Journal of Engineering Mathematics (2023) 138:13
https://doi.org/10.1007/s10665-022-10249-0

An efficient hybrid method based on cubic B-spline and fourth-order compact finite difference for solving nonlinear advection–diffusion–reaction equations

Seda Gulen[1]

Received: 7 June 2022 / Accepted: 8 November 2022 / Published online: 1 February 2023

Abstract
This paper proposes an efficient hybrid numerical method to obtain approximate solutions of nonlinear advection–diffusion–reaction (ADR) equations arising in real-world phenomena. The proposed method is based on finite differences for the approximation of time derivatives while a combination of cubic B-splines and a fourth-order compact finite difference scheme is used for spatial discretization with the help of the Crank–Nicolson method. Since desired accuracy and order of convergence cannot be reached using the traditional cubic B-spline method, to overcome this, the second-order derivatives are approximated using the unknowns and their first derivative approximations with the compact support. Thus, instead of expressing the second-order derivative in second-order accuracy, it is represented by the convergence of order four in the present method. The computed results revealed that this combined approach improves the accuracy of solutions of nonlinear ADR equations in comparison to up-to-date literature even using relatively larger step sizes. Besides, this method is seen to be capable of capturing the behavior of the models with very small viscosity values. The stability of the proposed method has been discussed by considering the matrix stability approach and it has been shown that the method is stable. In addition to the fact that the proposed method obtains sufficiently accurate solutions, another main superiority is its simplicity and applicability, which requires minimum computational effort.

Keywords Advection–diffusion–reaction equations · Cubic B-spline · Fourth-order compact finite difference

Mathematics Subject Classification 65M06 · 35G20

✉ Seda Gulen
sgulen@nku.edu.tr

[1] Department of Mathematics, Faculty of Arts and Science, Tekirdag Namik Kemal University, Tekirdağ, Turkey

1 Introduction

Nonlinear advection–diffusion–reaction (ADR) equations play an important role in representing various physical or biological phenomena arising in engineering and science. Therefore, researchers focus their attention on capturing the behaviors of these problems which is a challenging issue. Since obtaining an analytical solution for such problems is not easy due to the complexity of the transport process, researchers have attempted to investigate the behaviors of these problems by developing various numerical techniques. In this paper, a new hybrid numerical method has been proposed for the following ADR equation:

$$u_t + \alpha u^{\mu} u_x - v u_{xx} = F(u), \quad a \le x \le b, \quad t \ge 0, \tag{1}$$

subject to the initial condition

$$u(x, 0) = g(x), \quad a \le x \le b, \tag{2}$$

and the boundary conditions

$$u(a, t) = g_1(t), \tag{3}$$
$$u(b, t) = g_2(t), \quad 0 \le t \le T, \tag{4}$$

where v, α, and $F(u)$ are the eddy viscosity, the velocity component of the fluid, and source/sink in terms of u, respectively. The terms $g(x)$, $g_1(t)$, and $g_2(t)$ are known functions and the subscripts x and t represent differentiations with respect to space and time, respectively. The advection term u_x depicts the transportation of the quantity u by the velocity field. The diffusion term u_{xx} describes the dissemination of the quantity u.

In the literature, various numerical techniques based on finite difference methods [1, 2], Galerkin methods [3, 4], spectral collocation methods [5], and decomposition methods [6, 7] have been developed for numerical simulation of nonlinear ADR equations up to now. In addition, the aim of accurately capturing the behavior of these problems encountered in various fields of science and engineering has always prompted researchers to develop new various techniques such as a lattice Boltzmann method [8], high-order splitting methods [9], a finite volume and Fourier pseudospectral method [10], 2N order compact finite difference method [11], a meshless semi-analytical collocation technique [12], and a new heuristic method [13].

In the last decades, B-spline functions have attracted the attention of many researchers to find out effective numerical solutions to linear and nonlinear partial differential equations. The cubic B-spline collocation methods which use B-splines as basis functions are well-known efficient approximation methods. These methods with some special properties such as smoothness, local support of spline curve, good approximation rate, computationally fast, and numerical consistency are convenient to apply to differential equations. These methods are also able to approximate analytical solutions up to a certain smoothness [14]. Therefore, spline methods provide

the flexibility to get the approximation at any point in the domain with more accurate results compared to some other standard methods. The earliest studies about the theory of standard cubic B-spline methods, their properties, and some applications to ordinary and partial differential equations can be found in the books of Prenter [15] and Rubin [16]. More recently, various efficient methods based on various versions of B-splines and combined with other numerical techniques have been developed to capture the behavior of ADR equations accurately. For instance, hybrid B-spline collocation methods based on possessing a free parameter were proposed for various forms of ADR equations by Wasim et al. [14] and for a generalized nonlinear Klien–Gordon equation by Mat Zin [17]. Although those remarkable methods represented the behavior of the problems successfully, the lack of a technique for predicting an optimal parameter can be a drawback for those methods. In the study of Mittal and Jain [18], the cubic B-spline functions have needed to be modified into a set of basis functions for handling their problems. In addition to those studies, various kinds of B-spline collocation methods were also applied to deal with several nonlinear ADR equations [19–26]. Despite many advantages of the cubic B-spline collocation methods, the desired accuracy and order of convergence cannot be achieved compared to some competitors in solving differential equations. To overcome this, they are combined with other robust techniques in the literature such as compact finite difference methods which become one of the most popular techniques for solving partial differential equations arising in many applications. Compact finite difference methods have higher accuracy even when using larger mesh sizes. Thus, the system of equations has a small bandwidth coefficient matrix that can be solved efficiently. Besides, high-order compact finite difference methods consider not only the value of the function but also those of its first derivatives as unknowns at each discretization point [27]. Although the solutions of the ADR equations can be obtained through some effective numerical techniques, in terms of the aforementioned reasons, the need of obtaining flexible and computationally efficient solutions of the problems where it is difficult to control precision and stability motivated us to propose a new hybrid method based on combining with the standard cubic B-spline and a fourth-order compact finite difference method.

In this paper, a new numerical scheme by combining the standard cubic B-spline and a fourth-order compact finite difference scheme is proposed to capture the behavior of the Burgers, Burgers–Fisher, Burgers–Huxley, and Fisher's equations. Firstly, these equations have been discretized by the Crank–Nicolson and the forward finite difference methods for spatial and temporal domains, respectively, and then, the combined method has been applied for spatial derivatives. In this technique, spatial second derivative approximations are approached by the fourth-order compact finite difference scheme in which second derivative approximations of the unknowns are eliminated with the unknowns themselves and their first derivative approximations while retaining the fourth-order accuracy. Hence, the second-order derivative is represented by the convergence of order four by this approach, while it has second-order accuracy in the standard cubic B-spline method. Thus, with the aforementioned advantages of the cubic B-spline and fourth-order compact finite difference methods, the combination of these methods aims to pull up the capacity of the algorithm to achieve higher accuracy with minimal computational effort. The computed results for each equation have been evaluated in terms of accuracy, flexibility, and computational efficiency

depending on the dynamics of the problems with their different set of parameters by comparing them with up-to-date techniques in the literature. The calculations for the strong form of the Burgers equations revealed that the proposed method is in very good agreement with the exact solution and that the proposed method is superior to the cubic B-spline collocation method presented in the study of Dag et al. [28] and other compared techniques in the literature even when using fewer spatial and temporal elements. Moreover, the current scheme can capture the behavior of the model with very small viscosity efficiently and accurately. For the Burgers-Fisher and Burgers-Huxley equations, considerable improvement has been reached even if sometimes it is at a moderate level even using fewer spatial or temporal elements which reduces the computational complexity, CPU time, and memory space. Finally, implementing the proposed method to the Fisher's equation has demonstrated that the current scheme produces more accurate solutions than the standard cubic B-spline method [29] and the wavelet-Galerkin approach [30]. Besides, the current scheme solutions for the Fisher's equation have been found to be in reasonable agreement with other compared methods and is successful to represent the behavior of the problem having small viscosity. In addition, the proposed method provides much flexibility to implement complicated problems by importing second derivative terms with a high-order compact finite difference formula. The stability analysis of the method has been discussed by the matrix stability analysis and concluded that the present method is stable. Furthermore, the current scheme is quite easy to produce computer codes in any programming language and has not been implemented in any other study including ADR equations so far.

2 The proposed method

In this section, a new combining method based on the cubic B-spline and fourth-order compact finite difference methods will be constructed for the nonlinear ADR equations. Cubic B-splines are chosen as the basis function in the collocation methods due to the higher smoothness and sparse nature of matrices corresponding to splines. These methods which have some special properties such as smoothness, good approximation rate, computationally fast, numerically consistent, and ability to produce the shape of the data with the second order of continuity produce efficient algorithms for obtaining solutions of partial differential equations. Another important advantage of this method is that it is able to approximate the analytical curve up to a certain smoothness, providing the flexibility to obtain an approximation at any point in the domain [14]. In addition, the fourth-order compact finite difference method used in the proposed scheme is preferred in terms of solving various differential equations with providing low computational effort by using a small number of grid points.

Now, to construct the new combined method, let us introduce the cubic B-spline and the fourth-order compact finite difference method and derive the second-order derivative approximation which includes the unknowns themselves and their first derivatives only.

Firstly, a uniform partition of the domain $[a, b] \times [0, T]$ is considered by the knots $x_i, i = 0, 1, 2 \ldots, N$ and $t^n, n = 0, 1, 2, \ldots, M$ such that $x_i = a + ih$ and $t^n = n\mathrm{d}t$, $h = \frac{b-a}{N}$ and $\mathrm{d}t = \frac{T}{M}$. The numerical solutions of the nonlinear ADR problem (1)–(4)

are approximated by the cubic B-spline interpolation as follows:

$$u(x,t) \approx V_N(x,t) = \sum_{i=-1}^{N+1} \delta_i(t) B_i(x), \tag{5}$$

where δ_i is time-dependent parameter and B_i represents the well-known cubic B-spline functions given the following relationship [15]:

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

where $h = x_{i+1} - x_i, i = -1, \ldots, N+1$. The variation of $V_N(x,t)$ over typical element $[x_i, x_{i+1}]$ is expressed by

$$V_N(x,t) = \sum_{j=i-1}^{i+2} \delta_j(t) B_j(x). \tag{6}$$

By using the interpolating conditions, the values at the knots of $V(x,t)$ and its two derivatives $V'(x,t)$ and $V''(x,t)$ at the knots are stated in terms of time-dependent parameters δ_i as follows:

$$\begin{aligned} V_i &= V(x_i) = \delta_{i-1} + 4\delta_i + \delta_{i+1}, \\ V_i^{'} &= V'(x_i) = \frac{3}{h} (\delta_{i+1} - \delta_{i-1}), \\ V_i'' &= V''(x_i) = \frac{6}{h^2} (\delta_{i-1} - 2\delta_i + \delta_{i+1}). \end{aligned} \tag{7}$$

The space and time derivatives in Eq. (1) are discretized by using the Crank–Nicolson and the finite difference schemes:

$$\frac{V^{n+1} - V^n}{\mathrm{d}t} + \alpha \frac{(V^\mu V_x)^{n+1} + (V^\mu V_x)^n}{2} - \upsilon \frac{V_{xx}^{n+1} + V_{xx}^n}{2} = \frac{F^{n+1}(V) + F^n(V)}{2}. \tag{8}$$

The nonlinear terms in Eq. (8) can be linearized by using the following formula [16]

$$(V^\mu V_x)^{n+1} = \left(V^\mu\right)^n V_x^{n+1} + \mu (V^{\mu-1})^n V_x^n V^{n+1} - \mu (V^\mu)^n V_x^n.$$

Hence, Eq. (8) transforms to the following form:

$$\frac{V^{n+1}-V^n}{\mathrm{d}t}+\alpha\frac{(V^\mu)^n V_x^{n+1}+\mu(V^{\mu-1})^n V_x^n V^{n+1}-\mu(V^\mu)^n V_x^n+(V^\mu V_x)^n}{2}$$
$$-\upsilon\frac{V_{xx}^{n+1}+V_{xx}^n}{2}=\frac{F^{n+1}(V)+F^n(V)}{2}. \tag{9}$$

Now, to construct the new proposed scheme, let us introduce the fourth-order compact finite difference approximation for the first and second derivatives, respectively, as follows:

$$\frac{1}{4}V'_{i-1}+V'_i+\frac{1}{4}V'_{i+1}=\frac{1}{h}\left[-\frac{3}{4}V_{i-1}+\frac{3}{4}V_{i+1}\right], \tag{10}$$

$$\frac{1}{10}V''_{i-1}+V''_i+\frac{1}{10}V''_{i+1}=\frac{1}{h^2}\left[\frac{6}{5}V_{i-1}-\frac{12}{5}V_i+\frac{6}{5}V_{i+1}\right], \tag{11}$$

where V'_i and V''_i are the first- and second-order derivative approximations of unknown V at point x_i, respectively. In Eq. (10), the second-order derivative terms can be obtained by applying the first operator again:

$$\frac{1}{4}V''_{i-1}+V''_i+\frac{1}{4}V''_{i+1}=\frac{1}{h}\left[-\frac{3}{4}V'_{i-1}+\frac{3}{4}V'_{i+1}\right]. \tag{12}$$

Eliminating V''_{i-1} and V''_{i+1} from Eqs. (11) and (12), the second derivative is obtained as the following form [27]:

$$V''_i=2\frac{V_{i+1}-2V_i+V_{i-1}}{h^2}-\frac{V'_{i+1}-V'_{i-1}}{2h}. \tag{13}$$

Thus, the second derivative that is constructed by unknowns themselves and their first derivatives only has been represented by the convergence of the fourth order by this formula in the proposed method. Thus, in the solution of Eq. (9) while Eq. (7) is used for approximating unknown V and its first and second derivatives for boundary points, Eq. (13) will be used instead of the second derivative given in Eq. (7) for interior points.

3 Implementation of the method

In this section, the proposed method will be implemented for the Burgers, Burgers–Fisher, Burgers–Huxley, and Fisher's equations.

