{-5}	4.5716×10^{-5}	3.6790×10^{-7}
0.70	4.1844×10^{-5}	4.1510×10^{-5}	3.3405×10^{-7}
0.75	3.6573×10^{-5}	3.6281×10^{-5}	2.9197×10^{-7}
0.80	3.0401×10^{-5}	3.0158×10^{-5}	2.4270×10^{-7}
0.85	2.3481×10^{-5}	2.3294×10^{-5}	1.8746×10^{-7}
0.90	1.5983×10^{-5}	1.5855×10^{-5}	1.2760×10^{-7}
0.95	8.0910×10^{-6}	8.0264×10^{-6}	6.4593×10^{-8}
0.10	0.0000	0.0000	0.0000

The maximum absolute error reported in Table 3 is 4.1291×10^{-7}. A graph was plotted for the exact and approximate solution with the same simulation parameters from Table 3 to compare the error between the solutions. The result is depicted in Figure 2, where the numerical and reference curves align so closely that they are almost indistinguishable.

This visual comparison highlights the high accuracy achieved in the numerical solution, showcasing its effectiveness in closely approximating the system's expected behavior under the given conditions.

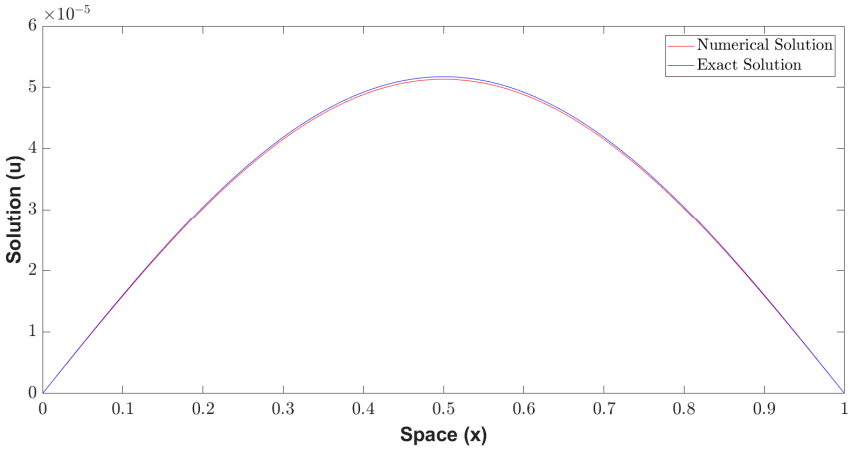


Figure 2. Numerical and exact solutions of the nonlinear convection–diffusion equation for $t = 0.1$ s, $nt = nx = 1000$.

Keeping the value of the coefficient $\mathcal{K}$ constant, the number of spatial and temporal nodes decreased, now using 500 nodes in both cases. Several values of the spatial variable are used for a time $t = 0.1$ s; in the same way, the absolute error is calculated obtaining the results of Table 4.

Table 4. Comparison of solutions using Newton–Jarratt-modified iterative method for $\mathcal{K} = 10$, $t = 0.1$ s, $nt = 500$, and $nx = 500$.

x	Exact Solution	Approximate Solution	Error
0	0.00000	0.0000	0.0000
0.05	8.0910×10^{-6}	7.8346×10^{-6}	2.5640×10^{-7}
0.10	1.5983×10^{-5}	1.5476×10^{-5}	5.0649×10^{-7}
0.15	2.3481×10^{-5}	2.2737×10^{-5}	7.4411×10^{-7}
0.20	3.0401×10^{-5}	2.9438×10^{-5}	9.6340×10^{-7}
0.25	3.6573×10^{-5}	3.5414×10^{-5}	1.1590×10^{-6}
0.30	$4,1844 \times 10^{-5}$	4.0518×10^{-5}	1.3260×10^{-6}
0.35	4.6084×10^{-5}	4.4624×10^{-5}	1.4604×10^{-6}
0.40	4.9190×10^{-5}	4.7631×10^{-5}	1.5588×10^{-6}
0.45	5.1085×10^{-5}	4.9466×10^{-5}	1.6189×10^{-6}
0.50	5.1722×10^{-5}	5.0083×10^{-5}	1.6390×10^{-6}
0.55	5.1085×10^{-5}	4.9466×10^{-5}	1.6189×10^{-6}
0.60	4.9190×10^{-5}	4.7631×10^{-5}	1.5588×10^{-6}
0.65	4.6084×10^{-5}	4.4624×10^{-5}	1.4604×10^{-6}
0.70	4.1844×10^{-5}	4.0518×10^{-5}	1.3260×10^{-6}
0.75	3.6573×10^{-5}	3.5414×10^{-5}	1.1590×10^{-6}
0.80	3.0401×10^{-5}	2.9438×10^{-5}	9.6341×10^{-6}
0.85	2.3481×10^{-5}	2.2737×10^{-5}	7.4411×10^{-7}
0.90	1.5983×10^{-5}	1.5476×10^{-5}	5.0649×10^{-7}
0.95	8.0910×10^{-6}	7.8346×10^{-6}	2.5640×10^{-7}
0.10	0.0000	0.0000	0.0000

