NUMERICAL SOLUTIONS OF SOME PARTIAL DIFFERENTIAL EQUATIONS USING B-SPLINE

SYNOPSIS

Submitted in partial fulfilment of the requirements for the award of the degree

of

DOCTOR OF PHILOSOPHY

in

MATHEMATICS

by

RACHNA BHATIA

Under the Supervision of

Prof. R. C. MITTAL

**Department of Mathematics
Indian Institute of Technology Roorkee
Roorkee- 247667, India**

November, 2014

Introduction

Partial differential equations are used in modeling of many physical, chemical and biological phenomena, besides these its uses have also spread into financial forecasting, image processing, economics and other fields as well. There are many theoretical results on existence and uniqueness, but only the simplest specific problems can be solved explicitly. Since limited classes of the equations are solved by analytical means, we usually construct approximate numerical solution of these differential equations especially nonlinear one, which is of practical importance. Various numerical techniques have been promulgated for finding the solution of partial differential equations among which the two most popular are (a) finite difference method and (b) finite element method.

Spline is a numeric function that is piecewise defined by polynomial functions and which possesses a sufficiently high degree of smoothness at the places where the polynomial pieces are connected. The first mathematical reference to splines was made in the year 1946 in an interesting paper by Schoenberg [1], which is probably the first place that the word "spline" is used in connection with smooth, piecewise polynomial approximation. Splines have many implementations in the numerical solution of a variety of problems in applied mathematics and engineering. Some of them are, Data fitting, Function approximation, Integro-differential equations, Optimal control problems, Computer-Aided Geometric Design (CAGD) and Wavelets. Some of the books which discuss splines thoroughly includes Ahlberg et al. [4], deBoor [3], Prenter [5], Schumaker [2].

B-spline is a spline function that has minimal support with respect to given degree, smoothness and domain partition. Every spline function of a given degree, smoothness, and domain partition can be uniquely represented as a linear combination of B-splines of that same degree and smoothness, and over that same partition. The term B-spline was coined by Isaac Jacob Schoenberg [1] and is short for basis spline. The original definition of the B-spline basis functions was given by Schumaker [2], which uses the idea of divided differences. In 1970s, a recurrence relation was independently established by Cox and deBoor [3] for computing the B-splines basis functions. By applying the Leibniz's theorem, deBoor was able to derive the following formula for i^{th} B-splines basis function of d^{th} degree in a recursive manner as follows:

$$B_{i,d}(x) = \left(\frac{x - x_i}{x_{i+d} - x_i}\right) B_{i,d-1} + \left(\frac{x_{i+d+1} - x}{x_{i+d+1} - x_{i+1}}\right) B_{i+1,d-1}. \tag{1}$$

This formula is known as Cox and deBoor [3] recursion formula and shows that the B-splines basis functions of any arbitrary degree can be stably evaluated as a linear combination of basis functions of lower degree. The recurrence relation starts with the first degree B-splines. For degree $d \geq 1$, basis function $B_{i,d}(x)$ is a linear combination of two $(d-1)^{th}$ degree basis functions.

B-splines are the smoothest interpolating functions compared with other piecewise polynomial interpolating functions and have been used as a basis functions in Finite element method, Galerkin method, Collocation method and Differential Quadrature method, to construct numerical methods for the approximate solutions of PDEs occurring in various engineering applications. In many papers various techniques using quadratic, cubic, quartic, quintic, sextic, septic and higher degree B-splines have been discussed for the numerical solution of linear and nonlinear PDEs. For example, Ahlberg and Ito [6] have presented a collocation method for two-point boundary value problems using cubic, quintic and septic B-splines. Dağ et al. [8] have solved RLW equation using quadratic and cubic B-splines collocation method. Kutluay et al. [9], have given a least-squares quadratic B-spline finite element method for Burgers' equation. Özis et al. [10] used a Galerkin quadratic B-spline finite element method to solve one-dimensional Burgers' equation. Kadalbajoo and Arora [12] have solved singular-perturbation problem using artificial viscosity to capture the exponential features on a uniform mesh with cubic B-spline collocation method. Kasi Vishwanadham and Krisnna [13, 14] have illustrated solution of fifth order and sixth order boundary value problems using quintic and septic B-splines collocation method with redefined basis functions. Kumar and Srivastava [16], have presented a survey of quadratic, quartic and octic spline techniques used to solve ordinary differential equations of different orders. Kumar and Srivastava [15], have presented a survey on computational techniques for solving boundary value problems by cubic, quintic, and sextic splines.

The first chapter of the thesis is introductory type and deals with the important ideas and historical back ground of the development of finding the solution of partial differential equations.

The chapters 2-7, deal with the numerical solutions of some linear and nonlinear partial differential equations using collocation and differential quadrature methods with B-spline basis functions.

Collocation method approximates the solution of differential equations in some form of linear combination of coordinate functions with linear coefficients. The idea is to choose a finite-dimensional space of candidate solutions (usually, polynomials up to a certain degree) and a number of points in the domain (called collocation points), and to select that solution which satisfies the given equation at the collocation points.

Differential quadrature method(DQM) is a higher order numerical technique for solving linear and nonlinear differential equations. The DQM can yield highly accurate solutions with relatively little computational effort and storage requirements. The method can easily and exactly satisfy a variety of boundary conditions and require much less formulation and programming effort. It have been pointed out that the DQM are basically equivalent to the collocation (pseudo-spectral) methods, but the DQM directly compute function values at grid points rather than spectral variables. Thus, they are more explicit and simple for some practical applications and especially advantageous for nonlinear problems. Moreover, the mathematical techniques involved in the method are not sophisticated. So the DQM is easily learned and used. The key procedure in the differential quadrature method is the determination of the weighting coefficients. DQM has been efficiently employed in a variety of problems occurring in engineering and physical sciences. For more detail, see [17].

Contents of the Thesis

The chapter-wise brief summary is given below:

Chapter 1. The first chapter is introductory type. It gives an introduction to B-spline collocation and B-spline differential quadrature method. Some properties of B-spline basis functions and existing literature review on methods using B-spline, is also discussed. Various degrees of B-splines function's definitions are extracted by using recursive function. Thomas algorithm and strong stability preserving Runge-Kutta (SSP-RK) methods of various stages and orders with their important properties are also briefly discussed.

Chapter 2. This chapter deals with numerical solution of nonlinear Klein-Gordon equation and Klein-Gordon-Schrödinger equations with Dirichlet and Neumann boundary conditions. One dimensional Klein-Gordon equation is given by

$$u_{tt} + \alpha u_{xx} + g(u) = f(x,t), \quad x \in (a,b)\ , t \geq 0$$

The parameter α is a known real constant, $f(x,t)$ is known analytic function and $g(u)$ is a nonlinear force, which may takes many forms such as: $\sin u$, $\sinh u$, $\sin u + \sin 2u$, $\sinh u + \sinh 2u$. The nonlinear Klein-Gordon equation describes a variety of physical phenomena such

as dislocations, ferroelectric and ferromagnetic domain walls, DNA dynamics, and Josephson junctions.
Yukawa-coupled Klein-Gordon-Schrödinger (KGS) equations is given by

$$\left.\begin{aligned} i\psi_t &= -\frac{1}{2}\psi_{xx} - \phi\psi, \\ \phi_{tt} &= \phi_{xx} - \phi + |\psi|^2, \end{aligned}\right\} \; x \in \mathbb{R},\ t \geq 0,$$

where $\psi(x,t)$ is a complex function representing a scalar neutron field and $\phi(x,t)$ is a real function representing a scalar neutral meson field.

Numerical solution of both the equations have been obtained using B-spline collocation method. The equations are decomposed into a system of partial differential equations, which are further converted to an amenable system of ODEs by using cubic B-spline basis functions for spatial variable and its derivatives. The system of equations so obtained have been solved by SSP-RK(54 or 43) scheme. Numerical results are presented for seven examples to demonstrate the usefulness and accuracy of approach. The results obtained by the presented method are of better accuracy than the results available in the earlier studies. The numerical approximate solutions of both the equations have been computed without using any transformation and linearization.

A part of this chapter has been published in **International Journal of Computer Mathematics (2014)**.

Chapter 3. In this chapter, modified cubic B-spline collocation method is discussed to find numerical solution of nonlinear sine-Gordon equation with Dirichlet boundary conditions.
We consider one-dimensional sine-Gordon equation

$$\frac{\partial^2 u}{\partial t^2} = \frac{\partial^2 u}{\partial x^2} - \sin(u), \qquad x \in (L_1, L_2),\ \ t \geq 0, \tag{2}$$

with suitable initial and boundary conditions.

The equation is decomposed into system of equations and modified cubic B-spline basis functions have been used for spatial variable and its derivatives, which results in an amenable system of ordinary differential equations. The resulting system of equation subsequently have been solved by SSP-RK54 scheme. Rate of convergence of method is computed and found to be approaching two. The efficacy of the approach has been confirmed with four numerical experiments, which shows the results obtained are acceptable and are in good agreement with earlier studies.

A part of this chapter has been published in **International Journal of Partial Differential Equations (2014).**

Chapter 4. This chapter is concerned with the numerical solution of one dimensional hyperbolic telegraph equation with Dirichlet and Neumann boundary conditions, using B-spline collocation method.
We consider second order one-dimensional hyperbolic telegraph equation

$$u_{tt}(x,t) + 2\alpha u_t(x,t) + \beta^2 u(x,t) = u_{xx}(x,t) + f(x,t), \qquad a \leq x \leq b,\ \ t \geq 0,$$

where α and β are known real constants. For $\alpha > 0$, $\beta = 0$ it represents a damped wave equation and for $\alpha > \beta > 0$ it is called as telegraph equation. We apply cubic B-splines collocation method, which produce a system of first order ordinary differential equations. This system is solved by SSP-RK54 scheme. Stability of scheme is discussed using matrix stability analysis and found to be unconditionally stable. Five illustrative examples are included to authenticate the effectiveness and applicability of the technique. The results accomplished with the developed approach are better in comparisons to available results in literature.

First part of this chapter has been published in **Applied Mathematics and Computation (2013)**.
Second part of this chapter has been published in **International Journal of Computational Mathematics (2014)**.

Chapter 5. This chapter deals with the numerical solution of two dimensional hyperbolic telegraph equation with Dirichlet and Neumann boundary conditions. We consider the following two dimensional hyperbolic telegraph equation

$$u_{tt}(x,y,t) + 2\alpha u_t(x,y,t) + \beta^2 u(x,y,t) = u_{xx}(x,y,t) + u_{yy}(x,y,t) + f(x,y,t), \\ (x,y,t) \in [a,b] \times [c,d] \times (0,T],$$

where, α, β are the real constants. For $\alpha > 0$, $\beta = 0$, it represents a damped wave equation and for $\alpha > 0$, $\beta > 0$, it is called telegraph equation.

In order to find the numerical solution of two dimensional hyperbolic telegraph equation, modified cubic B-spline basis functions based differential quadrature method is developed. The equation is converted into system of partial differential equation and further reduced into a system of ordinary differential equations using DQM. The obtained system of ODEs is then solved by a SSP-RK43 scheme. By employing DQM, accurate solutions can be obtained using less grid points in spatial domain. The stability of the scheme is studied using matrix stability analysis and found to be unconditionally stable. Seven example are solved to illustrate accuracy and efficiency of DQM.

A part of this chapter has been published in **Applied Mathematics and Computation (2014).**

Chapter 6. This chapter discusses the application of modified cubic B-spline differential quadrature method to find numerical solution of some nonlinear wave equations in one and two dimension with Dirichlet boundary conditions. We consider following mathematical model of the form

$$u_{tt} = u_{xx} + f(x,t,u,u_x,u_t), \;\; a \le x \le b, \; t \ge 0,$$

$$u_{tt} = u_{xx} + u_{yy} + f(x,y,t,u,u_x,u_y,u_t), \;\; (x,y,t) \in [a,b] \times [c,d] \times (0,T],$$

with suitable initial and boundary conditions. where f is some nonlinear expression in terms of u, u_x, u_t, u_y.

To obtain the numerical solution, above equations are decomposed into system of partial differential equations and modified cubic B-spline basis functions based differential quadrature method has been used for space discretization to obtain a system of nonlinear first order ordinary differential equations. The resulting system of equations have been solved using SSP-RK scheme. In numerical testing, the presented method is implemented on Vander pole type nonlinear wave equation, Dissipative nonlinear wave equation, telegraph equation. The accuracy of the approach has been confirmed with seven numerical experiments and results obtained are in good agreement with the exact solutions and earlier studies.

A part of this chapter has been accepted for publication in the proceedings of **3rd International conference on Advances in Computing, Communications and Informatics (ICACCI 2014)**.

Chapter 7. Chapter 7 presents the numerical solution of nonlinear two dimensional coupled Burgers' equation

$$\begin{aligned}
&\frac{\partial u}{\partial t} + u\frac{\partial u}{\partial x} + v\frac{\partial u}{\partial y} = \frac{1}{R}\left(\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2}\right) \\
&\frac{\partial v}{\partial t} + u\frac{\partial v}{\partial x} + v\frac{\partial v}{\partial y} = \frac{1}{R}\left(\frac{\partial^2 v}{\partial x^2} + \frac{\partial^2 v}{\partial y^2}\right) \qquad (x, y, t) \in [a, b] \times [c, d] \times (0, T],
\end{aligned}$$

where R is Reynolds number. This system models a large number of physical phenomena such as traffic flow, flow of a shock wave traveling in a viscous fluid, phenomena of turbulence, interaction between the non-linear convection process and the diffusive viscous processes, sedimentation of two kinds of particles in fluid suspensions under the effect of gravity. The equations are reduced into system of ordinary differential equations by modified cubic B-spline differential quadrature method and obtained system of nonlinear ODEs is then solved by SSP-RK scheme. The accuracy of the approach is tested on five test problems and computed results are compared with some earlier works. The results of computations indicate that modified cubic B-spline function based DQM combined with SSP-RK scheme gives more accurate results than earlier works with smaller grid points and larger time steps. Numerical results are computed for higher Reynolds number upto $R = 1500$. The strong point of the method is in ease to apply and less computational effort.

Chapter 8. This chapter addresses the conclusions, based on the present study.

References

[1] Schoenberg, I. J. Contributions to the problem of approximation of equidistant data by analytic functions. Quart. Appl. Math. 4 (1946) 45-99.

[2] Schumaker, L. L. Spline functions: basic theory. Pure and Applied Mathematics. A Wiley-Interscience Publication. John Wiley and Sons, Inc., New York, 1981.

[3] de Boor, C. A practical guide to splines. Revised edition. Applied Mathematical Sciences, 27. Springer-Verlag, New York, 2001.

[4] Ahlberg, J. H. and Nilson, E. N., Walsh, J. L. The theory of splines and their applications. Academic Press, New York-London 1967.

[5] Prenter, P. M. Splines and variational methods. Wiley-Interscience [John Wiley & Sons]. New York-London-Sydney. 1975.

[6] Ahlberg, J. H. and Ito, T. A collocation method for two-point boundary value problems. Math. Comp. 29 (1975), 761-776.

[7] Ali, A. H. A., Gardner, G. A. and Gardner, L. R. T. A collocation solution for Burgers' equation using cubic B-spline finite elements. Comput. Methods Appl. Mech. Engrg. 100(3) (1992), 325-337.

[8] Dağ, İ., Doan, A. and Saka, B. B-spline collocation methods for numerical solutions of the RLW equation. Int. J. Comput. Math. 80(6) (2003), 743-757.

[9] Kutluay, S., Esen, A. and Dag, I. Numerical solutions of the Burgers' equation by the least-squares quadratic B-spline finite element method. J. Comput. Appl. Math. 167(1) (2004), 21–33.

[10] Özis, T., Esen, A. and Kutluay, S. Numerical solution of Burgers' equation by quadratic B-spline finite elements. Appl. Math. Comput. 165(1) (2005), 237–249.

[11] Kadalbajoo, M. K. and Aggarwal, V. K. Fitted mesh B-spline collocation method for solving self-adjoint singularly perturbed boundary value problems. Appl. Math. Comput. 161 (2005) 973-987.

[12] Kadalbajoo, M. K. and Arora, P. B-spline collocation method for the singular-perturbation problem using artificial viscosity. Comput. Math. Appl. 57 (2009) 650-663.

[13] Kasi Viswanadham, K. N. S. and Krishna, P. M. Quintic B-splines Galerkin Method for fifth order boundary value problems. ARPN J. Eng. Appl. Sci. 5 (2010) 74-77.

[14] Kasi Viswanadham, K. N. S. and Krishna, P. M. Septic B-spline collocation method for sixth order boundary value problems. ARPN J. Eng. Appl. Sci. 5 (2010) 36-40.

[15] Kumar, M. and Srivastava, P. K. Computational techniques for solving differential equations by quadratic, quartic and octic spline. Adv. Eng. Softw. 39 (2008) 646-653.

[16] Kumar, M. and Srivastava, P. K. Computational techniques for solving differential equations by cubic, quintic, and sextic spline. Int. J. Comput. Methods Eng. Sci. Mech. 10(1) (2009) 108-115.

[17] Shu, C. Differential Quadrature and Its Application in Engineering. Springer-Verlag London Ltd., Great Britain, 2000.

Abstract

Collocation method is an emerging popular technique to solve initial and boundary value problems. It was developed to seek an approximate solution of differential equation in the form of linear combination of basis functions. The idea is to choose a finite dimensional space of candidate solution (usually, polynomials up to a certain degree) and a number of points in the domain (called collocation points), and to select that solution which satisfies the given equation at the collocation points.

Differential quadrature method (DQM) is a higher order numerical discretization technique for solving differential equations. DQM can provide the solution with a higher level of accuracy and with less computational effort. It has been also pointed out that the DQM is basically equivalent to the collocation (pseudo-spectral) method, in fact, DQM directly compute the functional value at the grid points rather than spectral variables. In this method, determination of the weighting coefficients is the key procedure which is of paramount importance. One of the advantage of this method is that it satisfies varieties of boundary conditions and require much less formulation and programming effort. Moreover, the mathematical techniques involved in the method are also not so sophisticated. And therefore, they are more explicit and simple for some practical applications and especially advantageous for nonlinear problems. So the DQM could be easily learned and successfully applied in the varieties of problems originated in the applied sciences.

In this research, we have developed collocation and differential quadrature methods with B-spline functions to solve linear/nonlinear partial differential equations (PDEs). The use of cubic B-spline basis functions in getting the numerical solutions of some partial differential equations is shown to provide an easy and simple algorithm. Strong stability preserving Runge-Kutta (SSP-RK) methods of different stages and order are also combined with these methods. In case of nonlinear PDEs, the numerical solutions can be obtained without using any transformation and linearization process of the equation. Therefore, the equations are solved more easily and elegantly using the developed techniques. These methods are simple and easy to use in comparison to other existing methods, e.g. finite element, finite volume and spectral methods, etc. All the chapters include several examples to demonstrate the applicability and efficiency of the presented methods. The chapter wise summary of the thesis is as follows:

Chapter 1 is preface which contains some relevant definitions, introduction to numerical techniques like the finite difference method, finite element method and existing literature review. B-spline functions of various degree are drawn out from the recursive formula. Some of the significant properties of B-spline functions are also discussed. Afterwards a brief introduction on B-spline functions, it contributes an introduction to collocation method, differential quadrature method and their execution process to solve linear/nonlinear PDEs. Strong stability preserving Runge-Kutta methods of various stages and orders with their significant attributes are also briefly talked about. The formulae for computing error norms and order of convergence are also discussed.

Chapter 2 deals with the numerical solutions of nonlinear Klein-Gordon equation and coupled Klein-Gordon-Schrödinger equation with Dirichlet and Neumann boundary conditions.

One dimensional Klein-Gordon equation is given by

$$u_{tt} + \alpha u_{xx} + g(u) = f(x,t), \quad x \in (a,b), \ t > 0,$$

with appropriate initial and boundary conditions. The parameter $\alpha < 0$ is a known real constant, $f(x,t)$ is known analytic function and $g(u)$ is a nonlinear force which may takes many forms such as: $\sin u$, $\sinh u$, $\sin u + \sin 2u$, $\sinh u + \sinh 2u$. The nonlinear Klein-Gordon equation describes a variety of physical phenomena such as dislocations, ferroelectric and ferromagnetic domain walls, DNA dynamics, and Josephson junctions.

Yukawa-coupled Klein-Gordon-Schrödinger (KGS) equation is given by

$$\begin{aligned} i\psi_t &= -\frac{1}{2}\psi_{xx} - \phi\psi, \\ \phi_{tt} &= \phi_{xx} - \phi + |\psi|^2, \ x \in \mathbb{R}, \ t > 0, \end{aligned}$$

with suitable initial and boundary conditions. Here ψ(x,t) is a complex function represents a scalar neutron field and ϕ(x,t) is a real function represents a scalar neutral meson field. The model describes the interaction between conservative complex neutron field and neutral meson Yukawa in quantum field theory and plays an important role in quantum physics.

Numerical solutions of both the equations are obtained using cubic B-spline collocation method. Modified cubic B-spline basis functions are used to handle the Dirichlet boundary conditions. The equations are decomposed into a system of partial differential equations, which is further converted to an amenable system of ordinary differential equations (ODEs). The obtained system of ODEs is solved by SSP-RK54 scheme. Numerical results are presented for six examples, to show the accuracy and utility of proposed approach. The approximate solutions of both the equations are computed without using any transformation and linearization process. The computed results are of better accuracy than earlier results available in the literature. The execution of this method is very easy and cost-effective.

A portion of this chapter has been published in **International Journal of Computer Mathematics (2014)**.

Chapter 3 addresses the modified cubic B-spline collocation method to find the numerical solution of nonlinear sine-Gordon equation with Dirichlet boundary conditions. One dimensional sine-Gordon equation turn out in many different applications such as propagation of fluxion in Josephson junctions, differential geometry, stability of fluid motion, nonlinear physics and applied sciences.
We consider one-dimensional nonlinear sine-Gordon equation

$$u_{tt} = u_{xx} - \sin(u), \qquad x \in (a,b), \;\; t > 0,$$

with suitable initial and boundary conditions.

The method is based on collocation of modified cubic B-splines over finite elements so that the continuity of the dependent variable and its first two derivatives throughout the solution range is preserved. The sine-Gordon equation is converted into a system of partial differential equations. Using modified cubic B-spline collocation method, we obtain a system of first order ordinary differential equations. Finally obtained system of ODEs is solved by SSP-RK54 scheme. The particular feature of SSP-RK scheme is that, it inherently perpetuates certain stability properties and maximum norm stability. It also controls spurious oscillations and non-linear instability during simulation. In terms of computational cost, SSP-RK schemes have drawn the same cost as traditional ODE solvers. To demonstrate the accuracy and usefulness of present scheme, four numerical examples are presented. The obtain results are of better precision and competent accuracy than the results available in the earlier works. The order of convergence of the scheme is also computed and found to be approaching two.

A part of this chapter has been published in **International Journal of Partial Differential Equations (2014)**.

Chapter 4 is concerned with the numerical solution of one dimensional hyperbolic telegraph equation with Dirichlet and Neumann boundary conditions, using cubic B-spline collocation method. The one dimensional hyperbolic telegraph equation is given by

$$u_{tt}(x,t) + 2\alpha u_t(x,t) + \beta^2 u(x,t) = u_{xx}(x,t) + f(x,t), \qquad x \in (a,b),\ \ t > 0,$$

where α and β are known real constants. For $\alpha > 0$, $\beta = 0$ it represents a damped wave equation and for $\alpha > \beta > 0$ it is called as telegraph equation.

The method is based on collocation of cubic B-spline basis functions over finite elements. Modified cubic B-spline basis functions are used to handle the Dirichlet boundary conditions. The use of B-spline basis functions for spatial variable and its derivatives, results in an amenable system of differential equations. The resulting system of equations is solved by SSP-RK54 scheme. Stability of scheme is discussed using matrix stability analysis and found unconditionally stable. The efficacy of approach is confirmed with four numerical experiments and the numerical results are found to be very good in comparison with the existing solutions found in the literature. The advantage of this scheme is that, it can be conveniently use to solve the complex problems and also capable of reducing the size of computational work.

First part of this chapter has been published in **Applied Mathematics and Computation (2013)**. Second part of this chapter has been published in **International**

Journal of Computational Mathematics (2014).

In chapter 5 we have proposed an efficient differential quadrature method, to find the numerical solution of two dimensional hyperbolic telegraph equation with Dirichlet and Neumann boundary conditions. The hyperbolic partial differential equations have significant role in formulating fundamental equations in atomic physics and are also very useful in understanding various phenomena in applied sciences like engineering, industry, aerospace as well as in chemistry and biology too.
Consider

$$\begin{aligned} u_{tt}(x,y,t)+2\alpha u_t(x,y,t)+\beta^2 u(x,y,t) =& u_{xx}(x,y,t)+u_{yy}(x,y,t)+f(x,y,t), \\ & (x,y,t)\in[a,b]\times[c,d]\times(0,T], \end{aligned}$$

with appropriate initial and boundary conditions. Here α, β are known real constants. For $\alpha>0$, $\beta=0$, it represents a damped wave equation and for $\alpha>0$, $\beta>0$, it is called telegraph equation.

In order to find the numerical solutions, modified cubic B-spline basis functions based differential quadrature method is developed. The equation is converted into a system of partial differential equations and further reduced into a system of ordinary differential equations using DQM. SSP-RK43 scheme is used to solve the obtained system of ODEs. By employing DQM, accurate solutions can be obtained using fewer grid points in spatial domain. The stability of the scheme is studied using matrix stability analysis and found to be unconditionally stable. The efficacy of proposed approach is confirmed with seven numerical experiments, where comparisons are made with some earlier works. It is observed that the obtained results are acceptable and are in good agreement with earlier studies. However, we obtain these results in much less CPU time. The method is very simple, efficient and produces very accurate numerical results in considerably smaller number of nodes and hence saves computational effort.

A part of this chapter has been published in **Applied Mathematics and Computation (2014).**

Chapter 6 discusses the application of modified cubic B-spline differential quadrature method to find the numerical solutions of some nonlinear wave equations in one and two dimensions with Dirichlet boundary conditions. We consider

$$u_{tt} = u_{xx} + f(x, t, u, u_x, u_t), \;\; x \in (a, b), \; t > 0,$$
$$u_{tt} = u_{xx} + u_{yy} + f(x, y, t, u, u_x, u_y, u_t), \;\; (x, y, t) \in [a, b] \times [c, d] \times (0, T],$$

with suitable initial and boundary conditions. Here f is some nonlinear expression in terms of u, u_x, u_t, u_y. Nonlinear wave equations are arise in many physical and engineering applications such as continuum physics, mixed models of transonic flows, fluid dynamics and many other fields of science and engineering.

To obtain the numerical solutions, above equations are decomposed into a system of partial differential equations. Modified cubic B-spline basis functions based differential quadrature method is used for space discretization to obtain a system of nonlinear first order ordinary differential equations. The resulting system of equations is solved using SSP-RK43 scheme. In numerical testing, the method is implemented on Vander pole type nonlinear wave equation, Dissipative nonlinear wave equation and Telegraph equation. The obtained numerical results are found to be very good in comparison with the existing solutions found in the literature. The numerical solutions of nonlinear equations are computed without linearizing the nonlinear term. The order of convergence of method is also computed and found to be two.

A part of this chapter has been published in the proceeding of **3rd International Conference on Advances in Computing, Communications and Informatics (ICACCI 2014)(IEEE Xplore)**.

Chapter 7 presents the numerical solution of two dimensional nonlinear coupled Burgers' equation

$$\begin{aligned}
\frac{\partial u}{\partial t} + u\frac{\partial u}{\partial x} + v\frac{\partial u}{\partial y} &= \frac{1}{R}\left(\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2}\right) \\
\frac{\partial v}{\partial t} + u\frac{\partial v}{\partial x} + v\frac{\partial v}{\partial y} &= \frac{1}{R}\left(\frac{\partial^2 v}{\partial x^2} + \frac{\partial^2 v}{\partial y^2}\right) \quad (x, y, t) \in [a, b] \times [c, d] \times (0, T],
\end{aligned}$$

where R is Reynolds number. This system models a large number of physical phenomena such as traffic flow, flow of a shock wave traveling in a viscous fluid, phenomena of turbulence, interaction between the non-linear convection process and the diffusive viscous process, sedimentation of two kinds of particles in fluid suspensions under the effect of gravity. Modified cubic B-spline differential quadrature method (MCB-DQM) is used to discretized the spatial derivatives of coupled Burgers' equation and reduces it into a system of first order ordinary differential equations. The obtained system of equations is solved by SSP-RK43 scheme. The accuracy of the approach is tested on five test problems and computed results are compared with some earlier works. The results indicate that MCB-DQM combined with SSP-RK scheme gives more accurate results than earlier works with less computational cost. Numerical results are computed for higher Reynolds number up to $R = 1500$. The strong points of the method are in ease to apply and less computational effort.

Finally, in **chapter 8** conclusions are drawn based on the present study and future research work is suggested, in this direction.

List of Research Papers

1. R. C. Mittal, Rachna Bhatia, Numerical solution of second order one dimensional hyperbolic telegraph equation by cubic B-spline collocation method, ***Applied Mathematics and Computation*** 220 (2013), 496-506.

2. R. C. Mittal, Rachna Bhatia, A numerical study of two dimensional hyperbolic telegraph equation by modified B-spline differential quadrature method, ***Applied Mathematics and Computation*** 244 (2014), 976-997.

3. R. C. Mittal, Rachna Bhatia, Numerical solution of nonlinear system of Klein-Gordon equations using cubic B-spline collocation method, ***International Journal of Computer Mathematics*** (2014), 1-21 (Taylor & Francis).

4. R. C. Mittal, Rachna Bhatia, Numerical solution of nonlinear sine-Gordon equation by modified cubic B-spline collocation method, ***International Journal of Partial Differential Equations*** vol. 2014, Article ID 343497, 8 pages, 2014.

5. R. C. Mittal, Rachna Bhatia, A collocation method for numerical solution of hyperbolic telegraph equation with Neumann boundary conditions, ***International Journal of Computational Mathematics*** vol. 2014, Article ID 526814, 9 pages, 2014.

6. R. C. Mittal, Rachna Bhatia, Numerical solution of some nonlinear wave equations using modified cubic B-spline differential quadrature method, Proceeding of ***International Conference on Advances in Computing, Communication and Informatics***(ICACCI 2014) pages 433-439 (IEEE Xplore).

7. R. C. Mittal, Rachna Bhatia, A numerical study of two dimensional coupled Burgers' equation using differential quadrature method (Under Review).

8. R. C. Mittal, Rachna Bhatia, Numerical solution of two dimensional nonlinear wave equations using differential quadrature method (Under Review)

Acknowledgements

This doctoral work would not have been possible without the support and encouragement of numerous people, including my family, well wishers and my friends. I would wish to show my gratitude to all those who have facilitated me to finish this thesis.

First and foremost, I would like to thank my supervisor and mentor Dr. R. C. Mittal, Professor and Head, Department of Mathematics, I.I.T. Roorkee, Roorkee for his expert guidance, valuable suggestions, support and encouragement throughout my research work. Without his active and expert guidance, it would be difficult for me to finish this thesis. I will be obliged to him throughout my life.

I sincerely thank my research committee members Prof. Kusum Deep, DRC Chairperson, Prof. P. N. Agarwal, former DRC Chairperson, Prof. Rama Bhargava, Prof. M. Srikhande and all the faculty members for their guidance, cooperation and many valuable comments that have helped me in improving the quality of my research work.

