International Journal of Applied Mathematics and Computation
ISSN: 0974 - 4665 (Print) 0974 - 4673 (Online)
Journal homepage: www.darbose.in/ijamc
Volume 4(2) 2012 115–127

Numerical solution of convection-diffusion equation using cubic B-splines collocation methods with Neumann's boundary conditions

R.C. Mittal[a, *], **R.K. Jain**[a, †]

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

ABSTRACT

In this paper, two numerical methods are proposed to approximate the solutions of the convection-diffusion partial differential equations with Neumann's boundary conditions. The methods are based on collocation of cubic B-splines over finite elements so that we have continuity of the dependent variable and its first two derivatives throughout the solution range. In Method-I, we discretize the time derivative with Crank Nicolson scheme and handle spatial derivatives with cubic B-splines. Stability of this method has been discussed and shown that it is unconditionally stable. In Method-II, we apply cubic B-splines for spatial variable and derivatives which produce a system of first order ordinary differential equations. We solve this system by using SSP-RK54 scheme. These methods needs less storage space that causes to less accumulation of numerical errors. In numerical test problems, the performance of these methods is shown by computing L_∞ and L_2 errors for different time levels. Illustrative five examples are included to demonstrate the validity and applicability of these methods. Results shown by these methods are found to be in good agreement with the exact solutions. Easy and economical implementation is the strength of these methods.

Keywords: Convection-diffusion partial differential equation; Neumann's boundary conditions; cubic B-splines basis functions; SSP-RK54 scheme; Thomas algorithm

1. Introduction

In this paper, we consider the numerical solution of the following one dimensional convection-diffusion equation

$$\frac{\partial u}{\partial t} + \varepsilon \frac{\partial u}{\partial x} = \gamma \frac{\partial^2 u}{\partial x^2}, \quad 0 \le x \le L, \;\; 0 \le t \le T \tag{1.1}$$

with initial condition

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

and Neumann's boundary conditions are as follows [7]:

$$\left(\frac{\partial u}{\partial x}\right)_{(0,t)} = g_0(t), \;\; \left(\frac{\partial u}{\partial x}\right)_{(L,t)} = g_1(t), \quad t \in [0,T] \tag{1.3}$$

*rcmmmfma@iitr.ernet.in
†rkjain_2000@rediffmail.com

where the parameter γ is the viscosity coefficient and ε is the phase speed and both are assumed to be positive. φ,g_0 and g_1 are known functions of sufficient smoothness.

The convection-diffusion problems arise in many important applications in science and engineering such as fluid motion, heat transfer, astrophysics, oceanography, meteorology, semiconductors, hydraulics, pollutant and sediment transport, and chemical engineering.

In literature, various numerical techniques such as finite differences, finite elements, spectral procedures, and the method of lines have been developed and compared for solving the one dimensional convection-diffusion equation with Dirichlets boundary conditions [1–6, 8–11, 13–15]. Most of these techniques are based on the two-level finite difference approximations. However, fewer difference schemes have been developed to solve the convection-diffusion equation with Neumann's boundary conditions, which are much more difficult to handle than Dirichlet conditions. In [7], Cao et. al developed a fourth-order compact finite difference scheme for solving the convection-diffusion equations with Neumann's boundary conditions.

In this paper, two numerical methods are developed to approximate the solutions of the convection-diffusion partial differential equations with Neumann's boundary conditions using cubic B-splines collocation methods. It is well known that B-spline collocation methods produce better numerical approximation in comparison to finite difference method. The finite difference solution is available only at predetermined nodal points. The solution at any other point must be obtained by interpolation. On the other hand from the cubic B-spline collocation method solutions can be given in terms of cubic B-splines defined over the whole interval $[0, L]$. Thus, the solution is known at least in principle at every point in $[0, L]$. In this work, it is aimed to effectively employ the collocation methods for solving convection-diffusion equation with Neumann's boundary conditions which are often encountered in engineering applications [7, 16]. It should be noted that the zero or constant Neumann's conditions (zero or constant flux on the boundary) are typical requirement to describe the actual processes in mathematical modeling. In Method-I, for solving equations (1.1)- (1.3), we discretize the time derivative by using Crank Nicolson scheme,then solve it by using collocation method which is based on cubic B-splines. We know that B-splines have mainly two main features which are useful in numerical work. One feature is that the continuity conditions are inherent. Hence, B-spline is the smoothest interpolating function compared with other piecewise polynomial interpolating functions. Another feature of B-splines is that they have small local support, i.e. each B-spline function is only non-zero over a few mesh subintervals, so that the resulting matrix for the discretization equation is tightly banded which is very attractive for practical engineering problems. Due to their smoothness and capability to handle local phenomena, B-splines offer distinct advantages. In combination with collocation, this significantly simplifies the solution procedure of differential equations. We prove the stability of Method-I by using von Neumann stability scheme [12] and show that the Method-I, is unconditionally stable. In Method-II, we use cubic B-splines basis functions for spatial variable and derivatives then the resulting system of differential equations are solved by using SSP-RK54 scheme [11].