Table 4 indicates that the maximum absolute error is 1.6390×10^{-6}, which is larger than the 4.1291×10^{-7} maximum error reported in Table 3. Figure 3 depicts the updated variation between the approximate and exact solutions, revealing a more noticeable difference

compared to Figure 2. This supports the maximum absolute error values in Table 4, offering a visual representation of the variations between the two solutions.

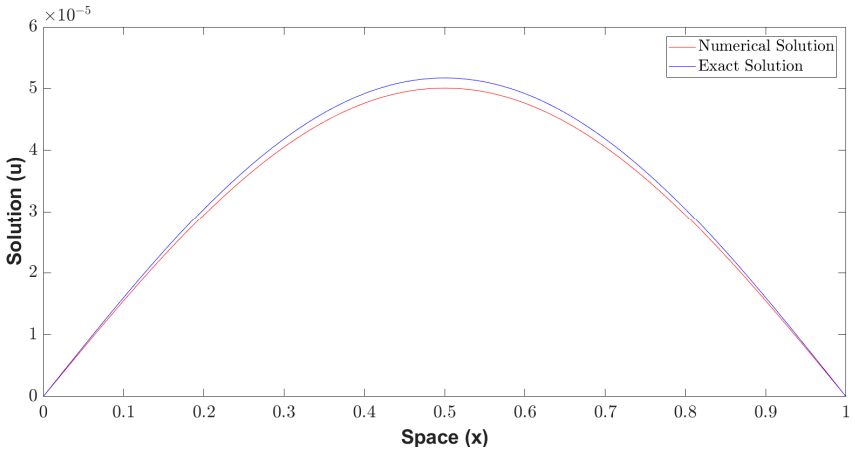


Figure 3. Numerical and exact solutions of the nonlinear convection–diffusion equation for $t = 0.1$ s, $nt = nx = 500$.

The simulation is now repeated with an even further reduction in the number of spatial and temporal nodes, setting both to 100. This adjustment is made to continue examining the error behavior [33]. As with the previous simulations, the corresponding results are presented in Table 5 for analysis.

Table 5. Comparison of solutions using the Newton–Jarratt-modified iterative method for $\mathscr{k} = 10$, $t = 0.1$ s, $nt = 100$, and $nx = 100$.

x	Exact Solution	Approximate Solution	Error
0	0.00000	0.0000	0.0000
0.05	8.0910×10^{-6}	3.2030×10^{-6}	4.8881×10^{-6}
0.10	1.5983×10^{-5}	6.3210×10^{-6}	9.6619×10^{-6}
0.15	2.3481×10^{-5}	9.2722×10^{-6}	1.4209×10^{-5}
0.20	3.0401×10^{-5}	1.1981×10^{-5}	1.8420×10^{-5}
0.25	3.6573×10^{-5}	1.4380×10^{-5}	2.2193×10^{-5}
0.30	4.1844×10^{-5}	1.6411×10^{-5}	2.5433×10^{-5}
0.35	4.6084×10^{-5}	1.8029×10^{-5}	2.8055×10^{-5}
0.40	4.9190×10^{-5}	1.9198×10^{-5}	2.9992×10^{-5}
0.45	5.1085×10^{-5}	1.9896×10^{-5}	3.1188×10^{-5}
0.50	5.1722×10^{-5}	2.0110×10^{-5}	3.1612×10^{-5}
0.55	5.1085×10^{-5}	1.9838×10^{-5}	3.1247×10^{-5}
0.60	4.9190×10^{-5}	1.9087×10^{-5}	3.0103×10^{-5}
0.65	4.6084×10^{-5}	1.7876×10^{-5}	2.8209×10^{-5}
0.70	4.1844×10^{-5}	1.6231×10^{-5}	2.5612×10^{-5}
0.75	3.6573×10^{-5}	1.4191×10^{-5}	2.2381×10^{-5}
0.80	3.0401×10^{-5}	1.1803×10^{-5}	1.8599×10^{-5}
0.85	2.3481×10^{-5}	9.1211×10^{-6}	1.4360×10^{-5}
0.90	1.5983×10^{-5}	6.2115×10^{-6}	9.7714×10^{-6}
0.95	8.0910×10^{-6}	3.1455×10^{-6}	4.9456×10^{-6}
0.10	0.0000	0.0000	0.0000