The inspiration, support, cooperation and forbearance which I have picked up from my friends Dr. R. K. Jain, Satish Kumar, Anu Bala, Shashi Sharma, Asha Rani, Sumita Dahiya, Binod Singh, Navneet Kumar, Shefali Thakur, Rama Krishna, Panithi Rama Devi, Sanjukta, Shaila Bantanur, Sujeet is beyond the scope of any identification, even then I would like to express my earnest gratitude to them.

My endless gratitude goes to my parents for their blessings, love, patience and moral support. I also cannot forget love, affection and care of my beloved and caring hubby Amit and my naughty son Arush Singh 'Azu' who always make me smile even in difficult times during the four years long study. My brother Deepak Bhatia deserves the special thanks, without his unconditional support and encouragement this would not have been possible. I would also like to express my gratitude towards my sisters Bhawana Jethi, Mona Bhatia and Neeru Bhatia. It is difficult to find adequate words to express my appreciation for the help given by them. I would also like to thank my neighbors Shefali & Navneet Chauhan and Panithi Rama Devi & Rama Krishna to make the four years long stay as a lifetime memory.

Last but not the least, my greatest regards to the Almighty God for bestowing upon me the courage to face the complexities of life and complete this thesis successfully.

Roorkee (Rachna Bhatia)

December , 2014

Table of Contents

List of Tables

List of Figures

Chapter 1

Introduction

1.1 Overview

Partial differential equations are used in modeling of many physical, chemical and biological phenomena. Besides these its uses have also spread into financial forecasting, image processing, economics and other fields as well. There are many theoretical results on existence and uniqueness, but only the simplest specific problems can be solved explicitly. Since limited classes of the equations are solved by analytical means, we usually construct approximate numerical solutions of these differential equations especially of nonlinear ones which are of practical importance. Various numerical techniques have been promulgated for finding the solutions of partial differential equations among which the two most popular are (a) finite difference method and (b) finite element method.

Finite Difference Method [75, 163]

In the numerical solutions of partial differential equations and their applications, finite difference methods are often dominant. In this method, the derivatives appearing in a PDEs are approximated by sums and differences of function values at a set of discrete points, usually evenly spaced with respect to each independent variable. Finally, a large algebraic system of equations are obtained, instead of solving the given differential equation. The appropriate types of differencing scheme and proper method for solutions are chosen in different applications. The accuracy of the method is based on the refinement of the grid points, where the solution is evaluated. For one dimensional case these methods can be easily formulated but in higher dimensions, meshes should be structured in either two or three dimensions.

Finite Element Method [75, 173]

The 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. In the finite element method, the solution is expanded in terms of a set of basis functions superimposed on a family of elements, which in turn span the region in which the solution of the differential equation is under study. In this method, the domain is divided into a finite number of subdomains, which may be of different shapes i.e. triangular, rectangular, curvilinear, etc. In addition, the accuracy of FEM programs can be improve by using adaptive meshing procedures and nonuniform unstructured meshes. Some limitations should be placed in the selection of the shape functions to guarantee the fact that with an arbitrary number of shape functions, the exact solution is approximated best, and in the limit we must obtain the exact solution. A good approximation is obtained by the residual formulation, where the residual is formulated as the difference of the analytical solution and the calculated numerical solution. This residual is weighted over the simulation domain and integrated with the requirement that the integral vanishes with a set of linearly independent weighting functions.

In this work, we have implemented the collocation and differential quadrature methods using third degree B-spline basis functions to solve linear and nonlinear partial differential equations. The combination of B-splines with these methods is shown to provide a simple and effective solution procedure to solve partial differential equations.

The first chapter of the thesis is prolegomenon, which deals with the important ideas and historical background of the development of finding the solutions of partial differential equations. It also provides an introduction to B-spline basis functions, collocation and differential quadrature methods along with the methodology used to solve PDEs. B-spline functions of various degree are extracted by using recursive formula. Some of the key features of these functions are also discussed. Strong stability preserving Runge-Kutta (SSP-RK) methods of various stages and orders with their important properties are also briefly discussed. The formulae for computing the error norms are also presented.

1.2 Literature Survey

Splines have many implementations in finding the numerical solutions of a variety of problems in applied mathematics and engineering. Some of them are, Computer-Aided Geometric Design (CAGD), Data fitting, Integro-differential equations, Function approximation, Wavelets and so on. Some of the research papers that have made outstanding contributions to the development of splines, include Birkhoff and Garabedian [22], Loscalzo and Talbot [136], Rubin and Khosla [185], Sastry [188], Schoenberg [190]. Some of the books which discuss splines thoroughly are Ahlberg et al. [5], de Boor [40], Prenter [168], Schumaker [191].

A lot of work has been reported in the literature using spline functions of various degree. Some of the earliest papers which demonstrate the approximate methods using spline functions for solution of ordinary and partial differential equations, include Albasiny and Hoskins [7], Bickley [21], Crank and Gupta [36], Jain and Aziz [83, 84], Rubin and Khosla [185], Sastry [188], Usmani [201], Usmani and Sakai [202], Usmani and Warsi [203]. Today, there are number of research papers in this subject and it remains an active research area. For example, Jain et al. [85] have developed implicit finite difference schemes for numerical integration of one-and two-dimensional scalar hyperbolic equations and a system of conservation laws using the spline (in compression) function approximation. Singular two-point boundary value problems have been solved by Iyengar and Jain [80], using spline difference methods and by Ravi Kanth and Reddy [183], using collocation method with cubic spline functions. Higher order splines are also used to solve higher-order boundary value problems. For instance, collocation method has been developed for the approximate solution of eighth order linear special case boundary value problem using nonic spline by Akram and Siddiqi [6]. Solution of twelfth-order boundary value problems has been discussed by Siddiqi and Twizell [195], using polynomial spline of degree twelve.

B-splines are the smoothest interpolating functions compared with other piecewise polynomial interpolating functions. These functions have been used as a basis functions in finite element method, collocation method and differential quadrature method, for constructing numerical methods for the solutions of PDEs that occur in various engineering applications. In many papers various techniques using quadratic, cubic, quartic, quintic, sextic, septic and higher degree B-splines have been discussed for the numerical solutions of linear/nonlinear ODEs and PDEs. Ahlberg and Ito [4] have presented a solution of two-point boundary value problems using collocation method with cubic, quintic and septic B-splines. Ali et al. [9] have developed a collocation method for Burgers' equation using cubic B-spline finite elements. Kasi Vishwanadham and Koneru [98] have proposed a B-spline finite element method for one-dimensional and two-dimensional time dependent problems. In a series of papers by Caglar et al. [30, 31], initial and boundary value problems of fifth and third order have been solved using B-spline basis functions of sixth and fourth degree, respectively. Dağ and Özer [38] have proposed a approximation of RLW equation by the least square cubic B-spline finite element method. Dağ et al. [37] have solved RLW equation using cubic B-spline collocation method. Kumar [114] has applied a second order spline finite difference method and fourth-order spline finite difference method based on a non-uniform mesh to find the numerical solutions of singular two-point boundary value problems. Dağ and Saka [39] have proposed a cubic B-spline collocation method to obtain numerical solutions of EW equation. Kutluay et al. [121] have given a least-square quadratic B-spline finite element method to solve Burgers' equation. A B-spline finite element method for thermistor problem with modified electrical conductivity has been developed by Kutluay and Esen [119]. Raslan [179, 180] has exhibited a collocation method for solving EW equation using quintic and quartic B-splines respectively, along with Runge-Kutta method. Saka and Dağ [187] have presented a collocation method using cubic B-spline for solving RLW equation. Özis et al. [164] have used a Galerkin quadratic B-spline finite element method to solve one-dimensional Burgers' equation. Kadalbajoo and Aggarwal [92] have solved a self-adjoint singularly perturbed boundary value problems using B-spline collocation method. Raslan [181] has proposed a computational collocation method for RLW equation. Kumar [115] has developed a fourth-order spline finite difference method based on a non-uniform mesh to find the numerical solution of singular two-point boundary value problems. Dehghan and Lakestani [45] have solved a nonlinear system of second-order boundary value problems using cubic B-spline scaling functions. Kadalbajoo and Yadaw [96] and Kadalbajoo and

Gupta [95] have developed a cubic B-spline collocation method for solving singularly perturbed convection-diffusion boundary value problems. Kumar and Srivastava [116] have presented a survey of quadratic, quartic and octic spline techniques, which are used to solve ordinary differential equations of different orders. Kadalbajoo and Arora [93] have solved singular-perturbation problem using artificial viscosity to capture the exponential features on a uniform mesh with cubic B-spline collocation method. Lakestani and Dehghan [122] have used the cubic B-spline scaling functions to solve Fokker-Planck equation. Kumar and Srivastava [117] have described a survey on computational techniques for solving boundary value problems by cubic, quintic, and sextic splines, which includes a large amount of work done in the area of application of spline functions during 2000-2007. A Taylor-Galerkin B-spline finite element method for one-dimensional advection-diffusion equation has been developed by Kadalbajoo and Arora [94]. Lakestani and Dehghan [124] have found the numerical solution of Riccati equation using the cubic B-spline scaling functions and Chebyshev cardinal functions. Kasi Vishwanadham and Krishnna [99, 100] have illustrated the solutions of fifth order and sixth order boundary value problems using quintic and septic B-splines collocation method with redefined basis functions. Mittal and Arora [139] have presented the numerical solution of Kuramoto-Sivashinsky equation using B-spline collocation method. Mittal and Jain [141] have proposed cubic B-spline and quintic B-spline collocation methods with redefined basis functions to solve fourth order parabolic partial differential equations. Mittal and Arora [138] have developed a cubic B-spline collocation method to find stable and accurate numerical solutions of Fisher's equation. A quartic B-spline differential quadrature method for solving one dimensional Burgers' equation has been proposed by Korkmaz et al. [112]. Mittal and Arora [140] have used a cubic B-spline collocation method to solve coupled Burgers' equation. Mittal and Jain [145] have proposed a redefined cubic B-spline collocation method for solving convection-diffusion equation. Lakestani and Dehghan [125] have found numerical solution of generalized Kuramoto-Sivashinsky equation using B-spline functions. Mittal and Jain [143] have solved nonlinear parabolic partial differential equations with Neumann boundary conditions, using cubic B-spline collocation method. Lakestani and Dehghan [126] have proposed four techniques based on the B-spline expansion and the collocation approach for the numerical solution of the Lane-Emden equation. Quintic B-spline collocation method to solve some Rosenau type nonlinear evolution equations has been developed by Mittal and Jain [142]. Lakestani et al. [127] have constructed the operational matrix of fractional derivatives using B-spline functions. Arora and Singh [11] have developed a B-spline differential quadrature method to solve one dimensional Burgers' equation. Korkmaz and Dağ [110, 111] have developed a cubic B-spline differential quadrature method to solve one dimensional Burgers' and boundary forced RLW equations, respectively. Mittal and Jain [144, 146] have developed a modified cubic B-spline collocation method for solving Burgers' and Fisher's equations, respectively. A finite element Galerkin method with redefined sextic B-spline functions has been developed by Kasi Viswanadham and Ballem [97] to solve a general tenth order boundary value problems.

1.3 Idea of Splines

The first reference to mathematical splines was made in an interesting paper by Schoenberg [189] in 1946. Splines are types of curves, originally developed for the shipbuilding industry in the days before computer modeling. Naval architects needed a way to draw a smooth curve through a set of points. The solution was to place metal weights (called knots) at the control points, and bend a thin metal or wooden beam (called a spline) through the weights. The physics of the bending spline meant that the influence of each weight was higher at the point of contact and decreased smoothly along the spline. For more control over a particular region, the draftsman just added more weights. This scheme had obvious problems with the exchange of data. There was a need for mathematically describing the shape of the curve. Splines have become more important with the advent of computers. They have been used primarily as a replacement for polynomials in interpolation and as a tool for creating a smooth and flexible shapes in computer graphics.

Therefore, the problem of working with higher degree polynomials can be resolved by using the piecewise polynomials. Instead of using polynomial for the entire domain, the function can be approximated by several polynomials defined over the sub-domains. The piecewise polynomial approximation allows us to construct highly accurate approximations, but some approximation functions are not smooth at the point connecting separate piecewise polynomial approximations. 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. So the basic idea of splines is to construct a piecewise polynomial approximation which not only interpolate the given data or function values given but it is also smooth i.e. it must be continuously differentiable to a certain degree.

1.4 B-splines

The first reference to the word B-spline was given by Schoenberg [189] in 1946, who described it as a smooth piecewise polynomial approximation. B-spline or basis spline, is define as a spline function that has minimal support with respect to a given degree, smoothness, and domain partition. Let X be a set of $N+1$ non-decreasing real numbers such that $x_0 \leq x_1 \leq \cdots \leq x_{N-1} \leq x_N$, where $x_j^{'}$s are called knots. The half open interval $[x_j, x_{j+1})$ is called the j^{th} knot span. If the knots are equally spaced i.e. $x_{j+1} - x_j = h =$ constant, for $0 \leq j \leq N-1$, then the knot vector or the knot sequence is said to be uniform; otherwise non-uniform.

The original definition of the B-spline basis functions was given by Schumaker [191], using the idea of divided differences. In 1970's, a recurrence relation was developed independently by de Boor [40] to calculate the B-spline basis functions of higher degree. By applying the Leibnitz's theorem, de Boor was able to derive the following formula for j^{th} B-spline basis function of d^{th} degree, in a recursive way as follows:

$$B_{j,d}(x) = \frac{(x - x_j)}{(x_{j+d} - x_j)} B_{j,d-1}(x) + \frac{(x_{j+d+1} - x)}{(x_{j+d+1} - x_{j+1})} B_{j+1,d-1}(x). \tag{1.1}$$

This formula is known as de Boor's recursion formula. Where $B_{j,d}(x)$ defines a j^{th} B-spline basis function of d^{th} degree and shows that the B-spline basis functions of any arbitrary degree can be stably evaluated as a linear combination of basis functions of lower degree. The recurrence relation starts with the first degree B-spline and builds the basis functions of higher degree.

1.4.1 Zero Degree B-spline

The zero degree B-spline is one of the simplest basis function and is given as

$$B_{j,0} = \begin{cases} 1, & x \in [x_j, x_{j+1}) \\ 0, & \text{otherwise} \end{cases} \tag{1.2}$$

Thus for $d = 0$, the B-spline function is just a step function i.e. it is equal to zero at all points except on the half open interval $[x_j, x_{j+1})$.

1.4.2 First Degree B-spline

The first degree B-splines are named as linear B-spline and can be obtained using the de Boor recursion formula (1.1) with $d = 1$ and using the definition of zero degree B-spline (1.2), as follows

$$B_{j,1}(x) = \frac{(x - x_j)}{(x_{j+1} - x_j)} B_{j,0}(x) + \frac{(x_{j+2} - x)}{(x_{j+2} - x_{j+1})} B_{j+1,0}(x)$$

$$B_{j,1} = \begin{cases} \frac{(x-x_j)}{(x_{j+1}-x_j)}, & x \in [x_j, x_{j+1}) \\ \frac{(x_{j+2}-x)}{(x_{j+2}-x_{j+1})}, & x \in [x_{j+1}, x_{j+2}) \\ 0 & \text{otherwise} \end{cases} \tag{1.3}$$

The first degree B-splines are like a Hat or Tent function and are non-zero for two consecutive intervals $[x_j, x_{j+1})$ and $[x_{j+1}, x_{j+2})$.

1.4.3 Second Degree B-spline

The second degree B-splines are also called as quadratic B-spline and can be obtained by using the de Boor recursion formula and linear B-splines, for $d = 2$

$$\begin{aligned} B_{j,2}(x) = \frac{(x - x_j)}{(x_{j+2} - x_j)} &\left(\frac{(x - x_j)}{(x_{j+1} - x_j)} B_{j,0}(x) + \frac{(x_{j+2} - x)}{(x_{j+2} - x_{j+1})} B_{j+1,0}(x) \right) \\ &+ \frac{(x_{j+3} - x)}{(x_{j+3} - x_{j+1})} \left(\frac{(x - x_{j+1})}{(x_{j+2} - x_{j+1})} B_{j+1,0}(x) + \frac{(x_{j+3} - x)}{(x_{j+3} - x_{j+2})} B_{j+2,0}(x) \right) \end{aligned}$$

$$\begin{aligned} =& \frac{(x - x_j)^2}{(x_{j+2} - x_j)(x_{j+1} - x_j)} B_{j,0}(x) + \frac{(x_{j+3} - x)^2}{(x_{j+3} - x_{j+1})(x_{j+3} - x_{j+2})} B_{j+2,0}(x) \\ &+ \left(\frac{(x - x_j)(x_{j+2} - x)}{(x_{j+2} - x_j)(x_{j+2} - x_{j+1})} + \frac{(x_{j+3} - x)(x - x_{j+1})}{(x_{j+3} - x_{j+1})(x_{j+2} - x_{j+1})} \right) B_{j+1,0}(x) \end{aligned}$$

The explicit formula for quadratic B-spline is given by

$$B_{j,2} = \frac{1}{2h^2} \begin{cases} (x - x_j)^2, & x \in [x_j, x_{j+1}) \\ (x - x_j)(x_{j+2} - x) + (x_{j+3} - x)(x - x_{j+1}), & x \in [x_{j+1}, x_{j+2}) \\ (x_{j+3} - x)^2, & x \in [x_{j+2}, x_{j+3}) \\ 0 & \text{otherwise} \end{cases} \tag{1.4}$$

Quadratic B-splines $B_{j,2}(x)$ are non-zero on three knot span $[x_j, x_{j+1})$, $[x_{j+1}, x_{j+2})$ and $[x_{j+2}, x_{j+3})$.

The coefficients of quadratic B-splines and its derivatives are reported in Table-1.1 and graph of quadratic B-spline is depicted in Figure-1.2.

1.4.4 Third Degree B-spline

The formula for cubic B-spline $B_{j,3}(x)$ at the knot is given by

$$B_{j,3} = \frac{1}{h^3}\begin{cases} (x - x_{j-2})^3, & x \in [x_{j-2}, x_{j-1}) \\ (x - x_{j-2})^3 - 4(x - x_{j-1})^3, & x \in [x_{j-1}, x_j) \\ (x_{j+2} - x)^3 - 4(x_{j+1} - x)^3, & x \in [x_j, x_{j+1}) \\ (x_{j+2} - x)^3, & x \in [x_{j+1}, x_{j+2}) \\ 0 & \text{otherwise} \end{cases} \tag{1.5}$$

It is evident that the nonzero part of $B_{j,3}(x)$ is localized to a small neighborhood of x_j namely in the interval $[x_{j-2}, x_{j+2})$ i.e. each cubic B-spline covers four intervals namely $[x_{j-2}, x_{j-1})$, $[x_{j-1}, x_j)$, $[x_j, x_{j+1})$, $[x_{j+1}, x_{j+2})$, so that an interval is covered by four cubic B-splines.

The coefficients of cubic B-splines and its derivatives are presented in Table-1.2 and Figure-1.3 shows the graph of cubic B-spline.

1.4.5 Properties of B-splines

Some of the important properties of B-spline functions are given below:

- **Non-negativity**: For all j,d and x, $B_{j,d}(x)$ is non-negative in the interval $[x_j, x_{j+d+1})$. The closed interval $[x_j, x_{j+d+1}]$ is called the support of $B_{j,d}(x)$.

- **Local Support**: If $x \notin [x_j, x_{j+d+1})$ then $B_{j,d}(x) = 0$. In particular if $x_j = x_{j+d+1}$, then $B_{j,d}(x) = 0$ i.e. B-splines have minimal compact support. More precisely, B-spline function of degree d is nonzero over $d + 1$ consecutive intervals.

- **Partition of Unity**: The sum of all non-zero d^{th} degree basis functions on span $[x_j, x_{j+1})$ is one.

- **Smoothness**: For degree $d \geq 1$, the B-splines belong to continuity class C^{d-1}.

- B-spline representation of functions is shape preserving. In particular, it does not exhibit spurious oscillations also known as Runge phenomenon, which is usually experienced with global polynomial approximations.
- Boundary conditions of various types are easily incorporated with B-spline functions.

1.5 Modified Cubic B-spline Basis Functions

In collocation method with cubic B-splines, there is a necessity to modify the basis functions into a new set of basis functions to handle with Dirichlet boundary conditions. The cubic B-splines set has been modified into a new set of basis functions in such manner that the number of basis functions matches with the selected points in the given domain and we obtain a diagonally dominant system of differential equations for handling with given Dirichlet boundary conditions [144, 146].

In the domain $[a, b]$ with uniform partition as $a = x_a < x_{a+h} < \cdots < x_{a+(N-1)h} < x_{a+Nh} = b$, the modified cubic B-spline basis functions are given as

$$\left.\begin{aligned} &\tilde{B}_a(x) = B_a(x) + 2B_{a-h}(x),\\ &\tilde{B}_{a+h}(x) = B_{a+h}(x) - B_{a-h}(x),\\ &\tilde{B}_j(x) = B_j(x),\ \ j = a+2h, \ldots, a+(N-2)h,\\ &\tilde{B}_{a+(N-1)h}(x) = B_{a+(N-1)h}(x) - B_{a+(N+1)h}(x),\\ &\tilde{B}_{a+Nh}(x) = B_{a+Nh}(x) + 2B_{a+(N+1)h}(x). \end{aligned}\right\} \tag{1.6}$$

1.6 Collocation Method

Collocation method [19, 100, 143, 144, 187] approximates the solution of differential equations in some form of a linear combination of basis functions. Consider a differential equation

$$f(x, u, u_x, u_{xx}) = 0, \tag{1.7}$$

where u is define in the domain $[a, b]$ with the following boundary conditions

$$u(a) = f_0, \quad u(b) = f_1. \tag{1.8}$$

The method begins with a proper choice of basis functions $\phi_0, \phi_1, \ldots, \phi_N$ and a set of points $a = x_0 < x_1 < x_2 < \cdots < x_N = b$, called collocation points.
The approximate solution is assumed as

$$U = \sum_{j=0}^{N} c_j \phi_j(x), \tag{1.9}$$

where N is considered as the quality parameter. As N increases, the error in the approximation must reduce. The method requires that the above approximation should satisfy the differential equation at each node and also the boundary conditions.
For this, residual must be zero at the selected points, which is defined as

$$r(x, u, u_x, u_{xx}) = f(x, u, u_x, u_{xx}) - f(x, U, U_x, U_{xx}). \tag{1.10}$$

Much attention has been given in the literature for the choice of the knots x_j and the basis functions ϕ_j and it has been found that spline curves or piece-wise polynomials are more effective in representing the solution of the differential equations than pure polynomials.

1.7 B-spline Collocation Method

B-spline collocation method is a method for finding the numerical solutions of differential equations. The idea is to choose a finite-dimensional space of candidate solutions (B-splines up to a certain degree) and a number of points within the domain (called collocation points), and to select that solution which satisfies the given equation at each collocation point.

We consider the basis functions for which the knots are equidistant. In such a case, the B-spline is said to be uniform. A fundamental theorem states that every spline function of a given degree, smoothness, and domain partition can be represented as a linear combination of B-splines of the same degree and smoothness, and over the same partition. This attribute immediately implies the banded structure of matrices occurring in interpolation and collocation problems. As a result, these matrices can be efficiently stored in a computer and inverted by Gaussian elimination without pivoting. The problem of stable calculation of polynomial B-splines and their derivatives may be found in Prenter [168].

A number of choices must be made while applying B-spline collocation method to approximate the solutions of differential equations. These include the particular choice of B-spline basis functions, the order of the B-spline functions, the regularity of the B-splines curve and the location of collocation points.

1.7.1 Cubic B-spline Collocation Method

Consider a mesh $a = x_0 < x_1 < \cdots < x_{N-1} < x_N = b$ as a uniform partition of the solution domain $a \leq x \leq b$ by the knots x_m, with the step length $h = x_{m+1} - x_m$, where $m = 0, 1, \ldots, N-1$.

In this method, the approximate solution of a partial differential equation can be written as linear combination of cubic B-spline basis functions as

$$U(x,t) = \sum_{j=-1}^{N+1} c_j(t) B_j(x), \tag{1.11}$$

$$U(x_j,t) = c_{j-1}\, B_{j-1}(x_j) + c_j\, B_j(x_j) + c_{j+1}\, B_{j+1}(x_j), \tag{1.12}$$

where the set of functions $\{B_{-1}, B_0, B_1, \ldots, B_{N-1}, B_N, B_{N+1}\}$ forms a basis for the function define over the region $a \leq x \leq b$ with the obvious adjustment of the boundary base functions to avoid undefined knots.

Using Table-1.2 and approximate solution (1.11), the approximate values of $U(x,t)$ and its two derivatives at the knot $x = x_j$ are computed as

$$\begin{aligned} U_j &= c_{j-1} + 4c_j + c_{j+1}, \\ U_j' &= \frac{3}{h}(c_{j+1} - c_{j-1}), \\ U_j'' &= \frac{6}{h^2}(c_{j-1} - 2c_j + c_{j+1}). \end{aligned} \tag{1.13}$$

In this work, collocation method using B-spline functions is shown to provide an easy and simple algorithm to solve one dimensional linear and nonlinear PDEs. The developed method is based on collocation of B-splines over finite elements so that we have continuity of the dependent variable and its derivatives throughout the solution range. Modified cubic B-spline basis functions have been used for solving the PDEs, when Dirichlet boundary conditions are prescribed. SSP-RK methods are also combined with B-spline collocation method. The special feature of this approach for solving nonlinear PDEs is that there is no need of any transformation and without using any linearization procedure, the nonlinearity can be treated. B-spline collocation method provides a piecewise continuous closed-form solution, which can be used to obtain the solution at any point of the domain. This method is simple and easy to apply in comparison to other existing methods, e.g. finite element, finite volume, etc.

1.8 Differential Quadrature Method

Differential quadrature method (DQM) is a higher order numerical discretization technique for solving linear and nonlinear differential equations. Numerical methods like finite difference, finite element are lower order method as they require a large number of grid points to get satisfactory results. Consequently, the requirements for CPU time and storage are often unnecessarily large in such cases. The above-mentioned shortcomings are not inherent in the differential quadrature method. DQM can provide the solution with a higher degree of accuracy with less computational effort. It has been also pointed out that the DQM is basically equivalent to the collocation (pseudo-spectral) method, in fact, DQM directly compute the functional value at grid points rather than spectral variables. In this method, determination of the weighting coefficients is the key procedure which is of paramount importance. One of the advantage of this method is that it satisfies a number of boundary conditions and require much less formulation and programming effort. Moreover, the mathematical techniques involved in the method are also not so sophisticated. And therefore, they are more explicit and easy for some practical applications and especially advantageous for nonlinear problems. So DQM could be easily learned and successfully applied in the varieties of problems originated in the applied sciences.

Richard Bellman and his associates [17] have introduced DQM in the early 1970's that require less number of grid points. DQM approximates the partial derivative of a function at any grid point as a linear combination of all the function values along a mesh line. Weighting coefficients determination is the key process of this method. Progress in the DQM took place after the late 1980's. As a consequence, the DQM has gone forth as a powerful tool in the past ten years. It has been applied efficaciously in a variety of problems encountered in engineering and applied sciences. For details, readers may refer to [192].

1.8.1 Approximation of First Order Derivatives

Consider a one dimensional problem over a closed interval $[a, b]$. Suppose there are N grid points with coordinates as $a = x_1, x_2, \ldots, x_N = b$. Bellman and his associates [17] supposed that a function $f(x)$ is sufficiently smooth over the interval $[a, b]$, so that its first order derivative $f_x^{(1)}(x)$ at a grid point $x = x_i$ can be approximated by the following relation

$$f_x^{(1)}(x_i) = \sum_{j=1}^{N} a_{ij}^{(1)} f(x_j), \;\; i = 1, 2, \ldots, N, \tag{1.14}$$

where $f(x_j)$ represents the function value at a grid point x_j, $f_x^{(1)}(x_i)$ indicates the first order derivative of $f(x)$ at point x_i and $a_{ij}^{(1)}$, $i, j = 1, 2, \ldots, N$ are the weighting coefficients. Now the main task is to determine $a_{ij}^{(1)}$ in the equation (1.14). Once these weighting coefficients are determined, the derivative can be known in terms of the functional values at the mesh points. In the literature, different approaches have been used to compute these weighting coefficients.

Bellman et al. [17] have proposed two approaches in which following two different test functions have been applied as the trial functions to calculate the weighting coefficients related to first order derivatives:

$$\begin{aligned} f_k(x) &= x^k, \;\; k = 0, 1, \ldots, N-1, \\ f_k(x) &= \frac{L_N(x)}{(x - x_k)\;\, L_N^{(1)}(x_k)}, \;\; k = 1, 2, \ldots, N, \end{aligned} \tag{1.15}$$

where $L_N(x)$ and $L_N^{(1)}(x)$ represent the Legendre polynomial of degree N and its first order derivative, respectively.

In the first approach when number of grid points N are large, the resulting matrix is ill-conditioned and therefore, in the practical application of this approach N is usually chosen to be less than 13. However, second approach is not as flexible as the first approach because in the second approach the roots of the Legendre polynomial of degree N are chosen as the coordinates of the grid points. So due to inflexibility associated with second approach in selecting the grid points, the first approach is usually adopted in practical applications. Many attempts have been made by researchers to improve Bellman's approaches.

One of the useful approach is suggested by Quan and Chang [169, 170]. They have used the following Lagrange interpolation polynomials as the test functions to compute $a_{ij}^{(1)}$

$$\begin{aligned} f_k(x) =& \frac{M(x)}{(x-x_k)\ M^{(1)}(x_k)},\ k=1,2,\ldots,N, \\ \text{where } M(x) &= (x-x_1)(x-x_2)\ldots(x-x_k), \\ M^{(1)}(x_i) &= \prod_{k=1,k\neq i}^{N}(x_i-x_k). \end{aligned} \tag{1.16}$$

Using (1.16) as the test functions, it has been noticed that there is no restriction on the choice of grid points.

Shu [192] has proposed a more general approach, which covers the approaches proposed by Bellman et al. [17] and Quan and Chang [169, 170]. Shu concluded that the weighting coefficients calculation by all the approaches is same. As from the properties of a linear vector space, if a set of base polynomials satisfies a linear operator then other sets of base polynomials obey the same. This means that each set of base polynomials would give the same weighting coefficients and finally it was observed that $a_{ij}^{(1)}$'s satisfy the following relation

$$\sum_{j=1}^{N} a_{ij}^{(1)} = 0 \quad \text{or} \quad a_{ii}^{(1)} = -\sum_{j=1,j\neq i}^{N} a_{ij}^{(1)}. \tag{1.17}$$

1.8.2 Approximation of Second Order Derivatives

To discretized the second order derivatives, a similar approximation has been used as

$$f_x^{(2)}(x_i) = \sum_{j=1}^{N} a_{ij}^{(2)} f(x_j),\ i=1,2,\ldots,N, \tag{1.18}$$

where $f_x^{(2)}(x_i)$ represents the second order derivative of $f(x)$ at the point x_i and $a_{ij}^{(2)}$ $i,j=1,2,\ldots,N$ are the weighting coefficients related to second order derivatives.

Shu [192] has proposed a general approach to compute the weighting coefficients $a_{ij}^{(2)}$, which is also based on polynomial approximation and linear vector space analysis.

