Numerical Solutions of Some Parabolic Partial Differential Equations Using Cubic B-Spline Collocation Method

Thesis submitted in partial fulfillment of the requirements

for the award of the degree of

Masters of Science

in

Mathematics and Computing

submitted by

Mandeep Kaur

Roll No: 301103008

under

the guidance of

Dr. Ram Jiwari

to the

School of Mathematics and Computer Applications
Thapar University
Patiala- 147004 (Punjab)
INDIA

Contents

CERTIFICATE

I hereby certify that the work which is being presented in the thesis entitled "Numerical Solutions of Some Parabolic Partial Differential Equations Using Cubic B-Spline Collocation Method" which is being submitted for the award of degree of master of Science, School of Mathematics and Computer Applications, Thapar University, Patiala is an authentic record of my own work carried out under the supervision of Dr. Ram Jiwari.

The matter presented in the thesis has not been submitted for the award of any other degree of this or any other university.

Mandeep Kaur

Mandeed Kaur

This is to certify that the above statement made by the candidate is correct and true to the best of my knowledge.

19/6/13

(Dr. Ram Jiwari)
SMCA, Thapar University
Patiala

Countersigned by

1.7.13

Dr. Rajesh Kumar
(Professor& Head)
School of Mathematics & Computer Applications
Thapar University, Patiala

2/7/13

Dr. S.K Mohapatra
Dean of Academic Affairs
Thapar Uniersity, Patiala

ACKNOWLEDGEMENT

It gives me immense pleasure to acknowledge my sincere gratitude to my academic supervisor, Dr. Ram Jiwari School of Mathematics and Computer Aplplications Thapar University, Patiala, for his constant help, encouragement and support throughout the course of this work.

I would like to extend my special thanks to Dr. Rajesh Kumar, Professor and Head, School of Mathematics and Computer Aplplications Thapar University, Patiala, for providing help and necessary facilities in the department and directly or indirectly encouraged me to work harder during the whole course.

I am very grateful to my close friend Harwinder Kaur for her kind support.

Mandeep Kaur.

(Mandeep Kaur)

Patiala
June 17, 2013.

ABSTRACT

Chapter 1 is introductory in nature. Besides stating some numerical techniques like Finite Difference methods, Finite Element method, Finite Volume method and methods of weighted residuals it gives an introduction to B-Spline and existing literature review.

In chapter 2, we consider the one dimensional heat equation

$$u_t = \alpha^2 u_{xx} \qquad , 0 \le x \le L \qquad t > 0$$

This problem is one of the well-known second order linear partial differential equation [1, 2]. It shows that heat equation describes irreversible process and makes a distance between the previous and next steps. Such equations arise very often in various applications of science and engineering describing the variation of temperature (or heat distribution) in a given region over some time [3]. It can be expressed as the heat flow in the rod with diffusion $\alpha^2 u_{xx}$ along the rod where the coefficient α is the thermal diffusivity of the rod and L is the length of the rod. In this model, the flow of the heat in one-dimension that is insulated everywhere except at the two end points.

In this chapter, cubic B-spline method is proposed for the numerical solutions of one dimensional heat equation. Two test examples are considered to test the accuracy and efficiency of the method. The absolute, L_2 space and Root Mean Square errors are calculated for the both examples.

In chapter 3, we consider the one dimensional advection-diffusion equation

$$\frac{\partial u}{\partial t} = D\frac{\partial^2 u}{\partial x^2} - C\frac{\partial u}{\partial x}, \quad 0 < x < L \ , \qquad 0 < t \le T$$

The equation described the mathematical model of transport, diffusion processes, mathematical modeling of heat transport, pollutants and suspended matter in groundwater.

In this chapter, cubic B-spline method is proposed for the numerical solutions of one dimensional advection-diffusion equation. Two test examples are considered to test the accuracy and efficiency of the method. The absolute, L_2 space and Root Mean Square errors are calculated for the both examples.

Chapter 1

Introduction

1.1 Numerical Solution of Differential Equations

Differential equations (PDE/ODEs) form the basis of many mathematical models of physical, chemical and biological phenomena, and more recently their use has spread into economics, financial forecasting, image processing and other fields. It is not easy to get analytical solution treatment of these equations, so, to investigate the predictions of PDE models of such phenomena it is often necessary to approximate their solution numerically. In most cases, the approximate solution is represented by functional values at certain discrete points (grid points or mesh points). There seems a bridge between the derivatives in the PDE and the functional values at the grid points. The numerical technique is such a bridge, and the corresponding approximate solution is termed the numerical solution. Currently, there are a number of numerical methods available for finding the numerical solutions of differential equations. . Among them, the finite difference (FD), finite element (FE), and finite volume (FV) methods fall under the category of low order methods, whereas spectral and pseudo spectral methods are considered global methods. Sometimes the latter two methods are considered as subsets of the method of weighted residuals.

1.1.1 Finite Difference Methods

Finite difference methods are widely dominant in the numerical solution of DEs and their application. The finite difference (FD) methods are based on the Taylor series expansion or the polynomial approximation. A finite differential method proceeds by replacing the derivatives in

the differential equations by finite difference approximation. This gives a large algebraic system of equations to be solved in place of the differential equation, which can be easily solved on computer. That is, the partial derivatives in differential equations are written in terms of discrete quantities of dependent and independent variables, resulting in simultaneous algebraic equations with all unknowns prescribed at discrete mesh points or grid points for the entire domain That is, the partial derivatives in PDEs are written in terms of discrete quantities of dependent and independent variables, resulting in simultaneous algebraic equations with all unknowns prescribed at discrete mesh points or grid points for the entire domain. Appropriate types of differencing schemes and suitable methods of solution are chosen in different applications. For example, in fluid dynamics applications, depending upon the particular physics of the flows, which may include in viscid, viscous, incompressible, compressible, irrigational, rotational, laminar, turbulent, supersonic, or hypersonic flows, finite difference schemes are written to conform to these different physical phenomena. The formulation of FD methods in one dimensional is simple but for multidimensional problems, meshes must be structured in either two or three dimensions. Curved meshes must be transformed into orthogonal Cartesian meshes. The challenge in analyzing finite difference methods for new classes of problems is often to find an appropriate definition of stability that allow one to prove convergence and to estimate the error in approximation.

Finite difference methods discredited the governing PDE directly using their strong form. Although it is most straight forward way to obtain the discrete system equations, but it is difficult to handle the typical boundary conditions. For a problem domain with complex geometry, the discretization of the geometry and the application of the natural and essential boundary

conditions can seldom be done automatically by a computer program with no human involvement.

1.1.2 Finite Element Method

Another most popular method is Finite Element method. Finite element method (FEM) represents a powerful and general class of techniques for the approximate solution of partial differential equations. The basic idea in the FEM is to find the solution of a complicated problem by replacing it by a simpler one. Since the actual problem is replaced by a simpler one in finding the solution, we will be able to find only an approximate solution rather than the exact solution. This method is mostly used for the accurate solution of complex engineering problems with abundant software available commercially. Finite Element method was first developed in 1956 for the analysis of aircraft structural problems. Thereafter, within a decade, the potentialities of the method for the solution of different types of applied science and engineering problems were recognized. Over the years, the FEM technique has been so well established that today it is considered to be one of the best methods for solving a wide variety of practical problems efficiently .It has been applied to a number of physical problems, where the governing differential equations are available. In Finite Element method the domain is divided into a finite number of sub domains called elements and nodes are located at predetermined locations around the elements boundary. The elements, along with the nodes, form the mesh, which can be refined to provide any level of accuracy desired.

1.1.3 Finite Volume Method

The finite volume method is a discretizations method for the approximation of single or a system of differential equation. The Finite Volume method has been extensively used in

several engineering fields such as fluid mechanics, heat and mass transfer or petroleum engineering. Some of the most features of finite volume method are similar to that element method. As in the finite element method a mesh is constructed, which consists in a partition of the domain where the space variable lives. The elements of the mesh are called control volumes. The integration of the PDE over each control volume results in a balance equation. The set of balance equations is then discretized with respect to a set of discrete unknowns. The main issue is the discretization of the fluxes at the boundaries of each control volume: in order for the FVM to be efficient, the numerical fluxes are generally *conservative*, *i.e.* the flux entering a control volume from its neighbour must be the opposite of the one entering the neighbour from the control volume, *consistent i.e.* the numerical flux of a regular function interpolation tends to the Continuous flux as the mesh size vanishes.

1.1.4 Differential Quadrature Method

The differential quadrature method (DQM) is a higher order numerical technique for solving partial differential equations. In the nineteen century, most of the numerical simulations of engineering problems can be carried out by the low order FD, FE, and FV methods using a large number of grid points. In some practical applications, however, numerical solutions of PDEs are required at only a few specified points in the physical domain. To achieve an acceptable degree of accuracy, low order methods still require the use of a large number of grid points to obtain accurate solutions at these specified points. In seeking an efficient discretization technique to obtain accurate numerical solutions using a considerably small number of grid points, Richard Bellman and his associates [35] introduced the method of differential quadrature in the early1970s. The DQM, akin to the conventional integral quadrature method, approximates the

partial derivative of a function at any location by a linear summation of all the function values along a mesh line. The key procedure in the differential quadrature application lies in the determination of the weighting coefficients. Initially, Bellman and his associates proposed two methods to compute the weighting coefficients for the first order derivative. The first method is based on an ill-conditioned algebraic equation system. The second method uses a simple algebraic formulation, but the coordinates of the grid points are fixed by the roots of the shifted Legendre polynomial. In earlier applications of the DQM, Bellman's first method was usually used because it allows the use of an arbitrary grid point distribution. However, since the algebraic equation system of this method is ill-conditioned, the number of the grid points usually used is less than 13. This drawback limits the application of the DQM.

The DQM and its applications were rapidly developed after the late 1980s, thanks to the innovative work in the computation of the weighting coefficients by researchers [36-40]. As a result, the DQM has emerged as a powerful numerical discretization tool in the past decade. As compared to the conventional low order finite difference and finite element methods, the DQM can obtain very accurate numerical results using a considerably smaller number of grid points and hence requiring relatively little computational effort. Jiwari et al. [41-55] have proposed some numerical techniques by using differential quadrature methods for some nonlinear physical problems.

1.1.5 Method of Weighted Residual

The methods of weighted residuals are the approximate methods which determine the solution of the differential equation in the form of functions which are closed in some sense to the exact solution. Consider a differential equation

$$l(u) = 0 \tag{1.1}$$

With initial condition, $\mathrm{I}(u)=0$ and boundary condition $S(u)=0$. The solution of differential equation $U(x)=0$ is approximated by a finite series of functions $\phi_k(x)$ as follows:

$$U(x)=U_0(x)+\sum_{k=1}^{N} a_k \phi_k(x) \tag{1.2}$$

where $\phi_k(x)$ are the basis or trial functions, *k a* are the coefficients to be determined that satisfy the differential equation, and *N* are the number of functions. The form of $U_0(x)$ is chosen to satisfy the boundary and the initial conditions exactly. There is another approach in which exact solutions of the differential equation are known and these are added together to satisfy the boundary conditions approximately. It is also possible to formulate a method in which the differential equation and the boundary conditions are satisfied approximately.

In general, the approximate solution does not satisfy the partial differential equation exactly, and substituting its value results in a residual, *R,*

$$R(x,a_1,a_2,...,a_N)=\ell(U(x)) \tag{1.3}$$

This in turn is minimized in some sense. For a given N, $a_k's$ are chosen by requiring that an integration of the weighted residual over the domain is zero. Thus

$$\langle W_k(x), R\rangle = 0 \tag{1.4}$$

By letting $k=1,2,...N$ a system of equations involving only $a_k's$ is obtained. For unsteady partial differential equation this would be a system of ordinary differential equations, for steady problems a system of algebraic equations obtained. Different choices of $W_k(x)$ give rise to the different methods within the class. Some of these methods are:

1.1.5.1 Galerkin Method

One of the most important weighted residual methods was invented by the Russian mathematician Boris Grigoryevich Galerkin. In the Galerkin method the weighting functions are chosen to be

$$W_k(x) = \phi_k(x) \tag{1.5}$$

i.e., the weighting functions are from the same family as the trial function in equation (1.2). Thus the residual becomes orthogonal to the space spanned by the trial functions.

In traditional Galerkin method each of the trial functions should satisfy the boundary condition but in spectral Tau method the trail functions need not satisfy the boundary condition instead, a supplementary set of equations is used to apply the boundary condition. A generalization of Galerkin method is Petrov-Galerkin method, in which, the weighting functions are different from trial functions.

1.1.5.2 Sub-domain Method

This method can be considered a modification of the collocation method. The idea is to force the weighted residual to zero not just at fixed points in the domain, but over various subsections of the domain. To accomplish this, the weight functions are set to unity, and the integral over the entire domain is broken into a number of sub domains sufficient to evaluate all unknown parameters.

1.1.5.3 Least Square Method

The basic idea of Least-Square is that the residual is minimized in a certain norm. The inner product of the governing equations is constructed, which are then differentiated with respect to

the nodal values of the variables. A general Least-Square formulation is the following minimization problem:

$$S = \min \int_X R(x, a_1, a_2, ..., a_N)^2 dx \tag{1.6}$$

In order to achieve a minimum of this scalar function, the derivatives of *S* with respect to all unknown parameters must be zero. That is,

$$\frac{\partial S}{\partial a_k} = 2 \int_X R(x) \frac{\partial R}{\partial a_k} dx = 0 \tag{1.7}$$

or

$$\int_X R(x) \frac{\partial R}{\partial a_k} dx = 0.$$

1.1.6 Collocation Method

Collocation method was developed to seek numerical solution of an initial or boundary value problem in form of linear combination of coordinate function with linear coefficients. In this method, we approximate a function by passing a polynomial through values of the function at selected points. The selected points are known as collocation points. Consider a differential equation

$$f(x, u, u_x) = 0 \tag{1.8}$$

To be solved by collocation method .Let the function is defined in the domain $[a, b]$with boundary conditions given by

$$u(a) = g_0 \,,\; u(b) = g_1 \tag{1.9}$$

The method begins with a proper choice of basic functions $\{\phi_1, \phi_2, .., \phi_n\}$ and a set of points $a = x_1 < x_2 <, .., < x_n = b$ called nodes or collocation points in the domain $[a, b]$.

The approximate solution can be written in the form

$$U = \sum_{i=1}^{n} c_i \, \phi_i(x) \tag{1.10}$$

Here n can be thought of as the quality parameter, as n increases, the error in the approximation must reduces. This method requires that this approximation satisfies the given differential equation at each of the nodes and also satisfies the boundary conditions.

For this, the residual must be zero at selected points .the residual is defined as

$$r(x, u, u_x) = f(x, u, u_x) - f(x, U, U_x) \tag{1.11}$$

where U is the approximate solution. The choice of $\phi_i's$ can vary depending on the problem.

In the last few years another numerical technique has been increasingly used to solve mathematical models in engineering research, the B-spline Collocation Method. The B-spline Collocation Method has a few distinct advantages over the Finite Element and Finite Difference Methods. The advantage over the Finite Difference Method is that the B-spline Collocation Method provides a piecewise-continuous, closed form solution. An advantage over the Finite Element Method is that the B-spline collocation method procedure is simpler and easy to apply many problems involving differential equations.

Our aim is to explore the implementation of collocation method using B-spline basis function to solve initial and boundary value problems. We have used the collocation method with

B-spline basis functions of third, fourth and fifth degree to find numerical solutions of some linear and non-linear equations. This chapter provides description of concept of spline and B-spline basis functions followed by the methodology adopted to solve the initial and boundary value problems using B-spline collocation method.

The term 'spline' is derived from the flexible device used by shipbuilder &draftsmen to draw a curve through pre-assigned points (knots) in such a way that not only the curve is continuous but also its slope and curvature are continuous functions .Draftsman attach the wooden or metal strip with weights called ducks, which can be adjusted to keep the strip in required shape. So weights are attached with the strip to keep it in the required shape.

1.2 Idea of Spline

In order to resolve the problem of working with higher degree polynomials the idea of piecewise polynomial come into existence .Instead of using polynomial for the entire domain ,the function can be approximated by several polynomials defined over the sub-domains.

A polynomial which is presented over a certain domain by means of several polynomials defined over its sub-domains called a piecewise polynomial. The piecewise polynomial approximation allows us to construct highly accurate approximations, but because some approximation functions are not smooth at the point connecting separate piecewise polynomial approximation. Sometimes, while the polynomial is continuous, it may not be continuously differentiable on the interval of approximation and the graph of the interpolant may not be smooth. Splines are an attempt to solve this problem.

