Mathematical Sciences (2020) 14:409–423
https://doi.org/10.1007/s40096-020-00352-7

ORIGINAL RESEARCH

The numerical study of advection–diffusion equations by the fourth-order cubic B-spline collocation method

R. C. Mittal[1,2] · **Rajni Rohila**[1,2]

Received: 6 August 2018 / Accepted: 8 September 2020 / Published online: 21 September 2020

Abstract
A fourth-order numerical method based on cubic B-spline functions has been proposed to solve a class of advection–diffusion equations. The proposed method has several advantageous features such as high accuracy and fast results with very small CPU time. We have applied the Crank–Nicolson method to solve the advection–diffusion equation. The stability analysis is performed, and the method is shown to be unconditionally stable. Error analysis is carried out to show that the proposed method has fourth-order convergence. The efficiency of the proposed B-spline method has been checked by applying on ten important advection–diffusion problems of three types, having Dirichlet, Neumann and periodic boundary conditions. Considered examples prove the mentioned advantages of the method. The computed results are also compared with those available in the literature, and it is found that our method is giving better results.

Keywords Advection–diffusion equation · Crank–Nicolson method · B-spline functions · Collocation method · Gauss elimination method

Introduction

Consider the one-dimensional convection–diffusion equation

$$\frac{\partial u}{\partial t}+\epsilon\frac{\partial u}{\partial x}=\gamma\frac{\partial^2 u}{\partial x^2}, a<x<b \text{ and } 0<t\leq T, \tag{1.1}$$

where ϵ and γ are convection and diffusion coefficients, respectively. The initial condition is defined by

$$u(x,0)=\psi(x). \tag{1.2}$$

The boundary conditions are as follows:

(a) The Dirichlet boundary conditions are given by

$$u(a,t)=f_1(t),\ \ u(b,t)=f_2(t), \tag{1.3}$$

(b) The Neumann boundary conditions are defined by

$$\frac{\partial u}{\partial x}(a,t)=g_1(t),\ \ \frac{\partial u}{\partial x}(b,t)=g_2(t), \tag{1.4}$$

and

(c) The periodic boundary conditions can be written as

$$u(a,t)=u(b,t). \tag{1.5}$$

If the boundary conditions are periodic, then the total heat(mass) is conserved, i.e.,

$$m(t)=\int_a^b u(x,t)\mathrm{d}t = \mathrm{k}\,(\text{constant}). \tag{1.6}$$

The advection–diffusion equation represents many problems of physics, chemistry and biology. It describes physical phenomena in which energy is transformed during two processes, namely advection and diffusion. It presents advection and diffusion of quantities such as mass, energy, heat, etc. The advection–diffusion equation can be used to depict water transfer in soils [1], transportation of pollutants in the atmosphere [2], diffusion of pollutants in rivers and stream [3], transfer of energy [4] and fluid [5] through the medium. Different numerical methods have been given to solve

✉ Rajni Rohila
rajnirohila89@gmail.com

R. C. Mittal
mittalrc@gmail.com

[1] Department of Mathematics, Indian Institute of Technology, Roorkee, Uttarakhand 247667, India

[2] Department of Applied Sciences, The NorthCap University, Gurugram, Haryana, India

advection–diffusion equations due to wide applicability of this equation in different areas. Rizwan and Uddin presented second-order space and time nodal method [6]. Different numerical methods have been given by Dehghan for finding numerical solutions for this equation. Most of his methods are based on finite difference implicit or explicit schemes and are used to solve one- and three-dimensional advection–diffusion equations. Dehghan presented the weighted finite difference scheme [5] and high-order compact scheme [7] for the numerical study of the advection–diffusion equation. Other methods given by Dehghan can be seen in [8–11]. The advection–diffusion equation was studied numerically by Salkuyeh [12]. Karahan [13, 14] developed finite difference methods to solve this equation numerically. Zhang and Ding [15] developed a new difference scheme to get numerical solutions. Several spline-based methods have been given to solve (1.1) such as spline functions [16], exponential cubic spline [17], cubic B-splines [18, 19], cubic B-spline differential quadrature method [20], redefined splines [4], B-spline finite element [21, 22], the Taylor–Galerkin method [22] and the Taylor–Galerkin finite element method [23]. In the classical method of B-spline collocation, we just replace the unknown function and its derivatives by spline approximations for the function and its derivatives, respectively, and apply the Crank–Nicolson scheme or Runge–Kutta methods to get solutions at the grid points. The B-spline approximations of the unknown function u and its first derivative are of the fourth order, and those for the double derivative are of the second order.

In this work, we have introduced two different end conditions for the double derivatives to develop a fourth-order method having a higher order of convergence. In short, we have improved approximation for the double derivative by using different end conditions which resulted in very good numerical solutions. The fourth-order approximation for $u''(x)$ can be obtained by taking one more term in the Taylor series expansion. A collocation method is applied to find the solution. One of the advantages of the method is that the solution can be obtained at any point of the domain.

The rest of the paper is organized as follows. In "Cubic B-spline functions" section, we have discussed the properties of spline functions. In "Implementation of the method" section, the implementation of the method to (1.1) has been explained. In "Error analysis" section, the convergence has been checked. " Stability analysis" section contains the stability analysis followed by "Numerical experiments" section which contains test problems taken from the literature. A conclusion has been added in "Conclusion" section to briefly summarize the outcomes of the research work.

Cubic B-spline functions

A spline is a function that is piecewise defined by polynomial functions and possesses a high degree of smoothness at the places where polynomial pieces connect. A B-spline is a spline function that has minimal support with respect to given degree smoothness and domain partition. The cubic B-spline functions are defined as follows

$$B_j(x) = \frac{1}{h^3}\begin{cases} (x-x_{j-2})^3 & x \in [x_{j-2}, x_{j-1}), \\ (x-x_{j-2})^3 - 4(x-x_{j-1})^3, & x \in [x_{j-1}, x_j), \\ (x_{j+2}-x)^3 - 4(x_{j+1}-x)^3, & x \in [x_j, x_{j+1}), \\ (x_{j+2}-x)^3 & x \in [x_{j+1}, x_{j+2}), \\ 0 & \text{otherwise} \end{cases} \tag{2.1}$$

where $\{B_{-1}(x), B_0(x), B_1(x), B_2(x), \ldots, B_N(x), B_{N+1}(x)\}$ forms a basis over the considered interval.

Consider a uniform partition of the problem domain $a \le x \le b$ by the knots x_i, $i = 0, 1, 2 \ldots N$ such that $x_i - x_{i-1} = h$ is the length of each interval.

In the cubic B-spline collocation method, we approximate exact solution $u(x, t)$ by $S(x, t)$ in the form [24]

$$S(x,t) = \sum_{j=-1}^{N+1} c_j(t) B_j(x), \tag{2.2}$$

where $c_j(t)$'s are time-dependent quantities which we determine from the boundary conditions and collocation from the differential equation. We assume that the approximate solution $S(x, t)$ satisfies the following interpolatory and boundary conditions

$$S(x_j, t) = u(x_j, t), \ \ 0 \le j \le N, \tag{2.3}$$

$$S''(x_j, t) = u''(x_j, t) - \frac{1}{12}h^2 u^{(4)}(x_j, t), \ \ j = 0, \ \ N. \tag{2.4}$$

If $u(x, t)$ is sufficiently smooth and $S(x, t)$ is the unique cubic spline interpolant satisfying the above end conditions, then we have from [25]

$$\begin{aligned} S'(x_j, t) &= u'(x_j, t) + O(h^4), \ \ 0 \le j \le N \\ S''(x_j, t) &= u''(x_j, t) - \frac{1}{12}h^2 u^{(4)}(x_j, t) + O(h^4), \ \ 0 \le j \le N. \end{aligned} \tag{2.5}$$

The approximate values $S(x, t)$ and their first-order derivatives at the knots are defined by using finite difference and Taylor expansions as follows

$$\begin{aligned} u^{(4)}(x_j, t) = {} & \frac{S''(x_0, t) - 5S''(x_1, t) + 4S''(x_2, t) - S''(x_3, t)}{h^2} \\ & + O(h^2), \ \ j = 0, \end{aligned} \tag{2.6}$$

$$u^{(4)}(x_j,t)=\frac{S''(x_{j-1},t)-2S''(x_j,t)+S''(x_{j+1},t)}{h^2}+O(h^2),\quad 1\le j\le N-1, \tag{2.7}$$

$$u^{(4)}(x_j,t)=\frac{S''(x_N,t)-5S''(x_{N-1},t)+4S''(x_{N-2},t)-S''(x_{N-3},t)}{h^2}+O(h^2),\quad j=N. \tag{2.8}$$

Using (2.7)–(2.9) in (2.5)–(2.6), we get

$$u'(x_j,t)=S'(x_j,t)+O(h^4),\quad 0\le j\le N, \tag{2.9}$$

$$u''(x_0,t)=\frac{14S''(x_0,t)-5S''(x_1,t)+4S''(x_2,t)-S''(x_3,t)}{12}+O(h^4),\quad j=0, \tag{2.10}$$

$$u''(x_j,t)=\frac{S''(x_{j-1},t)+10S''(x_j,t)+S''(x_{j+1},t)}{12}+O(h^4),\quad 1\le j\le N-1, \tag{2.11}$$

$$u''(x_N,t)=\frac{14S''(x_N,t)-5S''(x_{N-1},t)+4S''(x_{N-2},t)-S''(x_{N-3},t)}{12}+O(h^4),\quad j=N. \tag{2.12}$$