When comparing the exact and approximate solutions using 100 temporal and spatial nodes, the maximum absolute error obtained is 3.1612×10^{-5}; this value is higher than the ones obtained in Tables 3 and 4. In Figure 4, it is observed that unlike the graphs in Figures 2 and 3, there is a more marked variation between the exact solution and the

numerical (approximate) solution. For subsequent simulations, it is recommended to use as many spatial and temporal nodes as possible to avoid generating significant errors.

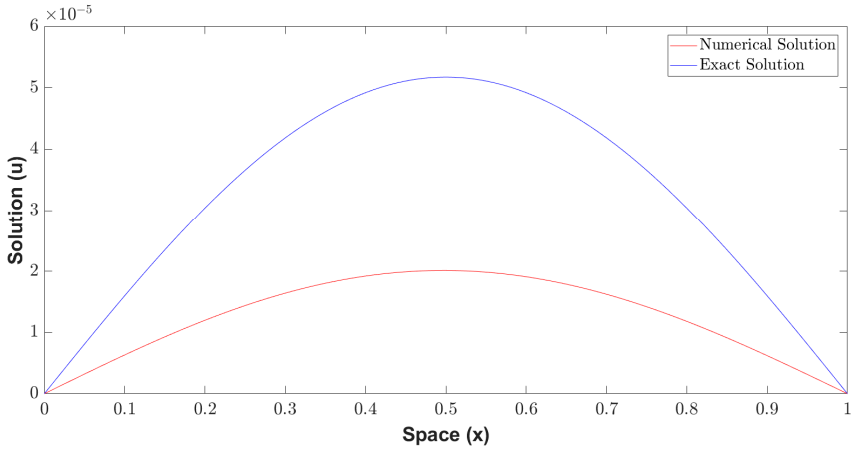


Figure 4. Numerical and exact solutions of the nonlinear convection–diffusion equation for $t = 0.1$ s, $nt = nx = 100$.

An additional simulation was conducted using 1000 spatial and temporal nodes, and the viscosity coefficient κ was adjusted to 0.1. The outcomes of this simulation are presented in Table 6, where an absolute error of 7.6580×10^{-7} was observed.

Table 6. Comparison of solutions using the Newton–Jarratt-modified iterative method for $\kappa = 0.1$, $t = 0.1$ s, $nt = 1000$, and $nx = 1000$.

x	Exact Solution	Approximate Solution	Error
0	0.0000	0.0000	0.0000
0.05	1.1241×10^{-1}	1.1241×10^{-1}	1.4197×10^{-7}
0.10	2.2345×10^{-1}	2.2345×10^{-1}	2.5930×10^{-7}
0.15	3.3173×10^{-1}	3.3173×10^{-1}	3.3020×10^{-7}
0.20	4.3580×10^{-1}	4.3580×10^{-1}	3.3851×10^{-7}
0.25	5.3414×10^{-1}	5.3414×10^{-1}	2.7606×10^{-7}
0.30	6.2512×10^{-1}	6.2512×10^{-1}	1.4465×10^{-7}
0.35	7.0696×10^{-1}	7.0696×10^{-1}	4.2865×10^{-8}
0.40	7.7772×10^{-1}	7.7772×10^{-1}	2.6268×10^{-7}
0.45	8.3528×10^{-1}	8.3528×10^{-1}	4.8178×10^{-7}
0.50	8.7728×10^{-1}	8.7728×10^{-1}	6.6191×10^{-7}
0.55	9.0118×10^{-1}	9.0118×10^{-1}	7.6580×10^{-7}
0.60	9.0425×10^{-1}	9.0425×10^{-1}	7.6569×10^{-7}
0.65	8.8370×10^{-1}	8.8371×10^{-1}	6.5266×10^{-7}
0.70	8.3692×10^{-1}	8.3692×10^{-1}	4.4415×10^{-7}
0.75	7.6180×10^{-1}	7.6180×10^{-1}	1.8564×10^{-7}
0.80	6.5731×10^{-1}	6.5731×10^{-1}	5.7410×10^{-8}
0.85	5.2421×10^{-1}	5.2421×10^{-1}	2.1844×10^{-7}
0.90	3.6575×10^{-1}	3.6575×10^{-1}	2.5451×10^{-7}
0.95	1.8809×10^{-1}	1.8809×10^{-1}	1.6596×10^{-7}
0.10	0.0000	0.0000	0.0000

In addition, Figure 5 provides a graphical representation of these results, visually illustrating the error and overall simulation performance under the updated conditions.