3.1 Algorithm for the Burgers equation

Equation (1) is called the well-known Burgers equation when $F(u)=0$ and $\alpha=\mu=1$. Considering Eq. (9) that is the discretization form of Eq. (1) for the Burgers equation

and rearranging the terms and simplifying them, it is expressed as the following form:

$$V^{n+1}\left\{1+\frac{\mathrm{d}t}{2}V_x^n\right\}+\frac{\mathrm{d}t}{2}V^nV_x^{n+1}-v\frac{\mathrm{d}t}{2}V_{xx}^{n+1}=V^n+v\frac{\mathrm{d}t}{2}V_{xx}^n. \tag{14}$$

Substituting Eq. (7) into Eq. (14) for the boundary points, $i=0, N$, it is obtained

$$\alpha_1\delta_{i-1}^{n+1}+\alpha_2\delta_i^{n+1}+\alpha_3\delta_{i+1}^{n+1}=\alpha_4\delta_{i-1}^n+\alpha_5\delta_i^n+\alpha_6\delta_{i+1}^n, \tag{15}$$

where

$$\begin{aligned}
&\alpha_1=1+\frac{\mathrm{d}t}{2}V_x^n-\frac{3\mathrm{d}t}{2h}V^n-3v\frac{\mathrm{d}t}{h^2},\\
&\alpha_2=4+2\mathrm{d}tV_x^n+6v\frac{\mathrm{d}t}{h^2},\quad \alpha_3=1+\frac{\mathrm{d}t}{2}V_x^n+\frac{3\mathrm{d}t}{2h}V^n-3v\frac{\mathrm{d}t}{h^2},\\
&\alpha_4=1+3v\frac{\mathrm{d}t}{h^2},\quad \alpha_5=4-6v\frac{\mathrm{d}t}{h^2},\quad \alpha_6=1+3v\frac{\mathrm{d}t}{h^2}.
\end{aligned} \tag{16}$$

Substituting the approximate value V and its first derivative V' in Eq. (7) and second-order derivative V'' in Eq. (13) into Eq. (14) for interior points, $i=1,\ldots,N-1$, it is obtained

$$\begin{aligned}
&\beta_1\delta_{i-2}^{n+1}+\beta_2\delta_{i-1}^{n+1}+\beta_3\delta_i^{n+1}+\beta_4\delta_{i+1}^{n+1}+\beta_5\delta_{i+2}^{n+1}=\beta_6\delta_{i-2}^n+\beta_7\delta_{i-1}^n\\
&+\beta_8\delta_i^n+\beta_9\delta_{i+1}^n+\beta_{10}\delta_{i+2}^n,
\end{aligned} \tag{17}$$

where

$$\begin{aligned}
&\beta_1=-v\frac{\mathrm{d}t}{4h^2},\quad \beta_2=1+\frac{\mathrm{d}t}{2}V_x^n-\frac{3\mathrm{d}t}{2h}V^n-2v\frac{\mathrm{d}t}{h^2},\quad \beta_3=4+2\mathrm{d}tV_x^n+\frac{9}{2}v\frac{\mathrm{d}t}{h^2},\\
&\beta_4=1+\frac{\mathrm{d}t}{2}V_x^n+\frac{3\mathrm{d}t}{2h}V^n-2v\frac{\mathrm{d}t}{h^2},\quad \beta_5=\beta_1,\quad \beta_6=-\beta_1,\quad \beta_7=1+2v\frac{\mathrm{d}t}{h^2},\\
&\beta_8=4-\frac{9}{2}v\frac{\mathrm{d}t}{h^2},\quad \beta_9=\beta_7,\quad \beta_{10}=-\beta_1.
\end{aligned} \tag{18}$$

3.2 Algorithm for the Burgers–Fisher equation

Equation (1) is called the Burgers–Fisher equation when $F(u)=\beta u(1-u^\mu)$ and $v=1$. These terms are evaluated in Eq. (9) and rearranging the other terms by using linearization and simplifications, it is obtained

$$\begin{aligned}
&V^{n+1}\left\{1+\alpha\frac{\mathrm{d}t}{2}\mu(V^{\mu-1})^nV_x^n-\beta\frac{\mathrm{d}t}{2}+\beta\frac{\mathrm{d}t}{2}(1+\mu)\left(V^\mu\right)^n\right\}+\alpha\frac{\mathrm{d}t}{2}(V^\mu)^nV_x^{n+1}-\frac{\mathrm{d}t}{2}V_{xx}^{n+1}\\
&=V^n\left\{1+\beta\frac{\mathrm{d}t}{2}-\beta\frac{\mathrm{d}t}{2}(1-\mu)\left(V^\mu\right)^n\right\}+\alpha\frac{\mathrm{d}t}{2}(\mu-1)(V^\mu)^nV_x^n+\frac{\mathrm{d}t}{2}V_{xx}^n.
\end{aligned} \tag{19}$$

Considering Eq. (7) into Eq. (19), Eq. (15) is obtained for the boundary points, $i = 0, N$, and its coefficients are given as follows:

$$\begin{aligned}
\alpha_1 &= 1+\alpha\frac{\mathrm{d}t}{2}\mu L_1^{\mu-1}L_2-\beta\frac{\mathrm{d}t}{2}+\beta\frac{\mathrm{d}t}{2}(1+\mu)L_1^{\mu}-\alpha\frac{3\,\mathrm{d}t}{2h}L_1^{\mu}-3\frac{\mathrm{d}t}{h^2},\\
\alpha_2 &= 4+2\alpha\,\mathrm{d}t\mu L_1^{\mu-1}L_2-2\beta\,\mathrm{d}t+2\beta\,\mathrm{d}t\,(1+\mu)\,L_1^{\mu}+6\frac{\mathrm{d}t}{h^2},\\
\alpha_3 &= 1+\alpha\frac{\mathrm{d}t}{2}\mu L_1^{\mu-1}L_2-\beta\frac{\mathrm{d}t}{2}+\beta\frac{\mathrm{d}t}{2}(1+\mu)\,L_1^{\mu}+\alpha\frac{3\,\mathrm{d}t}{2h}L_1^{\mu}-3\,\frac{\mathrm{d}t}{h^2},\\
\alpha_4 &= 1+\beta\frac{\mathrm{d}t}{2}-\beta\,\frac{\mathrm{d}t}{2}(1-\mu)\,L_1^{\mu}-\alpha\frac{3\,\mathrm{d}t}{2h}(\mu-1)\,L_1^{\mu}+3\frac{\mathrm{d}t}{h^2},\\
\alpha_5 &= 4+2\beta\,\mathrm{d}t-2\beta\,\mathrm{d}t\,(1-\mu)\,L_1^{\mu}-6\frac{\mathrm{d}t}{h^2},\\
\alpha_6 &= 1+\beta\frac{\mathrm{d}t}{2}-\beta\frac{\mathrm{d}t}{2}(1-\mu)\,L_1^{\mu}+\alpha\frac{3\,\mathrm{d}t}{2h}(\mu-1)L_1^{\mu}+3\frac{\mathrm{d}t}{h^2}.
\end{aligned}\tag{20}$$

Evaluating Eq. (7) for approximate value V and its first derivative V' and Eq. (13) for second derivative V'' into Eq. (19), Eq. (17) is obtained for interior points, $i = 1, \ldots, N-1$, and its coefficients are given by

$$\begin{aligned}
&\beta_1 = -\frac{\mathrm{d}t}{4h^2},\ \beta_2 = 1+\alpha\frac{\mathrm{d}t}{2}\mu L_1^{\mu-1}L_2-\beta\frac{\mathrm{d}t}{2}+\beta\frac{\mathrm{d}t}{2}(1+\mu)L_1^{\mu}-\alpha\frac{3\,\mathrm{d}t}{2h}L_1^{\mu}-\frac{2\,\mathrm{d}t}{h^2},\\
&\beta_3 = 4+2\alpha\,\mathrm{d}t\mu L_1^{\mu-1}L_2-2\beta\,\mathrm{d}t+2\beta\,\mathrm{d}t(1+\mu)L_1^{\mu}-\frac{9}{2}\frac{\mathrm{d}t}{h^2},\\
&\beta_4 = 1+\alpha\frac{\mathrm{d}t}{2}\mu L_1^{\mu-1}L_2-\beta\frac{\mathrm{d}t}{2}+\beta\frac{\mathrm{d}t}{2}(1+\mu)L_1^{\mu}+\alpha\frac{3\,\mathrm{d}t}{2h}L_1^{\mu}-2\frac{\mathrm{d}t}{h^2},\\
&\beta_5 = \beta_1,\quad \beta_6 = -\beta_1,\\
&\beta_7 = 4+2\beta\,\mathrm{d}t-2\beta\,\mathrm{d}t(1-\mu)L_1^{\mu}-\frac{9}{2}\frac{\mathrm{d}t}{h^2},\\
&\beta_8 = 1+\beta\frac{\mathrm{d}t}{2}-\beta\frac{\mathrm{d}t}{2}(1-\mu)L_1^{\mu}-\alpha\frac{3\,\mathrm{d}t}{2h}(\mu-1)L_1^{\mu}+2\frac{\mathrm{d}t}{h^2},\\
&\beta_9 = 1+\beta\frac{\mathrm{d}t}{2}-\beta\frac{\mathrm{d}t}{2}(1-\mu)L_1^{\mu}+\alpha\frac{3\,\mathrm{d}t}{2h}(\mu-1)L_1^{\mu}+2\frac{\mathrm{d}t}{h^2},\ \beta_{10} = -\beta_1.
\end{aligned}\tag{21}$$

3.3 Algorithm for the Burgers–Huxley equation

Equation (1) is called the Burgers–Huxley equation when $F(u) = \beta u(1-u^{\mu})(u^{\mu}-\gamma)$ and $v = 1$. Considering these terms into Eq. (9) and after the linearization and simplifications, Eq. (9) transforms to the following form:

$$\begin{aligned}
V^{n+1}\Big\{1+\alpha\frac{\mathrm{d}t}{2}\mu(V^{\mu-1})^nV_x^n-\beta\frac{\mathrm{d}t}{2}(\mu+1)(V^{\mu})^n+\gamma\beta\frac{\mathrm{d}t}{2}+\beta\frac{\mathrm{d}t}{2}(2\mu+1)(V^{2\mu})^n\\
-\beta\gamma\frac{\mathrm{d}t}{2}(\mu+1)(V^{\mu})^n\Big\}+\ \alpha\frac{\mathrm{d}t}{2}(V^{\mu})^nV_x^{n+1}-\frac{\mathrm{d}t}{2}V_{xx}^{n+1}
\end{aligned}$$

$$= V^n \left\{ 1 + \beta \frac{\mathrm{d}t}{2}(1-\mu)(V^{\mu})^n - \gamma\beta\frac{\mathrm{d}t}{2} - \beta\frac{\mathrm{d}t}{2}(1-2\mu)(V^{2\mu})^n \right.$$
$$\left. + \beta\gamma\frac{\mathrm{d}t}{2}(1-\mu)(V^{\mu})^n \right\} + \frac{\mathrm{d}t}{2}(V_{xx})^n - \alpha\frac{\mathrm{d}t}{2}(1-\mu)(V^{\mu})^n V_x^n. \tag{22}$$

Evaluating Eq. (7) for approximate value V and its first derivative V' and Eq. (13) for second derivative V'' into Eq. (22), Eq. (15) is obtained for the boundary points, $i = 0, N$, and its coefficients are obtained in the following form:

$$\alpha_1 = 1 + \alpha\frac{\mathrm{d}t}{2}\mu L_1^{\mu-1}L_2 - \beta\frac{\mathrm{d}t}{2}(\mu+1)L_1^{\mu} + \beta\gamma\frac{\mathrm{d}t}{2} + \beta\frac{\mathrm{d}t}{2}(2\mu+1)L_1^{2\mu}$$
$$-\beta\gamma\frac{\mathrm{d}t}{2}(\mu+1)L_1^{\mu} - \alpha\frac{3\,\mathrm{d}t}{2h}L_1^{\mu} - 3\frac{\mathrm{d}t}{h^2},$$
$$\alpha_2 = 4 + 2\alpha\,\mathrm{d}t\mu L_1^{\mu-1}L_2 - 2\beta\,\mathrm{d}t(\mu+1)L_1^{\mu} + 2\gamma\beta\,\mathrm{d}t + 2\beta\,\mathrm{d}t(2\mu+1)L_1^{2\mu}$$
$$-2\beta\gamma\,\mathrm{d}t(\mu+1)L_1^{\mu} + 6\frac{\mathrm{d}t}{h^2},$$
$$\alpha_3 = 1 + \alpha\frac{\mathrm{d}t}{2}\mu L_1^{\mu-1}L_2 - \beta\frac{\mathrm{d}t}{2}(\mu+1)L_1^{\mu} + \beta\gamma\frac{\mathrm{d}t}{2} + \beta\frac{\mathrm{d}t}{2}(2\mu+1)L_1^{2\mu}$$
$$-\beta\gamma\frac{\mathrm{d}t}{2}(\mu+1)L_1^{\mu} + \alpha\frac{3\,\mathrm{d}t}{2h}L_1^{\mu} - 3\frac{\mathrm{d}t}{h^2},$$
$$\alpha_4 = 1 + \beta\frac{\mathrm{d}t}{2}(1-\mu)L_1^{\mu} - \gamma\beta\frac{\mathrm{d}t}{2} - \beta\frac{\mathrm{d}t}{2}(1-2\mu)L_1^{2\mu} + \beta\gamma\frac{\mathrm{d}t}{2}(1-\mu)L_1^{\mu}$$
$$+\alpha\frac{3\,\mathrm{d}t}{2h}(1-\mu)L_1^{\mu} + 3\frac{\mathrm{d}t}{h^2},$$
$$\alpha_5 = 4 + 2\beta\,\mathrm{d}t(1-\mu)L_1^{\mu} - 2\gamma\beta\,\mathrm{d}t - 2\beta\,\mathrm{d}t(1-2\mu)L_1^{2\mu}$$
$$+2\beta\gamma\,\mathrm{d}t(1-\mu)L_1^{\mu} - 6\frac{\mathrm{d}t}{h^2},$$
$$\alpha_6 = 1 + \beta\frac{\mathrm{d}t}{2}(1-\mu)L_1^{\mu} - \beta\gamma\frac{\mathrm{d}t}{2} - \beta\frac{\mathrm{d}t}{2}(1-2\mu)L_1^{2\mu} + \beta\gamma\frac{\mathrm{d}t}{2}(1-\mu)L_1^{\mu}$$
$$-\alpha\frac{3\,\mathrm{d}t}{2h}(1-\mu)L_1^{\mu} + 3\frac{\mathrm{d}t}{h^2}. \tag{23}$$

For the interior points, the coefficients of Eq. (17) for the Burgers–Huxley equation are given by

$$\beta_1 = -\frac{\mathrm{d}t}{4h^2},$$
$$\beta_2 = 1 + \frac{\alpha\,\mathrm{d}t}{2}\mu L_1^{\mu-1}L_2 - \beta\frac{\mathrm{d}t}{2}(\mu+1)L_1^{\mu} + \beta\gamma\frac{\mathrm{d}t}{2} + \beta\frac{\mathrm{d}t}{2}(2\mu+1)L_1^{2\mu}$$
$$-\beta\gamma\frac{\mathrm{d}t}{2}(\mu+1)L_1^{\mu} - \alpha\frac{3\,\mathrm{d}t}{2h}L_1^{\mu} - 2\frac{\mathrm{d}t}{h^2},$$
$$\beta_3 = 4 + 2\alpha\,\mathrm{d}t\mu L_1^{\mu-1}L_2 - 2\beta\,\mathrm{d}t(\mu+1)L_1^{\mu} + 2\gamma\beta\,\mathrm{d}t + 2\beta\,\mathrm{d}t(2\mu+1)L_1^{2\mu}$$
$$-2\gamma\beta\,\mathrm{d}t(\mu+1)L_1^{\mu} + \frac{9\,\mathrm{d}t}{2h^2},$$

$$\begin{aligned}
\beta_4 &= 1+\alpha\frac{\mathrm{d}t}{2}\mu L_1^{\mu-1}L_2-\beta\frac{\mathrm{d}t}{2}(\mu+1)L_1^{\mu}+\beta\gamma\frac{\mathrm{d}t}{2}+\beta\frac{\mathrm{d}t}{2}(2\mu+1)L_1^{2\mu}\\
&\quad -\beta\gamma\frac{\mathrm{d}t}{2}(\mu+1)L_1^{\mu}+\alpha\frac{3\,\mathrm{d}t}{2h}L_1^{\mu}-2\frac{\mathrm{d}t}{h^2},\\
\beta_5 &= \beta_1,\quad \beta_6=-\beta_1,\\
\beta_7 &= 1+\beta\frac{\mathrm{d}t}{2}(1-\mu)L_1^{\mu}-\gamma\beta\frac{\mathrm{d}t}{2}-\beta\frac{\mathrm{d}t}{2}(1-2\mu)L_1^{2\mu}+\gamma\beta\frac{\mathrm{d}t}{2}(1-\mu)L_1^{\mu}\\
&\quad +\alpha\frac{3\,\mathrm{d}t}{2h}(1-\mu)L_1^{\mu}+2\frac{\mathrm{d}t}{h^2},\\
\beta_8 &= 4+2\beta\,\mathrm{d}t(1-\mu)L_1^{\mu}-2\gamma\beta\,\mathrm{d}t-2\beta\,\mathrm{d}t(1-2\mu)L_1^{2\mu}\\
&\quad +2\gamma\beta\,\mathrm{d}t(1-\mu)L_1^{\mu}-\frac{9\,\mathrm{d}t}{h^2},\\
\beta_9 &= 1+\beta\frac{\mathrm{d}t}{2}(1-\mu)L_1^{\mu}-\gamma\beta\frac{\mathrm{d}t}{2}-\beta\frac{\mathrm{d}t}{2}(1-2\mu)L_1^{2\mu}+\gamma\beta\frac{\mathrm{d}t}{2}(1-\mu)L_1^{\mu}\\
&\quad -\alpha\frac{3\,\mathrm{d}t}{2h}(1-\mu)L_1^{\mu}+2\frac{\mathrm{d}t}{h^2},\\
\beta_{10} &= -\beta_1.
\end{aligned}\tag{24}$$