Following simple formulation has been obtained to compute $a_{ij}^{(2)}$

$$\begin{aligned} a_{ij}^{(2)} &= 2\left(a_{ij}^{(1)}a_{ii}^{(1)} - \frac{a_{ij}^{(1)}}{x_i - x_j}\right), \text{for } \ j \neq i, \\ a_{ii}^{(2)} &= -\sum_{j=1, j\neq i}^{N} a_{ij}^{(2)}, \text{for } \ j = i; \quad i, j = 1, 2, \ldots, N. \end{aligned} \tag{1.19}$$

1.8.3 Shu's Formula for Higher Order Derivatives

Higher order derivatives are discretized in the following manner

$$f_x^{(n)}(x_i) = \sum_{j=1}^{N} a_{ij}^{(n)} f(x_j), \ \ i = 1, 2, \ldots, N, \tag{1.20}$$

where $f_x^{(n)}(x_i)$ denotes the n^{th} order derivative of $f(x)$ at the point x_i and $a_{ij}^{(n)}$ are the weighting coefficients.

Shu [192] has used the similar procedure for calculation of higher order derivatives. The following explicit recurrence formulation has been obtained for weighting coefficients $a_{ij}^{(n)}$

$$\begin{aligned} a_{ij}^{(n)} &= n\left(a_{ij}^{(1)}a_{ii}^{(n-1)} - \frac{a_{ij}^{(n-1)}}{x_i - x_j}\right), \text{for } \ j \neq i, \\ a_{ii}^{(n)} &= -\sum_{j=1, j\neq i}^{N} a_{ij}^{(n)}, \text{for } \ j = i; \ \ i, j = 1, 2, \ldots, N, \end{aligned} \tag{1.21}$$

where $a_{ij}^{(n-1)}$'s represent the weighting coefficients related to $(n-1)^{th}$ order derivative of $f(x)$.

1.8.4 Extension to Multi-Dimensional Case

This section demonstrate the extension of DQM from one dimensional case to two dimensional case. Shu [192] has shown that the one dimensional DQM can be directly extended to higher dimensions, if discretization domain is regular. The regular domain can be a rectangle, square or other shapes such as circle. Consider a rectangular domain divided into a mesh of uniform length along x and y coordinate direction and a two dimensional function $f(x, y)$ define on it, as shown in Figure-1.1. Along the line $y = b$, $f(x, b)$ (denoted by filled circles) can be approximated by a polynomial of degree $(N-1)$, $P_N(x)$.

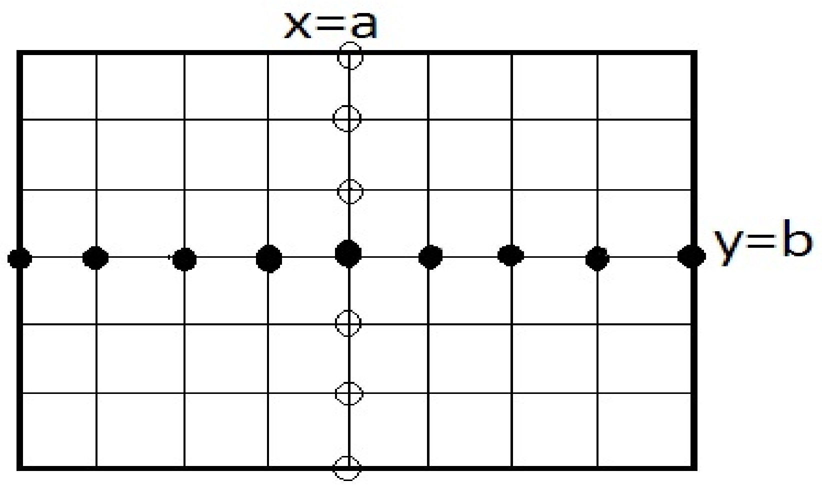


Figure 1.1: Rectangular domain.

$P_N(x)$ constitutes a N-dimensional linear vector space V_N with N base vectors (polynomials) $r_i(x),\ i = 1, 2, \ldots, N$. Similarly, along a vertical line $x = a$, $f(a, y)$ (denoted by the open circles) can be approximated by a polynomial of degree $(M - 1)$, $P_M(y)$ which constitutes a M-dimensional linear vector space V_M with M base vectors (polynomials) $s_j(y),\ j = 1, 2, \ldots, M$. Then the value of the function $f(x, y)$ at any location can be approximated by a polynomial $P_{N\times M}(x, y)$ of the form

$$f(x,y) = P_{N\times M}(x,y) = \sum_{i=1}^{N}\sum_{j=1}^{M} c_{ij}x^{i-1}y^{j-1}. \tag{1.22}$$

It is clear that $P_{N\times M}(x, y)$ constitutes a $N \times M$ dimensional linear polynomial vector space $V_{N\times M}$ with respect to the operation of vector addition and scalar multiplication, where $\phi_{ij}(x, y) = r_i(x)s_j(y)$ constitutes the base vectors in the linear vector space $V_{N\times M}$. Now it is assumed that the first order derivatives of $f(x, y)$ can be approximated as

$$f_x^{(1)}(x_i, y_j) = \sum_{k=1}^{N} a_{ik}^{(1)} f(x_k, y_j), \text{for } i = 1, 2, \ldots, N;\ j = 1, 2, \ldots, M. \tag{1.23}$$

$$f_y^{(1)}(x_i, y_j) = \sum_{k=1}^{M} b_{jk}^{(1)} f(x_i, y_k), \text{for } i = 1, 2, \ldots, N;\ j = 1, 2, \ldots, M. \tag{1.24}$$

From the properties of vector space, if all the base polynomials $\phi_{ij}(x, y)$ satisfy the linear operators defined by equations (1.23) and (1.24), then every polynomial in $V_{N\times M}$ will also satisfy. So substituting $\phi_{ij}(x,y)$ in the equations (1.23) and (1.24), we have

$$\sum_{k=1}^{N} a_{ik}^{(1)} r_j(x_k) = r_j^{(1)}(x_i), \;\; i,j = 1,2,\ldots,N, \tag{1.25}$$

$$\sum_{k=1}^{M} b_{ik}^{(1)} s_j(y_k) = s_j^{(1)}(y_i), \;\; i,j = 1,2,\ldots,M, \tag{1.26}$$

where $r_j^{(1)}(x_i)$ and $s_j^{(1)}(y_i)$ represent the first order derivatives of $r_j(x)$ and $s_j(y)$ at the point (x_i and y_i). It is clear that $a_{ik}^{(1)}$ or $b_{jk}^{(1)}$ is only related to $r_i(x)$ or $s_j(y)$ respectively. Hence formulation of one dimensional case can be directly extended to two dimensional case and the weighting coefficients for the second or higher degree derivatives are given by following recurrence relations:
Weighting coefficients related to n^{th} order derivatives w.r.to x are given by

$$\begin{aligned} a_{ij}^{(n)} &= n\left(a_{ij}^{(1)} a_{ii}^{(n-1)} - \frac{a_{ij}^{(n-1)}}{x_i - x_j} \right), \text{ for } j \neq i, \\ a_{ii}^{(n)} &= -\sum_{j=1, j\neq i}^{N} a_{ij}^{(n)}, \text{for } j = i; \;\; i,j = 1,2,\ldots,N. \end{aligned} \tag{1.27}$$

Similarly, weighting coefficients related to m^{th} order derivatives w.r.to y are given by

$$\begin{aligned} b_{ij}^{(m)} &= m\left(b_{ij}^{(1)} b_{ii}^{(m-1)} - \frac{b_{ij}^{(m-1)}}{y_i - y_j} \right), \text{ for } j \neq i, \\ b_{ii}^{(m)} &= -\sum_{j=1, j\neq i}^{M} b_{ij}^{(m)}, \text{for } j = i; \;\; i,j = 1,2,\ldots,M. \end{aligned} \tag{1.28}$$

Hence the n^{th} order derivatives of $f(x,y)$ w.r.t. x and m^{th} order derivatives of $f(x,y)$ w.r.t. y satisfy the following relations:

$$\begin{aligned} f^{(n)}(x_i,y_j) &= \sum_{k=1}^{N} a_{ik}^{(n)} f(x_k,y_j), \text{for } i = 1,2,\ldots,N; \; j = 1,2,\ldots,M. \\ f^{(m)}(x_i,y_j) &= \sum_{k=1}^{M} b_{jk}^{(m)} f(x_i,y_k), \text{for } i = 1,2,\ldots,N; \; j = 1,2,\ldots,M. \end{aligned} \tag{1.29}$$

The calculation of weighting coefficients is the key procedure in DQM, which is of paramount importance. Various other test functions have been used in the literature to compute these weighting coefficients such as Legendre polynomials and spline functions by Bellman et al. [16] and [17] respectively. Lagrange interpolated trigonometric functions have been used by Shu and Xue [194]. Striz et al. [198] have presented the harmonic differential quadrature method. Korkmaz and Dağ [109] have used Sinc functions. Shu and Wu [193] have presented an implicit approach using radial basis functions. Korkmaz et al. [112] have used the quartic B-spline basis functions based differential quadrature method to solve one dimensional Burgers' equation. Dehghan and Nikpour [51] have used the local radial basis functions based differential quadrature method for solving system of second-order boundary value problems. Korkmaz and Dağ [108] have used polynomial based differential quadrature method to solve Burgers' equation. Jiwari et al. [89] have presented a weighted average differential quadrature method to solve Burgers' equation. Korkmaz and Dağ [107] have developed a cosine expansion based differential quadrature method to solve nonlinear Schrödinger equation. Mittal and Jiwari [147] have presented a polynomial based differential quadrature method for two dimensional coupled Burgers' equation. Cubic B-spline functions based differential quadrature method has been used to solve one dimensional Burger's equation by Korkmaz and Dağ [110].

1.9 Modified Cubic B-spline Differential Quadrature Method

We have proposed an approach different from existing literature in which modified cubic B-spline basis functions (1.6) have been used to calculate the weighting coefficients of differential quadrature method.

1.9.1 Calculation of Weighting Coefficients

1.9.1.1 One Dimensional Case

For one dimensional DQM, the domain $a \leq x \leq b$ is divided into a mesh of uniform length $h = x_{i+1} - x_i = \frac{b-a}{N-1}$ by the knot x_i, where $i = 1, 2, \ldots, N-1$.

To calculate the weighting coefficients related to first order derivative, we use modified cubic B-spline basis functions $\tilde{B}_l(x)$, $l = 1, 2, \ldots, N$ in the equation (1.14) as

$$\tilde{B}_l'(x_i) = \sum_{j=1}^{N} a_{ij}^{(1)} \tilde{B}_l(x_j), \;\; i = 1, 2, \ldots, N; l = 1, 2, \ldots, N. \tag{1.30}$$

At the first knot (boundary point) $x = x_1$, we have

$$\tilde{B}_l'(x_1) = \sum_{j=1}^{N} a_{1j}^{(1)} \tilde{B}_l(x_j), \;\; l = 1, 2, \ldots, N,$$

which gives the following system of equations:

$$\begin{bmatrix} 6 & 1 & & & & & \\ 0 & 4 & 1 & & & & \\ & 1 & 4 & 1 & & & \\ & & \ddots & \ddots & \ddots & & \\ & & & 1 & 4 & 1 & \\ & & & & 1 & 4 & 0 \\ & & & & & 1 & 6 \end{bmatrix} \begin{bmatrix} a_{11}^{(1)} \\ a_{12}^{(1)} \\ a_{13}^{(1)} \\ \vdots \\ a_{1N-2}^{(1)} \\ a_{1N-1}^{(1)} \\ a_{1N}^{(1)} \end{bmatrix} = \begin{bmatrix} -\frac{6}{h} \\ \frac{6}{h} \\ 0 \\ \vdots \\ 0 \\ 0 \\ 0 \end{bmatrix} \tag{1.31}$$

The above tridiagonal system of equations has been solved using Thomas algorithm to get the weighting coefficients $a_{1j}^{(1)}$, $j = 1, 2, \ldots, N$ related to knot $x = x_1$.
Similarly, at the second knot $x = x_2$, we have

$$\tilde{B}_l'(x_2) = \sum_{j=1}^{N} a_{2j}^{(1)} \tilde{B}_l(x_j), \;\; l = 1, 2, \ldots, N, \tag{1.32}$$

which again produces a following tridiagonal system of equations:

$$\begin{bmatrix} 6 & 1 & & & & & \\ 0 & 4 & 1 & & & & \\ & 1 & 4 & 1 & & & \\ & & \ddots & \ddots & \ddots & & \\ & & & 1 & 4 & 1 & \\ & & & & 1 & 4 & 0 \\ & & & & & 1 & 6 \end{bmatrix} \begin{bmatrix} a_{21}^{(1)} \\ a_{22}^{(1)} \\ a_{23}^{(1)} \\ \vdots \\ a_{2N-2}^{(1)} \\ a_{2N-1}^{(1)} \\ a_{2N}^{(1)} \end{bmatrix} = \begin{bmatrix} -\frac{3}{h} \\ 0 \\ \frac{3}{h} \\ \vdots \\ 0 \\ 0 \\ 0 \end{bmatrix}.$$

Again using Thomas algorithm, weighting coefficients $a_{2j}^{(1)}, j = 1, 2, \ldots, N$ can be computed.

In the similar way, weighting coefficients $a_{3j}^{(1)}$, $a_{4j}^{(1)}$, $\ldots$, $a_{N-1,j}^{(1)}$ can be computed at each knot x_i, $i = 3, 4, \ldots, N-1$.
At the last knot $x = x_N$, the following tridiagonal system is obtained

$$\tilde{B}_l'(x_N) = \sum_{j=1}^{N} a_{Nj}^{(1)} \tilde{B}_l(x_j), \;\; l = 1, 2, \ldots, N, \tag{1.33}$$

which gives

$$\begin{bmatrix} 6 & 1 & & & & & \\ 0 & 4 & 1 & & & & \\ & 1 & 4 & 1 & & & \\ & & \ddots & \ddots & \ddots & & \\ & & & 1 & 4 & 1 & \\ & & & & 1 & 4 & 0 \\ & & & & & 1 & 6 \end{bmatrix} \begin{bmatrix} a_{N1}^{(1)} \\ a_{N2}^{(1)} \\ a_{N3}^{(1)} \\ \vdots \\ a_{NN-2}^{(1)} \\ a_{NN-1}^{(1)} \\ a_{NN}^{(1)} \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ \vdots \\ 0 \\ -\frac{6}{h} \\ \frac{6}{h} \end{bmatrix},$$

Hence for each point x_i, $i = 1, 2, \ldots, N$, we will get a tridiagonal system of equations. Once all the weighting coefficients $a_{ij}^{(1)}$ are determined, we use Shu's recurrence formulae (1.21) to get the weighting coefficients related to second or higher order partial derivatives.

1.9.1.2 Two Dimensional Case

According to Shu [192], one dimensional DQM can be directly extended to multi dimensional case, if discretization domain is regular. So for two dimensional DQM the domain $a \leq x \leq b$, $c \leq y \leq d$ is discretized by taking N and M grid points in x and y direction respectively, such that $h_x = x_{i+1} - x_i$ and $h_y = y_{j+1} - y_j$.

The n^{th} order partial derivative of $f(x, y)$ with respect to x and m^{th} order partial derivative of $f(x, y)$ with respect to y are approximated using the following formulae

$$f_x^{(n)}(x_i, y_j) = \sum_{k=1}^{N} a_{ik}^{(n)} f(x_k, y_j), \qquad \text{for } i = 1, 2, \ldots, N;\ j = 1, 2, \ldots, M, \tag{1.34}$$

$$f_y^{(m)}(x_i, y_j) = \sum_{k=1}^{M} b_{jk}^{(m)} f(x_i, y_k), \qquad \text{for } i = 1, 2, \ldots, N;\ j = 1, 2, \ldots, M. \tag{1.35}$$

where n and m denote the order of derivative.

To compute the weighting coefficients $a_{ik}^{(1)}$ related to first order derivatives, we use modified B-spline basis functions $\tilde{B}_l(x)$, $l = 1, 2, \ldots, N$ in equation (1.34) as y axis is fixed there, which gives

$$\tilde{B}_l'(x_i) = \sum_{k=1}^{N} a_{ik}^{(1)} \tilde{B}_l(x_k),\ i = 1, 2, \ldots, N;\ l = 1, 2, \ldots, N. \tag{1.36}$$

Now using the similar procedure as defined in the Section (1.9.1.1), weighting coefficients $a_{ik}^{(1)}$ for $i, k = 1, 2, \ldots, N$ can be computed.

Similarly, the weighting coefficients $b_{jk}^{(1)}$, for $j, k = 1, 2, \ldots, M$ are computed using modified B-spline basis functions $\tilde{B}_l(y),\ l = 1, 2, \ldots, M$ in equation (1.35) as

$$\tilde{B}_l'(y_j) = \sum_{k=1}^{M} b_{jk}^{(1)} \tilde{B}_l(y_k),\ j = 1, 2, \ldots, M;\ l = 1, 2, \ldots, M. \tag{1.37}$$

Once the weighting coefficients related to first order derivatives are determined, we use Shu's recurrence formulae (1.27) and (1.28) to determine the weighting coefficients $a_{ik}^{(n)}$, for $i, k = 1, 2, \ldots, N$ and $b_{jk}^{(m)}$, for $j, k = 1, 2, \ldots, M$ related to second or higher order partial derivatives.

1.10 Error Norms and Order of Convergence

To verify the performance of the proposed methods, the numerical approximation errors are obtained using the following formulae:

$$L_2 = \sqrt{h \sum_{j=0}^{N} |u_j - U_j|^2} \tag{1.38}$$

$$L_\infty = \max_j |u_j - U_j| \tag{1.39}$$

$$\text{RMS error} = \frac{1}{N+1} \sqrt{\sum_{j=0}^{N} |u_j - U_j|^2} \tag{1.40}$$

$$\text{Order of Conv.} = \frac{\log\left(\frac{\|u - U_{h_i}\|_{L_j}}{\|u - U_{h_{i+1}}\|_{L_j}}\right)}{\log\left(\frac{h_i}{h_{i+1}}\right)},\ i = 1, 2;\ j = 2, \infty \text{ and } h_{i+1} = \frac{h_i}{2} \tag{1.41}$$

where u and U denote the exact and approximate solutions, respectively.

1.11 Thomas Algorithm

In the process of numerical solutions of partial differential equations with B-spline collocation and differential quadrature methods, we get system of linear algebraic/ord
inary differential equations in which the elements distributed on the diagonal and above and below that diagonal.

Instead of storing the obtained system in matrix(order N) form with a very large number of zero elements (especially in case of large N), we store the system in vector format as: the diagonal elements in matrix A; say vector b is of order N, the elements below the diagonal; say vector a, is of order N and the elements above the diagonal; say vector c is also of order N with $a_1 = c_N = 0$, then we apply the elimination process of one row below the diagonal.

The benefits for solving such systems of equations using Thomas algorithm instead of using the general elimination (Gauss elimination on a full matrix) is to reduce the number of arithmetic operations from $O(N^3)$ to $O(N)$. The Thomas algorithm is very efficient in computer time and storage for solving tridiagonal system of equations.

1.12 SSP-RK Methods

Strong stability preserving time discretization methods were developed to address the need for nonlinear stability properties in time discretization as well as spatial discretization of hyperbolic PDEs. Hyperbolic PDEs typically have discontinuous solution and a stronger measure than linear stability is thus required.

When the spatial derivatives of time dependent partial differential equations are discretized, we obtain the system of ODEs in time variable as

$$u_t = F(u), \tag{1.42}$$

where u is a vector of approximation. This system is then discretized by an ODE solver. For problems with smooth solutions, usually a linear stability analysis is sufficient. The spatial discretization is carefully designed so that when this ODE is fully discretized using the forward Euler method

$$u^{n+1} = u^n + \Delta t L(u^n), \tag{1.43}$$

certain convex functional properties (such as the total variation) of the numerical solution u^n does not increase with time, i.e. the following so called Total Variation Diminishing (TVD) property holds

$$TV(u^{n+1}) \leq TV(u^n), \quad TV(u^n) = \sum_j |u^n_{j+1} - u^n_j|, \tag{1.44}$$

for all small enough step sizes $\Delta t \leq \Delta t_{FE}$.

Here Δt_{FE} is the largest allowable step size which guarantee that the above stability condition will hold for forward Euler with given PDE and spatial discretization.

Typically, we need methods of higher order and we wish to guarantee that the higher-order time discretization will preserve this strong stability property. This guarantee is obtained by observing that if a time discretization can be decomposed into convex combinations of forward Euler steps then any functional property (referred to here as strong stability property) satisfied by forward Euler will be preserved by the higher-order time discretization, perhaps under a different time-step restrictions.

Given a semi-discretization of the form (1.42) and convex functional $\|.\|$, we assume that there exists a value Δt_{FE} such that for all u,

$$\|u^n + \Delta t F(u^n)\| \leq \|u^n\|, \text{ for } 0 \leq \Delta t \leq \Delta t_{FE}. \tag{1.45}$$

An explicit s-stage Runge-Kutta method is written in the form

$$\begin{aligned} u^{(0)} &= u^n, \\ u^{(i)} &= \sum_{j=0}^{i-1} (\alpha_{ij} u^{(j)} + \Delta t \beta_{ij} F(u^{(j)})), \ i = 1, 2, \ldots, s, \\ u^{n+1} &= u^{(s)}, \end{aligned} \tag{1.46}$$

where all $\alpha_{ij} \geq 0$ and $\alpha_{ij} = 0$ only if $\beta_{ij} = 0$. If all the coefficients β_{ij} are non-negative then by consistency $\sum_{j=0}^{i-1} \alpha_{ij} = 1$ and intermediate stages of Runge-Kutta method can be arranged into convex combinations of forward Euler's steps with a modified step size $\frac{\beta_{ij}}{\alpha_{ij}} \Delta t$ as

$$\|u^{(i)}\| = \| \sum_{j=0}^{i-1} \left(\alpha_{ij} u^{(j)} + \Delta t \beta_{ij} F(u^{(j)}) \right) \| \leq \sum_{j=0}^{i-1} \alpha_{i,j} \|(u^{(j)} + \Delta t \frac{\beta_{ij}}{\alpha_{ij}} F(u^{(j)}))\| \tag{1.47}$$

where each $\|u^{(j)} + \Delta t \frac{\beta_{ij}}{\alpha_{ij}} F(u^{(j)})\| \leq \|u^{(j)}\|$ as long as $\frac{\beta_{ij}}{\alpha_{ij}} \Delta t \leq \Delta t_{FE}$.

Thus if forward Euler method applied to (1.42) is strongly stable under the time step restriction $\Delta t \leq \Delta t_{FE}$, and if α_{ij}, $\beta_{ij} \geq 0$ then the solution obtained by the RK method (1.46) is SSP i.e. $\|u^{n+1}\| \leq \|u^n\|$, under the time step restriction $\Delta t \leq \min_{i,j} \frac{\alpha_{i,j}}{\beta_{i,j}} \Delta t_{FE}$, where $c = \min_{i,j} \frac{\alpha_{i,j}}{\beta_{i,j}}$ is called the SSP coefficient of the method.

Properties of SSP-RK Methods

SSP-RK methods are designed to achieve certain stability properties of the original PDE such as total variation stability or maximum norm stability.

- SSP-RK high order time discretization maintains stability under a certain norm with a suitable restriction on the time step.

- Numerical experiments show that oscillations and nonlinear instability could occur when a non SSP-RK method is applied for time discretization. Thus, it is safer to use an SSP time discretization, especially for hyperbolic problems.

- These methods do not increase computational cost and have the added assurance of provable stability.

- In terms of computational cost, most SSP-RK methods are of the same form and have the same cost as traditional ODE solvers.

1.13 Solution of a System of First Order ODEs

Consider the system of first order ODEs

$$\frac{du}{dt} = L(u), \text{ with given initial condition } u(0) = u^0. \tag{1.48}$$

To solve above system Spiteri and Ruuth [197] have given the following explicit formulae:

SSP-RK43 Scheme

$$\begin{aligned} u^{(1)} &= u^n + \frac{1}{2}\,\Delta t\, L(u^n), \\ u^{(2)} &= u^{(1)} + \frac{1}{2}\,\Delta t\, L(u^{(1)}), \\ u^{(3)} &= \frac{2}{3}u^n + \frac{1}{3}u^{(2)} + \frac{1}{6}\,\Delta t\, L(u^{(2)}), \\ u^{n+1} &= u^{(3)} + \frac{1}{2}\,\Delta t\, L(u^{(3)}). \end{aligned} \tag{1.49}$$

SSP-RK54 Scheme

$$\begin{aligned}
u^{(1)} =& u^n + 0.39175222700392\Delta t L(u^n),\\
u^{(2)} =& 0.44437049406734u^n + 0.55562950593266u^{(1)}\\
&+0.36841059262959\,\Delta t\, L(u^{(1)})\\
u^{(3)} =& 0.62010185138540u^n + 0.379898148511597u^{(2)},\\
&+0.251891774271694\,\Delta t\, L(u^{(2)}),\\
u^{(4)} =& 0.178079954393132u^n + 0.821920045606868u^{(3)}\\
&+0.544974750228521\,\Delta t\, L(u^{(3)}),\\
u^{n+1} =& 0.00683325884039u^n + 0.517231671970585u^{(2)}\\
&+0.12759831133288u^{(3)} + 0.34833675773694u^{(4)}\\
&+0.08460416338212\,\Delta t\, L(u^{(3)}) + 0.22600748319395\,\Delta t\, L(u^{(4)}).
\end{aligned} \tag{1.50}$$

1.14 Contents of the Thesis

In this thesis, an attempt has been made to find the approximate solutions of some time dependent linear/nonlinear partial differential equations, using collocation and differential quadrature methods with B-spline basis functions. The chapter wise summary is as follows:

Chapter 2 deals with numerical solutions of nonlinear Klein-Gordon equation and Klein-Gordon-Schrödinger equations with Dirichlet and Neumann boundary conditions. One dimensional Klein-Gordon equation is given by

$$u_{tt} + \alpha u_{xx} + g(u) = f(x,t), \quad x \in (a,b),\ \ t > 0, \tag{1.51}$$

with initial conditions

$$u(x,0) = u_0(x), \quad u_t(x,0) = u_1(x),$$

and with following Dirichlet or Neumann boundary conditions

$$\begin{aligned}
&u(a,t) = f_1(t), \quad u(b,t) = f_2(t),\\
&u_x(a,t) = h_1(t), \quad u_x(b,t) = h_2(t),\ \ t \geq 0
\end{aligned}$$

where $\alpha < 0$ is a known real constant, $f(x,t)$ is known analytic function and $g(u)$ is a nonlinear force which may takes many forms such as $\sin u$, $\sinh u$, $\sin u + \sin 2u$, $\sinh u + \sinh 2u$.

Yukawa-coupled Klein-Gordon-Schrödinger (KGS) equation is given by

$$\begin{aligned} i\psi_t &= -\frac{1}{2}\psi_{xx} - \phi\psi, \\ \phi_{tt} &= \phi_{xx} - \phi + |\psi|^2, \ x \in \mathbb{R}, \ t > 0, \end{aligned} \tag{1.52}$$

with the following initial and boundary conditions

$$\begin{aligned} &\psi(x,0) = \psi_0(x), \quad \phi(x,0) = \phi_0(x), \quad \phi_t(x,0) = \phi_1(x), \\ &\lim_{|x|\to\infty} \psi(x,t) = 0, \quad \lim_{|x|\to\infty} \phi(x,t) = 0. \end{aligned}$$

Here ψ(x,t) is a complex function represents a scalar neutron field and ϕ(x,t) is a real function represents a scalar neutral meson field.

Numerical solutions of both the equations are obtained using cubic B-spline collocation method. Modified cubic B-spline basis functions are used to handle the Dirichlet boundary conditions. The equations are decomposed into a system of partial differential equations, which are further converted to an amenable system of ODEs. The obtained system of ODEs is solved by SSP-RK54 scheme. Numerical results are presented for six examples to show the accuracy and utility of proposed approach. The approximate solutions of both the equations are computed without using any transformation and linearization. The computed results are of better accuracy than earlier results available in the literature. The implementation of this method is very easy and cost-effective.

Chapter 3 addresses the modified cubic B-splines collocation method to find numerical solution of nonlinear sine-Gordon equation. We consider

$$u_{tt} = u_{xx} - \sin(u), \qquad x \in (a,b), \ t > 0, \tag{1.53}$$

with initial conditions

$$u(x,0) = \phi_1(x), \quad u_t(x,0) = \phi_2(x),$$

and with following Dirichlet boundary conditions

$$u(a,t) = \psi_1(t), \quad u(b,t) = \psi_2(t), \quad t \geq 0.$$

The nonlinear sine-Gordon equation is converted into a system of partial differential equations. We apply modified cubic B-splines for spatial variable and its derivatives, which produces a system of first order ordinary differential equations. The obtained system of ODEs is solved using SSP-RK54 scheme. The particular features of these schemes are that it inherently perpetuates certain stability properties and maximum norm stability. It also controls spurious oscillations and non-linear instability during simulation. The accuracy of scheme is tested on four numerical examples. The results obtained by the presented method are of better precision and competent accuracy than the results available in the earlier works. The order of convergence of the scheme is also computed and found to be approaching two.

Chapter 4 is concerned with the numerical solution of one dimensional hyperbolic telegraph equation with Dirichlet and Neumann boundary conditions. Consider

$$u_{tt}(x,t) + 2\alpha u_t(x,t) + \beta^2 u(x,t) = u_{xx}(x,t) + f(x,t), \qquad x \in (a,b),\ \ t > 0, \tag{1.54}$$

with initial conditions

$$u(x,0) = f_1(x),\ \ u_t(x,0) = f_2(x)$$

and the following Dirichlet or Neumann boundary conditions

$$\begin{aligned} &u(a,t) = g_1(t),\ \ u(b,t) = g_2(t),\\ &u_x(a,t) = w_1(t),\ \ u_x(b,t) = w_2(t),\ \ t \geq 0. \end{aligned}$$

Here α and β are known real constants. For $\alpha > 0$, $\beta = 0$ it represents a damped wave equation and for $\alpha > \beta > 0$ it is called as telegraph equation.

The cubic B-spline collocation method is developed to get the numerical solution of telegraph equation. Modified cubic B-spline basis functions are utilized to handle the Dirichlet boundary conditions. The use of B-splines basis functions for spatial variable and its derivatives, results in an amenable system of differential equations. The resulting system of equations is solved by the SSP-RK54 scheme. Stability of scheme is discussed using matrix stability analysis and found unconditionally stable. The efficacy of proposed approach is confirmed with five numerical experiments, which shows that the obtained solutions are acceptable and in good agreement with earlier studies.

In chapter 5 we propose an efficient differential quadrature method, to find the numerical solution of two dimensional hyperbolic telegraph equation with Dirichlet and Neumann boundary conditions. Consider

$$\begin{aligned} u_{tt}(x,y,t) + 2\alpha u_t(x,y,t) + \beta^2 u(x,y,t) = u_{xx}(x,y,t) + u_{yy}(x,y,t) + f(x,y,t),& \\ (x,y,t) \in D \times (0,T].& \end{aligned} \tag{1.55}$$

The initial conditions are given by

$$\begin{aligned} u(x,y,0) &= u_0(x,y), \\ u_t(x,y,0) &= v_0(x,y), \;\; (x,y) \in D. \end{aligned}$$

The Dirichlet and Neumann boundary conditions are given by

$$\begin{aligned} &u(a,y,t) = f_1(y,t), \;\; u(b,y,t) = f_2(y,t), \\ &u(x,c,t) = f_3(x,t), \;\; u(x,d,t) = f_4(x,t), \quad (x,y,t) \in \partial D \times (0,T] \end{aligned}$$

$$\begin{aligned} &u_x(a,y,t) = g_1(y,t), \;\; u_x(b,y,t) = g_2(y,t), \\ &u_y(x,c,t) = g_3(x,t), \;\; u_y(x,d,t) = g_4(x,t), \quad (x,y,t) \in \partial D \times (0,T] \end{aligned}$$

where D denotes the rectangular domain $[a,b] \times [c,d]$, ∂D is its boundary, α and β are the real constants. For $\alpha > 0$, $\beta = 0$, it represents a damped wave equation and for $\alpha > 0$, $\beta > 0$, it is called telegraph equation.

