MethodsX 10 (2023) 102190

Contents lists available at ScienceDirect

MethodsX

journal homepage: www.elsevier.com/locate/mex

Orthogonal cubic splines for the numerical solution of nonlinear parabolic partial differential equations

Javad Alavi[a], Hossein Aminikhah[a,b,*]

[a] Department of Applied Mathematics and Computer Science, Faculty of Mathematical Sciences, University of Guilan, P.O. Box 1914, Rasht 41938, Iran

[b] Center of Excellence for Mathematical Modelling, Optimization and Combinational Computing (MMOCC), Iran

ARTICLE INFO

Method name:
Operational matrices of integration

Keywords:
Orthogonal cubic spline basis
Nonlinear parabolic PDEs
Operational matrices of integration
B-spline
Convergence

ABSTRACT

In this paper, a new orthogonal basis for the space of cubic splines has been introduced. A linear combination of cubic orthogonal splines is considered to approximate the functions in which the coefficients are calculated with numerically stable formulae. Applications to the numerical solutions of some parabolic partial differential equations are given, in which the approximations are obtained using the first and second integral of orthogonal splines which leads to an efficient solution procedure. The convergence analysis in the approximate scheme is investigated. A comparison of the obtained numerical solutions with some other papers indicates that the presented method is reliable and yields result with good accuracy. The main parts of our study are as follows:

- We propose a robust approach based on the orthogonal cubic splines procedure in conjunction with the operational matrix.
- The convergence in the approximate scheme is analyzed.
- Numerical examples show that the proposed method is very accurate.

Specifications table

Subject Area:	Applied Mathematics
More specific subject area:	Numerical Analysis
Method name:	Operational matrix method
Name and reference of original method:	Orthogonal cubic B-splines approach
Resource availability:	This method has been developed in MATLAB

Method details

Background

There are many practical problems in natural model systems that lead to nonlinear partial differential equations (PDEs). Usually, many assumptions are made to make some nonlinear PDEs solvable, and many of them are difficult to solve [1,2]. Due to the potential

* Corresponding author at: Department of Applied Mathematics and Computer Science, Faculty of Mathematical Sciences, University of Guilan, P.O. Box 1914, Rasht 41938, Iran.
E-mail address: hossein.aminikhah@gmail.com (H. Aminikhah).

https://doi.org/10.1016/j.mex.2023.102190
Received 6 March 2023; Accepted 15 April 2023
Available online 18 April 2023

applications of such nonlinear PDEs, the investigation of efficient solutions to solve them has been of great interest [3–6]. Meanwhile, parabolic nonlinear PDEs are used to describe a wide range of time-dependent phenomena, including heat conduction, particle emission, pricing of derivative investment instruments, elasticity, biological species, chemical reactions, environmental pollution, fluid flow, filtration of liquids, gas dynamics, etc. see [7–16].

The numerical solutions of nonlinear parabolic PDEs using splines functions have been proposed by several authors such as Raslan [15]. He has described the collocation method using quartic B-spline for the equal width (EW) equation. Mittal and Jain [17], have developed a B-spline collocation method for solving the convection-diffusion equation. They also developed a B-spline collocation method for solving nonlinear parabolic PDEs with Neumann's boundary conditions [19]. Tamsir and Dhiman [20] have presented a Cubic trigonometric B-spline differential quadrature method for the numerical treatment of Fisher's reaction-diffusion equations. Mohammadi [21] has presented Spline solution of the generalized Burgers'–Fisher equation. His numerical method is based on the exponential spline and finite difference approximations. Goh et al. [22] have also developed a numerical method using cubic B-spline for the heat and wave equation. Their schemes are second order in both time and space for the heat equation. In this paper, we consider the nonlinear parabolic partial differential equation

$$u_t = \varphi(x,t,u,u_x,u_{xx}), \quad a < x < b, \ t > 0, \tag{1}$$

with the initial condition

$$u(x,0) = u_0(x), \quad a \le x \le b, \tag{2}$$

and Dirichlet boundary conditions

$$u(a,t) = g_a(t), t \ge 0, \tag{3}$$

$$u(b,t) = g_b(t), t \ge 0. \tag{4}$$

We assume that φ, u_0, g_a and g_b and their derivatives are continuous functions in their domains.

The B-spline collocation method is not suitable for solving parabolic PDEs with Dirichlet boundary conditions. We know that when there are Dirichlet-type boundary conditions in a problem and we want to use the collocation method to solve it, the basis functions in the collocation method must vanish on the boundary, while cubic B-splines do not vanish on the boundary. So, we have to either redefine the basis functions into a new set of basis functions vanishing on the boundary where the Dirichlet's type of boundary conditions is specified [17] or to solve a parabolic PDE problem with Neumann boundary conditions [19]. For this reason and many useful features of orthogonal functions, we introduce orthogonal bases for space of splines functions in Section 1.

Orthogonal bases for the space of cubic splines

In this section, an orthogonal basis for the space of cubic spline functions will be used, which is made using the Gram-Schmidt process from the cubic B-spline basis (see Mason et al. [23], Alavi and Aminikhah [24]).

Let $\mathbb{S}_4(\Delta_n)$ be the space of cubic splines on the knot sequence $\Delta_n = \{a = \varsigma_{-n}, \varsigma_{-n+1}, \dots, \varsigma_{n-1}, \varsigma_n = b\}$ and let $\{\Xi_k\}_{k=-n-1}^{n+1}$ be the cubic B-splines basis on the knot sequence $\{a = \varsigma_{-n-3} = \varsigma_{-n-2} = \varsigma_{-n-1} = \varsigma_{-n}, \varsigma_{-n+1}, \dots, \varsigma_{n-1}, \varsigma_n = \varsigma_{n+1} = \varsigma_{n+2} = \varsigma_{n+3} = b\}$, where $\varsigma_{k+1} - \varsigma_k = h$, $k = -n, -n+1, \dots, n-1$.

Using Gram-Schmidt process on $\{\Xi_k\}_{k=-n-1}^{n+1}$ the orthogonal basis $\{\mathfrak{P}_k\}_{k=-n-1}^{n+1}$ will be obtain as follow

$$\begin{cases}
\mathfrak{P}_{-n-1} = \Xi_{-n-1}, \\
\mathfrak{P}_{-n} = \Xi_{-n} - \varrho_{-n,1}\mathfrak{P}_{-n-1}, \\
\mathfrak{P}_{-n+1} = \Xi_{-n+1} - \varrho_{-n+1,1}\mathfrak{P}_{-n} - \varrho_{-n+1,2}\mathfrak{P}_{-n-1}, \\
\mathfrak{P}_{-j} = \Xi_{-j} - \varrho_{-j,1}\mathfrak{P}_{-(j+1)} - \varrho_{-j,2}\mathfrak{P}_{-(j+2)} - \varrho_{-j,3}\mathfrak{P}_{-(j+3)}, \\
\mathfrak{P}_{n+1} = \Xi_{n+1}, \\
\mathfrak{P}_{n} = \Xi_{n} - \varrho_{n,1}\mathfrak{P}_{n+1}, \\
\mathfrak{P}_{n-1} = \Xi_{n-1} - \varrho_{n-1,1}\mathfrak{P}_{n} - \varrho_{n-1,2}\mathfrak{P}_{n+1}, \\
\mathfrak{P}_{j} = \Xi_{j} - \varrho_{j,1}\mathfrak{P}_{j+1} - \varrho_{j,2}\mathfrak{P}_{j+2} - \varrho_{j,3}\mathfrak{P}_{j+3}, \\
\hat{\mathfrak{P}}_{-1} = \Xi_{-1} - \varrho_{-1,1}\mathfrak{P}_{-2} - \varrho_{-1,2}\mathfrak{P}_{-3} - \varrho_{-1,3}\mathfrak{P}_{-4} - \varrho_{-1,4}\mathfrak{P}_{2}, \\
\hat{\mathfrak{P}}_{1} = \Xi_{1} - \varrho_{1,1}\mathfrak{P}_{2} - \varrho_{1,2}\mathfrak{P}_{3} - \varrho_{1,3}\mathfrak{P}_{4} - \varrho_{1,4}\mathfrak{P}_{-2}, \\
\mathfrak{P}_{-1} = A\hat{\mathfrak{P}}_{-1} + B\hat{\mathfrak{P}}_{1}, \\
\mathfrak{P}_{1} = A\hat{\mathfrak{P}}_{1} + B\hat{\mathfrak{P}}_{-1}, \\
\mathfrak{P}_{0} = \Xi_{0} - \varrho_{0,-1}\mathfrak{P}_{-1} - \varrho_{0,-2}\mathfrak{P}_{-2} - \varrho_{0,-3}\mathfrak{P}_{-3} - \varrho_{0,1}\mathfrak{P}_{1} - \varrho_{0,2}\mathfrak{P}_{2} - \varrho_{0,3}\mathfrak{P}_{3}.
\end{cases} \tag{5}$$

where $j = n-2, n-3 \dots, 2$. The coefficients in (5) are introduced in Alavi and Aminikhah [24].