So the basic idea of splines are to construct a piecewise polynomial approximation that not only interpolate the given data or function values but it is also smooth i.e. it must be

continuously differentiable to some degree. A spline is sufficiently smooth piecewise polynomial function.

Let us consider a uniform partition $x_0 < x_1 < ... < x_n$ the domain $[a,b]$ *with* $x_0 = a$ *and* $x_n = b$ the abscissas x_i are called the knots. A function $s(x)$ is called the spline of degree k if it is a k^{th} degree polynomial $p(x)$ in each of that interval $[x_i - x_{i+1}], i = 0,...,n-1$ with the property the $p(x)$ and its first $(k-1)$ derivative are continuous in the domain $[x_0, x_n]$.

Thus a spline $s(x)$ on domain $[x_0, x_n]$ can be defined as

$$s(x) = \sum_{i=1}^{n} p_i(x) \tag{1.12}$$

Here $p(x)$ is a k^{th} degree polynomial in each partition.

Since each segment (partition) is defined by k^{th} degree polynomial, the number of coefficients in each segment is $(k+1)$ and there are n segments.

Thus total number of coefficients becomes $n(k+1)$. So therefore to define a spline,

Number of required equations= $n(k+1)$ (1.13)

Using the continuity property of splines we have for a k^{th} degree spline the polynomial $p(x)$ with its $(k-1)$ derivatives is continuous at all interior knots, we can write

0	$p(x_i) = p(x_{i+1})$
1	$p'(x_i) = p'(x_{i+1})$
:	
:	
$k-1$	$p^{k-1}(x_i) = p^{k-1}(x_{i+1})$

where $i = 1,2,...,n-1$. (i.e. internal knots).

As number of internal knots is $(n-1)$. So each of above relation provides $(n-1)$ equations.

Thus, the total number of conditions is

$$(n-1)\{1+(k-1)\}=(n-1)k \tag{1.14}$$

Additional conditions

$$n(k+1)-(n-1)k=(n+k) \tag{1.15}$$

These additional conditions can be obtained from the boundary conditions.

1.3 B-Spline

"B-spline is a spline function that has minimal support with respect to given degree, smoothness and domain partition" The first reference to the world B-spline function ('B' refers to basis) in the field of mathematics was given by Schonberg in 1946, who described it as a smooth piecewise polynomial approximation and is short for basis spline. A B –spline is defined as a spline function that has minimal support with respect to a given degree, smoothness, and domain partition.

The underlying core of the B-spline is its basis function. The defining feature of the basis function is knot sequence x_i. Let X be a set of N+1 non decreasing real numbers. $x_0 \le x_1 \le x_2 \le ... \le x_{N-1} \le x_N$ .Here $x_i's$ are called knots , the set X is the knot sequence which represents the active area of real numbers line that defines the B-spline basis ,and the half –open interval $[x_i, x_{i+1})$ the i^{th} knot span. If the knots are equally spaced $(i.e., x_{i+1}-x_i)$ is a constant for $0 \le i \le N-1)$, the knots vectors or the knot sequence is said to be uniform; otherwise, it is called non-uniform. Each B-spline function of degree k covers $k+1$ knots or k intervals.

In early 1970's, a recurrence relation was independently established by Cox and Boor for the purpose of computing B-spline basis function. By applying the Leibniz 'theorem, Boor was able to drive the following formula for m^{th} B-spline basis function of k^{th} degree in a recursive manner as follow:

$$B_{m,k}(x) = V_{m,k} B_{m,k-1}(x) + (1 - V_{m+1,k}) B_{m+1,k-1}(x) \tag{1.16}$$

where ,

$$V_{m,k} = \left(\frac{x - x_m}{x_{m+k} - x_m} \right)$$

This formula is known as Cox de- Boor recursion formula. Here $B_{m,k}(x)$ define a m^{th} B-spline basis function of degree $k, \{x_i\}$ is non –decreasing set of real numbers also called as the knot sequence and x is a parameter variable.

The recurrence relation starts with the first degree B-splines and builds the functions of successively higher orders. For degree $k \geq 1$ basis function $B_{m,k}(x)$ is a linear combination of two $(k-1)^{th}$ degree basis function.

1.3.1 Zero Degree B-Spline

For degree k =0, the basis function is just a step function. Thus, the zero degree B-spline is one of the simplest B-spline basis function and is given as

$$B_{m.,0} = \begin{cases} 1, & x \in [x_m, x_{m+1}) \\ 0, & otherwise \end{cases} \tag{1.17}$$

Thus, a zero degree B-spline is equal to zero at all points expects on the half open interval $[x_m, x_{m+1})$

1.3.2 First Degree B-Spline

The expression for the first degree B- spline, also called as linear B-spline can be obtained using the Cox and Boor recursion formula given by (1.16)

Put k=1 in (1.16) and use the definition of zero degree B -spline. The formula of the first degree B-spline basis function can be given as

$$B_{m,1} = \begin{cases} \dfrac{x - x_m}{x_{m+1} - x_m} & x \in [x_m, x_{m+1}) \\ \dfrac{x_{m+2} - x}{x_{m+2} - x_{m+1}} & x \in [x_{m+1}, x_{m+2}) \\ 0 & otherwise \end{cases} \tag{1.18}$$

The first degree B-spline is like a HAT or tent function which is non-zero for two knot spans $[x_m, x_{m+1})$ and $[x_{m+1}, x_{m+2})$

1.3.3 Second Degree (Quadratic) B-Spline

The formula for the second degree B-spline also called as quadratic B-spline can be obtained by using the formula of linear B-spline basis function (1.17) and de-Boor recursion formula for k=2

The formula for the second degree B-spline can be given as

$$B_{m,2}(x)=\begin{cases}\dfrac{(x-x_m)^2}{(x_{m-2}-x_m)(x_{m+1}-x_m)} & x\in[x_m,x_{m+1})\\ \dfrac{(x-x_m)(x_{m+2}-x)}{(x_{m+2}-x_m)(x_{m+2}-x_{m+1})}+\dfrac{(x_{m+3}-x)(x-x_{m+1})}{(x_{m+3}-x_{m+1})(x_{m+2}-x_{m+1})} & x\in[x_{m+1},x_{m+2})\\ \dfrac{(x_{m+3}-x)^2}{(x_{m+3}-x_{m+1})(x_{m+3}-x_{m+2})} & x\in[x_{m+2},x_{m+3})\\ 0 & otherwise\end{cases} \tag{1.19}$$

The second degree B-spline basis function is non-zero between three knot spans.

1.4 Collocation Method Using B -Spline Basis Function

The collocation method together with B-spline approximation represents an economical alternative, since it is based on evaluating the accuracy of a differential equation at a finite set of collocation points .the issue that effect the effectiveness and accuracy of B-spline collocation method for solving differential equations include, which points to use for collocation, what degree of B-spline to use and what level of continuity to maintain.

Let us consider

$$a=x_0<x_1<...,<x_{n-1}<x_n=b$$

as a uniform partition of the solution domain $[a,b]$by the knots $x_m$with step-length $h=x_{m+1}-x_m$ where $m=0,1,...,n-1$.Now to find the solution of differential equation using collocation method with B-spline basis function, the approximate solution $U(x)$can be assumed as a linear combination of basis functions as

$$U(x,t)=\sum_{j=m-k+2}^{m+k-2}c_j(t)B_j(x) \tag{1.20}$$

Where, k is the degree of the B-spline, m is the number of nodes and c_m are the unknown constants to be determined from the boundary conditions and collocation from of the differential equation.

Let us now derive the approximate formula with the basis function of third, fourth and fifth degree B-spline.

1.4.1 Third Degree B-Spline

The third degree B-spline called as cubic B-spline basis function is given by formula

$$B_{m,3}(x)=\frac{1}{h^3}\begin{cases}(x-x_{m-2})^3 & x\in[x_{m-2},x_{m-1})\\ (x-x_{m-2})^3-4(x-x_{m-1})^3 & x\in[x_{m-1},x_m)\\ (x_{m+2}-x)^3-4(x_{m+1}-x)^3 & x\in[x_m,x_{m+1})\\ (x_{m+2}-x)^3 & x\in[x_{m+1},x_{m+2})\\ 0 & otherwise\end{cases} \tag{1.21}$$

This definition of cubic B-spline basis functions is given with x_m as the middle knot and equal number of knots on the two sides. The third degree B-spline is non-zero on four knot spans.

From the definition given by (1.20), the value of $B_{m,k}(x)$ at the nodal points can be obtained. On differentiating with respect to x we can obtain the value of first and second derivatives of $B_{m,k}(x)$ the value of $B_{m,k}(x)$ and its first derivatives at the nodal points can be tabulated as in Table 1.1.

Table 1.1: value of $B_{m,k}(x)$ and its first derivatives at the nodal points

	x_{m-2}	x_{m-1}	x_m	x_{m+1}	x_{m+2}
$B_{m,3}(x)$	0	1	4	1	0
$B'_{m,3}(x)$	0	$\frac{3}{h}$	0	$\frac{-3}{h}$	0
$B''_{m,3}(x)$	0	$\frac{6}{h^2}$	$\frac{-12}{h^2}$	$\frac{6}{h^2}$	0

Now substitute $k=3$ in (1.20), we get

$$U(x,t)=\sum_{j=m-3+2}^{m+3-2} c_j(t)B_j(x)$$

So the approximate solution can be written as

$$U(x,t)=\sum_{j=m-1}^{m+1} c_j(t)B_j(x) \qquad (1.22)$$

Without loss of generality, equation (1.21) can be expressed as

$$U(x_m,t)=c_{m-1}B_{m-1}(x_m)+c_mB_m(x_m)+c_{m+1}B_{m+1}(x_m)$$

Or

$$U(x_m,t)=c_{m-1}B_m(x_{m+1})+c_mB_m(x_m)+c_{m+1}B_m(x_{m-1}) \qquad (1.23)$$

As we are evaluating for cubic spline, so (1.22) can be written as

$$U(x_m,t)=c_{m-1}B_{m,3}(x_{m+1})+c_mB_{m,3}(x_m)+c_{m+1}B_{m,3}(x_{m-1}) \qquad (1.24)$$

From here

$$U'(x_m,t)=c_{m-1}B'_{m,3}(x_{m+1})+c_mB'_{m,3}(x_m)+c_{m+1}B'_{m,3}(x_{m-1})$$

$$U''(x_m,t) = c_{m-1}B''_{m,3}(x_{m+1}) + c_m B''_{m,3}(x_m) + c_{m+1}B''_{m,3}(x_{m-1})$$

On substituting values of $B_{m,3}(x)$ at the knots from Table 1.1, we get

$$U(x_m,t) = c_{m-1} + 4c_m + c_{m+1}$$

$$U'(x_m,t) = \frac{3}{h}(c_{m+1} - c_{m-1}) \tag{1.25}$$

$$U''(x_m,t) = \frac{6}{h^2}(c_{m-1} - 2c_m + c_{m+1})$$

1.4.1 Fourth Degree B-Spline

The B-spline basis function of fourth degree also called as quartic B-spline is given by

$$B_{m,4}(x) = \frac{1}{h^4}\begin{cases} (x-x_{m-2})^4 & x \in [x_{m-2}, x_{m-1}) \\ (x-x_{m-2})^4 - 5(x-x_{m-1})^4 & x \in [x_{m-1}, x_m) \\ (x-x_{m-2})^4 - 5(x-x_{m-1})^4 + 10(x-x_m)^4 & x \in [x_m, x_{m+1}) \\ (x_{m+3}-x)^4 - 5(x_{m+2}-x)^4 & x \in [x_{m+1}, x_{m+2}) \\ (x_{m+3}-x)^4 & x \in [x_{m+2}, x_{m+3}) \\ 0 & otherwise \end{cases} \tag{1.26}$$

This basis function is non zero on five knot spans. From definition given by (1.25) the value of $B_{m,4}(x)$ at nodal points can be obtained on differentiating with respect to x we can obtain the value of its derivatives in similar way. The value of $B_{m,4}(x)$ and its derivatives at the nodal points may be tabulated as in the table 1.2.

Table 1.2: values of $B_{m,4}(x)$ and its derivatives at nodal points

x_{m-2}	x_{m-1}	x_m	x_{m+1}	x_{m+2}	x_{m+3}

$B_{m,4}(x)$	0	1	11	11	1	0
$B'_{m,4}(x)$	0	$\frac{4}{h}$	$\frac{12}{h}$	$\frac{-12}{h}$	$\frac{-4}{h}$	0
$B''_{m,4}(x)$	0	$\frac{12}{h^2}$	$\frac{-12}{h^2}$	$\frac{-12}{h^2}$	$\frac{12}{h^2}$	0
$B'''_{m,4}(x)$	0	$\frac{24}{h^3}$	$\frac{-72}{h^3}$	$\frac{72}{h^3}$	$\frac{-24}{h^3}$	0

By substituting $k=4$ in (1.19), we get

$$U(x,t)=\sum_{j=m-4+2}^{m+4-2} c_j(t)B_j(x,t)$$

So the approximate solution can be written as

$$U(x,t)=\sum_{j=m-2}^{m+2} c_j(t)B_j(x,t) \qquad (1.27)$$

Equation (1.26) can be written as

$$U(x_m,t)=c_{m-2}B_{m-2}(x_m)+c_{m-1}B_{m-1}(x_m)+c_mB_m(x_m)+c_{m+1}B_{m+1}(x_m)+c_{m+2}B_{m+2}(x_m)$$

$$U(x_m,t)=c_{m-2}B_m(x_{m+2})+c_{m-1}B_m(x_{m+1})+c_mB_m(x_m)+c_{m+1}B_m(x_{m-1})+c_{m+2}B_m(x_{m-2})$$

As we are evaluating for quartic B-spline, so can be written as

$$U(x_m,t)=c_{m-2}B_{m,4}(x_{m+2})+c_{m-1}B_{m,4}(x_{m+1})+c_mB_{m,4}(x_m)+c_{m+1}B_{m,4}(x_{m-1})+c_{m+2}B_{m,4}(x_{m-2}) \qquad (1.28)$$

So

$$U'(x_m,t)=c_{m-2}B'_{m,4}(x_{m+2})+c_{m-1}B'_{m,4}(x_{m+1})+c_mB'_{m,4}(x_m)+c_{m+1}B'_{m,4}(x_{m-1})+c_{m+2}B'_{m,4}(x_{m-2})$$

On substituting the value of $B_{m,4}(x)$ at the nodes from Table 1.2 we get

$$U(x_m,t)=c_{m-2}+11c_{m-1}+11c_m+c_{m+1}$$

$$U'(x_m,t)=\frac{4}{h}(-c_{m-2}-3c_{m-1}+3c_m+c_{m+1})$$

$$U''(x_m,t)=\frac{12}{h^2}(c_{m-2}-c_{m-1}-c_m+c_{m+1}) \tag{1.29}$$

$$U'''(x_m,t)=\frac{24}{h^3}(-c_{m-2}+3c_{m-1}-3c_m+c_{m+1})$$

1.4.2 Fifth Degree B-spline

The fifth degree B-spline also called as quintic B-spline basis function is given by formula

$$B_{m,5}(x)=\frac{1}{h^5}\begin{cases}(x-x_{m-3})^5 & x\in[x_{m-3},x_{m-2})\\ (x-x_{m-3})^5-6(x-x_{m-2})^5 & x\in[x_{m-2},x_{m-1})\\ (x-x_{m-3})^5-6(x-x_{m-2})^5+15(x-x_{m-1})^5 & x\in[x_{m-1},x_m)\\ (x_{m+3}-x)^5-6(x_{m+2}-x)^5+15(x_{m+1}-x)^5 & x\in[x_m,x_{m+1})\\ (x_{m+3}-x)^5-6(x_{m+2}-x)^5 & x\in[x_{m+1},x_{m+2})\\ (x_{m+3}-x)^5 & x\in[x_{m+2},x_{m+3})\\ 0 & otherwise\end{cases} \tag{1.30}$$

This basis function is non zero on six knot spans. From definition given by (1.29) the value of $B_{m,5}(x)$ at the nodal points can be obtained and on differentiating with respect to x the value of its four derivatives can be obtained in similar way. The value of $B_{m,5}(x)$ and its derivatives at the nodal points may be tabulated as in the table 1.3.