(2.11) to (2.13) can be written as

$$u''(x_0,t)=\frac{14\sum_{j=-1}^{N+1}c_j(t)B_j''(x_0)-5\sum_{j=-1}^{N+1}c_j(t)B_j''(x_1)+4\sum_{j=-1}^{N+1}c_j(t)B_j''(x_2)-\sum_{j=-1}^{N+1}c_j(t)B_j''(x_3)}{12}, \tag{2.13}$$

$$u''(x_j,t)=\frac{\sum_{j=-1}^{N+1}c_j(t)B_j''(x_{j-1})+10\sum_{j=-1}^{N+1}c_j(t)B_j''(x_j)+\sum_{j=-1}^{N+1}c_j(t)B_j''(x_{j+1})}{12},\quad 1\le j\le N-1, \tag{2.14}$$

$$u''(x_N,t)=\frac{14\sum_{j=-1}^{N+1}c_j(t)B_j''(x_N)-5\sum_{j=-1}^{N+1}c_j(t)B_j''(x_{N-1})+4\sum_{j=-1}^{N+1}c_j(t)B_j''(x_{N-2})-\sum_{j=-1}^{N+1}c_j(t)B_j''(x_{N-3})}{12},\quad j=N. \tag{2.15}$$

Using (2.1)–(2.2), (2.14)–(2.16) can be further simplified to get

$$u''(x_0,t)=\frac{14c_{-1}-33c_0+28c_1-14c_2+6c_3-c_4}{2h^2}, \tag{2.16}$$

$$u''(x_j,t)=\frac{c_{j-2}+8c_{j-1}-18c_j+8c_{j+1}+c_{j+2}}{2h^2}, \tag{2.17}$$

$$u''(x_N,t)=\frac{14c_{N+1}-33c_N+28c_{N-1}-14c_{N-2}+6c_{N-3}-c_{N-4}}{2h^2}. \tag{2.18}$$

We will use these formulae in the next section.

Implementation of the method

Applying the Crank–Nicolson scheme to Eq. (1.1), it can be written as

$$\frac{u^{n+1}-u^n}{\Delta t}+\epsilon\frac{u_x^{n+1}+u_x^n}{2}=\gamma\frac{u_{xx}^{n+1}+u_{xx}^n}{2}. \tag{3.1}$$

Separating the terms of advanced and initial level, we get

$$u^{n+1}+\frac{\epsilon\Delta t u_x^{n+1}}{2}-\frac{\gamma\Delta t u_{xx}^{n+1}}{2}=u^n-\frac{\epsilon\Delta t u_x^n}{2}+\frac{\gamma\Delta t u_{xx}^n}{2}. \tag{3.2}$$

For $j=0$,

$$
\begin{aligned}
&(c_{-1}^{n+1}+4c_0^{n+1}+c_1^{n+1})+3\frac{\epsilon\Delta t}{2h}(c_1^{n+1}-c_{-1}^{n+1})\\
&\quad-\frac{\gamma\Delta t}{4h^2}(14c_{-1}^{n+1}-33c_0^{n+1}+28c_1^{n+1}-14c_2^{n+1}\\
&\quad+6c_3^{n+1}-c_4^{n+1}).\\
&=(c_{-1}^{n}+4c_0^{n}+c_1^{n})-3\frac{\epsilon\Delta t}{2h}(c_1^{n}-c_{-1}^{n})\\
&\quad+\frac{\gamma\Delta t}{4h^2}(14c_{-1}^{n}-33c_0^{n}+28c_1^{n}-14c_2^{n}+6c_3^{n}-c_4^{n}).
\end{aligned}
\tag{3.3}
$$

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

$$
\begin{aligned}
&(c_{j-1}^{n+1}+4c_j^{n+1}+c_{j+1}^{n+1})+3\frac{\epsilon\Delta t}{2h}(c_{j+1}^{n+1}-c_{j-1}^{n+1})\\
&\quad-\frac{\gamma\Delta t}{4h^2}(c_{j-2}^{n+1}+8c_{j-1}^{n+1}-18^{n+1}c_j+8c_{j+1}^{n+1}+c_{j+2}^{n+1})\\
&=(c_{j-1}^{n}+4c_j^{n}+c_{j+1}^{n})-3\frac{\epsilon\Delta t}{2h}(c_{j+1}^{n}-c_{j-1}^{n})\\
&\quad+\frac{\gamma\Delta t}{4h^2}(c_{j-2}^{n}+8c_{j-1}^{n}-18^{n}c_j+8c_{j+1}^{n}+c_{j+2}^{n}),
\end{aligned}
\tag{3.4}
$$

for $j = N$, we have

$$
\begin{aligned}
&(c_{N-1}^{n+1}+4c_N^{n+1}+c_{N+1}^{n+1})+3\frac{\epsilon\Delta t}{2h}(c_{N+1}^{n+1}-c_{N-1}^{n+1})\\
&\quad-\frac{\gamma\Delta t}{4h^2}(14c_{N+1}^{n+1}-33c_N^{n+1}+28c_{N-1}^{n+1}\\
&\quad-14c_{N-2}^{n+1}\\
&\quad+6c_{N-3}^{n+1}-c_{N-4}^{n+1})\\
&=(c_{N-1}^{n}+4c_N^{n}+c_{N+1}^{n})-3\frac{\epsilon\Delta t}{2h}(c_{N+1}^{n}-c_{N-1}^{n})\\
&\quad+\frac{\gamma\Delta t}{4h^2}(14c_{N+1}^{n}-33c_N^{n}+28c_{N-1}^{n}-14c_{N-2}^{n}\\
&\quad+6c_{N-3}^{n}-c_{N-4}^{n}).
\end{aligned}
\tag{3.5}
$$

Rearranging the terms for $j = 0$, we may write

$$
\begin{aligned}
&s_1c_{-1}^{n+1}+s_2c_0^{n+1}+s_3c_1^{n+1}+s_4c_2^{n+1}+s_5c_3^{n+1}+s_6c_4^{n+1}\\
&\quad=v_1c_{-1}^{n}+v_2c_0^{n}+v_3c_1^{n}+v_4c_2^{n}+v_5c_3^{n}+v_6c_4^{n},
\end{aligned}
$$

where

$$
\begin{aligned}
s_1&=1-\frac{3\epsilon\Delta t}{2h}-\frac{7\gamma\Delta t}{2h^2},\\
s_2&=4+\frac{33\gamma\Delta t}{4h^2},\\
s_3&=1+\frac{3\epsilon\Delta t}{2h}-\frac{7\gamma\Delta t}{h^2},\\
s_4&=\frac{7\gamma\Delta t}{2h^2},\\
s_5&=-\frac{3\gamma\Delta t}{2h^2},\\
s_6&=\frac{\gamma\Delta t}{4h^2},
\end{aligned}
$$

and

$$
\begin{aligned}
v_1&=1+\frac{3\epsilon\Delta t}{2h}+\frac{7\gamma\Delta t}{2h^2},\\
v_2&=4-\frac{33\gamma\Delta t}{4h^2},\\
v_3&=1-\frac{3\epsilon\Delta t}{2h}+\frac{7\gamma\Delta t}{h^2},\\
v_4&=-\frac{7\gamma\Delta t}{2h^2},\\
v_5&=\frac{3\gamma\Delta t}{2h^2},\\
v_6&=-\frac{\gamma\Delta t}{4h^2},
\end{aligned}
$$

and for $j = 1, 2 \ldots N-1$, we get

$$
\begin{aligned}
&-x_1c_{j-2}^{n+1}+x_2c_{j-1}^{n+1}+x_3c_j^{n+1}+x_4c_{j+1}^{n+1}-x_1c_{j+2}^{n+1}\\
&\quad=x_1c_{j-2}^{n}+x_5c_{j-1}^{n}+x_6c_j^{n}+x_7c_{j+1}^{n}+x_1c_{j+2}^{n},
\end{aligned}
\tag{3.6}
$$

where $x_1 = \frac{\gamma\Delta t}{4h^2}$,

$$
\begin{aligned}
x_2&=1-\frac{3\epsilon\Delta t}{2h}-\frac{2\gamma\Delta t}{h^2},\\
x_3&=4+\frac{9\gamma\Delta t}{2h^2},\\
x_4&=1+\frac{3\epsilon\Delta t}{2h}-\frac{2\gamma\Delta t}{h^2},\\
x_5&=1+\frac{3\epsilon\Delta t}{2h}-\frac{2\gamma\Delta t}{h^2},\\
x_6&=4-\frac{9\gamma\Delta t}{2h^2},\\
x_7&=1+\frac{3\epsilon\Delta t}{2h}+\frac{2\gamma\Delta t}{h^2}.
\end{aligned}
$$

for $j = N$,

$$
\begin{aligned}
&t_1c_{N-5}^{n+1}+t_2c_{N-4}^{n+1}+t_3c_{N-3}^{n+1}+t_4c_{N-2}^{n+1}+t_5c_{N-1}^{n+1}+t_6c_N^{n+1}\\
&\quad=e_1c_{N-5}^{n}+e_2c_{N-4}^{n}+e_3c_{N-3}^{n}+e_4c_{N-2}^{n}+e_5c_{N-1}^{n}+e_6c_N^{n}
\end{aligned}
$$

where

$$
\begin{aligned}
t_1&=\frac{\gamma\Delta t}{4h^2},\\
t_2&=-\frac{3\gamma\Delta t}{2h^2},\\
t_3&=\frac{7\gamma\Delta t}{2h^2}\\
t_4&=1-\frac{3\epsilon\Delta t}{2h}-\frac{7\gamma\Delta t}{h^2},\\
t_5&=4+\frac{33\gamma\Delta t}{4h^2},\\
t_6&=1+\frac{3\epsilon\Delta t}{2h}-\frac{7\gamma\Delta t}{2h^2},
\end{aligned}
$$

and

$$e_1 = -\frac{\gamma\Delta t}{4h^2},$$
$$e_2 = \frac{3\gamma\Delta t}{2h^2},$$
$$e_3 = -\frac{7\gamma\Delta t}{2h^2},$$
$$e_4 = 1 + \frac{3\epsilon\Delta t}{2h} + \frac{7\gamma\Delta t}{h^2},$$
$$e_5 = 4 - \frac{33\gamma\Delta t}{4h^2},$$
$$e_6 = 1 - \frac{3\epsilon\Delta t}{2h} + \frac{7\gamma\Delta t}{2h^2}.$$