Proposition 1 [24]. *If* $f \in C^4[a,b]$ *and* $S_n(f)(x) = \sum_{k=-n-1}^{n+1} c_k \mathfrak{P}_k(x)$ *then* $S_n(f)$ *is uniformly converges to* f. *That is* $\lim_{n\to\infty} \|f - S_n(f)\|_\infty = 0$.

For convenience in numerical computations, we use vector form of $S_n(f)$ as follows

$$S_n(f)(x) = \sum_{k=-n-1}^{n+1} c_k \mathfrak{P}_k(x) = C_n^T \Pi_n(x), \tag{6}$$

where

$$C_n = \left(c_{-n-1}, c_{-n}, \dots, c_{n+1}\right)^T, \tag{7}$$

$$\Pi_n(x) = \left(\mathfrak{P}_{-n-1}(x), \mathfrak{P}_{-n}(x), \dots, \mathfrak{P}_{n+1}(x)\right)^T. \tag{8}$$

Furthermore, in order to achieve a square system in numerical computations the set of the nodes $\Delta_n^* = \left(\xi_i\right)_{i=-n-1}^{n+1}$ is used, where

$$\begin{cases} \xi_{-n-1} = \varsigma_{-n},\ \xi_{-n} = \varsigma_{-n} + \frac{h}{2}, \\ \xi_{-n+1} = \varsigma_{-n+1},\ \xi_{-n+2} = \varsigma_{-n+2}, \dots,\ \xi_{n-2} = \varsigma_{n-2}, \\ \xi_{n-1} = \varsigma_{n-1},\ \xi_n = \varsigma_n - \frac{h}{2},\ \xi_{n+1} = \varsigma_n. \end{cases}$$

Remark 1. Assume that $\mathcal{B}$, $I_1\mathcal{B}$, $I_2\mathcal{B}$, $I_1\mathcal{P}$ and $I_2\mathcal{P}$ are $(2n+3)$-square matrices defined by

$$(\mathcal{B})_{i,j} = \Xi_i(\xi_j), \quad \left(I_1\mathcal{B}\right)_{i,j} = \int_a^{\xi_j} \Xi_i(y)dy, \quad \left(I_2\mathcal{B}\right)_{i,j} = \int_a^{\xi_j}\int_a^{z} \Xi_i(y)dydz,$$
$$\left(I_1\mathcal{P}\right)_{i,j} = \int_a^{\xi_j} \mathfrak{P}_i(y)dy, \quad \left(I_2\mathcal{P}\right)_{i,j} = \int_a^{\xi_j}\int_a^{z} \mathfrak{P}_i(y)dydz.$$

where $i, j = -n-1, -n, \dots, n+1$. According to the definition of Ξ_ks we can calculate the elements of matrices $\mathcal{B}$, $I_1\mathcal{B}$ and $I_2\mathcal{B}$ as follows

$$\mathcal{B} = \begin{pmatrix}
1 & \frac{1}{8} & & & & & & & & & \\
 & \frac{19}{32} & \frac{1}{4} & & & & & & & & \\
 & \frac{25}{96} & \frac{7}{12} & \frac{1}{6} & & & & & & & \\
 & \frac{1}{48} & \frac{1}{6} & \frac{2}{3} & \frac{1}{6} & & & & & & \\
 & & & \frac{1}{6} & \frac{2}{3} & \frac{1}{6} & & & & & \\
 & & & & \ddots & \ddots & \ddots & & & & \\
 & & & & & \frac{1}{6} & \frac{2}{3} & \frac{1}{6} & & & \\
 & & & & & & \frac{1}{6} & \frac{2}{3} & \frac{1}{6} & \frac{1}{48} & \\
 & & & & & & & \frac{1}{6} & \frac{7}{12} & \frac{25}{96} & \\
 & & & & & & & & \frac{1}{4} & \frac{19}{32} & \\
 & & & & & & & & & \frac{1}{8} & 1
\end{pmatrix},$$

$$I_1\mathcal{B} = h\begin{pmatrix}
0 & \frac{15}{64} & \frac{1}{4} & \frac{1}{4} & \frac{1}{4} & \frac{1}{4} & \frac{1}{4} & \cdots & \frac{1}{4} & \frac{1}{4} & \frac{1}{4} \\
0 & \frac{55}{256} & \frac{7}{16} & \frac{1}{2} & \frac{1}{2} & \frac{1}{2} & \frac{1}{2} & \cdots & \frac{1}{2} & \frac{1}{2} & \frac{1}{2} \\
0 & \frac{37}{768} & \frac{13}{48} & \frac{17}{24} & \frac{3}{4} & \frac{3}{4} & \frac{3}{4} & \cdots & \frac{3}{4} & \frac{3}{4} & \frac{3}{4} \\
0 & \frac{1}{384} & \frac{1}{24} & \frac{1}{2} & \frac{23}{24} & 1 & 1 & \dots & 1 & 1 & 1 \\
 & & & \frac{1}{24} & \frac{1}{2} & \frac{23}{24} & 1 & \dots & 1 & 1 & 1 \\
 & & & & \ddots & \ddots & \ddots & & & & \\
 & & & & & \frac{1}{24} & \frac{1}{2} & \frac{23}{24} & 1 & 1 & 1 \\
 & & & & & & \frac{1}{24} & \frac{1}{2} & \frac{23}{24} & \frac{383}{384} & 1 \\
 & & & & & & & \frac{1}{24} & \frac{23}{48} & \frac{539}{768} & \frac{3}{4} \\
 & & & & & & & & \frac{1}{16} & \frac{73}{256} & \frac{1}{2} \\
 & & & & & & & & & \frac{1}{64} & \frac{1}{4}
\end{pmatrix},$$

$$I_2\mathcal{B} = h^2\begin{pmatrix}
0 & \frac{49}{640} & \frac{4}{20} & \frac{9}{20} & \frac{14}{20} & \frac{19}{20} & \frac{24}{20} & \cdots & \frac{4+5(2n-2)}{20} & \frac{20n-7}{40} & \frac{4+5(2n-1)}{20}\\
0 & \frac{107}{2560} & \frac{17}{80} & \frac{7}{10} & \frac{12}{10} & \frac{17}{10} & \frac{22}{10} & \cdots & \frac{7+5(2n-3)}{10} & \frac{20n-11}{20} & \frac{7+5(2n-2)}{10}\\
0 & \frac{49}{7680} & \frac{19}{240} & \frac{73}{120} & \frac{27}{20} & \frac{42}{20} & \frac{57}{20} & \cdots & \frac{27+15(2n-4)}{20} & \frac{60n-51}{40} & \frac{27+15(2n-3)}{20}\\
0 & \frac{1}{3840} & \frac{1}{120} & \frac{7}{30} & \frac{121}{120} & 2 & 3 & \cdots & 2n-3 & \frac{4n-5}{2} & 2n-2\\
 & & & \frac{1}{120} & \frac{7}{30} & \frac{121}{120} & 2 & & 2n-4 & \frac{4n-7}{2} & 2n-3\\
 & & & & \ddots & \ddots & \ddots & \ddots & & & \\
 & & & & & \frac{1}{120} & \frac{7}{30} & \frac{121}{120} & 2 & \frac{5}{2} & 3\\
 & & & & & & \frac{1}{120} & \frac{7}{30} & \frac{121}{120} & \frac{5761}{3840} & 2\\
 & & & & & & & \frac{1}{120} & \frac{11}{48} & \frac{4081}{7680} & \frac{9}{10}\\
 & & & & & & & & \frac{1}{80} & \frac{47}{512} & \frac{3}{10}\\
 & & & & & & & & & \frac{1}{640} & \frac{1}{20}
\end{pmatrix}.$$