Table 1.3: value of $B_{m,5}(x)$ for quintic b-spline and its derivatives at the nodal points.

	x_{m-3}	x_{m-2}	x_{m-1}	x_m	x_{m+1}	x_{m+2}	x_{m+3}
$B_{m,5}(x)$	0	1	26	66	26	1	0
$B'_{m,5}(x)$	0	$\frac{5}{h}$	$\frac{50}{h}$	0	$\frac{-50}{h}$	$\frac{-5}{h}$	0
$B''_{m,5}(x)$	0	$\frac{20}{h^2}$	$\frac{40}{h^2}$	$\frac{-120}{h^2}$	$\frac{40}{h^2}$	$\frac{20}{h^2}$	0
$B'''_{m,5}(x)$	0	$\frac{60}{h^3}$	$\frac{-120}{h^3}$	0	$\frac{120}{h^3}$	$\frac{-60}{h^3}$	0
$B^{iv}{}_{m,5}(x)$	0	$\frac{120}{h^4}$	$\frac{-480}{h^4}$	$\frac{720}{h^4}$	$\frac{-480}{h^4}$	$\frac{120}{h^4}$	0

Substitute $k=5$ in (1.19)

$$U(x,t)=\sum_{j=m-5+2}^{m+5-2} c_j(t)B_j(x)$$

We get the approximate solution

$$U(x,t)=\sum_{j=m-3}^{m+3} c_j(t)B_j(x) \tag{1.31}$$

Equation (1.31) can be expressed as

$$U(x_m,t)=c_{m-3}B_{m-3}(x_m)+c_{m-2}B_{m-2}(x_m)+c_{m-1}B_{m-1}(x_m)+c_mB_m(x_m)+c_{m+1}B_{m+1}(x_m)+c_{m+2}B_{m+2}(x_m)+c_{m+3}B_{m+3}(x_m)$$

Or

$$U(x_m,t)=c_{m-3}B_m(x_{m+3})+c_{m-2}B_m(x_{m+2})+c_{m-1}B_m(x_{m+1})+c_mB_m(x_m)+c_{m+1}B_m(x_{m-1})+c_{m+2}B_m(x_{m-2})+c_{m+3}B_m(x_{m-3})$$

As we are evaluating for quintic b-spline, so can be written as

$$U(x_m,t)=c_{m-3}B_{m,5}(x_{m+3})+c_{m-2}B_{m,5}(x_{m+2})+c_{m-1}B_{m,5}(x_{m+1})+c_mB_{m,5}(x_m)+c_{m+1}B_{m,5}(x_{m-1})+c_{m+2}B_{m,5}(x_{m-2})+c_{m+3}B_{m,5}(x_{m-3})$$

On substituting the values of $B_{m,5}(x)$ at the nodes from the Table 1.3, we get

$$U(x_m,t)=c_{m-2}+26c_{m-1}+66c_m+26c_{m+1}+c_{m+2}$$

$$U'(x_m,t)=\frac{5}{h}(-c_{m-2}-10c_{m-1}+10c_{m+1}+c_{m+2})$$

$$U''(x_m,t)=\frac{20}{h^2}(c_{m-2}+2c_{m-1}-6c_m+2c_{m+1}+c_{m+2}) \quad (1.32)$$

$$U'''(x_m,t)=\frac{60}{h^3}(-c_{m-2}+2c_{m-1}-2c_{m+1}+c_{m+2})$$

$$U^{iv}(x_m,t)=\frac{120}{h^4}(c_{m-2}-4c_{m-1}+6c_m-4c_{m+1}+c_{m+2})$$

1.5 Properties of B-Spline Basis Function

Some of the important properties of the B-spline basis functions are as follows:

1. $B_{m,k}(x)$ is a non-zero polynomial on $[x_m, x_{m+k+1})$ for degree $k \geq 0$.

2. On any span $[x_m, x_{m+1})$ at most k +1 basis functions of degree k are non-zero

$B_{m-k,k}(x), B_{m-k+1,k}(x), B_{m-k+2,k}(x), ..., and\ B_{m,k}(x)$

3. Non negativity

For all m, k and, x, $B_{m,k}(x)$is non-negative in the interval $[x_m, x_{m+1})$. The closed

interval is called the support of $B_{m,k}(x)$

4. Local knots

If x is outside the interval $[x_m, x_{m+1})$ then $B_{m,k}(x)=0$

Local support property indicates that each segment of B-spline curve is influenced by only k control points or each control points affects only k curve segment.

1.6 Organization of Thesis

In this thesis an attempt has been made to solve some Parabolic Partial differential equations by using cubic B-spline collocation methods. The chapter wise summary of thesis is as follows.

In chapter 2, we consider the one dimensional heat equation

$$u_t = \alpha^2 u_{xx} \qquad , 0 \le x \le L \qquad t > 0$$

This problem is one of the well-known second order linear partial differential equation [1,2]. It shows that heat equation describes irreversible process and makes a distance between the previous and next steps. Such equations arise very often in various applications of science and engineering describing the variation of temperature (or heat distribution) in a given region over some time [3]. It can be expressed as the heat flow in the rod with diffusion $\alpha^2 u_{xx}$ along the rod where the coefficient α is the thermal diffusivity of the rod and L is the length of the rod . In this model, the flow of the heat in one-dimension that is insulated everywhere except at the two end points.

In this chapter, cubic B-spline method is proposed for the numerical solutions of one dimensional heat equation. Two test examples are considered to test the accuracy and efficiency of the method. The absolute, L_2 space and Root Mean Square errors are calculated for the both examples.

In chapter 3, we consider the one dimensional advection-diffusion equation

$$\frac{\partial u}{\partial t} = D\frac{\partial^2 u}{\partial x^2} - C\frac{\partial u}{\partial x}, \quad 0 < x < L \ , \qquad 0 < t \le T$$

The equation described the mathematical model of transport, diffusion processes, mathematical modeling of heat transport, pollutants and suspended matter in groundwater.

In this chapter, cubic B-spline method is proposed for the numerical solutions of one dimensional advection-diffusion equation. Two test examples are considered to test the accuracy and efficiency of the method. The absolute, L_2 space and Root Mean Square errors are calculated for the both examples.

Chapter 2

Numerical Solutions of Heat Equation using Cubic B-Spline

2.1 Introduction

Consider the one dimensional heat equation

$$u_t = \alpha^2 u_{xx} \qquad , 0 \le x \le L \qquad t > 0 \tag{2.1}$$

with initial condition

$$u(x,0) = f(x) \tag{2.2}$$

and boundary conditions

$$u(0,t) = u(L,t) = 0 \qquad for \quad t > 0 \tag{2.3}$$

This problem is one of the well-known second order linear partial differential equation [1,2]. It shows that heat equation describes irreversible process and makes a distance between the previous and next steps. Such equations arise very often in various applications of science and engineering describing the variation of temperature (or heat distribution) in a given region over some time [3]. It can be expressed as the heat flow in the rod with diffusion $\alpha^2 u_{xx}$ along the rod where the coefficient α is the thermal diffusivity of the rod and L is the length of the rod . In this model, the flow of the heat in one-dimension that is insulated everywhere except at the two end points. Solutions of this equation are functions of the state along the rod and the time t . In the past, this problem has been widely worked over a number of years by numerous authors. But it is still an interesting problem since many physical phenomena can be formulated into PDEs with boundary conditions. The heat equation is of fundamental importance in diverse scientific fields.

It is the prototypical parabolic partial differential equation in mathematics. In probability theory, the heat equation is connected with the study of Brownian motion via the Fokker–Planck equation. Numerical solutions of those equations are very useful to study physical phenomena. One of the linear evolution equations which we deal with the numerical solution is the heat equation [4].

In 1946, Schoenberg first proposed the theory of B-splines [3]. Recurrence relations for the purpose of computing coefficients are given by Cox and de Boor [5, 6]. The cubic B-splines collocation method was developed for Burgers' equation and used for the numerical solution of the differential equations in [7, 8]. Recently, spline function theory has been extended and developed to solve the differential equations numerically by various papers S. G. Rubin, P. K. Khosla, H. Caglar, M. Özer, and N. Caglar [9, 10]. Furthermore some extraordinary problems has been numerically investigated by finite element methods such as Galerkin method, least square method and collocation method with quadratic, cubic, quintic and septic B-splines [11-13].

Various techniques of both the cubic spline and cubic B-spline collocation methods and their application have been developed to obtain the numerical solution of the differential equations. They possess some of advantages and are worth on using in the numerical techniques. So, cubic spline collocation procedures exhibit the following the desirable features: (1) obtained governing system is always diagonal which permits easy algorithms; (2) it provides low computer cost and easy problem formulation. The requirement of the continuity up to the second degree are guaranteed at the mesh points over the domain and the first and second degree of the derivatives are directly evaluated [4, 9].

In this study the cubic B-splines collocation method is used for solving the heat equation (2.1) subject to (2.2) and (2.3) and the solutions are compared with the exact solution [14]. For constructing the cubic B-splines finite element method, we use collocation techniques as it was extensively used in [4,15,16]. In the section two, proposed method is presented and it is also given how to apply the collocation method with cubic B-splines finite element technique. In the section three, the stability analysis is investigated considering Fourier stability method. Finally, the numerical results and the related tables are given in the next section.

2.2 Cubic B-Spline Collocation Method for Heat Problem

Let us consider domain $[0,L]$ that is equally-divided with nodal points x_j such that $0 = x_0 < x_1 < .... < x_N = L$, i.e., finite elements of length $h = x_{j+1} - x_j$, $for \quad j = 0,...,N-1$ and also suppose that $\phi_m(x)$ to cubic b-splines at the nodal points $x_m \quad for \quad m = -1,...,N+1.$ using cubic b-splines $\phi_m(x)$, the exact solution $U(x,t)$ is approached by an approximation $U_N(x,t)$ such that

$$U_N(x,t) = \sum_{m=-1}^{N+1} \delta_m(t)\phi_m(x) \tag{2.4}$$

where $\delta_m(t)$ is the parameter in terms of the time t for $m = -1,..,N+1$ to be identified by the boundary conditions. The cubic b-splines $\phi_m(x)$ is defined in [14]

$\phi_m(x)$ is non negative and is locally supported on $[x_{m-2}, x_{m+2}]$. Besides, it is easy to observe that

$\phi_m(x) = \phi_{m+1}(x+h)(m=-1,...,n+1)$ *and* $\sum_{m=-1}^{N+1}\phi_m(x) = 1(x \in [0,L])$

Furthermore by some trivial computation, we obtain the values of $\phi_m(x), (m=-1,0,...N+1)$ at the knots, which are listed in Table 2.1

Table 2.1: Value of $\phi_m(x)$ and its derivatives at the nodal points.

	x_{m-2}	x_{m-1}	x_m	x_{m+1}	x_{m+2}
$B_{m,3}(x)$	0	1	4	1	0
$B'_{m,3}(x)$	0	$\frac{3}{h}$	0	$\frac{-3}{h}$	0
$B''_{m,3}(x)$	0	$\frac{6}{h^2}$	$\frac{-12}{h^2}$	$\frac{6}{h^2}$	0

Considering the approximation function (2.4) and the cubic b-splines $\phi_m(x)$ defined in (2.5), the required values of U_m and its first and the second derivatives with respect to x at nodal points x_m are identified in terms of δ_m as

$$U_m = U(x_m,t) = \delta_{m-1} + 4\delta_m + \delta_{m+1} \tag{2.6}$$

$$U'_m = U'(x_m,t) = \frac{3}{h}(\delta_{m+1} - \delta_{m-1}) \tag{2.7}$$

$$U''_m = U''(x_m,t) = \frac{6}{h^2}(\delta_{m-1} - 2\delta_m + \delta_{m+1}) \tag{2.8}$$

Applying the weighted averaged method on equation (2.1), we have

$$(U_t)_m^n + (1-\theta) f_m^n + \theta\, f_m^{n+1} = 0 \tag{2.9}$$

where

$$f_m^n = \alpha^2 \left(U_{xx}\right)_m^n \tag{2.10}$$

Here θ is a parameter that when it takes the value 0, the scheme is called forward Euler and also if $\theta = 1$, the scheme is so called backward Euler. Then we discretize the time derivative by means of finite difference so we have

$$\frac{U_m^n - U_m^{n+1}}{\Delta t} + (1-\theta)\alpha^2 \left(U_{xx}\right)_m^n + \theta\alpha^2 \left(U_{xx}\right)_m^{n+1} = 0$$

$$U_m^n - U_m^{n+1} + \Delta t(1-\theta)\alpha^2 \left(U_{xx}\right)_m^n + \Delta t\,\theta\alpha^2 \left(U_{xx}\right)_m^{n+1} = 0 \tag{2.11}$$

Substituting (2.6) into (2.11), we obtain the following difference equation system for the variables δ, which has $n+1$ different equations with $n+3$ unknown values as

$$\delta_{m-1}^{n+1}\left(1 - \frac{6}{h^2}\Delta t\theta\alpha^2\right) + \delta_m^{n+1}\left(4 + \frac{12}{h^2}\Delta t\theta\alpha^2\right) + \delta_{m+1}^{n+1}\left(1 - \frac{6}{h^2}\Delta t\theta\alpha^2\right) = L_1 + \Delta t(1-\theta)\alpha^2 L_3 \tag{2.12}$$

where

$$L_1 = \delta_{m-1}^n + 4\delta_m^n + \delta_{m+1}^n \;,\quad L_2 = \frac{3}{h}\left(\delta_{m+1}^n - \delta_{m-1}^n\right),\; L_3 = \frac{6}{h^2}\left(\delta_{m-1}^n - 2\delta_m^n + \delta_{m+1}^n\right)$$

Then this set of equation is a recurrence relationship of element parameters vector $d^n = \left(\delta_{-1}^n, \delta_0^n, \delta_1^n, ..., \delta_N^n, \delta_{N+1}^n\right)$ using the boundary conditions and eliminating the parameters

$\delta_{-1}, \delta_{N+1}$ in (2.12), then the system may be written as

$$U(x_0) = \delta_{-1}^{n+1} + 4\delta_0^{n+1} + \delta_1^{n+1} = 0 \;\Rightarrow \delta_{-1}^{n+1} = -\left(4\delta_0^{n+1} + \delta_1^{n+1}\right) \tag{2.13}$$

$$U(x_N) = \delta_{N-1}^{n+1} + 4\delta_N^{n+1} + \delta_{N+1}^{n+1} = 0 \Rightarrow \delta_{N+1}^{n+1} = -\left(\delta_{N-1}^{n+1} + 4\delta_N^{n+1}\right) \tag{2.14}$$

by these substitutions, the equation (2.12) is turned out to be $N+1$ unknown at each level of the time n

$for\ \ m=0$

$$\left(1-\frac{6}{h^2}\Delta t\theta\alpha^2\right)\delta_{-1}^{n+1}+\left(4+\frac{12}{h^2}\Delta t\theta\alpha^2\right)\delta_0^{n+1}+\left(1-\frac{6}{h^2}\Delta t\theta\alpha^2\right)\delta_1^{n+1}=$$
$$\left(1+\frac{6}{h^2}\Delta t\alpha^2(1-\theta)\right)\delta_{-1}^{n}+\left(4-\frac{12}{h^2}\Delta t\alpha^2(1-\theta)\right)\delta_0^{n}+\left(1+\frac{6}{h^2}\Delta t\alpha^2(1-\theta)\right)\delta_1^{n}$$

$$\left(1-\frac{6}{h^2}\Delta t\theta\alpha^2\right)\left(-4\delta_0^{n+1}-\delta_1^{n+1}\right)+\left(4+\frac{12}{h^2}\Delta t\theta\alpha^2\right)\delta_0^{n+1}+\left(1-\frac{6}{h^2}\Delta t\theta\alpha^2\right)\delta_1^{n+1}=$$
$$\left(1+\frac{6}{h^2}\Delta t\alpha^2(1-\theta)\right)\left(-4\delta_0^{n}-\delta_1^{n}\right)+\left(4-\frac{12}{h^2}\Delta t\alpha^2(1-\theta)\right)\delta_0^{n}+\left(1+\frac{6}{h^2}\Delta t\alpha^2(1-\theta)\right)\delta_1^{n}$$

$$\frac{36}{h^2}\Delta t\theta\alpha^2\delta_0^{n+1}=\frac{-36}{h^2}\Delta t\alpha^2(1-\theta)\delta_0^{n}$$

$for\ \ m=1$

$$\left(1-\frac{6}{h^2}\Delta t\theta\alpha^2\right)\delta_0^{n+1}+\left(4+\frac{12}{h^2}\Delta t\theta\alpha^2\right)\delta_1^{n+1}+\left(1-\frac{6}{h^2}\Delta t\theta\alpha^2\right)\delta_2^{n+1}=$$
$$\left(1+\frac{6}{h^2}\Delta t\alpha^2(1-\theta)\right)\delta_0^{n}+\left(4-\frac{12}{h^2}\Delta t\alpha^2(1-\theta)\right)\delta_1^{n}+\left(1+\frac{6}{h^2}\Delta t\alpha^2(1-\theta)\right)\delta_2^{n}$$