Thus, we get the following system

$$Ac^{n+1} = Bc^n, \tag{3.7}$$

where $c = [c_{-1}, c_0, c_1, c_2 \ldots c_N, c_{N+1}]^T$

$$A = \begin{pmatrix} s_1 & s_2 & s_3 & s_4 & s_5 & s_6 & & & \\ -x_1 & x_2 & x_3 & x_4 & -x_1 & & & & \\ & -x_1 & x_2 & x_3 & x_4 & -x_1 & & & \\ .. & .. & .. & .. & .. & .. & .. & & \\ & & & & -x_1 & x_2 & x_3 & x_4 & -x_1 \\ & & & t_1 & t_2 & t_3 & t_4 & t_5 & t_6 \end{pmatrix},$$

and $B = \begin{pmatrix} v_1 & v_2 & v_3 & v_4 & v_5 & v_6 & & & \\ x_1 & x_5 & x_6 & x_7 & x_1 & & & & \\ & x_1 & x_5 & x_6 & x_7 & x_1 & & & \\ .. & .. & .. & .. & .. & .. & .. & & \\ & & & & x_1 & x_5 & x_6 & x_7 & x_1 \\ & & & e_1 & e_2 & e_3 & e_4 & e_5 & e_6 \end{pmatrix}.$

We have $N+1$ equations in $N+3$ unknowns. It may be observed that c_{-1} and c_{N+1} can be eliminated with the help of the Dirichlet or Neumann boundary conditions to get $N+1$ equations in $N+1$ unknowns. In the case of the periodic boundary condition, we solve for $c_0 \ldots c_{N-1}$ only. The system obtained on eliminating c_{-1} and c_{N+1} can be solved at any desired time level starting with the initial vector $[c_0^{(0)}, c_1^{(0)}, c_2^{(0)} \ldots c_{N-1}^{(0)}, c_N^{(0)}]^T$. The initial vector can be determined by using the B-spline approximation of the initial condition of the problem.

Error analysis

In this section, we have found an error bound for the presented method. From (2.1), we get the following expressions

$$S(x_i) = c_{i-1} + 4c_i + c_{i+1}, \tag{4.1}$$

$$S'(x_i) = \frac{3}{h}(c_{i+1} - c_{i-1}), \tag{4.2}$$

$$S'(x_i) = \frac{6}{h^2}(c_{i-1} - 2c_i + c_{i+1}), \tag{4.3}$$

$$[S'(x_{i-1}) + 4S'(x_i) + S'(x_{i+1})] = \frac{3}{h}[u(x_{i+1}) - u(x_{i-1})], \tag{4.4}$$

$$\begin{aligned} h^2S''(x_i) &= 6[u(x_{i+1}) - u(x_i)] \\ &\quad - 2h[2u'(x_i) + u'(x_{i+1})]. \end{aligned} \tag{4.5}$$

Then, as given in [26], we can use the operator notation $E(S(x_i)) = S(x_{i+1})$ and $E = e^{hD}$ where $D \equiv d/dx$. We obtain the following expressions

$$e^{hD} + e^{-hD} = 2\left(1 + \frac{h^2D^2}{2!} + \frac{h^4D^4}{4!} + \frac{h^6D^6}{6!} + \cdots\right) \tag{4.6}$$

$$e^{hD} - e^{-hD} = 2\left(hD + \frac{h^3D^3}{3!} + \frac{h^5D^5}{5!} + \frac{h^7D^7}{7!} + \cdots\right) \tag{4.7}$$

Using the above relations in (4.4) and (4.5), we obtain

$$h[E^{-1} + 4 + E]S'(x_i) = \frac{3}{h}(E - E^{-1})u(x_i) \tag{4.8}$$

$$h^2S''(x_i) = 6[E-1]u(x_i) - 2h(2+E)u'(x_i) \tag{4.9}$$

Now, using (4.6)–(4.7) in (4.8), we obtain

$$\begin{aligned} &\left[4 + 2\left(1 + \frac{h^2D^2}{2!} + \frac{h^4D^4}{4!} + \frac{h^6D^6}{6!} + \cdots\right)\right]S'(x_i) \\ &\quad = \frac{6}{h}\left[hD + \frac{h^3D^3}{3!} + \frac{h^5D^5}{5!} + \frac{h^7D^7}{7!} + \cdots\right]u(x_i) \end{aligned} \tag{4.10}$$

which can be simplified to get

$$\begin{aligned} S'(x_i) &= \frac{6}{h}\left[hD + \frac{h^3D^3}{3!} + \frac{h^5D^5}{5!} + \frac{h^7D^7}{7!} + \cdots\right] \\ &\quad \left[4 + 2\left(1 + \frac{h^2D^2}{2!} + \frac{h^4D^4}{4!} + \frac{h^6D^6}{6!} + \cdots\right)\right]^{-1}u(x_i) \end{aligned} \tag{4.11}$$

We further simplify the above to get

$$S'(x_i) = \left(D - \frac{h^4D^5}{180} + \cdots\right)u(x_i) \tag{4.12}$$

$$S'(x_i) = \left(u'(x_i) - \frac{h^4u^{(5)}(x_i)}{180} + \cdots\right) + O(h^6) \tag{4.13}$$

From (4.9) and (4.13), we obtain

$$h^2S''(x_i) = 6[e^{hD} - 1]u(x_i) - 2h[2 + e^{hD}]\Big[u'(x_i) - \frac{1}{180}h^4u^{(5)}(x_i) + \cdots\Big] \tag{4.14}$$

Expanding e^{hD} and rearranging the terms of the above equation, we obtain

$$S''(x_i) = \left(u''(x_i) - \frac{h^2u^{(4)}(x_i)}{12} + \frac{h^4u^{(6)}(x_i)}{360}...\right) + O(h^6) \tag{4.15}$$

Error analysis of convection–diffusion equation

As discussed above, we have applied the following approximations

$$S'(x_i) = u'(x_i) - \frac{1}{180}h^4u^{(5)}(x_i) + O(h^6) \tag{4.16}$$

$$S''(x_i) = u''(x_i) - \frac{h^2u^{(4)}(x_i)}{12} + \frac{1}{360}h^4u^{(6)}(x_i) + O(h^6) \tag{4.17}$$

Truncation error associated with the time discretization of convection–diffusion equation can be found by the typical finite difference formulation. Consider

$$u_t = f(u, u_x, u_{xx}) \tag{4.18}$$

We discretize the time derivative by finite difference to get

$$\frac{u_i^{n+1} - u_i^n}{\Delta t} = \frac{f^n + f^{n+1}}{2} \tag{4.19}$$

We apply the Taylor series expansion

$$(u_t)_i = f_i + \frac{1}{12}\Delta t^2 f_{tt} + \cdots \tag{4.20}$$

The truncation error of convection–diffusion equation is

$$e_i = (u_t)_i + \epsilon(u_x)_i - \gamma(u_{xx})_i \tag{4.21}$$

Substituting the values of errors introduced by space and time approximations, the truncation error of convection–diffusion equation may be given by

$$e_i = \left(\frac{1}{12}\right)\Delta t^2 f_{tt} - \epsilon\frac{1}{180}h^4u_i^{(5)} - \gamma\frac{1}{360}h^4u_i^{(6)} + O(\Delta t^3, h^6) \tag{4.22}$$

Hence, the final truncation error is

$$e_i = O(\Delta t^2) + O(h^4) \tag{4.23}$$

Hence, the spline approximation used in the proposed method with uniformly distributed interior points is $O(\Delta t^2 + h^4)$accurate.