3.4 Algorithm for the Fisher's equation

Equation (1) is called the Fisher's equation when $\alpha = 0$ and $F(u) = u\,(\beta - \gamma u)$. Evaluating Eq. (9) with these terms, it is obtained

$$V^{n+1}\left\{1-\beta\frac{\mathrm{d}t}{2}+\gamma\,\mathrm{d}t\,V^n\right\}-v\frac{\mathrm{d}t}{2}V_{xx}^{n+1}=V^n\left\{1+\beta\frac{\mathrm{d}t}{2}\right\}+v\frac{\mathrm{d}t}{2}V_{xx}^{n}.\tag{25}$$

Substituting Eq. (7) for approximate value V and its first derivative V' and Eq. (13) for second derivative V'' into Eq. (25), Eq. (15) is obtained for the boundary points, $i = 0, N$, and its coefficients are as follows:

$$\begin{aligned}
\alpha_1 &= 1-\beta\frac{\mathrm{d}t}{2}+\gamma\,\mathrm{d}t\,L_1-3v\frac{\mathrm{d}t}{h^2}, & \alpha_2 &= 4-2\beta\,\mathrm{d}t+4\gamma\,\mathrm{d}t\,L_1+6v\frac{\mathrm{d}t}{h^2},\\
\alpha_3 &= 1-\beta\frac{\mathrm{d}t}{2}+\gamma\,\mathrm{d}t\,L_1-3v\frac{\mathrm{d}t}{h^2}, & \alpha_4 &= 1+\beta\frac{\mathrm{d}t}{2}+3v\frac{\mathrm{d}t}{h^2},\\
\alpha_5 &= 4+2\beta\,\mathrm{d}t-6v\frac{\mathrm{d}t}{h^2}, & \alpha_6 &= 1+\beta\frac{\mathrm{d}t}{2}+3v\frac{\mathrm{d}t}{h^2}.
\end{aligned}$$

For interior points, $i = 1, \ldots, N-1$, Eq. (17) is obtained and its coefficients are given by

$$\begin{aligned}
&\beta_1 = -v\frac{\beta}{4h^2}, \quad \beta_2 = 1-\beta\frac{\mathrm{d}t}{2}+\gamma\,\mathrm{d}t\,L_1-2v\frac{\mathrm{d}t}{h^2}, \quad \beta_3 = 4-2\beta\,\mathrm{d}t+4\gamma\,\mathrm{d}t\,L_1+v\frac{9\,\mathrm{d}t}{2h^2},\\
&\beta_4 = 1-\beta\frac{\mathrm{d}t}{2}+\gamma\,\mathrm{d}t\,L_1-2v\frac{\mathrm{d}t}{h^2}, \quad \beta_5=\beta_1, \quad \beta_6=-\beta_1, \quad \beta_7 = 1+\beta\frac{\mathrm{d}t}{2}+2v\frac{\mathrm{d}t}{h^2},\\
&\beta_8 = 4+2\beta\,\mathrm{d}t-v\frac{9\,\mathrm{d}t}{2h^2}, \quad \beta_9 = 1+\beta\frac{\mathrm{d}t}{2}+2v\frac{\mathrm{d}t}{h^2}, \quad \beta_{10}=-\beta_1.
\end{aligned}$$

Here, $L_1 = \delta_{i-1}^n + 4\delta_i^n + \delta_{i+1}^n$, $L_2 = \frac{3}{h}\left(\delta_{i+1}^n - \delta_{i-1}^n\right)$.

Note that Eqs. (15) and (17) generate a system of $(N+1)$ equations in $(N+3)$ unknowns $d^n = \left\{\delta_{-1}^n, \delta_0^n, \delta_1^n, \ldots, \delta_N^n, \delta_{N+1}^n\right\}$ for the Burgers, Burgers–Fisher, Burgers–Huxley, and Fisher's equations as follows:

$$\begin{aligned}
&\alpha_1\delta_{-1}^{n+1} + \alpha_2\delta_0^{n+1} + \alpha_3\delta_1^{n+1} = \alpha_4\delta_{-1}^n + \alpha_5\delta_0^n + \alpha_6\delta_1^n, \quad i = 0,\\
&\beta_1\delta_{i-2}^{n+1} + \beta_2\delta_{i-1}^{n+1} + \beta_3\delta_i^{n+1} + \beta_4\delta_{i+1}^{n+1} + \beta_5\delta_{i+1}^{n+1}\\
&\quad = \beta_6\delta_{i-2}^n + \beta_7\delta_{i-1}^n + \beta_8\delta_i^n + \beta_9\delta_{i+1}^n + \beta_{10}\delta_{i+2}^n, \quad i = 1, \ldots, N-1,\\
&\alpha_1\delta_{N-1}^{n+1} + \alpha_2\delta_N^{n+1} + \alpha_3\delta_{N+1}^{n+1} = \alpha_4\delta_{N-1}^n + \alpha_5\delta_N^n + \alpha_6\delta_{N+1}^n, \quad i = N.
\end{aligned} \tag{26}$$

System (26) can be converted into the following matrix form:

$$A\boldsymbol{d}^{n+1} = B\boldsymbol{d}^n, \tag{27}$$

where

$$A = A_{ij} = \begin{bmatrix}
\alpha_1 & \alpha_2 & \alpha_3 & & & & & & \\
\beta_1 & \beta_2 & \beta_3 & \beta_4 & \beta_5 & & & & \\
 & \beta_1 & \beta_2 & \beta_3 & \beta_4 & \beta_5 & & & \\
 & & \ddots & \ddots & \ddots & \ddots & \ddots & & \\
 & & & & \beta_1 & \beta_2 & \beta_3 & \beta_4 & \beta_5 \\
 & & & & & & \alpha_1 & \alpha_2 & \alpha_3
\end{bmatrix},$$

$$B = B_{ij} = \begin{bmatrix}
\alpha_4 & \alpha_5 & \alpha_6 & & & & & & \\
\beta_6 & \beta_7 & \beta_8 & \beta_9 & \beta_{10} & & & & \\
 & \beta_6 & \beta_7 & \beta_8 & \beta_9 & \beta_{10} & & & \\
 & & \ddots & \ddots & \ddots & \ddots & \ddots & & \\
 & & & & \beta_6 & \beta_7 & \beta_8 & \beta_9 & \beta_{10} \\
 & & & & & & \alpha_4 & \alpha_5 & \alpha_6
\end{bmatrix} \text{ and}$$

$$\boldsymbol{d}^n = (\delta_{-1}, \delta_0, \delta_1, \ldots, \delta_N, \delta_{N+1})^{\mathrm{T}}.$$

To obtain a consistent system, the δ_{-1}^n and δ_{N+1}^n are eliminated using the boundary conditions:

$$\begin{aligned}
U(x_0) &= \delta_{-1}^{n+1} + 4\delta_0^{n+1} + \delta_1^{n+1} = g_1(t^{n+1}) \Rightarrow \delta_{-1}^{n+1} = g_1(t^{n+1}) - 4\delta_0^{n+1} - \delta_1^{n+1},\\
U(x_N) &= \delta_{N-1}^{n+1} + 4\delta_N^{n+1} + \delta_{N+1}^{n+1} = g_2(t^{n+1}) \Rightarrow \delta_{N+1}^{n+1} = g_2(t^{n+1}) - 4\delta_N^{n+1} - \delta_{N-1}^{n+1}.
\end{aligned} \tag{28}$$

Substituting equalities (28) into the combined system (26), it is obtained $(N+1) \times (N+1)$ matrix system as follows:

$$\tilde{A}\boldsymbol{d^{n+1}} = \tilde{B}\boldsymbol{d^n} + b, \tag{29}$$

where

$$\tilde{A} = \tilde{A}_{ij} = \begin{bmatrix} \kappa_1 & \kappa_2 & & & & & & & & \\ \kappa_3 & \kappa_4 & \kappa_5 & \kappa_6 & & & & & & \\ -\eta_1 & \eta_2 & \eta_3 & \eta_4 & -\eta_1 & & & & & \\ & -\eta_1 & \eta_2 & \eta_3 & \eta_4 & -\eta_1 & & & & \\ & & -\eta_1 & \eta_2 & \eta_3 & \eta_4 & -\eta_1 & & & \\ & & & \ddots & \ddots & \ddots & \ddots & \ddots & & \\ & & & & & & -\eta_1 & \eta_1 & \omega_1 & \omega_2 \\ & & & & & & & & \omega_3 & \omega_4 \end{bmatrix},$$

$$\tilde{B} = \tilde{B}_{ij} = \begin{bmatrix} \xi_1 & & & & & & & \\ \xi_2 & \xi_3 & \xi_4 & \xi_5 & & & & \\ \eta_1 & \psi_1 & \psi_2 & \psi_1 & \eta_1 & & & \\ & \eta_1 & \psi_1 & \psi_2 & \psi_1 & \eta_1 & & \\ & & \eta_1 & \psi_1 & \psi_2 & \psi_1 & \eta_1 & \\ & & & \ddots & \ddots & \ddots & \ddots & \ddots \\ & & & & & \eta_1 & \psi_1 & \xi_3 & \xi_2 \\ & & & & & & & & \xi_1 \end{bmatrix},$$

and

$$b = b_j = (b_1, b_2, 0, \ldots, 0, b_3, b_4)\,.$$

Here, the entries of A_{ij}, B_{ij}, and b_j are determined according to the type of the equation. The obtained system can be solved by using the Thomas algorithm. To evaluate variables δ_i^{n+1} for a particular time level, the initial vector d^0 is obtained from the initial condition (2).

4 Stability of the proposed method

Stability plays a very important role in the theory of numerical methods. Up to now, various techniques have been proposed to analyze the stability of numerical methods to be confident of the calculated predictions. Among them, von Neumann stability analysis which is based on the Fourier series analysis is the most common technique used to determine stability. Although this technique can be employed directly on linear problems with constant coefficients, it is not valid to estimate the stability of linear problems with variable coefficients and nonlinear problems. However, researchers have developed many ways to apply the von Neumann technique to these challenging problems with some constraints that change according to the specific nature of the problem to be confident in their predictions. While some authors linearize the nonlinear terms of the problem [14, 17, 24, 31–37] or freeze the coefficient [21, 26, 38–40] by applying the von Neumann techniques, some other authors adopted matrix stability [20, 22, 41–45] and other techniques [46, 47] by keeping the originality of their problems with some defects.

Though nonlinear advection–diffusion–reaction equations that are considered in this study are quite common, the von Neumann stability analysis does not directly apply to such nonlinear equations in the standard sense since the stability conditions are complicated and varying. Therefore, in this study, the matrix stability analysis which preserves the nonlinearity of the problem has been performed to analyze the stability of the proposed scheme.

Now, the stability of the present scheme for the numerical integration performed by the Crank–Nicolson method will be discussed using the matrix stability technique by computing the eigenvalues of the coefficient matrix. The proposed scheme for Eq. (1) can be rewritten in compact form as in Eq. (29):

$$\tilde{A}\boldsymbol{d^{n+1}} = \tilde{B}\boldsymbol{d^{n}} + b, \tag{30}$$

where $\tilde{A}$ and $\tilde{B}$ are the real matrices and $\tilde{b}$ is the column vector. Now, let us introduce the numerical error vector ϵ as

$$\epsilon^n = u^n_{exact} - u^n_{app}, \tag{31}$$

where u_{exact} and u_{app} are exact and approximate solutions, respectively. Hence, Eq. (30) is rewritten as [48]

$$\tilde{A}\epsilon^{n+1} = \tilde{B}\epsilon^n. \tag{32}$$

Equation (32) can be written, by assuming that A is nonsingular, as follows:

$$\varepsilon^{n+1} = \tilde{A}^{-1}\tilde{B}\varepsilon^n \implies \varepsilon^{n+1} = C\varepsilon^n, \tag{33}$$

where $C = \tilde{A}^{-1}\tilde{B}$. Here, the stability of the scheme depends on estimate of the coefficient matrix C. For a stable method, matrix C should satisfy the Lax–Richtmyer criterion [49]

$$||C|| \leq 1. \tag{34}$$

In addition, the following inequality

$$\rho(C) \leq ||C|| \tag{35}$$

is always satisfied for any matrix. Here, $\rho(C)$ is the largest eigenvalue of matrix C. Thus, the condition for the stability of system (30) is obtained by considering inequalities (34) and (35) as follows:

$$\rho(C) \leq 1. \tag{36}$$

As can be realized from the previous section and Eq. (29), the entries of matrix C vary depending on some factors such as α, β, μ, γ, spatial and temporal steps sizes, h and dt, and time-dependent parameter δ^n_i. As far as is known, there is no certain

technique to establish a relationship between these factors in the solution of these problems. However, during the calculations, it is observed that the entries of matrix C do not vary rapidly and do not change much with varying these factors. Thus, since the properties of the matrix do not change with different spatial and temporal points, to prove the stability of the proposed scheme, the entries of matrix C can be fixed at certain spatial and temporal points, generally, in maximum values, as a technique commonly used. Thus, for the stability of the proposed scheme given in Eq. (29), $|\rho(C)| \leq 1$ should be fulfilled for all i as $t \to \infty$ [48]. The computed eigenvalues have been presented in Fig. 1 for all examples in Sect. 5. Since Fig. 1 indicates that $max|\rho(C)| \leq 1$ for all examples, the proposed scheme satisfies the criteria of stability.

5 Numerical results and analysis

In this section, the proposed method is applied to the Burgers, Burgers–Fisher, Burgers–Huxley, and Fisher's equations, respectively. The accuracy of the method for these equations is measured by using the absolute error given in the following formula:

$$\text{Absolute error} = |V_{\text{exact}}(x_i, t^n) - V_{\text{app}}(x_i, t^n)|, \tag{37}$$

where V_{exact} and V_{app} are the exact and approximate solutions, respectively. The numerical order of convergence (OC) is obtained by using the following formula:

$$OC = \frac{Log(L_\infty(N)/L_\infty(2N))}{Log(2N/N)}, \tag{38}$$

where $L_\infty = \max_i |V_{\text{exact}}(x_i, t^n) - V_{\text{app}}(x_i, t^n)|$ and N is number of partition of problem domain. The theoretical convergence analysis of nonlinear ADR equations, which are solved by B-spline methods, can be found in the works [31–35] and therein references.

Numerical results obtained by the proposed method have been compared with the exact solution and available literature solutions. The numerical computations have been performed using the software MATLAB 2021. CPU times are measured by the "tic-toc" Matlab function.