$for\ \ m=2$

$$\left(1-\frac{6}{h^2}\Delta t\theta\alpha^2\right)\delta_1^{n+1}+\left(4+\frac{12}{h^2}\Delta t\theta\alpha^2\right)\delta_2^{n+1}+\left(1-\frac{6}{h^2}\Delta t\theta\alpha^2\right)\delta_3^{n+1}=$$
$$\left(1+\frac{6}{h^2}\Delta t\alpha^2(1-\theta)\right)\delta_1^{n}+\left(4-\frac{12}{h^2}\Delta t\alpha^2(1-\theta)\right)\delta_2^{n}+\left(1+\frac{6}{h^2}\Delta t\alpha^2(1-\theta)\right)\delta_3^{n}$$

$for\ \ m=N$

$$\left(1-\frac{6}{h^2}\Delta t\theta\alpha^2\right)\delta_{N-1}^{n+1}+\left(4+\frac{12}{h^2}\Delta t\theta\alpha^2\right)\delta_N^{n+1}+\left(1-\frac{6}{h^2}\Delta t\theta\alpha^2\right)\delta_{N+1}^{n+1}=$$
$$\left(1+\frac{6}{h^2}\Delta t\alpha^2(1-\theta)\right)\delta_{N-1}^{n}+\left(4-\frac{12}{h^2}\Delta t\alpha^2(1-\theta)\right)\delta_N^{n}+\left(1+\frac{6}{h^2}\Delta t\alpha^2(1-\theta)\right)\delta_{N+1}^{n}$$

$$\left(1-\frac{6}{h^2}\Delta t\theta\alpha^2\right)\delta_{N-1}^{n+1}+\left(4+\frac{12}{h^2}\Delta t\theta\alpha^2\right)\delta_N^{n+1}+\left(1-\frac{6}{h^2}\Delta t\theta\alpha^2\right)\left(-\delta_{N-1}^{n+1}-4\delta_N^{n+1}\right)=$$
$$\left(1+\frac{6}{h^2}\Delta t\alpha^2(1-\theta)\right)\delta_{N-1}^{n}+\left(4-\frac{12}{h^2}\Delta t\alpha^2(1-\theta)\right)\delta_N^{n}+\left(1+\frac{6}{h^2}\Delta t\alpha^2(1-\theta)\right)\left(-\delta_{N-1}^{n}-4\delta_N^{n}\right)$$

$$\frac{36}{h^2}\Delta t\theta\alpha^2\delta_N^{n+1}=\frac{-36}{h^2}\Delta t\alpha^2(1-\theta)\delta_N^{n}$$

$put\ \ \theta = \frac{1}{2}\ ,\ \Delta t = k$

We have the matrix, $AC^{n+1} = BC^{n}$

where

$$A=\begin{bmatrix} \frac{18}{h^2}k\alpha^2 & 0 & & & & \\ 1-\frac{3}{h^2}k\alpha^2 & 4+\frac{6}{h^2}k\alpha^2 & 1-\frac{3}{h^2}k\alpha^2 & & & \\ & 1-\frac{3}{h^2}k\alpha^2 & 4+\frac{6}{h^2}k\alpha^2 & 1-\frac{3}{h^2}k\alpha^2 & & \\ & & 1-\frac{3}{h^2}k\alpha^2 & 4+\frac{6}{h^2}k\alpha^2 & 1-\frac{3}{h^2}k\alpha^2 & \\ & & & 0 & \frac{18}{h^2}k\alpha^2 & \end{bmatrix}_{(N+1)\times(N+1)}$$

$$C^{n+1}=\begin{bmatrix} \delta_0^{n+1} \\ \delta_1^{n+1} \\ \\ \\ \delta_N^{n+1} \end{bmatrix}_{(N+1)\times 1}$$

$$B=\begin{bmatrix} \frac{-18}{h^2}k\alpha^2 & o & & & & \\ 1+\frac{3}{h^2}k\alpha^2 & 4-\frac{6}{h^2}k\alpha^2 & 1+\frac{3}{h^2}k\alpha^2 & & & \\ & 1+\frac{3}{h^2}k\alpha^2 & 4-\frac{6}{h^2}k\alpha^2 & 1+\frac{3}{h^2}k\alpha^2 & & \\ & & 1+\frac{3}{h^2}k\alpha^2 & 4-\frac{6}{h^2}k\alpha^2 & 1+\frac{3}{h^2}k\alpha^2 & \\ & & & 0 & \frac{-18}{h^2}k\alpha^2 & \end{bmatrix}_{(N+1)\times(N+1)} \quad \text{and}$$

$$C^n = \begin{bmatrix} \delta_0^n \\ \delta_1^n \\ \\ \\ \delta_N^n \end{bmatrix}_{(N+1)\times 1}$$

From the initial conditions, we have

(i) $\left(U_N\right)_x\left(x_0,0\right) = U_x\left(x_0\right) = f'\left(x_0\right)$

$$\frac{3}{h}\left[\delta_{-1}^0 - \delta_{-1}^0\right] = f'\left(x_0\right)$$

$$also, u\left(x_0\right) = \delta_{-1}^0 + 4\delta_0^0 + \delta_1^0$$

$$f\left(x_0\right) - 4\delta_0^0 + \delta_1^0 = \delta_{-1}^0$$

$$\Rightarrow \frac{3}{h}\left[f\left(x_0\right) - 4\delta_0^0 + \delta_1^0 - \delta_1^0\right] = f'\left(x_0\right)$$

$$4\delta_0^0 + 2\delta_1^0 = f\left(x_0\right) - \frac{h}{3}f'\left(x_0\right)$$

(ii) $U_N\left(x_j,0\right) = U\left(x_j,0\right)$

$$\delta_{j-1}^0 + 4\delta_j^0 + \delta_{j+1}^0 = f\left(x_j\right)$$

$$for\ \ j = 1,\ \delta_0^0 + 4\delta_1^0 + \delta_2^0 = f\left(x_1\right)$$

$$for\ \ j = 2,\ \delta_1^0 + 4\delta_2^0 + \delta_3^0 = f\left(x_2\right)$$

$$for\ j = n-1,\ \delta_{N-2}^0 + 4\delta_{N-1}^0 + \delta_N^0 = f\left(x_{N-1}\right)$$

(iii) $\left(U_N\left(x_N,0\right)\right)_x = U_x\left(x_N,0\right)$

$$4\delta_N^0 + 2\delta_{N+1}^0 = f\left(x_N\right) - \frac{h}{3}f'\left(x_N\right)$$

Then, we get the initial matrix

$$\begin{bmatrix} 4 & 2 & & & & \\ 1 & 4 & 1 & & & \\ & 1 & 4 & 1 & & \\ & & & & & \\ & & & 1 & 4 & 1 \\ & & & & 2 & 4 \end{bmatrix}_{(N+1)\times(N+1)} \begin{bmatrix} \delta_0^0 \\ \delta_1^0 \\ \\ \\ \\ \delta_N^0 \end{bmatrix}_{(N+1)\times 1} = \begin{bmatrix} f(x_0) - \frac{h}{3} f'(x_0) \\ f(x_1) \\ \\ \\ f(x_{n-1}) \\ f(x_N) - \frac{h}{3} f'(x_N) \end{bmatrix}_{(N+1)\times 1}$$

2.3 The stability analysis

Now, the stability analysis is investigated by using very useful technique which is called Fourier stability method. Additionally the stability analysis is also used for various finite difference methods [17]. Considering the equation (2.11) and (2.12) together, we obtain

$$U_m^{n+1} - U_m^n - \alpha^2 \Delta t (1-\theta)(U_{xx})_m^n - \alpha^2 \theta \Delta t (U_{xx})_m^{n+1} = 0 \tag{2.15}$$

Now substituting (2.6) into (2.15) yields

$$\delta_{m-1}^{n+1}\left(1 - \frac{6}{h^2}\alpha^2\theta\Delta t\right) + \delta_m^{n+1}\left(4 + \frac{12}{h^2}\alpha^2\theta\Delta t\right) + \delta_{m+1}^{n+1}\left(1 - \frac{6}{h^2}\alpha^2\theta\Delta t\right) =$$
$$\delta_{m-1}^{n}\left(1 + \frac{6}{h^2}\alpha^2(1-\theta)\Delta t\right) + \delta_m^{n}\left(4 - \frac{12}{h^2}\alpha^2(1-\theta)\Delta t\right) + \delta_{m+1}^{n}\left(1 + \frac{6}{h^2}\alpha^2(1-\theta)\Delta t\right) \tag{2.16}$$

Using the Fourier stability method, let us assume

$$\delta_m^n = \xi^n e^{i\alpha m} \ , \ \delta_{m+1}^{n+1} = \xi^{n+1} e^{i\alpha(m+1)} \tag{2.17}$$

$$\delta_{m+1}^n = \xi^n e^{i\alpha(m+1)} \ , \ \delta_{m-1}^{n+1} = \xi^{n+1} e^{i\alpha(m-1)}$$

Put (2.17) in (2.16), we get

$$\xi^{n+1}e^{i\alpha(m-1)}\left(1-\frac{6}{h^2}\alpha^2\theta\Delta t\right)+\xi^{n+1}e^{i\alpha m}\left(4+\frac{12}{h^2}\alpha^2\theta\Delta t\right)+\xi^{n+1}e^{i\alpha(m+1)}\left(1-\frac{6}{h^2}\alpha^2\theta\Delta t\right)=$$

$$\xi^{n}e^{i\alpha(m-1)}\left(1+\frac{6}{h^2}\alpha^2(1-\theta)\Delta t\right)+\xi^{n}e^{i\alpha m}\left(4-\frac{12}{h^2}\alpha^2(1-\theta)\Delta t\right)+\xi^{n}e^{i\alpha(m+1)}\left(1+\frac{6}{h^2}\alpha^2(1-\theta)\Delta t\right)$$

divide by $\xi^{n}e^{i\alpha m}$

$$\xi e^{-i\alpha}\left(1-\frac{6}{h^2}\alpha^2\theta\Delta t\right)+\xi\left(4+\frac{12}{h^2}\alpha^2\theta\Delta t\right)+\xi e^{i\alpha}\left(1-\frac{6}{h^2}\alpha^2\theta\Delta t\right)$$

$$= e^{-i\alpha}\left(1+\frac{6}{h^2}\alpha^2(1-\theta)\Delta t\right)+\left(4-\frac{12}{h^2}\alpha^2(1-\theta)\Delta t\right)+e^{i\alpha}\left(1+\frac{6}{h^2}\alpha^2(1-\theta)\Delta t\right)$$

$$\xi(\cos\alpha-i\sin\alpha)\left(1-\frac{6}{h^2}\alpha^2\theta\Delta t\right)+\xi\left(4+\frac{12}{h^2}\alpha^2\theta\Delta t\right)+\xi(\cos\alpha+i\sin\alpha)\left(1-\frac{6}{h^2}\alpha^2\theta\Delta t\right)$$

$$=(\cos\alpha-i\sin\alpha)\left(1+\frac{6}{h^2}\alpha^2(1-\theta)\Delta t\right)+\left(4-\frac{12}{h^2}\alpha^2(1-\theta)\Delta t\right)+(\cos\alpha+i\sin\alpha)\left(1+\frac{6}{h^2}\alpha^2(1-\theta)\Delta t\right)$$

$$\xi\left((2\cos\alpha)\left(1-\frac{6}{h^2}\alpha^2\theta\Delta t\right)+\left(4+\frac{12}{h^2}\alpha^2\theta\Delta t\right)\right)=(2\cos\alpha)\left(1+\frac{6}{h^2}\alpha^2(1-\theta)\Delta t\right)+\left(4-\frac{12}{h^2}\alpha^2(1-\theta)\Delta t\right)$$

$$\xi=\frac{(2\cos\alpha)\left(1+\frac{6}{h^2}\alpha^2(1-\theta)\Delta t\right)+\left(4-\frac{12}{h^2}\alpha^2(1-\theta)\Delta t\right)}{(2\cos\alpha)\left(1-\frac{6}{h^2}\alpha^2\theta\Delta t\right)+\left(4+\frac{12}{h^2}\alpha^2\theta\Delta t\right)}$$

then the solution is stable for $\theta\in\left[\frac{1}{2},1\right]$ using the equation (2.12), since the inequality $|\xi|\le 1$ holds by the Fourier stability method.

2.4 Numerical Experiments

In this subsection, two examples are considered to check the accuracy and efficiency of the proposed method. The following formulas are used to find the absolute, L_2 and RMS errors

$$Absolute\ \ Error = \max\left|u^{num} - u^{exact}\right|$$

$$L_2\ Space\ \ Error\ = \sqrt{\Delta t \sum_{i=1}^{n}\left(u^{num} - u^{exact}\right)^2}$$

$$Root\ \ Mean\ \ Square\ \ Error\ \ = \sqrt{\sum_{i=1}^{n}\frac{\left(u^{num} - u^{exact}\right)^2}{n}}$$

Example 2.1: Consider the one dimensional heat equation

$$\frac{\partial u}{\partial t} = \frac{\partial^2 u}{\partial x^2}\ ,\quad 0 < x < 1\ ,\quad t > 0 \tag{2.18}$$

with initial and boundary conditions

$$u(x,0) = \sin(\pi x)\ ,\ 0 < x < 1\ ,\ u(0,t) = u(1,t) = 0\ \ ,\quad t \geq 0 \tag{2.19}$$

The exact solution is given by

$$u(x,t) = e^{-\pi^2 t}\sin(\pi x) \tag{2.20}$$

The numerical solutions of the example are shown in Table 2.2-2.4 and Figures 1-3. Table 2.2 shows absolute error at different time and time step length *k*. Table 2.3 shows L_2 space error at different time and time step length *k*. Table 2.4 shows root mean square error at different time and time step length *k*. Figure 1 -2 show the behavior of absolute errors for *k* =0.00001 at *t*=2,*t* =4 respectly. Figure 3 shows behavior of numerical solutions at *t*= 0.3, 0.5, 0.7 for *k*=0.00001.

Example 2.2: Consider the one dimensional heat equation

$$\frac{\partial u}{\partial t} = \frac{1}{\pi^2}\frac{\partial^2 u}{\partial x^2}\ ,\quad 0 < x < 1\ ,\quad t > 0 \tag{2.21}$$

with initial and boundary conditions

$$u(x,0)=\sin(\pi x)\ ,\ 0<x<1\ ,\ u(0,t)=u(1,t)=0\ ,\ t\geq 0 \tag{2.22}$$

The exact solution is given by

$$u(x,t)=e^{-t}\sin(\pi x) \tag{2.23}$$

The numerical solutions of the example are shown in Table 2.5-2.7 and Figures 4-6. Table 2.5 shows absolute error at different time and time steps length *k*. Table 2.6 shows L_2 space error at different time and time steps length *k* .Table 2.7 shows root mean square error at different time and time steps length *k*. Figure 4 -5 show the behavior of absolute errors for *k* =0.00001 at *t*=0.7,*t* =2 respectly. Figure 6 shows numerical behavior of numerical solutions at *t*= 0.5,1.5 for *k*=0.00001.