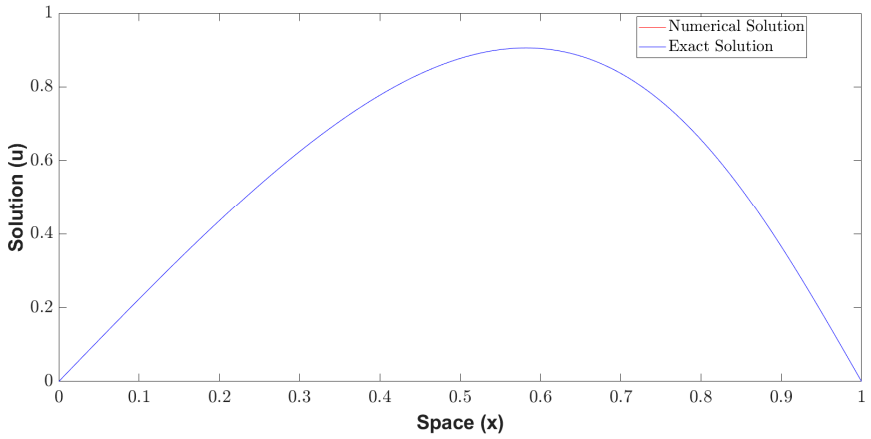


Figure 5. Numerical and exact solutions of the nonlinear convection–diffusion equation for $\mathscr{k} = 0.1$, $t = 0.1$ s, $nt = nx = 100$.

The nonlinear convection–diffusion equation incorporates spatial and temporal variables, making its behavior dependent on these dimensions. To facilitate its physical interpretation and provide insight into how changes in the viscous diffusion coefficient influence the solution, Figure 6 includes a visual representation. This graphical depiction is constructed using specific parameters detailed below, allowing for a clear observation of the relationship between the equation's components and the resulting solution. This representation highlights the equation's dynamics and the impact of varying diffusion constants.

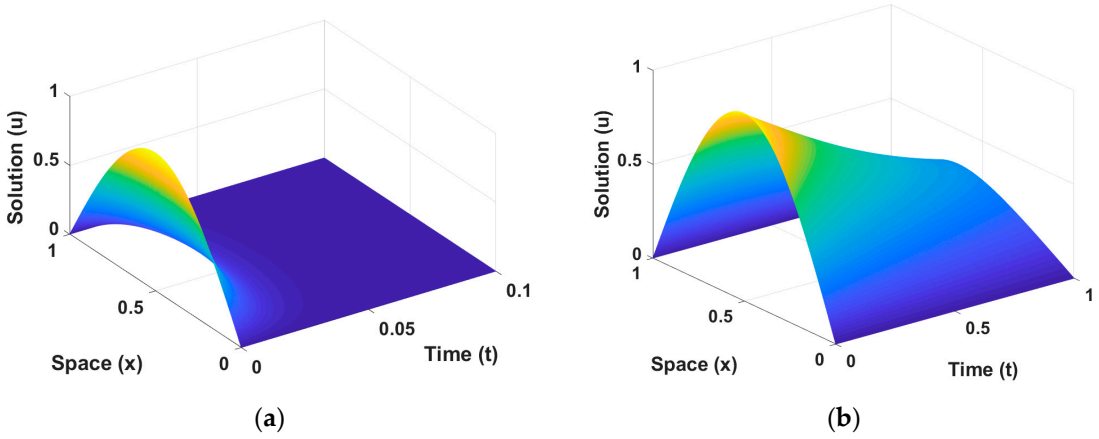


Figure 6. Numerical solution of the nonlinear convection–diffusion equation as a function of space and time with $nt = nx = 1000$ for (**a**) $T_{max} = 0.1$ s, $\mathscr{k} = 10$; (**b**) $T_{max} = 1$ s, $\mathscr{k} = 0.1$.