This paper is organized as follows. In Section 2, description of cubic B-splines collocation method is explained. In Sections 3, procedure for implementation of Method-I is describe for equation (1.1) and in Section 4, stability of this method is discussed by applying von-Neumann stability method. In Section 5, procedure to obtain initial vector which is required to start Method-I is explained. In Section 6, implementation of Method-II is discussed. We present five numerical examples to establish the applicability and accuracy of the proposed methods computationally in Section 7. The final conclusion is given in Section 8 that briefly summarizes the numerical outcomes.

2. Description of the methods

In cubic B-splines collocation method the approximate solution can be written as a linear combination of cubic B-splines basis functions for the approximation space under consideration.

Table 1: Coefficient of cubic B-splines and derivatives at nodes x_j

x	x_{j-2}	x_{j-1}	x_j	x_{j+1}	x_{j+2}
$B_j(x)$	0	1	4	1	0
$B_j^{'}(x)$	0	$\frac{3}{h}$	0	$\frac{-3}{h}$	0
$B_j^{''}(x)$	0	$\frac{6}{h^2}$	$\frac{-12}{h^2}$	$\frac{6}{h^2}$	0

We consider a mesh $0 = x_0 < x_1, \ldots, x_{N-1} < x_N = L$ as a uniform partition of the solution domain $0 \leq x \leq L$ by the knots x_j with $h = x_{j+1} - x_j = \dfrac{L}{N}, j = 0, \ldots, N-1$. Our numerical treatment for solving equation (1.1) using the collocation method with cubic B-splines is to find an approximate solution $U^N(x,t)$ to the exact solution $u(x,t)$ in the form:

$$U^N(x,t) = \sum_{j=-1}^{N+1} \alpha_j(t) B_j(x) \tag{2.1}$$

where $\alpha_j(t)$ are unknown time dependent quantities to be determined from the boundary conditions and collocation from the differential equation. The cubic B-splines $B_j(x)$ at the knots is given by

$$B_j(x) = \frac{1}{h^3} \begin{cases} (x - x_{j-2})^3, & x \in [x_{j-2}, x_{j-1}) \\ (x - x_{j-2})^3 - 4(x - x_{j-1})^3, & x \in [x_{j-1}, x_j) \\ (x_{j+2} - x)^3 - 4(x_{j+1} - x)^3, & x \in [x_j, x_{j+1}) \\ (x_{j+2} - x)^3, & x \in [x_{j+1}, x_{j+2}) \\ 0, & \text{otherwise} \end{cases} \tag{2.2}$$

where $B_{-1}, B_0, B_1, \ldots, B_{N-1}, B_N, B_{N+1}$ forms a basis over the region $0 \leq x \leq L$. Each cubic B-splines cover four elements so that each element is covered by four cubic B-splines. The values of $B_j(x)$ and its derivative may be tabulated as in Table-1.

Using approximate function (2.1) and cubic B-splines functions (2.2), the approximate values of $U^N(x)$ and its two derivatives at the knots/nodes are determined in terms of the time parameters α_j as follows:

$$\begin{aligned} U_j &= \alpha_{j-1} + 4\alpha_j + \alpha_{j+1} \\ hU_j^{'} &= 3(\alpha_{j+1} - \alpha_{j-1}) \\ h^2 U_j^{''} &= 6(\alpha_{j-1} - 2\alpha_j + \alpha_{j+1}) \end{aligned} \tag{2.3}$$

Using (2.1) and the boundary condition (1.3), we get the approximate solution at the boundary points as

$$\begin{aligned} U_x(x_0,t) &= \sum_{j=-1}^{1} \alpha_j(t) B_j^{'}(x_0) \\ U_x(x_N,t) &= \sum_{j=N-1}^{N+1} \alpha_j(t) B_j^{'}(x_N) \end{aligned} \tag{2.4}$$

Using Table 1, in (2.4) we get

$$\begin{aligned} \alpha_1 - \alpha_{-1} &= \frac{h}{3} g_0(t) \\ \alpha_{N+1} - \alpha_{N-1} &= \frac{h}{3} g_1(t) \end{aligned} \tag{2.5}$$

3. Implementation of Method-I

First, discretizing the time derivative in the usual finite difference way and applying Crank-Nicolson scheme to space derivative in (1.1), we get

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

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

Now, using (2.1) in (3.2), we get

$$\begin{aligned} &\sum_{j=-1}^{N+1} \alpha_j^{n+1}(t) B_j(x) + \varepsilon \frac{\Delta t}{2} \sum_{j=-1}^{N+1} \alpha_j^{n+1}(t) B_j^{'}(x) - \gamma \frac{\Delta t}{2} \sum_{j=-1}^{N+1} \alpha_j^{n+1}(t) B_j^{''}(x) \\ &\quad = \sum_{j=-1}^{N+1} \alpha_j^{n}(t) B_j(x) - \varepsilon \frac{\Delta t}{2} \sum_{j=-1}^{N+1} \alpha_j^{n}(t) B_j^{'}(x) + \gamma \frac{\Delta t}{2} \sum_{j=-1}^{N+1} \alpha_j^{n}(t) B_j^{''}(x) \end{aligned} \tag{3.3}$$