Example 1 [28]

Consider the nonlinear Burgers equation by taking $\alpha = \mu = 1$ and $F(u) = 0$ in Eq. (1) over the domain [0, 1] with the initial condition

$$u(x, 0) = \sin(\pi x), \quad 0 \leq x \leq 1, \tag{39}$$

and Dirichlet boundary conditions

$$u(0, t) = u(1, t) = 0, \quad t > 0. \tag{40}$$

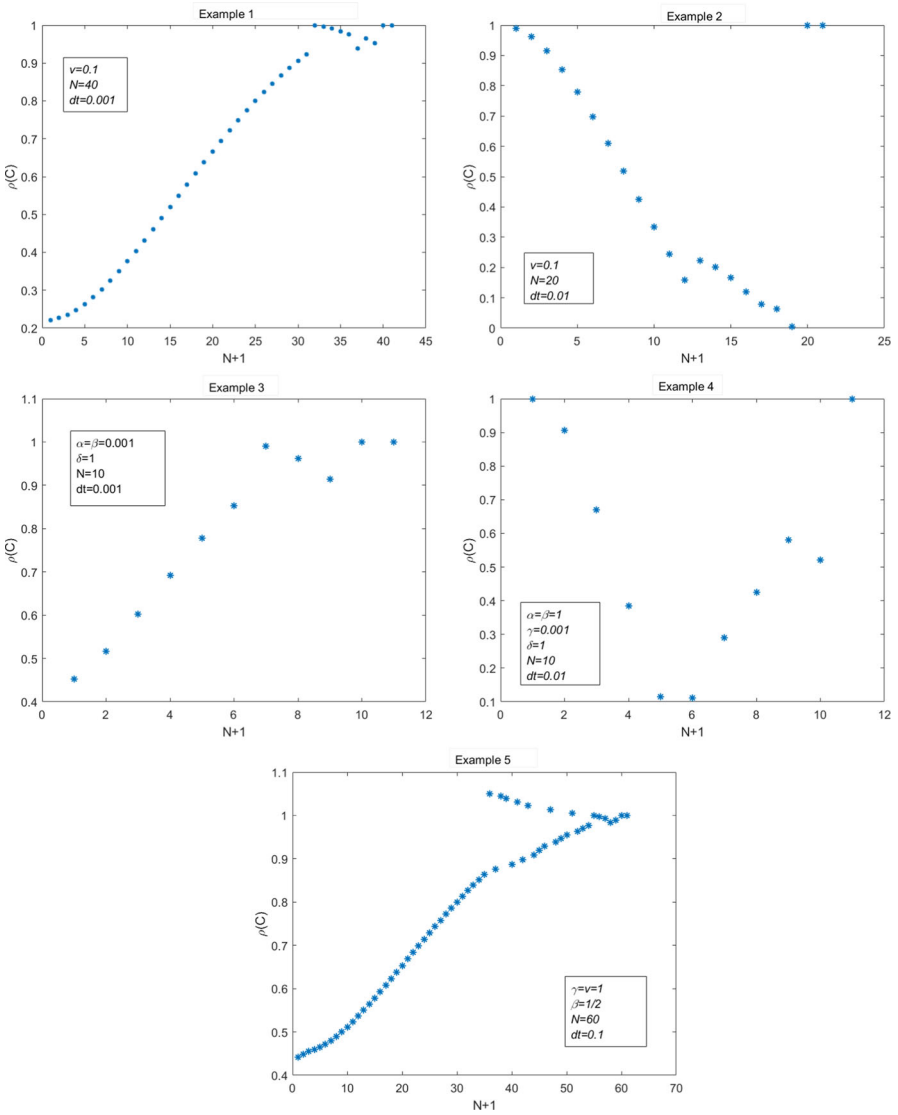

Fig. 1 Eigenvalues of matrix C for Examples 1–5

The exact solution is

$$u(x,t) = 2\pi v \frac{\sum_n^{\infty} a_n \exp(-n^2\pi^2 vt) n \sin(n\pi x)}{a_0 + \sum_{n=1}^{\infty} a_n \exp(-n^2\pi^2 vt)\cos(n\pi x)} \tag{41}$$

with the following Fourier coefficients:

$$a_0 = \int_0^1 \exp\left\{-(2\pi v)^{-1}\left[1-\cos(\pi x)\right]\right\} \mathrm{d}x,$$

Table 1 Comparison of the proposed method solutions with the exact and cubic B-spline solutions [28] for various mesh sizes and $v = 1$ at $t = 0.1$ in Example 1

x	Exact	Dag et al. [28] $N = 160$ $\mathrm{d}t = 0.00001$	Present $N = 40$ $\mathrm{d}t = 0.00001$	Present $N = 80$ $\mathrm{d}t = 0.00001$	Present $N = 40$ $\mathrm{d}t = 0.0001$	Present $N = 80$ $\mathrm{d}t = 0.0001$
0.1	0.10953815	0.10953	0.10953815	0.10953815	0.10953814	0.10953814
0.2	0.20979215	0.20977	0.20979215	0.20979215	0.20979213	0.20979213
0.3	0.29189635	0.29186	0.29189635	0.29189635	0.29189632	0.29189632
0.4	0.34792391	0.34788	0.34792390	0.34792391	0.34792387	0.34792388
0.5	0.37157748	0.37153	0.37157745	0.37157747	0.37157742	0.37157744
0.6	0.35904558	0.35900	0.35904554	0.35904558	0.35904551	0.35904555
0.7	0.30990500	0.30986	0.30990495	0.30990500	0.30990493	0.30990497
0.8	0.22781741	0.22778	0.22781736	0.22781740	0.22781734	0.22781739
0.9	0.12068669	0.12067	0.12068666	0.12068669	0.12068665	0.12068668

$$a_n = 2\int_0^1 \exp\left\{-(2\pi v)^{-1}\left[1-\cos(\pi x)\right]\right\}\cos(n\pi x)\mathrm{d}x, \quad n = 1, 2, 3, \ldots$$

In order to compute the exact solutions given by Eq. (41), the infinite series solutions have been truncated after 500 terms for all calculations. The numerical solutions of Example 1 obtained by the proposed method have been compared with the exact solution, cubic B-spline collocation method [28], and other up-to-date techniques [41, 50–52] in Tables 1–4. Tables 1, 2, and 3 reveal that the proposed scheme converges to the exact solution very fast and is more accurate than the standard cubic B-spline method presented in the work of Dag et al. [28] even when using larger spatial and temporal step sizes, and in this way, it reduces the computational time and complexity. In Table 2, the comparisons of the proposed method solutions with the exact solution and some recently developed techniques [41, 50] are given at different time levels and spatial points for $v = 0.1$. It is observed from the table that the computed results by the current scheme are in reasonable agreement with the solutions in the work of Singh and Gupta [41] for $N = 40$ and $\mathrm{d}t = 0.004$. Besides, when using smaller time increments $\mathrm{d}t = 0.001$ in the proposed scheme, both methods produce the same accurate solutions. In addition, the results of the current scheme are more accurate than those obtained from the solutions presented in the work of Jiwari et al. [50]. Similarly, Table 3 presents the comparisons of the current scheme with the exact solution and the literature solutions [50, 51] at different time levels and spatial points for $v = 0.01$. As seen from the table that the current scheme is far more accurate than the literature [28, 51], and for five decimal places, it is compatible with Jiwari [50].

The exact and numerical behaviors of Example 1 for $v = 0.1$ and $v = 0.01$ are illustrated in Fig. 2. It is seen from the figure, the solutions become smaller as time increases due to the effect of the nonlinear term. The numerical behaviors of the problem exhibited in Fig. 2 are very close to the exact solution and similar to those in the compared studies [28, 41, 50].

Table 2 Comparison of the proposed method solutions with the exact and available literature solutions at different temporal and spatial points for $v = 0.1$ in Example 1

x	t	Exact	Dag et al. [28] $N = 80$ $dt = 0.0001$	Jiwari et al. [50] $dt = 0.001$	Singh and Gupta [41] $N = 40$ $dt = 0.004$	Present $N = 40$ $dt = 0.004$	Present $N = 40$ $dt = 0.001$
0.25	0.4	0.30889423	0.30890	0.30888	0.308894	0.30889260	0.30889410
	0.6	0.24073902	0.24075	0.24073	0.240739	0.24073813	0.24073893
	0.8	0.19567557	0.19569	0.19567	0.195676	0.19567512	0.19567555
	1.0	0.16256486	0.16258	0.16256	0.162565	0.16256470	0.16256488
	3.0	0.02720231	0.02720	0.02719	0.027202	0.02720228	0.02720235
0.5	0.4	0.56963245	0.56965	0.56964	0.569632	0.56963156	0.56963243
	0.6	0.44720552	0.44723	0.44721	0.447206	0.44720508	0.44720565
	0.8	0.35923606	0.35925	0.35924	0.359236	0.35923605	0.35923633
	1.0	0.29191596	0.29192	0.29192	0.291916	0.29191612	0.29191629
	3.0	0.04020492	0.04019	0.04018	0.040205	0.04020485	0.04020497
0.75	0.4	0.62543790	0.62538	0.62554	0.625438	0.62544293	0.62543974
	0.6	0.48721497	0.48715	0.48731	0.487216	0.48721716	0.48721652
	0.8	0.37392175	0.37385	0.37399	0.373923	0.37392276	0.37392275
	1.0	0.28747441	0.28741	0.28751	0.287475	0.28747461	0.28747500
	3.0	0.02977213	0.02976	0.02975	0.029772	0.02977205	0.02977216
CPU-time(s)						0.044165	0.0948552

Table 4 lists comparisons of the solutions obtained by the proposed method and some up-to-date good techniques [50–52] in the literature for smaller viscosity values $v = 0.003$ and $v = 0.004$. Although many numerical methods in the literature fail for these viscosity values, the presented method solutions are in good agreement with the exact solution. Also, the present scheme revealed that one can find similar or sometimes better results than the compared methods even using larger spatial or temporal step sizes. The findings presented in all tables have clearly revealed that less computational time and memory space could be accomplished when using larger spatial or temporal sizes by the proposed method. Also, in addition to the commendable contribution of the aforementioned references in the tables for improving the accuracy of the solutions of Eq. (1), the proposed method provides flexibility in implementation by avoiding some of the additional efforts of the compared methods such as reducing ordinary differential equations, requiring to be modified into a set of basis functions [41] and selecting an optimal parameter [50].

Due to the slow rate of convergence of series solutions, the analytical solutions of the Burgers equations considered in this study which include small viscosity values diverge. Therefore, computationally intensive numerical techniques are required to obtain the solution of such problems. In the work of Jiwari [53], it is indicated that many numerical schemes in the literature fail to capture the physical behavior of the Burgers equation for very small viscosity values. Recently, Jiwari [53], Seydaoglu [51], and Verma et al. [54] presented numerical techniques which produce accurate solutions for small viscosity values. In this work, it has been seen that the proposed

Table 3 Comparison of the proposed method solutions with the exact and available literature solutions at different temporal and spatial points for $\upsilon = 0.01$ in Example 1

x	t	Exact	Dag et al. [28] $N = 80$ $dt = 0.0001$	Seydaoglu [51] $N = 80$ $dt = 0.01$	Jiwari et al. [50] $dt = 0.001$	Present $N = 40$ $dt = 0.001$
0.25	0.4	0.34191493	0.34192	0.34187	0.34190	0.34191493
	0.6	0.26896485	0.26897	0.26894	0.26896	0.26896484
	0.8	0.22148191	0.22148	–	0.22148	0.22148191
	1.0	0.18819396	0.18819	0.18818	0.18819	0.18819396
	3.0	0.07511408	0.07511	0.07511	0.07510	0.07511408
0.5	0.4	0.66071097	0.66071	0.66065	0.66071	0.66071107
	0.6	0.52941826	0.52942	0.52937	0.52942	0.52941830
	0.8	0.43913825	0.43914	–	0.43914	0.43913826
	1.0	0.37442003	0.37442	0.37439	0.37442	0.37442004
	3.0	0.15017900	0.15018	0.15017	0.15014	0.15017899
0.75	0.4	0.91026454	0.91027	0.91032	0.91041	0.91026376
	0.6	0.76724328	0.76725	0.76721	0.76729	0.76724302
	0.8	0.64739523	0.64740	–	0.64741	0.64739529
	1.0	0.55605070	0.55605	0.55601	0.55605	0.55605071
	3.0	0.22481125	0.22483	0.22485	0.22481	0.22480818
CPU-time(s)						0.087824

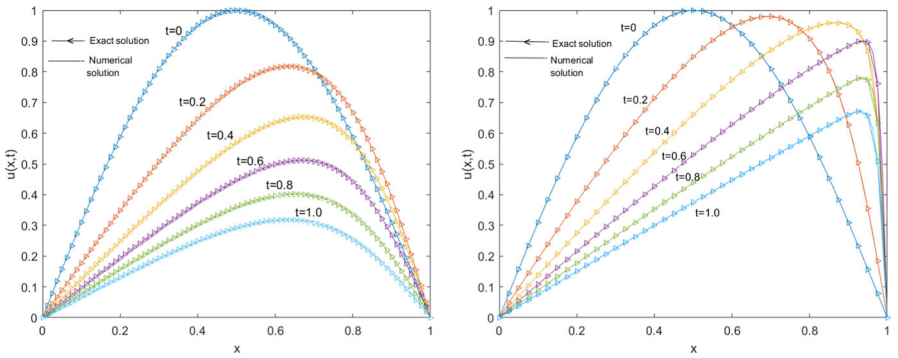

Fig. 2 Simulation of behaviors of the Burgers equation in Example 1 for different time points when $\upsilon = 0.1$ (left) and $\upsilon = 0.01$ (right)

method is successful to capture the behavior of the Burgers equation for $\upsilon = 0.001$ and even a very small value $\upsilon = 0.0005$. Figures 3 and 4 illustrate the behavior of the numerical solutions of Example 1 for small viscosity values $\upsilon = 0.001$ and $\upsilon = 0.0005$ at different time points and these graphs are similar to studies [51, 53, 54]. As clearly seen from the figures, the approximate solutions have a steep descent

Table 4 Comparison of the proposed method solutions with the exact and available literature solutions at different temporal and spatial points for $v = 0.003$ and $v = 0.004$ in Example 1

x	t	$v = 0.003$					$v = 0.004$				
		Exact	Seydaoglu [51] $N = 200$ $dt = 0.01$	Jiwari et al. [50] $dt = 0.001$	Tunc and Sari [52] $N = 100$ $dt = 0.01$	Present $N = 80$ $dt = 0.01$	Exact	Seydaoglu [51] $N = 200$ $dt = 0.01$	Jiwari et al. [50] $dt = 0.001$	Tunc and Sari [52] $N = 100$ $dt = 0.0125$	Present $N = 80$ $dt = 0.01$
0.25	1	0.18901910	0.18902	0.18901	0.18902	0.18901926	0.18890403	0.18890	0.18889	0.18890	0.18890416
	5	0.04698094	0.04698	0.04698	0.04698	0.04698094	0.04697225	0.04697	0.04697	0.04697	0.04697225
	10	0.02422174	0.02422	0.02422	0.02422	0.02422174	0.02421935	0.02422	0.02421	0.02422	0.02421935
	15	0.01631712	0.01632	0.01631	0.01632	0.01631712	0.01631540	0.01632	0.01631	0.01632	0.01631540
0.5	1	0.37622719	0.37623	0.37620	0.37623	0.37622919	0.37597616	0.37597	0.01631	0.37598	0.37597810
	5	0.09395531	0.09396	0.09395	0.09396	0.09395531	0.09393781	0.09394	0.09393	0.09394	0.09393781
	10	0.04844299	0.04844	0.04843	0.04844	0.04844299	0.04843716	0.04844	0.04843	0.04844	0.04843716
	15	0.03263170	0.03263	0.03263	0.03263	0.03263170	0.03259459	0.03259	0.03259	0.03259	0.03259459
0.75	1	0.55928516	0.55928	0.55925	0.55929	0.55928516	0.55882335	0.55883	0.55882	0.55885	0.55884111
	5	0.14091634	0.14092	0.14090	0.14092	0.14091636	0.14088685	0.14089	0.14087	0.14089	0.14088685
	10	0.07260298	0.07260	0.07260	0.07260	0.07260292	0.07220247	0.07221	0.07220	0.07220	0.07220242
	15	0.04838642	0.04839	0.04841	0.04839	0.04838639	0.04677529	0.04678	0.04680	0.04678	0.04677531
CPU-time(s)						0.061186					0.061549

Table 5 Order of convergence when $t = 3$ for different values of N in Example 1

N	$v = 0.1$, d$t = 0.01$		$v = 0.01$, d$t = 0.001$	
	L_∞	OC	L_∞	Ratio
10	1.478459e−05	–	3.490820e−02	–
20	8.788543e−07	4.072326	2.007779e−03	4.119893
40	4.698686e−08	4.225295	9.485849e−05	4.403680
80	5.347528e−09	3.135313	6.150983e−06	3.946888

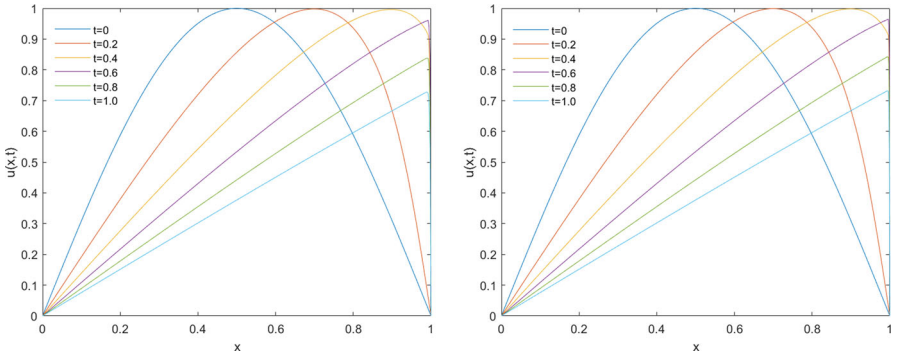

Fig. 3 Simulation of behaviors of the Burgers equation in Example 1 for d$t = 0.01$, $N = 400$ (left) and d$t = 0.01$, $N = 800$ (right) at different time t when $v = 0.001$ and $v = 0.0005$, respectively

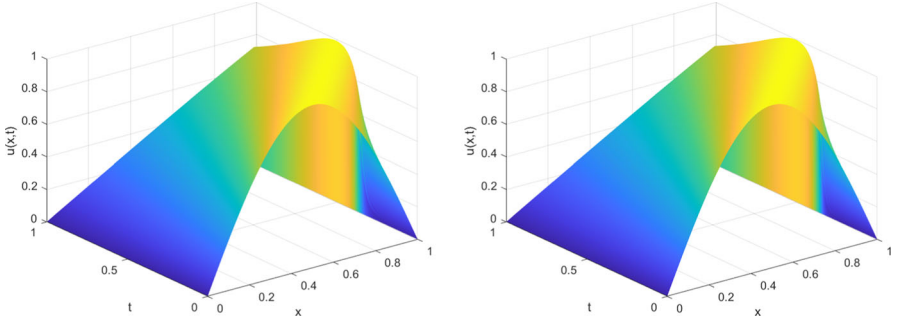

Fig. 4 Simulation of the approximate solution in Example 1 for d$t = 0.01$,$N = 400$ (left) and d$t = 0.01$, $N = 800$ (right) when $v = 0.001$ and $v = 0.0005$, respectively

and as time increases, approximate solutions decrease. The order of convergence is presented in Table 5 for different values of N and it is seen that the convergence rate reaches 4 in $||.||_\infty$.