By comparing both images, it becomes evident how the viscous diffusion coefficient influences the solution of the nonlinear convection–diffusion equation. Specifically, when the value of this coefficient increases, the solution exhibits a noticeable tendency to diminish, eventually approaching zero over a shorter time interval. This behavior explains the critical role of the diffusion coefficient in determining the equation's dynamics and underscores its significant impact on the temporal evolution and attenuation of the solution in the modeled system.

Figure 7 presents error progression curves. The highest error among all x values is taken for each time step within the interval from 0 to 0.1 s.

In Figure 7a, an equal number of spatial and temporal nodes is used; in this case, 100 nodes are chosen, and the coefficient $\mathscr{k} = 10$.

In Figure 7b, an equal number of spatial and temporal nodes is used; in this case, 1000 nodes are chosen, and the coefficient $\mathscr{k} = 10$.

In Figure 7c, an equal number of spatial and temporal nodes is used; in this case, 1000 nodes are chosen, and the coefficient $\mathscr{k} = 0.1$

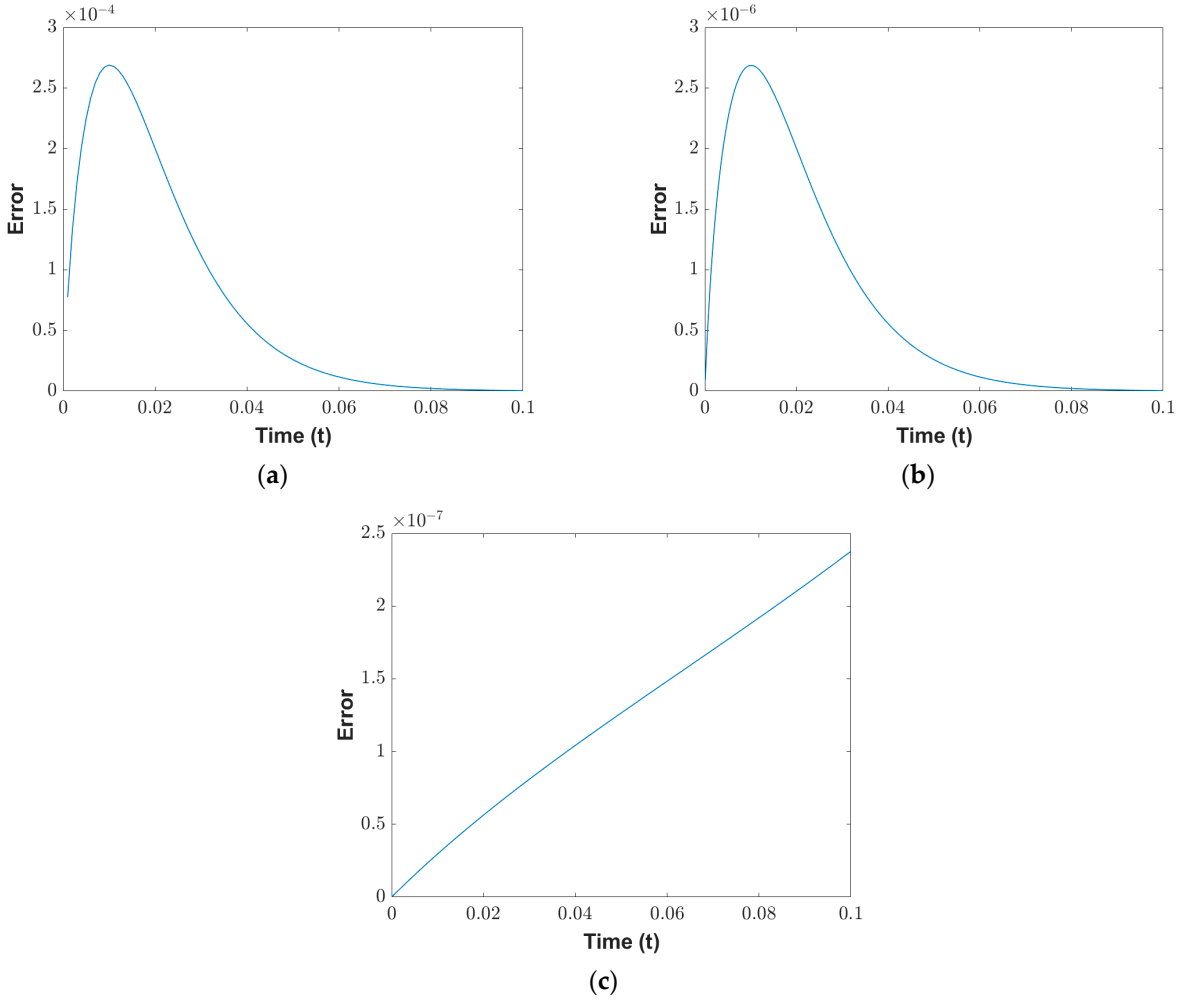


Figure 7. Error progression curve: (**a**) $\mathscr{k} = 10$, $nt = nx = 100$; (**b**) $\mathscr{k} = 10$, $nt = nx = 1000$; (**c**) $\mathscr{k} = 0.1$, $nt = nx = 1000$.

When varying the viscous diffusion coefficient, it is observed that for large values of $\mathscr{k}$, as in Figure 7a,b, the Péclet number is small. In this case, the diffusion term in Equation (1) dominates, effectively linearizing the equation, which leads to a decrease in error progression.

Additionally, when $\mathscr{k}$ is small, as in Figure 7c, the Péclet number is greater than one, meaning the convection term in Equation (1) prevails. This increases error progression since the diffusion term is not strong enough to linearize the equation.

As the boundary condition propagates through the field, the velocity decreases, causing the convection term, which introduces the nonlinear structure of Equation (1), to tend towards zero. Consequently, the contribution of the diffusion term dominates, effectively linearizing the equation. Numerically, the error also tends to be zero in this region, as observed in Figure 7a,b.

The analysis of Figure 7 shows that the method used to discretize and solve the nonlinear convection–diffusion equation is stable and convergent when increasing the number of nodes. The maximum error decreases significantly when increasing from 100 nodes (Figure 7a) to 1000 nodes (Figure 7b), reducing by two orders of magnitude without signs of numerical instability. However, when the coefficient $\mathscr{k}$ decreases from 10 to 0.1 (Figure 7c), the error exhibits an increasing trend over time, suggesting that this behavior

may be related to the Péclet number, which increases as $\mathscr{k}$ decreases, causing convection to dominate over diffusion and potentially generating instabilities if the numerical scheme does not handle the advective terms properly.