Using Table 1 with (2.3) and (2.5) in (3.3) we get

$$\mathbf{A}\alpha^{\mathbf{n+1}} = \mathbf{B}\alpha^{\mathbf{n}} + \mathbf{b} \tag{3.4}$$

where

$$\mathbf{A} = \begin{pmatrix} y & x+z & & & & \\ x & y & x & & & \\ & \cdot & \cdot & \cdot & & \\ & & \cdot & \cdot & \cdot & \\ & & x & y & x \\ & & & x+z & y \end{pmatrix}, \mathbf{B} = \begin{pmatrix} q & p+r & & & & \\ p & q & p & & & \\ & \cdot & \cdot & \cdot & & \\ & & \cdot & \cdot & \cdot & \\ & & p & q & p \\ & & & p+r & q \end{pmatrix}$$

$$\mathbf{b} = \begin{pmatrix} b_0 \\ 0 \\ \cdot \\ \cdot \\ 0 \\ b_N \end{pmatrix}, \alpha^{n+1} = \begin{pmatrix} \alpha_0^{n+1} \\ \alpha_1^{n+1} \\ \cdot \\ \cdot \\ \alpha_{N-1}^{n+1} \\ \alpha_N^{n+1} \end{pmatrix}, \alpha^{n} = \begin{pmatrix} \alpha_0^{n} \\ \alpha_1^{n} \\ \cdot \\ \cdot \\ \alpha_{N-1}^{n} \\ \alpha_N^{n} \end{pmatrix}$$

$$x = 1 - \frac{3}{2}\varepsilon(\frac{k}{h}) - 3\gamma(\frac{k}{h^2}), y = 4 + 6\gamma(\frac{k}{h^2}), z = 1 + \frac{3}{2}\varepsilon(\frac{k}{h}) - 3\gamma(\frac{k}{h^2})$$

$$p = 1 + \frac{3}{2}\varepsilon(\frac{k}{h}) + 3\gamma(\frac{k}{h^2}), q = 4 - 6\gamma(\frac{k}{h^2}), r = 1 - \frac{3}{2}\varepsilon(\frac{k}{h}) + 3\gamma(\frac{k}{h^2})$$

$$b_0 = \frac{h}{3}[xg_1(t^{n+1}) - pg_1(t^n)], b_N = \frac{h}{3}[-zg_2(t^{n+1}) + rg_2(t^n)],$$

Here **A** and **B** are $(N+1) \times (N+1)$ tri-diagonal matrices and **b** is an $(N+1)$ order column vector, which depends on the boundary conditions. The time evolution of the approximate solution $U^N(x,t)$ is determined by vector α^n. This is found by repeatedly solving the recurrence relationship once the initial vector α^0 has been computed from the initial conditions. The matrix **A** in (3.4) is tri-diagonal and so the system (3.4)can be solved using Thomas algorithm.

4. Stability of the Method-I

We have investigated stability of the proposed method by applying von-Neumann stability method. For testing stability, we consider the equation

$$x\alpha_{j-1}^{n+1} + y\alpha_j^{n+1} + z\alpha_{j+1}^{n+1} = p\alpha_{j-1}^n + q\alpha_j^n + r\alpha_{j+1}^n \tag{4.1}$$

where x, y, z, p, q and r are given in equation (3.4).

Now, we consider the trial solution (one Fourier mode out of the full solution) at a given point x_j

$$\alpha_j^n = \xi^n \exp(ij\beta h) \tag{4.2}$$

where $i = \sqrt(-1), \beta$ is the mode number and h is the element size. Now, by substituting

$$\alpha_j^n = \xi^n \exp(ij\beta h)$$

in (4.1), and simplifying the equation, we get

$$\xi = \frac{p\exp(-i\beta h) + q + r\exp(i\beta h)}{x\exp(-i\beta h) + y + z\exp(i\beta h)} \tag{4.3}$$

Now, substituting the values of x, y, z, p, q and r from (3.4) in (4.3) and simplifying, we get

$$\xi = \frac{[2(\cos\beta h + 2) - 6\gamma(\frac{k}{h^2})(1 - \cos\beta h)] - i[3\varepsilon(\frac{k}{h})\sin\beta h]}{[2(\cos\beta h + 2) + 6\gamma(\frac{k}{h^2})(1 - \cos\beta h)] + i[3\varepsilon(\frac{k}{h})\sin\beta h]} \tag{4.4}$$

$$\Rightarrow \xi = \frac{X_1 - iY}{X_2 + iY} \tag{4.5}$$

where

$$X_1 = 2(\cos\beta h + 2) - 6\gamma(\frac{k}{h^2})(1 - \cos\beta h)$$
$$X_2 = 2(\cos\beta h + 2) + 6\gamma(\frac{k}{h^2})(1 - \cos\beta h)$$
$$Y = 3\varepsilon(\frac{k}{h})\sin\beta h$$