Example 2 [37]

The next Burgers equation over the domain [0, 1] with the initial condition

$$u(x,0) = 4x(1-x), \quad 0 \le x \le 1 \tag{42}$$

and Dirichlet boundary conditions

$$u(0,t) = u(1,t) = 0, \quad t > 0. \tag{43}$$

The exact solution for this example is given by Eq. (41) with the Fourier coefficients

$$a_0 = \int_0^1 \exp\left\{-x^2(3v)^{-1}(3-2x)\right\} \mathrm{d}x,$$
$$a_n = 2\int_0^1 \exp\left\{-x^2(3v)^{-1}(3-2x)\right\} \cos(n\pi x)\mathrm{d}x.$$

After the numerical experiments have been tried for various mesh sizes, all calculations have been performed for values of $N = 20, 40$ and $M = 100, 1000$. Table 6 exhibits the comparisons of proposed method solutions with the exact and some good techniques in the literature for various mesh sizes and $v = 0.1$ at different spatial and temporal points. As clearly seen from the table, the numerical solutions approximate the exact solution up to seven decimal places when the mesh size is $N = 40$ and $\mathrm{d}t = 0.01$ and are more accurate than the compared methods. Besides, even using fewer temporal and spatial elements in the proposed method, while obtained solutions are the same or sometimes better than those in Erdogan et al. [55], they are more accurate as compared to the literature [37, 50, 56]. Also, the current scheme produces the same accurate solutions as the work of Singh and Gupta [41] for the mesh size values $N = 20$ and $\mathrm{d}t = 0.001$. Thus, the computed results revealed that the proposed method approximates the exact solution very well than the other recently developed techniques even using larger spatial and temporal step sizes. Similarly, a comparison of the proposed method with the exact solution and up-to-date good techniques [37, 41, 50, 55, 56] in the literature is presented in Table 7 for $v = 0.01$. The currently obtained solutions for $N = 40$ and $\mathrm{d}t = 0.01$ are seen to be more accurate than the literature [37, 50, 55, 56] even using smaller spatial or temporal elements compared to the works [37, 50, 56]. Compared to the reference [55], it is seen from the table that the proposed method produces better results even using a far less number of elements. The current scheme is in very good agreement with the exact solution and reference [41] for the mesh size values $N = 40$ and $\mathrm{d}t = 0.001$.

The behaviors of the problem obtained by the proposed method are depicted in Fig. 5 for $v = 0.1$ and $v = 0.01$. As seen in Fig. 5, the numerical solutions and exact solutions are very close to each other. When analyzing figures, it is seen that the peaks of the solution decrease as time increases.

In Table 8, the proposed method has been compared with a hybrid method [53], LRBF-DQM [50], and TGFEM [52] for smaller viscosity values $v = 0.003$ and

Table 6 Comparison of the proposed method solutions with the exact and available literature solutions at different temporal and spatial points for $v = 0.1$ in Example 2

x	t	Exact	Kutluay and Esen [37] $N = 40$ $dt = 0.00001$	Erdogan et al. [55] $N = 40$ $dt = 0.01$	Jiwari et al. [50] $dt = 0.001$	Hussain [56] $N = 40$ $dt = 0.001$	Singh and Gupta [41] $N = 20$ $dt = 0.001$	Present $N = 40$ $dt = 0.001$	Present $N = 20$ $dt = 0.01$	Present $N = 20$ $dt = 0.001$
0.25	0.4	0.31752288	0.31749	0.31750	0.31751	0.31745	0.317520	0.31752259	0.31750748	0.31752017
	0.6	0.24613845	0.24612	0.24612	–	–	–	0.24613828	0.24613005	0.24613657
	0.8	0.19955531	0.19954	0.19955	0.19954	0.19949	0.199554	0.19955522	0.19955096	0.19955433
	1.0	0.16559863	0.16559	0.16559	0.16558	0.16554	0.165599	0.16559862	0.16559700	0.16559866
	3.0	0.02775871	0.02773	–	0.02774	0.02773	0.027759	0.02775875	0.02775880	0.02775928
0.5	0.4	0.58453726	0.58448	0.58453	0.58454	0.58448	0.584535	0.58453705	0.58452957	0.58453470
	0.6	0.45797640	0.45793	0.45796	–	–	–	0.45797639	0.45797243	0.45797685
	0.8	0.36739819	0.36736	0.36739	0.36738	0.36733	0.367402	0.36739838	0.36739902	0.36740153
	1.0	0.29834311	0.29831	0.29834	0.29832	0.29827	0.298348	0.29834338	0.29834613	0.29834778
	3.0	0.04106499	0.04106	–	0.04104	0.04103	0.041066	0.04106503	0.04106492	0.04106579
0.75	0.4	0.64561551	0.64547	0.64566	0.64573	0.64552	0.645644	0.64561748	0.64566435	0.64564400
	0.6	0.50267575	0.50255	0.50270	–	–	–	0.50267735	0.50270745	0.50270043
	0.8	0.38533552	0.38523	0.38534	0.38537	0.38521	0.385352	0.38533653	0.38535150	0.38535187
	1.0	0.29585668	0.29578	0.29586	0.29586	0.29574	0.295867	0.29585726	0.29586350	0.29586668
	3.0	0.03043965	0.03044	–	0.03042	0.03041	0.030440	0.03043967	0.03043940	0.03044018
CPU-time(s)								0.0872392	0.0330608	0.0566462

Table 7 Comparison of the proposed method solutions with the exact and available literature solutions at different temporal and spatial points for $v = 0.01$ in Example 2

x	t	Exact	Kutluay and Esen [37] $N = 40$ $dt = 0.00001$	Erdogan et al. [55] $N = 40$ $dt = 0.01$	Jiwari et al. [50] $dt = 0.001$	Hussain [56] $N = 40$ $dt = 0.001$	Singh and Gupta [41] $N = 20$ $dt = 0.001$	Present $N = 40$ $dt = 0.01$	Present $N = 40$ $dt = 0.001$	Present $N = 20$ $dt = 0.01$
0.25	0.4	0.36225938	0.36218	0.36223	0.36225	0.36222	0.362259	0.36226074	0.36225861	0.36224822
	0.6	0.28203659	0.28197	0.28201	–	0.28199	0.282036	0.28203613	0.28203575	0.28202230
	0.8	0.23045115	0.23040	0.23043	0.23045	0.23040	0.230451	0.23045038	0.23045046	0.23043920
	1.0	0.19469041	0.19465	0.19467	0.19469	0.19463	0.194690	0.19468969	0.19468988	0.19468112
	3.0	0.07613410	0.07613	–	0.07613	0.07610	0.076134	0.07613397	0.07613401	0.07613258
0.5	0.4	0.68367860	0.68364	0.68369	0.68368	0.68365	0.683679	0.68369771	0.68367873	0.68361306
	0.6	0.54831637	0.54829	0.54830	–	0.54828	0.548316	0.54832564	0.54831631	0.54814395
	0.8	0.45371356	0.45368	0.45369	0.45370	0.45368	0.453713	0.45371800	0.45371331	0.45365590
	1.0	0.38567577	0.38564	0.38565	0.38567	0.38564	0.385675	0.38567788	0.38567542	0.38566345
	3.0	0.15217998	0.15217	–	0.15216	0.15215	0.152180	0.15217982	0.15217982	0.15217729
0.75	0.4	0.92050032	0.92047	0.92060	0.92060	0.92048	0.920499	0.92055556	0.92049956	0.92715008
	0.6	0.78299394	0.78297	0.78304	–	0.78296	0.782993	0.78302444	0.78299366	0.79518102
	0.8	0.66272038	0.66270	0.66273	0.66272	0.66269	0.662720	0.66273643	0.66272044	0.66971112
	1.0	0.56931867	0.56930	0.56931	0.56931	0.56929	0.569319	0.56932751	0.56931862	0.57242322
	3.0	0.22774304	0.22773	–	0.22781	0.22771	0.227740	0.22774001	0.22773980	0.22769949
CPU-time(s)								0.0351886	0.0879584	0.0317206

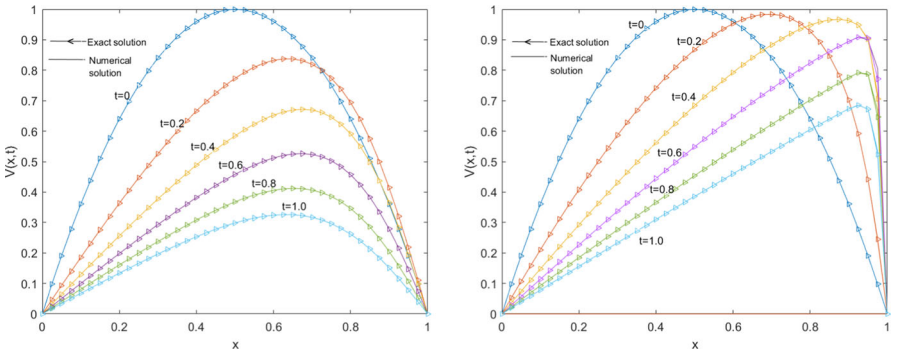

Fig. 5 Simulation of behaviors of the Burgers equation in Example 2 for different time points when $v = 0.1$ (left) and $v = 0.01$ (right)

$v = 0.004$. As indicated before, for these values of viscosity, the numerical methods in the literature fail or do not reach the desired accuracy. As seen from the table, the presented method solutions are in good agreement with the exact solution and produce the same or more accurate solutions than compared with other solutions in the table even using larger spatial step sizes. Figures 6 and 7 show the physical behavior of numerical solutions obtained by the proposed method for very small viscosity values $v = 0.001$ and $v = 0.0005$ at different time levels. It is seen from the figures that the proposed scheme produces accurate and reliable results for small viscosity values.

Example 3 [11]

Consider the Burgers–Fisher equation by taking $v = 1$ and $F(u) = \beta u(1 - u^{\mu})$ in Eq. (1) subject to initial condition

$$u(x,0) = \left(\frac{1}{2} + \frac{1}{2}tanh\left(\left(\frac{-\alpha\mu}{2(1+\mu)}\right)x\right)\right)^{\frac{1}{\mu}}, \quad 0 \le x \le 1, \tag{44}$$

and boundary conditions

$$u(0,t) = \left(\frac{1}{2} - \frac{1}{2}\tanh\left(\left(\frac{-\alpha\mu}{2(1+\mu)}\right)\left(\frac{\alpha}{1+\mu} + \frac{\beta(1+\mu)}{\alpha}\right)t\right)\right)^{\frac{1}{\mu}}, \tag{45}$$

$$u(1,t) = \left(\frac{1}{2} + \frac{1}{2}\tanh\left(\left(\frac{-\alpha\mu}{2\,(1+\mu)}\right)\left(1 - \left(\frac{\alpha}{1+\mu} + \frac{\beta(1+\mu)}{\alpha}\right)t\right)\right)\right)^{\frac{1}{\mu}}. \tag{46}$$

The exact solution is given

$$u(x,t) = \left(\frac{1}{2} + \frac{1}{2}\tanh\left(\frac{-\alpha\mu}{2\,(\mu+1)}\left(x - \left(\frac{\alpha}{\mu+1} + \frac{\beta(\mu+1)}{\alpha}\right)t\right)\right)\right)^{\frac{1}{\mu}}, \quad t \ge 0. \tag{47}$$

Table 8 Comparison of the proposed method solutions with the exact and available literature solutions at different temporal and spatial points for $v = 0.003$ and $v = 0.004$ in Example 2

		$v = 0.003$					$v = 0.004$				
x	t	Exact	Jiwari [53] $dt = 0.001$	Jiwari et al. [50] $dt = 0.001$	Tunc and Sari [52] $N = 100$ $dt = 0.01$	Present $N = 40$ $dt = 0.01$	Exact	Jiwari [53] $dt = 0.001$	Jiwari et al. [50] $dt = 0.001$	Tunc and Sari [52] $N = 100$ $dt = 0.01$	Present $N = 40$ $dt = 0.01$
0.25	1	0.19672202	0.19673	0.19670	0.19672	0.19672094	0.19639300	0.19640	0.19638	0.19639	0.19639203
	5	0.04746474	0.04747	0.04746	0.04746	0.04746454	0.04743858	0.04744	0.04744	0.04744	0.04743844
	10	0.02434970	0.02434	0.02434	0.02435	0.02434965	0.02434263	0.02434	0.02434	0.02434	0.02434260
	15	0.01637507	0.01637	0.01637	0.01638	0.01637504	0.01637125	0.01637	0.01637	0.01637	0.01637123
0.5	1	0.38896706	0.38898	0.38893	0.38897	0.38896472	0.38849076	0.38850	0.38845	0.38849	0.38849376
	5	0.09491170	0.09491	0.09491	0.09491	0.09491137	0.09486089	0.09487	0.09486	0.09486	0.09486066
	10	0.04869814	0.04869	0.04868	0.04870	0.04869804	0.04868313	0.04868	0.04867	0.04868	0.04868305
	15	0.03274752	0.03275	0.03274	0.03275	0.03274746	0.03270700	0.03271	0.03270	0.03271	0.03270692
0.75	1	0.57378	0.57383	0.57378	0.57383	0.57154537	0.57321873	0.57320	0.57315	0.57321	0.57281015
	5	0.14232395	0.14233	0.14232	0.14232	0.14232358	0.14224850	0.14225	0.14225	0.14225	0.14224786
	10	0.07298598	0.07299	0.07297	0.07299	0.07298493	0.07258104	0.07258	0.07258	0.07258	0.07258009
	15	0.04856836	0.04857	0.04858	0.04857	0.04856789	0.04696437	0.04697	0.04698	0.04696	0.04696455
CPU-time(s)						0.0301478					0.0298588

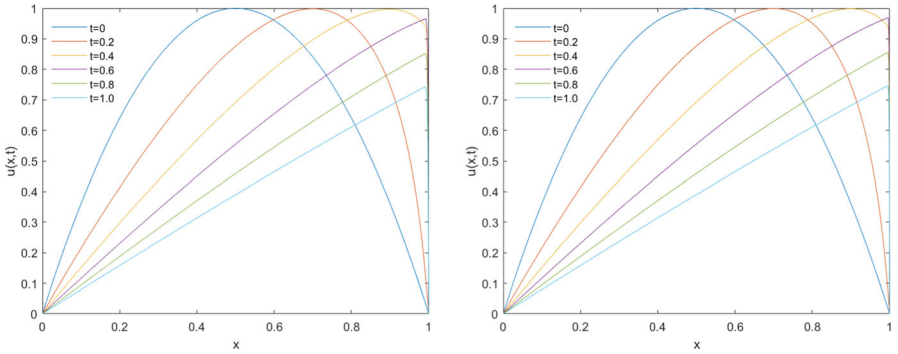

Fig. 6 Simulation of behaviors of the Burgers equation in Example 2 for $dt = 0.01$, $N = 400$ (left) and $dt = 0.01$, $N = 800$ (right) at different time t when $v = 0.001$ and $v = 0.0005$, respectively

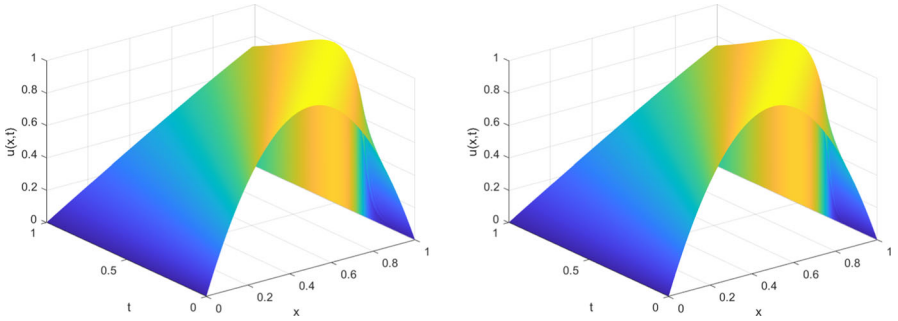

Fig. 7 Simulation of the approximate solution in Example 2 for $dt = 0.01$,$N = 400$ (left) and $dt = 0.01$, $N = 800$ (right) when $v = 0.001$ and $v = 0.0005$, respectively

In this example, in order to test the accuracy and efficiency of the proposed method, a comparison of the absolute errors of the proposed scheme and up-to-date techniques in the literature is given in Tables 9, 10, and 11.