4. Discussion

This research introduces a novel iterative method with a convergence order of five. The convergence order was rigorously demonstrated using Taylor series expansions, which facilitated the derivation of the error equation to validate the hypothesis.

The results confirm that the NJN method is stable, with a high order of convergence ($p \approx 5$), minimal residual errors, and efficiency comparable to higher-order methods. This supports the method's robustness as a reliable alternative for solving nonlinear systems of equations.

The exact solution of the nonlinear convection–diffusion equation, obtained using the Cole–Hopf transformation, served as a benchmark for evaluating the approximate solution derived from the proposed method. Tables 3–5 compare the exact and approximate solutions. The results demonstrate that this approach achieves a strong approximation, making it suitable for addressing systems of nonlinear equations commonly encountered in various physical and mathematical phenomena.

Moreover, the research findings reveal that increasing the number of spatial and temporal nodes significantly improves the accuracy of the solution. This relationship is evident in the decreasing error values displayed in Figures 2–4. The visual representation of these figures underscores how higher nodal resolution reduces discrepancies between the exact and approximate solutions.

The iterative method's ability to balance computational efficiency with high accuracy further enhances its applicability in solving nonlinear systems. This makes it a valuable tool for researchers and practitioners working with problems modeled by the nonlinear convection–diffusion equation and other nonlinear equations. By integrating a structured mathematical framework with computational techniques, the proposed method demonstrates its potential for advancing numerical solutions in scientific and engineering disciplines.

Variations in the viscous diffusion coefficient influence the solution's accuracy, where lower values typically result in a higher error. However, the increase in error associated with smaller diffusion coefficients can be effectively mitigated by increasing the number of spatial and temporal nodes in the simulation. By refining the discretization, the impact of the error is reduced, allowing for a more precise approximation of the solution. This approach demonstrates the importance of balancing diffusion coefficient values with adequate nodal resolution to achieve accurate computational results.

Author Contributions: Conceptualization, methodology, S.Q.; validation, W.P.; investigation, software, N.O.; writing—original draft preparation, H.C.; writing—review and editing, G.Y.; supervision, M.Q. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by Universidad de las Fuerzas Armadas with $3000 USD and Universidad UTE with $3000 USD.

Data Availability Statement: Data are contained within the article.