Now substitute $\lambda = \dfrac{k}{h^2}$,$\rho = \gamma\lambda$,Péclet number$(P_e) = \dfrac{\varepsilon h}{\gamma}$and $\mu = \cos\beta h$, in (4.4), we get

$$\xi = \frac{[2(\mu+2) - 6\rho(1-\mu)] - i[3P_e\rho\surd(1-\mu^2)]}{[2(\mu+2) + 6\rho(1-\mu)] + i[3P_e\rho\surd(1-\mu^2)]}$$

$$\Rightarrow |\xi|^2 = \frac{[2(\mu+2) - 6\rho(1-\mu)]^2 + [9P_e^2\rho^2(1-\mu^2)]}{[2(\mu+2) + 6\rho(1-\mu)]^2 + [9P_e^2\rho^2(1-\mu^2)]} \tag{4.6}$$

Since numerator in (4.6) is less than denominator, therefore $|\xi| \leq 1$, hence the method is unconditionally stable. It means that there is no restriction on the grid size, i.e. on h and Δt, but we should choose them in such a way that the accuracy of the scheme is not degraded.

5. The initial vector α^0

The initial vector α^0 can be obtained from the initial condition and boundary values of the derivatives of the initial condition as the following expressions:

$$U_x(x_j, 0) = \varphi^{'}(x_j), j = 0,$$
$$U(x_j, 0) = \varphi(x_j), j = 1, \ldots, N-1,$$
$$U_x(x_j, 0) = \varphi^{'}(x_j), j = N.$$

This yields a $(N+1) \times (N+1)$ system of equations, of the form

$$\mathbf{A}\alpha^{\mathbf{0}} = \mathbf{b} \tag{5.1}$$

$$\mathbf{A} = \begin{pmatrix} 4 & 2 & & & \\ 1 & 4 & 1 & & \\ & \cdot & \cdot & \cdot & \\ & & \cdot & \cdot & \cdot \\ & & 1 & 4 & 1 \\ & & & 2 & 4 \end{pmatrix}, \alpha^0 = \begin{pmatrix} \alpha_0^0 \\ \alpha_1^0 \\ \cdot \\ \cdot \\ \alpha_{N-1}^0 \\ \alpha_N^0 \end{pmatrix} \text{and} \mathbf{b} = \begin{pmatrix} \varphi(x_0) + \frac{h}{3}\varphi^{'}(x_0) \\ \varphi(x_1) \\ \cdot \\ \cdot \\ \varphi(x_{N-1}) \\ \varphi(x_N) - \frac{h}{3}\varphi^{'}(x_N) \end{pmatrix}$$

The solution of (5.1) can be found by Thomas algorithm.

6. Implementation of Method-II

Our numerical treatment for solving equation (1.1) using the collocation method with cubic B-splines is to find an approximate solution $U^N(x,t)$ to the exact solution $u(x,t)$ is given in (2.1), where $\alpha_j(t)$ are time dependent quantities to be determined from the boundary conditions and collocation from the differential equation.

Using (2.1) in (1.1), we have

$$\sum_{j=-1}^{N+1} \dot{\alpha_j} B_j(x) = -\varepsilon \sum_{j=-1}^{N+1} \alpha_j B_j^{'}(x) + \gamma \sum_{j=-1}^{N+1} \alpha_j B_j^{''}(x) \tag{6.1}$$

Using approximate function (2.1) and cubic B-splines functions (2.2), the approximate values of $U_t^N(x)$ and two derivatives of $U^N(x)$ at the knots/nodes are determined in terms of the time parameters α_j as follows:

$$\begin{aligned} (U_t)_j &= \dot{\alpha}_{j-1} + 4\dot{\alpha_j} + \dot{\alpha}_{j+1} \\ hU_j^{'} &= 3(\alpha_{j+1} - \alpha_{j-1}) \\ h^2 U_j^{''} &= 6(\alpha_{j-1} - 2\alpha_j + \alpha_{j+1}) \end{aligned} \tag{6.2}$$

From (2.5), we have

$$\begin{aligned} \dot{\alpha_1} - \dot{\alpha}_{-1} &= \frac{h}{3}\dot{g_0}(t) \\ \dot{\alpha}_{N+1} - \dot{\alpha}_{N-1} &= \frac{h}{3}\dot{g_1}(t) \end{aligned} \tag{6.3}$$