The absolute errors of the proposed method and contemporary methods [11, 47, 57, 58] in the literature are presented for $\alpha = \beta = 1$ and $\mu = 2, 4$ when $h = 0.1$ and $dt = 0.0001$ at different time levels with step sizes in Table 9. The superiority of the proposed scheme is shown by comparing it with the literature solutions [47, 57] in Table 9. Besides, as clearly seen in the table, although the solutions obtained by the proposed method are less accurate than the compared methods [11, 58] at the starting points, it should be noted that the solutions are highly accurate at the middle points and endpoints with the effect of the fourth-order formula and the error decreases with a decrease in time. The evaluation of computed solutions with spatial and temporal variables for the values in Table 9 is illustrated in Fig. 8.

The pointwise absolute errors of the proposed method and up-to-date developed methods in the literature are listed for $\alpha = \beta = 0.001$, $\mu = 2, 8$, and $h = 0.1$ in Table 10. It can be noticed from the table that the current method provides more accurate solutions than those obtained in the literature [57, 58] even using larger temporal step sizes. Besides, as compared to the other references [11, 26], although the

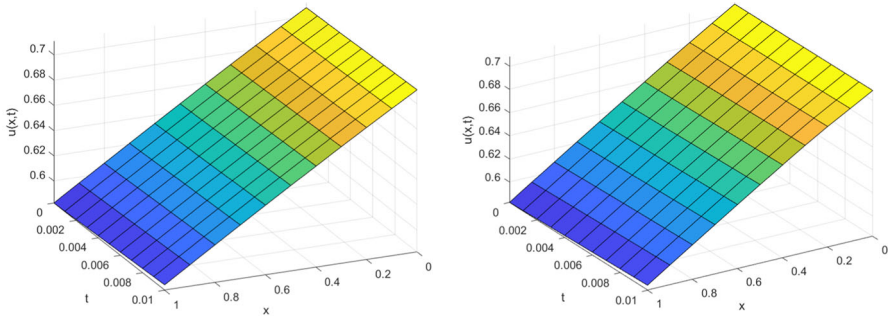

Fig. 8 Simulations of the exact (left) and proposed method (right) solutions for the generalized Burgers–Fisher equation with $\alpha = \beta = 1$, $\mu = 2$, $h = 0.1$, and $\mathrm{d}t = 0.001$

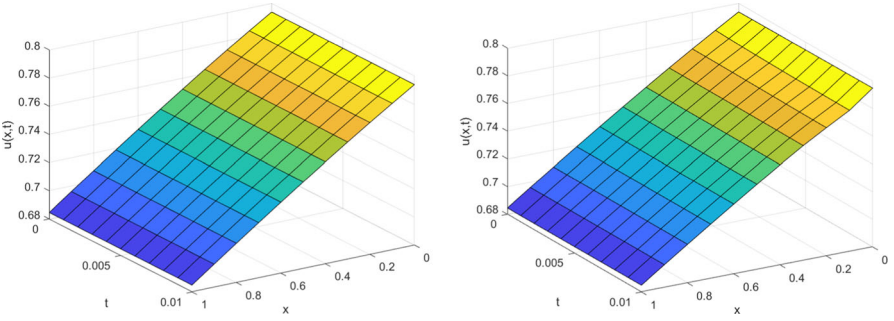

Fig. 9 Simulations of the exact (left) and proposed scheme (right) for the generalized Burgers–Fisher equation with $\alpha = 1$, $\beta = 0$, $\mu = 3$, $h = 0.1$, and $\mathrm{d}t = 0.001$

proposed method produces less accurate solutions at the starting points of the domain, it is more accurate in the middle points and endpoints. In other words, solutions are less accurate than the other compared methods at the starting points, but they are highly accurate at the middle points and endpoints. It can be concluded from these results observed in Tables 9 and 10, using the fourth-order compact formula in interior points has increased the accuracy of the solution of these points. Contrary to this, due to the domination of the relatively low-order cubic B-spline in the starting points, in these points, it is obtained a less accurate solution by comparison with the interior points. Also, as the time increases, the error decreases as seen in the table.

In Table 11, the absolute errors of the proposed and up-to-date literature [11, 47] solutions are listed for $\alpha = 1$, $\beta = 0$, $\mu = 3, 8$, and $h = 0.1$. It is noticed from the table that the proposed method produces the same or sometimes more accurate solutions, especially at the middle and endpoints of the domain even using larger temporal step sizes. Figure 9 shows the behavior of Example 3 for values in Table 11.

Table 9 Absolute error comparison of the proposed method and available literature solution at different temporal and space points for $\alpha = \beta = 1$ and $\mu = 2, 8$ in Example 3

		$\delta = 2$					$\delta = 8$			
x	t	İsmail et al. [57] $dt = 0.0001$	Sari et al. [58] $dt = 0.0001$	Hammad and El-Azab [11] $dt = 0.0001$	Verma and Kayenat [47] $dt = 0.0001$	Present $dt = 0.0001$	Sari et al. [58] $dt = 0.0001$	Hammad and El-Azab [11] $dt = 0.0001$	Verma and Kayenat [47] $dt = 0.0001$	Present $dt = 0.0001$
0.1	0.0001	2.81E−04	1.55E−05	4.48E−10	3.99E−05	2.23E−05	2.07E−05	8.29E−09	5.16E−05	3.01E−04
	0.0005	1.40E−03	7.62E−05	2.47E−05	3.98E−05	1.09E−04	1.02E−04	1.07E−05	5.16E−05	1.41E−03
	0.001	2.80E−03	1.50E−04	5.39E−05	3.97E−05	2.13E−04	2.00E−04	2.34E−05	5.14E−05	2.61E−03
0.5	0.0001	2.69E−04	1.83E−05	7.13E−11	4.18E−05	1.74E−08	2.75E−05	6.24E−09	6.08E−05	5.82E−08
	0.0005	1.34E−03	9.14E−05	2.09E−05	4.15E−05	8.50E−08	1.37E−04	9.38E−06	6.09E−05	6.72E−07
	0.001	2.69E−03	1.83E−04	4.71E−05	4.11E−05	1.41E−07	2.74E−04	2.11E−05	6.09E−05	1.52E−06
0.9	0.0001	2.55E−04	2.07E−05	5.53E−10	4.27E−05	1.16E−10	3.43E−05	4.91E−09	6.90E−05	2.34E−10
	0.0005	1.27E−03	1.02E−04	1.93E−05	4.22E−05	4.27E−10	1.69E−04	8.71E−06	6.92E−05	2.02E−09
	0.001	2.55E−03	2.00E−04	4.22E−11	4.16E−05	5.25E−10	3.31E−04	1.91E−05	6.94E−05	2.45 E−09
CPU-time(s)						0.0408154				0.0403208

Table 10 Absolute error comparison of the proposed method and available literature solution at different temporal and space points for $\alpha = \beta = 0.001$ and $\mu = 1, 4$ in Example 3

x	t	$\mu = 1$						$\mu = 4$				
		İsmail et al. [57] $dt = 0.0001$	Sari et al. [58] $dt = 0.0001$	Hammad and El-Azab [11] $dt = 0.0001$	Singh et al. [26] $dt = 0.0001$	Present $dt = 0.0001$	Present $dt = 0.001$	Sari et al. [58] $dt = 0.0001$	Hammad and El-Azab [11] $dt = 0.0001$	Verma and Kayenat [47] $dt = 0.0001$	Present $dt = 0.0001$	Present $dt = 0.001$
0.1	0.001	1.94E−06	1.01E−07	5.81E−11	4.64E−11	6.95E−09	7.36E−09	1.75E−08	3.91E−11	4.20E−08	1.77E−08	1.88E−08
	0.005	9.69E−06	4.38E−07	2.61E−10	5.85E−10	9.02e−08	8.97E−08	7.37E−07	1.75E−10	4.20E−08	2.46E−07	2.45E−07
	0.01	1.94E−05	7.53E−07	4.46E−10	8.84E−10	3.48E−07	3.48E−07	1.27E−06	3.00E−10	4.20E−08	9.41E−07	9.40E−07
0.5	0.001	1.94E−06	1.04E−07	5.62E−11	5.39E−14	8.71E−12	5.45E−11	1.75E−08	3.78E−11	4.21E−08	2.40E−11	5.46E−11
	0.005	9.69E−06	5.21E−07	3.06E−10	2.73E−13	6.78E−12	5.48E−12	8.77E−07	2.06E−10	4.21E−08	1.75E−11	1.39E−11
	0.01	1.95E−05	1.04E−06	6.19E−10	2.04E−12	1.05E−10	1.17E−10	1.75E−06	4.16E−10	4.21E−08	2.93E−10	3.25E−10
0.9	0.001	1.94E−06	1.01E−07	5.81E−11	4.64E−11	2.76E−14	3.99E−14	1.75E−08	3.91E−11	4.21E−08	7.33E−14	1.10E−13
	0.005	9.69E−06	4.38E−07	2.61E−10	5.85E−10	4.61E−15	2.59E−14	7.38E−07	1.75E−10	4.21E−08	1.41E−14	7.21E−14
	0.01	1.94E−05	7.53E−07	4.46E−10	8.84E−10	9.60E−15	1.10E−14	1.27E−06	3.00E−10	4.21E−08	2.61E−14	3.10E−14
CPU-time(s)						0.0185514	0.0076276				0.019311	0.0074688

Table 11 Absolute error comparison of the proposed method and available literature solution at different temporal and space points for $\alpha = 1$, $\beta = 0$, and $\mu = 3, 8$ in Example 3

x	t	$\mu = 3$				$\mu = 8$			
		Hammad and El-Azab [11] $dt = 0.0001$	Verma and Kayenat [47] $dt = 0.000001$	Present $dt = 0.00001$	Present $dt = 0.0001$	Hammad and El-Azab [11] $dt = 0.0001$	Verma and Kayenat [47] $dt = 0.000001$	Present $dt = 0.00001$	Present $dt = 0.0001$
0.1	0.0001	5.79E−11	7.94E−08	5.62E−05	5.62E−05	2.77E−11	1.17E−07	2.62E−04	2.63E−04
	0.0005	2.09E−05	2.90E−07	2.65E−04	2.65E−04	1.08E−05	5.53E−07	1.23E−03	1.23E−03
	0.001	4.55E−05	5.39E−07	4.93E−04	4.93E−04	2.35E−05	1.07E−06	2.26E−03	2.27E−03
0.5	0.0001	5.17E−11	2.85E−08	9.62E−11	3.31E−10	2.04E−11	8.93E−08	2.73E−08	2.62E−08
	0.0005	1.79E−05	3.27E−08	9.70E−08	9.64E−08	9.41E−06	4.23E−07	5.22E−07	5.20E−07
	0.001	4.03E−05	3.82E−08	2.66E−07	2.66E−07	2.12E−05	8.40E−07	1.28E−06	1.28E−06
0.9	0.0001	5.31E−11	2.14E−08	4.37E−10	4.39E−10	1.81E−04	5.72E−08	3.79E−11	4.71E−11
	0.0005	1.65E−05	2.18E−07	3.95E−11	3.67E−11	8.73E−06	2.54E−07	1.73E−09	1.72E−09
	0.001	3.62E−05	4.49E−07	2.65E−10	2.66E−10	1.91E−05	4.91E−07	2.33E−09	2.34E−09
CPU-time(s)				0.0323082	0.0197684			0.0334982	0.0190348

Example 4 [11]

Now, let us consider the generalized Burgers–Huxley equation by taking $v = 1$ and $F(u) = \beta u\,(1 - u^{\mu})\,(u^{\mu} - \gamma)$ in Eq. (1) subject to the initial condition

$$u(x,0) = \left(\frac{\gamma}{2} + \frac{\gamma}{2}\tanh\left(\frac{-\alpha\mu + \mu\sqrt{\alpha^2 + 4\beta(1+\mu)}}{4(1+\mu)}\gamma x\right)\right)^{\frac{1}{\mu}}, \quad 0 \le x \le 1, \tag{48}$$

and boundary conditions

$$u(0,t) = \left(\frac{\gamma}{2} - \frac{\gamma}{2}\tanh\left(\frac{-\alpha\mu + \delta\sqrt{\alpha^2 + 4\beta(1+\mu)}}{4(1+\mu)}\left(\frac{\alpha\gamma}{1+\mu} - \frac{(1+\mu-\gamma)\left(-\alpha + \sqrt{\alpha^2 + 4\beta(1+\mu)}\right)}{2(1+\mu)}\right)t\right)\right)^{\frac{1}{\mu}}, \tag{49}$$

$$u(1,t) = \left(\frac{\gamma}{2} + \frac{\gamma}{2}\tanh\left(\frac{-\alpha\mu + \mu\sqrt{\alpha^2 + 4\beta(1+\mu)}}{4(1+\mu)}\left(1 - \left(\frac{\alpha\gamma}{1+\mu} - \frac{(1+\mu-\gamma)\left(-\alpha + \sqrt{\alpha^2 + 4\beta(1+\mu)}\right)}{2(1+\mu)}\right)t\right)\right)\right)^{\frac{1}{\mu}}. \tag{50}$$

The exact solution is

$$u(x,t) = \left(\frac{\gamma}{2} + \frac{\gamma}{2}\tanh\left(\frac{-\alpha\mu + \mu\sqrt{\alpha^2 + 4\beta(1+\mu)}}{4(1+\mu)}\left(x - \left(\frac{\alpha\gamma}{1+\mu} - \frac{(1+\mu-\gamma)\left(-\alpha + \sqrt{\alpha^2 + 4\beta(1+\mu)}\right)}{2(1+\mu)}\right)t\right)\right)\right)^{\frac{1}{\mu}}. \tag{51}$$

Tables 12, 13, and 14 illustrate the comparison of the absolute errors of the proposed method and up-to-date techniques in the literature solutions for different values of α, β, γ, and μ at different temporal and spatial points.