The equation is converted into a system of partial differential equations and further reduced into a system of ordinary differential equations using DQM. The resulting system of ODEs in time is solved by SSP-RK43 scheme. The stability of the scheme is analyzed by matrix stability analysis and found unconditionally stable. The efficacy of proposed approach is confirmed with seven numerical experiments, where comparisons are made with some earlier work. It has been observed that obtained results are acceptable and are in good agreement with earlier studies. However, we obtained these results in much less CPU time. The method is very simple, efficient and produces very accurate numerical results in considerably smaller number of nodes and hence saves computational effort.

Chapter 6 discusses the application of modified cubic B-spline differential quadrature method to find the numerical solutions of some nonlinear wave equations in one and two dimensions with Dirichlet boundary conditions. We consider

$$u_{tt} = u_{xx} + f(x, t, u, u_x, u_t), \;\; x \in (a, b), \; t > 0, \tag{1.56}$$

$$u_{tt} = u_{xx} + u_{yy} + f(x, y, t, u, u_x, u_y, u_t), \;\; (x, y, t) \in [a, b] \times [c, d] \times (0, T] \tag{1.57}$$

with suitable initial and boundary conditions. Here f is some nonlinear expression in terms of u, u_x, u_y, u_t,.

To obtain the numerical solutions, above equations are decomposed into a system of partial differential equations. Modified cubic B-spline basis functions based differential quadrature method is used for space discretization to obtain a system of nonlinear first order ordinary differential equations. The resulting system of equations is solved using SSP-RK43 scheme. In numerical testing, the method is implemented on Vander pole type nonlinear wave equation, Dissipative nonlinear wave equation and Telegraph equation. The numerical solutions of nonlinear wave equations are computed without linearizing the nonlinear term. The accuracy of the approach is confirmed with seven numerical experiments and obtained results are found better in comparison with previous studies. The order of convergence of method is also computed and found to be two.

Chapter 7 presents the numerical solution of nonlinear two dimensional coupled Burgers' equation, given by

$$\begin{aligned} \frac{\partial u}{\partial t} + u\frac{\partial u}{\partial x} + v\frac{\partial u}{\partial y} &= \frac{1}{R}\left(\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2}\right), \\ \frac{\partial v}{\partial t} + u\frac{\partial v}{\partial x} + v\frac{\partial v}{\partial y} &= \frac{1}{R}\left(\frac{\partial^2 v}{\partial x^2} + \frac{\partial^2 v}{\partial y^2}\right) \quad (x, y, t) \in D \times (0, T], \end{aligned} \tag{1.58}$$

with initial conditions

$$\begin{aligned} u(x, y, 0) &= \phi(x, y), \\ v(x, y, 0) &= \psi(x, y), \;\; (x, y) \in D. \end{aligned}$$

and boundary conditions

$$\begin{aligned} u(x, y, t) &= \phi_1(x, y, t), \\ v(x, y, t) &= \psi_1(x, y, t), \quad (x, y, t) \in \partial D \times (0, T], \end{aligned}$$

where D is the rectangular domain $[a, b] \times [c, d]$ and R denotes the Reynolds number.

The equations are reduced into a system of ordinary differential equations by modified cubic B-spline differential quadrature method. The obtained system of nonlinear ordinary differential equations is then solved by SSP-RK43 scheme. The accuracy of the approach is tested on five test problems and computed results are compared with some earlier works. The results of computations indicate that modified cubic B-spline differential quadrature method combined with SSP-RK scheme gives more accurate results than earlier works with less computational cost. Numerical results are computed for higher Reynolds number up to $R = 1500$. The strong points of the method are in ease to apply and less computational effort.

Finally, in **chapter 8** conclusions are drawn based on the present study and future research work is suggested, in this direction.

Table 1.1: Values of quadratic B-spline and its derivative at different knots.

x	x_{j-1}	x_j	x_{j+1}	x_{j+2}
$B_j(x)$	0	0	1	0
$B_j'(x)$	0	$\frac{2}{h}$	$\frac{-2}{h}$	0

Table 1.2: Values of cubic B-spline and its derivatives at different knots.

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

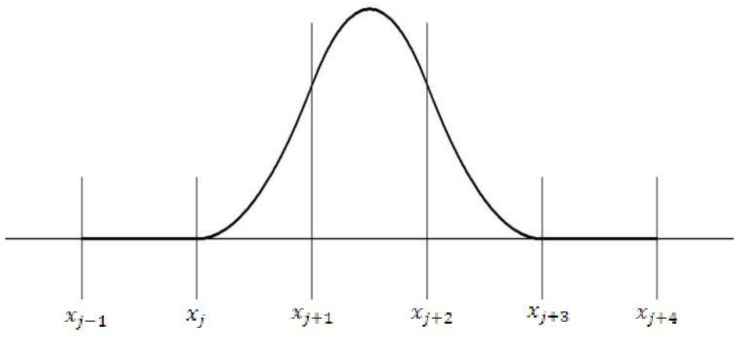


Figure 1.2: Graph of quadratic B-spline.

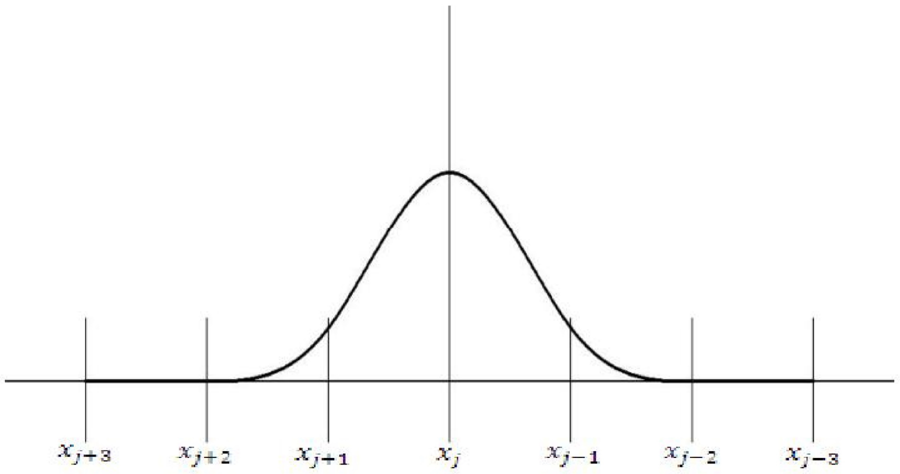


Figure 1.3: Graph of cubic B-spline.

Chapter 2

Numerical Solution of a System of Nonlinear Klein-Gordon Equations

2.1 Introduction

The study in this chapter is concerned with the numerical solutions of nonlinear Klein-Gordon and coupled Klein-Gordon-Schrödinger equations.

2.1.1 Klein-Gordon Equation

The nonlinear Klein-Gordon (KG) equation describes a variety of physical phenomena such as dislocations, ferroelectric and ferromagnetic domain walls, DNA dynamics, and Josephson junctions. It plays a pivotal role in mathematical physics and in condensed matter physics. It has attracted much attention in investigating the interaction of solitons in a collisionless plasma, initial states and in examining the nonlinear wave equations [33, 60]. With polynomial nonlinearity, it models many problems in classical and quantum mechanics and in nonlinear optics.
Consider nonlinear Klein-Gordon equation

$$u_{tt} + \alpha u_{xx} + g(u) = f(x,t), \quad x \in (a,b), \;\; t > 0, \tag{2.1}$$

subjected to the initial conditions

$$u(x,0) = u_0(x), \quad u_t(x,0) = u_1(x), \;\; x \in [a,b]. \tag{2.2}$$

The Dirichlet boundary conditions are given by

$$u(a,t) = f_1(t), \quad u(b,t) = f_2(t), \;\; t \geq 0, \tag{2.3}$$

or Neumann boundary conditions are given by

$$u_x(a,t) = h_1(t), \quad u_x(b,t) = h_2(t), \;\; t \geq 0, \tag{2.4}$$

where $f_1(t)$, $f_2(t)$, $h_1(t)$, $h_2(t)$ and their derivatives up to second order are continuous in the domain of interest.

The parameter $\alpha < 0$ is a known real constant, $f(x,t)$ is known analytic function and $g(u)$ is a nonlinear force, which may takes many forms such as: $\sin u$, $\sinh u$, $\sin u + \sin 2u$, $\sinh u + \sinh 2u$, that characterize the sine-Gordon, sinh-Gordon, double sine-Gordon and double sinh-Gordon equations [56, 123] respectively. Function $g(u)$ also appears in the form of a polynomial like $\beta u + \gamma u^n$ [42], where β, $\gamma \in \mathbb{R}$.

The nonlinear Klein-Gordon equations are the Hamiltonian partial differential equations and the solitons that appears in form of $u_{tt} + \alpha u_{xx} + g(u) = 0$, for a wide class of function $g(u)$, Hamiltonian energy and momentum are conserved [42, 88].

$$\text{Energy}(E(t)) = \int_a^b \left(\frac{1}{2}u_t^2 + \frac{1}{2}u_x^2 + G(u) \right) \, dx, \quad \text{where } G^{'}(u) = g(u). \tag{2.5}$$

$$\text{Momentum}(P(t)) = \int_a^b \left(\frac{1}{2}u_x u_t \right) \, dx. \tag{2.6}$$

2.1.2 Coupled Klein-Gordon-Schrödinger Equation

We also consider following Yukawa-coupled Klein-Gordon-Schrödinger (KGS) equation

$$\begin{aligned} i\psi_t &= -\frac{1}{2}\psi_{xx} - \phi\psi, \\ \phi_{tt} &= \phi_{xx} - \phi + |\psi|^2, \;\; x \in \mathbb{R}, \;\; t > 0, \end{aligned} \tag{2.7}$$

where ψ(x,t) is a complex function represents a scalar neutron field and ϕ(x,t) is a real function represents a scalar neutral meson field. This model describes the interaction between conservative complex neutron field and neutral meson Yukawa in quantum field theory and plays an important role in quantum physics.

In large domain $|x| \geq 0$, solutions of the above equations decay rapidly to zero. Therefore, the system (2.7) has been solved in a compact space domain. The initial and homogenous boundary conditions are given by

$$\psi(x,0) = \psi_0(x), \quad \phi(x,0) = \phi_0(x), \quad \phi_t(x,0) = \phi_1(x), \tag{2.8}$$

$$\lim_{|x|\to\infty} \psi(x,t) = 0, \quad \lim_{|x|\to\infty} \phi(x,t) = 0, \tag{2.9}$$

where $\psi_0(\mathrm{x})$ is a complex-valued function and $\phi_0(\mathrm{x})$, $\phi_1(\mathrm{x})$ are known real valued functions.

Much work has been reported in the literature on the progress of numerical schemes for the KG and coupled KGS equations. Jiménez and Vázquez [88] have compared the properties of four explicit finite difference schemes for KG equation and found undesirable characteristics in some of these schemes. Lee [129] has developed a collocation method for the numerical solution of KG equation in two dimension and showed the stability, convergence of the method using spectral method technique. Decomposition methods [41, 67, 102] have been successfully applied to find the approximate analytical solutions of nonlinear KG equation in terms of a convergent power series. Wang and Cheng [205] have proposed the numerical solution of damped nonlinear Klein-Gordon equation using variational method and finite element approximations. The existence and uniqueness of the solution of damped nonlinear Klein-Gordon equation have been discussed in [103]. Ravi Kanth and Aruna [182] have implemented differential transform method to find exact series solutions of linear and nonlinear KG equations. Dehghan and Shokri [56] have proposed a collocation method to solve one-dimensional nonlinear KG equation with quadratic and cubic nonlinearity. Lakestani and Dehghan [123] have presented a collocation method using cubic B-spline scaling functions to solve equation (2.1). Rashidinia et al. [175] have developed a B-spline collocation method to solve equation (2.1) with both Dirichlet and Neumann boundary conditions. Rashidinia and Mohammadi [177] have constructed a three time-level spline-difference scheme. Bratsos [24] has presented a predictor-corrector scheme, Li et al. [130] have proposed a lattice Boltzmann scheme and Rashidinia et al. [174] have developed a B-spline collocation method to solve linear Klein-Gordon equations. However, very few researchers have paid attention to find numerical solutions of equation (2.1) with Neumann boundary conditions (2.4). Dehghan et al. [50] have developed an unconditionally stable fourth order compact method, which has fourth order accuracy in both space and time. In [42], the approximate solution of equation (2.1) has been obtained by employing the dual reciprocity boundary integral equation method. Recently, Doha et al. [63] have developed a Jacobi-Gauss-Lobatto collocation (J-GLC) method for finding the numerical solution of nonlinear coupled Klein-Gordon equation.

Numerical solutions of nonlinear KGS system have been also studied by various approaches in literature. Hong et al. [78] have applied five difference schemes to solve the KGS equations (2.7). The symplectic and multisymplectic methods have been discussed in [104, 105]. Zhang [210] has discussed a conservative difference scheme for a class of KGS equations. Wang and Zhang [206] have developed a class of discrete-time orthogonal spline collocation scheme by using piecewise cubic Hermite interpolations in space, combined with finite difference methods in time. Wang and Zhou [204] have studied the periodic wave solutions and Kong et al. [106] have proposed an explicit symplectic partitioned Runge-Kutta Fourier pseudo-spectral scheme to solve nonlinear system of KGS equations (2.7).

In this chapter, we have proposed a numerical scheme based on collocation of cubic B-spline basis functions for solving Klein-Gordon and coupled Klein-Gordon-Schrödinger equations. For Neumann boundary conditions, we have used cubic B-splines and for Dirichlet boundary conditions, we have used modified cubic B-splines. The use of B-splines for spatial variable and its derivatives produces a system of first order nonlinear ordinary differential equations. Obtained system of equations is solved using SSP-RK54 [73] scheme. The presented method approximates both the equations (2.1) and (2.7), without using any transformation and quasi-linearization approach. Finally, the efficiency of the method is tested on different type of six examples.

This chapter is structured as follows. In Section-2.2, cubic B-spline collocation method is exemplified. In Section-2.3, execution procedure of this method is explained for equations (2.1)-(2.4). Numerical scheme for Yukawa-coupled Klein-Gordon-Schrödinger equations (2.7)-(2.9) is presented in Section-2.4. In Section-2.5, six numerical test examples are considered to show the accuracy of proposed method, computationally. The conclusions are given in Section-2.6, that briefly summarizes the numerical outcomes.

2.2 Description of Method

The domain $[a, b]$ is divided into a number of subinterval by taking uniform interval $h = x_{j+1} - x_j$, such that $a = x_0 < x_1 < \cdots < x_{N-1} < x_N = b$.

To solve equations (2.1) and (2.7), using collocation method with cubic B-splines, we assume our approximate solution $U(x,t)$ in the form

$$U(x,t) = \sum_{j=-1}^{N+1} c_j(t) B_j(x), \tag{2.10}$$

where $c_j(t)$ are the unknown time dependent quantities.

The cubic B-splines $B_j(x)$ at the knots are given as equation (1.5). In Table-1.2, values of $B_j(x)$ and its two derivatives are reported at different knots. Since $B_j(x) = 0$, outside the interval $[x_{j-2}, x_{j+2})$, so there is no need to tabulate $B_j(x)$ for other values of x.

Using approximate solution (2.10) and Table-1.2, the approximate values of $U(x,t)$ and its two derivatives at the knots are determined in terms of the time parameters c_j, which have been reported in equation (1.13).

2.3 Numerical Solution of Klein-Gordon Equation

KG equation (2.1) is converted into a system of partial differential equations using the transformation $u_t(x,t) = v(x,t)$, as

$$\begin{aligned} u_t &= v, \\ v_t &= -\alpha u_{xx} - g(u) + f(x,t), \end{aligned} \tag{2.11}$$

where the boundary conditions for v are obtained from corresponding values of u_t at the boundary points.

2.3.1 Treatment of Dirichlet Boundary Conditions

We have used the modified cubic B-spline basis functions (1.6) to solve equation (2.1) with Dirichlet boundary conditions. In collocation method with cubic B-splines, there is a requirement to change the basis functions into a new set of functions such that the number of basis functions matches with the selected points in the given domain. The Dirichlet boundary conditions can be easily tackled using these basis functions and finally, we get a diagonally dominant system of equations.

We assume our approximate solution as the linear combination of modified cubic B-spline basis functions (1.6) as

$$U(x,t) = \sum_{j=0}^{N} c_j(t) \tilde{B}_j(x). \tag{2.12}$$

Using equation (2.12), approximate value of $U_t(x,t)$ can be written as

$$U_t(x,t) = \sum_{j=0}^{N} \dot{c}_j(t)\tilde{B}_j(x), \tag{2.13}$$

where $\dot{c}_j(t)$ is the derivative of $c_j(t)$ with respect to time t.

Using modified basis functions and Table-1.2 in equation (2.13), the values of $U_t(x,t)$ at different knots will be given by

$$\begin{aligned} U_t(x_0,t) &= 6\dot{c}_0, \ \text{for}\, j = 0, \\ U_t(x_j,t) &= \dot{c}_{j-1} + 4\dot{c}_j + \dot{c}_{j+1}, \ \text{for}\, j = 1,2,\dots,N-1, \\ U_t(x_N,t) &= 6\dot{c}_N, \ \text{for}\, j = N. \end{aligned} \tag{2.14}$$

Using (2.13) in equation (2.11) and imposing the boundary conditions (2.3) at the boundary points, we have

$$\begin{aligned} U_t(x_0,t) &= \dot{f}_1(t), \ \text{for}\, i = 0, \\ U_t(x_i,t) &= v_i, \ \text{for}\, i = 1,2,\dots,N-1, \\ U_t(x_N,t) &= \dot{f}_2(t), \ \text{for}\, i = N. \end{aligned} \tag{2.15}$$

$$\begin{aligned} v_t(x_0,t) &= \ddot{f}_1(t), \ \text{for}\, i = 0, \\ v_t(x_i,t) &= -\alpha\sum_{j=0}^{N} c_j\tilde{B}_j^{''}(x_i) - g\left(\sum_{j=0}^{N} c_j\tilde{B}_j(x_i)\right) + f(x_i,t), \ \text{for}\, i = 1,\dots,N-1, \\ v_t(x_N,t) &= \ddot{f}_2(t), \ \text{for}\, i = N. \end{aligned} \tag{2.16}$$

Now using (2.14) in equation (2.15) and using modified basis functions (1.6), Table-1.2 in equation (2.16), we have

$$\begin{aligned} 6\dot{c}_0 &= \dot{f}_1(t), \ \text{for}\, j = 0, \\ \dot{c}_{j-1} + 4\dot{c}_j + \dot{c}_{j+1} &= v_j, \ \text{for}\, j = 1,2,\dots,N-1, \\ 6\dot{c}_N &= \dot{f}_2(t), \ \text{for}\, j = N. \end{aligned} \tag{2.17}$$

$$\begin{aligned} \dot{v}_0 &= \ddot{f}_1(t), \ \text{for}\, j = 0, \\ \dot{v}_j &= -\frac{6\alpha}{h^2}(c_{j-1} - 2c_j + c_{j+1}) - g(c_{j-1} + 4c_j + c_{j+1}) + f(x_j,t), \ j = 1,..,N-1 \\ \dot{v}_N &= \ddot{f}_2(t), \ \text{for}\, j = N. \end{aligned} \tag{2.18}$$

Systems (2.17) and (2.18) represent the following systems of first order ODEs

$$\begin{aligned} A\dot{\mathbf{c}} &= G, \\ \dot{V} &= D, \end{aligned} \tag{2.19}$$

where

$$A = \begin{bmatrix} 6 & 0 & \dots & \dots & 0 \\ 1 & 4 & 1 & \dots & 0 \\ & & \dots & \dots & \dots \\ & & \dots & \dots & \dots \\ & & 1 & 4 & 1 \\ & & & 0 & 6 \end{bmatrix}_{(N+1)\times(N+1)},$$

$\dot{\mathbf{c}}$ and $\dot{V}$ are column vectors each of order $(N+1)$. G and D are column vectors of order $(N+1)$ which depend on the boundary conditions and represent the rhs of equations (2.17) and (2.18), respectively.

2.3.2 Treatment of Neumann Boundary Conditions

In this case, we use collocation of cubic B-spline basis functions (1.5) to solve coupled system (2.11). We assume our approximate solution as

$$U(x,t) = \sum_{j=-1}^{N+1} c_j(t) B_j(x), \tag{2.20}$$

which gives

$$U_t(x,t) = \sum_{j=-1}^{N+1} \dot{c}_j(t) B_j(x). \tag{2.21}$$

Using (2.20), Neumann boundary conditions (2.4) are approximated as

$$\begin{aligned} U_x(x_0,t) &= \sum_{j=-1}^{1} c_j B_j^{'}(x_0) = h_1(t), \\ U_x(x_N,t) &= \sum_{j=N-1}^{N+1} c_j B_j^{'}(x_N) = h_2(t). \end{aligned} \tag{2.22}$$

Using Table-1.2 in above, we have

$$\begin{aligned} \frac{3}{h}(c_1 - c_{-1}) &= h_1(t), \\ \frac{3}{h}(c_{N+1} - c_{N-1}) &= h_2(t). \end{aligned} \tag{2.23}$$

From (2.23), we have

$$\begin{aligned} \dot{c}_1 - \dot{c}_{-1} &= \left(\frac{h}{3}\right) \dot{h}_1(t), \\ \dot{c}_{N+1} - \dot{c}_{N-1} &= \left(\frac{h}{3}\right) \dot{h}_2(t). \end{aligned} \tag{2.24}$$

Now using (2.20) and (2.21) in coupled system (2.11), we have

$$\sum_{j=-1}^{N+1} \dot{c}_j B_j(x_i) = v_i, \text{ for } i = 0, 1, \ldots, N. \tag{2.25}$$

$$\dot{v}_i = -\alpha \sum_{j=-1}^{N} c_j B_j''(x_i) - g\left(\sum_{j=-1}^{N} c_j B_j(x_i)\right) + f(x_i, t), \text{ for } i = 0, 1, \ldots, N. \tag{2.26}$$

Using approximate solution (1.13) and Table-1.2 in equations (2.25) and (2.26), we get following systems of ODEs

$$\dot{c}_{j-1} + 4\dot{c}_j + \dot{c}_{j+1} = v_j, \;\; j = 0, 1, \ldots, N. \tag{2.27}$$

$$\dot{v}_j = \frac{-6\alpha}{h^2}(c_{j-1} - 2c_j + c_{j+1}) - g(c_{j-1} + 4c_j + c_{j+1}) + f(x_j, t), \;\; j = 0, 1, \ldots, N. \tag{2.28}$$

Now eliminating $c_{-1}, c_{N+1}, \dot{c}_{-1}, \dot{c}_{N+1}$ from (2.27) and (2.28) by using (2.23) and (2.24), we have

$$\begin{aligned} 4\dot{c}_0 + 2\dot{c}_1 &= v_0 + \left(\frac{h}{3}\right) \dot{h}_1(t), \text{ for } j = 0, \\ \dot{c}_{j-1} + 4\dot{c}_j + \dot{c}_{j+1} &= v_j, \text{ for } j = 1, \ldots, N-1, \\ 2\dot{c}_{N-1} + 4\dot{c}_N &= v_N - \left(\frac{h}{3}\right) \dot{h}_2(t), \text{ for } j = N. \end{aligned} \tag{2.29}$$

$$\begin{aligned} \dot{v}_0 &= \frac{-6\alpha}{h^2}(-2c_0 + 2c_1 - (\frac{h}{3})h_1(t)) - g(4c_0 + 2c_1 - (\frac{h}{3})h_1(t)) + f(x_0, t), \text{ for } j = 0, \\ \dot{v}_j &= \frac{-6\alpha}{h^2}(c_{j-1} - 2c_j + c_{j+1}) - g(c_{j-1} + 4c_j + c_{j+1}) + f(x_j, t), \text{ for } j = 1, \ldots, N-1, \\ \dot{v}_N &= \frac{-6\alpha}{h^2}(2c_{N-1} - 2c_N + (\frac{h}{3})h_2(t)) - g(2c_{N-1} + 4c_N + (\frac{h}{3})h_2(t)) + f(x_N, t), \text{ for } j = N. \end{aligned} \tag{2.30}$$

Systems (2.29) and (2.30) are represented in the compact form as:

$$\begin{aligned} B\dot{\mathbf{c}} &= M, \\ \dot{V} &= X, \end{aligned} \tag{2.31}$$

where

$$B = \begin{bmatrix} 4 & 2 & \dots & \dots & 0 \\ 1 & 4 & 1 & \dots & 0 \\ & & \dots & \dots & \dots \\ & & \dots & \dots & \dots \\ & & 1 & 4 & 1 \\ & & & 2 & 4 \end{bmatrix}_{(N+1)\times(N+1)}$$

M and X are $(N+1)$ order vectors represent the rhs of the systems (2.29) and (2.30), respectively.

To compute the approximate solution $U(x,t)$, we need the vector $\mathbf{c}$. So for Dirichlet boundary conditions, system $A\dot{\mathbf{c}} = G$ in equation (2.19) is solved by using Thomas algorithm to get a vector $\dot{c}$. Then obtained first order ODEs with the second system in (2.19) i.e $\dot{V} = D$ will give $(2N+2)$ first order ordinary differential equations. Finally $(2N+2)$ first order ODEs have been solved by SSP-RK54 [73] scheme and consequently the approximate solution $U(x,t)$ is computed. For Neumann boundary, system (2.31) can be solved using the same procedure and approximate solution can be computed.

2.3.3 Initial Vectors

The initial vectors $\mathbf{c}^0$ and $\mathbf{v}^0$ can be obtained from the given initial conditions (2.2), as the following expressions:

2.3.3.1 Initial Vector $\mathbf{c}^0$ for Dirichlet Boundary Conditions

We consider

$$\begin{aligned} U(x_0,0) &= f_1(0), \ \ j=0, \\ U(x_j,0) &= u_0(x_j), \ j=1,2,\dots,N-1, \\ U(x_N,0) &= f_2(0), \ j=N. \end{aligned}$$

This gives $(N+1)$ linear equations in $(N+1)$ unknowns, which can be written in the form

$$A\mathbf{c^0} = F^0, \tag{2.32}$$

where $\mathbf{c}^0 = [c_0^0, \, c_1^0, \, \dots, c_{N-1}^0, \, c_N^0]^T, F^0 = [f_1(0), \ u_0(x_1), \ \dots\dots, u_0(x_{N-1}), \ f_2(0)]^T$. Using Thomas-algorithm, solution of (2.32) can be easily computed.

2.3.3.2 Initial Vector $\mathbf{c}^0$ for Neumann Boundary Conditions

The initial vector $\mathbf{c}^0$ for Neumann case can be computed as

$$U_x(x_0,0) = h_1(0),$$
$$U(x_j,0) = u_0(x_j),\ j = 0,1,2,\ldots,N-1,N,$$
$$U_x(x_N,0) = h_2(0).$$

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

$$B\mathbf{c^0} = H^0, \tag{2.33}$$

where $H^0 = [u_0(x_0) + (\frac{h}{3})h_1(0),\ u_0(x_1),\ \ldots\ldots, u_0(x_{N-1}),\ u_0(x_N) - (\frac{h}{3})h_2(0)]^T$.
Using Thomas-algorithm in (2.33), initial vector $\mathbf{c}^0$ can be computed.

2.3.3.3 Initial Vector $\mathbf{v}^0$

Initial vector v^0 for both the above cases can be found using the initial condition (2.2) as:

$$\begin{aligned} U_t(x,0) &= u_1(x),\\ v(x_j,0) &= u_1(x_j),\ \ j = 0,1,\ldots,N-1,N. \end{aligned} \tag{2.34}$$

2.4 Numerical Solution of Yukawa-Coupled KGS Equation

In this section, we consider the system (2.7) and let

$$\begin{aligned} \psi(x,t) &= p(x,t) + iq(x,t),\\ \phi_t(x,t) &= w(x,t), \end{aligned}$$

where $p(x,t)$ and $q(x,t)$ are real valued functions. Using above substitutions, we get the following system of partial differential equations

$$\begin{aligned} p_t &= -\frac{1}{2}q_{xx} - \phi q,\\ q_t &= \frac{1}{2}p_{xx} + \phi p,\\ \phi_t &= w,\\ w_t &= \phi_{xx} - \phi + p^2 + q^2, \end{aligned} \tag{2.35}$$

with the initial conditions (2.8) and Dirichlet boundary conditions (2.9).