Acknowledgments: This paper presents partial results of the project among Universidad de las Fuerzas Armadas (ESPE) and Universidad UTE.

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

References

1. Parada, H. Stability Analysis of a Korteweg-de Vries Equation with Saturated Boundary Feedback. *IFAC-PapersOnLine* **2022**, *55*, 1–6. [CrossRef]
2. Miura, T.-H. Singular Limit Problem for the Navier–Stokes Equations in a Curved Thin Domain. *Adv. Stud. Pure Math.* **2020**, *85*, 291–302. [CrossRef]
3. Sabermahani, S.; Ordokhani, Y. A Numerical Technique for Solving Fractional Benjamin-Bona-Mahony-Burgers Equations with Bibliometric Analysis. In *Fractional Order Systems and Applications in Engineering*; Academic Press: Cambridge, MA, USA, 2023; pp. 93–108.
4. Guo, Q.; Duan, L. Infinitely Many Solutions for the Fractional Nonlinear Schrödinger Equations of a New Type. *J. Partial. Differ. Equ.* **2022**, *35*, 259–280. [CrossRef]
5. Darehmiraki, M.; Rezazadeh, A.; Ahmadian, A.; Salahshour, S. An Interpolation Method for the Optimal Control Problem Governed by the Elliptic Convection–Diffusion Equation. *Numer. Methods Partial. Differ. Equ.* **2022**, *38*, 137–159. [CrossRef]
6. Yan, X.; Haiying, W.; Weiping, Z.; Zhanjun, X. Comparison between High Order Schemes Related Convection Diffusion of Navier-Stokes Equations. *Procedia Eng.* **2015**, *99*, 628–633. [CrossRef]
7. Brizitskii, R.V.; Saritskaya, Z.Y. Boundary Value and Extremal Problems for the Nonlinear Convection-Diffusion-Reaction Equation. *Sib. Electron. Math. Rep.* **2015**, *15*, 447–456. [CrossRef]
8. Aswin, V.S.; Awasthi, A.; Anu, C. A Comparative Study of Numerical Schemes for Convection-Diffusion Equation. *Procedia Eng.* **2015**, *127*, 621–627. [CrossRef]
9. Liu, B.; Zhu, L.; Yang, Z.; He, G. Quantum Implementation of Numerical Methods for Convection-Diffusion Equations: Toward Computational Fluid Dynamics. *Commun. Comput. Phys.* **2023**, *33*, 425–451. [CrossRef]
10. Garzón-Alvarado, D.A.; Galeano, C.H.; Mantilla, J.M. Computational Examples of Reaction-Convection-Diffusion Equations Solution under the Influence of Fluid Flow: First Example. *Appl. Math. Model.* **2012**, *36*, 5029–5045. [CrossRef]
11. Lima, S.A.; Kamrujjaman, M.D.; Islam, M.S. Numerical Solution of Convection-Diffusion-Reaction Equations by a Finite Element Method with Error Correlation. *AIP Adv.* **2021**, *11*, 085225. [CrossRef]
12. Nouri, Z.; Bendaas, S.; Kadem, H.E. N Wave and Periodic Wave Solutions for Burgers Equations. *Int. J. Anal. Appl.* **2020**, *18*, 304–318. [CrossRef]
13. Abd-El-Malek, M.B.; Hassan, H.S. Solution of Burgers' Equation via Lie-Group Analysis. In Proceedings of the 3rd Chaotic Modeling and Simulation International Conference, CHAOS, Chania, Greece, 1–4 June 2010.
14. Kurt, A.; Çenesiz, Y.; Tasbozan, O. On the Solution of Burgers' Equation with the New Fractional Derivative. *Open Phys.* **2015**, *13*, 355–360. [CrossRef]
15. Petchamé-Guerrero, J.; Carrera, J. An Improved Scheme for the Finite Difference Approximation of the Advective Term in the Heat or Solute Transport Equations. *Transp. Porous. Media* **2024**, *151*, 2795–2817. [CrossRef]
16. Mukmin, Y.; Wagianto, E.H.; Maryanto, A.; Suryadi, D.; Irwan, R.H. Drilling Best Practice for Offshore Cutting Discharge: Environmental Impact Assessment and Drill Cutting Dispersion Simulation in Water—PHE WMO Study Case. In Proceedings of the SPE/IADC Middle East Drilling Technology Conference and Exhibition, Abu Dhabi, United Arab Emirates, 23–25 May 2023.
17. Grigor'Ev, M.M. A Boundary Element Method for the Solution of Convective Diffusion and Burgers' Equations. *Int. J. Numer. Methods Heat Fluid Flow* **1994**, *4*, 527–552. [CrossRef]
18. Fajie Wang, F.W.; Chia-Ming Fan, C.-M.F.; Chuanzeng Zhang, C.Z.; Ji Lin, J.L. A Localized Space-Time Method of Fundamental Solutions for Diffusion and Convection-Diffusion Problems. *Adv. Appl. Math. Mech.* **2020**, *12*, 940–958. [CrossRef]
19. Wang, F.; Zhao, Q.; Chen, Z.; Fan, C.-M. Localized Chebyshev Collocation Method for Solving Elliptic Partial Differential Equations in Arbitrary 2D Domains. *Appl. Math. Comput.* **2021**, *397*, 125903. [CrossRef]
20. Darehmiraki, M.; Rezazadeh, A. A New Solution for Optimal Control of Fractional Convection–Reaction–Diffusion Equation Using Rational Barycentric Interpolation. *Bull. Iran. Math. Soc.* **2020**, *46*, 1307–1340. [CrossRef]
21. Liu, H.; Huang, J.; Zhang, W.; Ma, Y. Meshfree Approach for Solving Multi-Dimensional Systems of Fredholm Integral Equations via Barycentric Lagrange Interpolation. *Appl. Math. Comput.* **2019**, *346*, 295–304. [CrossRef]
22. Cordero, A.; Franques, A.; Torregrosa, J.R. Numerical Solution of Turbulence Problems by Solving Burger' Equation. *Algorithms* **2015**, *8*, 224–233. [CrossRef]
23. Hopf, E. The Partial Differential Equation Ut + Uux = Mxx. *Commun. Pure Appl. Math.* **1950**, *3*, 201–230. [CrossRef]
24. Cole, J.D. On a Quasi-Linear Parabolic Equation Occurring in Aerodynamics. *Q. Appl. Math.* **1951**, *9*, 225–236.
25. Sander, G.C. Exact Solutions to Nonlinear Diffusion-Convection Problems on Finite Domains. *J. Aust. Math. Society. Ser. B Appl. Math.* **1992**, *33*, 384–401. [CrossRef]
26. Cordero, A.; Rojas-Hiciano, R.V.; Torregrosa, J.R.; Vassileva, M.P. Maximally Efficient Damped Composed Newton-Type Methods to Solve Nonlinear Systems of Equations. *Appl. Math. Comput.* **2025**, *492*, 129231. [CrossRef]
27. Capdevila, R.R.; Cordero, A.; Torregrosa, J.R. A New Three-Step Class of Iterative Methods for Solving Nonlinear Systems. *Mathematics* **2019**, *7*, 1221. [CrossRef]