Stability analysis

To check the stability of the scheme derived by the Crank–Nicolson and the fourth-order B-spline method, we have applied the Fourier-series (von Neumann) method. As discussed in "Cubic B-spline functions" section, applying the Crank–Nicolson on (1.1), we obtain the following difference scheme

$$\begin{aligned} &- x_1c_{j-2}^{n+1} + x_2c_{j-1}^{n+1} + x_3c_j^{n+1} + x_4c_{j+1}^{n+1} - x_1c_{j+2}^{n+1} \\ &= x_1c_{j-2}^n + x_5c_{j-1}^n + x_6c_j^n + x_7c_{j+1}^n + x_1c_{j+2}^n \end{aligned} \tag{5.1}$$

where x_i's have values as given in the previous section. Substituting $c_j^n = A\xi^n exp(ij\phi h)$, where $i = \sqrt{-1}$, A is amplitude, h is step length and ϕ is mode number, we get

$$\xi = \frac{x_1e^{-2\phi ih} + x_5e^{-\phi ih} + x_6 + x_7e^{\phi ih} + x_1e^{2\phi ih}}{-x_1e^{-2\phi ih} + x_2e^{-\phi ih} + x_3 + x_4e^{\phi ih} - x_1e^{2\phi ih}} \tag{5.2}$$

We further simplify (5.2) by substituting values of x_i's. We thus get the following expression

$$\xi = \frac{4h^2(\cos(\phi h) + 2) - \gamma\Delta t(9 - 8\cos(\phi h) - \cos(2\phi h)) - 6i\epsilon h\Delta t\sin(\phi h)}{4h^2(\cos(\phi h) + 2) + \gamma\Delta t(9 - 8\cos(\phi h) - \cos(2\phi h)) + 6i\epsilon h\Delta t\sin(\phi h)} \tag{5.3}$$

which may be written as

$$\xi = \frac{X_1 - iY}{X_2 + iY} \tag{5.4}$$

where $X_1 = 4h^2(\cos(\phi h) + 2) - \gamma\Delta t(9 - 8\cos(\phi h) - \cos(2\phi h))$

$X_2 = 4h^2(\cos(\phi h) + 2) + \gamma\Delta t(9 - 8\cos(\phi h) - \cos(2\phi h))$

and $Y = 6\epsilon h\Delta t\sin(\phi h)$

which gives $|\xi|^2 = \frac{X_1^2+Y^2}{X_2^2+Y^2}$

For stability, we must have $|\xi| \le 1$.

That is, $\frac{X_1^2+Y^2}{X_2^2+Y^2} \le 1$

Hence, for stability, we need to show that $X_1^2 \le X_2^2$

We note that $X_1^2 - X_2^2 = (X_1 + X_2)(X_1 - X_2) = (8h^2(\cos(\phi h) + 2))\big[-2\gamma\Delta t(9 - 8\cos(\phi h) - \cos(2\phi h))\big] \le 0$

Hence, the difference scheme is stable for any value of Δt and any value of Δx. That is, the difference scheme is unconditionally stable.

Numerical experiments

We apply the cubic B-spline based method to 11 problems taken from the literature which have been categorized into problems having Dirichlet, Neumann and periodic boundary conditions. The accuracy of the method has been calculated by using maximum absolute error norms given by

$$L_\infty = \|u^{\text{exact}} - U^N\|_\infty = \max|u_i^{\text{exact}} - u_i^N| \tag{6.1}$$

where U^N stands for numerical solution. u_i^{exact} and U_i^N represent exact solution and numerical solution at the knot x_i, respectively. The rate of convergence for the method has been calculated using the following formula

$$r = \frac{\ln(E(N_2)/E(N_1))}{\ln(N_1/N_2)} \tag{6.2}$$

where $E(N_1)$ and $E(N_2)$ represent maximum absolute error with number of grid points N_1 and N_2, respectively. The Peclet number is defined by

P_e =heat transport by convection/heat transport by diffusion=$\frac{\epsilon}{\gamma}$. The obtained results are presented in tabular form and illustrated in graphs also.

Problems having Dirichlet boundary conditions

Example 1 We consider the advection–diffusion problem [9] in (0, 2) with exact solution

$$u(x,t) = \exp(-\gamma t)\sin(x - \epsilon t) \tag{6.3}$$

Boundary conditions are taken from the exact solution, and initial condition is

$$\psi(x) = \sin(x) \tag{6.4}$$

This problem is solved with $\Delta t = 0.001$ for different values of N. Computed results for $N = 120$ at $t = 1.0$ for different values of P_e are reported in Table 1 and compared with those obtained with $\Delta t = 0.00078125$ by the fourth-order method given by Dehghan [9]. It may be noticed that our method is giving good results even at high Peclet number. Table 2 shows the results for different values of γ and $\epsilon = 1.0$. Absolute errors for different values of N at $t = 1.0$, $\epsilon = 1.0$ and $\gamma = 0.001$ are reported in Table 3. It can be observed that solutions are very accurate and the CPU time is very small. Numerical solutions are depicted in Figs. 1 and 2 with $\gamma = 1.0, t \le 2$ and $\gamma = 0.001, t \le 1$, respectively.

Example 2 We consider the advection–diffusion Eq. (1.1) in (0, 1) with $\epsilon = 1.0, \gamma = 0.1$ and following initial condition

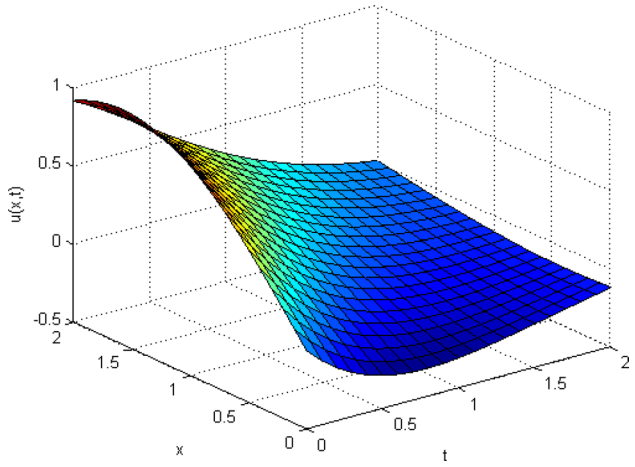

Fig. 1 Plot of numerical solutions of Example 1 at $t \le 2$ with $N = 20\epsilon = 1, \gamma = 1.0$ and $\Delta t = 0.001$

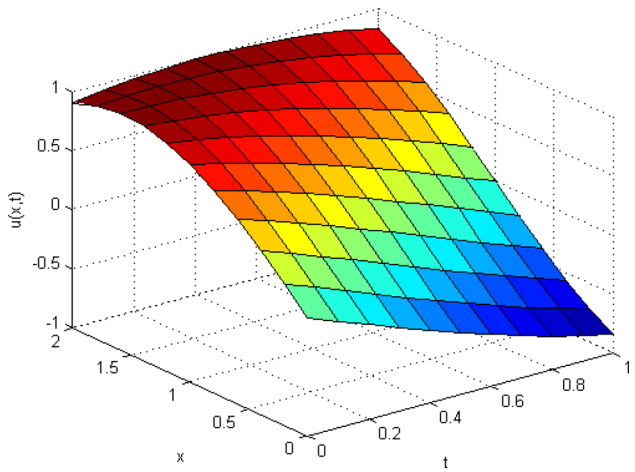

Fig. 2 Plot of numerical solutions of Example 1 at $t \le 1$ with $N = 10, \epsilon = 1, \gamma = 0.001$ and $\Delta t = 0.001$

$$u(x,0) = exp(5x)\left[cos\left(\frac{\pi x}{2}\right) + .25sin\left(\frac{\pi x}{2}\right)\right] \tag{6.5}$$

The exact solution [9] of this advection–diffusion equation is given by

$$u(x,t) = \exp(5(x - t/2))\exp\left(-\frac{\pi^2 t}{40}\right) \left[\cos\left(\frac{\pi x}{2}\right) + .25\sin\left(\frac{\pi x}{2}\right)\right] \tag{6.6}$$

Absolute errors with $\Delta t = 0.01$ for different values of h at $t = 2$ are reported in Table 4. It is observed that our results are better than those given by Dehghan [9]. In Table 6, absolute errors at $t = 5.0$ are given and also compared with the errors obtained by Mittal and Jain [4]. We can conclude from the table that the results obtained by the present method are

Table 1 Absolute errors for Example 1 at $t = 1.0$ at different values of P_e

x	$p_e = 1000$		$p_e = 10,000$		$p_e = 20,000$	
	Dehghan [9]	Proposed method	Dehghan [9]	Proposed method	Dehghan [9]	Proposed method
0.25	1.45E−06	1.53E−08	1.06E−06	1.49E−08	1.02E−04	1.42E−08
0.50	1.75E−06	3.67E−08	4.91E−07	3.60E−08	2.20E−04	3.53E−08
0.75	1.90E−06	6.08E−08	6.94E−06	6.32E−08	3.47E−04	6.51E−08
1.00	9.75E−07	8.22E−08	2.56E−05	7.80E−08	4.71E−04	7.50E−08
1.25	1.10E−07	8.12E−08	7.02E−05	9.11E−08	6.52E−04	9.52E−08
1.50	1.05E−07	7.32E−08	1.58E−04	5.60E−08	8.02E−04	1.16E−08
1.75	2.88E−07	6.63E−08	2.69E−04	9.05E−08	7.40E−04	9.37E−08

Table 2 Absolute errors of Example 1 with $\epsilon = 1.0$ and different values of γ

x	$\gamma = 1.0$	$\gamma = 0.1$	$\gamma = 0.01$
0.1	4.58E−09	4.00E−09	5.59E−09
0.2	8.74E−09	9.15E−09	1.14E−08
0.3	1.24E−08	1.52E−08	1.89E−08
0.4	1.56E−08	2.21E−08	2.72E−08
0.5	1.81E−08	2.94E−08	3.62E−08
0.6	2.01E−08	3.68E−08	4.56E−08
0.7	2.16E−08	4.42E−08	5.52E−08
0.8	2.24E−08	5.12E−08	6.45E−08
0.9	2.27E−08	5.75E−08	7.25E−08

Table 3 Maximum absolute errors for Example 1 at $t = 1.0$ with different values of n for $\epsilon = 1.0$ and $\gamma = 0.001$

n	$\Delta t = 0.01$	$\Delta t = 0.001$	CPU time(s)
10	2.08E−05	1.08E−05	1
20	1.03E−05	7.52E−07	1
40	9.64E−06	1.33E−07	1
80	9.74E−06	9.99E−08	1
160	8.29E−06	9.10E−08	1

better than those obtained by redefined B-splines. Numerical results obtained for $t \leqslant 2$ are depicted in Fig. 3 for $N = 20$ and $\Delta t = 0.01$ (Figs. 4, 5).