Furthermore (5) entail

$$\begin{cases}
(I_\nu\mathcal{P})_{-n-1,i} = (I_\nu\mathcal{B})_{-n-1,i}, \quad (I_\nu\mathcal{P})_{-n,i} = (I_\nu\mathcal{B})_{-n,i} - \varrho_{-n,1}(I_\nu\mathcal{P})_{-n-1,i},\\
(I_\nu\mathcal{P})_{-n+1,i} = (I_\nu\mathcal{B})_{-n+1,i} - \varrho_{-n+1,1}(I_\nu\mathcal{P})_{-n,i} - \varrho_{-n+1,2}(I_\nu\mathcal{P})_{-n-1,i},\\
(I_\nu\mathcal{P})_{-j,i} = (I_\nu\mathcal{B})_{-j,i} - \varrho_{-j,1}(I_\nu\mathcal{P})_{-(j+1),i} - \varrho_{-j,2}(I_\nu\mathcal{P})_{-(j+2),i} - \varrho_{-j,3}(I_\nu\mathcal{P})_{-(j+3),i},\\
(I_\nu\mathcal{P})_{n+1,i} = (I_\nu\mathcal{B})_{n+1,i}, \quad (I_\nu\mathcal{P})_{n,i} = (I_\nu\mathcal{B})_{n,i} - \varrho_{n,1}(I_\nu\mathcal{P})_{n+1,i},\\
(I_\nu\mathcal{P})_{n-1,i} = (I_\nu\mathcal{B})_{n-1,i} - \varrho_{n-1,1}(I_\nu\mathcal{P})_{n,i} - \varrho_{n-1,2}\mathcal{P}_{n+1,i},\\
(I_\nu\mathcal{P})_{j,i} = (I_\nu\mathcal{B})_{j,i} - \varrho_{j,1}(I_\nu\mathcal{P})_{j+1,i} - \varrho_{j,2}(I_\nu\mathcal{P})_{j+2,i} - \varrho_{j,3}(I_\nu\mathcal{P})_{j+3,i},\\
(I_\nu\hat{\mathcal{P}})_{-1,i} = (I_\nu\mathcal{B})_{-1,i} - \varrho_{-1,1}(I_\nu\mathcal{P})_{-2,i} - \varrho_{-1,2}(I_\nu\mathcal{P})_{-3,i} - \varrho_{-1,3}(I_\nu\mathcal{P})_{-4,i} - \varrho_{-1,4}(I_\nu\mathcal{P})_{2,i},\\
(I_\nu\hat{\mathcal{P}})_{1,i} = (I_\nu\mathcal{B})_{1,i} - \varrho_{1,1}(I_\nu\mathcal{P})_{2,i} - \varrho_{1,2}(I_\nu\mathcal{P})_{3,i} - \varrho_{1,3}(I_\nu\mathcal{P})_{4,i} - \varrho_{1,4}(I_\nu\mathcal{P})_{-2,i},\\
(I_\nu\mathcal{P})_{-1,i} = \frac{\alpha}{1+\alpha^2}(I_\nu\hat{\mathcal{P}})_{-1,i} + \frac{1}{1+\alpha^2}(I_\nu\hat{\mathcal{P}})_{1,i},\\
(I_\nu\mathcal{P})_{1,i} = \frac{\alpha}{1+\alpha^2}(I_\nu\hat{\mathcal{P}})_{1,i} + \frac{1}{1+\alpha^2}(I_\nu\hat{\mathcal{P}})_{-1,i},\\
(I_\nu\mathcal{P})_{0,i} = (I_\nu\mathcal{B})_{0,i} - \varrho_{0,-1}(I_\nu\mathcal{P})_{-1,i} - \varrho_{0,-2}(I_\nu\mathcal{P})_{-2,i} - \varrho_{0,-3}(I_\nu\mathcal{P})_{-3,i} - \varrho_{0,1}(I_\nu\mathcal{P})_{1,i}\\
\qquad\qquad -\varrho_{0,2}(I_\nu\mathcal{P})_{2,i} - \varrho_{0,3}(I_\nu\mathcal{P})_{3,i}.
\end{cases}$$

where $j = n-2, n-3, \dots, 2$, $i = -n-1, -n, \dots, n+1$ and $\nu = 1, 2$. Thus we can write

$$\int_a^{\xi_j} S_n(f)(y)dy = C_n^T I_1^j, \tag{9}$$

$$\int_{ya}^{\xi_j}\int_a^z S_n(f)(y)dydz = C_n^T I_2^j, \tag{10}$$

where I_ν^j is jth column of matrix $I_\nu\mathcal{P}$, $\nu = 1, 2$.

Application of the Method to parabolic partial differential equation

In this section, we have tried to solve parabolic partial differential Eqs. (1)–(4) using S_n, as an approximation tool. Let $0 \le t \le T$ and $t_s = s\Delta t$, $s = 0, 1, \dots, S$ are the equal parts of $[0, T]$ where $\Delta t = \frac{T}{S}$. To discretize the problem (1)–(4), the method of Hariharan et al. [25] is used. We assume that $u_{txx}(x,t)$ can be expanded in terms of cubic orthogonal splines (6) as

$$u_{txx}(x,t) = \sum_{k=-n-1}^{n+1} c_k^s \mathfrak{P}_k(x) = C_n^T \Pi_n(x), \tag{11}$$

where C_n and Π_n are given by (7) and (8). The row vector C_n^T is assumed constant in the subinterval $[t_s, t_{s+1}]$. By integrating (11) with respect to t from t_s to t, we obtain

$$u_{xx}(x,t) = u_{xx}(x,t_s) + (t-t_s)C_n^T\Pi_n(x). \tag{12}$$

Also, by integrating (11) with respect to x from a to x we have

$$u_{tx}(x,t) = u_{tx}(a,t) + \sum_{k=-n-1}^{n+1} c_k^s \int_a^x \mathfrak{P}_k(y)dy \tag{13}$$

Integrating (13) with respect to x from a to x and using (3), gives

$$u_t(x,t) = \dot{g}_a(t) + (x-a)u_{tx}(a,t) + \sum_{k=-n-1}^{n+1} c_k^s \int_a^x\int_a^z \mathfrak{P}_k(y)dydz, \tag{14}$$

where $\dot{}$ denotes the differentiation with respect to t. By substituting $x = b$ in Eq. (14) and using (4), we get

$$u_{tx}(a,t) = \frac{1}{b-a}\left(\dot{g}_b(t) - \dot{g}_a(t) - \sum_{k=-n-1}^{n+1} c_k^s \int_a^b \int_a^z \mathfrak{P}_k(y)dydz\right). \tag{15}$$

Substituting Eq. (15) into (14), held

$$u_t(x,t) = \dot{g}_a(t) + \frac{x-a}{b-a}\left(\dot{g}_b(t) - \dot{g}_a(t)\right) + \sum_{k=-n-1}^{n+1} c_k^s \left(\int_a^x \int_a^z \mathfrak{P}_k(y)dydz - \frac{x-a}{b-a}\int_a^b \int_a^z \mathfrak{P}_k(y)dydz\right). \tag{16}$$

By integrating (16) with respect to t from t_s to t, we obtain

$$u(x,t) = u(x,t_s) + g_a(t) - g_a(t_s) + \frac{x-a}{b-a}\left(g_b(t) - g_b(t_s) - g_a(t) + g_a(t_s)\right) + (t-t_s)\sum_{k=-n-1}^{n+1} c_k^s \left(\int_a^x \int_a^z \mathfrak{P}_k(y)dydz - \frac{x-a}{b-a}\int_a^b \int_a^z \mathfrak{P}_k(y)dydz\right) \tag{17}$$

Also, by substituting Eq. (15) in (13) we get

$$u_{tx}(x,t) = \frac{1}{b-a}\left(\dot{g}_b(t) - \dot{g}_a(t)\right) + \sum_{k=-n-1}^{n+1} c_k^s \left(\int_a^x \mathfrak{P}_k(y)dy - \frac{1}{b-a}\int_a^b \int_a^z \mathfrak{P}_k(y)dydz\right). \tag{18}$$

Integrating (18) with respect to t from t_s to t, yields

$$u_x(x,t) = u_x(x,t_s) + \frac{1}{b-a}\left(g_b(t) - g_b(t_s) - g_a(t) + g_a(t_s)\right) + (t-t_s)\sum_{k=-n-1}^{n+1} c_k^s \left(\int_a^x \mathfrak{P}_k(y)dy - \frac{1}{b-a}\int_a^b \int_a^z \mathfrak{P}_k(y)dydz\right). \tag{19}$$