We assume approximate solutions $p_N(x,t)$, $q_N(x,t)$ and $\phi_N(x,t)$ to the exact solutions $p(x,t)$, $q(x,t)$ and $\phi(x,t)$ respectively, using the collocation of modified cubic B-spline basis functions (1.6) as

$$\begin{aligned} p_N(x,t) &= \sum_{j=0}^{N} \alpha_j(t)\tilde{B}_j(x), \\ q_N(x,t) &= \sum_{j=0}^{N} \beta_j(t)\tilde{B}_j(x), \\ \phi_N(x,t) &= \sum_{j=0}^{N} \gamma_j(t)\tilde{B}_j(x). \end{aligned} \tag{2.36}$$

From equation (2.35), we have

$$\begin{aligned} &\sum_{j=0}^{N} \dot{\alpha}_j(t)\tilde{B}_j(x_i) = -\frac{1}{2}\left(\sum_{j=0}^{N} \beta_j(t)\tilde{B}_j''(x_i)\right) - \left(\sum_{j=0}^{N} \gamma_j(t)\tilde{B}_j(x_i)\right)\left(\sum_{j=0}^{N} \beta_j(t)\tilde{B}_j(x_i)\right), \\ &\sum_{j=0}^{N} \dot{\beta}_j(t)\tilde{B}_j(x_i) = \frac{1}{2}\left(\sum_{j=0}^{N} \alpha_j(t)\tilde{B}_j''(x_i)\right) + \left(\sum_{j=0}^{N} \gamma_j(t)\tilde{B}_j(x_i)\right)\left(\sum_{j=0}^{N} \alpha_j(t)\tilde{B}_j(x_i)\right), \\ &\sum_{j=0}^{N} \dot{\gamma}_j(t)\tilde{B}_j(x_i) = w_i, \\ &\dot{w}_i = \left(\sum_{j=0}^{N} \gamma_j(t)\tilde{B}_j''(x_i)\right) - \left(\sum_{j=0}^{N} \gamma_j(t)\tilde{B}_j(x_i)\right) + \left(\sum_{j=0}^{N} \alpha_j(t)\tilde{B}_j(x_i)\right)^2 + \\ &\left(\sum_{j=0}^{N} \beta_j(t)\tilde{B}_j(x_i)\right)^2, \quad i = 0, 1, \ldots, N. \end{aligned} \tag{2.37}$$

Using modified cubic B-spline basis functions (1.6), Table-1.2 and applying the boundary conditions (2.9) at boundary points, we get the following systems

$$\begin{gathered} 6\dot{\alpha}_0 = 0, \ \text{ for } j = 0, \\ \dot{\alpha}_{j-1} + 4\dot{\alpha}_j + \dot{\alpha}_{j+1} = -\frac{3}{h^2}(\beta_{j-1} - 2\beta_j + \beta_{j+1}) - \\ (\gamma_{j-1} + 4\gamma_j + \gamma_{j+1})(\beta_{j-1} + 4\beta_j + \beta_{j+1}), \ \text{ for } j = 1, 2, \ldots, N-1, \\ 6\dot{\alpha}_N = 0, \ \text{ for } j = N, \end{gathered} \tag{2.38}$$

$$
\begin{aligned}
6\dot{\beta}_0 &= 0, \ \text{for}\, j = 0, \\
\dot{\beta}_{j-1} + 4\dot{\beta}_j + \dot{\beta}_{j+1} &= \frac{3}{h^2}(\alpha_{j-1} - 2\alpha_j + \alpha_{j+1}) + \\
&(\gamma_{j-1} + 4\gamma_j + \gamma_{j+1})(\alpha_{j-1} + 4\alpha_j + \alpha_{j+1}), \ \text{for}\, j = 1, 2, \ldots, N-1, \\
6\dot{\beta}_N &= 0, \ \text{for}\, j = N,
\end{aligned}
\tag{2.39}
$$

$$
\begin{aligned}
6\dot{\phi}_0 &= 0, \ \text{for}\, j = 0, \\
\dot{\phi}_{j-1} + 4\dot{\phi}_j + \dot{\phi}_{j+1} &= w_j, \ \text{for}\, j = 1, 2, \ldots, N-1, \\
6\dot{\phi}_N &= 0, \ \text{for}\, j = N,
\end{aligned}
\tag{2.40}
$$

$$
\begin{aligned}
\dot{w}_0 &= 0, \ \text{for}\, j = 0, \\
\dot{w}_j &= \frac{6}{h^2}(\gamma_{j-1} - 2\gamma_j + \gamma_{j+1}) - (\gamma_{j-1} + 4\gamma_j + \gamma_{j+1}) + (\alpha_{j-1} + 4\alpha_j + \alpha_{j+1})^2 + \\
&(\beta_{j-1} + 4\beta_j + \beta_{j+1})^2, \ \text{for}\, j = 1, 2, \ldots, N-1, \\
\dot{w_N} &= 0, \ \text{for}\, j = N.
\end{aligned}
\tag{2.41}
$$

Systems (2.38)-(2.41) can be written in the compact form as:

$$
\begin{aligned}
A\dot{\boldsymbol{\alpha}} &= F_1, \\
A\dot{\boldsymbol{\beta}} &= F_2, \\
A\dot{\boldsymbol{\gamma}} &= F_3, \\
\dot{\boldsymbol{w}} &= F_4,
\end{aligned}
\tag{2.42}
$$

where
$\dot{\boldsymbol{\alpha}} = \begin{bmatrix} \dot{\alpha}_0, & \dot{\alpha}_1, & \ldots & , \dot{\alpha}_{N-1}, & \dot{\alpha}_N \end{bmatrix}^T$, $\dot{\boldsymbol{\beta}} = \begin{bmatrix} \dot{\beta}_0, & \dot{\beta}_1, & \ldots & , \dot{\beta}_{N-1}, & \dot{\beta}_N \end{bmatrix}^T$,
$\dot{\boldsymbol{\gamma}} = \begin{bmatrix} \dot{\gamma}_0, & \dot{\gamma}_1, & \ldots & , \dot{\gamma}_{N-1}, & \dot{\gamma}_N \end{bmatrix}^T$, $\dot{\boldsymbol{w}} = \begin{bmatrix} \dot{w}_0, & \dot{w}_1, & \ldots & , \dot{w}_{N-1}, & \dot{w}_N \end{bmatrix}^T$.
A is tridiagonal matrix of order $(N+1)$, detailed in the Section-(2.3.1). F_1, F_2, F_3, F_4 are column vectors of order $(N+1)$ representing rhs of equations (2.38-2.41), respectively. The system (2.42) represents a system of $(4N+4)$ first order ODEs and solved by SSP-RK54 scheme with Thomas algorithm. Once the vectors $\boldsymbol{\alpha}$, $\boldsymbol{\beta}$ and $\boldsymbol{\gamma}$ are computed, using them we can easily find the approximate solutions $\psi_N(x,t)$ and $\phi_N(x,t)$ at the require knots.

2.4.1 Initial Vectors

The initial vectors $\boldsymbol{\alpha}^{\mathbf{0}}$, $\boldsymbol{\beta}^{\mathbf{0}}$, $\boldsymbol{\gamma}^{\mathbf{0}}$ and $\mathbf{w}^{\mathbf{0}}$ can be computed from the initial conditions (2.8), using the same procedure as given in the Section-2.3.3.

2.5 Numerical Results

To prove the effectiveness of the method, six numerical examples are taken and maximum absolute error, L_2 error and root mean square error (RMS) are computed to assess the accuracy of the method.

Example 2.1

We consider the Klein-Gordon equation with quadratic nonlinearity [56, 123, 175],

$$u_{tt} - u_{xx} + u^2 = -x\cos(t) + x^2\cos^2(t),\ -1 < x < 1,\ t > 0, \tag{2.43}$$

with initial conditions

$$u(x,0) = x,\ u_t(x,0) = 0. \tag{2.44}$$

The exact solution is given by

$$u(x,t) = x\cos(t). \tag{2.45}$$

The Dirichlet boundary conditions (2.3) are obtained from the exact solution.

In Table-2.1, results are reported with $h = .02$ and $\Delta t = .01$, $.001$, respectively. L_2, L_∞, RMS errors and CPU time are given in Table-2.2 with $h = .02$, $\Delta t = .0001$ and compared with the results given by Dehghan and Shokri [56]. We found that our solutions are better and CPU time is very less in our case. In our next computation we take $h = .02$, $\Delta t = .00002$. Results are reported in Table-2.3 and compared with the results of Li et al. [130]. From table it is clearly seen that our results are much better than that of [130]. In Table-2.4, errors are reported at different time levels with $h = .1$ and $\Delta t = .0001$ and compared with those given by Lakestani and Dehghan [123]. A comparison of exact and numerical solutions at $t = 20$ with $h = .02$ and $\Delta t = .0001$ is depicted in Figure-2.1(a) and corresponding space-time graph is shown in Figure-2.1(b).

Example 2.2

We consider the following Klein-Gordon equation with cubic nonlinearity [56, 130] with constants $\alpha = -2.5, \beta = 1, \gamma = 1.5$,

$$u_{tt} + \alpha u_{xx} + \beta u + \gamma u^3 = 0,\ 0 < x < 1,\ t > 0, \tag{2.46}$$

with initial conditions

$$u(x,0) = B\tan(Kx),\ u_t(x,0) = BKc\sec^2(Kx). \tag{2.47}$$

The exact solution is given by

$$u(x,t) = B\tan(K(x+ct)) \tag{2.48}$$

where $B = \sqrt{\frac{\beta}{\gamma}}, K = \sqrt{\frac{-\beta}{2(\alpha+c^2)}}$.

The Dirichlet boundary conditions (2.3) can be obtained from the exact solution.

L_2, L_∞, RMS errors and CPU time with $h = .01$, $\Delta t = .001$, for $c = .05$ and $c = 0.5$ are reported in Table-2.5 and compared with those given by Dehghan and Shokri [56]. We noticed that our results are compatible with [56] but CPU time is less in our computation. Next we take $h = .01$, $\Delta t = .00005$ and compare our results with those of Li et al. [130], in Table-2.6. We note that our method gives more accurate results compared to the results in [130]. Figure-2.2(a) and Figure-2.2(b) depict the comparison of exact and approximate solutions at $t = 4$ for $c = .05$ and $c = 0.5$ with $h = .01$, $\Delta t = .001$.

Example 2.3

Now we consider the KG equation with cubic nonlinearity [42, 50],

$$u_{tt} - u_{xx} + u + u^3 = 0,\ 0 < x < 1.28,\ t > 0, \tag{2.49}$$

with initial conditions

$$u(x,0) = A\left[1 + \cos\left(\frac{2\pi x}{1.28}\right)\right],\ u_t(x,0) = 0, \tag{2.50}$$

where A is the amplitude.

The Neumann boundary conditions are given by

$$u_x(0,t) = 0,\ u_x(1.28,t) = 0. \tag{2.51}$$

In this problem due to the periodic boundary conditions the continuous solution remains always symmetric with respect to the center of the spatial interval. It has been solved in the literature for different values of A. Bratsos [24] has investigated initial energy $E(0) = 26.5963$ for $A = 1.5$. Dehghan and Ghesmati [42] have reported the initial energy and momentum for $A = 1.5$ and $t \leq 150$. Dehghan et al. [50] have depicted their results for $A = 1.5$ and $t \leq 1000$. In Table-2.7, we report the energy and momentum for $A = 1.5$ for $t \in [0, 1000]$ and found them similar to the results of [42, 50] and it is clear that solution preserves the conservation laws during the propagation. Figure-2.3 depicts the graphs of approximate solutions and energy for $t \in [0, 1000]$ and $A = 1.5$ with $h = .01$, $\Delta t = .001$. In Figure-2.4, approximate solutions are plotted for $A = 100$ and $t \in [0, 7]$ with $h = .01$, $\Delta t = .001$. Approximate solutions at $t = 0, .1, 32$ for amplitude $A = 150$ are illustrated in Figure-2.5, which shows that the solution remains bounded for $t \in [0, 32]$.

Example 2.4

We consider the KG equation with quadratic nonlinearity [56],

$$u_{tt} - u_{xx} + u^2 = 6xt(x^2 - t^2) + x^6 t^6, \ 0 < x < 1, \ t > 0, \tag{2.52}$$

with initial conditions

$$u(x, 0) = 0, \ u_t(x, 0) = 0. \tag{2.53}$$

The exact solution is given by

$$u(x, t) = x^3 t^3. \tag{2.54}$$

Dirichlet boundary conditions (2.3) and Neumann boundary conditions (2.4) can be obtained from the exact solution.

For Dirichlet problem, results are reported in Table-2.8 at $t = 1, 2, 3, 4, 5$ with $h = .02$ and $\Delta t = .0001$. Comparisons have been made with the results given by Dehghan and Shokri [56]. We note that obtained solutions are compatible with those given in [56] but CPU time is much less in our case. Table-2.9 shows the comparison of errors at different time with Lakestani and Dehghan [123] with $h = .2$ and $\Delta t = .0001$. A graphical comparison of exact and numerical solutions at $t = 5$ with $h = .02$, $\Delta t = .0001$ is depicted in Figure-2.6.

With Neumann boundary conditions, we have solved above problem with $h = .02$ and $\Delta t = .0001$. Computed results are reported up to $t = 5$ in Table-2.10.

Example 2.5

Single soliton solution

In this example, we consider KGS equations (2.7) having analytic solitary wave solution

$$\begin{aligned}
\psi(x - x_0, t, q) &= \frac{3\sqrt{2}}{4\sqrt{1-q^2}} \operatorname{sech}^2 \frac{1}{2\sqrt{1-q^2}}(x - qt - x_0)\ e^{\left(i\left(qx + \frac{1-q^2+q^4}{2(1-q^2)}t\right)\right)}, \\
\phi(x - x_0, t, q) &= \frac{3}{4(1-q^2)} \operatorname{sech}^2 \frac{1}{2\sqrt{1-q^2}}(x - qt - x_0),
\end{aligned} \tag{2.55}$$

where $|q| < 1$ represents the propagating velocity of wave and x_0 is the initial phase. The initial conditions (2.8) are given as

$$\begin{aligned}
\psi_0(x) &= \frac{3\sqrt{2}}{4\sqrt{1-q^2}} \operatorname{sech}^2 \frac{1}{2\sqrt{1-q^2}}(x - x_0)\ e^{(iqx)}, \\
\phi_0(x) &= \frac{3}{4(1-q^2)} \operatorname{sech}^2 \frac{1}{2\sqrt{1-q^2}}(x - x_0), \\
\phi_1(x) &= \frac{3q}{4(1-q^2)^{\frac{3}{2}}} \operatorname{sech}^2 \frac{1}{2\sqrt{1-q^2}}(x - x_0) \tanh \frac{1}{2\sqrt{1-q^2}}(x - x_0).
\end{aligned} \tag{2.56}$$

We take initial phase $x_0 = -10$ and initial velocity $q = 0.8$. Solutions are obtained in the spatial interval $[-20, 20]$ up to $t = 30$. The evolutions of $|\psi_N(x,t)|$ and $\phi_N(x,t)$ at different time levels are depicted in Figure-2.7 and Figure-2.8, respectively. Next, we take $x_0 = 0$ and $q = 0.5$ in the domain $[-40, 40]$ and solve the problem till $t = 40$ with $h = .1$ and $\Delta t = .001$. L_2 and L_∞ error norms of $|\psi_N(x,t)|$ and $\phi_N(x,t)$ are reported in Table-2.11.

Example 2.6

Collision of two solitons

In this example, we consider KGS equations (2.7) with the following initial conditions

$$\begin{aligned}
\psi_0(x) &= \psi(x - x_1, 0, q_1) + \psi(x - x_2, 0, q_2), \\
\phi_0(x) &= \phi(x - x_1, 0, q_1) + \phi(x - x_2, 0, q_2), \\
\phi_1(x) &= \phi_t(x - x_1, 0, q_1)|_{t=0} + \phi_t(x - x_2, 0, q_2)|_{t=0},
\end{aligned} \tag{2.57}$$

where x_1, x_2 are the initial phases and q_1, q_2 are propagating velocities of two solitons.

Interaction of two symmetric solitons: First we consider the collision of two symmetric solitons. The two solitons are said to be symmetric when they are symmetrically distributed around the origin. We consider the collision of two solitons with same velocities but different directions. We take $x_1 = 20$, $x_2 = -20$, $q_1 = -0.5$, $q_2 = 0.5$ and solve the problem in the domain $[-40, 40]$ with $h = .16$ and $\Delta t = .001$. The graphs of interaction of solitons at different times are depicted in Figure-2.9. We note that the two solitary waves collides at $t = 40$ and propagate in their original directions after collision.

Interaction of two asymmetric solitons: Next we consider two solitons propagating with both different velocities and opposite directions. The parameters $x_1 = -15$, $x_2 = 15$ and $q_1 = 0.8$, $q_2 = -0.4$ are chosen in the space domain $[-40, 40]$. In Figure-2.10, graphs of interaction of solitons at different time levels are depicted with $h = .16$ and $\Delta t = .001$. We observe that two solitons preserve their own shapes and velocities unchanged before collision. At $t = 20$ collision occurs, results in a fusion and are accompanied by a series of emission of waves after interaction.

2.6 Conclusions

The following observations have been made based on present study:

1. The cubic B-spline collocation method has been developed to solve Klein-Gordon and coupled Klein-Gordon Schrödinger equations.

2. The method is able to handle Dirichlet's as well as Neumann's boundary conditions.

3. The solutions of these equations are obtained without using any transformation and linearization process.

4. The accuracy of the method is tested by solving six numerical examples known in the literature. It has been proved that the proposed method produces better results in comparison to those available in literature. However, we obtain these results in much less CPU time.

5. This scheme provides the solution not only at the grid points but at any point in the solution domain.

Table 2.1: Errors of Example 2.1 with $h = .02$.

t	$\Delta t = .01$		$\Delta t = .001$	
	L_2	L_∞	L_2	L_∞
1	3.3759E-3	4.1286E-3	3.3796E-4	4.1294E-4
3	6.3474E-4	6.8066E-4	6.3647E-5	7.0137E-5
5	3.8419E-3	4.7009E-3	3.8720E-4	4.7217E-4
7	2.7703E-3	3.2785E-3	2.7262E-4	3.2415E-4
10	2.3023E-3	2.7282E-3	2.6212E-4	2.7022E-4
15	2.6794E-3	3.1783E-3	2.6766E-4	3.1886E-4
20	3.5556E-3	4.4488E-3	3.5493E-4	4.4491E-4

Table 2.2: Errors and CPU time of Example 2.1 with $h = .02$ and $\Delta t = .0001$.

t	Proposed method				Dehghan and Shokri [56]			
	L_2	L_∞	RMS	CPU time(s)	L_2	L_∞	RMS	CPU time(s)
1	3.3799E-5	4.1295E-5	2.3663E-6	3.07	6.5422E-5	1.2540E-5	6.5097E-6	5
3	6.3684E-6	7.0385E-6	4.4586E-7	8.09	1.1717E-4	1.5554E-5	1.1659E-5	22
5	3.8721E-5	4.7220E-5	2.7108E-6	12.97	2.2011E-4	3.3792E-5	2.1902E-5	49
7	2.2727E-5	3.2420E-5	1.9091E-6	14.04	2.5892E-4	3.7753E-5	2.5763E-5	83
10	2.2571E-5	2.7001E-5	1.5802E-6	17.45	7.9854E-5	1.3086E-5	7.9458E-6	150
15	2.6763E-5	3.1901E-5	1.8737E-6	32	-	-	-	-
20	3.5485E-5	4.4490E-5	2.4843E-6	60	-	-	-	-

Table 2.3: Errors of Example 2.1 with $h = .02$ and $\Delta t = .00002$.

t	Proposed method			Li et al. [130]		
	L_2	L_∞	RMS	L_2	L_∞	RMS
1	6.7600E-6	8.2590E-6	4.7327E-7	1.9558E-3	1.1135E-3	1.1294E-4
3	1.2736E-6	1.4074E-6	8.9608E-7	1.3664E-3	7.6676E-3	7.6295E-4
5	7.7440E-6	9.4439E-6	5.4216E-7	1.5260E-3	8.5602E-3	8.5178E-4
7	5.4538E-6	6.4841E-6	3.8183E-7	2.5892E-3	3.7753E-3	2.5763E-4
10	4.5133E-6	5.3996E-6	3.1598E-7	1.0465E-3	6.9848E-3	6.9501E-4

Table 2.4: Errors and CPU time of Example 2.1 with $h = .1$ and $\Delta t = .0001$.

t	Proposed method				Lakestani and Dehghan [123]	
	L_2	L_∞	RMS	CPU time(s)	L_2	L_∞
.1	2.0292E-7	3.2048E-7	4.1253E-7	.06	4.1E-5	6.7E-5
.2	1.3572E-6	2.1884E-6	2.7590E-7	.08	3.6E-5	4.2E-5
.3	3.5537E-6	5.7855E-6	7.2241E-7	.16	4.9E-5	5.6E-5
.4	6.4119E-6	1.0226E-5	1.3034E-7	.18	5.4E-5	7.3E-5
.5	9.6776E-6	1.4751E-5	1.9672E-7	.23	6.3E-5	8.3E-5
.6	1.3255E-5	1.9158E-5	2.6944E-7	.27	4.5E-5	6.6E-5
.7	1.7052E-5	2.3398E-5	3.4663E-7	.34	5.8E-5	7.4E-5
.8	2.0995E-5	2.7366E-5	4.2678E-7	.37	7.0E-5	8.1E-5
.9	2.5130E-5	3.1020E-5	5.1085E-7	.42	5.2E-5	7.9E-5
1	2.9376E-5	3.4281E-5	5.9717E-7	.44	5.3E-5	7.7E-5

Table 2.5: Errors and CPU time of Example 2.2 with $h = .01$ and $\Delta t = .001$.

	Proposed method				Dehghan and Shokri [56]			
t	L_2	L_∞	RMS	CPU time(s)	L_2	L_∞	RMS	CPU time(s)
$c = .05$								
1	1.8930E-6	3.6481E-6	1.8742E-7	.39	3.6497E-7	1.7861E-6	1.7772E-7	0
2	2.5245E-6	3.5691E-6	2.4996E-7	.60	3.8952E-7	1.5383E-6	1.5306E-7	1
3	2.7593E-6	3.7474E-6	2.7320E-7	.62	4.2123E-7	1.7275E-6	1.7190E-7	1
4	2.1504E-6	3.9616E-6	2.1291E-7	.80	4.5928E-7	2.0097E-6	1.9997E-7	2
$c = .5$								
1	2.2480E-5	3.5702E-5	2.2258E-6	.37	5.9964E-6	4.0761E-5	4.0559E-6	0
2	7.5075E-5	1.3161E-4	7.4332E-6	.51	2.1973E-5	1.5769E-4	1.5691E-5	1
3	2.1963E-4	4.2853E-4	2.1745E-5	.71	9.0893E-5	6.4792E-4	6.4470E-5	1
4	1.8264E-4	2.1806E-3	8.6272E-5	.90	8.2945E-4	5.3572E-3	5.3306E-4	2

Table 2.6: Errors of Example 2.2 with $h = .01$ and $\Delta t = .00005$.

	Present Scheme		Li et al. [130]	
t	L_2	L_∞	L_2	L_∞
$c = .05$				
1	2.8010E-6	3.7221E-6	2.9718E-4	5.6970E-5
2	2.0174E-6	3.8238E-6	3.8699E-4	7.4878E-5
3	3.1641E-6	3.9990E-6	5.2203E-4	1.1972E-4
4	2.7095E-6	4.7432E-6	4.4143E-4	1.4008E-4
$c = .5$				
1	3.6802E-6	6.7946E-6	6.6508E-4	1.4189E-4
2	8.2992E-6	1.5770E-5	1.5438E-3	4.6601E-4
3	2.7509E-5	7.0254E-5	4.9588E-3	1.9445E-3
4	3.2482E-4	1.2584E-3	7.1870E-2	2.8219E-2

Table 2.7: The energy $E(t)$ and momentum $|p(t)|$ of Example 2.3 with $h = .01$, $\Delta t = .001$, $A = 1.5$ for $t \in [0, 1000]$.

t	0	10	50	100	150	500	1000
Proposed method							
E(t)	26.5964	26.5984	26.5995	26.5966	26.5987	26.5994	26.5974
$\|p(t)\|$	0	2.060E-15	6.528E-14	8.021E-14	3.406E-14	3.580E-15	2.292E-14
Dehghan and Ghesmati [42]							
$E(t)$	-	26.5957	26.5902	26.5725	26.5308	-	-
$\|p(t)\|$	0	9.419E-11	1.459E-10	5.131E-8	6.172E-6	-	-

Table 2.8: Errors and CPU time of Example 2.4 with $h = .02$ and $\Delta t = .0001$.

t	Proposed method				Dehghan and Shokri [56]			
	L_2	L_∞	RMS	CPU time(s)	L_2	L_∞	RMS	CPU time(s)
1	8.9965E-5	2.3442E-4	1.2473E-5	1.26	1.1012E-5	5.4998E-5	5.4725E-6	6
2	1.8101E-3	2.4415E-3	1.6362E-4	2.45	1.6496E-4	1.1522E-3	1.1465E-4	14
3	2.6825E-3	8.2711E-3	3.7193E-4	3.02	5.9728E-4	3.2588E-3	3.2426E-4	25
4	4.6003E-3	1.8738E-2	6.3782E-4	3.60	1.8264E-3	9.8191E-3	9.7704E-4	37
5	7.1257E-3	3.4252E-2	9.8797E-4	4.87	3.6915E-3	1.9139E-2	1.9044E-3	52

Table 2.9: Errors and CPU time of Example 2.4 with $h = .2$ and $\Delta t = .0001$.

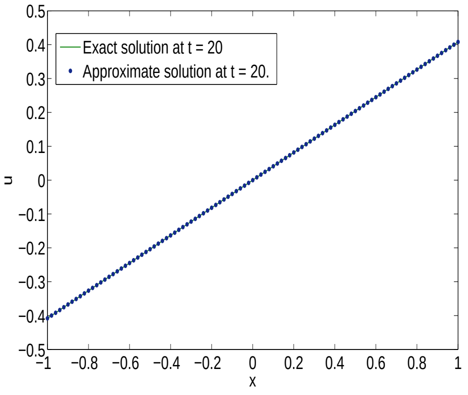
t	Proposed method				Lakestani and Dehghan [123]	
	L_2	L_∞	RMS	CPU time(s)	L_2	L_∞
.1	4.5177E-7	1.0976E-6	1.0088E-7	.02	3.6E-5	5.3E-5
.2	1.8924E-6	5.4978E-6	4.2258E-7	.14	3.9E-5	5.7E-5
.3	9.2599E-6	3.5926E-5	2.0677E-6	.16	2.7E-5	4.1E-5
.4	3.0119E-5	1.0909E-5	6.7433E-6	.18	3.8E-5	5.6E-5
.5	7.3277E-5	2.4346E-5	1.6362E-6	.23	3.2E-5	4.5E-5
.6	1.4837E-5	4.5742E-5	3.3131E-6	.25	3.4E-5	5.9E-5
.7	2.6639E-5	7.6917E-5	5.9485E-5	.32	3.5E-5	5.9E-5
.8	4.3907E-5	1.1963E-5	9.8042E-5	.34	3.1E-5	4.5E-5
.9	6.7862E-5	1.7572E-4	1.5153E-5	.36	3.8E-5	6.3E-5
1	9.9749E-5	2.4676E-4	2.2273E-5	.38	3.3E-5	4.6E-5

Table 2.10: Errors and CPU time of Example 2.4 with $h = .02$ and $\Delta t = .0001$.

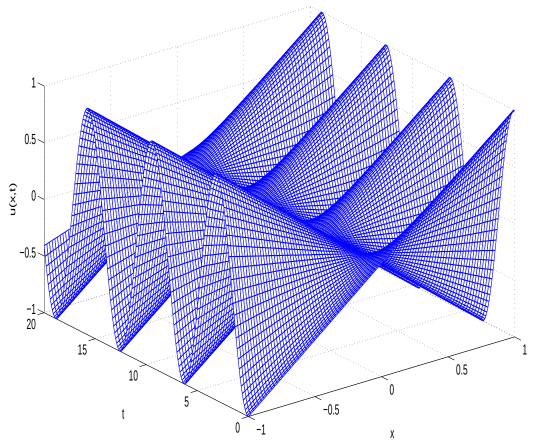
t	L_2	L_∞	RMS	CPU time(s)
1	5.7658E-5	1.4852E-4	5.7089E-6	1.5
2	2.3059E-4	5.9398E-4	2.2830E-5	2.6
3	5.1883E-4	1.3365E-3	5.1369E-5	3.5
4	9.2237E-4	2.3759E-3	9.1323E-5	5.0
5	1.4412E-3	3.7124E-3	1.4269E-4	5.8

Table 2.11: Errors of Example 2.5 with $h = .1$ and $\Delta t = .001$.

	$\lvert\psi\rvert$		ϕ	
t	L_2	L_∞	L_2	L_∞
10	2.4046E-2	1.4552E-2	3.1071E-2	1.2044E-2
20	4.8021E-2	2.8054E-2	6.1636E-2	2.3323E-2
30	7.1836E-2	4.1339E-2	9.2170E-2	3.3013E-2
40	9.5477E-2	5.5029E-2	1.2272E-2	4.3792E-2

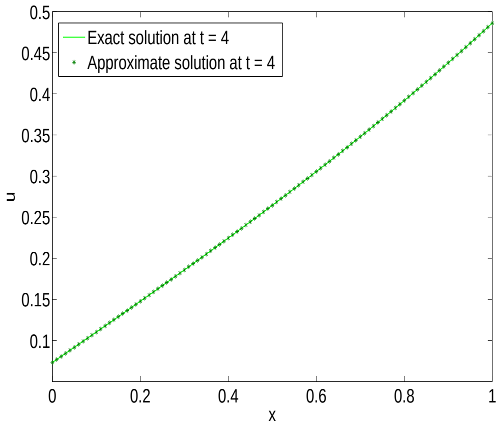


(a) Exact and approximate solution at $t = 20$.

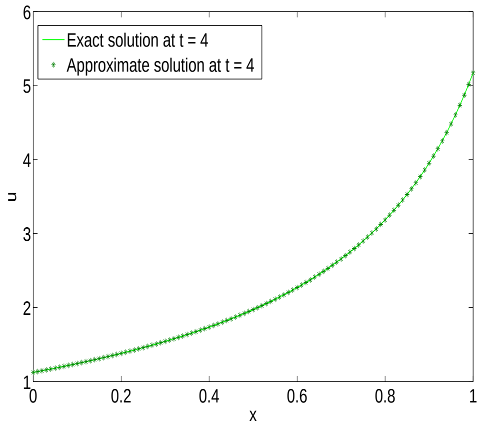


(b) Space-time graph of approximate solution up to $t = 20$.

Figure 2.1: Exact and approximate solutions of Example 2.1 with $h = .02$ and $\Delta t = .0001$.

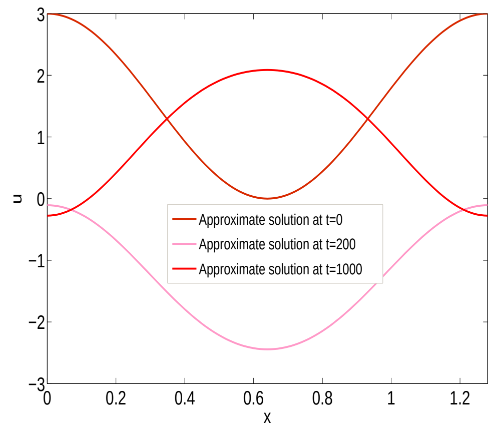


(a) Exact and approximate solution for $c = .05$.

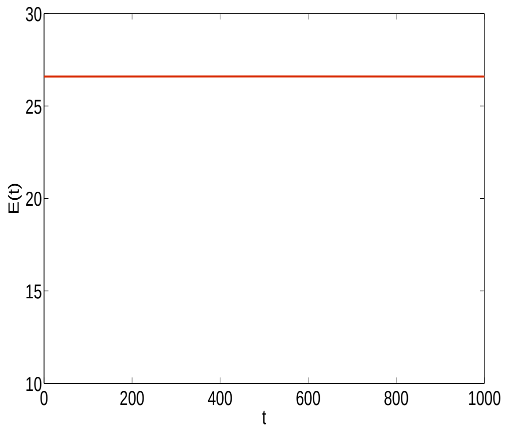


(b) Exact and approximate solution for $c = .5$

Figure 2.2: Exact and approximate solutions of Example 2.2 at $t = 4$ with $h = .01$, $\Delta t = .001$.

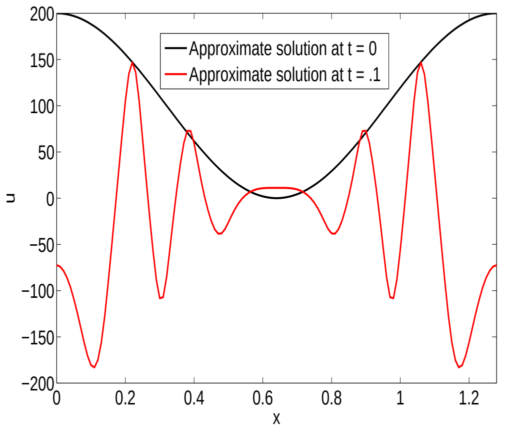


(a) Approximate solutions at $t = 0,\ 200,\ 1000.$

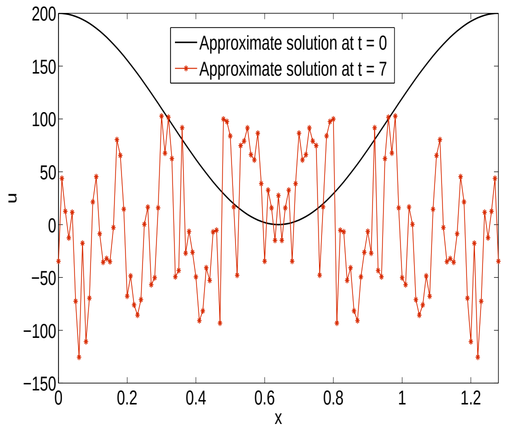


(b) The energy from $t = 0$ to 1000.

