Hindawi
Journal of Mathematics
Volume 2024, Article ID 2439343, 16 pages
https://doi.org/10.1155/2024/2439343

Research Article

Numerical Solution of Burgers–Huxley Equation Using a Higher Order Collocation Method

Aditi Singh,[1] **Sumita Dahiya**,[2] **Homan Emadifar**,[3,4,5] **and Masoumeh Khademi**[5]

[1]*Department of Mathematics, Dayalbagh Educational Institute, Agra 282005, India*
[2]*Department of Mathematics, Netaji Subhas University of Technology, Delhi 110078, India*
[3]*Department of Mathematics, Saveetha School of Engineering, Saveetha Institute of Medical and Technical Sciences, Saveetha University, Chennai 602105, Tamil Nadu, India*
[4]*MEU Research Unit, Middle East University, Amman 11831, Jordan*
[5]*Department of Mathematics, Hamedan Branch, Islamic Azad University, Hamedan, Iran*

Correspondence should be addressed to Homan Emadifar; homan_emadi@yahoo.com

Received 20 August 2023; Revised 28 December 2023; Accepted 7 February 2024; Published 29 February 2024

Academic Editor: M. M. Bhatti

In this paper, the collocation method with cubic B-spline as basis function has been successfully applied to numerically solve the Burgers–Huxley equation. This equation illustrates a model for describing the interaction between reaction mechanisms, convection effects, and diffusion transport. Quasi-linearization has been employed to deal with the nonlinearity of equations. The Crank–Nicolson implicit scheme is used for discretization of the equation and the resulting system turned out to be semi-implicit. The stability of the method is discussed using Fourier series analysis (von Neumann method), and it has been concluded that the method is unconditionally stable. Various numerical experiments have been performed to demonstrate the authenticity of the scheme. We have found that the computed numerical solutions are in good agreement with the exact solutions and are competent with those available in the literature. Accuracy and minimal computational efforts are the key features of the proposed method.

1. Introduction

The Burgers–Huxley equation is given as

$$\frac{\partial v}{\partial t} - \frac{\partial^2 v}{\partial x^2} + \alpha v \frac{\partial v}{\partial x} - \beta v f(v) = 0. \tag{1}$$

Its initial and boundary conditions are

$$\begin{aligned} v(x,0) &= f(x), \\ v(a,t) &= g_0(t),\ v(b,t) = g_1(t). \end{aligned} \tag{2}$$

Here $a \le x \le b$, $t \ge 0$, and nonlinear reaction term is $f(v) = (1-v)(v-\gamma)$. α and $\beta \ge 0$ are advection and reaction coefficients, respectively. γ is a parameter such that $\gamma \in (a,b)$. When $\alpha = 0$, equation (1) is reduced to the Huxley equation which describes nerve pulse propagation in nerve fibres and wall motion in liquid crystals [1]. When $\beta = 0$, it is reduced to the well-known Burgers equation which describes the far field in wave propagation in nonlinear dissipative systems. When $\alpha = 0$ and $\beta = 1$, it becomes the FitzHugh–Nagumo equation which is the reaction diffusion equation used in circuit theory and biology [1]. When $\alpha = 0$ and $\beta = 0$, the equation turns into an important equation called Burgers–Huxley equation which describes many physical phenomena encountered in models where reaction, convection, and diffusion take place.

There are many computational methods for the solution of Burgers–Huxley equation, such as the Adomian decomposition method [2–4], homotopy analysis method [5], and variational iteration method [6, 7]. In 2006, Javidi [8] employed modified pseudospectral method to numerically obtain the solution of generalized Burgers–Huxley equation. Javidi and Golbabai [9] employed Chebyshev polynomials-based new domain decomposition algorithm, Ismail et al. [2] applied the Adomian decomposition method (ADM), Kaushik [10] used grid equidistribution, Babolian et al. [5]

the applied homotopy analysis method (HAM), and Khattak [11] used the computational meshless method to get analytic or numerical solutions of the generalized Burgers–Huxley equation. Tomasiello [12] used the IDQ method, Zhang et al. [13] used local discontinuous Galerkin methods, and Duan et al. [14] developed the lattice Boltzmann model to obtain the solution of the Burgers–Huxley equation.

Recently, Mittal and Tripathi developed a collocation method using cubic B-splines as basis functions to numerically solve generalized Burgers–Fisher and generalized Burgers–Huxley equations [15]. Celik [16] employed the Haar wavelet method, Reza Mohammadi [17] used B-spline collocation algorithm, and Dehghan et al. made use of different methods which are based on interpolation scaling functions as well as mixed collocation finite difference schemes for the numerical solution of the nonlinear generalized Burgers–Huxley equation [18]. Recently, in 2023, Chin [19] used the coupling of the nonstandard finite difference approach in the time variables with the Galerkin method and the compactness methods in the space to obtain solution of the Burgers–Huxley equation.

Nowadays, B-spline functions are becoming popular and are being used as a powerful tool in various fields such as approximation theory, image processing, atomic and molecular physics, numerical simulations, and computer-aided designs. Basis spline functions have been incorporated in various numerical methods such as differential quadrature method and collocation method to deal with the differential equations. The cubic B-spline collocation method was developed by Goh et al. [20] to numerically solve one-dimensional heat and advection diffusion equations. Dag and Saka [21] used this method for equal width equation. Different variants of this method have been used by Kadalbajoo and Arora [22], Zahra [23], Dag [24], and Khater et al. [25] to solve various other important equations. Mittal and Dahiya [26] used quintic B-splines in the differential quadrature method to solve a class of Fisher–Kolmogorov equations. Fourth-order collocation methods have been developed by Mittal and Rohila [27] to numerically study the reaction diffusion Fisher's equation. In 2020, Singh et al. [28] employed the fourth-order collocation method to get numerical solutions of the Burgers–Fisher equation. Roul et al. used B-spline collocation methods in various studies [29–34] to find solution of some important problems occurring in the field of science and engineering. Recently, Kumar et al. [35] used a spline approximation technique to solve the boundary value inverse problem associated with the generalized Burgers–Fisher and generalized Burgers–Huxley equations.

A novel method called the fourth-order cubic B-spline collocation method is adopted in the proposed work to solve the Burgers–Huxley equation. The present method does not involve any integrals to get the final set of equations, and thus computational efforts have been reduced to a great extent. Fourth-order approximation for single as well as double derivatives is employed. This is done by using different end conditions and taking one extra term in the Taylor series expansion which has resulted in accurate and efficient numerical solutions. The aim of this work is to study the numerical solutions of the Burgers–Huxley equation for different parametric values using collocation method with cubic B-splines as basis functions. A preprint of this work has been previously published by Singh et al. [36] in 2023.

The present scheme gives the approximate solution at any point of the solution domain. Accuracy obtained in this work is satisfactory and comparable with those present in the previous literature. The proposed method is quite simple and produces highly efficient results and hence reduces complexity and computational cost.

2. Mathematical Formulation

Let us consider an equal partition of the domain Ω by the knots x_j, $j = 0, 1, 2, \ldots, N$, such that $h = x_j - x_{j-1}$ is the length of each interval. The third-degree B-spline termed as cubic B-spline is given as

$$F_{i,3}(x) = \frac{1}{h^3}\begin{cases} (x - x_{i-2})^3, & x \in [x_{i-2}, x_{i-1}), \\ (x - x_{i-2})^3 - 4(x - x_{i-1})^3, & x \in [x_{i-1}, x_i), \\ (x_{i+2} - x)^3 - 4(x_{i+1} - x)^3, & x \in [x_i, x_{i+1}), \\ (x_{i+2} - x)^3, & x \in [x_{i+1}, x_{i+2}), \\ 0, & \text{otherwise}, \end{cases} \tag{3}$$

where $[F_{-1}(x), F_0(x), F_1(x), \ldots\ldots, F_N(x), F_{N+1}(x)]$ are basis functions over the interval.