For comparison purposes, the absolute errors of the proposed method and up-to-date good techniques [11, 14, 25, 59] have been presented for $\alpha = \beta = 1$, $\gamma = 0.001$, and $\mu = 1, 2$ at different time levels and spatial points in Table 12. The obtained results are more accurate than Inan and Bahadir [59] and are in reasonable agreement with other compared methods [11, 14, 25] for both values $\mu = 1$ and $\mu = 2$ even with

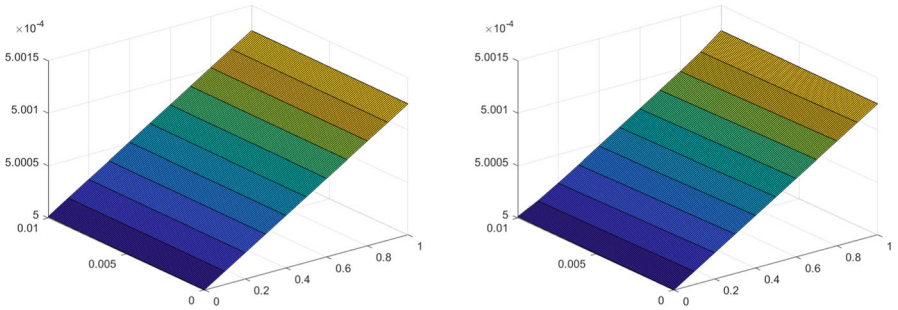

Fig. 10 Simulations of the exact (left) and proposed method (right) solutions for the generalized Burgers–Huxley equation with $\alpha = \beta = 1$, $\gamma = 0.001$, $\mu = 1$, $h = 0.1$ and $dt = 0.0001$ at $t = 0.01$

using a larger temporal mesh size. Also, it can be noticed from the table that the order of errors is the same for all compared methods. Furthermore, it can be observed that the larger temporal size does not affect the accuracy of the current method and it still approximates as adequately as literature solutions. Figure 10 illustrates the behavior of the exact and numerical solution results, and as clearly seen in the figure, there is a close agreement between the two solutions.

Table 13 represents the absolute errors of the proposed method and available literature solutions at different time levels with spatial steps for $\alpha = 0$, $\beta = 1$, and $\gamma = 0.001$. It is concluded from the table that the current scheme is very well compatible with the literature solutions. In addition, as seen from the table, when it is used larger temporal size the accuracy of the suggested scheme is not disturbed so much.

Finally, the absolute errors of the proposed method and hybrid B-spline collocation method [14] are listed at different space points and time levels for $\alpha = 5$, $\beta = 10$, and various γ values in Table 14. The comparisons demonstrated that the proposed method produces the same accurate solution with the reference [14] by using larger temporal mesh sizes. Therefore, it is concluded that the proposed method is more efficient and economical than the compared method. Although, in addition to this, the work of Wasim [14] captures the behavior of the problem accurately, a lack of a technique for predicting the optimal parameter used in the method can be a defect in the flexibility of their method.

Example 5 [29]

Let us consider the Fisher's equation by taking $\alpha = 0$ and $F(u) = \beta u(1 - \gamma u)$ with the following initial condition

$$u(x, 0) = -\frac{1}{4}\frac{\beta}{\gamma}\left[\operatorname{sech}^2\left(-\sqrt{\frac{\beta}{24v}}x\right) - 2\tanh\left(-\sqrt{\frac{\beta}{24v}}x\right) - 2\right], \tag{52}$$

and boundary conditions

$$\lim_{x\to-\infty} u(x, t) = 0.5 \quad \text{and} \quad \lim_{x\to\infty} u(x, t) = 0. \tag{53}$$

Table 12 Absolute error comparisons of the proposed method and available literature solution at different temporal and spatial points for $\alpha = \beta = 1$ and $\gamma = 0.001$ in Example 4

x	t	$\mu = 1$						$\mu = 2$				
		Inan and Bahadır [59] $dt = 0.0001$	Hammad and El-Azab [11] $dt = 0.001$	Wasim et al. [14] $dt = 0.0001$	Alinia and Zarebnia [25] $dt = 0.0001$	Present $dt = 0.01$	Present $dt = 0.0001$	Inan and Bahadır [59] $dt = 0.0001$	Wasim et al. [14] $dt = 0.0001$	Alinia and Zarebnia [25] $dt = 0.0001$	Present $dt = 0.01$	Present $dt = 0.001$
0.1	0.05	1.5448E−08	7.7006E−09	7.587E−09	–	7.73834E−09	7.72791E−09	1.4030E−06	3.545E−07	–	3.61278E−07	3.60795E−07
	0.1	2.2587E−08	1.1268E−08	1.109E−08	6.1060E−09	1.13017E−08	1.12972E−08	2.0510E−06	5.182E−07	2.85100E−07	5.27632E−07	5.27424E−07
	1	3.3727E−08	1.6863E−08	1.656E−08	6.5588E−09	1.68647E−08	1.68646E−08	3.0562E−06	7.734E−07	3.06164E−07	7.87167E−07	7.87167E−07
0.5	0.05	3.4691E−08	1.7284E−08	1.704E−08	–	1.73607E−08	1.73545E−08	3.1502E−06	7.960E−07	–	8.10475E−07	8.10186E−07
	0.1	5.7640E−08	2.8738E−08	2.830E−08	1.6754E−08	2.88456E−08	2.88314E−08	5.2339E−06	1.322E−06	7.82228E−07	1.34664E−06	1.34598E−06
	1	9.3698E−08	4.6841E−08	4.599E−08	1.8219E−08	4.68490E−08	4.68490E−08	8.4901E−06	2.148E−06	8.50425E−07	2.18666E−06	2.18666E−06
0.9	0.05	1.5450E−08	7.7006E−09	7.587E−09	–	7.73885E−09	7.72842E−09	1.4029E−06	3.544E−07	–	3.61259E−07	3.60777E−07
	0.1	2.2590E−08	1.1268E−08	1.109E−08	6.1058E−09	1.12983E−08	1.12983E−08	2.0511E−06	5.182E−07	2.85074E−07	5.27637E−07	5.27429E−07
	1	3.3734E−08	1.6863E−08	1.656E−08	6.5586E−09	1.68668E−08	1.68068E−08	3.0564E−06	7.734E−07	3.06137E−07	7.87222E−07	7.87221E−07
CPU-time(s)						0.0382628	0.2305236				0.0374110	0.0559352

Table 13 Absolute error comparisons of the proposed method and available literature solution at different temporal and spatial points for $\alpha = 0$, $\beta = 1$, and $\gamma = 0.001$ in Example 4

x	t	$\mu = 1$				$\mu = 2$			$\mu = 3$			
		Hammad and El-Azab [11] $dt = 0.001$	Wasim et al. [14] $dt = 0.001$	Present $dt = 0.01$	Present $dt = 0.001$	Hammad El-Azab [11] $dt = 0.01$	Present $dt = 0.01$	Present $dt = 0.001$	Hammad El-Azab [11] $dt = 0.001$	Wasim et al. [14] $dt = 0.001$	Present $dt = 0.01$	Present $dt = 0.001$
0.1	0.05	1.0269E−08	1.030E−08	1.03043E−08	1.03043E−08	4.5929E−07	4.61484E−07	4.60862E−07	1.6303E−06	1.635E−06	1.63810E−06	1.63589E−06
	0.1	1.5027E−08	1.506E−08	1.50697E−08	1.50637E−08	6.7207E−07	6.73981E−07	6.73712E−07	2.3856E−06	2.391E−06	2.39233E−06	2.39138E−06
	1	2.2488E−08	2.248E−08	2.24877E−08	2.24877E−08	1.0053E−06	1.00533E−06	1.00533E−06	3.5670E−06	3.567E−06	3.56702E−06	3.56702E−06
0.5	0.05	2.3049E−08	2.313E−08	2.31477E−08	2.31393E−08	1.0308E−06	1.03522E−06	1.03484E−06	3.6588E−06	3.673E−06	3.67446E−06	3.67313E−06
	0.1	3.8324E−08	3.844E−08	3.84609E−08	3.84419E−08	1.7139E−06	1.72005E−06	1.71920E−06	6.0835E−06	6.102E−06	6.10520E−06	6.10219E−06
	1	6.2465E−08	6.246E−08	6.24654E−08	6.24654E−08	2.7925E−06	2.79249E−06	2.79249E−06	9.9079E−06	9.907E−06	9.90791E−06	9.90790E−06
0.9	0.05	1.0269E−08	1.030E−08	1.03181E−08	1.03042E−08	4.5922E−07	4.61415E−07	4.60793E−07	1.6299E−06	1.635E−06	1.63377E−06	1.63547E−06
	0.1	1.5027E−08	1.506E−08	1.58164E−08	1.50637E−08	6.7200E−07	6.73905E−07	6.73636E−07	2.3851E−06	2.390E−06	2.39186E−06	2.39091E−06
	1	2.2488E−08	2.248E−08	2.24877E−08	2.24877E−08	1.0053E−06	1.00525E−06	1.00525E−06	3.5665E−06	3.566E−06	3.56654E−06	3.56654E−06
CPU-time(s)				0.0484688	0.07074		0.0488202	0.0701566			0.048763	0.0780096

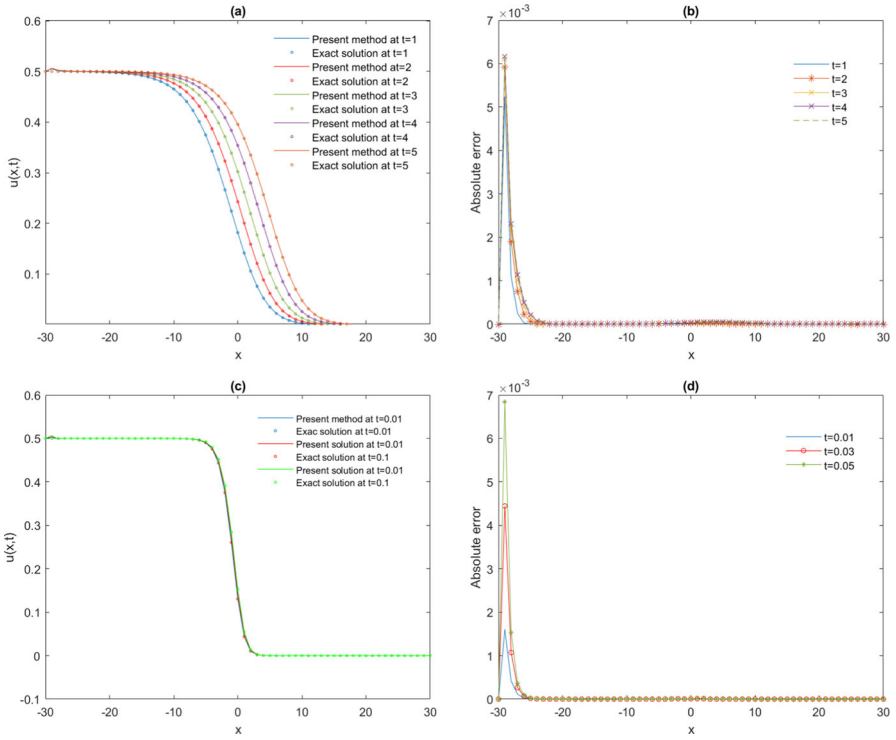

Fig. 11 Numerical behavior of the proposed method solutions and exact solutions for the Fisher's equation

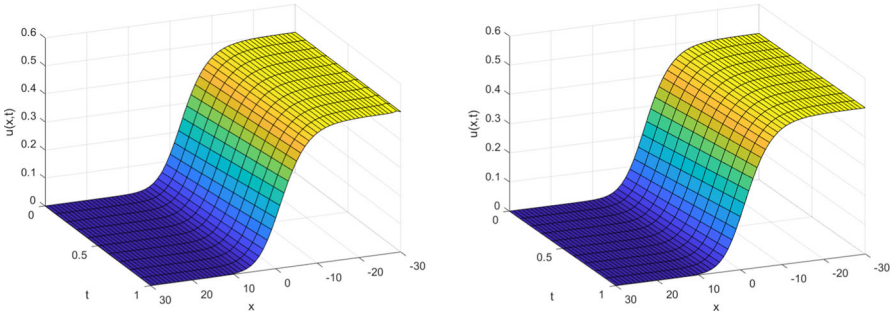

Fig. 12 Simulations of the proposed method (left) and exact (right) solutions for the Fisher's equation with the viscosity value $v = 1$

The exact solution is

$$u(x,t) = -\frac{1}{4}\frac{\beta}{\gamma}\left[\operatorname{sech}^2\left(\pm\sqrt{\frac{\beta}{24v}}x + \frac{5\beta}{12}t\right) - 2\tanh\left(-\sqrt{\frac{\beta}{24v}}x + \frac{5\beta}{12}t\right) - 2\right]. \tag{54}$$

The behavior of the Fisher's equation depends on infection rate $\beta u(1-\gamma u)$. The infection rate is measured by βu, while the nonlinear term $-\beta\gamma u^2$ represents the diffusion rate. Wazwaz [60] proposed an analytical solution for this problem and the solution indicates a shock-like traveling wave. The amplitude of the wave is proportional to $\frac{\beta}{\gamma}$. The numerical solutions of the equation have been presented in Tables 15 and 16 for the parameters $\gamma = v = 1$, $\beta = 1/2$ in $[-30, 30]$, when $t = 2$ and $t = 4$ with $h = 1$ and $dt = 0.1$. It is seen from the tables that while the proposed method produces more accurate solutions than the wavelet-Galerkin approach [30] and the standard cubic B-spline method [29], it is in reasonable agreement with other compared literature [18, 61, 62].

Figure 11 illustrates the behaviors of the exact and present method solutions and pointwise error graphs for $v = 1$ at times $t = 1, 2, 3, 4, 5$ (Fig. 11a, b) and for $v = 0.1$ at times $t = 0.1, 0.2, 0.3, 0.4, 0.5$ (Fig. 11c, d). As indicated in the figure, the currently obtained solutions and exact solution profiles are very close. In addition, Figs. 12 and 13 show the surface behaviors of the exact and numerical solutions for $v = 1$ and $v = 0.1$, respectively and it can be observed that numerical and exact behaviors look similar.

6 Conclusion

In this study, a new hybrid method based on the combination of the cubic B-spline method and fourth-order compact finite difference scheme has been introduced for the spatial discretization of the ADR equations. Since the desired accuracy cannot be reached using the cubic B-spline method, the second-order derivatives have been obtained by the fourth-order compact finite difference scheme in which second derivatives are approximated with the unknowns and their first derivatives. Thus, the second-order derivatives are represented by the fourth-order convergence in the proposed method. The combined method has been applied to the Burgers, Burgers–Fisher, Burgers–Huxley, and Fisher's equations after the equations have been discretized with help of the Crank–Nicolson scheme. The numerical results are in very good agreement with the exact solutions and quite satisfactory and competent with contemporary literature solutions. Furthermore, the solutions of some equations have superior performance even when relatively large temporal and spatial step sizes are used in computations and, thus, this reduces the computational complexity, computational time, and memory space. In addition, the proposed method can capture the physical behavior of the Burgers equation in the presence of very small viscosity values accurately and efficiently. The stability of the proposed method has been discussed by the matrix stability analysis and it is concluded that the method is stable. The convergence of the solution has been measured by an error norm and it is confirmed that the present method is convergent. Furthermore, the proposed method is easy to implement and provides much flexibility with the minimal computational effort to deal with challenging problems by altering the second-order approach with the high-order compact finite difference. Therefore, it is concluded that the new hybrid method that has computationally efficient and economical properties is a good alternative for investigating the nonlinear partial differential equations.