Further, by discretizing (12), (16), (17), (19), assuming $x \to \xi_j$, $t \to t_{s+1}$, and using (9) and (10), we get

$$u_{xx}(\xi_j, t_{s+1}) = u_{xx}(\xi_j, t_s) + \Delta t C_n^T \Pi_n(\xi_j), \tag{20}$$

$$u_t(\xi_j, t_{s+1}) = \dot{g}_a(t_{s+1}) + \lambda_j\left(\dot{g}_b(t_{s+1}) - \dot{g}_a(t_{s+1})\right) + C_n^T\left(I_2^j - \lambda_j I_2^{n+1}\right), \tag{21}$$

$$u(\xi_j, t_{s+1}) = u(\xi_j, t_s) + g_a(t_{s+1}) - g_a(t_s) + \lambda_j\left(g_b(t_{s+1}) - g_b(t_s) - g_a(t_{s+1}) + g_a(t_s)\right) + \Delta t C_n^T\left(I_2^j - \lambda_j I_2^{n+1}\right), \tag{22}$$

$$u_x(\xi_j, t_{s+1}) = u_x(\xi_j, t_s) + \frac{1}{b-a}\left(g_b(t_{s+1}) - g_b(t_s) - g_a(t_{s+1}) + g_a(t_s)\right) + \Delta t C_n^T\left(I_1^j - \frac{1}{b-a} I_2^{n+1}\right), \tag{23}$$

where $\lambda_j = \frac{\xi_j - a}{b-a}$. Assuming $x \to \xi_j$, $t \to t_{s+1}$ in (1), we get

$$u_t\left(\xi_j, t_{s+1}\right) = \varphi\left(\xi_j, t_{s+1}, u\left(\xi_j, t_{s+1}\right), u_x\left(\xi_j, t_{s+1}\right), u_{xx}\left(\xi_j, t_{s+1}\right)\right). \tag{24}$$

Substituting (20)–(23) in (24) gives a system of $(2n+3)$ equations in $(2n+3)$ unknowns c_k^s, $k = -n-1, -n, \ldots, n+1$.

If Eq. (1) is linear, then (24) leads to a system of linear equations that will be easily solvable. In the cases that (1) be nonlinear, then Eq. (24) leads to a system of nonlinear equations that will be solved by the trust-region-dogleg algorithm [26].

Convergence analysis

To prove convergence of the solution of the presented method, we need to show that the maximum errors tend to zero as $h \to 0$, $\Delta \to 0$.

Assume that $\tilde{u}_{txx}(x,t) = \sum_{k=-n-1}^{n+1} c_k^s \mathfrak{P}_k(x)$, $t \in [t_s, t_{s+1}]$ is the approximation of the exact solution $u_{txx}(x,t)$. Let

$$e_n(x,t) = u_{txx}(x,t) - \tilde{u}_{txx}(x,t), \tag{25}$$

represent error in (x,t). Using Proposition 1 we know that

$$\lim_{n\to\infty} \left\| e_n(x,t_j) \right\|_\infty = 0. \tag{26}$$

Now, the numerical method presented in Section 2, could be rewritten along with the error terms.

By integrating (25) with respect to x from a to x, leads to

$$\tilde{u}_{tx}(x,t) - u_{tx}(x,t) = \tilde{u}_{tx}(a,t) - u_{tx}(a,t) - \int_a^x e_n(y,t)dy. \tag{27}$$

According to the initial and boundary conditions (2), (3), and (4), we set

$$\begin{cases} u(x,0) = \tilde{u}(x,0), \\ u(a,t) = \tilde{u}(a,t), \\ u(b,t) = \tilde{u}(b,t), \end{cases} \tag{28}$$

By integrating (27) with respect to x from a to x and using (28) we have

$$\tilde{u}_t(x,t) - u_t(x,t) = (x-a)\big(\tilde{u}_{tx}(a,t) - u_{tx}(a,t)\big) - \int_a^x \int_a^z e_n(y,t)\,dy\,dz. \tag{29}$$

Putting $x = b$ in (29) and using (28) gives

$$\tilde{u}_{tx}(a,t) - u_{tx}(a,t) = \frac{1}{b-a}\int_a^b \int_a^z e_n(y,t)\,dy\,dz. \tag{30}$$

By substituting (30) in (29) we obtain

$$\tilde{u}_t(x,t) - u_t(x,t) = \frac{x-a}{b-a}\int_a^b \int_a^z e_n(y,t)\,dy\,dz - \int_a^x \int_a^z e_n(y,t)\,dy\,dz. \tag{31}$$

Integrating (31) with respect to t from t_s to t, leads to

$$\tilde{u}(x,t) - u(x,t) = \tilde{u}(x,t_s) - u(x,t_s) + \frac{x-a}{b-a}\int_a^b \int_a^z \int_{t_s}^t e_n(y,t)\,dt\,dy\,dz - \int_a^x \int_a^z \int_{t_s}^t e_n(y,t)\,dt\,dy\,dz. \tag{32}$$

Applying Eq. (32) successively implies

$$\begin{aligned}
\tilde{u}(x,t_{s+1}) - u(x,t_{s+1}) &= \tilde{u}(x,t_s) - u(x,t_s) \\
&\quad + \frac{x-a}{b-a}\int_a^b \int_a^z \int_{t_s}^{t_{s+1}} e_n(y,t)\,dt\,dy\,dz - \int_a^x \int_a^z \int_{t_s}^{t_{s+1}} e_n(y,t)\,dt\,dy\,dz \\
&= \tilde{u}(x,t_{s-1}) - u(x,t_{s-1}) \\
&\quad + \frac{x-a}{b-a}\int_a^b \int_a^z \int_{t_{s-1}}^{t_s} e_n(y,t)\,dt\,dy\,dz - \int_a^x \int_a^z \int_{t_{s-1}}^{t_s} e_n(y,t)\,dt\,dy\,dz \\
&\quad + \frac{x-a}{b-a}\int_a^b \int_a^z \int_{t_s}^{t_{s+1}} e_n(y,t)\,dt\,dy\,dz - \int_a^x \int_a^z \int_{t_s}^{t_{s+1}} e_n(y,t)\,dt\,dy\,dz \\
&= \\
&\quad \vdots \\
&= \tilde{u}(x,t_0) - u(x,t_0) \\
&\quad + \frac{x-a}{b-a}\int_a^b \int_a^z \int_{t_0}^{t_1} e_n(y,t)\,dt\,dy\,dz - \int_a^x \int_a^z \int_{t_0}^{t_1} e_n(y,t)\,dt\,dy\,dz \\
&\quad \vdots \\
&\quad + \frac{x-a}{b-a}\int_a^b \int_a^z \int_{t_{s-1}}^{t_s} e_n(y,t)\,dt\,dy\,dz - \int_a^x \int_a^z \int_{t_{s-1}}^{t_s} e_n(y,t)\,dt\,dy\,dz \\
&\quad + \frac{x-a}{b-a}\int_a^b \int_a^z \int_{t_s}^{t_{s+1}} e_n(y,t)\,dt\,dy\,dz - \int_a^x \int_a^z \int_{t_s}^{t_{s+1}} e_n(y,t)\,dt\,dy\,dz.
\end{aligned}$$

Now the Eq. (28) concludes

$$\tilde{u}(x,t_{s+1}) - u(x,t_{s+1}) = \sum_{j=0}^{s}\left(\frac{x-a}{b-a}\int_a^b \int_a^z \int_{t_j}^{t_{j+1}} e_n(y,t)\,dt\,dy\,dz - \int_a^x \int_a^z \int_{t_j}^{t_{j+1}} e_n(y,t)\,dt\,dy\,dz\right). \tag{33}$$

Let $\max_{t_j \le t \le t_{j+1}} |e_n(y,t)| = \left|e_n(y,t_{j*})\right|$, then from (33) we have

$$\left|\tilde{u}(x,t_{s+1}) - u(x,t_{s+1})\right| \le \Delta t \sum_{j=0}^{s}\left(\frac{x-a}{b-a}\int_a^b \int_a^z \left|e_n(y,t_{j*})\right| dy\,dz + \int_a^x \int_a^z \left|e_n(y,t_{j*})\right| dy\,dz\right). \tag{34}$$

From relation (34) we obtain

$$\begin{aligned}\left\|\tilde{u}(x,t_{s+1})-u(x,t_{s+1})\right\|_\infty &\le \Delta t\sum_{j=0}^{s}\left\|e_n(y,t_{j*})\right\|_\infty\left(\frac{x-a}{b-a}\int_a^b\int_a^z dydz+\int_a^x\int_a^z dydz\right)\\ &=\frac{1}{2}\left((b-a)(x-a)+(x-a)^2\right)\Delta t\sum_{j=0}^{s}\left\|e_n(y,t_{j*})\right\|_\infty\\ &\le \Delta t(b-a)^2\sum_{j=0}^{s}\left\|e_n(y,t_{j*})\right\|_\infty. \end{aligned}\tag{35}$$

From Eqs. (35), (26) we have $\lim_{h\to 0}\left\|\tilde{u}(x,t_{s+1})-u(x,t_{s+1})\right\|_\infty=0$ and, for fixed x (35) leads to $\lim_{\Delta t\to 0}\left\|\tilde{u}(x,t_{s+1})-u(x,t_{s+1})\right\|_\infty=0$.