Using (2.4), (6.2) and (6.3) in (6.1) we get a system of ordinary differential equations of the form

$$\mathbf{A}\dot{\alpha} = \mathbf{B}\alpha + \mathbf{b} \tag{6.4}$$

where

$$\mathbf{A} = \begin{pmatrix} 4 & 2 & & & \\ 1 & 4 & 1 & & \\ & \cdot & \cdot & \cdot & \\ & & \cdot & \cdot & \cdot \\ & & 1 & 4 & 1 \\ & & & 2 & 4 \end{pmatrix}, \mathbf{B} = \begin{pmatrix} q & p+r & & & \\ p & q & p & & \\ & \cdot & \cdot & \cdot & \\ & & \cdot & \cdot & \cdot \\ & & p & q & p \\ & & & p+r & q \end{pmatrix}$$

$$\mathbf{b} = \begin{pmatrix} b_0 \\ 0 \\ \cdot \\ \cdot \\ 0 \\ b_N \end{pmatrix}, \dot{\alpha} = \begin{pmatrix} \dot{\alpha_0} \\ \dot{\alpha_1} \\ \cdot \\ \cdot \\ \dot{\alpha}_{N-1} \\ \dot{\alpha_N} \end{pmatrix}, \alpha = \begin{pmatrix} \alpha_0 \\ \alpha_1 \\ \cdot \\ \cdot \\ \alpha_{N-1} \\ \alpha_N \end{pmatrix}$$

$$p = \frac{3\varepsilon}{h} + \frac{6\gamma}{h^2}, q = \frac{-12\gamma}{h^2}, p = \frac{-3\varepsilon}{h} + \frac{6\gamma}{h^2},$$

$$b_0 = \frac{h}{3}(-pg_0^n + g_0^{\cdot\, n+1}), \qquad b_N = \frac{h}{3}(rg_1^n - g_1^{\cdot\, n+1})$$

Here **A** and **B** are $(N+1) \times (N+1)$ tri-diagonal matrices and **b** is an $(N+1)$ order column vector, which depends on the boundary conditions.

Now, we solve the first order ordinary differential equation system (6.4) by using SSP-RK54 scheme [11]. Here, the parameter α has been determined at an initial time level, to follow the procedure as given in Section 5.

7. Numerical Experiments and discussion

In this section, we present the numerical results of present method on several problems. We tested the accuracy and stability of this method for different values of $h, \Delta t, \varepsilon$ and γ.

Some important non-dimensional parameters in numerical analysis are defined as follows:

Courant number: The Courant number is defined as $C_r = \varepsilon\frac{\Delta t}{h}$

Diffusion number: The diffusion number is defined as $s = \gamma\frac{\Delta t}{h^2}$

Grid Pèclet number: The Pèclet number is defined as $P_e = \frac{C_r}{s} = \frac{\varepsilon}{\gamma}h$

When the Pèclet number is high, the convection term dominates and when the Pèclet number is low the diffusion term dominates.

To gain insight into the performance of the presented methods, five numerical examples are given in this section with L_∞ and L_2 errors ,which are obtained by following formulae:

$$L_\infty = \max|u_j^{exact} - U_j^{num}|$$

$$L_2 = \sqrt{h\left(\sum_0^N \left|u_j^{exact} - U_j^{num}\right|^2\right)}$$

Example 1 We consider the following equation [5]

$$\frac{\partial u}{\partial t} + \varepsilon\frac{\partial u}{\partial x} = \gamma\frac{\partial^2 u}{\partial x^2}, 0 \le x \le 1, 0 \le t \le T$$

with $\varepsilon = 0.1, \gamma = 0.02$ and the following initial condition $\varphi(x) = \exp(\alpha x)$, and boundary conditions are $(\frac{\partial u}{\partial t})_{(0,t)} = \alpha\exp(\beta t), (\frac{\partial u}{\partial t})_{(1,t)} = \alpha\exp(\alpha + \beta t)$.

The exact solution is given by $u(x,t) = \exp(\alpha x + \beta t)$.

In our computation, we take $\varepsilon = 0.1, \gamma = 0.02, \alpha = 1.17712434446770, \beta = -0.09, h = 0.1, k = 0.01$, so that $C_r = 0.01, s = 0.02, P_e = 0.5$. The results are computed for different time levels. The L_∞ and

Table 2: L_∞ and L_2 errors (Example-1)

T	Method-I	Method-I	Method-II	Method-II
	L_∞	L_2	L_∞	L_2
0.2	$1.81E-05$	$1.17E-05$	$1.92E-05$	$1.18E-05$
0.4	$3.52E-05$	$2.29E-05$	$4.17E-05$	$2.37E-05$
0.6	$5.13E-05$	$3.37E-05$	$6.32E-05$	$3.55E-05$
0.8	$6.68E-05$	$4.42E-05$	$8.36E-05$	$4.72E-05$
1.0	$8.15E-05$	$5.42E-05$	$1.03E-04$	$5.90E-05$
5.0	$2.68E-04$	$1.96E-04$	$4.02E-04$	$2.91E-04$
10.0	$3.58E-04$	$2.89E-04$	$6.35E-04$	$5.25E-04$
20.0	$4.21E-04$	$3.76E-04$	$8.73E-04$	$7.93E-04$