Exact solution $v(x,t)$ is approximated by $K(x,t)$ in the cubic B-spline collocation method as

$$K(x,t) = \sum_{j=-1}^{N+1} a_j(t) F_j(x), \tag{4}$$

where $a_j(t)$'s are unknown quantities which we have to find. $K(x, t)$ is considered to satisfy the following interpolatory and boundary conditions:

$$\begin{aligned} K(x_j, t) &= v(x_j, t), \quad 0 \le j \le N, \\ K''(x_j, t) &= v''(x_j, t) - \frac{1}{12} h^2 v^{(4)}(x_j, t), \quad j = 0, N. \end{aligned} \tag{5}$$

If $K(x, t)$ is a unique cubic spline interpolant which satisfies above boundary conditions and $v(x,t)$ is a smooth function, then from [38], we have

$$K'(x_j,t) = v'(x_j,t) + O(h^4), \quad 0 \le j \le N, \tag{6}$$

$$K''(x_j,t) = v''(x_j,t) - \frac{1}{12}h^2 v^{(4)}(x_j,t) + O(h^4), \quad 0 \le j \le N. \tag{7}$$

Using Taylor's expansion and finite differences, the approximate values of $K(x, t)$ and its first-order derivatives at the knots are defined as follows.

For $j = 0$,

$$v^{(4)}(x_j,t) = \frac{2K''(x_0,t) - 5K''(x_1,t) + 4K''(x_2,t) - K''(x_3,t)}{h^2} + O(h^2). \tag{8}$$

For $1 \le j \le N-1$,

$$v^{(4)}(x_j,t) = \frac{K''(x_{j-1},t) - 2K''(x_j,t) + K''(x_{j+1},t)}{h^2} + O(h^2). \tag{9}$$

For $j = N$,

$$v^{(4)}(x_j,t) = \frac{2K''(x_N,t) - 5K''(x_{N-1},t) + 4K''(x_{N-2},t) - K''(x_{N-3},t)}{h^2} + O(h^2). \tag{10}$$

Using equations (8)–(10) in (6) and (7), we get

$$v'(x_j,t) = K'(x_j,t) + O(h^4), \quad 0 \le j \le N. \tag{11}$$

For $j = 0$,

$$v''(x_0,t) = \frac{14K''(x_0,t) - 5K''(x_1,t) + 4K''(x_2,t) - K''(x_3,t)}{12} + O(h^4). \tag{12}$$

For $1 \le j \le N-1$,

$$v''(x_j,t) = \frac{K''(x_{j-1},t) + 10K''(x_j,t) + K''(x_{j+1},t)}{12} + O(h^4). \tag{13}$$

For $j = N$,

$$v''(x_N,t) = \frac{14K''(x_N,t) - 5K''(x_{N-1},t) + 4K''(x_{N-2},t) - K''(x_{N-3},t)}{12} + O(h^4). \tag{14}$$

Using equations (3) and (4), equations (12)–(14) can be simplified as follows.

For $j = 0$,

$$v''(x_0, t) = \frac{14a_{-1} - 33a_0 + 28a_1 - 14a_2 + 6a_3 - a_4}{2h^2}. \quad (15)$$

For $1 \le j \le N - 1$,

$$v''(x_j, t) = \frac{a_{j-2} + 8a_{j-1} - 18a_j + 8a_{j+1} + a_{j+2}}{2h^2}. \quad (16)$$

For $j = N$,

$$v''(x_N, t) = \frac{14a_{N+1} - 33a_N + 28a_{N-1} - 14a_{N-2} + 6a_{N-3} - a_{N-4}}{2h^2}. \quad (17)$$

3. Implementation of the Method

We now use the Crank–Nicolson scheme to discretize Burgers–Huxley equation (1) and then we get

$$\frac{v^{(n+1)} - v^{(n)}}{\Delta t} - \frac{v_{xx}^{(n+1)} + v_{xx}^{(n)}}{2} + \alpha \frac{(vv_x)^{(n+1)} + (vv_x)^{(n)}}{2} + \beta \frac{(v(1-v)(v-\gamma))^{(n+1)} + (v(1-v)(v-\gamma))^{(n)}}{2} = 0. \quad (18)$$

Quasi-linearization formula is

$$f\left(v^{(n+1)}, v_x^{(n+1)}, v_{xx}^{(n+1)}\right) = f\left(v^{(n)}, v_x^{(n)}, v_{xx}^{(n)}\right) + \left(v^{(n+1)} - v^{(n)}\right)\frac{\partial f^{(n)}}{\partial v} + \left(v_x^{(n+1)} - v_x^{(n)}\right)\frac{\partial f^{(n)}}{\partial v_x} + \left(v_{xx}^{(n+1)} - v_{xx}^{(n)}\right)\frac{\partial f^{(n)}}{\partial v_{xx}}. \quad (19)$$

By applying quasi-linearization in equation (18), we get

$$\begin{aligned} &\frac{v^{(n+1)} - v^{(n)}}{\Delta t} + \gamma\beta \frac{v^{(n+1)} + v^{(n)}}{2} - \epsilon \frac{v_{xx}^{(n+1)} + v_{xx}^{(n)}}{2} + (\alpha - (1+\gamma)\beta) \frac{v^{(n+1)} v_x^{(n)} + v_x^{(n+1)} v^{(n)}}{2} \\ &+ \beta \frac{3v^{(n+1)}\left(v^2\right)^{(n)} - \left(v^3\right)^{(n)}}{2} = 0. \end{aligned} \quad (20)$$

Now, terms of (n)th and $(n+1)$th time levels are separated to get the equation of the form

$$\begin{aligned} &v^{(n+1)}\left[1 + \frac{\gamma\beta\Delta t}{2} + \frac{(\alpha - (1+\gamma)\beta)\Delta t}{2} v_x^{(n)} + \frac{3\beta\Delta t}{2}\left(v^2\right)^{(n)}\right] + v_x^{(n+1)}\left[\frac{(\alpha - (1+\gamma)\beta)\Delta t}{2} v^{(n)}\right] - \frac{\Delta t}{2} v_{xx}^{(n+1)} \\ &= v^{(n)}\left[1 - \frac{\gamma\beta\Delta t}{2} + \frac{\beta\Delta t}{2}\left(v^2\right)^{(n)}\right] + \frac{\Delta t}{2} v_{xx}^{(n)}. \end{aligned} \quad (21)$$

For $j=0$,

$$\left[a_{-1}^{(n+1)}+4a_0^{(n+1)}+a_1^{(n+1)}\right]\left(1+\frac{\gamma\beta\Delta t}{2}+\frac{(\alpha-(1+\gamma)\beta)\Delta t}{2}v_x^{(n)}+\frac{3\beta\Delta t}{2}\left(v^2\right)^{(n)}\right)+\left[a_1^{(n+1)}-a_{-1}^{(n+1)}\right]$$
$$\cdot\left(\frac{3(\alpha-(1+\gamma)\beta)\Delta t}{2h}v^{(n)}\right)-\frac{\Delta t}{4h^2}\left[14a_{-1}^{(n+1)}-33a_0^{(n+1)}+28a_1^{(n+1)}-14a_2^{(n+1)}+6a_3^{(n+1)}-a_4^{(n+1)}\right] \tag{22}$$
$$=\left[a_{-1}^{(n)}+4a_0^{(n)}+a_1^{(n)}\right]\left(1-\frac{\gamma\beta\Delta t}{2}+\frac{\beta\Delta t}{2}\left(v^2\right)^{(n)}\right)+\frac{\Delta t}{4h^2}\left[14a_{-1}^{(n)}-33a_0^{(n)}+28a_1^{(n)}-14a_2^{(n)}+6a_3^{(n)}-a_4^{(n)}\right].$$

We may write it as

$$s_1a_{-1}^{(n+1)}+s_2a_0^{(n+1)}+s_3a_1^{(n+1)}+s_4a_2^{(n+1)}+s_5a_3^{(n+1)}+s_6a_4^{(n+1)}=b_1a_{-1}^{(n)}+b_2a_0^{(n)}+b_3a_1^{(n)}+b_4a_2^{(n)}+b_5a_3^{(n)}+b_6a_4^{(n)}. \tag{23}$$