Example 3 We consider (1.1) with the steady-state solution [27] given by

$$u(x,t) = \frac{e^{\epsilon x/\gamma} - 1}{e^{\epsilon/\gamma} - 1}, 0 < x < 1 \tag{6.7}$$

Initial and boundary conditions are taken from the exact solution(4.6). Numerical solutions at $t = 1.0, t = 5.0$ and

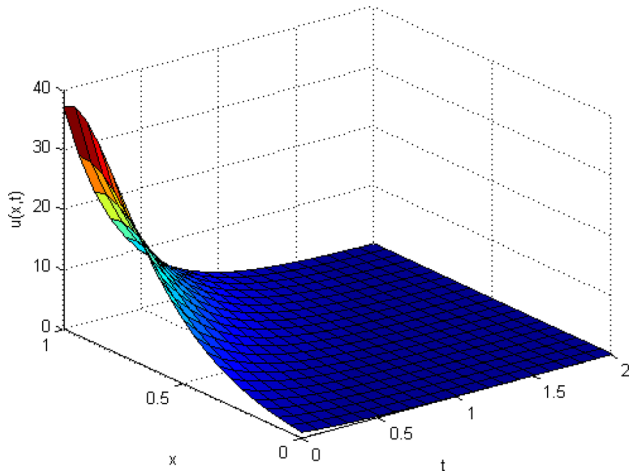

Fig. 3 Plot of numerical solutions of Example 2 with $N = 20$ and $\Delta t = 0.01$

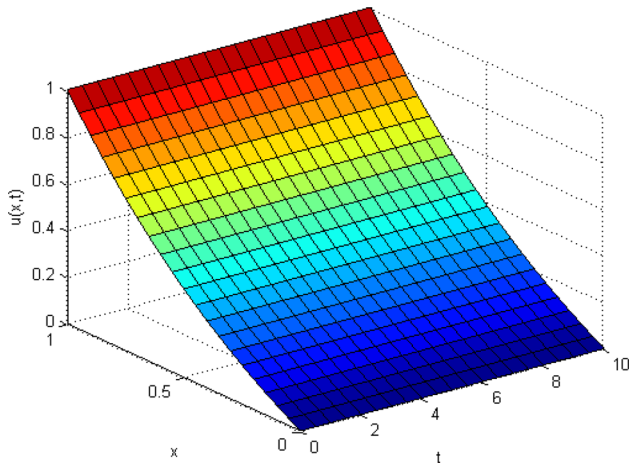

Fig. 4 Plot of numerical solutions of Example 3 with $N = 20$ and $\Delta t = 0.1$

$t = 10.0$ are reported in Table 5 with $\Delta t = 0.1$ and $N = 300$. Numerical solutions have been plotted in Fig. 4 against the values of x and t. CPU time has been estimated to be 1 second. Maximum absolute errors, rate of convergence

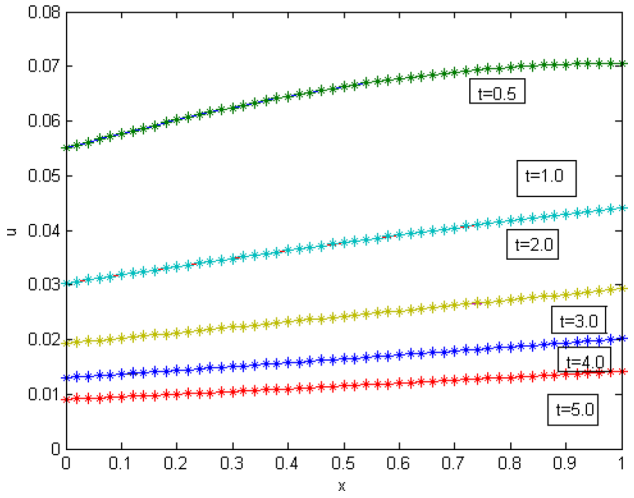

Fig. 5 Plot of numerical solutions of Example 4 with $N = 50, \Delta t = 0.001, \epsilon = 1.0$ and $\gamma = 1.0$

Table 4 Absolute errors for Example 2 at $t = 2.0$

x	$h = 0.02$		$h = 0.01$	
	Dehghan [9]	Proposed method	Dehghan [9]	Proposed method
0.1	1.8035E−06	4.99E−07	7.17E−06	5.03E−07
0.2	2.7685E−06	1.44E−06	1.10E−05	1.46E−06
0.3	4.1679E−06	3.06E−06	1.66E−05	3.09E−06
0.4	6.1705E−06	5.62E−06	2.46E−05	5.69E−06
0.5	9.0026E−06	9.36E−06	3.59E−05	9.50E−06
0.6	1.2955E−05	1.43E−05	5.16E−05	1.45E−05
0.7	1.8360E−05	1.98E−05	7.32E−05	2.01E−05
0.8	2.5476E−05	2.37E−05	1.02E−04	2.42E−05
0.9	3.4134E−05	2.08E−05	1.36E−04	2.13E−05

Table 5 Absolute errors of Example 3 with $\Delta t = 0.1$ and $N = 300$

x	$t = 1.0$	$t = 5.0$	10.0
0.1	8.81E−16	4.62E−15	6.59E−16
0.2	6.72E−15	1.14E−14	5.16E−15
0.3	1.76E−14	2.22E−14	1.68E−14
0.4	2.71E−14	3.24E−14	2.85E−14
0.5	3.54E−14	4.72E−14	4.03E−14
0.6	4.58E−14	5.31E−14	4.57E−14
0.7	5.03E−14	6.19E−14	4.35E−14
0.8	4.55E−14	5.31E−14	1.59E−14
0.9	2.64E−14	1.76E−14	2.33E−15

and CPU time at different values of N are represented in Table 6. We observe that CPU time is very small even at large values of N.

Table 6 Maximum absolute errors and rate of convergence for Example 3 at $t = 1.0$ with different values of n

n	L_∞ errors	Order of convergence	CPU time(s)
10	9.46E−08	–	1
20	6.18E−09	3.94	1
40	3.91E−10	3.98	1
80	2.45E−11	3.99	1
160	1.51E−12	4.02	1
320	5.77E−14	4.71	1

Table 7 Absolute errors for Example 4 with $\epsilon = 1.0$, $\gamma = 1.0$ and, $\Delta t = 0.001$

x	$t = 0.5$	$t = 1.0$	$t = 5.0$
0.1	7.86E−08	5.44E−10	2.93E−12
0.2	1.51E−07	8.60E−10	3.80E−12
0.3	2.16E−07	4.20E−12	6.38E−07
0.4	2.65E−07	8.28E−10	4.09E−12
0.5	2.91E−07	6.01E−10	3.63E−12
0.6	2.91E−07	3.36E−10	2.96E−12
0.7	2.59E−07	1.25E−10	2.27E−12
0.8	1.98E−07	5.19E−11	1.69E−12
0.9	1.11E−07	3.10E−10	1.69E−12

Example 4 We consider (1.1) in (0, 1) with the following boundary conditions

$$u(0,t) = \frac{1}{s}\exp\left(-50\frac{t^2}{s}\right) \text{ and } u(1,t) = \frac{1}{s}exp\left(-50\frac{(1-t)^2}{s}\right)$$

and with the initial condition given by

$$u(x,0) = \frac{1}{s}\exp\left(\frac{-50x^2}{s}\right) \tag{6.8}$$

The exact solution [4] of the problem is

$$u(x,t) = \frac{1}{s}\exp\left(-50\frac{(x-t)^2}{s}\right) \tag{6.9}$$

where $s = 1 + 200\gamma t$. Absolute errors at different grid points are given in Table 7 at time levels of $t = 0.5$, $t = 1.0$ and $t = 5.0$ with $N = 10$. Results obtained by the present method are compared with those obtained by Nazir et al [28] in Table 8. Our solutions are found to be better than the solutions given in [4, 29] and [28]. Numerical and exact

Table 8 Absolute errors of Example 4 with $\epsilon = 1.0$ and $\gamma = 1.0$

n	$t = 1.0$		$t = 2.0$	
	$\Delta t = 0.001$	$\Delta t = 0.001$	$\Delta t = 0.01$	$\Delta t = 0.001$
	Proposed method	Nazir et al. [28]	Proposed method	Nazir et al. [28]
4	6.34E−07	3.38E−05	9.49E−09	6.66E−06
8	3.07E−08	7.93E−06	1.47E−08	1.63E−06
16	3.90E−09	1.92E−06	1.64E−08	4.08E−07
32	1.95E−09	4.66E−07	1.65E−08	1.02E−07
64	1.83E−09	1.03E−07	1.65E−08	2.56E−08
128	1.83E−09	1.14E−08	1.66E−08	6.54E−09

solutions have been plotted in Fig. 5. We can see that numerical results coincide with the exact results.