Table 2.2: Maximum absolute error of Example 1 at different time and time step length k.

t	$k = 0.01$	$k = 0.001$	$k = 0.0001$	$k = 0.00001$
0.1	3.2809×10^{-4}	3.3240×10^{-5}	3.0283×10^{-5}	3.0254×10^{-5}
0.3	1.3707×10^{-4}	1.3851×10^{-5}	1.2611×10^{-5}	1.2606×10^{-5}
0.5	3.1707×10^{-5}	3.2065×10^{-6}	2.9190×10^{-6}	2.9185×10^{-6}
0.7	6.1608×10^{-6}	6.2353×10^{-7}	5.6760×10^{-7}	5.6748×10^{-7}
1.0	4.5506×10^{-7}	4.6116×10^{-8}	4.1974×10^{-8}	4.1966×10^{-8}
1.5	4.8983×10^{-9}	4.9277×10^{-10}	4.5270×10^{-10}	4.5268×10^{-10}
2.0	4.6867×10^{-11}	4.7234×10^{-12}	4.3401×10^{-12}	4.3399×10^{-12}
3.0	3.3079×10^{-15}	3.6620×10^{-16}	3.6902×10^{-16}	3.3655×10^{-16}
4.0	3.8453×10^{-17}	1.7883×10^{-17}	8.0518×10^{-19}	6.7635×10^{-18}

Table 2.3: L_2 space error of Example 1 at different time and time step length k.

t	$k = 0.01$	$k = 0.001$	$k = 0.0001$	$k = 0.00001$
0.1	2.3199×10^{-4}	7.4328×10^{-6}	2.1413×10^{-6}	6.7650×10^{-7}
0.3	9.6926×10^{-5}	3.0972×10^{-6}	8.9173×10^{-7}	2.8190×10^{-7}
0.5	2.2420×10^{-5}	7.1700×10^{-7}	2.0640×10^{-7}	6.5259×10^{-8}
0.7	4.3563×10^{-6}	1.3942×10^{-7}	4.0135×10^{-8}	1.2689×10^{-8}
1.0	3.2178×10^{-7}	1.0310×10^{-8}	2.9680×10^{-9}	9.3840×10^{-10}
1.5	3.4636×10^{-9}	1.1018×10^{-10}	3.2011×10^{-11}	1.0122×10^{-11}
2.0	3.3140×10^{-11}	1.0562×10^{-12}	3.0689×10^{-13}	9.7044×10^{-14}
3.0	2.3268×10^{-15}	8.1861×10^{-17}	2.3823×10^{-17}	7.5241×10^{-18}
4.0	1.3812×10^{-17}	5.7900×10^{-19}	5.4881×10^{-20}	1.1827×10^{-19}

Table 2.4: Root mean square error of Example 1 at different time and time step length k.

t	$k = 0.01$	$k = 0.001$	$k = 0.0001$	$k = 0.00001$
0.1	2.3084×10^{-4}	2.3387×10^{-5}	2.1307×10^{-5}	2.1286×10^{-5}
0.3	9.6445×10^{-5}	9.7456×10^{-6}	8.8730×10^{-6}	8.8702×10^{-6}
0.5	2.2309×10^{-5}	2.2561×10^{-6}	2.0538×10^{-6}	2.0534×10^{-6}
0.7	4.3347×10^{-6}	4.3871×10^{-7}	3.9936×10^{-7}	3.9928×10^{-7}
1.0	3.2018×10^{-7}	3.2440×10^{-8}	2.9533×10^{-8}	2.9527×10^{-8}
1.5	3.4464×10^{-9}	3.4671×10^{-10}	3.1852×10^{-10}	3.1850×10^{-10}
2.0	3.2976×10^{-11}	3.3234×10^{-12}	3.0536×10^{-12}	3.0535×10^{-12}
3.0	2.3153×10^{-15}	2.5758×10^{-16}	2.3705×10^{-16}	2.3675×10^{-16}
4.0	1.3743×10^{-17}	1.8218×10^{-18}	5.4608×10^{-19}	3.7217×10^{-18}

Table 2.5: Maximum absolute error of Example 2 at different time and time step length k.

t	$k = 0.01$	$k = 0.001$	$k = 0.0001$	$k = 0.00001$
0.1	8.9263×10^{-6}	7.4497×10^{-6}	7.4422×10^{-6}	7.4428×10^{-6}
0.3	2.0131×10^{-5}	1.8297×10^{-5}	1.8283×10^{-5}	1.8279×10^{-5}
0.5	2.7470×10^{-5}	2.4968×10^{-5}	2.4945×10^{-5}	2.4942×10^{-5}
0.7	3.1487×10^{-5}	2.8618×10^{-5}	2.8591×10^{-5}	2.8589×10^{-5}
1.0	3.3322×10^{-5}	3.0289×10^{-5}	3.0256×10^{-5}	3.0256×10^{-5}
1.5	3.0316×10^{-5}	2.7545×10^{-5}	2.7526×10^{-5}	2.7526×10^{-5}
2.0	2.4516×10^{-5}	2.2272×10^{-5}	2.2259×10^{-5}	2.2260×10^{-5}
3.0	1.3438×10^{-5}	1.2287×10^{-5}	1.2283×10^{-5}	1.2283×10^{-5}
4.0	6.5856×10^{-6}	6.0263×10^{-6}	6.0248×10^{-6}	6.0248×10^{-6}

Table 2.6: L_2 space error of Example 2 at different time and time step length k.

t	$k = 0.01$	$k = 0.001$	$k = 0.0001$	$k = 0.00001$
0.1	6.3118×10^{-6}	1.6658×10^{-6}	5.2624×10^{-7}	1.6642×10^{-7}
0.3	1.4235×10^{-5}	4.0915×10^{-6}	1.2928×10^{-6}	4.0873×10^{-7}
0.5	1.9424×10^{-5}	5.5830×10^{-6}	1.7639×10^{-6}	5.5773×10^{-7}
0.7	2.2264×10^{-5}	6.3993×10^{-6}	2.0217×10^{-6}	6.3929×10^{-7}
1.0	2.3562×10^{-5}	6.7724×10^{-6}	2.1394×10^{-6}	6.7655×10^{-7}
1.5	2.1436×10^{-5}	6.1593×10^{-6}	1.9463×10^{-6}	6.1551×10^{-7}
2.0	1.7335×10^{-5}	4.9801×10^{-6}	1.5740×10^{-6}	4.9776×10^{-7}
3.0	9.5021×10^{-6}	2.7475×10^{-6}	8.6857×10^{-7}	2.7466×10^{-7}
4.0	4.6567×10^{-6}	1.3475×10^{-6}	4.2602×10^{-7}	1.3471×10^{-7}

Table 2.7: Root mean square error of Example 2 at different time and time step length k.

t	$k = 0.01$	$k = 0.001$	$k = 0.0001$	$k = 0.00001$
0.1	6.2805×10^{-6}	5.2416×10^{-6}	5.2363×10^{-6}	5.2367×10^{-6}
0.3	1.4164×10^{-5}	1.2874×10^{-5}	1.2864×10^{-5}	1.2861×10^{-5}
0.5	1.9328×10^{-5}	1.7567×10^{-5}	1.7551×10^{-5}	1.7549×10^{-5}
0.7	2.2154×10^{-5}	2.0136×10^{-5}	2.0116×10^{-5}	2.0115×10^{-5}
1.0	2.3445×10^{-5}	2.1310×10^{-5}	2.1288×10^{-5}	2.1288×10^{-5}
1.5	2.1330×10^{-5}	1.19380×10^{-5}	1.9367×10^{-5}	1.9368×10^{-5}
2.0	1.7249×10^{-5}	1.5670×10^{-5}	1.5661×10^{-5}	1.5662×10^{-5}
3.0	9.4549×10^{-6}	8.6455×10^{-6}	8.6426×10^{-6}	8.6425×10^{-6}
4.0	4.6336×10^{-6}	4.2401×10^{-6}	4.2390×10^{-6}	4.2390×10^{-6}

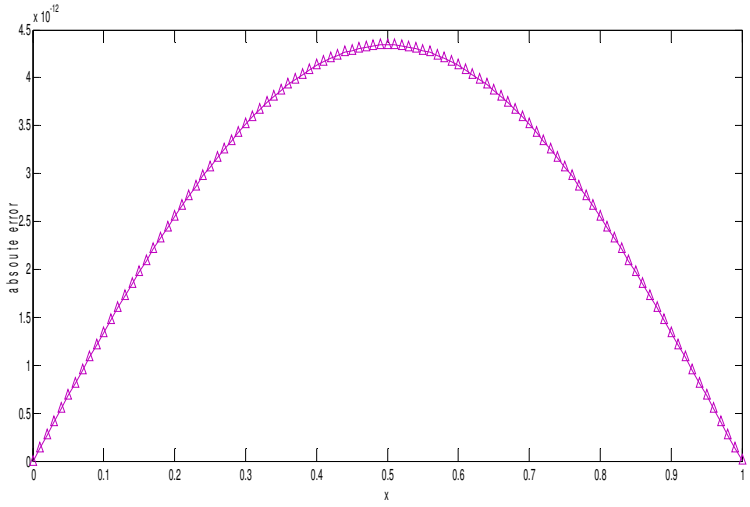

Figure1: Behavior of absolute errors of Example1 for k=0.00001 at t=2.

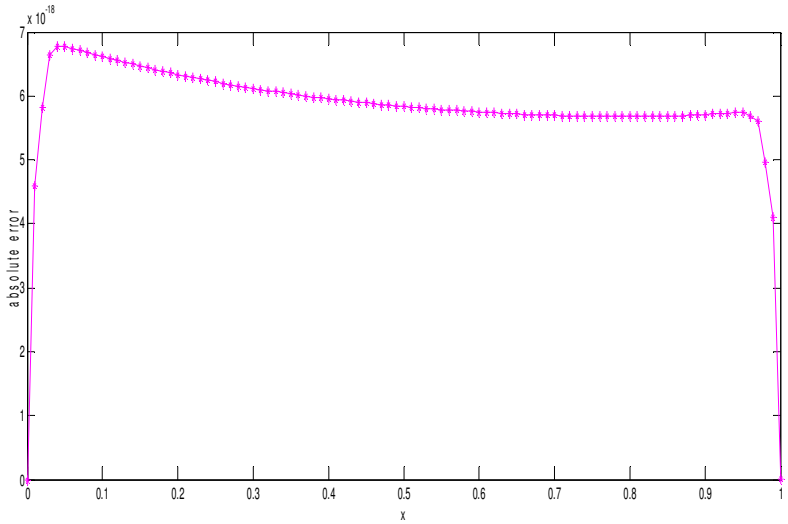

Figure 2: Behavior of absolute errors of Example1 for k=0.00001 at t=4.

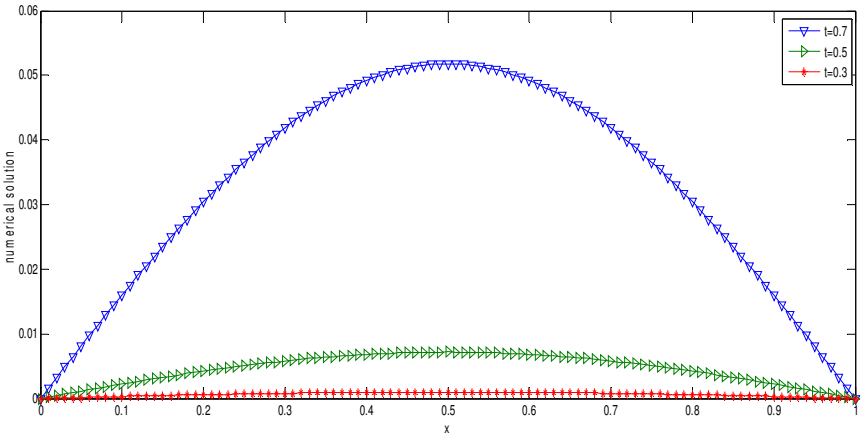

Figure 3: Numerical behavior of numerical solutions of Example1 at different values of *t*.

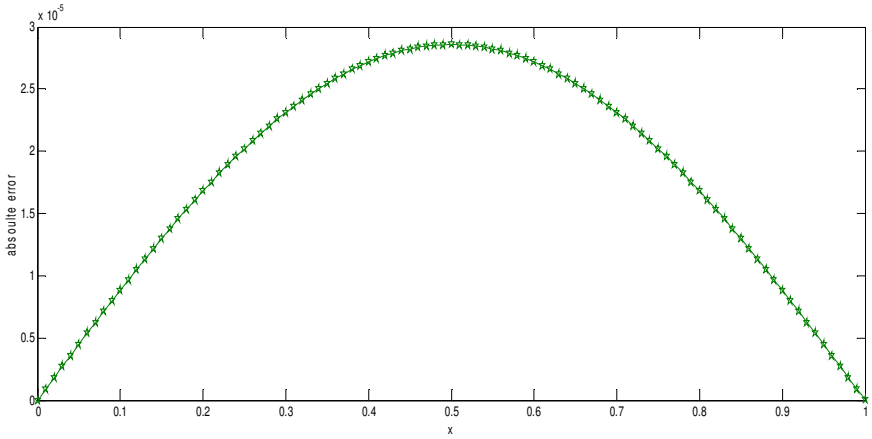

Figure 4: Behavior of absolute errors of Example 2 for k=0.00001 at t=0.7.

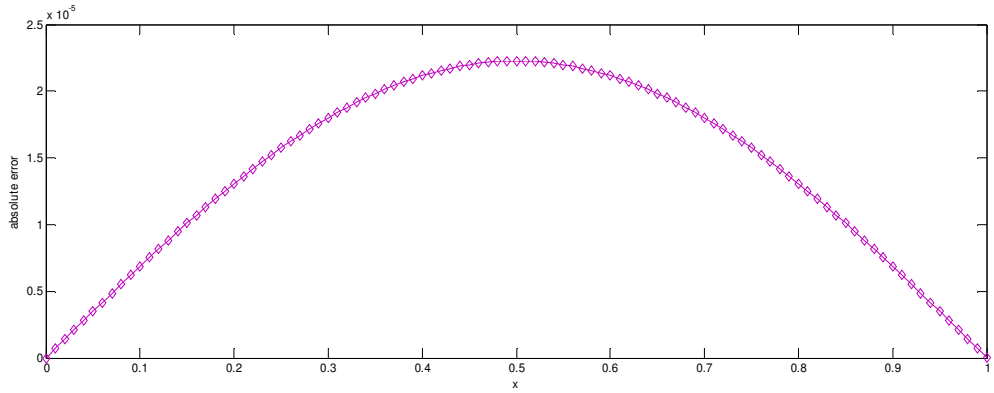

Figure 5: Behavior of absolute errors of Example 2 for k=0.00001 at t=2.

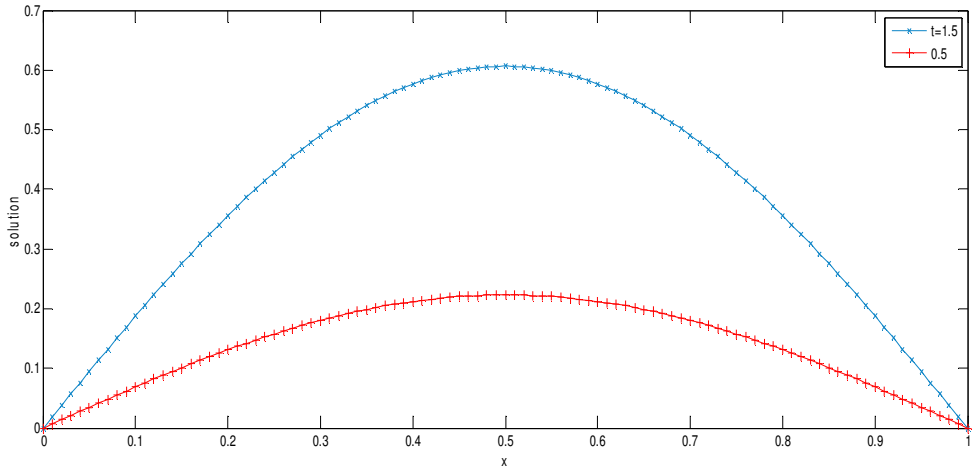

Figure 6: Numerical behavior of numerical solutions of Example 2 at different values of t.

Chapter 3

Numerical Solutions of Advection-Diffusion Equation using Cubic B-Spline

3.1 Introduction

Consider the one dimensional advection-diffusion equation

$$\frac{\partial u}{\partial t} = D\frac{\partial^2 u}{\partial x^2} - C\frac{\partial u}{\partial x} \quad , \quad 0 < x < L \quad , \qquad 0 < t \le T \tag{3.1}$$

with initial conditions

$$u(x,0) = f(x) \qquad 0 \le x \le L \tag{3.2}$$

and boundary conditions

$$u(0,t) = g(t), \qquad 0 < t \le T$$

$$u(L,t) = h(t), \qquad 0 < t \le T \tag{3.3}$$

The mathematical model describing the transport and diffusion processes is the one-dimensional advection-diffusion equation. Mathematical modeling of heat transport, pollutants and suspended matter in groundwater involves the solution of a convection–diffusion equation. When the velocity field is complex and changing in time, transport processes can't be analytically calculated, numerical approximations are necessary. The finite difference method is a well-established and solution techniques are covered in textbooks [18-22]. Many popular finite difference methods, such as Noye and Tan [23] have used a weighted discretisation with the modified equivalent partial differential equation for solving one-dimensional advection–diffusion equations (ADE). Later, the authors extended this scheme to solve two-dimensional ADE [24]. The upwind scheme of Spal-ding [25] and the flux-corrected scheme are available for the

solution of the depth-averaged form of the ADE. Also, widely used is the split-operator approach [26, 27] in which the advection and diffusion terms are solved simultaneously by two different numerical methods.

Lam [28] points out that the central difference approximation will overestimate the advective flux so much that it often causes a negative concentration to appear in the neighboring cells. Some researchers have suggested flux corrected methods, which take into account the mass flow rates and flow directions on the grid cell boundaries by interpolation [29, 30]. To prevent this situation Leonard [31] introduced an upstream interpolation method, namely QUICK (Quadratic Upstream Interpolation Convective Kinematics) for one-dimensional unsteady flow to prevent this situation. Later, Leonard improved this scheme, eliminating the wiggles completely by introducing exponential integration into regions with sharp fronts. The upwind or donor cell method introduced by Gentry et al. [32] is also generally used.