For $1\le j\le N-1$,

$$\left[a_{j-1}^{(n+1)}+4a_j^{(n+1)}+a_{j+1}^{(n+1)}\right]\left(1+\frac{\gamma\beta\Delta t}{2}+\frac{(\alpha-(1+\gamma)\beta)\Delta t}{2}v_x^{(n)}+\frac{3\beta\Delta t}{2}\left(v^2\right)^{(n)}\right)+\left[a_{j+1}^{(n+1)}-a_{j-1}^{(n+1)}\right]$$
$$\cdot\left(\frac{3(\alpha-(1+\gamma)\beta)\Delta t}{2h}v^{(n)}\right)-\frac{\Delta t}{4h^2}\left[a_{j-2}^{(n+1)}+8a_{j-1}^{(n+1)}-18a_j^{(n+1)}+8a_{j+1}^{(n+1)}+a_{j+2}^{(n+1)}\right]=\left[a_{j-1}^{(n)}+4a_j^{(n)}+a_{j+1}^{(n)}\right] \tag{24}$$
$$\left(1-\frac{\gamma\beta\Delta t}{2}+\frac{\beta\Delta t}{2}\left(v^2\right)^{(n)}\right)+\frac{\Delta t}{4h^2}\left[a_{j-2}^{(n)}+8a_{j-1}^{(n)}-18a_j^{(n)}+8a_{j+1}^{(n)}+a_{j+2}^{(n)}\right].$$

We may write it as

$$t_1a_{j-2}^{(n+1)}+t_2a_{j-1}^{(n+1)}+t_3a_j^{(n+1)}+t_4a_{j+1}^{(n+1)}+t_1a_{j+2}^{(n+1)}=p_1a_{j-2}^{(n)}+p_2a_{j-1}^{(n)}+p_3a_j^{(n)}+p_2a_{j+1}^{(n)}+p_1a_{j+2}^{(n)}. \tag{25}$$

For $j=N$,

$$\left[a_{N-1}^{(n+1)}+4a_N^{(n+1)}+a_{N+1}^{(n+1)}\right]\left(1+\frac{\gamma\beta\Delta t}{2}+\frac{(\alpha-(1+\gamma)\beta)\Delta t}{2}v_x^{(n)}+\frac{3\beta\Delta t}{2}\left(v^2\right)^{(n)}\right)+\left[a_N^{(n+1)}-a_{N-2}^{(n+1)}\right]$$
$$\cdot\left(\frac{3(\alpha-(1+\gamma)\beta)\Delta t}{2h}v^{(n)}\right)-\frac{\Delta t}{4h^2}\left[14a_{-1}^{(n+1)}-33a_N^{(n+1)}+28a_{N-1}^{(n+1)}-14a_{N-2}^{(n+1)}+6a_{N-3}^{(n+1)}-a_{N-4}^{(n+1)}\right] \tag{26}$$
$$=\left[a_{N-1}^{(n)}+4a_N^{(n)}+a_{N+1}^{(n)}\right]\left(1-\frac{\gamma\beta\Delta t}{2}+\frac{\beta\Delta t}{2}\left(v^2\right)^{(n)}\right)+\frac{\Delta t}{4h^2}\left[14a_{N+1}^{(n)}-33a_N^{(n)}+28a_{N-1}^{(n)}-14a_{N-2}^{(n)}+6a_{N-3}^{(n)}-a_{N-4}^{(n)}\right].$$

We may write it as

$$v_1 a_{N-4}^{(n+1)} + v_2 a_{N-3}^{(n+1)} + v_3 a_{N-2}^{(n+1)} + v_4 a_{N-1}^{(n+1)} + v_5 a_N^{(n+1)} + v_6 a_{N+1}^{(n+1)} = d_1 a_{N-4}^{(n)} + d_2 a_{N-3}^{(n)} + d_3 a_{N-2}^{(n)} + d_4 a_{N-1}^{(n)} + d_5 a_N^{(n)} + d_6 a_{N+1}^{(n)}. \tag{27}$$

Finally, the following system of linear equations is obtained:

$$AC^{(n+1)} = BC^{(n)}, \tag{28}$$

where

$$C = \left[a_{-1}, a_0, a_1, \ldots\ldots, a_{N+1}\right]^T,$$

$$A = \begin{bmatrix} s_1 & s_2 & s_3 & s_4 & s_5 & s_6 & \ldots & \ldots \\ t_1 & t_2 & t_3 & t_4 & t_1 & \ldots & \ldots & \ldots \\ & t_1 & t_2 & t_3 & t_4 & t_1 & \ldots & \ldots \\ & \ldots & \ldots & \ldots & \ldots & \ldots & & \\ & \ldots & \ldots & \ldots & \ldots & \ldots & & \\ & \ldots & \ldots & t_1 & t_2 & t_3 & t_4 & t_1 \\ & \ldots & v_1 & v_2 & v_3 & v_4 & v_5 & v_6 \end{bmatrix},$$

$$B = \begin{bmatrix} b_1 & b_2 & b_3 & b_4 & b_5 & b_6 & \ldots & \ldots \\ p_1 & p_2 & p_3 & p_2 & p_1 & \ldots & \ldots & \ldots \\ & p_1 & p_2 & p_3 & p_2 & p_1 & \ldots & \ldots \\ & \ldots & \ldots & \ldots & \ldots & \ldots & & \\ & \ldots & \ldots & \ldots & \ldots & \ldots & & \\ & \ldots & \ldots & p_1 & p_2 & p_3 & p_2 & p_1 \\ & \ldots & d_1 & d_2 & d_3 & d_4 & d_5 & d_6 \end{bmatrix}. \tag{29}$$

The resulting system of equation (28) is semi-implicit. We can clearly observe that there are $N+1$ equations in $N+3$ unknowns. a_{-1} and a_{N+1} are eliminated with the help of Dirichlet or Neumann's boundary conditions. After elimination, we get $N+1$ equations in $N+1$ unknowns. B-spline approximation of initial condition is used to get the initial vector $[a_0^{(0)}, a_1^{(0)}, a_2^{(0)}, \ldots\ldots, a_N^{(0)}]^T$. Now, the system of equations can be solved at any desired time level with the help of initial vector.

4. Stability Analysis

In equation (21), we assume

$$\begin{aligned} v^n &= k, \\ r_1 &= 1 + \frac{\gamma\beta\Delta t}{2} + \frac{(\alpha - (1+\gamma)\beta)\Delta t}{2} v_x^{(n)} + \frac{3\beta\Delta t}{2}\left(v^2\right)^{(n)}, \\ r_2 &= \frac{(\alpha - (1+\gamma)\beta)\Delta t}{2} v^{(n)}, \\ r_3 &= 1 - \frac{\gamma\beta\Delta t}{2} + \frac{\beta\Delta t}{2}\left(v^2\right)^{(n)}. \end{aligned} \tag{30}$$

Then

$$\begin{aligned} &\left[a_{j-1}^{(n+1)} + 4a_j^{(n+1)} + a_{j+1}^{(n+1)}\right] r_1 + \left[a_{j+1}^{(n+1)} - a_{j-1}^{(n+1)}\right] r_2 - \frac{\Delta t}{4h^2}\left[a_{j-2}^{(n+1)} + 8a_{j-1}^{(n+1)} - 18a_j^{(n+1)} + 8a_{j+1}^{(n+1)} + a_{j+2}^{(n+1)}\right] \\ &= \left[a_{j-1}^{(n)} + 4a_j^{(n)} + a_{j+1}^{(n)}\right] r_3 + \frac{\Delta t}{4h^2}\left[a_{j-2}^{(n)} + 8a_{j-1}^{(n)} - 18a_j^{(n)} + 8a_{j+1}^{(n)} + a_{j+2}^{(n)}\right]. \end{aligned} \tag{31}$$

Assume $(\Delta t/h^2) = L$.

$$\begin{aligned} &\frac{-L}{4} a_{j-2}^{(n+1)} + [r_1 - r_2 - 2L] a_{j-1}^{(n+1)} + \left[4r_1 + \frac{9L}{2}\right] a_j^{(n+1)} + [r_1 + r_2 - 2L] a_{j+1}^{(n+1)} - \frac{L}{4} a_{j+2}^{(n+1)} \\ &= \frac{L}{4} a_{j-2}^{(n)} + [r_3 + 2L] a_{j-1}^{(n)} + \left[4r_3 - \frac{9L}{2}\right] a_j^{(n)} + [r_3 + 2L] a_{j+1}^{(n)} + \frac{L}{4} a_{j+2}^{(n)}. \end{aligned} \tag{32}$$