Example 5 We consider problem [30] with an analytical solution of a Gaussian pulse of unit height with its center at $x = 1$ in a domain $[0, 9]$. Its analytical solution is given by

$$u(x,t) = \frac{1}{\sqrt{4t+1}} \exp\left[\frac{x - x_0 - \epsilon t}{\gamma(4t+1)}\right] \tag{6.10}$$

The initial condition is obtained from the exact solution by taking $t = 0$. The boundary conditions are given by $u(0, t) = u(9, t) = 0$. We performed numerical simulations up to time $t = 5$ with $\epsilon = .008$ and $\gamma = .5$, and results are reported in Table 9 and compared with those given in [30]. We found that the fourth-order method is giving better results than QRTDQ and QNTDQ with very small CPU time.

Problems having Neumann boundary conditions

Example 6 We consider the advection–diffusion equation with the boundary conditions

$$\begin{aligned} \left(\frac{\partial u}{\partial x}\right)_{(a,t)} &= \exp(-\gamma t)\cos(a - \epsilon t) \ \ and \\ \left(\frac{\partial u}{\partial x}\right)_{(b,t)} &= \exp(-\gamma t)\cos(b - \epsilon t) \end{aligned} \tag{6.11}$$

and with the initial condition defined by

$$u(x, 0) = \sin(x) \tag{6.12}$$

The problem has the exact solution [28] of the form

$$u(x,t) = \exp(-\gamma t)\sin(x - \epsilon t) \tag{6.13}$$

We consider this problem in (0, 2). Absolute errors at grid points for different Peclet numbers are illustrated in Table 10. It may be seen that excellent results have been obtained. Table 11 contains absolute errors and CPU time for different values of N at $t = 1.0$. Space and time graphs of numerical and exact solutions are given in Figs. 6 and 7 for $\gamma = 0.01$ and $\gamma = 0.001$, respectively. In Fig. 8, numerical and exact solutions are depicted up to time $t = 5.0$ with $\epsilon = \gamma = 1.0$. It may be seen that approximate solutions are very close to the exact solutions.

Example 7 Let us consider the advection–diffusion Eq. (1.1) with the following exact solution [22]

$$u(x,t) = e^{\alpha x + \beta t}, \quad 0 < x < 1 \tag{6.14}$$

Boundary conditions are defined as follows

Table 9 A comparison of maximum absolute (L_∞) errors and CPU time with different values of N for Example 5

N	QRTDQ [30]	CPU(s)	QNTDQ [30]	CPU(s)	Present method	CPU (s)
21	1.42E−03	1	1.40E−03	2	1.31E−05	2
31	1.42E−03	4	1.53E−03	4	3.75E−06	3
46	1.26E−03	8	1.24E−03	8	5.80E−07	4
91	5.09E−05	30	1.17E−04	31	2.93E−08	5
181	7.75E−07	121	6.28E−08	123	5.43E−09	9

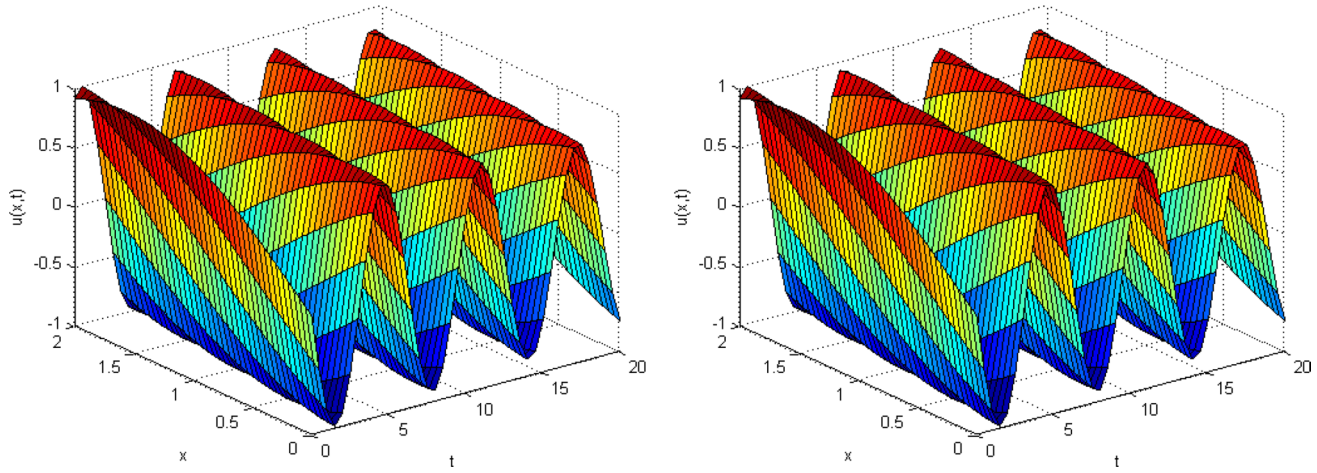

Fig. 6 Numerical and exact solutions of Example 6 with $N = 40, \Delta t = 0.001, \epsilon = 1.0$ and $\gamma = 0.01$ up to $t \leq 20$

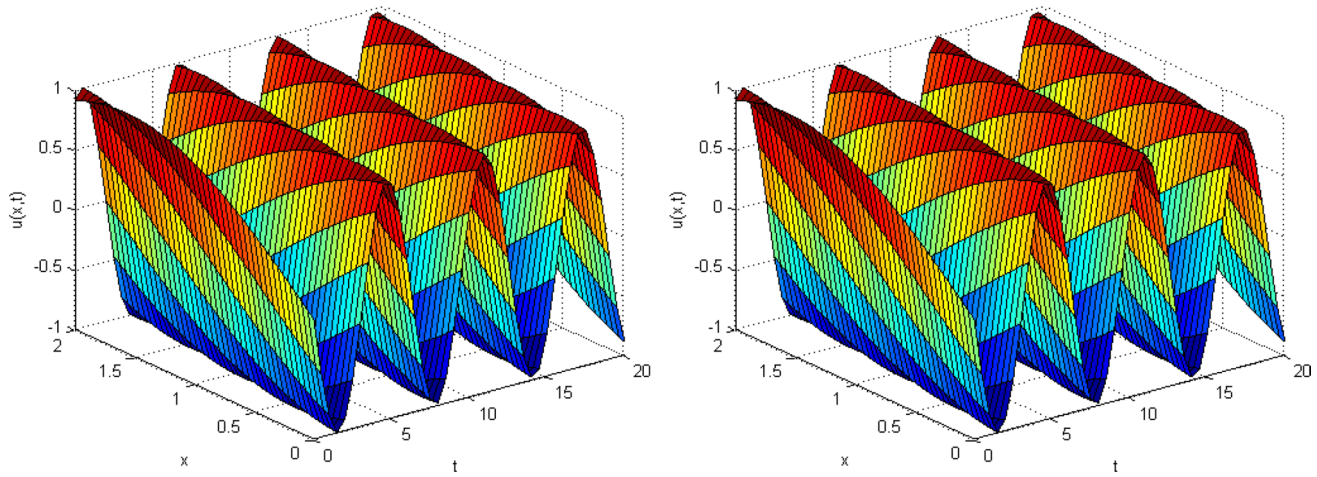

Fig. 7 Numerical and exact solutions of Example 6 with $N = 40, \Delta t = 0.001, \epsilon = 1.0$ and $\gamma = 0.001$ up to $t \leq 20$

$$\left(\frac{\partial u}{\partial x}\right)_{(0,t)} = \alpha e^{\beta t}, \left(\frac{\partial u}{\partial x}\right)_{(1,t)} = \alpha e^{\alpha+\beta t} \tag{6.15}$$

Initial condition is given by

$$u(x,0) = e^{\alpha x} \tag{6.16}$$

This problem has been solved by taking $\alpha = 0.02854797991928, \beta = -0.0999, \Delta t = 0.001, N = 40, \epsilon = 3.5$ and $\gamma = 0.022$. Excellent results have been obtained which are presented in tabular form (Table 12) and compared with the solutions obtained by Mittal and Jain [22]. It is noticed that our method is better than the methods given in [22].

Example 8 Consider (1.1) with the exact solution [22]

$$u(x,t) = e^{\alpha x+\beta t}, \ 0 < x < 1 \tag{6.17}$$

where $\alpha = 1.17712434446770, \ \beta = -0.09, \ \Delta t = 0.001, N = 80, \epsilon = 0.1$ and $\gamma = 0.2$. Initial and Neumann boundary conditions are taken from the exact solution. We computed results for different values of parameters which are reported in Table 13. Our results are better than the results obtained by [22].