Table 3: L_∞ and L_2 errors (Example-2)

T	Method-I	Method-I	Method-II	Method-II
	L_∞	L_2	L_∞	L_2
0.2	$1.67E-09$	$1.57E-09$	$8.46E-06$	$4.53E-06$
0.4	$3.29E-09$	$3.10E-09$	$1.75E-05$	$1.22E-05$
0.6	$4.87E-09$	$4.60E-09$	$2.70E-05$	$2.08E-05$
0.8	$6.42E-09$	$6.07E-09$	$3.63E-05$	$2.96E-05$
1.0	$7.93E-09$	$7.51E-09$	$4.54E-05$	$3.82E-05$
5.0	$3.27E-08$	$3.11E-08$	$1.95E-04$	$1.81E-04$
10.0	$5.25E-08$	$4.98E-08$	$3.14E-04$	$2.96E-04$
20.0	$7.18E-08$	$6.81E-08$	$4.30E-04$	$4.07E-04$

L_2 errors are reported in Table-2. The computed errors obtained from Method-I and Method-II are approximately similar. Hence, both methods can apply for solving this problem for different time levels.

Example 2 We consider the following equation [5]

$$\frac{\partial u}{\partial t} + \varepsilon \frac{\partial u}{\partial x} = \gamma \frac{\partial^2 u}{\partial x^2}, 0 \leq x \leq 1, 0 \leq t \leq T$$

with $\varepsilon = 3.5, \gamma = 0.022$ and the following initial condition $\varphi(x) = \exp(\alpha x)$, and boundary conditions are $(\frac{\partial u}{\partial t})_{(0,t)} = \alpha \exp(\beta t), (\frac{\partial u}{\partial t})_{(1,t)} = \alpha \exp(\alpha + \beta t)$.

The exact solution is given by $u(x,t) = \exp(\alpha x + \beta t)$.

In our computation, we take $\varepsilon = 3.5, \gamma = 0.022, \alpha = 0.02854797991928, \beta = -0.0999, h = 0.1, k = 0.01$, so that $C_r = 0.35, s = 0.022, P_e = 15.90909091$. The results are computed for different time levels. The L_∞ and L_2 errors are reported in Table-3. In this problem, the results obtained from Method-I, are more accurate in comparison to Method-II. We observe that both methods work well for higher value of $P_e = 15.90909091$ for a long time $T = 20$.

Example 3 We consider the following equation [7]

$$\frac{\partial u}{\partial t} + \varepsilon \frac{\partial u}{\partial x} = \gamma \frac{\partial^2 u}{\partial x^2}, 0 \leq x \leq 1, 0 \leq t \leq T$$

with initial condition $\varphi(x) = a \exp(-cx)$, and boundary conditions are

Table 4: L_∞ and L_2 errors (Example-3)

T	Method-I	Method-I	Method-II	Method-II
	L_∞	L_2	L_∞	L_2
0.2	$1.76E-12$	$1.07E-12$	$5.36E-07$	$3.14E-07$
0.4	$3.60E-12$	$2.11E-12$	$1.31E-06$	$9.27E-07$
0.6	$5.35E-12$	$3.03E-12$	$2.62E-06$	$1.49E-06$
0.8	$7.05E-12$	$3.82E-12$	$3.34E-06$	$2.01E-06$
1.0	$8.12E-12$	$\mathbf{4.47E-12}$	$4.29E-06$	$2.45E-06$
5.0	$1.95E-11$	$1.42E-11$	$2.26E-05$	$1.90E-05$
10.0	$3.40E-11$	$2.61E-11$	$6.51E-05$	$5.72E-05$
20.0	$9.45E-11$	$7.33E-11$	$2.55E-04$	$2.34E-04$
Cao et. al [7]				
$(h=0.1, k=0.0001)$ $T=1.0$		$\mathbf{3.34E-09}$		

$(\dfrac{\partial u}{\partial t})_{(0,t)} = -ac\exp{(bt)}, (\dfrac{\partial u}{\partial t})_{(1,t)} = -ac\exp{(bt-c)}$, where $c = \dfrac{-\varepsilon+\sqrt{(\varepsilon^2+4\gamma b)}}{2\gamma}$.

The exact solution is given by $u(x,t) = a\exp(bt-cx)$.