Substituting $a_j^{(n)} = \mathrm{D}\ \xi^{(n)} \exp(ijmh)$, where h is step length, D is amplitude, and m is mode number, we have

$$\begin{aligned}
&D\xi^{(n+1)}\left(\frac{-L}{4}e^{-2imh} + [r_1 - r_2 - 2L]e^{-imh} + \left[4r_1 + \frac{9L}{2}\right] + [r_1 + r_2 - 2L]e^{imh} - \frac{L}{4}e^{2imh}\right) \\
&= D\xi^{(n)}\left(\frac{L}{4}e^{-2imh} + [r_3 + 2L]e^{-imh} + \left[4r_3 - \frac{9L}{2}\right] + [r_3 + 2L]e^{imh} + \frac{L}{4}e^{2imh}\right), \\
&\xi = \frac{(L/4)e^{-2imh} + [r_3 + 2L]e^{-imh} + [4r_3 - (9L/2)] + [r_3 + 2L]e^{imh} + (L/4)e^{2imh}}{(-L/4)e^{-2imh} + [r_1 - r_2 - 2L]e^{-imh} + [4r_1 + (9L/2)] + [r_1 + r_2 - 2L]e^{imh} - (L/4)e^{2imh}}, \\
&\xi = \frac{(L/2)\cos(2mh) + 2[r_3 + 2L]\cos(mh) + [4r_3 - (9L/2)]}{(-L/2)\cos(2mh) + 2[r_1 - 2L]\cos(mh) + 2ir_2\sin(mh) + [4r_1 + (9L/2)]},
\end{aligned} \tag{33}$$

or

$$\xi = \frac{A}{B + Ci}, \tag{34}$$

where

$$\begin{aligned}
A &= \frac{L}{2}\cos(2mh) + 2[r_3 + 2L]\cos(mh) + \left[4r_3 - \frac{9L}{2}\right], \\
B &= \frac{-L}{2}\cos(2mh) + 2[r_1 - 2L]\cos(mh) + \left[4r_1 + \frac{9L}{2}\right], \\
C &= 2r_2\sin(mh).
\end{aligned} \tag{35}$$

For stability of the present method, we should have

$$\begin{aligned}
&|\xi|^2 < 1, \\
&\Rightarrow \left|\frac{A}{B + Ci}\right|^2 < 1, \\
&\Rightarrow \left|\frac{B + Ci}{A}\right|^2 > 1, \\
&\Rightarrow \frac{B^2 + C^2}{A^2} > 1.
\end{aligned} \tag{36}$$

We need to show

$$B^2 + C^2 - A^2 > 0. \tag{37}$$

For minimum value of $B^2 + C^2 - A^2$, $\cos(mh) = 1$. Thus, on putting values of A, B, C from equation (35) in $B^2 + C^2 - A^2$, we get $72\beta\Delta t(\gamma + k^2)(1 + \beta\Delta t k^2)$ which is obviously positive.

Hence, the proposed method is unconditionally stable.

5. Numerical Experiments

The exact solution of equation (1) was derived by Wang et al. [1] using nonlinear transformations and is given by

$$v(x,t) = \left[\frac{\gamma}{2} + \frac{\gamma}{2}\tanh\left(a_1\left(x - a_2 t\right)\right)\right], \tag{38}$$

where

$$\begin{aligned}
a_1 &= \frac{-\alpha + \sqrt{\alpha^2 + 8\beta}}{8}\gamma, \\
a_2 &= \frac{\alpha\gamma}{2} - \frac{(2 - \gamma)\left(-\alpha + \sqrt{\alpha^2 + 8\beta}\right)}{4}.
\end{aligned} \tag{39}$$

This exact solution satisfies the following initial and boundary conditions:

$$\begin{aligned}
v(x,0) &= \left[\frac{\gamma}{2} + \frac{\gamma}{2}\tanh\left(a_1 x\right)\right], \\
v(0,t) &= \left[\frac{\gamma}{2} + \frac{\gamma}{2}\tanh\left(-a_1 a_2 t\right)\right], \\
v(1,t) &= \left[\frac{\gamma}{2} + \frac{\gamma}{2}\tanh\left(a_1\left(1 - a_2 t\right)\right)\right].
\end{aligned} \tag{40}$$

Error norms are given by

$$\begin{aligned}
L_2 &= \sqrt{\sum_{i=1}^{N}\left|u(x_i,t) - v(x_i,t)\right|^2}, \\
L_\infty &= \max_i\left|u(x_i,t) - v(x_i,t)\right|, \quad i = 1, 2, \ldots, n, \ldots,
\end{aligned} \tag{41}$$

where $u(x, t)$ and $v(x, t)$ are the exact and numerical solutions, respectively.

Formula for the rate of convergence is given by

$$\text{Rate} = \frac{\log\left(E(N_2)/E(N_1)\right)}{\log\left(N_1/N_2\right)}, \tag{42}$$

where $E(N_1)$ and $E(N_2)$ are L_∞ errors at grid sizes N_1 and N_2, respectively.

Accuracy and efficiency of the proposed method have been verified by comparing obtained results with the above exact solution and traditional methods available in the literature [2–4, 6, 7, 9, 13, 15–18, 39, 40].

Table 1: Absolute error comparison of the present scheme with different schemes for $\alpha = 0$, $\beta = 1$, $\gamma = 0.001$.

x	T	Present method	Bratsos [37]	Mohammadi [17]
0.1	0.05	$1.4431E-008$	$2.4987E-008$	$1.0299E-008$
	0.10	$2.1405E-008$	$4.9975E-008$	$1.5023E-008$
	1	$3.2318E-008$	$4.9975E-007$	$2.2487E-008$
0.5	0.05	$3.4513E-008$	$2.4987E-008$	$2.3131E-008$
	0.10	$5.7034E-008$	$4.9975E-008$	$3.8436E-008$
	1	$9.2342E-008$	$4.9974E-007$	$6.2465E-008$
0.9	0.05	$2.4987E-008$	$2.4987E-008$	$1.0299E-008$
	0.10	$2.1405E-008$	$4.9974E-008$	$1.5023E-008$
	1	$3.2316E-008$	$4.9974E-007$	$2.2487E-008$
CPU-time	0.05	0.0617 s	—	—
	0.10	0.0778 s	—	—
	1	0.0845 s	—	—

Table 2: L_∞ error comparison with three methods considered by Dehghan [18] taking $\alpha = 0$, $\beta = 1$, $\gamma = 0.001$.

	T	L_∞	CPU-time (s)
Present method	0.3	$7.76E-008$	0.0656
	0.6	$8.09E-008$	0.0561
	1	$8.09E-008$	0.0634
Dehghan 2012 ISF-Gm1	0.3	$5.92E-008$	—
	0.6	$6.23E-008$	—
	1	$6.24E-008$	—
Dehghan 2012 ISF-Gm2	0.3	$4.21E-008$	—
	0.6	$4.11E-008$	—
	1	$4.49E-008$	—
Dehghan 2012 MFDCM	0.3	$7.50E-008$	—
	0.6	$7.82E-008$	—
	1	$7.84E-008$	—

In order to find numerical solution, the space and time step lengths are taken as $h = 0.1$ and $\Delta t = 0.1$ or $\Delta t = 0.01$ for all examples unless otherwise stated. CPU-time in seconds is calculated for all the examples and shown in their respective tables. It is found that accuracy of the present method is satisfactory and comparable to or rather higher than those available in the literature. As we can see in all the examples, the present method is taking very less computational time and thus it is highly efficient. Solutions have been calculated for large values of t and it can be seen that it is taking very small CPU-time. There is one more advantage of the present method, that is, it requires less number of grid points resulting in low memory storage.

Example 1. When $\alpha = 0$, $\beta = 1$, equation (1) becomes the FitzHugh–Nagumo equation and can be written as

$$\frac{\partial v}{\partial t} = \frac{\partial^2 v}{\partial x^2} + v(1 - v)(v - \gamma). \tag{43}$$

For $\gamma = 0.001$, comparison of absolute errors of the present method with Bratsos [37] and Mohammadi [17] at different grid points at times $T = 0.05$, 0.10, 1 is shown in Table 1. Similarly, Table 2 compares L_∞ error of the present method with three different methods of Dehghan [18] at times $T = 0.3$, 0.6, 1. For $\gamma = 0.0001$, comparison of L_∞ error of the present method with three different methods of Dehghan [18] is established in Table 3. For large T, absolute errors of the present method are shown in Table 4. This shows that the method is giving good results for large times as well. The approximate solutions of the present method are shown graphically at times $T = 0.3$, 0.6, 1 in Figure 1. 3D form of the approximate solution for $T = 1$ is shown in Figure 2. In the relevant nonlinear dissipative systems, the solutions obtained here describe the special coherent structures.