Numerical experiments

In order to show the efficiency and accuracy of the presented method, three numerical examples are prepared in this section. In numerical examples, we suppose that $u(x,t)$ denotes the exact solution and $\tilde{u}(x,t)$ denotes the estimated solution.

The versatility and the accuracy of the methods are measured using the L_∞ and L_2 error norms which are defined as [27]

$$L_\infty=\max_j\left|u(x_j,t)-\tilde{u}(x_j,t)\right|,$$

$$L_2=\sqrt{\frac{\sum_j\left|u(x_j,t)-\tilde{u}(x_j,t)\right|^2}{\sum_j\left|u(x_j,t)\right|^2}},$$

where $t\in[0,T]$ ia a fix number.

Example 1. In this problem we consider the following convection-diffusion equation

$$u_t(x,t)+\epsilon u_x(x,t)=\gamma u_{xx}(x,t),\ 0<x<1,\ t>0.\tag{36}$$

where $\epsilon=0.1$, $\gamma=0.02$ and with the initial condition

$$u_0(x)=\exp(\alpha x).$$

The exact solution of (36) is

$$u(x)=\exp(\alpha x+\beta t),$$

where $\alpha=1.17712434446770$ and $\beta=-0.09$. The boundary conditions $g_a(t)=u(0,t)$ and $g_b(t)=u(1,t)$ can be gained from the exact solution. According to (24) and (36), we conclude

$$C_n^T\mathbf{z}_j=r_{s,j},\tag{37}$$

in which

$$\mathbf{z}_j=I_2^j-\lambda_j I_2^{n+1}+\varepsilon\Delta t\left(I_1^j-\frac{1}{b-a}I_2^{n+1}\right)-\gamma\Delta t\Pi_n(\xi_j),$$

$$r_{s,j}=\gamma u_{xx}(\xi_j,t_s)-\epsilon u_x(\xi_j,t_s)-\frac{\epsilon}{b-a}\left(g_b(t_{s+1})-g_b(t_s)-g_a(t_{s+1})+g_a(t_s)\right)-\dot{g}_a(t_{s+1})-\lambda_j\left(\dot{g}_b(t_{s+1})-\dot{g}_a(t_{s+1})\right).$$

By rewriting (37) for $j=-n-1,-n,\dots,n+1$, the following main system is produced

$$A^TC_n=\mathbf{r}_s,\tag{38}$$

so that $A=\left(\mathbf{z}_{-n-1},\mathbf{z}_{-n},\dots,\mathbf{z}_{n+1}\right)$ and $\mathbf{r}_s=\left(r_{s,-n-1},r_{s,-n},\dots,r_{s,n+1}\right)^T$.

For each s, the amounts of C_n are computed after solving the linear system (38). Then, the values of $u\left(\xi_j,t_{s+1}\right)$ are uncovered applying (22).

Note that for $s=0$ in (2), $u_x\left(\xi_j,0\right)=u_0'(x)$, and $u_{xx}\left(\xi_j,0\right)=u_0''(x)$, otherwise $u_x\left(\xi_j,t_s\right)$ and $u_{xx}\left(\xi_j,t_s\right)$ are updated using (23) and (20), respectively.

The maximum absolute errors at the final time $t=3$ of the proposed method and some spline-based methods include; redefined cubic B-splines collocation (method I) [17,18], cubic B-Spline Quasi-Interpolation (method II) [28–30] and B-spline differences (method III) [31] are reported in Table 1. Furthermore, the error norms L_∞ and L_2 for the proposed method and methods I, II and, III are tabulated in Table 2. In Table 3, the maximum absolute errors in final time $t=3$ for fixed $h=0.1$ and different values of Δt are shown. In Fig. 1, the logarithm of absolute errors at the final time $t=3$ of the proposed method and methods I, II and, III are depicted. The exact solutions and absolute errors of the numerical method for (36) are depicted in Fig. 2. In order to have a fair comparison, we assume that $h=0.01$ and $\Delta t=0.0001$ in all methods.

Example 2. In this problem, we consider the following generalized Fisher equation [32]

$$u_t=((1-u)u_x)_x+2u-2u^2,\ -\frac{\pi}{2}<x<\frac{\pi}{2},\ t>0.\tag{39}$$

Table 1
Maximum absolute errors of Example 1 for different spline methods in $t = 3$ for $h = 0.01$ and $\Delta t = 0.0001$.

x	Present method	Method I	Method II	Method III
0.1	$4.095409e-08$	$2.525707e-07$	$5.690595e-07$	$6.711264e-08$
0.2	$9.171525e-08$	$5.007888e-07$	$1.131648e-06$	$1.330686e-07$
0.3	$1.451018e-07$	$7.381093e-07$	$1.669654e-06$	$1.961289e-07$
0.4	$1.940863e-07$	$9.594134e-07$	$2.171395e-06$	$2.549332e-07$
0.5	$2.319977e-07$	$1.158246e-06$	$2.622121e-06$	$3.077662e-07$
0.6	$2.524701e-07$	$1.320182e-06$	$2.988978e-06$	$3.507948e-07$
0.7	$2.490144e-07$	$1.410071e-06$	$3.192103e-06$	$3.746789e-07$
0.8	$2.140111e-07$	$1.349596e-06$	$3.053780e-06$	$3.586091e-07$
0.9	$1.367619e-07$	$9.797077e-07$	$2.213315e-06$	$2.603237e-07$

Table 2
The error norms of Example 1 for different spline methods for $h = 0.01$ and $\Delta t = 0.0001$.

t	Error norm	Present method	Method I	Method II	Method III
1	L_∞	$1.149070e-07$	$6.676803e-07$	$1.510797e-06$	$1.773833e-07$
	L_2	$4.372485e-08$	$2.564121e-07$	$5.800060e-07$	$6.813040e-08$
2	L_∞	$1.984331e-07$	$1.113716e-06$	$2.521026e-06$	$2.959304e-07$
	L_2	$8.136983e-08$	$4.639174e-07$	$1.049667e-06$	$1.232706e-07$
3	L_∞	$2.543450e-07$	$1.413629e-06$	$3.199975e-06$	$3.756242e-07$
	L_2	$1.128187e-07$	$6.349489e-07$	$1.436727e-06$	$1.687166e-07$

Table 3
The maximum absolute errors of Example 1 for $t = 3$ and $h = 0.1$.

Δt	Max. absolute error
1/2	$1.197011e-03$
1/4	$6.141829e-04$
1/8	$3.112529e-04$
1/16	$1.567016e-04$
1/32	$7.862573e-05$
1/64	$3.938392e-05$
1/128	$1.971156e-05$
1/256	$9.862463e-06$
1/512	$4.934673e-06$
1/1024	$2.469967e-06$
1/2048	$1.237412e-06$
1/4096	$6.210833e-07$

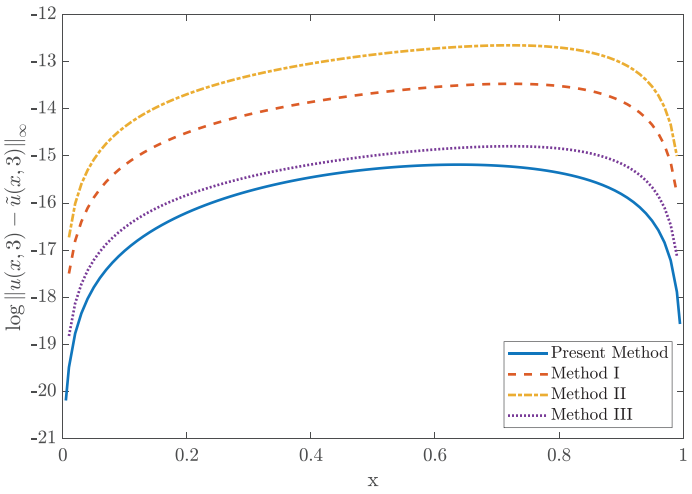

Fig. 1. The logarithms of absolute errors of Example 1 for different spline methods in $t = 3$ for $h = 0.01$ and $\Delta t = 0.0001$.