Sommeijer and Kok [33] used various time-integration techniques to improve a finite differences model for the numerical solution of three-dimensional ADE. Their model was validated by comparing with analytical solutions of transport of a Gaussian pulse in unsteady non-uniform flow. Sankaranarayanan et al. [34] used a third-order upwind difference scheme as given by Kowalik and Murty for the advective terms of the ADE. However, previously mentioned techniques require extensive matrix algebra at each time step. One of the tools for solving a PDE is a spreadsheet like Excel. There are many advantages of spreadsheets such as numerical, visual feedback and a graphical interface. Solutions obtained through the spreadsheet are readily plotted in the same worksheet. Any changes in the input parameters of the solution domain are then reflected graphically. Spreadsheets are applied in different fields of engineering

to solve partial differential equations such as free-surface, steady-state groundwater flow, transient groundwater problems and groundwater parameter estimation.

The main objective of this study is to develop a user friendly, flexible advection–diffusion modeling simulation algorithm for the ADE with FDM. Thus ADE Explicit Spreadsheet Simulation (ADEESS) model is proposed. ADEESS model uses the Saulyev's FD schemes. The main advantage of these schemes is that they are unconditionally stable and are explicit in nature. Saulyev developed a new series of techniques for the model parabolic partial differential equation, this followed by interesting variations due to Larkin, Brakat and Clark. ADEESS model is not required to use one of the most significant features of spreadsheets: that is ''iterative calculation''. Only the " copy & paste '' feature of spreadsheets has been automatized and solved with sequential steps with the help of a basic macro. The results of the model are validated with the analytical and numerical solutions in literature. By changing only the weighting parameter in the proposed model is solves ADE and the results can be examined simultaneously with the spreadsheet interface. Thus, the effects of the model parameters (such as $u, \Delta t, \Delta x, D$) to the results can easily be examined graphically. In addition, model parameters $(such\ as\ \Delta t, \Delta x)$ may be adjusted easily in order to overcome the problem of overshooting and negative concentrations.

3.2 Cubic B-Spline Collocation Method for Advection-diffusion Equation

Let us consider domain $[a,b]$ that is equally-divided with nodal points x_j such that

$a = x_0 < x_1 < ... < x_N = b$, i.e., finite elements of length

$h = x_{j+1} - x_j \quad for \quad j = 0,..., N-1$, and also suppose that $\phi_m(x)$ to cubic b-splines at the

nodal points x_m *for* $m=-1,...,N+1$. using cubic b-splines $\phi_m(x)$, the exact solution $U(x,t)$ is approached by an approximation $U_N(x,t)$ such that

$$U_N(x,t)=\sum_{m=-1}^{N+1}\delta_m(t)\phi_m(x) \tag{3.4}$$

Where $\delta_m(t)$ is the parameter in terms of the time t for $m=-1,...N+1$ to be identified by the boundary conditions and the collocation conditions.

The cubic b-splines $\phi_m(x)$ is defined as

$$\phi_m(x)=\frac{1}{h^3}\begin{cases}(x-x_{m-2})^3 & [x_{m-2},x_{m-1})\\ h^3+3h^2(x-x_{m-1})+3h(x-x_{m-1})^2-3(x-x_{m-1})^3 & [x_{m-1},x_m)\\ h^3+3h^2(x_{m+1}-x)+3h(x_{m+1}-x)^2-3(x_{m+1}-x)^3 & [x_m,x_{m+1})\\ (x_{m+2}-x)^3 & [x_{m+1},x_{m+2})\\ 0 & otherwise\end{cases} \tag{3.5}$$

$\phi_m(x)$ is non negative and is locally supported on $[x_{m-2},x_{m+2}]$. Besides ,it is easy to observe that

$\phi_m(x)=\phi_{m+1}(x+h)(m=-1,...,n+1)$ *and* $\sum_{m=-1}^{N+1}\phi_m(x)=1(x\in[a,b])$

Furthermore by some trivial computation, we obtain the values of $\phi_m(x)(m=-1,0,...N+1)$ at the knots, which are listed in Table3.1

Table 3.1: Value of $\phi_m(x)$ and its derivatives at the nodal points.

	x_{m-2}	x_{m-1}	x_m	x_{m+1}	x_{m+2}

$B_{m,3}(x)$	0	1	4	1	0
$B'_{m,3}(x)$	0	$\frac{3}{h}$	0	$\frac{-3}{h}$	0
$B''_{m,3}(x)$	0	$\frac{6}{h^2}$	$\frac{-12}{h^2}$	$\frac{6}{h^2}$	0

Now from Table 3.1,we have

$$U_m = U(x_m,t) = \delta_{m-1} + 4\delta_m + \delta_{m+1} \tag{3.6}$$

$$U'_m = U'(x_m,t) = \frac{3}{h}(\delta_{m+1} - \delta_{m-1}) \tag{3.7}$$

$$U''_m = U''(x_m,t) = \frac{6}{h^2}(\delta_{m-1} - 2\delta_m + \delta_{m+1}) \tag{3.8}$$

Now $\quad \dfrac{\partial u}{\partial t} = D\dfrac{\partial^2 u}{\partial x^2} - C\dfrac{\partial u}{\partial x}$

$$u_j^{n+1} - u_j^n = \frac{\Delta t}{2} D\left(u_{xx}^{n+1} + u_{xx}^n\right) - \frac{\Delta t}{2} C\left(u_x^{n+1} + u_x^n\right)$$

$$\left(u_{j-1}^{n+1} + 4u_j^{n+1} + u_{j+1}^{n+1}\right) - \left(u_{j-1}^n + 4u_j^n + u_{j+1}^n\right) = \frac{\Delta t}{2} D\left[\frac{6}{h^2}\left\{u_{j-1}^{n+1} - 2u_j^{n+1} + u_{j+1}^{n+1}\right\} + \frac{6}{h^2}\left\{u_{j-1}^n - 2u_j^n + u_{j+1}^n\right\}\right]$$
$$- \frac{\Delta t}{2} C\left[\frac{3}{h}\left\{u_{j+1}^{n+1} - u_{j-1}^{n+1}\right\} + \frac{3}{h}\left\{u_{j+1}^n - u_{j-1}^n\right\}\right]$$

$$\left(u_{j-1}^{n+1} + 4u_j^{n+1} + u_{j+1}^{n+1}\right) - \frac{\Delta tD}{2}\frac{6}{h^2}\left(u_{j-1}^{n+1} - 2u_j^{n+1} + u_{j+1}^{n+1}\right) + \frac{\Delta tC}{2}\frac{3}{h}\left(u_{j+1}^{n+1} - u_{j-1}^{n+1}\right) =$$
$$\left(u_{j-1}^n + 4u_j^n + u_{j+1}^n\right) + \frac{\Delta tD}{2}\frac{6}{h^2}\left(u_{j-1}^n - 2u_j^n + u_{j+1}^n\right) - \frac{\Delta tC}{2}\frac{3}{h}\left(u_{j+1}^{n+1} - u_{j-1}^{n+1}\right)$$

$$\left(1 - \frac{3\Delta tD}{h^2} - \frac{3\Delta tC}{2h}\right)u_{j-1}^{n+1} + \left(4 + \frac{6\Delta tD}{h^2}\right)u_j^{n+1} + \left(1 - \frac{3\Delta tD}{h^2} + \frac{3\Delta tC}{2h}\right)u_{j+1}^{n+1} =$$
$$\left(1 + \frac{3\Delta tD}{h^2} + \frac{3\Delta tC}{2h}\right)u_{j-1}^n + \left(4 - \frac{6\Delta tD}{h^2}\right)u_j^n + \left(1 + \frac{3\Delta tD}{h^2} - \frac{3\Delta tC}{2h}\right)u_{j+1}^n$$

$$for \quad j=0$$

$$\left(1-\frac{3\Delta tD}{h^2}-\frac{3\Delta tC}{2h}\right)c_{-1}^{n+1}+\left(4+\frac{6\Delta tD}{h^2}\right)c_0^{n+1}+\left(1-\frac{3\Delta tD}{h^2}+\frac{3\Delta tC}{2h}\right)c_1^{n+1}=$$
$$\left(1+\frac{3\Delta tD}{h^2}+\frac{3\Delta tC}{2h}\right)c_{-1}^{n}+\left(4-\frac{6\Delta tD}{h^2}\right)c_0^{n}+\left(1+\frac{3\Delta tD}{h^2}-\frac{3\Delta tC}{2h}\right)c_1^{n}$$

from the boundary conditions , we have

$$c_{-1}^{n+1}+4c_0^{n+1}+c_1^{n+1}=g^{n+1}(t)$$

$$c_{-1}^{n+1}=g^{n+1}(t)-4c_0^{n+1}-c_1^{n+1}$$

Then, we get

$$\left(1-\frac{3\Delta tD}{h^2}-\frac{3\Delta tC}{2h}\right)\left(g^{n+1}(t)-4c_0^{n+1}-c_1^{n+1}\right)+\left(4+\frac{6\Delta tD}{h^2}\right)c_0^{n+1}+\left(1-\frac{3\Delta tD}{h^2}+\frac{3\Delta tC}{2h}\right)c_1^{n+1}=$$
$$\left(1+\frac{3\Delta tD}{h^2}+\frac{3\Delta tC}{2h}\right)\left(g^{n}(t)-4c_0^{n}-c_1^{n}\right)+\left(4-\frac{6\Delta tD}{h^2}\right)c_0^{n}+\left(1+\frac{3\Delta tD}{h^2}-\frac{3\Delta tC}{2h}\right)c_1^{n}$$

$$\left(\frac{18\Delta tD}{h^2}+\frac{6\Delta tD}{h}\right)c_0^{n+1}+\frac{3C\Delta t}{h}c_1^{n+1}=\left(1+\frac{3\Delta tD}{h^2}+\frac{3\Delta tC}{2h}\right)g^{n}(t)-\left(1-\frac{3\Delta tD}{h^2}-\frac{3\Delta tC}{2h}\right)g^{n+1}(t)$$
$$+\left(-\frac{18\Delta tD}{h^2}-\frac{6\Delta tD}{h}\right)c_0^{n}-\frac{3C\Delta t}{h}c_1^{n}$$

$$for \quad j=1$$

$$\left(1-\frac{3\Delta tD}{h^2}-\frac{3\Delta tC}{2h}\right)c_{-1}^{n+1}+\left(4+\frac{6\Delta tD}{h^2}\right)c_0^{n+1}+\left(1-\frac{3\Delta tD}{h^2}+\frac{3\Delta tC}{2h}\right)c_1^{n+1}=$$
$$\left(1+\frac{3\Delta tD}{h^2}+\frac{3\Delta tC}{2h}\right)c_{-1}^{n}+\left(4-\frac{6\Delta tD}{h^2}\right)c_0^{n}+\left(1+\frac{3\Delta tD}{h^2}-\frac{3\Delta tC}{2h}\right)c_1^{n}$$

$for \quad j = 2$

$$\left(1-\frac{3\Delta tD}{h^2}-\frac{3\Delta tC}{2h}\right)c_1^{n+1}+\left(4+\frac{6\Delta tD}{h^2}\right)c_2^{n+1}+\left(1-\frac{3\Delta tD}{h^2}+\frac{3\Delta tC}{2h}\right)c_3^{n+1}=$$
$$\left(1+\frac{3\Delta tD}{h^2}+\frac{3\Delta tC}{2h}\right)c_1^{n}+\left(4-\frac{6\Delta tD}{h^2}\right)c_2^{n}+\left(1+\frac{3\Delta tD}{h^2}-\frac{3\Delta tC}{2h}\right)c_3^{n}$$

$for \quad j = N$

$$\left(1-\frac{3\Delta tD}{h^2}-\frac{3\Delta tC}{2h}\right)c_{N-1}^{n+1}+\left(4+\frac{6\Delta tD}{h^2}\right)c_N^{n+1}+\left(1-\frac{3\Delta tD}{h^2}+\frac{3\Delta tC}{2h}\right)c_{N+1}^{n+1}=$$
$$\left(1+\frac{3\Delta tD}{h^2}+\frac{3\Delta tC}{2h}\right)c_{N-1}^{n}+\left(4-\frac{6\Delta tD}{h^2}\right)c_N^{n}+\left(1+\frac{3\Delta tD}{h^2}-\frac{3\Delta tC}{2h}\right)c_{N+1}^{n}$$

from the boundary conditions, we have

$$c_{N-1}^{n+1}+4c_N^{n+1}+c_{N+1}^{n+1}=h^{n+1}(t)$$

$$c_{N+1}^{n+1}=h^{n+1}(t)-c_{N-1}^{n+1}-4c_N^{n+1}$$

$$\left(1-\frac{3\Delta tD}{h^2}-\frac{3\Delta tC}{2h}\right)c_{N-1}^{n+1}+\left(4+\frac{6\Delta tD}{h^2}\right)c_N^{n+1}+\left(1-\frac{3\Delta tD}{h^2}+\frac{3\Delta tC}{2h}\right)\left(h^{n+1}(t)-c_{N-1}^{n+1}-4c_N^{n+1}\right)=$$
$$\left(1+\frac{3\Delta tD}{h^2}+\frac{3\Delta tC}{2h}\right)c_{N-1}^{n}+\left(4-\frac{6\Delta tD}{h^2}\right)c_N^{n}+\left(1+\frac{3\Delta tD}{h^2}-\frac{3\Delta tC}{2h}\right)\left(h^{n}(t)-c_{N-1}^{n}-4c_N^{n}\right)$$

$$\frac{-3\Delta tC}{h}c_{N-1}^{n+1}+\left(\frac{18\Delta tD}{h^2}-\frac{6\Delta tC}{h}\right)c_N^{n+1}=\left(1+\frac{3\Delta tD}{h^2}-\frac{3\Delta tC}{2h}\right)h^{n}(t)-\left(1-\frac{3\Delta tD}{h^2}+\frac{3\Delta tC}{2h}\right)h^{n+1}(t)$$
$$+\frac{3\Delta tC}{h}c_{N-1}^{n}+\left(-\frac{18\Delta tD}{h^2}-\frac{6\Delta tC}{h}\right)c_N^{n}$$

$$\begin{bmatrix} \frac{18\Delta tD}{h^2}+\frac{6\Delta tD}{h} & \frac{3C\Delta t}{h} & 0 & & \\ 1-\frac{3\Delta tD}{h^2}-\frac{3\Delta tC}{2h} & 4+\frac{6\Delta tD}{h^2} & 1-\frac{3\Delta tD}{h^2}+\frac{3\Delta tC}{2h} & & \\ 0 & 1-\frac{3\Delta tD}{h^2}-\frac{3\Delta tC}{2h} & 4+\frac{6\Delta tD}{h^2} & 1-\frac{3\Delta tD}{h^2}+\frac{3\Delta tC}{2h} & \\ & 1-\frac{3\Delta tD}{h^2}-\frac{3\Delta tC}{2h} & 4+\frac{6\Delta tD}{h^2} & 1-\frac{3\Delta tD}{h^2}+\frac{3\Delta tC}{2h} & \\ & 0 & -\frac{3C\Delta t}{h} & \frac{18\Delta tD}{h^2}-\frac{6\Delta tD}{h} & \end{bmatrix} c^{n+1} =$$

$$\begin{bmatrix} -\frac{18\Delta tD}{h^2}-\frac{6\Delta tC}{h} & \frac{-3\Delta tC}{h} & 0 & & \\ 1+\frac{3\Delta tD}{h^2}+\frac{3\Delta tC}{2h} & 4-\frac{6\Delta tD}{h^2} & 1+\frac{3\Delta tD}{h^2}-\frac{3\Delta tC}{2h} & & \\ & 1+\frac{3\Delta tD}{h^2}+\frac{3\Delta tC}{2h} & 4-\frac{6\Delta tD}{h^2} & 1+\frac{3\Delta tD}{h^2}-\frac{3\Delta tC}{2h} & \\ & & 1+\frac{3\Delta tD}{h^2}+\frac{3\Delta tC}{2h} & 4-\frac{6\Delta tD}{h^2} & 1+\frac{3\Delta tD}{h^2}-\frac{3\Delta tC}{2h} \\ & & & \frac{3\Delta tC}{h} & -\frac{18\Delta tD}{h^2}-\frac{6\Delta tC}{h} \end{bmatrix}$$

$$+\begin{bmatrix} \left(1+\frac{3\Delta tD}{h^2}+\frac{3\Delta tC}{2h}\right)g^n(t)-\left(1-\frac{3\Delta tD}{h^2}-\frac{3\Delta tC}{2h}\right)g^{n+1}(t) \\ \\ \left(1+\frac{3\Delta tD}{h^2}-\frac{3\Delta tC}{2h}\right)h^n(t)-\left(1-\frac{3\Delta tD}{h^2}+\frac{3\Delta tC}{2h}\right)h^{n+1}(t) \end{bmatrix}_{(N+1)\times 1}$$