It can be concluded from these figures and tables that the results are found to be quite competent with the literature. It can be clearly seen that it requires very less CPU-time; hence, the proposed method is efficient and requires minimal computational efforts.

Example 2. We set $\alpha = 2$ and $\beta = 1$. Table 5 compares L_∞ errors of the present scheme with three different methods of Dehghan [18] at times $T = 0.3$, 0.6, 1 for $\gamma = 0.001$ and $\gamma = 0.0001$. We get similar nature of results as that of Example 1.

Example 3. In this example, we take negative value of convection coefficient, i.e., $\alpha = -0.1$. β and γ are taken as 0.1 and 0.001, respectively. Comparison of absolute errors of the present method is done with Celik [16] at time $T = 0.9$ in Table 6. Results are comparable and show good accuracy.

Table 3: L_∞ error comparison with different methods considered by Dehghan [18] taking $\alpha = 0$, $\beta = 1$, $\gamma = 0.0001$.

	T	L_∞	CPU-time (s)
Present method	0.3	$7.76E-010$	0.0536
	0.6	$8.09E-010$	0.0633
	1	$8.08E-010$	0.0610
Dehghan 2012 ISF-Gm1	0.3	$5.92E-010$	—
	0.6	$6.23E-010$	—
	1	$6.25E-010$	—
Dehghan 2012 ISF-Gm2	0.3	$4.23E-010$	—
	0.6	$4.12E-010$	—
	1	$4.48E-010$	—
Dehghan 2012 MFDCM	0.3	$7.52E-010$	—
	0.6	$7.82E-010$	—
	1	$7.85E-010$	—

Table 4: Absolute errors of the present scheme for $\alpha = 0$, $\beta = 1$, $\gamma = 0.0001$ at different times T.

x	$T=5$	$T=10$	$T=50$
0.1	$2.1043E-010$	$2.1072E-010$	$2.1117E-010$
0.2	$4.7379E-010$	$4.7351E-010$	$4.7430E-010$
0.3	$6.6060E-010$	$6.6102E-010$	$6.6209E-010$
0.4	$7.7357E-010$	$7.7359E-010$	$7.7478E-010$
0.5	$8.1087E-010$	$8.1107E-010$	$8.1234E-010$
0.6	$7.7357E-010$	$7.7359E-010$	$7.7478E-010$
0.7	$6.6060E-010$	$6.6102E-010$	$6.6209E-010$
0.8	$4.7379E-010$	$4.7351E-010$	$4.7430E-010$
0.9	$2.1043E-010$	$2.1072E-010$	$2.1117E-010$
CPU-time (s)	0.0835	0.1225	0.2448

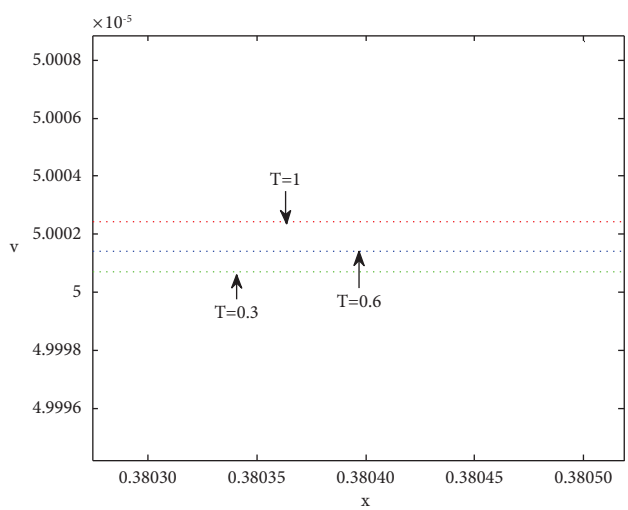

Figure 1: Numerical solutions of the present method for $\alpha = 0$, $\beta = 1$, $\gamma = 0.0001$ at different times T.

Example 4. In this example, we consider different values of β and γ as $\beta = 1$, 10, 100 and $\gamma = 0.001$, 0.0001, 0.00001. Here, α and Δt are taken as 5 and 0.003, respectively. The L_∞ errors of the obtained results are presented for $T = 0.3$ and 0.9 in Table 7 and compared with those of CM [15]. Figure 3 represents the pictorial view of absolute errors at $T = 1$ for $\beta = 10$ to $\beta = 100$ with an increment of 10. For this figure, $\gamma = 0.00001$ and $\Delta t = 0.01$. It can inferred that as β or γ decreases, error decreases. Thus, the smaller the diffusion coefficient, the better the accuracy.

Example 5. We set $\alpha = \gamma = 0.1$ and $\beta = 0.001$. Table 8 shows comparison of absolute errors of the present method with different methods given in the literature such as ADM [2–4], VIM [6, 7], DTM [39], and LDM [39] at times $T = 0.05$, 0.1, 1. The results of the present method show better accuracy

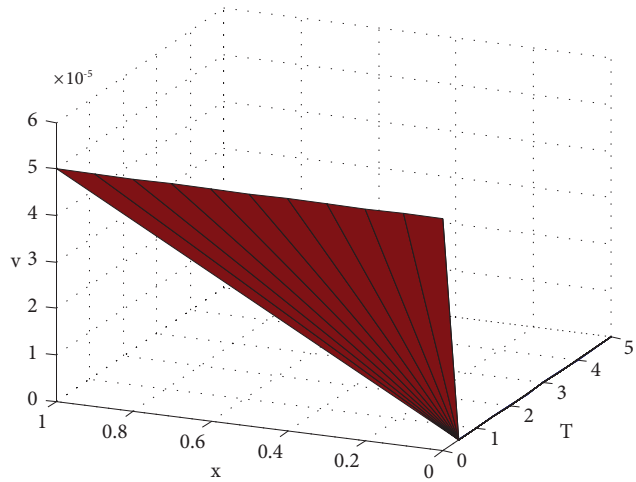

Figure 2: Evolution of numerical solution with space and time variables for $T = 1$.

Table 5: Comparison of L_∞ error with three methods considered by Dehghan [18] taking $\alpha = 2$, $\beta = 1$ at different times T.

	γ	T	L_∞	CPU-time (s)
Present method	0.001	0.3	$6.44E-008$	0.0612
		0.6	$6.74E-008$	0.0665
		1	$6.74E-008$	0.0689
	0.0001	0.3	$6.44E-010$	—
		0.6	$6.74E-010$	—
		1	$6.74E-010$	—
Dehghan 2012 ISF-Gm1	0.001	0.3	$3.75E-008$	—
		0.6	$3.95E-008$	—
		1	$3.96E-008$	—
	0.0001	0.3	$3.77E-010$	—
		0.6	$3.96E-010$	—
		1	$3.96E-010$	—
Dehghan 2012 ISF-Gm2	0.001	0.3	$2.87E-008$	—
		0.6	$2.85E-008$	—
		1	$3.05E-008$	—
	0.0001	0.3	$2.88E-010$	—
		0.6	$2.86E-010$	—
		1	$3.05E-010$	—
Dehghan 2012 MFDCM	0.001	0.3	$4.57E-008$	—
		0.6	$4.77E-008$	—
		1	$4.78E-008$	—
	0.0001	0.3	$4.57E-010$	—
		0.6	$4.78E-010$	—
		1	$4.78E-010$	—

and require less computational efforts. Figure 4 depicts absolute errors for $T = 0.05$ to $T = 1$ with $\Delta t = 0.05$. So, it can be easily seen from this figure that error decreases with decrease in time.

Example 6. We consider $\alpha = 0.01$, $\beta = \gamma = 0.0001$. Table 9 compares absolute errors of the present method with different methods such as ADM [2–4], VIM [6, 7], DTM [39], and LDM [39] at times $T = 0.05$, 0.1, 1.

Example 7. In this example, α, β, and γ are taken as 0.1, 0.001 and 0.0001, respectively. Absolute errors of the present scheme for large T are mentioned in Table 10. For $T = 50$, absolute errors of the present method can be seen in Figure 5. After getting the results in tabular as well as graphical form, it can inferred that the method is effective for large times. Table 11 compares L_∞ error of the present method with different schemes like Javidi [9], Zhang [13], and CM [15] at $T = 0.2$ and $T = 1$. Also, Table 12 validates the accuracy of the method by making L_∞ error comparison with HBCSM [41] for different n.