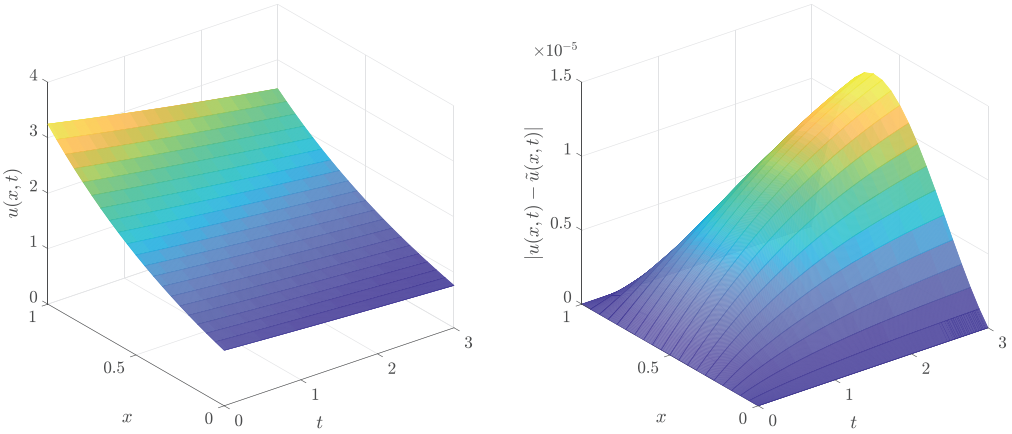

Fig. 2. The exact solutions (left) and the absolute errors (right) of Example 1 with $h = 0.05$ and $\Delta t = 0.005$.

Table 4
Maximum absolute errors of Example 2 for different spline methods in $t = 1.5$ for $h = \pi/20$ and $\Delta t = 0.0001$.

x	Presented method	Method II	Method III	Method IV
$-9\pi/20$	$3.940962e-07$	$3.875122e-07$	$2.573821e-07$	$2.309600e-07$
$-7\pi/20$	$8.321440e-07$	$8.617166e-07$	$1.935984e-06$	$1.497486e-06$
$-5\pi/20$	$8.595661e-07$	$4.617298e-07$	$3.623414e-06$	$2.099887e-06$
$-3\pi/20$	$5.859313e-07$	$2.297674e-06$	$3.413060e-06$	$4.970394e-07$
$-\pi 20$	$1.415094e-07$	$7.296242e-06$	$1.160199e-06$	$3.748889e-06$
$\pi/20$	$3.360662e-07$	$1.329430e-05$	$2.347445e-06$	$1.015345e-05$
$3\pi/20$	$7.121270e-07$	$1.840305e-05$	$5.657620e-06$	$1.764220e-05$
$5\pi/20$	$8.722955e-07$	$2.048811e-05$	$7.168941e-06$	$2.487994e-05$
$7\pi/20$	$7.392216e-07$	$1.735549e-05$	$5.725458e-06$	$3.056752e-05$
$9\pi/20$	$3.010297e-07$	$6.695846e-06$	$1.966998e-06$	$3.368879e-05$

Table 5
The error norms of Example 2 for different spline methods for $h = \pi/20$ and $\Delta t = 0.0001$.

t	Error norm	Presented method	Method II	Method III	Method IV
1	L_∞	$1.028282e-06$	$3.769251e-05$	$2.511016e-05$	$7.295803e-05$
	L_2	$6.370047e-07$	$1.576174e-05$	$1.075739e-05$	$2.917490e-05$
2	L_∞	$5.570207e-07$	$9.847311e-06$	$1.898947e-06$	$1.371097e-05$
	L_2	$3.233114e-07$	$4.163493e-06$	$8.371396e-07$	$5.207501e-06$
3	L_∞	$1.428927e-07$	$1.922356e-06$	$7.112927e-07$	$1.941321e-06$
	L_2	$8.258020e-08$	$8.524094e-07$	$3.159984e-07$	$7.315347e-07$

The exact solution of (39) is given by

$$u(x,t) = \frac{1}{3}(2 + \tanh(t) - (1 - \tanh(t))\sin(x)).$$

The initial and boundary conditions can be obtained from the exact solution.

Using (20)–(24) and (39), the following nonlinear system of equations is obtained

$$u_t(\xi_j, t_{s+1}) = \big(1 - u(\xi_j, t_{s+1})\big)\big(u_{xx}(\xi_j, t_{s+1}) + 2u(\xi_j, t_{s+1})\big) - \big(u_x(\xi_j, t_{s+1})\big)^2, \tag{40}$$

where $j = -n-1, -n, \ldots, n+1$.

For each s, the values of C_n are calculated by solving the nonlinear system (40) and then the values of $u(\xi_j, t_{s+1})$ are determined by utilizing (22).

The maximum absolute errors at the final time $t = 1.5$ of the proposed method are compared with those of cubic B-Spline Quasi-Interpolation (Method II) [28,29], B-spline differences (method III) [31] and a cubic B-splines collocation method (Method IV) [32] in Table 4. Also, the error norms L_∞ and L_2 corresponding to the suggested method and the methods II, III and, IV are expressed in Table 5. In Table 6, the maximum absolute errors at the final time $t = 3$ for the fixed $h = \pi/10$ and different values of Δt are reported. The logarithm of absolute errors at the final time $t = 3$ of the proposed method and the methods II, III and, IV are depicted in Fig. 3. The exact solutions and absolute errors of the numerical method for (39) are illustrated in Fig. 4. In order to have a fair comparison, we assume that $h = \pi/20$ and $\Delta t = 0.0001$ in all methods.

Table 6
The maximum absolute errors of Example 2 for $t = 3$ and $h = \pi/10$.

Δt	Max. absolute error
1/2	$1.013505e-03$
1/4	$4.386564e-04$
1/8	$1.998553e-04$
1/16	$9.472917e-05$
1/32	$4.602310e-05$
1/64	$2.267089e-05$
1/128	$1.124972e-05$
1/256	$5.603761e-06$
1/512	$2.797277e-06$
1/1024	$1.398236e-06$
1/2048	$7.000375e-07$
1/4096	$3.629958e-07$

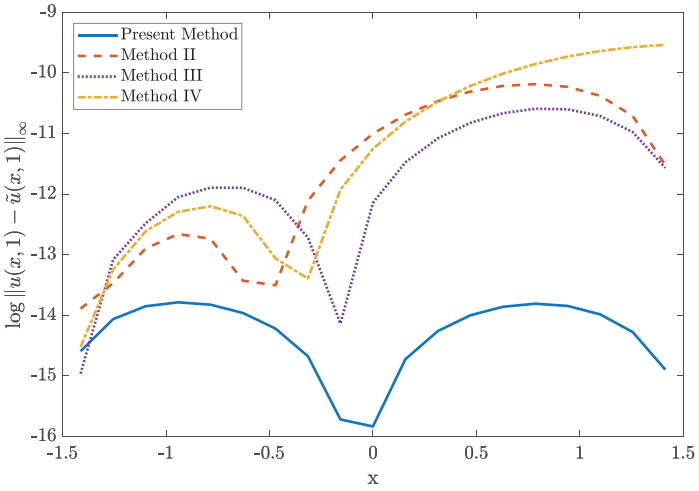

Fig. 3. The logarithms of absolute errors of Example 2 for different spline methods in $t = 1$ for $h = \pi/20$ and $\Delta t = 0.0001$.

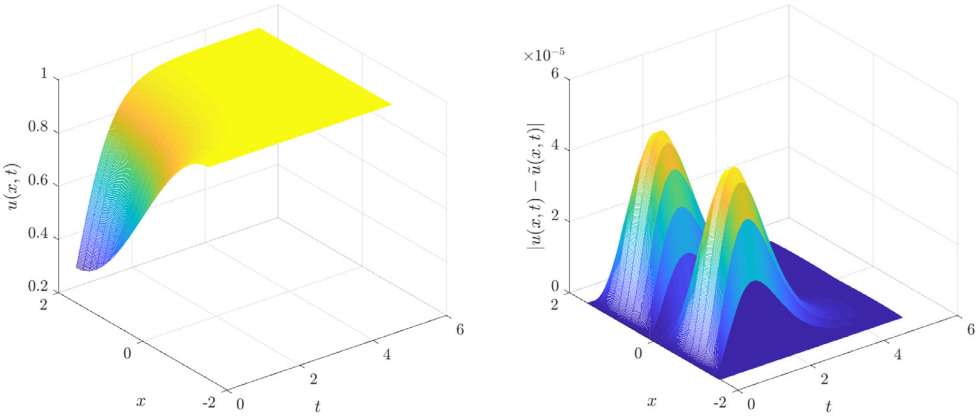

Fig. 4. The exact solutions (left) and the absolute errors (right) of Example 2 with $h = \pi/20$ and $\Delta t = 0.005$.