Figure 2.3: Approximate solutions and energy of Example 2.3 for $A = 1.5$ at $t = 0, 200, 1000$.

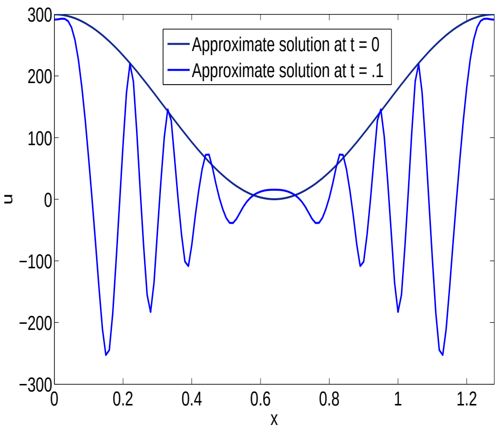


(a) Approximate solutions at $t = 0, .1$

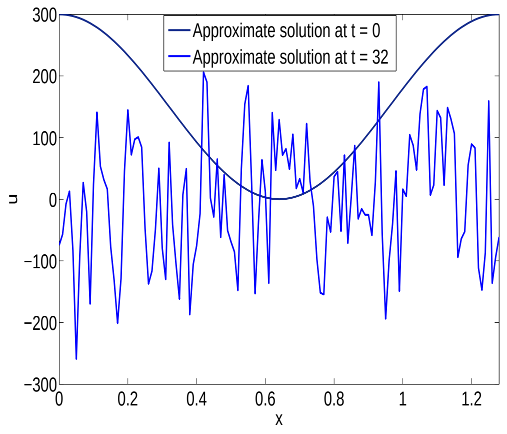


(b) Approximate solutions at $t = 0, 7$.

Figure 2.4: Approximate solutions of Example 2.3 for $A = 100$ at $t = 0, .1, 7$.



(a) Approximate solutions at $t = 0, .1$.



(b) Approximate solutions at $t = 0, 32$.

Figure 2.5: Approximate solutions of Example 2.3 for $A = 150$ at $t = 0$, .1, 32.

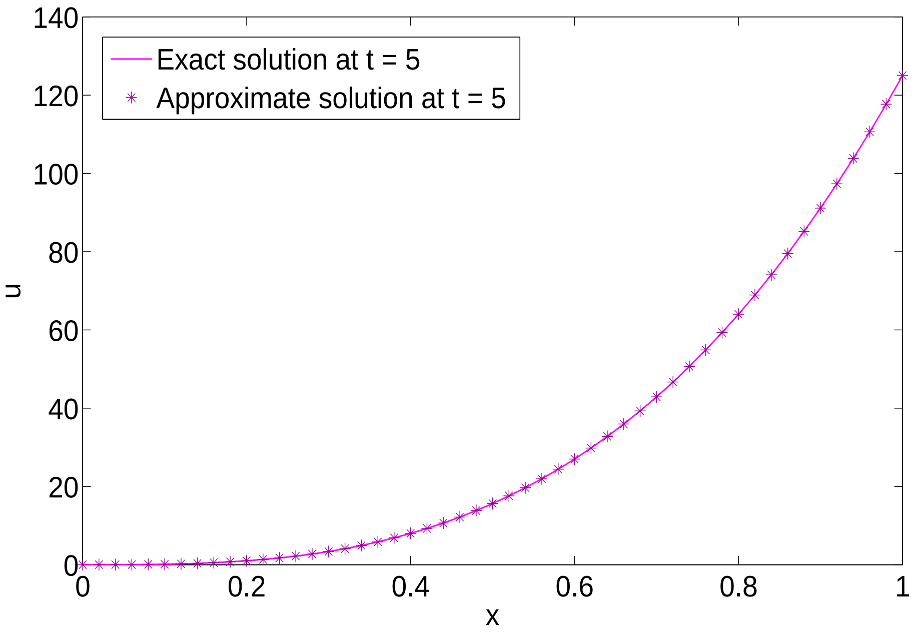


Figure 2.6: Exact and approximate solutions of Example 2.4 with $h = .02$, $\Delta t = .0001$.

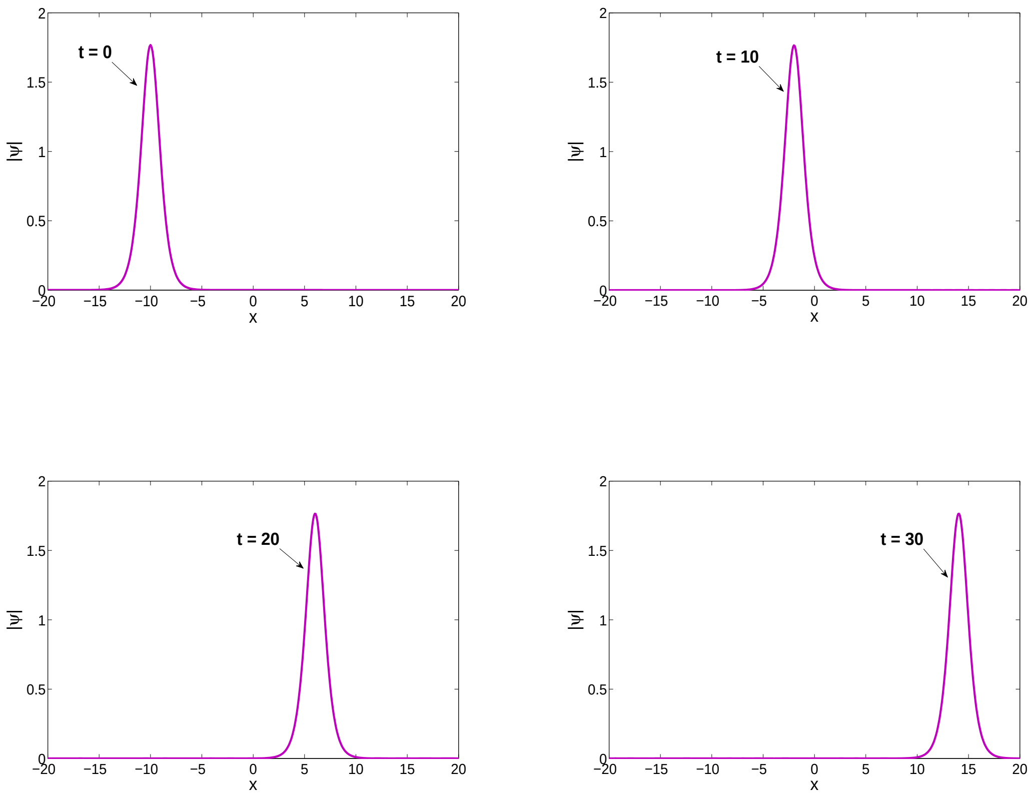


Figure 2.7: Single soliton solutions $|\psi(x,t)|$ of Example 2.5 for $t \in [0, 30]$ with $h = .1$ and $\Delta t = .001$.

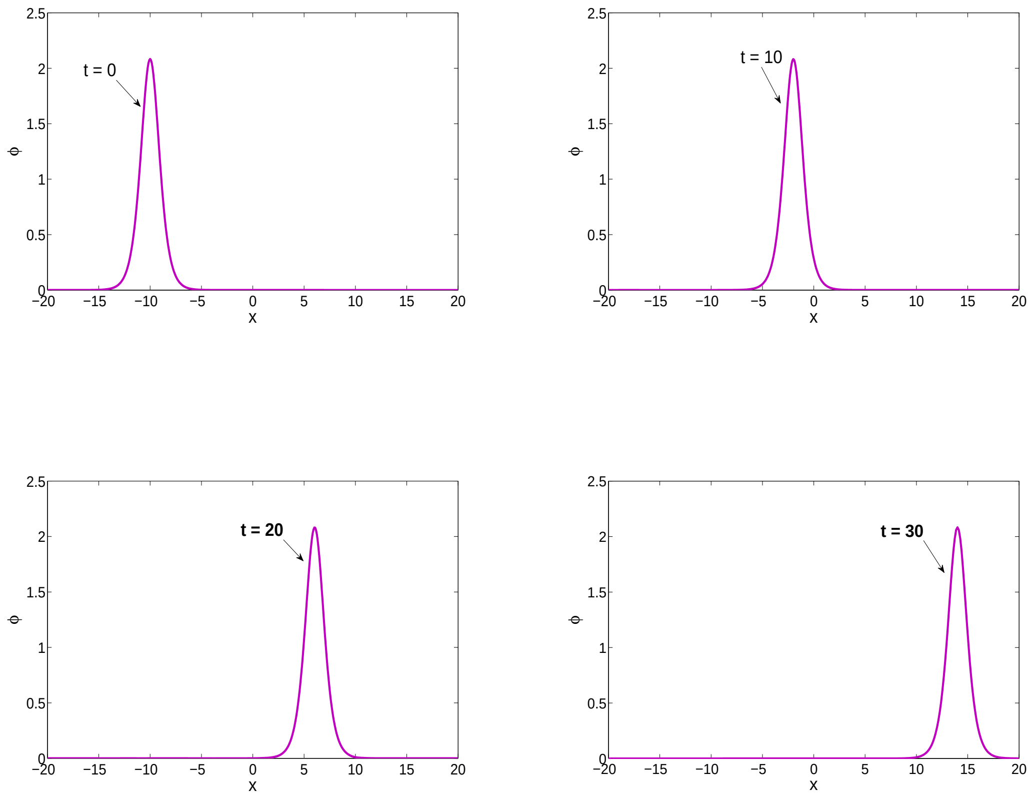


Figure 2.8: Single soliton solutions $\phi(x,t)$ of Example 2.5 for $t \in [0,30]$ with $h = .1$ and $\Delta t = .001$.

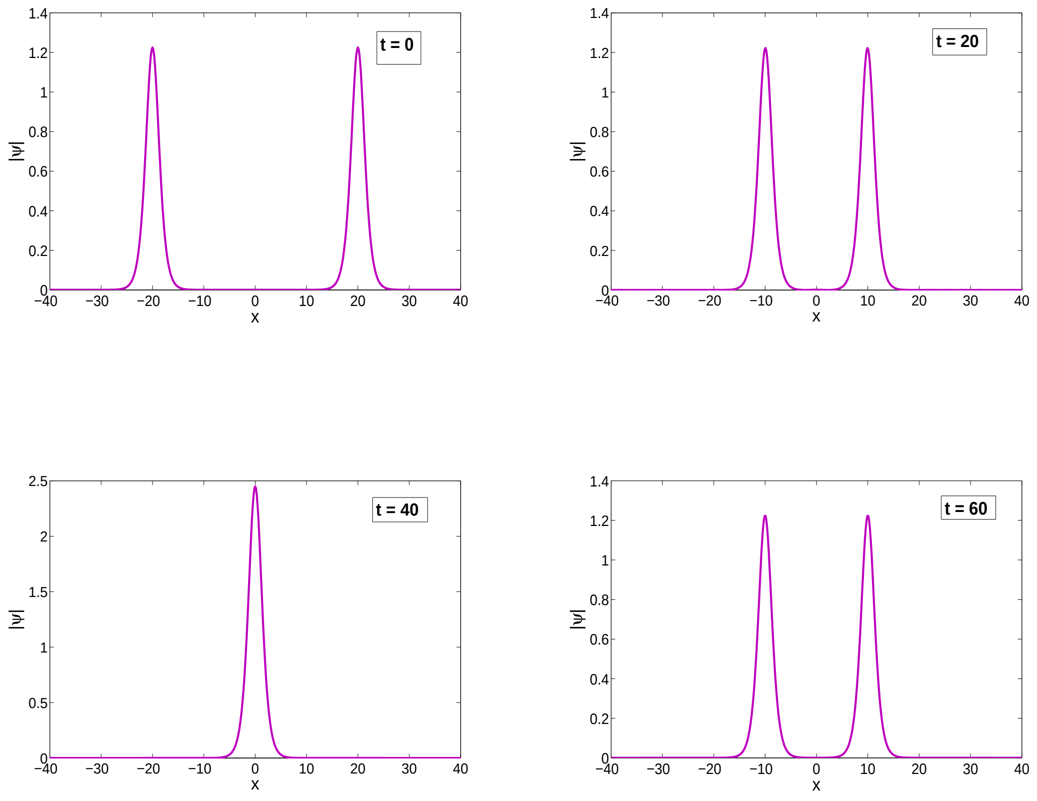


Figure 2.9: Collision of symmetric solitons of Example 2.6 for $t \in [0, 60]$ with $h = .16$ and $\Delta t = .001$.

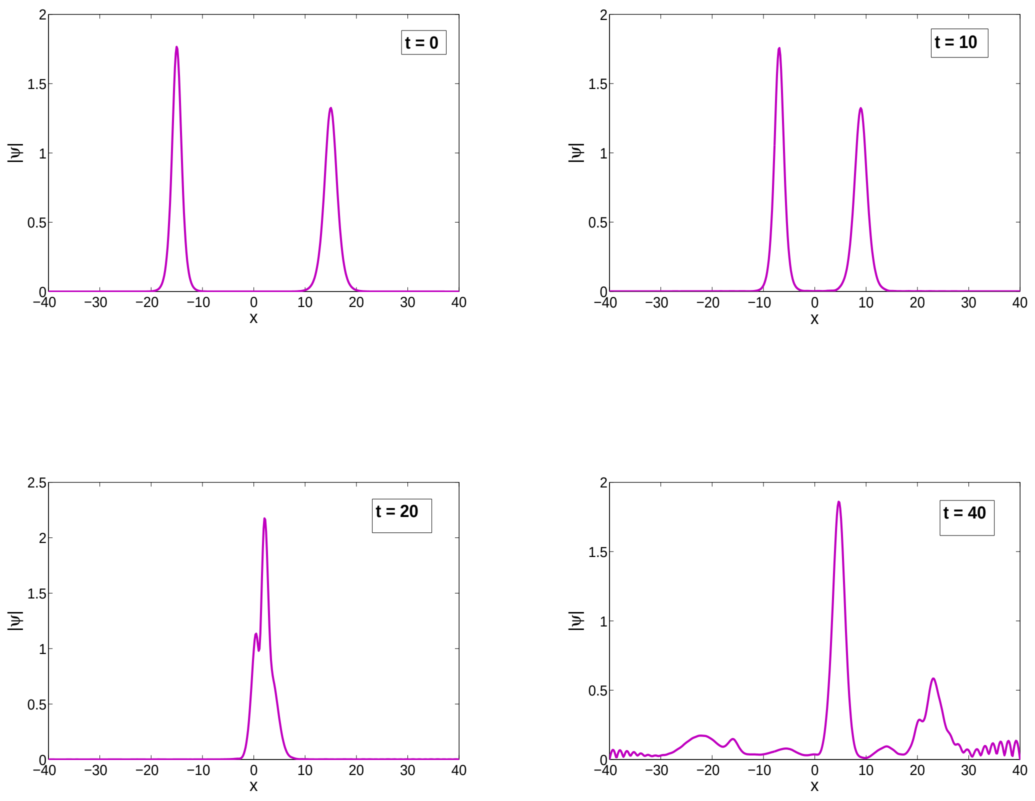


Figure 2.10: Collision of asymmetric solitons of Example 2.6 for $t \in [0, 40]$ with $h = .16$ and $\Delta t = .001$.

Chapter 3

Numerical Solution of Nonlinear Sine-Gordon Equation

3.1 Introduction

The nonlinear sine-Gordon equation appears in many different applications such as differential geometry, propagation of fluxion in Josephson junctions [167], dislocations in crystals, stability of fluid motion, nonlinear physics and applied sciences [13]. The present chapter is concerned with the numerical solution of one-dimensional nonlinear sine-Gordon equation

$$u_{tt} = u_{xx} - \sin(u), \quad x \in (a, b), \;\; t > 0, \tag{3.1}$$

with initial conditions

$$u(x, 0) = \phi_1(x), \quad u_t(x, 0) = \phi_2(x), \;\; x \in [a, b], \tag{3.2}$$

and the following Dirichlet boundary conditions

$$u(a, t) = \psi_1(t), \quad u(b, t) = \psi_2(t), \;\; t \geq 0. \tag{3.3}$$

In the literature a lot of work has been reported for the numerical solution of sine-Gordon equation. Ben-Yu et al. [18] have proposed two difference schemes; Bratsos and Twizell [25] have used method of lines to transform the initial/boundary value problem associated with equation (3.1) in to a first order nonlinear initial value problem. Mohebbi and Dehghan [162] have presented a combination of a compact finite difference approximation of fourth order and a fourth-order A-stable DIRKN method. Kuang and Lu [113] have proposed two classes of finite difference method for generalized sine-Gordon equation; Bratsos and Twizell [26] have converted the equation in to a linear algebraic system using a family of finite difference methods. Wei [207] has used a discrete singular convolution algorithm for solving (3.1). A variational iteration method has been developed by Batiha et al. [15], to obtain approximate analytical solution of the sine-Gordon equation without any discretization. A second order numerical scheme has been presented by Zheng [212] to solve sine-Gordon equation define on whole real axis. Bratsos [23] has used a fourth order numerical method to solve equation (3.1). Dehghan and Shokri [53] have developed a collocation method using radial basis functions; Dehghan and Mirzaei [47] have used a boundary integral equation method; Rashidinia and Mohammadi [178] have developed two implicit finite difference schemes by using spline function approximations. Li-Min and Zong-Min [131] have presented a meshless scheme by using a multiquadric quasi-interpolation without solving a large-scale linear system of equations but a polynomial was needed to improve the accuracy of the scheme, while Jiang and Wang [87] have proposed a meshless approach by using high accuracy MQ quasi-interpolation without using any polynomial. In [68], exp-function method has been used to obtain generalized travelling wave solutions of the MKdV-sine-Gordon and Boussinesq-double sine-Gordon equations with free parameters. A modified decomposition method has been proposed by Kaya [101]. Uddin et al. [200] have proposed a meshfree approach based on radial basis functions to solve the sine-Gordon equation (3.1).

B-splines with collocation provide a simple solution procedure of differential equations. They produce a spline function which is useful to obtain the solution at any point of the domain, while in finite difference methods [18, 26, 113, 162, 178], we can find the solution only at the selected knots. In the present method, approximate solutions of sine-Gordon equation are obtained using modified cubic B-spline collocation method in space and SSP-RK54 scheme [73] in time. The equation is converted into a system of partial differential equations and then using modified B-spline collocation method, it reduces into a system of first order ODEs. Finally, we use SSP-RK54 scheme to solve obtained system of ODEs. The method does not need any extra effort to tackle the non-linearity and hence equations may be solved very easily.

In Section-3.2, cubic B-spline collocation method is described. In Section- 3.3, process for execution of present method is illustrated for equations (3.1)-(3.3), using modified cubic B-spline functions. Initial vectors have been computed in Section-3.4, which are require to start our method. To show the efficiency of the present approach computationally, four numerical experiments are taken in Section-3.5. Finally, brief conclusions drawn from the present study are presented in Section-3.6.

3.2 Description of Method

The solution domain $a \leq x \leq b$ is partitioned into a mesh of uniform length $h = x_{j+1} - x_j$, where $j = 0, 1, 2, ..., N-1$, such that $a = x_0 < x_1 < ..... < x_{N-1} < x_N = b$.

In the cubic B-spline collocation method the approximate solution can be written as the linear combination of cubic B-spline basis functions for the approximation space under consideration. Consider approximate solution $U(x,t)$ to the exact solution $u(x,t)$ in the form

$$U(x,t) = \sum_{j=-1}^{N+1} c_j(t) B_j(x), \tag{3.4}$$

where $c_j(t)$ are the time dependent quantities, to be determined from boundary conditions and collocation from the differential equation.

The values of $B_j(x)$ and its derivatives are tabulated in Table-1.2. Approximate values of $U(x,t)$ and its two derivatives in terms of time parameter c_j are given in equation (1.13).

3.3 Implementation of Method

The sine-Gordon equation (3.1) is rewritten as a coupled equation using the transformation $u_t(x,t) = v(x,t)$ as

$$\begin{aligned} u_t &= v, \\ v_t &= u_{xx} - \sin u. \end{aligned} \tag{3.5}$$

Modified cubic B-spline basis functions (1.6) have been used for handling the Dirichlet boundary conditions. So we assume our approximate solution as the linear combination of modified cubic B-spline basis functions i.e.

$$U(x,t) = \sum_{j=0}^{N} c_j(t)\tilde{B}_j(x). \tag{3.6}$$

Using the approximate solution (3.6), the approximate value of $U_t(x,t)$ can be written as

$$U_t(x,t) = \sum_{j=0}^{N} \dot{c}_j(t)\tilde{B}_j(x), \tag{3.7}$$

where $\dot{c}_j(t)$ is the derivative of $c_j(t)$ with respect to time t.

Using modified cubic B-spline basis functions (1.6) and Table-1.2 in (3.7), the values of $U_t(x,t)$ at different knots are calculated as

$$\begin{aligned} U_t(x_0,t) &= 6\dot{c}_0, \text{ for } j=0,\\ U_t(x_j,t) &= \dot{c}_{j-1} + 4\dot{c}_j + \dot{c}_{j+1}, \text{ for } j=1,2,\ldots,N-1,\\ U_t(x_N,t) &= 6\dot{c}_N, \text{ for } j=N. \end{aligned} \tag{3.8}$$

Using (3.6) in coupled system (3.5) and imposing boundary conditions (3.3) at the boundary points, we have

$$\begin{aligned} U_t(x_0,t) &= \dot{\psi}_1(t), \text{ for } i=0,\\ U_t(x_i,t) &= v_i, \text{ for } i=1,2,\ldots,N-1,\\ U_t(x_N,t) &= \dot{\psi}_2(t), \text{ for } i=N. \end{aligned} \tag{3.9}$$

$$\begin{aligned} v_t(x_0,t) &= \ddot{\psi}_1(t), \quad \text{for } i=0,\\ v_t(x_i,t) &= \sum_{j=0}^{N} c_j\tilde{B}_j''(x_i) - \sin(U(x_i,t)), \text{ for } i=1,2,\ldots,N-1,\\ v_t(x_N,t) &= \ddot{\psi}_2(t), \text{ for } i=N. \end{aligned} \tag{3.10}$$

Now using (3.8) in (3.9), and using (1.6), Table-1.2 in (3.10), we obtain the following system of equations

$$\begin{aligned} 6\dot{c}_0 &= \dot{\psi}_1(t), \text{ for } j=0,\\ \dot{c}_{j-1} + 4\dot{c}_j + 4\dot{c}_{j+1} &= v_j, \text{ for } j=1,2,\ldots,N-1,\\ 6\dot{c}_N &= \dot{\psi}_2(t), \text{ for } j=N. \end{aligned} \tag{3.11}$$

$$\begin{aligned} \dot{v}_0 &= \ddot{\psi}_1(t), \ \text{ for } j = 0, \\ \dot{v}_j &= \frac{6}{h^2}(c_{j-1} - 2c_j + c_{j+1}) - \sin(c_{j-1} + 4c_j + c_{j+1}), \ \text{ for } j = 1, 2, \ldots, N-1, \\ \dot{v}_N &= \ddot{\psi}_2(t), \ \text{ for } j = N. \end{aligned} \tag{3.12}$$

The systems (3.11) and (3.12) represent the system of first order ODEs, which can be expressed as

$$A\dot{c} = F,$$
$$\dot{V} = G.$$

i.e.

$$\begin{bmatrix} 6 & 0 & \ldots & \ldots & 0 \\ 1 & 4 & 1 & \ldots & 0 \\ & & \ldots & \ldots & \ldots \\ & & \ldots & \ldots & \ldots \\ & & 1 & 4 & 1 \\ & & & 0 & 6 \end{bmatrix} \begin{bmatrix} \dot{c}_0 \\ \dot{c}_1 \\ \ldots \\ \ldots \\ \dot{c}_{N-1} \\ \dot{c}_N \end{bmatrix} = \begin{bmatrix} F_0 \\ F_1 \\ \ldots \\ \ldots \\ F_{N-1} \\ F_N \end{bmatrix} \tag{3.13}$$

$$\begin{bmatrix} \dot{v}_0 \\ \dot{v}_1 \\ \ldots \\ \ldots \\ \dot{v}_{N-1} \\ \dot{v}_N \end{bmatrix} = \begin{bmatrix} G_0 \\ G_1 \\ \ldots \\ \ldots \\ G_{N-1} \\ G_N \end{bmatrix} \tag{3.14}$$

where $F_0 = \dot{\psi}_1(t), \ \ G_0 = \ddot{\psi}_1(t),$
$F_j = v_j, \ \ G_j = \frac{6}{h^2}(c_{j-1} - 2c_j + c_{j+1}) - \sin(c_{j-1} + 4c_j + c_{j+1}), \ \text{ for } \ j = 1, 2, \ldots, N-1,$
$F_N = \dot{\psi}_2(t), \ \ G_N = \ddot{\psi}_2(t).$

Once the vector $\mathbf{c}$ is determined at a specific time level, using (1.13) we can compute the approximate solution. So first we solve system (3.13) for vector $\dot{\mathbf{c}}$, by using Thomas algorithm only once at each time level $t > 0$. Then obtained system with the system (3.14) will give $(2N+2)$ first order ordinary differential equations, which are solved using SSP-RK54 [73] scheme and consequently the approximate solution $U(x,t)$ is completely known.

3.4 Computation of Initial Vectors

To find the solution at a specific time level $t > 0$, we need the initial vectors $\mathbf{c^0}$ and $\mathbf{v^0}$.

3.4.1 Initial Vector $\mathbf{c^0}$

Initial vector c^0 can be computed using initial condition (3.2) as the following expressions

$$\begin{aligned} U(x_0, 0) &= \psi_1(0), \text{ for } j = 0, \\ U(x_j, 0) &= \phi_1(x_j), \text{ for } j = 1, \ldots, N-1, \\ U(x_N, 0) &= \psi_2(0), \text{ for } j = N. \end{aligned}$$

Above system represents the following $(N+1) \times (N+1)$ tridiagonal system of equations

$$A\mathbf{c^0} = B \tag{3.15}$$

where

$$A = \begin{bmatrix} 6 & 0 & \ldots & \ldots & 0 \\ 1 & 4 & 1 & \ldots & 0 \\ & & \ldots & \ldots & \ldots \\ & & \ldots & \ldots & \ldots \\ & & 1 & 4 & 1 \\ & & & 0 & 6 \end{bmatrix}, \ \mathbf{c^0} = \begin{bmatrix} c_0^0 \\ c_1^0 \\ \ldots \\ \ldots \\ c_{N-1}^0 \\ c_N^0 \end{bmatrix}, \ B = \begin{bmatrix} \psi_1(0) \\ \phi_1(x_1) \\ \ldots \\ \ldots \\ \phi_1(x_{N-1}) \\ \psi_2(0) \end{bmatrix}$$

Using Thomas-algorithm the solution of (3.15) can be easily found.

3.4.2 Initial Vector $\mathbf{v^0}$

Initial vector v^0 is computed using the following initial condition

$$\begin{aligned} U_t(x, 0) &= \phi_2(x), \\ v(x_j, 0) &= \phi_2(x_j) \quad j = 0, 1, \ldots, N-1, N. \end{aligned} \tag{3.16}$$

3.5 Numerical Experiments

To determine the efficacy of the method, it is tested on the following four test problems and accuracy of the method is measured using the maximum absolute error, L_2 error and root mean square error (RMS) between the numerical and exact solution.

Example 3.1

We consider the equation (3.1) in the computational domain $x \in (-a, a)$ with the following initial conditions

$$u(x, 0) = 0, \qquad u_t(x, 0) = 4\,\mathrm{sech}(x). \tag{3.17}$$

The exact solution is given in [131, 162, 178] as

$$u(x, t) = 4 \arctan(t\,\mathrm{sech}(x)). \tag{3.18}$$

The boundary conditions (3.3) are obtained from the exact solution.

Case I:

First we solve this problem in the computational domain $(-1, 1)$ with $\Delta t = .0001$ and for space step sizes $h = .02, .04$, respectively. In Table-3.1, L_2 and L_∞ errors are reported and compared with those obtained by Dehghan and Shokri [53]. We observe that our results are compatible with the results given in [53], for $h = .04$. For $h = .02$, in terms of L_2 error our method produces better results than those in [53]. Table-3.2 reports the absolute errors at $t = .01, .1, 1$ with $h = .02$. A graph comparing the exact and numerical solutions at $t = 1$ with $h = .02$, $\Delta t = .0001$ is depicted in Figure-3.1.

Case II:

In the domain $(-2, 2)$, the solutions of Example 3.1 are obtained with $\Delta t = .01$ and $h = .01$. In Table-3.3, L_∞ and RMS errors are given at different time levels and compared with those obtained by Li-Min and Zong-Min [131]. We observe that our results in terms of L_∞ errors are in good agreement with the results of [131]. In terms of RMS errors our results are better than those given in [131]. Figure-3.2 depicts the comparison of exact and numerical solutions at $t = 1$.

Example 3.2

We consider equation (3.1) in the computational domain $x \in (-3, 3)$ with the following initial conditions

$$u(x,0) = 4\arctan(\exp(\gamma x)), \qquad u_t(x,0) = \frac{-4c\gamma\exp(\gamma x)}{1+\exp(2\gamma x)}, \tag{3.19}$$

and the exact solution is given in [162, 178] as

$$u(x,t) = 4\arctan(\exp(\gamma(x-ct))), \tag{3.20}$$

where c is the velocity of solitary wave and $\gamma = 1/\sqrt{1-c^2}$.
The boundary conditions (3.3) can be obtained from the exact solution.

In Table-3.4, L_2 and L_∞ errors are reported for $h = .04, .02$ with $c = 0.5$ and $\Delta t = .0001$. These errors are compared with those given in Dehghan and Shokri [53]. From table it is clear that our scheme and that of [53], have approximately similar L_2 errors, for $h = .04$. For $h = .02$ our results are better than the results of [53], in term of L_2 errors and approximately similar in terms of L_∞ errors. The absolute errors are shown in Table-3.5, at $t = .01$, .1, 1. L_2, L_∞ errors and order of convergence are given in Table-3.6 at $t = 1$ with $\Delta t = .0001$. Figure-3.3 shows the graph between exact and numerical solutions at $t = 1$.

Example 3.3

We consider nonlinear sine-Gordon equation (3.1) in the computational domain $x \in (-10, 10)$ with the following initial conditions

$$u(x,0) = 0, \qquad u_t(x,0) = 4\bar{\gamma}\,\mathrm{sech}(\bar{\gamma}x). \tag{3.21}$$

The exact solution is given in [23, 200] as

$$u(x,t) = 4\arctan(c^{-1}\sin(\bar{\gamma}ct)\,\mathrm{sech}(\bar{\gamma}x)), \tag{3.22}$$

where c is the velocity of solitary wave and $\bar{\gamma} = 1/\sqrt{1+c^2}$.
The boundary conditions (3.3) can be obtained from the exact solution.

In our numerical computation we take $c = 0.5, h = .01, \Delta t = .001$. Computed errors are presented in Table-3.7 and compared with the results given in Uddin et al. [200] and

Bratsos [23]. It is clear from the table that our scheme produces much better results than those in [23, 200]. Table-3.8 reports the L_2 and L_∞ error norms at $t = 20$ for different values of h with $\Delta t = .001$. Order of convergence is also reported in Table-3.8. Figure-3.4 depicts the space-time graph of approximate solutions for $t \leq 20$ with $h = .01$ and $\Delta t = .001$.