From the initial conditions , we have

(i) $\quad (U_N)_x(x_0,0)=U_x(x_0)=f'(x_0)$

$$\frac{3}{h}\left[\delta_{-1}^0-\delta_{-1}^0\right]=f'(x_0)$$

$$also, u(x_0)=\delta_{-1}^0+4\delta_0^0+\delta_1^0$$

$$f(x_0)-4\delta_0^0+\delta_1^0=\delta_{-1}^0$$

$$\Rightarrow \frac{3}{h}\left[f(x_0)-4\delta_0^0+\delta_1^0-\delta_1^0\right]=f'(x_0)$$

$$4\delta_0^0+2\delta_1^0=f(x_0)-\frac{h}{3}f'(x_0)$$

(ii) $$U_N(x_j,0)=U(x_j,0)$$

$$\delta_{j-1}^0+4\delta_j^0+\delta_{j+1}^0=f(x_j)$$

$$for\ \ j=1,\ \delta_0^0+4\delta_1^0+\delta_2^0=f(x_1)$$

$$for\ \ j=2,\ \delta_1^0+4\delta_2^0+\delta_3^0=f(x_2)$$

$$for\ j=n-1,\ \delta_{N-2}^0+4\delta_{N-1}^0+\delta_N^0=f(x_{N-1})$$

(iii) $$\left(U_N(x_N,0)\right)_x=U_x(x_N,0)$$

$$4\delta_N^0+2\delta_{N+1}^0=f(x_N)+\frac{h}{3}f'(x_N)$$

Then we get the initial matrix

$$\begin{bmatrix} 4 & 2 & & & & \\ 1 & 4 & 1 & & & \\ & 1 & 4 & 1 & & \\ & & & & & \\ & & & & & \\ & & & 1 & 4 & 1 \\ & & & & 2 & 4 \end{bmatrix}_{(N+1)\times(N+1)} \begin{bmatrix} \delta_0^0 \\ \delta_1^0 \\ \\ \\ \\ \delta_N^0 \end{bmatrix}_{(N+1)\times 1} = \begin{bmatrix} f(x_0)-\frac{h}{3}f'(x_0) \\ f(x_1) \\ \\ \\ f(x_{n-1}) \\ f(x_N)-\frac{h}{3}f'(x_N) \end{bmatrix}_{(N+1)\times 1}$$

3.3 Numerical Experiment

In this subsection, two examples are considered to check the accuracy and efficiency of the proposed method. The following formulas are used to find the absolute, L_2 space and Root Mean Square errors

$$Absolute\ \ Error = \max\left|u^{num} - u^{exact}\right|$$

$$L_2\ \ Space\ \ \ Error\ \ = \sqrt{\Delta t \sum_{i=1}^{n}\left(u^{num} - u^{exact}\right)^2}$$

$$Root\ \ \ Mean\ \ \ Square\ \ Error\ \ \ = \sqrt{\sum_{i=1}^{n}\frac{\left(u^{num} - u^{exact}\right)^2}{n}}$$

Example 3.1: Consider the one dimensional advection-diffusion equation

$$\frac{\partial u}{\partial t} = D\frac{\partial^2 u}{\partial x^2} - C\frac{\partial u}{\partial x}\ ,\quad 0 < x < 1\ ,\qquad 0 < t \le T \tag{3.9}$$

The initial and boundary conditions are given by

$$c(x,0) = \exp\left[-\frac{(x+0.5)^2}{0.00125}\right] \tag{3.10}$$

$$c(0,t) = \frac{0.025}{\sqrt{0.000625 + 0.02t}}\exp\left[-\frac{(0.5-t)^2}{(0.00125 + 0.04t)}\right] \tag{3.11}$$

$$c(1,t) = \frac{0.025}{\sqrt{0.000625 + 0.02t}}\exp\left[-\frac{(1.5-t)^2}{(0.00125 + 0.04t)}\right] \tag{3.12}$$

Its exact solution is given by

$$u(x,t)=\frac{0.025}{\sqrt{0.000625+0.02t}}\exp\left(\frac{-(x+.5-t)^2}{.00125+0.04t}\right) \tag{3.13}$$

The numerical solutions of the example are shown in Table 3.2-3.4 and Figures 1-4. Table 3.2 shows absolute error at different time and time step length *k*. Table 3.3 shows L_2 space error at different time and time step length *k* .Table 3.4 shows root mean square error at different time and time step length *k*. Figure 1 shows comparsion of exact and numerical solution for C=1.0,D=0.01,*k*=0.00001 at *t*=2.5. Figure 2 shows the absolute error for C=1.0,D=0.01, *k*=0.00001 at *t*=2. Figure 3 shows numerical solution for C=1.0, D=0.01,*k*=0.00001 at *t*=0.5 while figure 4 shows numerical solution for C=1.0,D=0.01,*k*=0.00001 at *t*=0.5.

Example 3.2: Consider the one dimensional advection-diffusion equation

$$\frac{\partial u}{\partial t}=D\frac{\partial^2 u}{\partial x^2}-C\frac{\partial u}{\partial x}\ ,\quad 0<x<L\ ,\qquad 0<t\le T \tag{3.14}$$

The initial and boundary conditions

$$c(x,0)=\exp\left[-\frac{(x-x_0)^2}{D}\right] \tag{3.15}$$

$$c(0,t)=\frac{1}{\sqrt{4t+1}}\exp\left[-\frac{(-x_0-ut)^2}{D(4t+1)}\right] \tag{3.16}$$

$$c(9,t)=\frac{1}{\sqrt{4t+1}}\exp\left[-\frac{(9-x_0-ut)^2}{D(4t+1)}\right] \tag{3.17}$$

The exact solution is given by

$$u(x,t)=\frac{1}{\sqrt{4t+1}}\exp\left(\frac{-(x-b-Ct)^2}{D(4t+1)}\right) \tag{3.18}$$

The numerical solutions of the example are shown in Table 3.5-3.7 and Figures 5-8. Table 3.5 shows absolute error at different time and time step length *k*. Table 3.6 shows L_2 space error at different time and time step length *k* .Table 3.7 shows root mean square error at different time and time step length *k*. Figure 5 shows comparsion of exact and numerical solution for C=1.0,D=0.01,*k*=0.00001 at *t*=3. Figure 6 shows the absolute error for C=1.0, D=0.01, *k*=0.00001 at *t*=4. Figure 7 shows numerical solution for C=1.0,D=0.01,*k*=0.00001 at *t*=1. Figure 8 shows numerical solution for C=1.0, D=0.01,*k*=0.00001 at *t*=3

Table 3.2: Maximum absolute error of Example1 at different time and time step length k.

t	$k = 0.01$	$k = 0.001$	$k = 0.0001$	$k = 0.00001$
0.1	1.1966×10^{-13}	7.7990×10^{-16}	8.6149×10^{-16}	8.6541×10^{-16}
0.3	4.5009×10^{-5}	6.6276×10^{-6}	7.1472×10^{-6}	7.1385×10^{-6}
0.5	2.1981×10^{-4}	3.8657×10^{-5}	4.0739×10^{-5}	4.0747×10^{-5}
0.7	3.7478×10^{-4}	4.6601×10^{-5}	4.9536×10^{-5}	4.9560×10^{-5}
1.0	$3.9382\times10^{-}$	7.3733×10^{-5}	7.4311×10^{-5}	7.4119×10^{-5}
2.5	1.3314×10^{-6}	1.4327×10^{-6}	1.4546×10^{-6}	1.4546×10^{-6}
3.0	2.2377×10^{-10}	2.3936×10^{-10}	2.4360×10^{-10}	2.4359×10^{-10}
4.0	3.6441×10^{-19}	3.9453×10^{-19}	4.0254×10^{-19}	4.0253×10^{-19}
6.0	1.6562×10^{-32}	1.9721×10^{-31}	1.2227×10^{-29}	6.6243×10^{-27}

Table 3.3: L_2 space error of Example1 at different time and time step length k.

t	$k = 0.01$	$k = 0.001$	$k = 0.0001$	$k = 0.00001$
0.1	1.4726×10^{-14}	2.6990×10^{-17}	9.4217×10^{-18}	2.9930×10^{-18}
0.3	1.1676×10^{-5}	5.4798×10^{-7}	1.8705×10^{-7}	5.9075×10^{-8}
0.5	7.9358×10^{-5}	3.6879×10^{-6}	1.2355×10^{-6}	3.9077×10^{-7}
0.7	1.3840×10^{-4}	6.0350×10^{-6}	1.9969×10^{-6}	6.3172×10^{-7}
1.0	1.6894×10^{-4}	6.9289×10^{-6}	2.2693×10^{-6}	7.1767×10^{-7}
2.5	1.5660×10^{-7}	5.0579×10^{-8}	1.6230×10^{-8}	5.1324×10^{-9}
3.0	2.7901×10^{-11}	8.7634×10^{-12}	2.8179×10^{-12}	8.9109×10^{-13}
4.0	4.8737×10^{-20}	1.5453×10^{-20}	4.9812×10^{-20}	1.5751×10^{-21}
6.0	1.3162×10^{-32}	1.0473×10^{-32}	1.5809×10^{-31}	2.0568×10^{-28}

Table 3.4: Root mean square error of Example1 at different time and time step length k.

t	$k = 0.01$	$k = 0.001$	$k = 0.0001$	$k = 0.00001$
0.1	1.4653×10^{-14}	8.4929×10^{-17}	9.3750×10^{-17}	9.4179×10^{-17}
0.3	1.1618×10^{-5}	1.7242×10^{-6}	1.8613×10^{-6}	1.8588×10^{-6}
0.5	7.8964×10^{-5}	1.1604×10^{-5}	1.2294×10^{-5}	1.2296×10^{-5}
0.7	1.3771×10^{-4}	1.8989×10^{-5}	1.9870×10^{-5}	1.9877×10^{-5}
1.0	1.6810×10^{-4}	2.1805×10^{-5}	2.2580×10^{-5}	2.2582×10^{-5}
2.5	1.5582×10^{-7}	1.5915×10^{-7}	1.6149×10^{-7}	1.6149×10^{-7}
3.0	2.7763×10^{-11}	2.7574×10^{-11}	2.8039×10^{-11}	2.8039×10^{-11}
4.0	4.8495×10^{-20}	4.8625×10^{-20}	4.9565×10^{-20}	4.9563×10^{-20}
6.0	1.3097×10^{-32}	3.2955×10^{-32}	1.5731×10^{-30}	6.4720×10^{-27}

Table 3.5: Maximum absolute error of Example 2 at different time and time step length k.

t	$k = 0.01$	$k = 0.001$	$k = 0.0001$	$k = 0.00001$
2.0	7.9834×10^{-4}	8.0122×10^{-5}	8.2186×10^{-5}	8.2216×10^{-5}
3.0	5.7378×10^{-4}	4.8009×10^{-5}	4.918×10^{-5}	4.9195×10^{-5}
4.0	4.4811×10^{-4}	3.2837×10^{-5}	3.3538×10^{-5}	3.3548×10^{-5}
5.0	3.6750×10^{-4}	2.4262×10^{-5}	2.4721×10^{-5}	2.4728×10^{-5}
6.0	3.1134×10^{-4}	1.8863×10^{-5}	1.9185×10^{-5}	1.9191×10^{-5}
6.5	2.8924×10^{-4}	1.6881×10^{-5}	1.7141×10^{-5}	1.7152×10^{-5}

Table 3.6: L_2 space error of Example 2 at different time and time step length k.

t	$k = 0.01$	$k = 0.001$	$k = 0.0001$	$k = 0.00001$
2.0	4.8523×10^{-4}	1.3300×10^{-5}	4.3051×10^{-6}	1.3618×10^{-6}
3.0	3.8249×10^{-4}	8.7506×10^{-6}	2.8239×10^{-6}	8.9326×10^{-7}
4.0	3.1941×10^{-4}	6.4003×10^{-6}	2.0592×10^{-6}	6.5136×10^{-7}
5.0	2.7612×10^{-4}	4.9894×10^{-6}	1.6001×10^{-6}	5.0615×10^{-7}
6.0	2.4438×10^{-4}	4.0565×10^{-6}	1.2971×10^{-6}	4.1030×10^{-7}
6.5	2.3144×10^{-4}	3.7136×10^{-6}	1.1857×10^{-6}	3.7506×10^{-7}

Table 3.7 : Root mean square error of Example 2 at different time and time step length k.

t	$k = 0.01$	$k = 0.001$	$k = 0.0001$	$k = 0.00001$
2.0	1.6165×10^{-4}	1.4011×10^{-5}	1.4342×10^{-5}	1.4347×10^{-5}
3.0	1.2742×10^{-4}	9.2188×10^{-6}	9.4078×10^{-6}	9.4106×10^{-6}
4.0	1.0641×10^{-4}	6.7427×10^{-6}	6.8602×10^{-6}	6.8622×10^{-6}
5.0	9.1989×10^{-5}	5.2564×10^{-6}	5.3308×10^{-6}	5.3323×10^{-6}
6.0	8.1414×10^{-5}	4.2735×10^{-6}	4.3213×10^{-6}	4.3226×10^{-6}
6.5	7.7105×10^{-5}	3.9123×10^{-6}	3.9501×10^{-6}	3.9513×10^{-6}

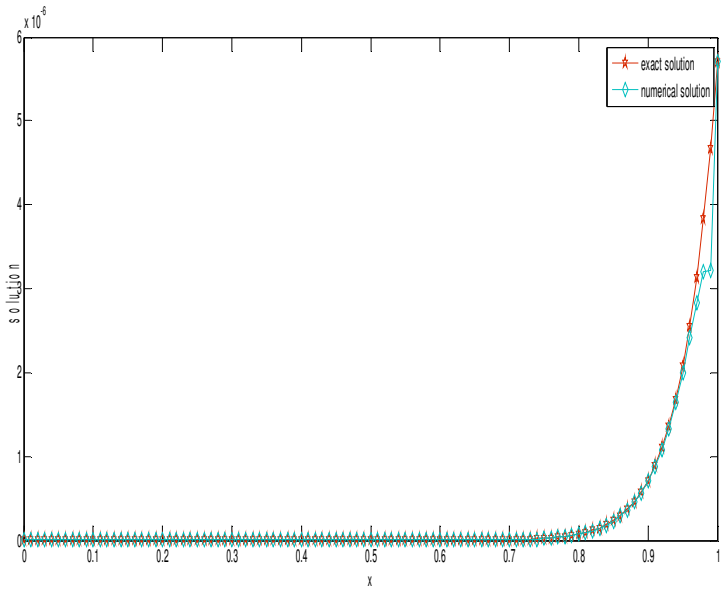

Figure 1: Comparsion of exact and numerical solution of Example 1 for C=1, D=0.01, k=0.00001 at t=2.5.

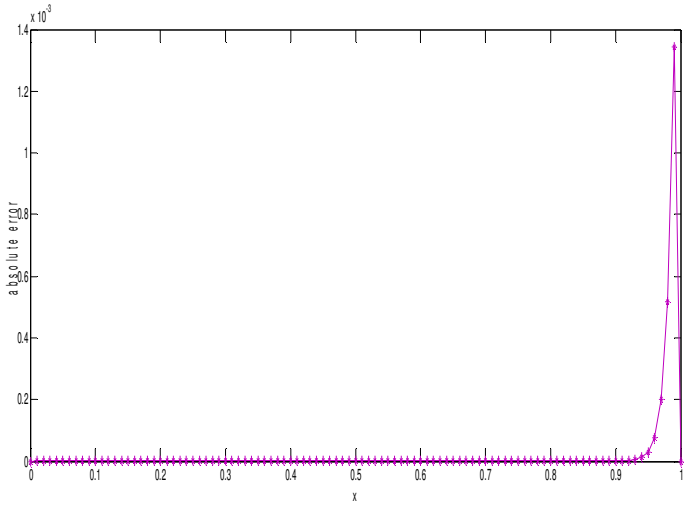

Figure 2: Behavior of absolute errors of Example 1 for C=1.0, D=0.01, k=0.00001 at t=2

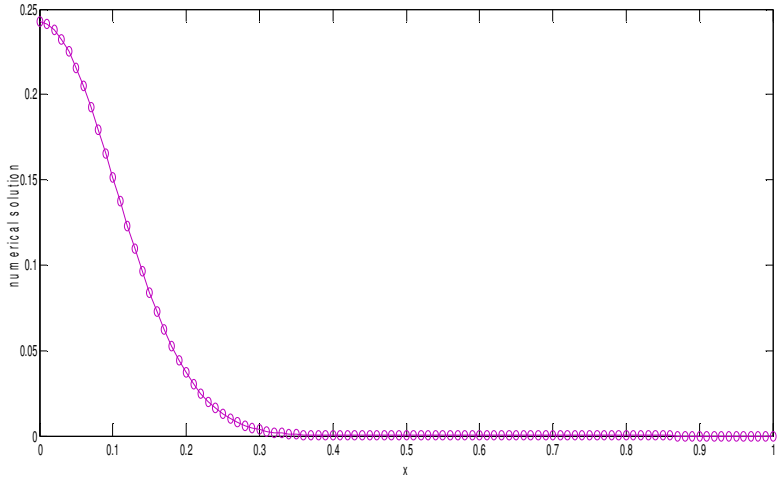

Figure 3: Numerical solution of Example 1for C=1.0,D=0.01,k=0.00001 at t=0.5.