Table 14 Absolute error comparisons of the proposed method and available literature solution at different temporal and spatial points for $\alpha = 5$, $\beta = 10$ and $\mu = 2$ in Example 4

x	t	$\gamma = 0.001$			$\gamma = 0.0001$			$\gamma = 0.00001$		
		Wasim et al. [14] $dt = 0.0001$	Present $dt = 0.01$	Present $dt = 0.001$	Wasim et al. [14] $dt = 0.0001$	Present $dt = 0.01$	Present $dt = 0.001$	Wasim et al. [14] $dt = 0.0001$	Present $dt = 0.01$	Present $dt = 0.001$
0.1	0.2	6.230E−06	6.2314E−06	6.2300E−06	1.971E−07	1.9720E−07	1.9716E−07	6.235E−09	6.2365E−09	6.2351E−09
	0.5	7.056E−06	7.0557E−06	7.0556E−06	2.233E−07	2.2338E−07	2.2338E−07	7.065E−09	7.0648E−09	7.0646E−09
	0.8	7.094E−06	7.0942E−06	7.0942E−06	2.247E−07	2.2472E−07	2.2472E−07	7.108E−09	7.1076E−09	7.1075E−09
0.5	0.2	1.690E−06	1.6914E−05	1.6910E−05	5.350E−07	5.3514E−07	5.3500E−07	1.692E−08	1.6923E−08	1.6919E−08
	0.5	1.958E−05	1.9585E−05	1.9585E−05	6.198E−07	6.1988E−07	6.1986E−07	1.960E−08	1.9604E−08	1.9603E−08
	0.8	1.970E−05	1.9712E−05	1.9712E−05	6.242E−07	6.2422E−07	6.2421E−07	1.974E−08	1.9742E−08	1.9742E−08
0.9	0.2	6.231E−06	6.2340E−06	6.2326E−06	1.971E−07	1.9721E−07	1.9716E−07	6.235E−09	6.2351E−09	6.2351E−09
	0.5	7.057E−06	7.0592E−06	7.0590E−06	2.233E−07	2.2339E−07	2.2339E−07	7.065E−09	7.0648E−09	7.0647E−09
	0.8	7.095E−06	7.0976E−06	7.0976E−06	2.247E−07	2.2473E−07	2.2473E−07	7.108E−09	7.1076E−09	7.1076E−09
CPU-time(s)			0.3851820	0.05753840		0.0388528	0.05800740		0.03926560	0.057571

Table 15 Comparison of the proposed method with the exact and available literature solution at different temporal and spatial points at $t = 2$ in Example 5

x	Cattani and Kudreyko [30]	Mittal and Arora [29]	Mittal and Jain [18]	Tamsir and Huntul [61]	Arona and Bhatia [62]	Present	Exact	Absolute Error
− 20	0.498681	0.498653	0.498652	0.498652	0.498652	0.49865179	0.49865163	1.660044E−07
− 16	0.495130	0.495745	0.495741	0.495740	0.495740	0.49574080	0.49574027	5.298864E−07
− 12	0.486758	0.486679	0.486670	0.486669	0.486669	0.48667103	0.48666922	1.805584E−06
− 8	0.459576	0.459478	0.459477	0.459478	0.459478	0.45948366	0.45947757	6.089858E−06
− 4	0.386681	0.386742	0.386787	0.386791	0.386791	0.38680714	0.38679089	1.625516E−05
2	0.158878	0.159011	0.158859	0.158852	0.158851	0.15887366	0.15884986	2.380073E−05
6	0.041822	0.041877	0.041852	0.041851	0.041850	0.04186320	0.04185133	1.187060E−05
10	0.006455	0.006426	0.006462	0.006464	0.006465	0.00646834	0.00646469	3.644098E−06
14	0.000750	0.000746	0.000754	0.000754	0.000755	0.00075577	0.00075510	6.697870E−07
18	7.617E−05	7.79E−05	0.000079	7.915E−05	7.92E−05	7.92391189E−05	7.91544789E−05	8.464001E−08
CPU-time(s)						0.031961		

Table 16 Comparison of the proposed method with the exact and available literature solution at different temporal and spatial points at $t = 4$ in Example 5

x	Cattani and Kudreyko [30]	Mittal and Arora [29]	Mittal and Jain [18]	Erdogan et al. [55]	Tamsir and Huntul [61]	Present	Exact	Absolute Error
− 20	0.498678	0.499412	0.499413	0.499413	0.499414	0.49941422	0.49941333	8.920186E−07
− 16	0.498525	0.498146	0.498142	0.498142	0.498145	0.49814248	0.49814201	4.646013E−07
− 12	0.494757	0.494149	0.494140	0.494141	0.494152	0.49414158	0.49414000	1.582161E−06
− 8	0.481776	0.481763	0.481756	0.481757	0.481787	0.48176116	0.48175570	5.458112E−06
− 4	0.445508	0.445372	0.445395	0.445397	0.445504	0.44541468	0.44539772	1.695781E−05
2	0.279025	0.280082	0.279947	0.279928	0.280193	0.27999008	0.27994126	4.882832E−05
6	0.116980	0.117196	0.116975	0.116972	0.117260	0.11700812	0.11696339	4.472505E−05
10	0.025927	0.025881	0.025967	0.025992	0.026038	0.02599378	0.02597412	1.966220E−05
14	0.003695	0.003559	0.003618	0.003627	0.003629	0.00362733	0.00362235	4.980211E−06
18	0.000409	0.000395	0.000405	0.000406	0.000408	0.00040647	0.00040569	7.790834E−07
CPU-time(s)						0.037545		

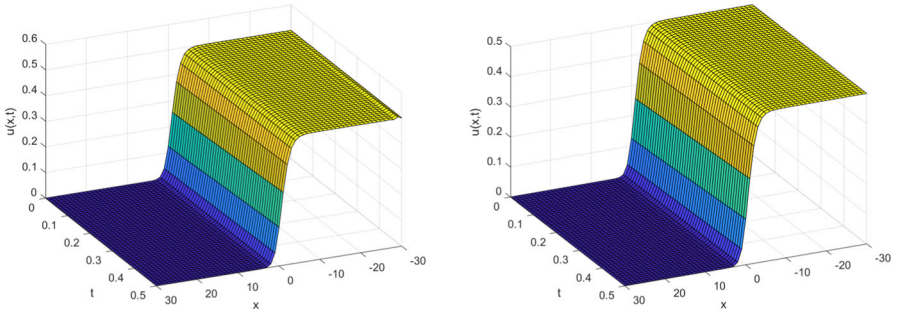

Fig. 13 Simulations of the proposed method (left) and exact (right) solutions for Fisher's equation with the viscosity value $v = 0.1$

Acknowledgements There are no funders to report for this submission.

Declarations

Conflict of interest The authors declare that this work does not have any conflicts of interest.

References

1. Chen Z, Gumel AB, Mickens RE (2003) Nonstandard discretizations of the generalized Nagumo reaction–diffusion equation. Numer Methods Partial Differ Equ 19(3):363–379
2. Namjoo M, Zeinadini M, Zibaei S (2018) Nonstandard finite-difference scheme to approximate the generalized Burgers–Fisher equation. Math Methods Appl Sci 41(17):8212–8228
3. Chen Y, Zhang T (2019) A weak Galerkin finite element method for Burgers′ equation. J Comput Appl Math 348(1):103–119
4. Kumar S, Saha Ray S (2021) Numerical treatment for Burgers–Fisher and generalized Burgers–Fisher equations. Math Sci 15:21–28
5. Khater AH, Temsah RS, Hassan MM (2008) A Chebyshev spectral collocation method for solving Burgers′-type equations. J Comput Appl Math 222(2):333–350
6. Hashim I, Noorani MSM, Al-Hadidi MRS (2006) Solving the generalized Burgers–Huxley equation using the Adomian decomposition method. Math Comput Model 43(11–12):1404–1411
7. Javidi M, Golbabai A (2009) A new domain decomposition algorithm for generalized Burger′s–Huxley equation based on Chebyshev polynomials and preconditioning. Chaos Solitions Fractals 39(2):849–857
8. Duan Y, Kong L, Zhang R (2012) A lattice Boltzmann model for the generalized Burgers–Huxley equation. Phys A 391(3):625–632
9. Seydaoglu M, Erdogan U, Ozis T (2016) Numerical solution of Burgers′ equation with high order splitting methods. J Comput Appl Math 291:410–421
10. Nascimento AA, Mariano FP, Silveria-Neto A, Padilla ELM (2014) A comparison of Fourier pseudospectral method and finite volume method used to solve the Burgers equation. J Braz Soc Mech Sci Eng 36(4):737–742
11. Hammad DA, El-Azab MS (2015) 2N order compact finite difference scheme with collocation method for solving the generalized Burger′s–Huxley and Burger′s–Fisher equations. Appl Math Comput 258:296–311
12. Lin J, Reutskiy SY (2018) An accurate meshless formulation for the simulation of linear and fully nonlinear advection diffusion reaction problems. Adv Eng Softw 126:127–146
13. Malik SA, Qureshi IM, Amir M, Malik AN, Haq I (2015) Numerical solution to generalized Burgers–Fisher equation using exp-function method hybridized with heuristic computation. PLoS ONE 10(3):e0121728

14. Wasim I, Abbas M, Amin M (2018) Hybrid B-spline collocation method for solving the generalized Burgers–Fisher and Burgers–Huxley equations. Math Probl Eng 2018:18
15. Prenter PM (1975) Splines and variational methods. Wiley, New York
16. Rubin SG, Graves RA (1975) Cubic spline approximation for problems in fluid mechanics. Nasa TR R-436, Washington
17. Mat Zin S, Abd Majid A, Ismail AIM, Abbas, M (2014) Application of hybrid cubic B-spline collocation approach for solving a generalized nonlinear Klien–Gordon equation. Math Probl Eng Article ID 108560
18. Mittal RC, Jain RK (2013) Numerical solution of non-linear Fisher′s reaction–diffusion equation with modified cubic B-splines collocation method. Math Sci 7(1):6–18
19. Mohammadi R (2012) Spline solution of the generalized Burgers–Fisher equation. Appl Anal 91(12):2189–2215
20. Mittal RC, Tripathi A (2015) Numerical solution of generalized Burgers–Fisher and generalized Burgers–Huxley equations using collocation of cubic B-Splines. Int J Comput Math 92(5):1053–1077
21. Rohila R, Mittal RC (2018) Numerical study of reaction diffusion Fisher′s equation by fourth order cubic B-spline collocation method. Math Sci 12:79–89
22. Jiwari R, Alshomrani AS (2017) A new algorithm based on modified trigonometric cubic B-splines functions for nonlinear Burgers-type equations. Int J Numer Methods Heat Fluid Flow 27(8):1638–1661
23. Jiwari R, Pandit S, Koksal ME (2019) A class of numerical algorithms based on cubic trigonometric B-spline functions for numerical simulation of nonlinear parabolic problems. Comput Appl Math 38:140
24. Tamsir M, Dhiman N, Chauhan A, Chauhan A (2021) Solution of parabolic PDEs by modified quintic B-spline Crank–Nicolson collocation method. Ains Shams Eng J 12(2):2073–2082
25. Alinia N, Zarebnia M (2019) A numerical algorithm based on a new kind of tension B-spline function for solving Burgers–Huxley equation. Numer Algorithms 82:1121–1142
26. Singh A, Dahiya S, Singh PA (2020) A fourth-order B-spline collocation method for nonlinear Burgers–Fisher equation. Math Sci 14:75–85
27. Patel KS, Mehra M (2018) A numerical study of Asian option with high-order compact finite difference scheme. J Appl Math Comput 57:467–491
28. Dag I, Irk D, Saka B (2005) A numerical solution of the Burgers equation using cubic B-splines. Appl Math Comput 163:199–211
29. Mittal RC, Arora G (2010) Efficient numerical solution of Fisher′s equation by using B-spline method. Int J Comput Math 87(13):3039–3051
30. Cattani C, Kudreyko A (2008) Multiscale analysis of the Fisher equation, ICCSA Part I, vol 5072. Lecture Notes in Computer Science. Springer, Berlin pp 1171–1180
31. Sharifi S, Rashidinia J (2019) Collocation method for convection–reaction–diffusion equations. J King Saud Univ-Sci 31(4):1115–1121
32. Nazir T, Abbas M (2021) New cubic B-spline approximation technique for numerical solutions of coupled viscous Burgers equations. Eng Comput 38(1):83–106
33. Huntul MJ, Tamsir M, Ahmadini AAH, Thottoli SR (2022) A novel collocation technique for parabolic partial differential equations. Ain Shams Eng J 13(1):101497
34. Nair LC, Awasthi A (2019) Quintic trigonometric spline based numerical scheme for nonlinear modified Burgers' equation. Numer Methods Partial Differ Equ 35:1269–1289
35. Tamsir M, Huntul MJ (2021) A numerical approach for solving Fisher's reaction–diffusion equation via a new kind of spline functions. Ain Shams Eng J 12(3):3157–3165
36. Dhiman N, Tamsir M (2018) A collocation technique based on modified form of trigonometric cubic B-spline basis functions for Fisher′s reaction–diffusion equation. Multidiscip Model Mater Struct 14(5):923–939
37. Kutluay S, Esen A (2004) A lumped Galerkin method for solving the Burgers equation. Int J Comput Math 81(11):1433–1444
38. Pandit S (2022) Local radial basis functions and scale-3 Haar wavelets operational matrices based numerical algorithms for generalized regularized long wave model. Wave Motion 109:102846
39. Appadu AR, Tijani YO (2022) 1D generalised Burgers–Huxley: proposed solutions revisited and numerical solution using FTCS and NSFD methods. Front Appl Math Stat 7:773733
40. Rasoulizadeh MN, Avazzadeh Z, Nikan O (2022) Solitary wave propagation of the generalized Kuramoto–Sivashinsky equation in fragmented porous media. Int J Appl Comput Math 8:252

41. Singh BK, Gupta M (2021) A new efficient fourth order collocation scheme for solving Burgers equation. Appl Math Comput 399:126011
42. Hepson OE, Yigit G, Allahviranloo T (2021) Numerical simulations of reaction–diffusion systems in biological and chemical mechanisms with quartic-trigonometric B-splines. Comput Appl Math 40:144
43. Hepson OE, Yigit G (2022) A numerical scheme for the wave simulations of the Kuromoto–Sivashinsky model via quartic-trigonometric tension B-spline. Wave Motion 114:103045
44. Wang L, Li H, Meng Y (2021) Numerical solution of coupled Burgers' equation using finite difference and sinc collocation method. Eng Lett 29(2):668–674
45. Hussain M, Haq S (2021) Numerical solutions of strongly non-linear generalized Burgers–Fisher equation via meshfree spectral technique. Int J Comput Math 98(9):1727–1748
46. Sharma KK, Singh P (2008) Hyperbolic partial differential-difference equation in the mathematical modeling of neuronal firing and its numerical solution. Appl Math Comput 201:229–238
47. Verma AK, Kayenat S (2019) On the stability of Mickens type NSFD schemes for generalized Burgers Fisher equation. J Differ Appl 25(12):1706–1737
48. Jain MK (1983) Numerical solution of differential equations. Wiley, New York
49. Smith GD (1985) Numerical solution of partial differential equations: finite difference methods, 3rd edn. Oxford University Press, New York
50. Jiwari R, Kumar S, Mittal RC (2019) Meshfree algorithms based on radial basis functions for numerical simulation and to capture shocks behavior of Burgers type problems. Eng Comput 36(4):1142–1168
51. Seydaoglu M (2018) An accurate approximation algorithm for Burgers equation in the presence of small viscosity. J Comput Appl Math 344:473–481
52. Tunc H, Sari M (2020) Simulations of nonlinear advection–diffusion models through various finite element techniques. Scientia Iranica B 27(6):2853–2870
53. Jiwari R (2015) A hybrid numerical scheme for the numerical solution of the Burgers equation. Comput Phys Commun 188:59–67
54. Verma AK, Rawani MK, Agarwal RP (2020) A high-order weakly L-stable time integration scheme with application to Burgers equation. Computation 8:72
55. Erdogan U, Sari M, Kocak H (2019) Efficient numerical treatment of nonlinearities in the advection–diffusion–reaction equations. Int J Numer Methods Heat Fluid Flow 29(1):132–145
56. Hussain M (2021) Hybrid radial basis function methods of lines for the numerical solution of viscous Burgers equation. Comput Appl Math 40(107):1–49
57. Ismail HNA, Raslan K, Rabboh AAA (2004) Adomian decomposition method for Burger′s–Huxley and Burger′s–Fisher equations. Appl Math Comput 159(1):291–301
58. Sari M, Gurarslan G, Dag I (2010) A compact finite difference method for the solution of the generalized Burgers–Fisher equation. Numer Methods Partial Differ Equ 26(1):125–134
59. Inan B, Bahadır AR (2015) Numerical solutions of the generalized Burgers–Huxley equation by implicit exponential finite difference method. J Appl Math Stat Inform 11:57–67
60. Wazwaz AM (2004) The tanh method for travelling wave solutions of nonlinear equations. Appl Math Comput 154(3):713–723
61. Tamsir M, Huntul MJ (2021) A numerical approach for solving Fisher′s reaction–diffusion equation via a new kind of spline functions. Ains Shams Eng J 12:3157–3165
62. Arora G, Bhatia GS (2020) A meshfree numerical technique based on radial basis function pseudospectral method for Fisher′s equation. Int J Nonlinear Sci Numer Simul 21(1):37–49