Example 9 We consider the following Neumann boundary conditions for (1.1) in (0, 1)

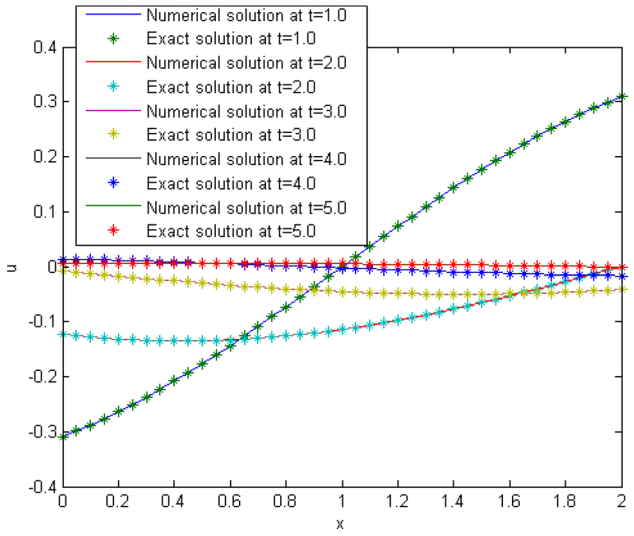

Fig. 8 Plot of numerical solutions of Example 6 with $N = 40, \Delta t = 0.001, \epsilon = 1.0$ and $\gamma = 1.0$

Table 10 Absolute errors for Example 6 at $t = 1.0$ with $\Delta t = 0.001$ and $N = 120$

x	$P_e = 1000$	$P_e = 10,000$	$P_e = 20,000$
0.25	7.2015E−08	7.2714E−08	7.4100E−08
0.50	7.6598E−08	7.5796E−08	7.3890E−08
0.75	8.1385E−08	8.2858E−08	8.5450E−08
1.00	8.3670E−08	8.1700E−08	7.8370E−08
1.25	8.1139E−08	8.4100E−08	8.8200E−08
1.50	7.3548E−08	6.9378E−08	6.4500E−08
1.75	6.1460E−08	6.6985E−08	7.2530E−08

Table 11 Maximum absolute errors for Example 6 at $t = 1.0$ with different values of n

n	L_∞ errors	CPU time(s)
5	4.20E−04	2
10	1.06E−05	2
20	7.10E−07	3
40	1.20E−07	3
80	8.50E−08	3
160	8.30E−08	3

$$\left(\frac{\partial u}{\partial x}\right)_{(0,t)} = -ac\ \exp(bt), \left(\frac{\partial u}{\partial x}\right)_{(1,t)} = -ac\ \exp(bt - c) \tag{6.18}$$

Initial condition is $u(x, 0) = a\ exp(-cx)$, and the exact solution is [31]

Table 12 Comparison of absolute errors for Example 7

t	Proposed method	Method I [22]	Method II [22]
0.2	1.66E−11	1.67E−09	8.46E−06
0.4	3.28E−11	3.29E−09	1.75E−05
0.6	4.87E−11	4.87E−09	2.70E−05
0.8	6.44E−11	6.42E−09	3.63E−05
1.0	7.94E−11	7.93E−09	4.54E−05
5.0	3.78E−10	3.27E−08	1.95E−04
20.0	7.19E−10	7.18E−08	4.30E−04

Table 13 Comparison of absolute errors for Example 8

x	Proposed method	Method I [22]	Method II [22]
0.2	2.98E−11	1.81E−05	1.92E−05
0.4	4.67E−11	3.52E−05	4.17E−05
0.6	6.28E−11	5.13E−05	6.32E−05
0.8	7.76E−11	6.68E−05	8.36E−05
1.0	9.16E−11	8.15E−05	1.03E−04
5.0	2.98E−10	2.68E−04	4.02E−04
10.0	1.11E−10	3.58E−04	6.35E−04
20.0	6.26E−10	4.21E−04	8.73E−04

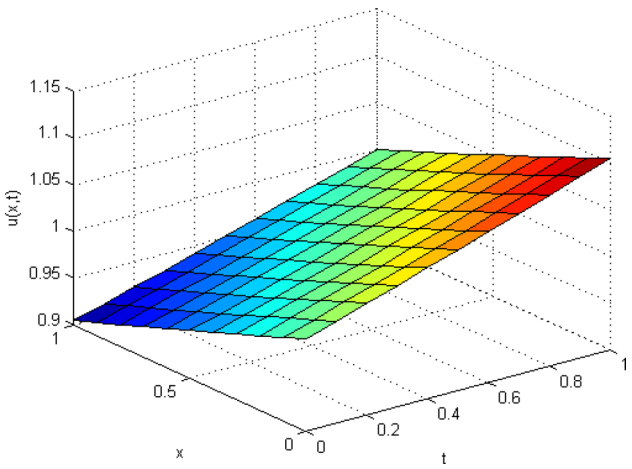

Fig. 9 Plot of numerical solutions of Example 9 with $N = 10, \Delta t = 0.001, \epsilon = 1.0$ and $\gamma = 0.001$

$$u(x, t) = ae^{bt-cx} \tag{6.19}$$

where $c = \frac{-\epsilon+\sqrt{\epsilon^2+4\gamma b}}{2\gamma}$, $a = 1.0$ and $b = 0.1$. We carried out numerical experiments by taking different values of ϵ and γ. Results obtained for $\Delta t = 0.001$ and $h = 0.1$ are reported in Table 14. It may be observed that results are very good. Computed results up to time $t = 1$ are depicted in Fig. 9. It may be found that the space and time graph is the same as illustrated by [22, 31].

Table 14 Absolute errors for Example 9 at $N = 10$ with $\epsilon = 1.0$ and $\gamma = 0.001$

x	$t = 1.0$	$t = 5.0$	$t = 20.0$
0.1	8.50E−11	5.87E−10	6.32E−09
0.2	8.54E−11	5.69E−10	6.20E−09
0.3	8.38E−11	5.81E−10	6.29E−09
0.4	8.40E−11	5.62E−10	6.16E−09
0.5	8.27E−11	5.76E−10	6.26E−09
0.6	8.26E−11	5.56E−10	6.12E−09
0.7	8.12E−11	5.70E−10	6.22E−09
0.8	8.04E−11	5.48E−10	6.08E−09
0.9	7.90E−11	5.64E−10	6.19E−09

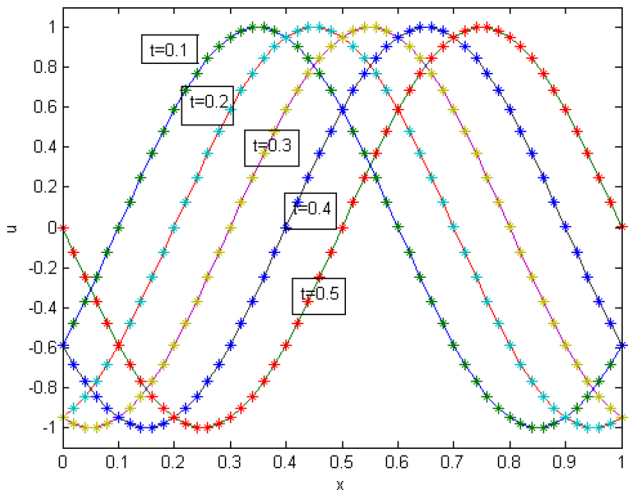

Fig. 10 Plot of numerical solutions of Example 10 with $N = 50, \Delta t = 0.001, \epsilon = 1.0$ and $\gamma = 10^{-6}$

Table 15 Absolute errors for Example 10 at $N = 50$ with $\epsilon = 1.0$ and $\Delta t = 0.001$

x	$\gamma = 10^{-5}$		$\gamma = 10^{-6}$	
	$t = 0.5$	$t = 1.0$	$t = 0.5$	$t = 1.0$
0.1	1.01E−04	2.02E−04	5.80E−07	1.16E−06
0.2	1.78E−04	3.57E−04	1.38E−05	2.75E−05
0.3	1.87E−04	3.75E−04	2.28E−05	4.57E−05
0.4	1.25E−05	2.50E−04	2.32E−05	4.64E−05
0.5	1.47E−04	2.94E−05	1.47E−05	2.94E−05
0.6	1.01E−04	2.02E−04	5.81E−07	1.16E−06
0.7	1.78E−04	3.57E−04	1.37E−05	2.75E−05
0.8	1.87E−04	3.75E−04	2.83E−05	4.57E−05
0.9	1.25E−04	2.50E−04	2.32E−05	4.64E−05

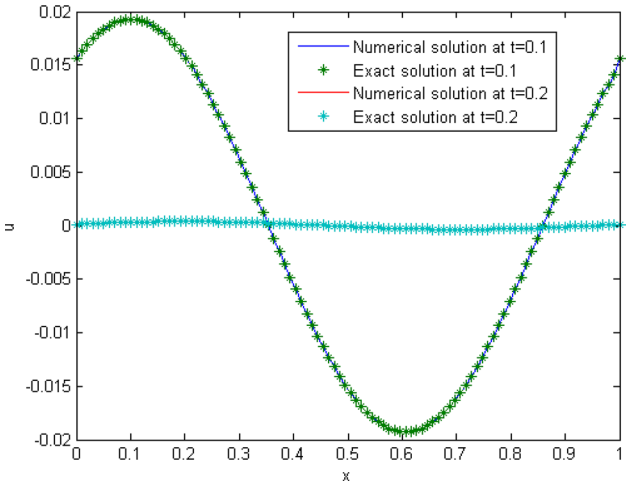

Fig. 11 Plot of numerical solutions of Example 11 with $N = 10, \Delta t = 0.001, \epsilon = 1.0$ and $\gamma = 0.001$