28. Kuo, C.-K.; Lee, S.-Y. A New Exact Solution of Burgers' Equation with Linearized Solution. *Math. Probl. Eng.* **2015**, *2015*, 414808. [CrossRef]
29. Sihwail, R.; Said Solaiman, O.; Zainol Ariffin, K.A. New Robust Hybrid Jarratt-Butterfly Optimization Algorithm for Nonlinear Models. *J. King Saud Univ. Comput. Inf. Sci.* **2022**, *34*, 8207–8220. [CrossRef]
30. Junjua, M.-D.; Akram, S.; Yasmin, N.; Zafar, F. A New Jarratt-Type Fourth-Order Method for Solving System of Nonlinear Equations and Applications. *J. Appl. Math.* **2015**, *2015*, 805278. [CrossRef]
31. Sharma, J.R.; Gupta, P. An Efficient Fifth Order Method for Solving Systems of Nonlinear Equations. *Comput. Math. Appl.* **2014**, *67*, 591–601. [CrossRef]
32. Padilla, J.J.; Chicharro, F.I.; Cordero, A.; Hernández-Díaz, A.M.; Torregrosa, J.R. A Class of Efficient Sixth-Order Iterative Methods for Solving the Nonlinear Shear Model of a Reinforced Concrete Beam. *Mathematics* **2024**, *12*, 499. [CrossRef]
33. Barrionuevo, G.O.; Calvopiña, H.; Debut, A.; Pérez-Salinas, C. Experimental and Numerical Investigation of Sliding Wear of Heat-Treated 316L Stainless Steel Additively Manufactured. *J. Mater. Res. Technol.* **2024**, *33*, 2692–2703. [CrossRef]

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