Example 3.4

We consider equation (3.1) in the computational domain $x \in (-20, 20)$ with the initial conditions

$$u(x, 0) = 4 \arctan(c \sinh(\gamma x)), \qquad u_t(x, 0) = 0. \tag{3.23}$$

The exact solution is given in [23, 200] as

$$u(x, t) = 4 \arctan(c \sinh(\gamma x) \operatorname{sech}(\gamma c t)), \tag{3.24}$$

where c is the velocity of solitary wave and $\gamma = 1/\sqrt{1 - c^2}$. The boundary conditions (3.3) can be obtained from the exact solution.

This example is solved for $c = 0.5$, $h = .01$ and $\Delta t = .001$. Results are reported in Table-3.9 and compared with those given in Uddin et al. [200] and Bratsos [23] in term of L_∞ errors and found much better. In Table-3.10, L_2 and L_∞ error norms are reported at $t = 20$ for different values of h with $\Delta t = .001$. Order of convergence of the method is also computed and found to be two. Figure-3.5 represents the space-time graph of approximate solutions for $t \leq 20$ with $h = .01$ and $\Delta t = .001$.

3.6 Conclusions

The following observations have been made based on present study:

1. A collocation method with modified cubic B-spline basis functions has been proposed to solve nonlinear sine-Gordon equation with Dirichlet boundary conditions.

2. Approximate solutions have been computed without using any transformation and quasi-linearization approach.

3. It has been observed that the proposed method produces better results in comparison to those available in literature.

4. From the order of convergence reported in Tables-3.6, 3.8 and 3.10, the scheme is shown to have a second order of convergence.

Table 3.1: L_2 and L_∞ errors of Example 3.1 in the domain $[-1, 1]$ with $\Delta t = .0001$.

t	Present Scheme				Dehghan and Shokri [53]	
	$h = .04$		$h = .02$		$h = .04$	
	L_2	L_∞	L_2	L_∞	L_2	L_∞
.25	1.18E-5	2.32E-5	3.71E-6	8.20E-6	3.91E-5	5.89E-6
.50	4.19E-5	4.11E-5	1.34E-5	1.62E-5	1.30E-4	2.01E-5
.75	7.78E-5	1.02E-4	2.40E-5	2.54E-5	2.35E-4	3.63E-5
1	1.30E-4	1.64E-4	3.00E-5	4.14E-5	3.27E-4	5.07E-5

Table 3.2: Absolute errors of Example 3.1 with $h = .02$, $\Delta t = .0001$.

x	$t = .01$	$t = .1$	$t = 1$
-.80	4.19E-11	4.24E-8	1.11E-5
-.60	1.72E-11	1.94E-8	6.17E-7
-.40	3.32E-11	3.01E-8	1.47E-5
0	1.15E-10	1.09E-7	4.13E-5
.40	3.32E-11	3.01E-8	1.47E-5
.60	1.72E-11	1.94E-8	6.17E-7
.80	4.19E-11	4.24E-8	1.11E-5

Table 3.3: Errors of Example 3.1 in the domain $[-2, 2]$ with $h = .01$, $\Delta t = .01$.

t	Present Scheme		Li-Min and Zong-Min [131]	
	L_∞	RMS	L_∞	RMS
.1	7.20E-6	6.53E-8	1.54E-6	7.43E-6
.2	2.26E-5	2.69E-7	9.25E-5	1.76E-5
.3	4.54E-5	6.36E-7	9.02E-5	3.60E-5
.4	7.52E-5	1.19E-6	1.62E-4	1.62E-4
.5	1.12E-4	1.96E-6	2.58E-4	1.10E-4
.6	1.55E-4	2.96E-6	3.73E-4	1.65E-4
.7	2.04E-4	4.21E-6	4.98E-4	2.29E-4
.8	2.59E-4	5.72E-6	6.24E-4	2.98E-4
.9	3.19E-4	7.50E-6	7.44E-4	3.69E-4
1	3.84E-4	9.56E-6	8.49E-4	4.37E-4

Table 3.4: L_2 and L_∞ errors of Example 3.2 with $c = 0.5$, $\Delta t = .0001$.

t	Present Scheme				Dehghan and Shokri [53]	
	$h = .04$		$h = .02$		$h = .04$	
	L_2	L_∞	L_2	L_∞	L_2	L_∞
.25	3.66E-5	4.90E-5	9.26E-6	1.21E-6	1.76E-5	4.95E-6
.50	9.00E-5	7.55E-5	2.24E-5	1.89E-5	4.31E-5	8.42E-6
.75	1.60E-4	1.43E-4	3.98E-5	3.57E-5	8.25E-5	1.65E-5
1	2.27E-4	2.10E-4	5.66E-5	5.25E-5	1.27E-4	2.51E-5

Table 3.5: Absolute errors of Example 3.2 with $h = .02$, $\Delta t = .0001$.

x	$t = .01$	$t = .1$	$t = 1$
-2.5	6.05E-10	5.96E-8	5.28E-6
-2	8.76E-10	8.69E-8	1.21E-6
-1.5	5.64E-10	5.89E-8	9.16E-8
-1	2.68E-9	2.53E-7	2.02E-5
0	5.80E-11	5.64E-8	2.51E-5
1	2.72E-9	2.92E-7	4.82E-5
1.5	5.57E-10	5.08E-8	1.27E-5
2	8.78E-10	8.81E-8	2.21E-6
2.5	6.07E-10	6.15E-8	3.41E-6

Table 3.6: Errors and order of conv. of Example 3.2 at $t = 1$ with $\Delta t = .0001$.

h	L_2	Order of Conv.	L_∞	Order of Conv.
.08	9.08E-4		8.40E-4	
.04	2.27E-4	2	2.10E-4	2
.02	5.66E-5	2.003	5.25E-5	2
.01	1.40E-5	2.015	1.31E-5	2.002
.005	3.65E-6	1.939	3.28E-6	1.998

Table 3.7: L_2 and L_∞ errors of Example 3.3 with $c = 0.5$, $h = .01$, $\Delta t = .001$.

t	Present Scheme		Bratsos [23]	Uddin et al. [200]
	L_2	L_∞	L_∞	L_∞
1	7.448E-6	7.029E-6	.98816E-3	1.474E-3
10	3.999E-5	2.226E-5	.16291E-2	9.215E-3
20	6.467E-4	3.567E-4	.10379E-2	3.038E-1

Table 3.8: Errors and order of conv. of Example 3.3 at $t = 20$ with $\Delta t = .001$.

h	L_2	Order of Conv.	L_∞	Order of Conv.
.08	4.171E-2		2.302E-2	
.04	1.038E-2	2.006	5.729E-3	2.006
.02	2.593E-3	2.001	1.430E-3	2.002
.01	6.467E-4	2.003	3.567E-4	2.003

Table 3.9: L_2 and L_∞ errors of Example 3.4 with $c = 0.5$, $h = .01$, $\Delta t = .001$.

t	Present Scheme		Bratsos [23]	Uddin et al. [200]
	$h = .01$		$h = .01$	$N = 201$
	L_2	L_∞	L_∞	L_∞
2	2.564E-5	1.818E-5	.12760E-3	1.568E-3
10	8.850E-5	5.228E-5	.19115E-3	3.151E-3
20	1.713E-4	9.438E-5	.25189E-3	1.828E-2

Table 3.10: Errors and order of conv. of Example 3.4 at $t = 20$ with $\Delta t = .001$.

h	L_2	Order of Conv.	L_∞	Order of Conv.
.08	1.094E-2		6.025E-3	
.04	2.740E-3	1.997	1.509E-3	1.997
.02	6.851E-4	1.999	3.775E-4	1.999
.01	1.713E-4	1.999	9.438E-5	1.999

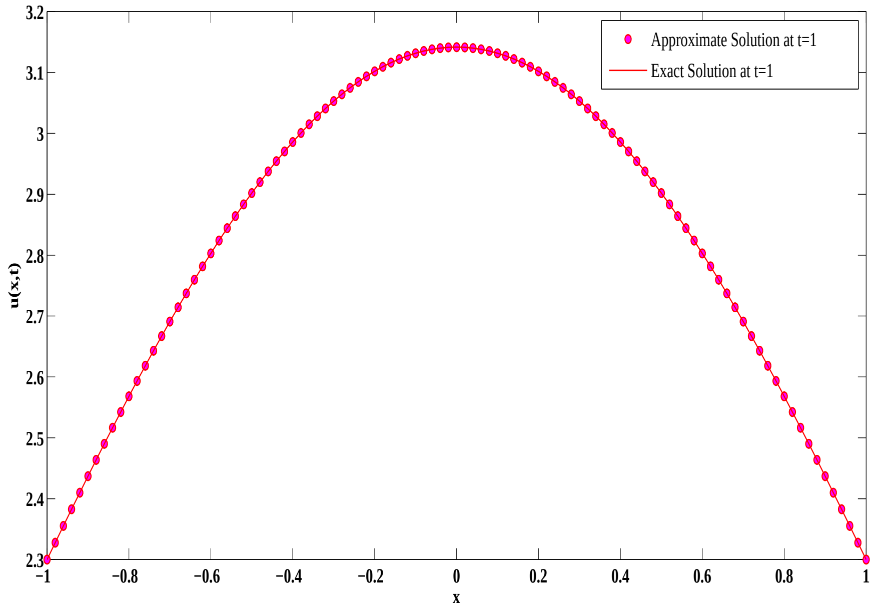


Figure 3.1: Solutions of Example 3.1 at $t = 1$ with $h = .02$, $\Delta t = .0001$.

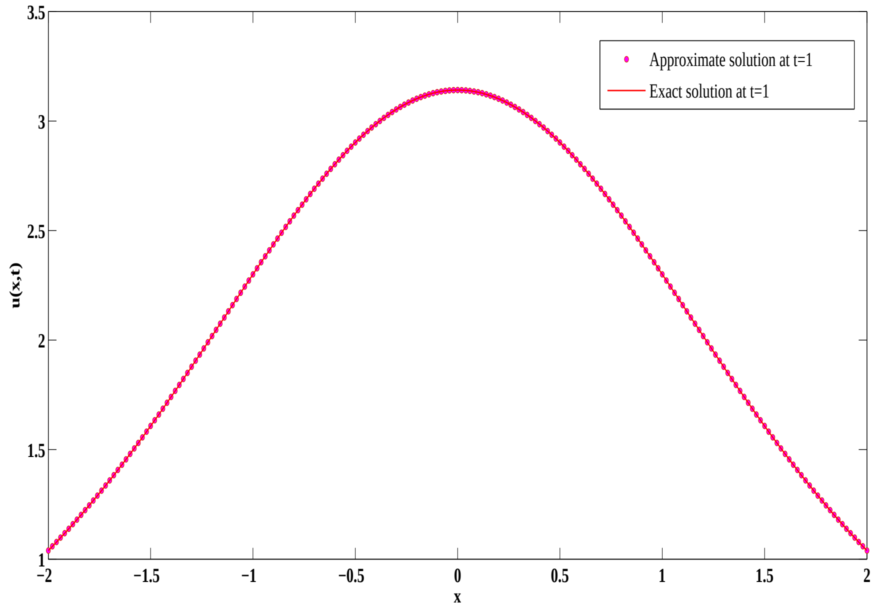


Figure 3.2: Solutions of Example 3.1 at $t = 1$ with $h = .01$, $\Delta t = .01$.

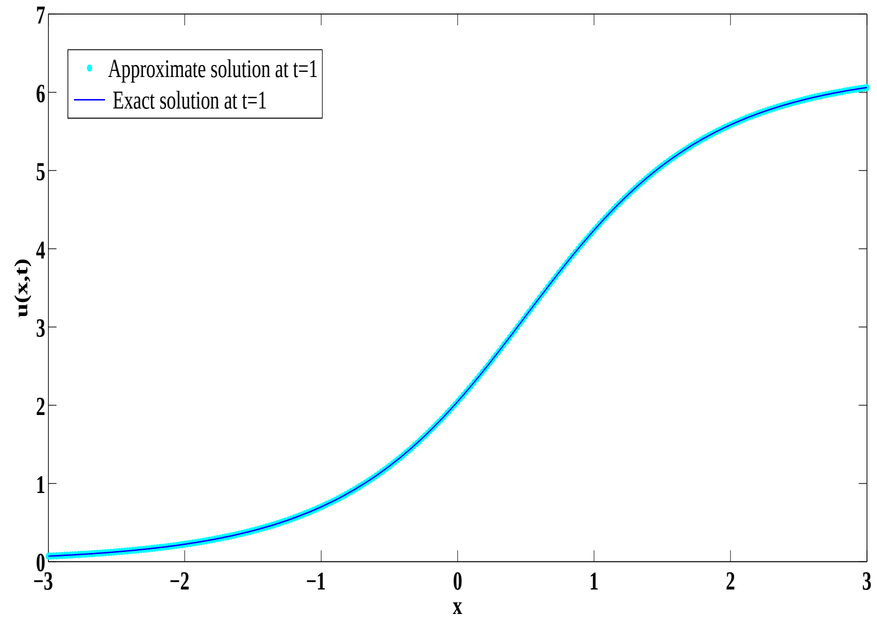


Figure 3.3: Solutions of Example 3.2 at $t = 1$ with $h = .02$, $\Delta t = .0001$.

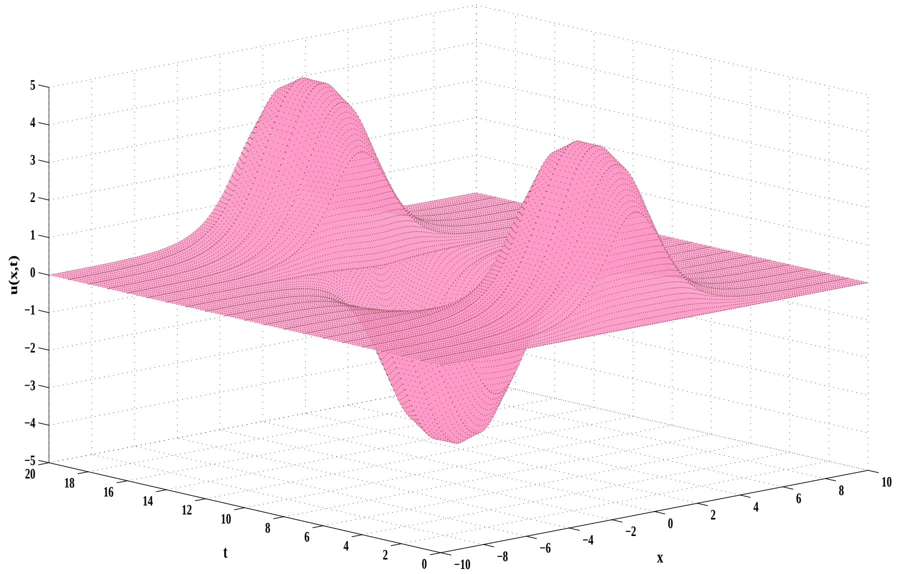


Figure 3.4: Approximate solutions of Example 3.3 up to $t = 20$.

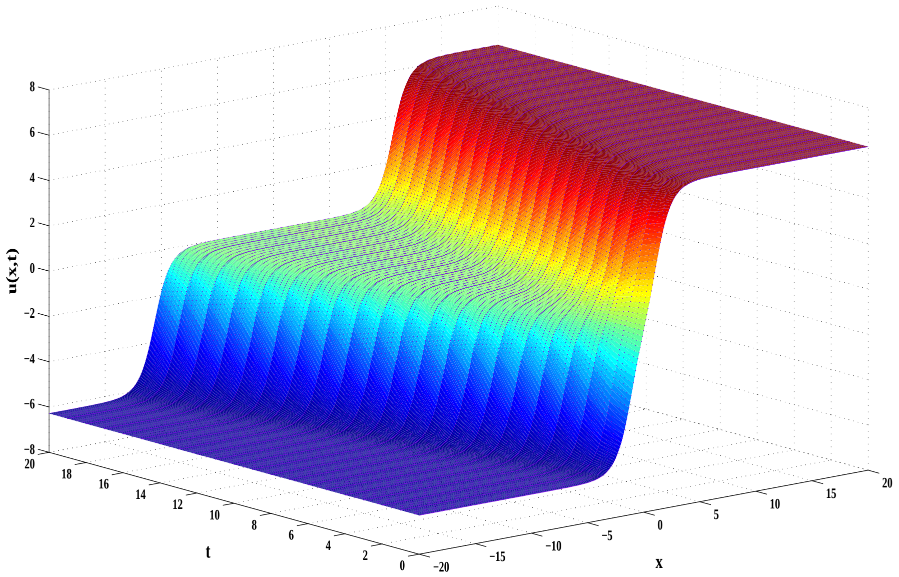


Figure 3.5: Approximate solutions of Example 3.4 up to $t = 20$.

Chapter 4

Numerical Solution of One Dimensional Linear Telegraph Equation

4.1 Introduction

The hyperbolic partial differential equations have a significant role in formulating fundamental equations in atomic physics [128] and are also very useful in understanding various phenomena in applied sciences like engineering industry aerospace as well as in chemistry and biology too. The one dimensional telegraph equation arises in the study of propagation of electric signal in a cable of transmission line and wave phenomena. Interaction between convection and diffusion or reciprocal action of reaction and diffusion describe a number of nonlinear phenomena in physical and biological process [161]. In fact telegraph equation is more suitable than ordinary diffusion equation in modeling reaction-diffusion for such branches of sciences.

The present chapter deals with the numerical solution of following second-order one dimensional telegraph equation with constant coefficients

$$u_{tt}(x,t) + 2\alpha u_t(x,t) + \beta^2 u(x,t) = u_{xx}(x,t) + f(x,t),\ \ x \in (a,b),\ \ t > 0, \tag{4.1}$$

with initial conditions

$$u(x,0) = f_1(x),\ \ u_t(x,0) = f_2(x),\ \ x \in [a,b], \tag{4.2}$$

and the following Dirichlet or Neumann boundary conditions

$$u(a,t) = g_1(t),\ u(b,t) = g_2(t),\ \ t \geq 0, \tag{4.3}$$

$$u_x(a,t) = w_1(t),\ u_x(b,t) = w_2(t),\ \ t \geq 0, \tag{4.4}$$

where α and β are known real constants. For $\alpha > 0$, $\beta = 0$, equation (4.1) represents a damped wave equation and for $\alpha > \beta > 0$, it is called as telegraph equation.

In the literature several numerical schemes have been developed for solving second order linear hyperbolic equation. Mohanty and Jain [155] have proposed an unconditionally stable ADI scheme for two space dimensional hyperbolic equation. Mohanty et al. [156] have presented an unconditionally stable ADI scheme for three dimensional hyperbolic equation. Dehghan and Mohebbi [49] have presented a higher order implicit collocation method for solving two dimensional linear hyperbolic equation. Numerical solution of hyperbolic equation with variable coefficients in one and two space dimensions has been presented by Mohanty [151] and Dehghan and Shokri [55], respectively. One dimensional nonlinear hyperbolic equation with variable coefficients has been tackled by Mohanty et al. [158].

Numerical solution of one dimensional linear hyperbolic equation with Dirichlet boundary conditions has been proposed by many authors. Liu and Liu [134] have proposed an unconditionally stable three level difference scheme, which was based on quartic spline interpolation in space and finite difference discretization in time. Mohanty [150] has presented an unconditionally stable three level implicit difference scheme. Gao and Chi [72] have developed an unconditionally stable difference scheme to solve the equation (4.1). Dehghan and Lakestani [46] have used the elements of Chebyshev cardinal functions, whereas Saadatmandi and Dehghan [186] have used the Chebyshev tau method for expanding the approximate solution of the equation (4.1). Mohebbi and Dehghan [161] have reported a higher order compact finite difference approximation of fourth order for discretizing spatial derivatives and used collocation method for time direction. Dehghan and Shokri [54] have proposed a scheme using collocation points and approximate the solution using thin plates spline radial basis functions. Lakestani and Saray [128] have used the elements of interpolating scaling functions for expanding the approximate solution of equation (4.1). Some of other numerical schemes, which have been developed to solve equation (4.1) are radial basis functions based collocation method [69], quartic B-spline collocation method [64] and differential quadrature method [90]. Thus much work has been done to solve the equation (4.1) with Dirichlet boundary conditions. However, very little has been reported to solve equation (4.1) with Neumann boundary conditions. Dehghan and Ghesmati [44] have reported a dual reciprocity boundary integral equation (DRBIE) method, in which three different types of radial basis functions have been used. Recently, Liu and Liu [135] have developed an unconditionally stable compact difference schemes for solving equation (4.1) with Neumann boundary conditions.

In this chapter, a collocation method is proposed to solve equations (4.1-4.4). For Dirichlet boundary conditions, approximate solution is considered as collocation of modified cubic B-splines. In case of Neumann boundary conditions, approximate solution is assumed as collocation of cubic B-spline functions. The stability of presented method is discussed using matrix stability analysis. It is shown that the proposed method is unconditionally stable. Four numerical examples are solved in this chapter. The obtained solutions are tabulated and presented graphically for the considered examples and results are also compared with those already available in the literature.

4.2 Description of Method

The solution domain $a \leq x \leq b$ is partitioned into a mesh of uniform length $h = x_{j+1} - x_j$, where $j = 0, 1, 2, ..., N-1$, such that $a = x_0 < x_1 < ..... < x_{N-1} < x_N = b$.

In the cubic B-spline collocation method the approximate solution $U(x,t)$ can be written as the linear combination of cubic B-spline basis functions for the approximation space under consideration as

$$U(x,t) = \sum_{j=-1}^{N+1} c_j(t) B_j(x), \tag{4.5}$$

where $c_j(t)$ are the time dependent quantities to be determined from boundary conditions and collocation from the differential equation.

The set of functions $\{B_{-1}, B_0, B_1, \ldots, B_{N-1}, B_N, B_{N+1}\}$ forms a basis for the function define over the region $a \leq x \leq b$ with the obvious adjustment of the boundary base functions to avoid undefined knots. The values of $B_j(x)$ and its derivatives are tabulated in Table-1.2. Approximate values of $U(x,t)$ and its two derivatives in terms of the time parameters c_j are given in equation (1.13).

4.3 Implementation of Method

First the telegraph equation (4.1) is decomposed into a system of equations using the transformation $u_t(x,t) = v(x,t)$ as

$$\begin{aligned} u_t &= v, \\ v_t &= -2\alpha v - \beta^2 u + u_{xx} + f(x,t), \end{aligned} \tag{4.6}$$

where the boundary conditions for v are obtained from corresponding values of u_t at the boundary points.

4.3.1 Treatment of Dirichlet Boundary Conditions

In collocation method with cubic B-splines, there is a need to change the basis functions into a new set of functions such that they matches in the number with selected points in given domain. The Dirichlet boundary conditions can be easily tackled using these basis functions and produce a diagonally dominant system of equations.

We assume our approximate solution as the linear combination of modified cubic B-spline basis functions (1.6) as

$$U(x,t) = \sum_{j=0}^{N} c_j(t)\tilde{B}_j(x). \tag{4.7}$$

Using (4.7), approximate value of $U_t(x,t)$ can be written as

$$U_t(x,t) = \sum_{j=0}^{N} \dot{c}_j(t)\tilde{B}_j(x), \tag{4.8}$$

where $\dot{c}_j(t)$ is the derivative of $c_j(t)$.

Using modified basis functions (1.6) and Table–1.2 in (4.8), we have

$$\begin{aligned} U_t(x_0,t) &= 6\dot{c}_0, \text{ for } j = 0, \\ U_t(x_j,t) &= \dot{c}_{j-1} + 4\dot{c}_j + \dot{c}_{j+1}, \text{ for } j = 1,2,\ldots,N-1, \\ U_t(x_N,t) &= 6\dot{c}_N, \text{ for } j = N. \end{aligned} \tag{4.9}$$

Using (4.8) in coupled system (4.6) and imposing boundary conditions (4.3) at the boundary points, we have

$$\begin{aligned} U_t(x_0,t) &= \dot{g}_1(t), \text{ for } i=0,\\ U_t(x_i,t) &= v_i, \text{ for } i=1,2,\ldots,N-1,\\ U_t(x_N,t) &= \dot{g}_2(t), \text{ for } i=N. \end{aligned} \tag{4.10}$$

$$\begin{aligned} v_t(x_0,t) &= \ddot{g}_1(t), \text{ for } i=0,\\ v_t(x_i,t) &= -2\alpha v_i - \beta^2\sum_{j=0}^{N} c_j\tilde{B}_j(x_i) + \sum_{j=0}^{N} c_j\tilde{B}_j''(x_i) + f(x_i,t), \text{ for } i=1,2,\ldots,N-1,\\ v_t(x_N,t) &= \ddot{g}_2(t), \text{ for } i=N. \end{aligned} \tag{4.11}$$

Now using (4.9) in (4.10) and using modified basis functions (1.6) and Table-1.2 in (4.11), we have

$$\begin{aligned} 6\dot{c}_0 &= \dot{g}_1(t), \text{ for } j=0,\\ \dot{c}_{j-1}+4\dot{c}_j+\dot{c}_{j+1} &= v_j, \text{ for } j=1,2,\ldots,N-1,\\ 6\dot{c}_N &= \dot{g}_2(t), \text{ for } j=N. \end{aligned} \tag{4.12}$$

$$\begin{aligned} \dot{v}_0 &= \ddot{g}_1(t), \text{ for } j=0,\\ \dot{v}_j &= -2\alpha v_j - \beta^2(c_{j-1}+4c_j+c_{j+1}) + \frac{6}{h^2}(c_{j-1}-2c_j+c_{j+1}) + f(x_j,t), \text{ for } j=1,2,\ldots,N-1,\\ \dot{v}_N &= \ddot{g}_2(t), \text{ for } j=N. \end{aligned} \tag{4.13}$$

Systems (4.12) and (4.13) can be written in compact form as

$$\begin{aligned} A\dot{c} &= F,\\ \dot{V} &= G. \end{aligned}$$

i.e.

$$\begin{bmatrix} 6 & 0 & \ldots & \ldots & 0\\ 1 & 4 & 1 & \ldots & 0\\ & & \ldots & \ldots & \ldots\\ & & \ldots & \ldots & \ldots\\ & & 1 & 4 & 1\\ & & & 0 & 6 \end{bmatrix} \begin{bmatrix} \dot{c}_0\\ \dot{c}_1\\ \ldots\\ \ldots\\ \dot{c}_{N-1}\\ \dot{c}_N \end{bmatrix} = \begin{bmatrix} F_0\\ F_1\\ \ldots\\ \ldots\\ F_{N-1}\\ F_N \end{bmatrix} \tag{4.14}$$

$$\begin{bmatrix} \dot{v}_0 \\ \dot{v}_1 \\ \cdots \\ \cdots \\ \dot{v}_{N-1} \\ \dot{v}_N \end{bmatrix} = \begin{bmatrix} G_0 \\ G_1 \\ \cdots \\ \cdots \\ G_{N-1} \\ G_N \end{bmatrix} \tag{4.15}$$

where $F_0 = \dot{g}_1(t),\ \ G_0 = \ddot{g}_1(t)$,

$F_j = v_j,\ \ G_j = -2\alpha v_j - \beta^2(c_{j-1}+4c_j+c_{j+1}) + \frac{6}{h^2}(c_{j-1}-2c_j+c_{j+1}) + f(x_j,t), j = 1,\ldots,N-1$,

$F_N = \dot{g}_2(t),\ \ G_N = \ddot{g}_2(t)$.

4.3.2 Treatment of Neumann Boundary Conditions

For solving couple of equations (4.6) with Neumann boundary conditions, we assume our approximate solution as (4.5) i.e linear combination of cubic B-spline basis functions. Using (4.5), Neumann boundary conditions (4.4) are approximated as

$$\begin{aligned} U_x(x_0,t) &= \sum_{j=-1}^{1} c_j B_j^{'}(x_0) = w_1(t), \\ U_x(x_N,t) &= \sum_{j=N-1}^{N+1} c_j B_j^{'}(x_N) = w_2(t), \end{aligned} \tag{4.16}$$

which gives

$$\begin{aligned} \dot{c}_1 - \dot{c}_{-1} &= \frac{h}{3}\dot{w}_1(t), \\ \dot{c}_{N+1} - \dot{c}_{N-1} &= \frac{h}{3}\dot{w}_2(t). \end{aligned} \tag{4.17}$$

Using approximate solution (4.5) and its time derivative, in system (4.6), we have

$$\sum_{j=-1}^{N+1} \dot{c}_j B_j(x_i) = v_i, \text{ for } i = 0,1,\ldots,N. \tag{4.18}$$

$$\dot{v}_i = -2\alpha v_i - \beta^2 \sum_{j=-1}^{N+1} c_j B_j(x_i) + \sum_{j=-1}^{N+1} c_j B_j^{''}(x_i) + f(x_i,t), \text{ for } i = 0,1,\ldots,N. \tag{4.19}$$

Now using Table-1.2 in above, we get

$$\dot{c}_{j-1} + 4\dot{c}_j + \dot{c}_{j+1} = v_j, \quad j = 0, 1, \ldots, N. \tag{4.20}$$

$$\dot{v}_j = -2\alpha v_j - \beta^2(c_{j-1} + 4c_j + c_{j+1}) + \frac{6}{h^2}(c_{j-1} - 2c_j + c_{j+1}) + f(x_j, t), \; j = 0, 1, \ldots, N. \tag{4.21}$$

Eliminating $c_{-1}, c_{N+1}, \dot{c}_{-1}, \dot{c}_{N+1}$, we get the following systems

$$M\dot{c} = X,$$
$$\dot{V} = Y.$$

i.e.

$$\begin{bmatrix} 4 & 2 & \ldots & \ldots & \\ 1 & 4 & 1 & \ldots & \\ & & \ldots & \ldots & \ldots \\ & & \ldots & \ldots & \ldots \\ & & 1 & 4 & 1 \\ & & & 2 & 4 \end{bmatrix} \begin{bmatrix} \dot{c}_0 \\ \dot{c}_1 \\ \ldots \\ \ldots \\ \dot{c}_{N-1} \\ \dot{c}_N \end{bmatrix} = \begin{bmatrix} \mu_0 \\ \mu_1 \\ \ldots \\ \ldots \\ \mu_{N-1} \\ \mu_N \end{bmatrix}, \tag{4.22}$$

$$\begin{bmatrix} \dot{v}_0 \\ \dot{v}_1 \\ \ldots \\ \ldots \\ \dot{v}_{N-1} \\ \dot{v}_N \end{bmatrix} = \begin{bmatrix} \chi_0 \\ \chi_1 \\ \ldots \\ \ldots \\ \chi_{N-1} \\ \chi_N \end{bmatrix}, \tag{4.23}$$

where $\mu_0 = v_0 + \frac{h}{3}\dot{w}_1(t)$,

$\mu_j = v_j, \; j = 1, 2, \ldots, N-1$,

$\mu_N = v_N - \frac{h}{3}\dot{w}_2(t)$,

$\chi_0 = -2\alpha v_0 - \beta^2(4c_0 + 2c_1 - \frac{h}{3}w_1(t)) + \frac{6}{h^2}(-2c_0 + 2c_1 - \frac{h}{3}w_1(t)) + f(x_0, t)$,

$\chi_j = -2\alpha v_j - \beta^2(c_{j-1} + 4c_j + c_{j+1}) + \frac{6}{h^2}(c_{j-1} - 2c_j + c_{j+1}) + f(x_j, t), j = 1, 2, \ldots, N-1$,

$\chi_N = -2\alpha v_N - \beta^2(2c_{N-1} + 4c_N + \frac{h}{3}w_2(t)) + \frac{6}{h^2}(-2c_N + 2c_{N-1} + \frac{h}{3}w_2(t)) + f(x_N, t)$.