Example 8. We consider Burgers–Huxley equation (1) with $\alpha = \beta = 1$, $\gamma = 0.001$. Absolute errors of the present method are compared with methods given in ADM [2–4], VIM [6, 7], DTM [39], and Bratsos [37] at different grids for times

Table 6: Comparison of absolute errors with Celik [16] for time $T = 0.9$ and $\alpha = -0.1$, $\beta = 0.1$, $\gamma = 0.001$.

x	Present method	Celik [16]
0.03125	$5.2396E-010$	$8.1600E-010$
0.09375	$2.0197E-009$	$2.3533E-009$
0.15625	$3.7168E-009$	$3.6711E-009$
0.21875	$5.2899E-009$	$4.7682E-009$
0.28125	$6.5979E-009$	$5.6464E-009$
0.34375	$7.5882E-009$	$6.3052E-009$
0.40625	$8.2475E-009$	$6.7438E-009$
0.46875	$8.5759E-009$	$6.9628E-009$
0.53125	$8.5759E-009$	$6.9630E-009$
0.59375	$8.2475E-009$	$6.7437E-009$
0.65625	$7.5882E-009$	$6.3054E-009$
0.71875	$6.5978E-009$	$5.6463E-009$
0.78125	$5.2899E-009$	$4.7682E-009$
0.84375	$3.7168E-009$	$3.6711E-009$
0.90625	$2.0197E-009$	$2.3535E-009$
0.96875	$5.2395E-010$	$8.1600E-010$
CPU-time (s)	0.0740	

Table 7: Comparison of L_∞ error with CM [15] taking $\alpha = 5$.

T	β	$\gamma = 0.00001$		$\gamma = 0.0001$		$\gamma = 0.001$		CPU-time (present method)
		Present method	CM [15]	Present method	CM [15]	Present method	CM [15]	
0.3	1	$6.117E-012$	$8.019E-012$	$6.117E-010$	$8.019E-010$	$6.118E-008$	$8.015E-008$	0.1053 s
	10	$6.916E-011$	$1.008E-011$	$6.916E-009$	$1.008E-009$	$6.913E-007$	$1.007E-007$	0.1004 s
	100	$7.960E-010$	$1.277E-010$	$7.958E-008$	$1.277E-008$	$7.936E-006$	$1.276E-006$	0.1062 s
0.9	1	$6.461E-012$	$2.405E-012$	$6.461E-010$	$2.405E-010$	$6.462E-008$	$2.404E-008$	0.1762 s
	10	$7.305E-011$	$3.024E-011$	$7.305E-009$	$3.024E-009$	$7.307E-007$	$3.022E-007$	0.1811 s
	100	$8.409E-010$	$3.832E-010$	$8.417E-008$	$3.832E-008$	$8.493E-006$	$3.829E-006$	0.1758 s

Given CPU-time is for the present method.

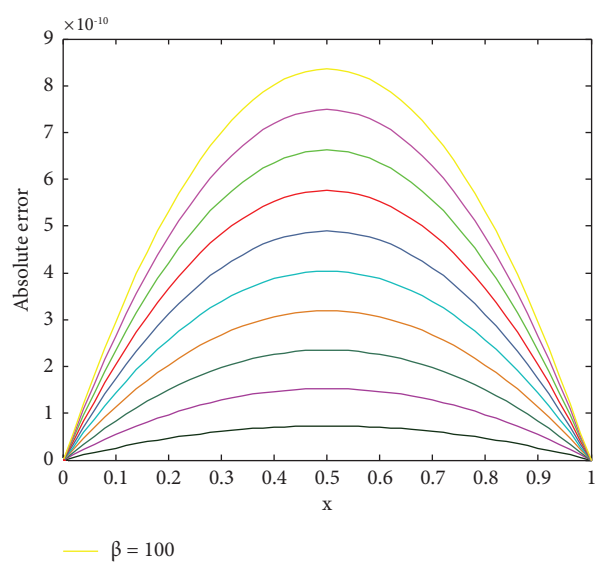

Figure 3: Absolute errors at $T = 1$ for $\beta = 10$ to $\beta = 100$ with an increment of 10.

Table 8: Comparison of absolute error at grid points with different schemes taking $\alpha = \gamma = 0.1$ and $\beta = 0.001$.

x	T	Present method	ADM [2–4]	VIM [6, 7]	DTM [39]	LDM [40]
0.1	0.05	$1.1189E-007$	$1.3634E-007$	$1.3608E-007$	$1.3608E-007$	$1.3607E-007$
	0.10	$1.6373E-007$	$2.7243E-007$	$2.7216E-007$	$2.7216E-007$	$2.7215E-007$
	1	$2.4465E-007$	$2.7220E-006$	$2.7215E-006$	$2.7215E-006$	$2.7215E-006$
0.5	0.05	$2.5319E-007$	$1.3733E-007$	$1.3608E-007$	$1.3608E-007$	$1.3607E-007$
	0.10	$4.2041E-007$	$2.7345E-007$	$2.7216E-007$	$2.7216E-007$	$2.7216E-007$
	1	$6.8253E-007$	$2.7230E-006$	$2.7215E-006$	$2.7215E-006$	$2.7215E-006$
0.9	0.05	$1.1199E-007$	$1.3838E-007$	$1.3608E-007$	$1.3608E-007$	$1.3607E-007$
	0.10	$1.6392E-007$	$2.7447E-007$	$2.7216E-007$	$2.7216E-007$	$2.7215E-007$
	1	$2.4500E-007$	$2.7240E-006$	$2.7215E-006$	$2.7215E-006$	$2.7215E-006$
CPU-time (present method)	0.05	0.0585 s	—	—	—	—
	0.10	0.0793 s	—	—	—	—
	1	0.1021 s	—	—	—	—

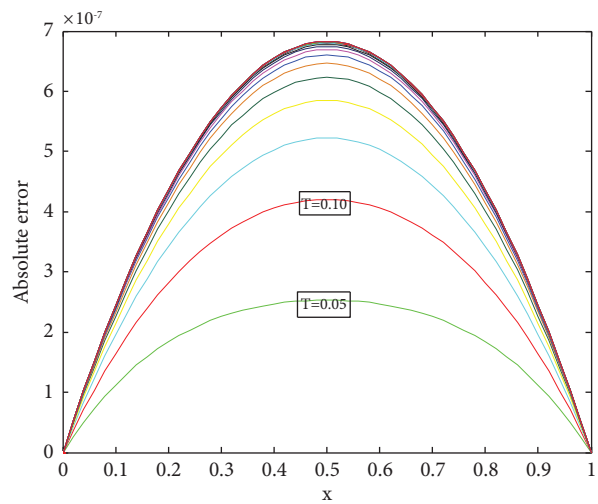

Figure 4: Absolute errors for $T = 0.05$ to $T = 1$ with $\Delta t = 0.05$ ($n = 50$).

Table 9: Comparison of absolute error at grid points with different schemes taking $\alpha = 0.01$, $\beta = \gamma = 0.0001$.

x	T	Present method	ADM [2–4]	VIM [6, 7]	DTM [39]	LDM [40]
0.1	0.05	$1.2376E-014$	$2E-014$	$2E-014$	$2E-014$	$1.87E-014$
	0.10	$1.8241E-014$	$4E-014$	$3E-014$	$3E-014$	$3.74E-014$
	1	$2.7405E-014$	$3.7E-014$	$3.7E-013$	$3.7E-013$	$3.74E-014$
0.5	0.05	$2.8836E-014$	$2E-014$	$2E-014$	$2E-014$	$1.87E-014$
	0.10	$4.7758E-014$	$4E-014$	$3E-014$	$3E-014$	$3.74E-014$
	1	$7.7411E-014$	$3.7E-013$	$3.7E-013$	$3.7E-013$	$3.74E-014$
0.9	0.05	$1.2376E-014$	$2E-014$	$2E-014$	$2E-014$	$1.87E-014$
	0.10	$1.8241E-014$	$4E-014$	$3E-014$	$3E-014$	$3.74E-014$
	1	$2.7405E-014$	$3.7E-013$	$3.7E-013$	$3.7E-013$	$3.74E-014$
CPU-time (present method)	0.05	0.0693 s	—	—	—	—
	0.10	0.0724 s	—	—	—	—
	1	0.1028 s	—	—	—	—