In our computation, we take $\varepsilon = 1.0, \gamma = 0.001, a = 1.0, b = 0.1, h = 0.1, k = 0.001$, so that $C_r = 0.01, s = 0.0001, P_e = 100$. The results are computed for different time levels. The L_∞ and L_2 errors are reported in Table-4. The results obtained from Method-I, are more accurate in comparison to Method-II. In [7], Huai-Huo Cao et. al computed L_2 error with $h = 0.1$ and $k = 0.0001$ was $3.34E-09$ at $T = 1.0$. Hence, Method-I, produce more accurate result than [7] with $h = 0.1$ and $k = 0.001$. In Fig-1, Numerical solutions at $t = 1.0$ are also depicted graphically which is similar to [7]. Both presented methods work well for higher value of $P_e = 100$ for a long time $T = 20$.

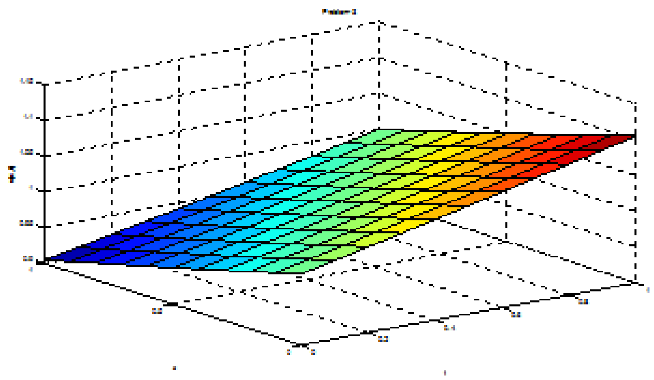

Figure 1: Approximate Solutions with $(h = 0.1, k = 0.001)$ at $T = 1.0$.

Example 4 We consider the following equation [7]

$$\frac{\partial u}{\partial t} + \varepsilon\frac{\partial u}{\partial x} = \gamma\frac{\partial^2 u}{\partial x^2}, 0 \le x \le 2, 0 \le t \le 2$$

with initial condition $\varphi(x) = \sin(x)$, and boundary conditions are

$(\dfrac{\partial u}{\partial t})_{(0,t)} = \exp{(-\gamma t)}\cos(\varepsilon t), (\dfrac{\partial u}{\partial t})_{(2,t)} = \exp{(-\gamma t)}\cos(2-\varepsilon t)$.

The exact solution is given by $u(x,t) = \exp(-\gamma t)\sin(x - \varepsilon t)$.

In our computation, we take $\varepsilon = 1.0, \gamma = 0.1, h = 0.1, k = 0.01$, so that $C_r = 0.1, s = 0.1, P_e = 1$. The results are computed for different time levels. The L_∞ and L_2 errors are reported in Table-5.The L_∞ and L_2 errors obtained from Method-I, are less in comparison to Method-II. In this problem, both methods work well for small value of $P_e = 1$ for a long time $T = 20$. In this problem, numerical results shown in Huai-Huo Cao et. al [7] are more accurate with $h = 0.1$ and $k = 0.0001$.

Table 5: L_∞ and L_2 errors (Example-4)

T	Method-I	Method-I	Method-II	Method-II
	L_∞	L_2	L_∞	L_2
0.2	$1.60E-05$	$1.48E-05$	$2.04E-04$	$8.64E-05$
0.4	$3.08E-05$	$2.56E-05$	$5.96E-04$	$2.63E-04$
0.6	$4.26E-05$	$3.25E-05$	$1.14E-03$	$5.52E-04$
0.8	$4.99E-05$	$3.65E-05$	$1.82E-03$	$9.56E-04$
1.0	$5.16E-05$	**4.01E − 05**	$2.59E-03$	$1.47E-03$
5.0	$1.37E-04$	$1.27E-04$	$7.88E-03$	$8.90E-03$
10.0	$1.40E-04$	$1.68E-04$	$6.72E-03$	$8.81E-03$
20.0	$7.47E-05$	$9.47E-05$	$4.66E-03$	$6.02E-03$
Cao et. al [7] ($h = 0.1, k = 0.0001$) $T = 1.0$		**2.82E − 06**		

Example 5 We consider the following equation [9]

$$\frac{\partial u}{\partial t} + \varepsilon\frac{\partial u}{\partial x} = \gamma\frac{\partial^2 u}{\partial x^2}, 0 \le x \le 1, 0 \le t \le T$$

with $\varepsilon = 1.0, \gamma = 1.0$ and the following initial condition $\varphi(x) = \dfrac{1}{\sqrt{s}}\exp\left(-50\dfrac{x^2}{s}\right)$, $s = 1.0$, and boundary conditions are $\left(\dfrac{\partial u}{\partial t}\right)_{(0,t)} = \dfrac{100t}{s}u(0,t), \left(\dfrac{\partial u}{\partial t}\right)_{(1,t)} = \dfrac{-100(1-t)}{s}u(1,t)$.

The exact solution is given by $u(x,t) = \dfrac{1}{\sqrt{s}}\exp\left(-50\dfrac{(x-t)^2}{s}\right)$, $s = (1 + 200\gamma t)$.