Example 3. As the final example, the following generalized Burgers–Fisher equation is considered [27]

$$u_t + \alpha u^\delta u_x - \varepsilon u_{xx} = \beta u(1 - u^\delta), \quad -1 < x < 1, \ t > 0. \tag{41}$$

The exact solution is given by

$$u(x,t) = \left(\frac{1}{2} + \frac{1}{2}\tanh(\sigma_1(x - \sigma_2 t))\right)^{\frac{1}{\delta}},$$

Table 7
Maximum absolute errors of Example 3 for different spline methods in $t = 4$ for $h = 0.05$, $\Delta t = 0.0001$.

x	Present method	Method II	Method III	Method IV
−0.9	$2.510289e-09$	$3.500592e-08$	$4.523877e-06$	$4.691982e-04$
−0.7	$7.927065e-09$	$1.041859e-07$	$1.498539e-05$	$4.604582e-04$
−0.5	$1.333691e-08$	$1.690881e-07$	$2.757963e-05$	$4.533378e-04$
−0.3	$1.811131e-08$	$2.263077e-07$	$4.263947e-05$	$4.482437e-04$
−0.1	$2.165096e-08$	$2.715832e-07$	$6.054250e-05$	$4.456459e-04$
0.1	$2.340849e-08$	$2.996171e-07$	$8.171590e-05$	$4.460873e-04$
0.3	$2.291644e-08$	$3.038728e-07$	$1.066416e-04$	$4.501930e-04$
0.5	$1.982312e-08$	$2.763384e-07$	$1.358619e-04$	$4.586822e-04$
0.7	$1.393666e-08$	$2.072547e-07$	$1.699852e-04$	$4.723796e-04$
0.9	$5.281705e-09$	$8.481648e-08$	$2.096925e-04$	$4.922300e-04$

Table 8
The error norms of Example 3 for different spline methods using $h = 0.05$, $\Delta t = 0.0001$.

t	Error norm	Present method	Method II	Method III	Method IV
1	L_∞	$3.454354e-07$	$1.915832e-06$	$4.688442e-03$	$3.244907e-03$
	L_2	$2.635589e-07$	$1.633663e-06$	$2.271386e-03$	$2.041180e-03$
2	L_∞	$5.256692e-08$	$2.033358e-06$	$2.245220e-03$	$2.223872e-03$
	L_2	$3.211588e-08$	$1.409211e-06$	$9.192477e-04$	$1.806935e-03$
3	L_∞	$4.567310e-08$	$9.192536e-07$	$7.671616e-04$	$1.139620e-03$
	L_2	$2.767084e-08$	$5.940740e-07$	$2.936482e-04$	$9.995790e-04$
4	L_∞	$2.351230e-08$	$3.054486e-07$	$2.318744e-04$	$5.070087e-04$
	L_2	$1.392544e-08$	$1.931589e-07$	$8.693930e-05$	$4.623098e-04$

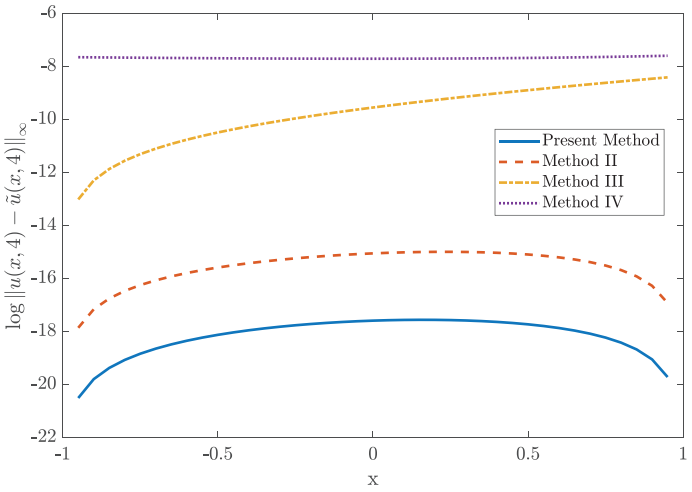

Fig. 5. The logarithms of absolute errors of Example 3 for different spline methods in $t = 4$ using $h = 0.05$ and $\Delta t = 0.0001$.

where

$$\sigma_1 = \frac{-\alpha\delta}{2(1+\delta)}, \quad \sigma_2 = \frac{\alpha}{1+\delta} + \frac{\beta(1+\delta)}{\alpha},$$

and $\alpha = 1$, $\beta = 1$, $\epsilon = 1$, $\delta = 1$. Furthermore, the initial and boundary conditions can be obtained from the exact solution.

Using (20)–(24) and (41) the following nonlinear system of equations will be obtained

$$u_t(\xi_j, t_{s+1}) + \alpha\big(u(\xi_j, t_{s+1})\big)^\delta u_x(\xi_j, t_{s+1}) - \epsilon u_{xx}(\xi_j, t_{s+1}) = \beta u(\xi_j, t_{s+1})\Big(1 - \big(u(\xi_j, t_{s+1})\big)^\delta\Big), \tag{42}$$

where $j = -n-1, -n, \ldots, n+1$.

For each s, the values of C_n are calculated by solving nonlinear system (42) and then the values of $u(\xi_j, t_{s+1})$ are obtained using (22).

The maximum absolute errors at the final time $t = 4$ of the proposed method are compared with those of cubic B-Spline Quasi-Interpolation (Method II) [28,29], B-spline differences (method III) [31] and a cubic B-splines collocation method (Method IV) [32] in Table 7. Moreover, the logarithm of absolute errors in final time $t = 4$ of the proposed method and the methods II, III, and IV are depicted in Fig. 5. The error norms L_∞ and L_2 of the presented scheme and the methods II, III, and IV are tabulated in Table 8. In

Table 9
The maximum absolute errors of Example 3 for $t = 4$ and $h = 0.05$.

Δt	Max. absolute error
1/2	$1.254373e-04$
1/4	$6.214032e-05$
1/8	$3.040334e-05$
1/16	$1.496881e-05$
1/32	$7.418337e-06$
1/64	$3.691718e-06$
1/128	$1.841384e-06$
1/256	$9.195592e-07$
1/512	$4.594947e-07$
1/1024	$2.296760e-07$
1/2048	$1.148201e-07$
1/4096	$5.740565e-08$

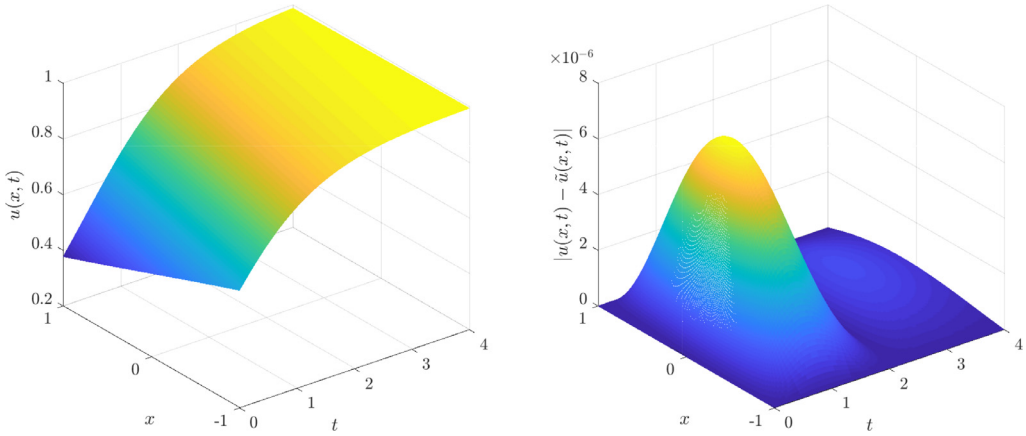

Fig. 6. The exact solutions (left) and the absolute errors (right) of Example 3 with $h = 0.02$ and $\Delta t = 0.002$.

Table 9, the maximum absolute errors at the final time $t = 4$ for the fixed $h = 0.05$ and different values of Δt are calculated. The exact solutions and absolute errors of the numerical method for (41) are demonstrated in Fig. 6. To present a fair comparison, $h = 0.05$ and $\Delta t = 0.0001$ have been supposed in all methods.