Table 10: Absolute errors of the present scheme for different times T taking $\alpha = 0.1$, $\beta = 0.001$, $\gamma = 0.0001$.

x	$T = 5$	$T = 10$	$T = 50$
0.1	$2.2286E-013$	$2.2291E-013$	$2.2291E-013$
0.2	$4.1080E-013$	$4.1071E-013$	$4.1071E-013$
0.3	$5.4476E-013$	$5.4482E-013$	$5.4483E-013$
0.4	$6.2533E-013$	$6.2530E-013$	$6.2530E-013$
0.5	$6.5212E-013$	$6.5212E-013$	$6.5213E-013$
0.6	$6.2533E-013$	$6.2530E-013$	$6.2530E-013$
0.7	$5.4476E-013$	$5.4482E-013$	$5.4483E-013$
0.8	$4.1080E-013$	$4.1071E-013$	$4.1071E-013$
0.9	$2.2286E-013$	$2.2291E-013$	$2.2291E-013$
CPU-time (s)	0.0821	0.0924	0.2281

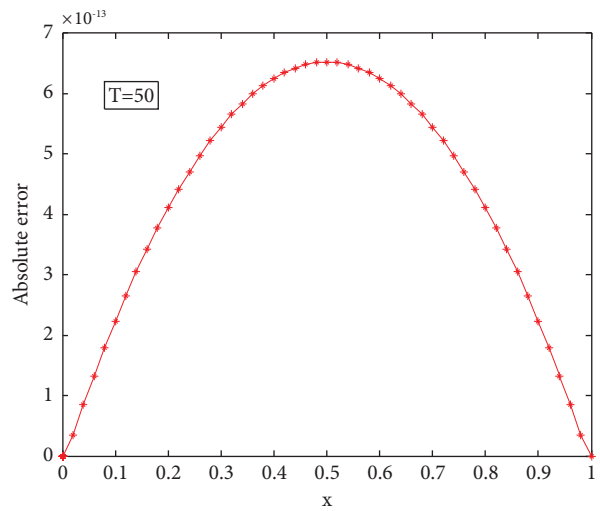

Figure 5: Absolute errors for $\alpha = 0.1$, $\beta = 0.001$, and $\gamma = 0.0001$ at time $T = 50$.

Table 11: L_∞ error comparison with different schemes taking $\alpha = 0.1$, $\beta = 0.001$, $\gamma = 0.0001$.

	$T = 0.2$	$T = 1$
Present method	$5.7295E-013$	$6.6850E-013$
CM [15]	$3.0271E-013$	$3.4889E-013$
Zhang p^4 [13]	$2.8828E-013$	$3.3565E-013$
Zhang p^2 [13]	$2.8832E-013$	$3.3567E-013$
Javidi [9]	$3.0689E-013$	$3.6182E-013$
CPU-time (s) (present method)	0.0857	0.1009

Table 12: L_∞ error comparison of the present method with HBCSM [41] for different n.

n	4	8	16	32	64
Present method	$6.6723E-013$	$6.6836E-013$	$6.6865E-013$	$6.6872E-013$	$6.6874E-013$
HBCSM [41]	$7.008E-007$	$1.761E-007$	$4.874E-008$	$1.710E-008$	$9.212E-009$

Table 13: Comparison of absolute error at grid points with different schemes taking $\alpha = \beta = 1$ and $\gamma = 0.001$.

x	T	Present method	ADM [2–4]	VIM [6, 7]	DTM [39]	FD6 [37]
0.1	0.05	$1.12377E-008$	$1.87406E-008$	$1.87405E-008$	$1.87406E-008$	$1.8740E-008$
	0.10	$1.8242E-008$	$3.74812E-008$	$3.74813E-008$	$3.74813E-008$	$3.7481E-008$
	1	$2.7410E-008$	$3.74812E-007$	$3.74812E-007$	$3.74812E-007$	$3.7481E-007$
0.5	0.05	$2.8838E-008$	$1.87406E-008$	$1.87405E-008$	$1.87406E-008$	$1.8739E-008$
	0.10	$4.7761E-008$	$3.74812E-008$	$3.74813E-008$	$3.74813E-008$	$3.7473E-008$
	1	$7.7424E-008$	$3.74812E-007$	$3.74813E-007$	$3.74813E-007$	$3.7210E-007$
0.9	0.05	$1.2379E-008$	$1.87406E-008$	$1.87405E-008$	$1.87406E-008$	$1.8725E-008$
	0.10	$1.8243E-008$	$3.74812E-008$	$3.74813E-008$	$3.74813E-008$	$3.7418E-008$
	1	$2.7411E-008$	$3.74812E-007$	$3.74813E-007$	$3.74812E-007$	$3.6842E-007$
CPU-time	0.05	0.0548 s	—	—	—	—
	0.10	0.0799 s	—	—	—	—
	1	0.1035 s	—	—	—	—

Table 14: L_∞ error comparison with different schemes taking $\alpha = \beta = 1$, $\gamma = 0.001$.

	$T = 0.2$	$T = 1$
Present method	$6.6374E-008$	$7.7424E-008$
CM [15]	$3.7487E-008$	$4.2939E-008$
Zhang p^4 [13]	$3.7715E-008$	$4.3912E-008$
Zhang p^2 [13]	$3.7725E-008$	$4.3914E-008$
Javidi [9]	$4.0138E-008$	$4.6849E-008$
CPU-time (s) (present method)	0.0798	0.1077

$T = 0.05, 0.1, 1$. The results are shown in Table 13. L_∞ errors are calculated at times $T = 0.2, 1$ and compared with different methods such as Javidi [9], Zhang [13], and CM [15] as reported in Table 14.

6. Conclusion

In this work, we have proposed a fourth-order cubic B-spline collocation method to solve the second-order nonlinear Burgers–Huxley equation. The various numerical experiments show that this method can produce accurate as well as efficient solutions. MATLAB programming is done for calculations and plotting. The main inferences are as follows:

(1) A technique based on fourth-order approximation of the solution has been used.

(2) From the numerical section, it is evident that the results are in full agreement with the exact solution and are quite competent with those available in the literature. The method satisfies the physical behavior of the nonlinear Burgers–Huxley equation.

(3) Stability of the present method has been verified and found to be unconditionally stable.

(4) In different settings of the parameters, this method can successfully provide highly efficient solutions.

(5) The method is reliable, easy to implement, and economical.

(6) Results illustrate that the present scheme is a valuable tool for studying various nonlinear problems. It can be extended to higher dimensional partial differential equations.

(7) The advantage of the present method over other methods is that the present method is convenient for solving boundary value problems with numerical ease, high accuracy, and minimal time consumption.

Data Availability

No data were used to support this study.

Disclosure

An earlier version of this manuscript is also available on ResearchGate and can be accessed through https://www.researchgate.net/publication/369069921_Numerical_solution_of_Burgers'-Huxley_equation_using_a_higher_order_collocation_method.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

The authors are very thankful to Prof. S. P. Singh for his unconditional guidance and support.

References

[1] X. Y. Wang, Z. S. Zhu, and Y. K. Lu, "Solitary wave solutions of the generalised Burgers–Huxley equation," *Journal of Physics A: Mathematical and General*, vol. 23, no. 3, pp. 271–274, 1990.

[2] H. N. A. Ismail, K. Raslan, and A. A. Abd Rabboh, "Adomian decomposition method for Burger's–Huxley and Burger's–Fisher equations," *Applied Mathematics and Computation*, vol. 159, no. 1, pp. 291–301, 2004.

[3] I. Hashim, M. S. M. Noorani, and B. Batiha, "A note on the Adomian decomposition method for the generalized Huxley equation," *Applied Mathematics and Computation*, vol. 181, no. 2, pp. 1439–1445, 2006.

[4] I. Hashim, M. S. M. Noorani, and M. R. Said Al-Hadidi, "Solving the generalized Burgers–Huxley equation using the Adomian decomposition method," *Mathematical and Computer Modelling*, vol. 43, no. 11–12, pp. 1404–1411, 2006.

[5] E. Babolian and J. Saeidian, "Analytic approximate solutions to Burgers, Fisher, Huxley equations and two combined forms

of these equations," *Communications in Nonlinear Science and Numerical Simulation*, vol. 14, no. 5, pp. 1984–1992, 2009.