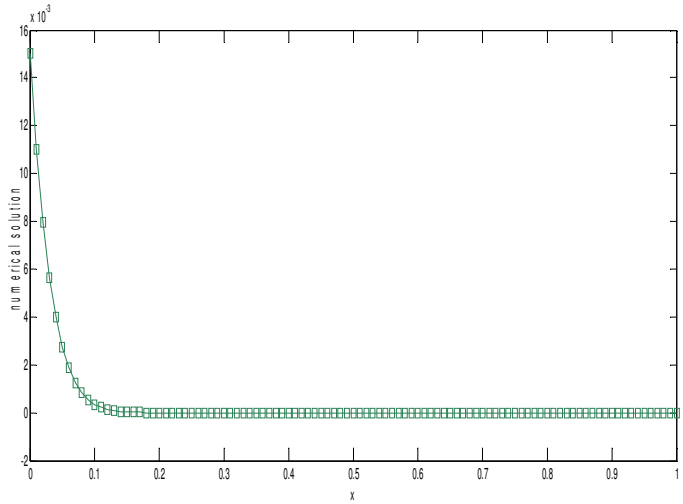

Figure 4: Numerical solution of Example 1 for C=1.0,D=0.01,k=0.00001 at t=0.3.

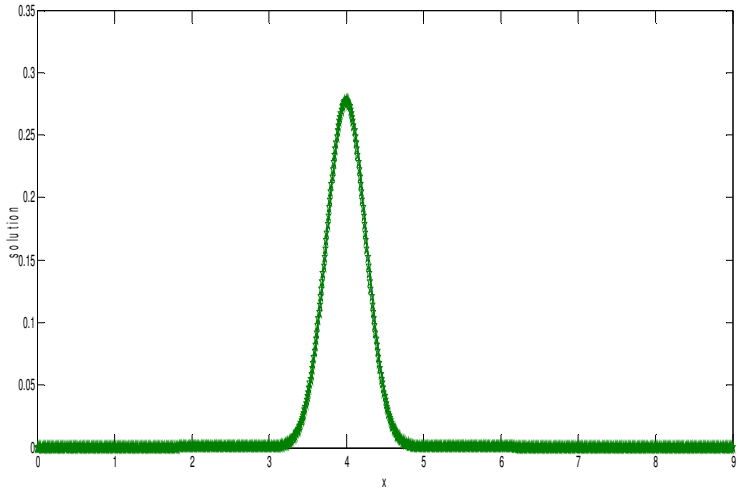

Figure 5: Comparsion of exact and numerical solution of Example 2 for C=1.0, D=0.01, k=0.00001 at t=3.

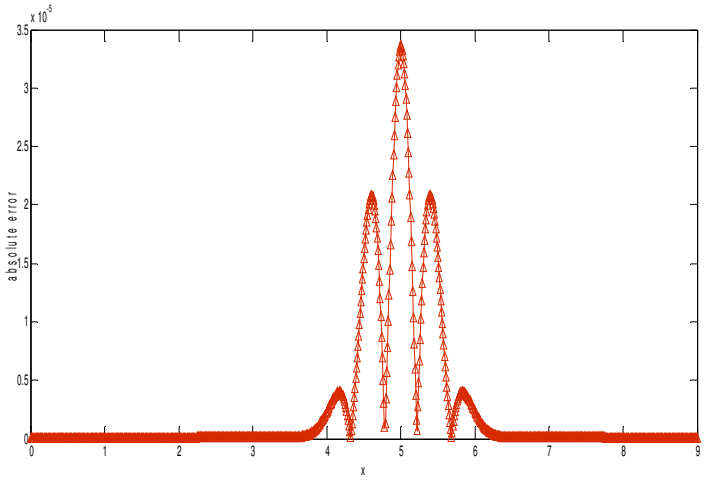

Figure 6: Behavior of absolute errors of Example 2 for C=1.0, D=0.01, k=0.00001 at t=4.

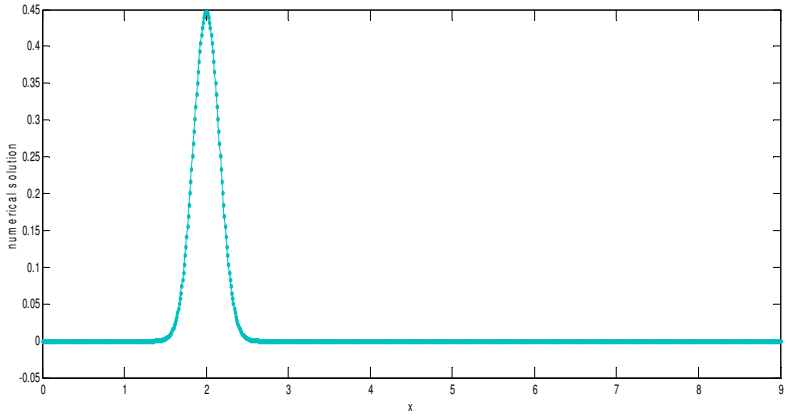

Figure 7: Numerical solution of Example 2 for C=1.0, D=0.01,*k*=0.00001 at *t*=1.

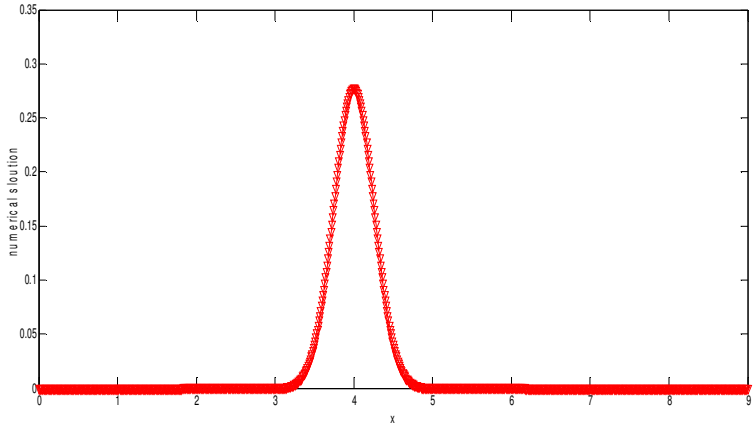

Figure 8: Numerical solution of Example 2 for C=1.0, D=0.01, *k*=0.00001 at t=3.

REFERENCES

1. D. V. Widder, the heat equation, Academic Press, 1976.
2. J. R. Cannon, the one-dimensional heat equation, Cambridge University Press, 1984.
3. D. D. Demir and N. Bildik, Numerical solution of heat problem using cubic B-spline, Applied Mathematics, 2 (4) (2004) 131-135.
4. I. Dag, B. Saka and D. Irk , Application cubic B-splines for numerical solution of the RLW equation, Appl. Maths. Comp., 159 (2004) 373-389.
5. C. de Boor, A Practicle guide to splines, Applied Mathematical Sciences, Springer-Verlag, 2001.
6. C. de Boor, On Calculating with B-splines, Journal of Approximation Theory, 6 (1972) 50- 62.
7. S. Kutluay, A. R. Bahadır, A. Özdes, numerical solution of one-dimensional burger equation: explicit and exact-explicit finite difference methods, J.Comp. App. Math., 103 (1999) 251-261.
8. I. Dag , D. Irk and B. Saka , A numerical solution of the burgers' equation using cubic B-splines, Appl. Maths. Comp., 163 (2005) 199–211.
9. S. G. Rubin, P. K. Khosla, Higher-order numerical solutions using cubic splines, AIAA J., 14 (1976) 851-858.
10. H. Caglar, M. Özer, and N. Caglar, The numerical solution of the one-dimensional heat equation by using third degree B-spline functions, Chaos, Solitons & Fractals, 38 (2008) 1197-1201.

11. B. Saka, I. Dag, Quartic B-spline collocation method to the numerical solutions of the burgers' equation, Chaos, Solitons & Fractals, 32 (2007) 1125-1137.

12. B. Saka, I. Dag, and A. Boz, B-spline Galerkin methods for numerical solutions of the burgers' equation, Appl. Maths. Comp., 166 (2005) 506-522.

13. M. A. Ramadan , T. S. El-Danaf, and F .E .I. Abd Alaal, A numerical solution of the burgers' equation using septic B-splines, Chaos, Solitons & Fractals, 26 (2005) 795-804.

14. P. M .Prenter, Splines and variational methods, John Wiley, New York, 1975.

15. A. K. Khalifa, K. R. Raslana and H.M. Alzubaidi , a collocation method with cubic B-splines for solving the MRLW Equation, Journal of Computational and Applied Mathematics 212 (2008) 406 – 418.

16. A. H. A. Ali, L. R. T. Gardner, and G. A. Gardner, a collocation method for burgers' equation using cubic splines, Comp. Math. Appl. Mech. Eng., 100 (1992) 325-337.

17. G. Sewell, The numerical solution of ordinary and partial differential equations, John Wiley and Sons, 2005.

18. Bear J, Bachmat Y. Introduction to modeling of transport phenomena. Dordrecht: Kluweri: 1990.

19. Kinzelbach W. Groundwater modeling: An introduction with sample programs in BASIC. Amsterdam: Elsevier; 1986.

20. Remson I, Hornberger GM, Molz FJ. Numerical methods in subsurface hydrology. New York: Wiley Interscience; 1971.

21. Wang HF, Anderson MP. Introduction to groundwater modeling: Finite difference and Finite Element. London: Academic Press; 1982.
22. Zheng C, Bennett GD. Applied contaminant transports modeling. New York: Int. Thomson Publishing Inc.; 1995.
23. Noye BJ, Tan HH. A third-order semi-implicit finite difference method for solving the one-dimensional convection–diffusion equation. Int J Numer Meth Eng 1988; 26:1615–29.
24. Noye BJ, Tan HH. Finite difference methods for the two dimensional advection diffusion equations. Int J Numer Meth Fluids 1989; 9:75–98.
25. Spalding DB, A novel finite difference formulation for differential expression involving both first and second derivatives. Int J Numer Meth Fluids1972; 4:551–9.
26. Li YS, Chen CP.An efficient split operator scheme for 2D advection diffusion equation using finite elements and characteristics. Appl Math Mod 1989; 13:248–53.
27. Sobey RJ, Fractional step algorithm for estuarine mass transport. Int J Numer Meth Fluids 1983; 3:567–81.
28. Lam DCL.Computer modeling of pollutant transport in Lake Erie. Water Pollut 1975; 25:75– 86.
29. Michael GE, Rubin B. Total variation diminishing schemes and the two point upstream mobility weighting schemes. Soc Petr Eng Dallas, Techn Publ 23947; 1990.
30. Rubin B, Blunt MJ. High order implicit flux limiting schemes for Black oil simulation. Soc Petr Eng Dallas, Techn Publ 21222; 1990.
31. Leonard BP, A stable and accurate convective modeling procedure based on upstream formulation. Comput Meth Appl Mech Eng 1979; 19:59–98.

32. Gentry RA, Martin RE, and Daly BJ. An eulerian differencing method for unsteady compressible flow problems. J Comput Phys 1966; 8:55-76.

33. Sommeijer BP, Kok J. Implementation and performance of the time integration of a 3D numerical transport model. Int J Numer Meth Fluids 1995; 21:349–67.

34. Sankaranarayanan S, Shankar NJ, and Cheong HF. Three-dimensional finite difference model for transport of conservative pollutants. Ocean Eng 1998; 25(6):425–42.

35. Bellman R., Kashef B.G. and J. Casti, Differential quadrature: A technique for the rapid solution of nonlinear partial differential equations, J. Comput. Phys., 10 (1972) 40-52.

36. Bert C.W. and Malik M., Differential quadrature method in computational mechanics, Appl. Mech. Rev., 49 (1) (1996e) 1-28.

37. Chen W., Zhong T.X. and Shu C., A Lyapunov formulation for efficient solution of the Poisson and convection-diffusion equations by differential quadrature method, J.Comput. Phys., 141 (1) (1998) 78-84.

38. Civan F. and Sliepcevich C.M., Differential quadrature for multi-dimensional problems, J. Math. Anal. Appl., 101 (1984b) 423-443.

39. Malekzadeh P. and Karami G., Polynomial and harmonic differential quadrature methods for free vibration of variable thickness thick skew plates, Eng. Struc., 27 (2005), 1563-1574.

40. Shu.C and Y. T. Chew, Fourier expansion-based differential quadrature and its application to Helmholtz eigenvalue problems, Commun. Numer. Methods Eng., 13 (8)(1997) 64

41. Ram Jiwari, R.C. Mittal and K. K Sharma**,** A numerical scheme based on weighted average differential quadrature method for the numerical solution of Burgers' equation, Applied Mathematics and Computation, 219 (2013) 6680–6691.
42. R .C. Mittal, Ram Jiwari and K. K Sharma**,** A numerical scheme based on differential quadrature method to solve time dependent Burgers' equation, Engineering Computations, 30 (1) (2013) 117-131.
43. Ram Jiwari, Haar wavelet quasilinearization approach for numerical simulation of Burgers' equation, Computer Physics Communications, 183 (2012) 2413-2423.
44. R.C. Mittal and Ram Jiwari, A differential quadrature method for solving Burgers'-type equation, International Journal of numerical methods for Heat and Fluid flow, 22 (7), (2012), 880-895.
45. Ram Jiwari, S. Pandit and R C Mittal**,** Numerical simulation of two-dimensional sine-Gordon solitons by differential quadrature method, Computer Physics Communications, 183 (2012) 600-616.
46. Ram Jiwari, S. Pandit and R C Mittal**,** A differential quadrature algorithm to solve the two dimensional linear hyperbolic telegraph equation with diriclet and neumann boundary conditions, Applied Mathematics and Computation, 218 (2012) 7279–7294.
47. R.C. Mittal and Ram Jiwari, differential quadrature method for numerical solution of coupled viscous burgers' equations, Int. J. for Comput. Methods in Eng. Science and Mech, 13 (2012), 1-5.
48. D. Sharma**,** Ram Jiwari, and Sheo Kumar**,** A comparative study of modal matrix and finite elements methods for two point boundary value problems**,** Int. J. of Appl. Math. And Mech. 8 (13) (2012), 29-45.

49. Ram Jiwari, Sapna Pandit and R C Mittal**,** A differential quadrature algorithm for the numerical solution of the second-order one dimensional hyperbolic telegraph equation, Int J of Nonlinear Sciences, 13 (3) (2012), 259-266.

50. Ram Jiwari, Dinkar Shrma and Sheo Kumar**,** Numerical solutions of two point boundary value problems using Galerkin-Finite element method, Int J of Nonlinear Sciences, 13 (2)(2012), 204-210.

51. R.C. Mittal and Ram Jiwari, A numerical scheme for singularly perturbed burger-huxley equation, J. Appl. Math. & Informatics, 29 (2011), No. 3-4, 813-829.

52. R.C. Mittal and Ram Jiwari, A numerical scheme for some nonlinear differential equations models in Biology, Int. J. for Comput. Methods in Eng. Science and Mech., 12 (3), (2011), 134- 140.

53. R.C. Mittal and Ram Jiwari, Numerical study of two-dimensional reaction-diffusion brusselator System, Appl. Math. Comput, 217 (12) (2011), 5404-5415.

54. Ram Jiwari, Dinkar Shrma and Sheo Kumar**,** Galerkin-finite element method for the numerical solution of advection-diffusion equation, IJPAM, 70 (3) (2011), 389-399.

55. R.C. Mittal and Ram Jiwari, Numerical study of burger-huxley equation by differential quadrature method, Int. J. of Appl. Math. And Mech., 5(8) (2009), 1-9.