Conclusions

In this article, an orthogonal basis for the space of cubic spline functions were obtained by using cubic spline bases and an orthogonalization process. Next, a linear combination of the members of this base was used to approximate the functions, and the first-order and second-order integrals of this linear combination were obtained at the knot points. This approximation tool was used to numerically solution of a general case of the nonlinear parabolic partial differential equation. We started the approximations from the highest order of the derivative in the equation and with multiple integrations, at the knot points we reached a system of algebraic equations. This system becomes linear or nonlinear, depending on whether the initial problem related to it is linear or nonlinear, respectively. The linear can be solved by conventional methods and the trust-region-dogleg algorithm is used to solve the nonlinear case. Also, the convergence in the approximate scheme is analyzed. Next, one linear problem and two nonlinear problems were presented and the numerical results of the presented method were compared with other numerical methods that used splines. The numerical results indicated the accuracy and reliability of the presented method.

The main difference between the presented method with other methods that use splines to numerically solution of partial differential equations is that we use the linear combination of bases to approximate the highest order of the spatial and temporal derivative and then with integrating and placing the knot points, we arrive at a numerical solution for the equation. Another difference is the use of a special type of orthogonal cubic splines that have not been used before to solve these types of equations. These basis functions have interesting numerical properties and are numerically stable. We have examined some of their properties before, but according to the numerical results, it seems that more properties of these bases can be obtained in the future, which will be useful for solving various problems.

Ethics statements

NA.

Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

CRediT authorship contribution statement

Javad Alavi: Conceptualization, Methodology, Investigation, Software, Writing – original draft, Writing – review & editing. **Hossein Aminikhah:** Conceptualization, Methodology, Investigation, Supervision, Writing – review & editing.

Data availability

No data was used for the research described in the article.

Acknowledgement

Authors are very grateful to anonymous referees for their careful reading and valuable comments which led to the improvement of this paper.

References

[1] A.H. Khater, R.S. Temsah, Numerical solutions of some nonlinear evolution equations by Chebyshev spectral collocation methods, Int. J. Comput. Math. 84 (2007) 326–339.
[2] A.F. Bastani, M.V. Dastgerdi, A. Mighani, On multilevel RBF collocation to solve nonlinear PDEs arising from endogenous stochastic volatility models, Commun. Nonlinear Sci. Numer. Simul. 59 (2018) 88–104.
[3] X.Y. Wang, Exact and explicit solitary wave solutions for the generalized Fishers equation, Phys. Lett. A. 131 (1988) 277–291.
[4] W. Herman, M. Takaoka, Solitary wave solutions of nonlinear evolution and wave equations using a direct method and MACSYMA, J. Phys. A 23 (1990) 34–48.
[5] J. Ramírez, J.L. Romero, C. Muriel, Two new reductions methods for polynomial differential equations and applications to nonlinear PDEs, J. Comput. Appl. Math. 333 (2018) 36–50.
[6] F.M. Samuel, S.S. Motsa, A highly accurate trivariate spectral collocation method of solution for two-dimensional nonlinear initial-boundary value problems, Appl. Math. Comput. 360 (2019) 221–235.
[7] J.Y. Chang, C.C. Tsai, D.L. Young, Homotopy method of fundamental solutions for solving nonlinear heat conduction problems, Eng. Anal. Bound. Elem. 108 (2019) 179–191.
[8] H. Mi, L. Xu, Optimal investment with derivatives and pricing in an incomplete market, J. Comput. Appl. Math. 368 (2020). Article 112522, April
[9] A.J. Torija, D.P. Ruiz, A general procedure to generate models for urban environmental-noise pollution using feature selection and machine learning methods, Sci. Total Environ. 505 (2015) 680–693.
[10] I.A. Aziz, K.M.A. Manmi, R.K. Saeed, A. Dadvand, Modeling three dimensional gas bubble dynamics between two curved rigid plates using boundary integral method, Eng. Anal. Bound. Elem. 109 (2019) 19–31.
[11] R.M. Cherniha, New exact solutions of one nonlinear equation in mathematical biology and their properties, Ukrainian Math. J. 53 (10) (2018) 1712–1727.
[12] R. Cherniha, V. Dutka, Exact and numerical solutions of the generalized Fisher equation, Rep. Math. Phys. 47 (3) (2001) 393–411.
[13] R.C. Mittal, R. Jiwari, A higher-order numerical scheme for some nonlinear differential equations: models in biology, Int. J. Comput. Methods Eng. Sci. Mech. 12 (3) (2011) 134–140.
[14] R.K. Mohanty, An implicit high accuracy variable mesh scheme for 1-D non-linear singular parabolic partial differential equations, Appl. Math. Comput. 186 (1) (2007) 219–229.
[15] K.R. Raslan, Collocation method using quartic B-spline for the equal width (EW) equation, Appl. Math. Comput. 168 (2) (2005) 795–805.
[16] M. Dehghan, Weighted finite difference techniques for the one-dimensional advection-diffusion equation, Appl. Math. Comput. 147 (2004) 307–319.
[17] R.C. Mittal, R.K. Jain, Redefined cubic B-splines collocation method for solving convection-diffusion equations, Appl. Math. Model. 36 (11) (2012) 5555–5573.
[18] A.M. Elsherbeny, R.M.I. El-hassani, H. El-badry, M.I. Abdallah, Solving 2D-Poisson equation using modified cubic B-spline differential quadrature method, Ain Shams Eng. J. 9 (2018) 2879–2885.
[19] R.C. Mittal, R.K. Jain, Cubic B-splines collocation method for solving nonlinear parabolic partial differential equations with Neumann boundary conditions, Commun. Nonlinear Sci. Numer. Simul. 17 (12) (2012) 4616–4625.
[20] M. Tamsir, N. Dhiman, V.K. Srivastava, Cubic trigonometric B-spline differential quadrature method for the numerical treatment of Fisher's reaction-diffusion equations, Alex. Eng. J. 57 (3) (2018) 2019–2026.
[21] R. Mohammadi, Spline solution of the generalized Burgers'–Fisher equation, Appl. Anal. Int. J. 91 (12) (2012) 2189–2215.
[22] J. Goh, A.A. Majid, A.I.M. Ismail, Numerical method using cubic B-spline for the heat and wave equation, Comput. Math. Appl. 62 (12) (2011) 4492–4498.
[23] J.C. Mason, G. Rodriguez, S. Seatzu, Orthogonal splines based on B-splines with applications to least squares, smoothing and regularisation problems, Numer. Algorithms 5 (1993) 25–40.
[24] J. Alavi, H. Aminikhah, Orthogonal cubic spline basis and its applications to a partial integro-differential equation with a weakly singular kernel, Comput. Appl. Math. 40 (2021). Article number: 55
[25] G. Hariharan, K. Kannan, K.R. Sharma, Haar wavelet method for solving Fishers equation, Appl. Math. Comput. 211 (2009) 284–292.
[26] R.H. Byrd, R.B. Schnabel, G.A. Shultz, A trust region algorithm for nonlinearly constrained optimization, SIAM J. Numer. Anal. 24 (5) (1987) 1152–1170.
[27] R.C. Mittal, A. Tripathi, Numerical solutions of generalized Burgers–Fisher and generalized Burgers–Huxley equations using collocation of cubic B-splines, Int. J. Comput. Math. 92 (5) (2015) 1053–1077.
[28] H. Aminikhah, J. Alavi, Numerical solution of convection-diffusion equation using cubic B-spline quasi-interpolation, Thai J. Math. 14 (3) (2016) 599–613.
[29] C.G. Zhu, W.S. Kang, Numerical solution of Burgers–Fisher equation by cubic B-spline quasi-interpolation, Appl. Math. Comput. 216 (2010) 2679–2686.
[30] J. Zhang, J. Zheng, Q. Gao, Numerical solution of the Degasperis–Procesi equation by the cubic B-spline quasi-interpolation method, Appl. Math. Comput. 324 (2018) 218–227.
[31] H. Aminikhah, J. Alavi, An efficient B-spline difference method for solving the system of nonlinear parabolic PDEs, SeMA J. 75 (2) (2018) 335–348.
[32] R.C. Mittal, R.K. Jain, Cubic B-splines collocation method for solving nonlinear parabolic partial differential equations with Neumann boundary conditions, Commun. Nonlinear Sci. Numer. Simul. 17 (12) (2012) 4616–4625.