[6] B. Batiha, M. S. M. Noorani, and I. Hashim, "Numerical simulation of the generalized Huxley equation by He's variational iteration method," *Applied Mathematics and Computation*, vol. 186, no. 2, pp. 1322–1325, 2007.

[7] B. Batiha, M. S. M. Noorani, and I. Hashim, "Application of variational iteration method to the generalized Burgers–Huxley equation," *Chaos, Solitons and Fractals*, vol. 36, no. 3, pp. 660–663, 2008.

[8] M. Javidi, "A numerical solution of the generalized Burger's–Huxley equation by pseudospectral method and Darvishi's preconditioning," *Applied Mathematics and Computation*, vol. 175, no. 2, pp. 1619–1628, 2006.

[9] M. Javidi and A. Golbabai, "A new domain decomposition algorithm for generalized Burger's–Huxley equation based on Chebyshev polynomials and preconditioning," *Chaos, Solitons and Fractals*, vol. 39, no. 2, pp. 849–857, 2009.

[10] A. Kaushik, "Pointwise uniformly convergent numerical treatment for the non-stationary Burger–Huxley equation using grid equidistribution," *International Journal of Computer Mathematics*, vol. 84, no. 10, pp. 1527–1546, 2007.

[11] A. J. Khattak, "A computational meshless method for the generalized Burger's—Huxley equation," *Applied Mathematical Modelling*, vol. 33, no. 9, pp. 3718–3729, 2009.

[12] S. Tomasiello, "Numerical solutions of the Burger's–Huxley equation by the IDQ method," *International Journal of Computer Mathematics*, vol. 87, no. 1, pp. 129–140, 2010.

[13] R. Zhang, X. Yu, and G. Zhao, "The local discontinuous Galerkin method for Burger's–Huxley and Burger's–Fisher equations," *Applied Mathematics and Computation*, vol. 218, no. 17, pp. 8773–8778, 2012.

[14] Y. Duan, L. Kong, and R. Zhang, "A lattice Boltzmann model for the generalized Burgers–Huxley equation," *Physica A: Statistical Mechanics and Its Applications*, vol. 391, no. 3, pp. 625–632, 2012.

[15] R. C. Mittal and A. Tripathi, "Numerical solutions of generalized Burgers–Fisher and generalized Burgers–Huxley equations using collocation of cubic B-splines," *International Journal of Computer Mathematics*, vol. 92, no. 5, pp. 1053–1077, 2015.

[16] I. Celik, "Haar Wavelet method for solving generalized Burgers–Huxley equation," *Arab Journal of Mathematical Sciences*, vol. 18, no. 1, pp. 25–37, 2012.

[17] R. Mohammadi, "B-spline collocation algorithm for numerical solution of the generalized burger's-huxley equation," *Numerical Methods for Partial Differential Equations*, vol. 29, no. 4, pp. 1173–1191, 2013.

[18] M. Dehghan, B. N. Saray, and M. Lakestani, "Three methods based on the interpolation scaling functions and the mixed collocation finite difference schemes for the numerical solution of the nonlinear generalized Burgers-Huxley equation," *Mathematical and Computer Modelling*, vol. 55, no. 3-4, pp. 1129–1142, 2012.

[19] P. W. M. Chin, "The analysis of the solution of the Burgers–Huxley equation using the Galerkin method," *Numerical Methods for Partial Differential Equations*, vol. 39, no. 4, pp. 2787–2807, 2023.

[20] J. Goh, A. Abd Majid, and A. I. M. Ismail, "Cubic B-spline collocation method for one-dimensional heat and advection-diffusion equations," *Journal of Applied Mathematics*, vol. 2012, Article ID 458701, 8 pages, 2012.

[21] I. Dag and B. Saka, "A cubic B-spline collocation method for the EW equation," *Mathematical and Computational Applications*, vol. 9, no. 3, pp. 381–392, 2004.

[22] M. K. Kadalbajoo and P. Arora, "B-spline collocation method for the singular-perturbation problem using artificial viscosity," *Computers & Mathematics with Applications*, vol. 57, no. 4, pp. 650–663, 2009.

[23] W. K. Zahra, "Trigonometric B-spline collocation method for solving PHI-four and AllenCahn equations," *Mediterranean Journal of Mathematics*, vol. 14, no. 3, p. 122, 2017.

[24] O. Ersoy and I. Dag, "The exponential cubic B-spline collocation method for the Kuramoto–Sivashinsky equation," *Filomat*, vol. 30, no. 3, pp. 853–861, 2016.

[25] A. H. Khater, R. S. Temsah, and M. M. Hassan, "A Chebyshev spectral collocation method for solving Burgers-type equations," *Journal of Computational and Applied Mathematics*, vol. 222, no. 2, pp. 333–350, 2008.

[26] R. C. Mittal and S. Dahiya, "A study of quintic B-spline based differential quadrature method for a class of semi-linear Fisher-Kolmogorov equations," *Alexandria Engineering Journal*, vol. 55, no. 3, pp. 2893–2899, 2016.

[27] R. Rohila and R. C. Mittal, "Numerical study of reaction diffusion Fisher's equation by fourth order cubic B-spline collocation method," *Mathematical Sciences*, vol. 12, no. 2, pp. 79–89, 2018.

[28] A. Singh, S. Dahiya, and S. P. Singh, "A fourth-order B-spline collocation method for nonlinear Burgers–Fisher equation," *The Mathematical Scientist*, vol. 14, no. 1, pp. 75–85, 2020.

[29] P. Roul, "A high accuracy numerical method and its convergence for time-fractional Black-Scholes equation governing European options," *Applied Numerical Mathematics*, vol. 151, pp. 472–493, 2020.

[30] P. Roul, V. Rohil, G. Espinosa-Paredes, and K. Obaidurrahman, "An efficient computational technique for solving a fractional-order model describing dynamics of neutron flux in a nuclear reactor," *Annals of Nuclear Energy*, vol. 185, Article ID 109733, 2023.

[31] P. Roul, "A fourth order numerical method based on B-spline functions for pricing Asian options," *Computers and Mathematics with Applications*, vol. 80, no. 3, pp. 504–521, 2020.

[32] P. Roul, "A fourth-order non-uniform mesh optimal B-spline collocation method for solving a strongly nonlinear singular boundary value problem describing electrohydrodynamic flow of a fluid," *Applied Numerical Mathematics*, vol. 153, pp. 558–574, 2020.

[33] P. Roul and V. Prasad Goura, "A superconvergent B-spline technique for second order nonlinear boundary value problems," *Applied Mathematics and Computation*, vol. 414, Article ID 126615, 2022.

[34] P. Roul, V. M. K. P. Goura, and R. Cavoretto, "A numerical technique based on B-spline for a class of time-fractional diffusion equation," *Numerical Methods for Partial Differential Equations*, vol. 39, no. 1, pp. 45–64, 2023.

[35] J. Alavi, H. Aminikhah, and H. Aminikhah, "Numerical study of the inverse problem of generalized Burgers–Fisher and generalized burgers–huxley equations," *Advances in Mathematical Physics*, vol. 2021, Article ID 6652108, 15 pages, 2021.

[36] A. Singh, S. Dahiya, and S. P. Singh, "Numerical solution of Burgers'-Huxley equation using a higher order collocation method," 2023, https://www.researchsquare.com/article/rs-2648922/v1.

[37] A. G. Bratsos, "A fourth order improved numerical scheme for the Generalized Burger's–Huxley equation," *American*

Journal of Computational Mathematics, vol. 1, no. 3, pp. 152–158, 2011.

[38] T. R. Lucas, "Error bounds for interpolating cubic splines under various end conditions," *SIAM Journal on Numerical Analysis*, vol. 11, no. 3, pp. 569–584, 1974.

[39] J. Biazar and F. Mohammadi, "Application of differential transform method to the generalized Burgers–Huxley equation," *Applications and Applied Mathematics: An International Journal*, vol. 5, no. 10, pp. 1726–1740, 2010.

[40] M. R. Yaghouti and A. Zabihi, "Application of laplace decomposition method for burgers–huxley and Burgers–Fisher equations," *Journal of Mathematical Modeling*, vol. 1, no. 1, pp. 41–67, 2013.

[41] I. Wasim, M. Abbas, and M. Amin, "Hybrid B-spline collocation method for solving the generalized Burgers-Fisher and burgers-huxley equations," *Mathematical Problems in Engineering*, vol. 2018, Article ID 6143934, 18 pages, 2018.