In our computation, we take $\varepsilon = 1.0, \gamma = 1.0, h = 0.1, k = 0.0001$, so that $C_r = 0.001, s = 0.01, P_e = 0.1$. The results are computed for different time levels. The L_∞ and L_2 errors are reported in Table-6. The L_∞ and L_2 errors obtained from Method-I and Method-II are approximately similar. In this problem, both methods work well for smaller value of $P_e = 0.1$ at $T = 1$.

8. Conclusions

In this work, the convection-diffusion equations was dealt by using Method-I in which time derivative is discretized by using Crank Nicolson scheme and apply cubic B-splines for spatial derivative. The stability of this method is discussed by using von Neumann stability and shown that it is unconditionally stable. This method works very well for different values of Péclet number, $0.1 \le P_e \le 100$ and time $T \le 20$. In Method- II, we apply cubic B-splines for spatial variable and its derivatives, then the

Table 6: L_∞ and L_2 errors (Example-5)

T	Method-I	Method-I	Method-II	Method-II
	L_∞	L_2	L_∞	L_2
0.2	$2.73E-03$	$1.77E-03$	$2.74E-03$	$1.77E-03$
0.4	$2.17E-03$	$1.92E-03$	$2.18E-03$	$1.92E-03$
0.6	$2.11E-03$	$1.98E-03$	$2.11E-03$	$1.99E-03$
0.8	$2.11E-03$	$2.00E-03$	$2.12E-03$	$2.01E-03$
1.0	$2.12E-03$	$2.01E-03$	$2.13E-03$	$2.02E-03$

resulting system of first order ordinary differential equations are solved by using SSP-RK54 scheme. Method-II also worked successfully and gives reliable but less accurate solutions. The implementations of algorithms for these methods are very easy and economical. The accuracy of the numerical solutions indicates that the presented methods are well suited for convection-diffusion equations with Neumann's boundary conditions.

Acknowledgements

One of the authors R.K. Jain thankfully acknowledges the sponsorship under QIP, provided by Technical Education and Training Department, Bhopal (M.P.), India.

References

[1] A. Mohebbi and M. Dehghan, High-order compact solution of the one-dimensional heat and advection-diffusion equations, Appl. Math. Model. 34(2010) 3071-3084.
[2] D.K. Salkuyeh, On the finite difference approximation to the convection-diffusion equation, Appl. Math. Comput., 179(2006) 79-86.
[3] H. Karahan, , Unconditional stable explicit finite difference technique for the advection-diffusion equation using spreadsheets, Adv. Eng. Software 38(2007) 80-86.
[4] H. Karahan, Implicit finite difference techniques for the advection-diffusion equation using spreadsheets, Adv. Eng. Software 37(2006) 601-608.
[5] H.N.A. Ismail, E.M.E. Elbarbary, G.S.E. Salem, Restrictive Taylors approximation for solving convection-diffusion equation, Appl. Math. Comput. 147(2004) 355-363.
[6] H.F. Ding and Y.X. Zhang, A new difference scheme with high accuracy and absolute stability for solving convection-diffusion equations, J. Comput. Appl. Math. 230(2009) 600-606.
[7] Huai-Huo Cao, Li-Bin Liu, Yong Zhang and Sheng-mao Fu, A fourth-order method of the convection-diffusion equations with Neumann boundary conditions, Appl. Math. Comput. 217(2011) 9133-9141.
[8] M.M. Chawla, M.A. Al-Zanaidi and D.J. Evans, Generalized trapezoidal formulas for convection-diffusion equations, Int. J. Comput. Math. 72(1999) 141-154.
[9] M.M. Chawla, M.A. Al-Zanaidi and M.G. Al-Aslab, Extended one step time-integration schemes for convection-diffusion equations, Comput. Math. Appl. 39(2000) 71-84.
[10] M. Dehghan, Weighted finite difference techniques for the one-dimensional advection-diffusion equation, Appl. Math. Comput. 147(2004) 307-319.
[11] R.J. Spiteri and S.J. Ruuth, A new class of optimal high-order strong-stability-preserving time-stepping schemes, SIAM J. Numer. Anal. 40 (2002) 469-491.
[12] R.C. Mittal and Geeta Arora, Numerical solution of the coupled viscous Burgers equation, Commun Nonlinear Sci Numer Simulat, 16(2011) 1304-1313.
[13] Rizwan-Uddin, A second-order space and time nodal method for the one-dimensional convection-diffusion equation, Computers & Fluids 26(3) (1997) 233-247.
[14] S. Karaa and J. Zhang, High order ADI method for solving unsteady convection-diffusion problems, J. Comput. Phys. 198(2004) 1-9.

[15] X.F. Feng and Z.F. Tian, Alternating group explicit method with exponential-type for the diffusion-convection equation, Int. J. Comput. Math. 83(2006) 765-775.
[16] Weizhong Dai, A new accurate finite difference scheme for Neumann (insulated) boundary condition of heat condution, International Journal of Thermal Sciences 49(2010) 571-579.