Table 16 Absolute errors for Example 11 at $N = 100$ with $\epsilon = 1.0$ and $\gamma = 1.0$

x	$t = 0.1$	$t = 0.2$	$t = 0.3$
0.1	7.37E−08	1.31E−06	5.65E−09
0.2	2.98E−08	1.45E−06	6.90E−09
0.3	3.26E−08	1.03E−06	9.66E−09
0.4	8.98E−08	1.12E−07	1.56E−09
0.5	1.20E−07	6.96E−07	1.43E−08
0.6	1.11E−07	1.35E−06	3.19E−08
0.7	6.66E−08	1.49E−06	4.44E−08
0.8	4.08E−09	1.07E−06	4.72E−08
0.9	5.27E−08	2.50E−07	3.91E−08

Periodic problems

Example 10 We consider the advection–diffusion Eq. (1.1) with the exact solution given by

$$u(x,t) = e^{-\gamma t}\sin(2\pi(x-\epsilon t)), 0 < x < 1 \tag{6.20}$$

This equation represents a decaying traveling wave. The initial condition is taken from the exact solution of the problem. The boundary conditions are periodic in the considered interval and may be given as

$$u(0,t) = u(1,t) \tag{6.21}$$

Computed results are illustrated in Fig. 10 and presented in Table 15. Figure 10 depicts exact and numerical solutions

at $t = 0.1, 0.2, 0.3, 0.4$ and $t = 0.5$. It may be observed that numerical solutions are very close to the exact values. We calculated the value of $m(t)$ and found that it is a constant for different time levels.

$$m(t) = \int_a^b u(x,t)\mathrm{d}t = 0.0025 \text{ for t} = 0.5 \text{ and t} = 1.0 \tag{6.22}$$

Example 11 We consider the periodic problem [32] with the boundary condition

$$u(0,t) = u(1,t), 0 < t < T \tag{6.23}$$

and with the initial condition $u(x,0) = cos(2\pi x), 0 < x < 1$. The problem has exact solution given by

$$u(x,t) = e^{-4\pi^t} cos(2\pi(x-t)), 0 < x < 1 \tag{6.24}$$

The computed absolute errors at $t = 0.1, 0.2$ and $t = 0.3$ are given in Table 16. Figure 11 depicts the numerical and the exact results with $\epsilon = 1.0$ and $\gamma = 0.001$ at $t = 0.1$ and $t = 0.2$. It may be seen that the results obtained by the proposed method are very good.

We evaluated $m(t)$ for t= 0.1, 0.2 and 0.3.

$$m(t) = \int_a^b u(x,t)\mathrm{d}t \quad m(t) = 1.87E-08 \text{ for t} = 0.1, 0.2 \text{ and } 0.3. \tag{6.25}$$

Conclusion

(a) The proposed method produces very accurate results with very small CPU time.
(b) The stability analysis is performed by using the von Neumann's method, and proposed spline-based fourth-order method has been shown to be unconditionally stable.
(c) The convergence analysis has been carried out, and it is shown that the proposed method is $O(\Delta t^2 + h^4)$ accurate.
(d) The method is stable, so the time step can be taken large compared with some other known methods. This allows one to find solution at a high level of time.
(e) Ten important convection–diffusion problems have been studied and shown to have very accurate solutions.
(f) The obtained results are presented in tables and depicted in figures also. We can observe that the method is applicable for convection-dominated problems as well as for diffusion-dominated problems.
(g) The method can solve very efficiently a large class of partial differential equations having the Dirichlet, Neumann or periodic boundary conditions.
(h) We would like to extend our research work in this direction by taking hyperbolic and two-dimensional problems.

References

1. Parlarge, J.Y.: Water transport in soils. Ann. Rev. Fluid Mech. **2**, 77–102 (1980)
2. Zlatev, Z., Berkowicz, R., Prahm, L.P.: Implementation of a variable step-size variable formula in the time-integration part of a code for treatment of longrange transport of air pollutants. J. Comput. Phys. 55 (1984)
3. Chatwin, P.C., Allen, C.M.: Mathematical models of dispersion in rivers and estuaries. Ann. Rev. Fluid Mech. **17**, 119–149 (1985)
4. Mittal, R.C., Jain, R.K.: Redefined cubic B-spline collocation method for solving convection diffusion equations. Appl. Math. Model. **36**, 5555–5573 (2012)
5. Dehghan, M.: Weighted finite difference techniques for the one-dimensional advection–diffusion equation. Appl. Math. Comput. **147**, 307–319 (2004)
6. Rizwan-Uddin: A second-order space and time nodal method for the one-dimensional convection–diffusion equation. Comput. Fluids **26**, 233–247 (1997)
7. Mohebbi, A., Dehghan, M.: High-order compact solution of the one-dimensional heat and advection–diffusion equations. Appl. Math. Model. **34**, 3071–3084 (2010)
8. Dehghan, M.: Numerical solution of the three-dimensional advection–diffusion equation. Appl. Math. Comput. **150**, 5–19 (2004)
9. Dehghan, M., Mohebbi, A.: High-order compact boundary value method for the solution of unsteady convection–diffusion problems. Math. Comput. Simul. **79**, 683–699 (2010)
10. Dehghan, M.: Quasi-implicit and two-level explicit finite-difference procedures for solving the one-dimensional advection equation. Appl. Math.Comput. Simul. **167**, 46–67 (2005)
11. Singh, I., Kumar, S.: Approximate solution of convection diffusion equation using Haar wavelet. Italian J. Pure Appl. Math. **35**, 143–154 (2015)
12. Salkuyeh, D.K.: On the finite difference approximation to the convection-diffusion equation. Appl. Math. Comput. 179 (2006)
13. Karahan, H.: Unconditional stable explicit finite difference technique for the advection–diffusion equation using spreadsheets. Adv. Eng. Softw. **38**, 80–86 (2007)
14. Karahan, H.: Implicit finite difference techniques for the advection–diffusion equation using spreadsheets. Adv. Eng. Softw. **37**, 601–608 (2006)
15. Ding, H.F., Zhang, Y.X.: A new difference scheme with high accuracy and absolute stability for solving convection–diffusion equations. J. Comput. Appl. Math. **230**, 600–606 (2009)
16. Szymkiewicz, R.: Solution of the advection–diffusion equation using the spline function and finite elements. Commun. Numer. Methods Eng. **9**, 197–206 (1993)
17. Mohammadi, R.: Exponential B-spline solution of convection–diffusion equations. Appl. Math. **4**, 933–944 (2013)
18. Mittal, R.C., Jain, R.K.: Numerical solution of convection–diffusion equation using cubic B-splines collocation methods with

Neumann's boundary conditions. Int. J. Appl. Math. Comput. **4**, 115–127 (2012)
19. Goh, J., Majid, A.A., Ismail, A.I.B. Md.: Cubic B-spline collocation method for one-dimensional heat and advection–diffusion equations. J. Appl. Math. Article ID 458701 (2012)
20. Korkmaz, A., Dag, I.: Cubic B-spline differential quadrature methods for the advection–diffusion equation. Int. J. Numer. Methods Heat Fluid Flow **22**, 1021–1036 (2012)
21. Gardner, L.R.T., Dag, I.: A numerical solution of the advection-diffusion equation using B-spline finite element. In: Proceedings of the International AMSE Conference on Systems Analysis, Control and Design, Lyon, France, pp. 109–116 (1994)
22. Dag, I., Canivar, A., Sahin, A.: Taylor–Galerkin method for advection–diffusion equation. Kybernetes **40**, 762–777 (2011)
23. Kadalbajoo, M.K., Arora, P.: Taylor–Galerkin B-spline finite element method for the one dimensional advection–diffusion equation. Numer. Methods Par. Diff. Eqs. **26**, 1206–1223 (2009)
24. Prenter, P.M.: Splines and Variational Methods. Wiley, Hoboken (1989)
25. Lucas, T.R.: Error bounds for interpolating cubic splines under various end conditions. Siam J. Numer. Anal. **11**, 569–584 (1975)
26. Sastry, S.S.: Introductory Methods of Numerical Analysis, fourth ed., PHI Learning (2009)
27. Salama, A.A., Zidan, H.Z.: Fourth order schemes of exponential type for singularly perturbed parabolic partial differential equations. J. Math. 36 (2006)
28. Nazir, T., Abbas, M., Ahmad Izani, M., Ismail, A.A., Majid, A.R.: The numerical solution of advection–diffusion problems using new cubic trigonometric B-splines approach. Appl. Math. Model. **40**, 4586–4611 (2016)
29. Chawla, M.M., Al-Zanaidi, M.A., Al-Aslab, M.G.: Extended one step time-integration schemes for convection–diffusion equations. Comput. Math. Appl. **39**, 71–84 (2000)
30. Korkmaz, A., Dag, I.: Quartic and quintic B-spline methods for advection–diffusion equation. Appl. Math. Comput. **274**, 208–219 (2016)
31. Cao, H.-H., Liu, L.-B., Zhang, Y., Fu, S.: A fourth-order method of the convection–diffusion equations with Neumann boundary conditions. Appl. Math. Comput. **217**, 9133–9141 (2011)
32. Chen, N., Haiming, G.: Alternating group explicit iterative methods for one-dimensional advection–diffusion equation. Am. J. Comput. Math. **5**, 274–282 (2015)