To compute the approximate solution at any time level, we need the vector **c**. So for Dirichlet boundary conditions, first we solve the system (4.14) for vector $\dot{c}$ by using

Thomas algorithm. Then obtained system of ODEs with the system (4.15) i.e. total $(2N+2)$ first order ordinary differential equations have been solved using SSP-RK54 [197] scheme and consequently the approximate solution $U(x,t)$ is completely known. For Neumann conditions, systems (4.22-4.23) are solved using the same procedure and approximate solution can be computed.

4.3.3 Initial Vectors

The initial vectors $\mathbf{c}^0$ and $\mathbf{v}^0$ can be determined from the initial conditions (4.2) as:

4.3.3.1 Initial Vector $\mathbf{c}^0$ for Dirichlet Boundary Conditions

We consider

$$\begin{aligned} &U(x_0,0) = g_1(0),\ \ j=0,\\ &U(x_j,0) = f_1(x_j),\ \ j=1,2,\dots,N-1,\\ &U(x_N,0) = g_2(0),\ \ j=N. \end{aligned}$$

This gives following $(N+1)$ tridiagonal system of equations

$$A\mathbf{c}^0 = F^0 \tag{4.24}$$

where A is tridiagonal matrix defined in Section-4.3.1,
$F^0 = [g_1(0), f_1(x_1), f_1(x_2), \dots, f_1(x_{N-1}), g_2(0)]^T$.
This system is solved using Thomas algorithm to get initial vector $\mathbf{c}^0$=$[c_0^0, c_1^0, \dots, c_{N-1}^0, c_N^0]^T$.

4.3.3.2 Initial Vector $\mathbf{c}^0$ for Neumann Boundary Conditions

We consider

$$\begin{gathered} U_x(x_0,0) = w_1(0)\\ U(x_j,0) = f_1(x_j),\ \ j=0,1,\dots,N-1,N,\\ U_x(x_N,0) = w_2(0). \end{gathered}$$

which gives

$$M\mathbf{c}^0 = H^0 \tag{4.25}$$

where M is tridiagonal matrix defined in Section-4.3.2 and
$H^0 = [f_1(x_0) + \frac{h}{3}w_1(0), f_1(x_1), f_1(x_2), \dots, f_1(x_{N-1}), f_1(x_N) - \frac{h}{3}w_2(0)]^T$.

4.3.4 Initial Vector $\mathbf{v}^0$

Initial vector v^0 for both the above cases can be found using the following initial condition

$$U_t(x_j, 0) = f_2(x_j), \ \ j = 0, 1, \dots, N-1, N. \tag{4.26}$$

4.4 Stability Analysis

The stability of obtained system of ODEs is related to the stability of numerical scheme for solving it. If the system is unstable then stable numerical scheme for temporal discretization may not produce converged solution. The stability of these systems depends on the eigenvalues of coefficient matrix, since its exact solution can be directly found using its eigenvalues.

We consider

$$N\frac{dC}{dt} = BC + F(t), \tag{4.27}$$

where $C = [\ c_1, c_2, \dots, c_{N-1}, v_1, v_2, \dots, v_{N-1}\ ]^T$ is the unknown vector, $F(t)$ is a column vector of order $2(N-1)$.

Above system can be written as

$$\begin{bmatrix} T_1 & O \\ O & I \end{bmatrix}\begin{bmatrix} \dot{c} \\ \dot{v} \end{bmatrix} = \begin{bmatrix} O & I \\ T_2 & -2\alpha I \end{bmatrix}\begin{bmatrix} c \\ v \end{bmatrix} + F(t) \tag{4.28}$$

where T_1 and T_2 are symmetric tridiagonal matrices of order $N-1$, given by

$$T_1 = \begin{bmatrix} 4 & 1 & \dots & \dots & & \\ 1 & 4 & 1 & \dots & & \\ & & \dots & \dots & \dots & \\ & & \dots & \dots & \dots & \\ & & 1 & 4 & 1 & \\ & & & 1 & 4 & \end{bmatrix}, \ T_2 = \begin{bmatrix} -4\beta^2 - \frac{12}{h^2} & -\beta^2 + \frac{6}{h^2} & \dots & \dots & \\ -\beta^2 + \frac{6}{h^2} & -4\beta^2 - \frac{12}{h^2} & -\beta^2 + \frac{6}{h^2} & \dots & \\ & & \dots & \dots & \dots \\ & & \dots & \dots & \dots \\ & & -\beta^2 + \frac{6}{h^2} & -4\beta^2 - \frac{12}{h^2} & -\beta^2 + \frac{6}{h^2} \\ 0 & & & -\beta^2 + \frac{6}{h^2} & -4\beta^2 - \frac{12}{h^2} \end{bmatrix}$$

I and O are identity and null matrices of order $N-1$, respectively.

Stability of system (4.27) depends on the eigenvalues of the coefficient matrix $N^{-1}B$. If all the eigenvalues are having negative real part then the system will be stable.

Let λ is an eigenvalue of $N^{-1}B$ and x_1 and x_2 be two components of eigenvector (each of order $(N-1)$) corresponding to eigenvalue λ.

We have

$$\begin{bmatrix} O & T_1^{-1} \\ T_2 & -2\alpha I \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \lambda \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \tag{4.29}$$

from (4.29), we have

$$T_1^{-1} x_2 = \lambda x_1,$$

$$T_2 x_1 - 2\alpha x_2 = \lambda x_2,$$

which gives

$$T_2 T1^{-1} x_2 = (\lambda^2 + 2\alpha\lambda) x_2 \tag{4.30}$$

$\Rightarrow \lambda(\lambda + 2\alpha)$ is an eigenvalue of $T_2 T_1^{-1}$.

Since T_1 and T_2 are tridiagonal matrices, so their eigenvalues can be obtained as

$$\lambda_s(T_1) = 4 + 2\cos\frac{\pi s}{N}, \quad s = 1, 2, \ldots, N-1. \tag{4.31}$$

$$\lambda_s(T_2) = -2\beta^2 - 4\beta^2 \cos^2\frac{\pi s}{2N} - \frac{24}{h^2}\sin^2\frac{\pi s}{2N}, \quad s = 1, 2, \ldots, N-1. \tag{4.32}$$

T_1 and T_2 are symmetric also, so eigenvalues of matrix $T_2 T_1^{-1}$ will be given by

$$\lambda_s(T_2 T_1^{-1}) = \frac{-2\beta^2 - 4\beta^2 \cos^2\frac{\pi s}{2N} - \frac{24}{h^2}\sin^2\frac{\pi s}{2N}}{4 + 2\cos\frac{\pi s}{N}}, \quad s = 1, 2, \ldots, N-1, \tag{4.33}$$

which are negative and real.

Let $\lambda = x + iy$, where x and y are real numbers.

Then from (4.30) and (4.33) we have, $(x + iy)(x + iy + 2\alpha)$ is real and negative. i.e.

$$\left.\begin{aligned} x(x + 2\alpha) - y^2 < 0, \\ y(x + \alpha) = 0. \end{aligned}\right\} \tag{4.34}$$

From above set of equations, we get the solution as:

(i) If $x + \alpha = 0$ and y is arbitrary real number.

$\Rightarrow x$ is negative real number, since α is real and positive.

(ii) If $y = 0$

$\Rightarrow x(x + 2\alpha) < 0,$

$\Rightarrow (x + \alpha)^2 < \alpha^2,$

$\Rightarrow -2\alpha < x < 0$, since α is real and positive.

Hence we conclude that the real part of eigenvalues of $N^{-1}B$ are always negative i.e system (4.27) is stable.

4.5 Numerical Experiments

Example 4.1

We consider telegraph equation (4.1) in the domain (a, b) with $f(x,t) = (2 - 2\alpha + \beta^2)e^{-t}\sin(x)$ and the following initial conditions

$$u(x,0) = \sin(x), \quad u_t(x,0) = -\sin(x). \tag{4.35}$$

The exact solution of this example is given in [44, 72] as

$$u(x,t) = e^{-t}\sin(x). \tag{4.36}$$

The boundary conditions can be obtained from exact solution.

For Dirichlet boundary conditions, we have solved this example in the computational domain $(0, \pi)$ with $\alpha = 2,\ \beta = \sqrt{2}$. L_2, L_∞ errors and CPU time are reported in Table-4.1 with $h = .02$ and $\Delta t = .0001$ and compared with those given in Dehghan and Shokri [54]. It may be noted that our results are better and CPU time is less in our computations. In Table-4.2, L_2 and L_∞ errors are presented at different levels of time with $h = .02$ and $\Delta t = .01$. A comparison of numerical and exact solutions with $h = .02$, $\Delta t = .01$ at $t = 1, 2, 3$ is presented in Figure-4.1. Space-time graph of approximate solutions up to $t = 2$ is shown in Figure-4.2.

For Neumann boundary conditions, this example has been solved in domain $(0, 2\pi)$ with $\alpha = 4$, $\beta = 2$. Errors are reported in Table-4.3 with $\Delta t = .001$ and $h = .05, .02$. Table-4.4 shows the comparison of RMS errors with those given in Dehghan and Ghesmati [44]. We note that our solutions are compatible with [44].

Example 4.2

In this example, we consider the telegraph equation (4.1) with $\alpha = 10$, $\beta = 5$, $f(x,t) = \alpha(1 + \tan^2(\frac{x+t}{2})) + \beta^2\tan(\frac{x+t}{2})$, in the domain $(0, 2)$.
The initial and boundary conditions are given as

$$u(x,0) = \tan\left(\frac{x}{2}\right),\ u_t(x,0) = \frac{1}{2}\left(1 + \tan^2\left(\frac{x}{2}\right)\right), \tag{4.37}$$

$$u(0,t) = \tan\left(\frac{t}{2}\right),\ u\left(2,t\right) = \tan\left(\frac{2+t}{2}\right). \tag{4.38}$$

The exact solution is given in [64, 90] as

$$u(x,t) = \tan\left(\frac{x+t}{2}\right). \tag{4.39}$$

In our numerical computation, we take $h = .02$ and $\Delta t = .001,.0001$, respectively. L_2 and L_∞ errors for different time levels are reported in Table-4.5. Table-4.6 shows the comparison of our results in terms of L_∞ errors with those of Dosti and Nazemi [64] at different time levels with $\Delta t = .001$, $h = .001$. The graphs of exact and approximate solutions for different times are presented in Figure-4.3 and space-time graph of numerical solutions up to $t = 1$ is shown in Figure-4.4.

Example 4.3

In this example we consider the telegraph equation (4.1) with $\alpha = \frac{1}{2}$, $\beta = 1$, $f(x,t) = (2 - 2t + t^2)(x - x^2)e^{-t} + 2t^2e^{-t}$, in the domain $0 < x < 1$ with the following initial and boundary conditions

$$u(x,0) = u_t(x,0) = 0, \tag{4.40}$$

$$u(0,t) = u(1,t) = 0. \tag{4.41}$$

The exact solution is given in [54, 186] as

$$u(x,t) = (x - x^2)t^2e^{-t}. \tag{4.42}$$

L_2, L_∞ errors and CPU time are reported in Table-4.7 with $\Delta t = .001, h = .01$. These errors are compared with those given in Dehghan and Shokri [54]. We can see that the solutions obtained by our method are good in comparison with [54] and also CPU time taken is less in our case. The graphs of exact and numerical solutions at $t = 1, 2, 3, 4, 5$ are depicted in Figure-4.5 and the space time graph of numerical solutions is plotted up to $t = 5$, which is shown in Figure-4.6.

Example 4.4

Consider the telegraph equation (4.1) with $\alpha = 6$, $\beta = 2$, $f(x,t) = -2\alpha\sin(t)\sin(x) + \beta^2\cos(t)\sin(x)$, in the domain $a < x < b$ with the following initial conditions

$$u(x,0) = \sin(x),\ u_t(x,0) = 0. \tag{4.43}$$

The exact solution is given in [44] as

$$u(x,t) = \cos(t)\sin(x). \tag{4.44}$$

The boundary conditions are taken from (4.44).

For Dirichlet boundary conditions we have solved this example in computational domain $(0,1)$. Table-4.8 reports the L_2, L_∞ and RMS errors with $\Delta t = .0001$, $h = .01$. In Table-4.9 errors are presented for different values of h with $\Delta t = .001$ and compared with those given in Dosti and Nazemi [64]. We found that our results are comparable to that of [64] in terms of L_∞ errors. Figure-4.7 depicts the comparison of numerical and exact solutions at different time levels with $h = .01$ and $\Delta t = .001$. Space-time graph of numerical solutions up to $t = 1$ is depicted in Figure-4.8.

For Neumann boundary conditions this example is solved in the domain $(0,4)$. The computed errors are reported in Table-4.10 with $h = .05, .02$ and $\Delta t = .001$. In Table-4.11, we show the comparison of RMS errors with those obtain by Dehghan and Ghesmati [44] by taking different space step sizes and observe them in good agreement with [44].

4.6 Conclusions

The following observations have been made based on present study:

1. The cubic B-spline collocation method has been developed to solve one dimensional hyperbolic telegraph equation.

2. The presented method is able to handle Dirichlet as well as Neumann boundary conditions.

3. It has been observed that the proposed method produces better results in comparison to those available in the literature and CPU time is quite less in our case.

4. The stability of the scheme is tested using matrix stability analysis and found to be unconditionally stable.

5. This scheme produces a solution in the form of spline function, which can be use to obtain the solution can be computed at any point in the domain.

Table 4.1: Errors and CPU time of Example 4.1 with $\Delta t = .0001$, $h = .02$.

t	Proposed method			Dehghan and Shokri [54]		
	L_2	L_∞	CPU time(s)	L_2	L_∞	CPU time(s))
.5	2.3328E-6	1.8612E-6	3.04	7.9491E-5	8.3721E-6	5
1	4.3667E-6	3.4839E-6	4.89	1.4554E-4	1.5680E-5	12
1.5	4.7817E-6	3.8251E-6	5.27	1.5895E-4	1.7412E-5	19
2	4.2706E-6	3.4073E-6	7.53	1.4185E-4	1.5813E-5	28

Table 4.2: Errors and CPU time of Example 4.1 with $h = .02$, $\Delta t = .01$.

t	L_2	L_∞	CPU time(s)
.5	2.3328E-6	1.8612E-6	0.51
1	4.3667E-6	3.4839E-6	0.59
1.5	4.7817E-6	3.8251E-6	0.63
2	4.2706E-6	3.4073E-6	0.68

Table 4.3: Errors of Example 4.1 with $\Delta t = .001$.

t	$h = .05$			$h = .02$		
	L_2	L_∞	RMS errors	L_2	L_∞	RMS errors
1	5.1145E-4	4.9491E-4	2.0328E-4	1.7865E-4	1.6756E-4	7.1176E-5
2	3.2011E-4	2.4541E-4	1.2722E-4	1.5356E-4	1.0723E-4	6.1178E-5
3	1.9952E-4	1.3436E-4	7.8599E-4	1.0689E-4	6.6156E-5	4.2587E-5

Table 4.4: Comparison of RMS errors of Example 4.1 at $t = 3$.

h	Proposed Method		Dehghan and Ghesmati [44]	
	RMS errors	CPU time(s)	RMS errors	CPU time(s)
.05	7.8599E-4	.70	3.010E-4	–
.02	4.2587E-5	1.7	7.128E-5	–
.01	3.7601E-5	3.1	4.320E-5	–

Table 4.5: Errors of Example 4.2 at different time levels with $h = .02$.

t	$h = .02, \Delta t = .0001$		$h = .02, \Delta t = .001$	
	L_2	L_∞	L_2	L_∞
.2	5.0305E-5	3.4797E-5	1.8782E-4	2.6332E-4
.4	9.5163E-5	5.3456E-5	4.8888E-4	6.9997E-4
.6	2.2041E-4	9.4725E-5	9.4864E-4	1.4860E-3
.8	7.8267E-4	1.8792E-4	1.8743E-3	3.4057E-3
1	7.9277E-3	5.8720E-4	5.0981E-3	1.1148E-2

Table 4.6: Errors of Example 4.2 with $h = .001, \Delta t = .001$ at different time levels.

t	Proposed method		Dosti and Nazemi [64]
	L_2	L_∞	L_∞
.2	2.1800E-4	3.6103E-4	2.774E-4
.4	5.6618E-4	1.0368E-4	7.0782E-4
.6	1.1535E-4	2.5996E-3	1.3848E-3
.8	2.6058E-3	7.6254E-3	3.0930E-3
1	1.0351E-2	4.6642E-2	1.3424E-2

Table 4.7: Errors of Example 4.3 with $h = .01$, $\Delta t = .001$.

t	Proposed method			Dehghan and Shokri [54]		
	L_2	L_∞	CPU time(s)	L_2	L_∞	CPU time(s)
1	4.5526E-5	5.9153E-5	.43	1.4386E-4	1.8479E-5	0
2	1.4307E-5	1.7864E-5	.77	8.0879E-5	1.0713E-5	0
3	6.4273E-6	1.4309E-5	1.19	1.2944E-4	1.8161E-5	1
4	8.9203E-6	1.3529E-5	1.29	1.1845E-4	1.6489E-5	1
5	3.0161E-6	5.2032E-6	1.46	7.5545E-5	1.0455E-5	2

Table 4.8: Errors of Example 4.4 with $h = .01$, $\Delta t = .0001$.

t	L_2	L_∞	RMS errors
.2	2.6903E-6	5.2412E-6	2.6775E-6
.4	5.6183E-6	8.6095E-6	5.5904E-6
.6	9.7545E-6	1.2529E-5	9.7061E-6
.8	1.3774E-5	2.0274E-5	1.3706E-5
1	1.7347E-5	2.7555E-5	1.7261E-5

Table 4.9: Errors of Example 4.4 at different time levels with $\Delta t = .001$.

t	Proposed method				Dosti and Nazemi [64]
	$h = .01, \Delta t = .001$		$h = .005, \Delta t = .001$		$h = .005, \Delta t = .001$
	L_2	L_∞	L_2	L_∞	L_∞
.2	3.6711E-5	7.9139E-5	3.4308E-5	6.8272E-5	2.4249E-5
.4	8.8989E-5	1.5967E-4	8.5756E-5	1.4935E-4	7.9315E-5
.6	1.3733E-4	2.3362E-4	1.3367E-4	2.2410E-4	1.2097E-4
.8	1.7913E-4	2.9820E-4	1.7532E-4	2.8978E-4	1.4883E-4
1	2.1302E-4	3.5085E-4	2.0929E-4	3.4386E-4	1.6462E-4

Table 4.10: Errors of Example 4.4 with $\Delta t = .001$.

t	$h = .05$			$h = .02$		
	L_2	L_∞	RMS errors	L_2	Ł$_\infty$	RMS errors
1	6.7752E-4	6.6070E-4	3.3667E-4	5.2299E-4	3.8981E-4	2.6084E-4
2	6.0042E-4	5.1361E-4	2.9835E-4	5.7309E-4	4.3320E-4	2.8588E-4
3	1.7361E-4	1.7448E-4	8.6268E-5	7.8085E-5	6.6723E-5	3.8945E-5

Table 4.11: Errors of Example 4.4 at $t = 2$.

h	Proposed Method		Dehghan and Ghesmati [44]	
	RMS errors	CPU time(s)	RMS errors	CPU time(s)
.05	2.9835E-4	.56	2.153E-4	–
.01	2.8758E-4	2.4	7.012E-5	–
.005	2.8819E-4	4.2	4.003E-5	–

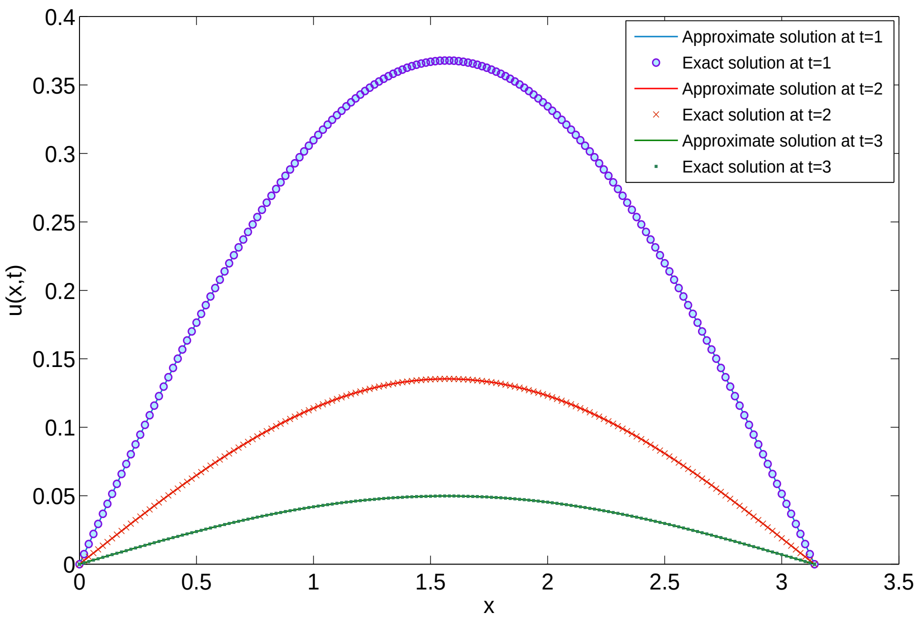


Figure 4.1: Numerical and exact solutions of Example 4.1 with $h = .02$, $\Delta t = .01$.

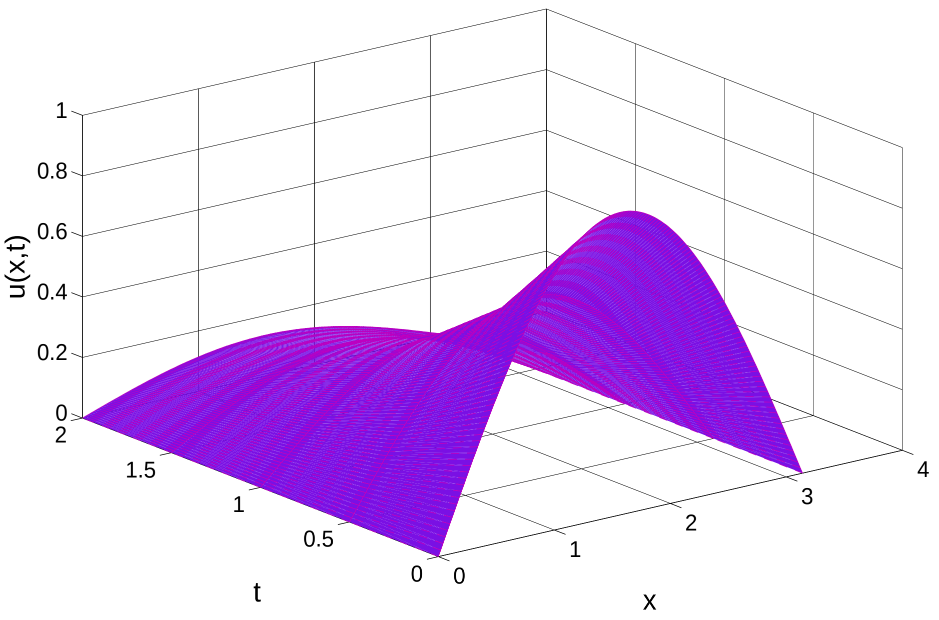


Figure 4.2: Space-time graph of numerical solutions of Example 4.1 up to $t = 2$.

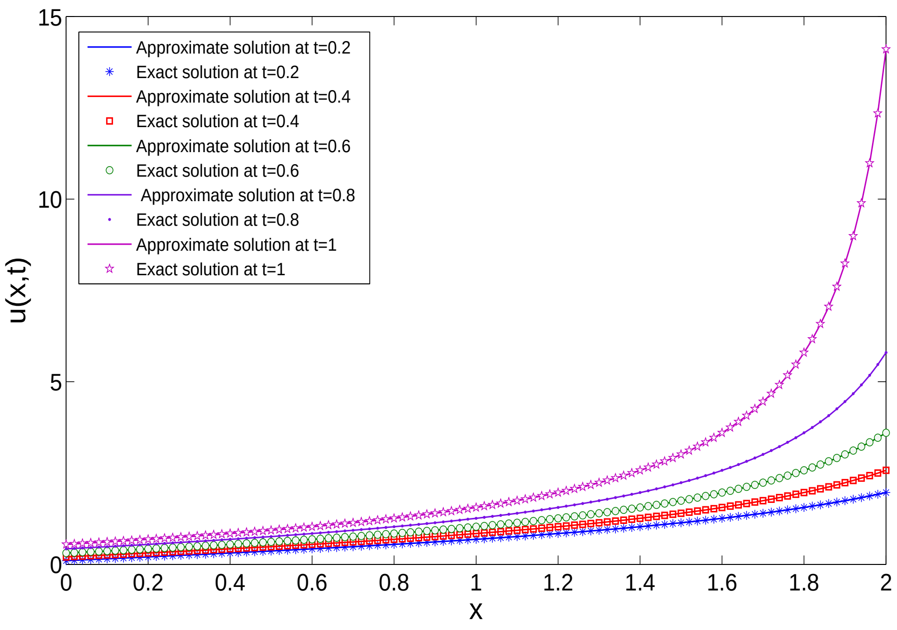


Figure 4.3: Numerical and exact solutions of Example 4.2 with $h = .02$, $\Delta t = .001$.

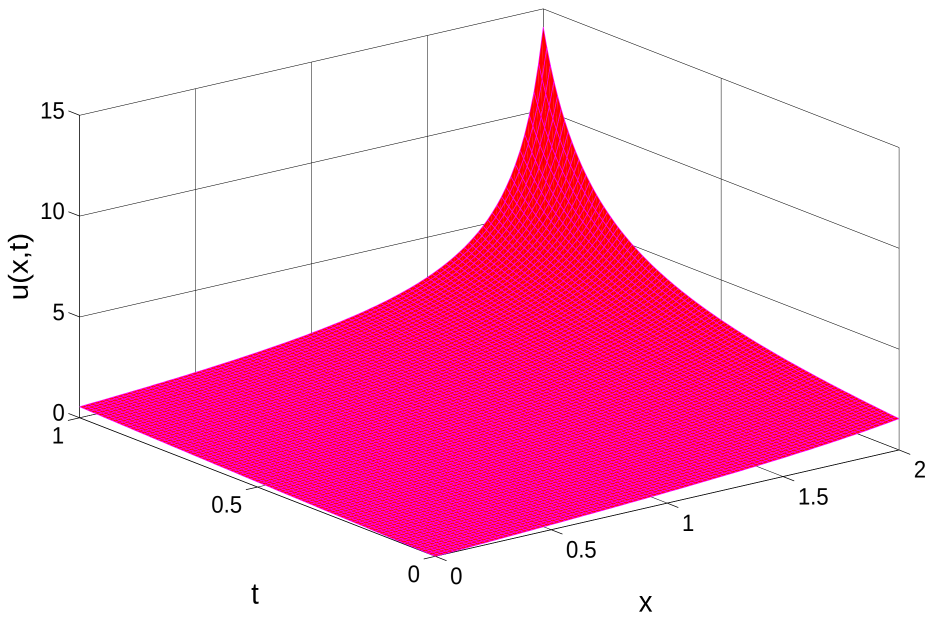


Figure 4.4: Space-time graph of numerical solutions of Example 4.2 up to $t = 1$.

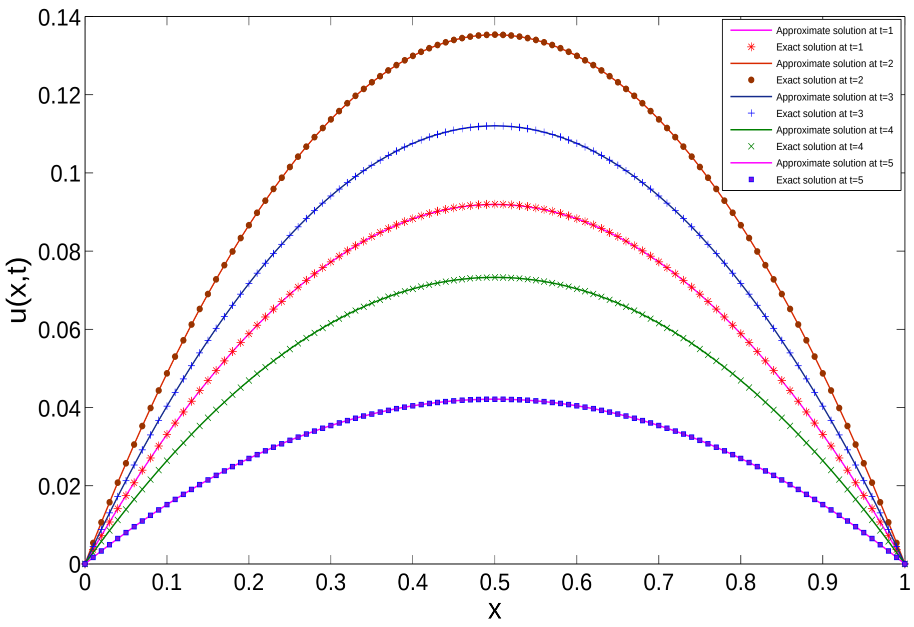


Figure 4.5: Numerical and exact solutions of Example 4.3 with $h = .01$, $\Delta t = .001$.

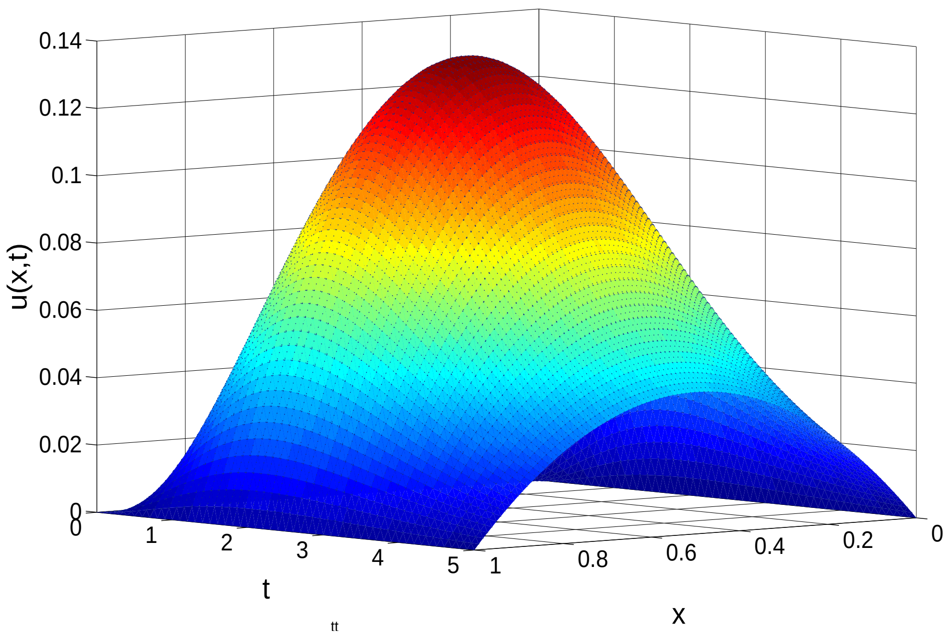


Figure 4.6: Space-time graph of numerical solutions of Example 4.3 up to t=5.

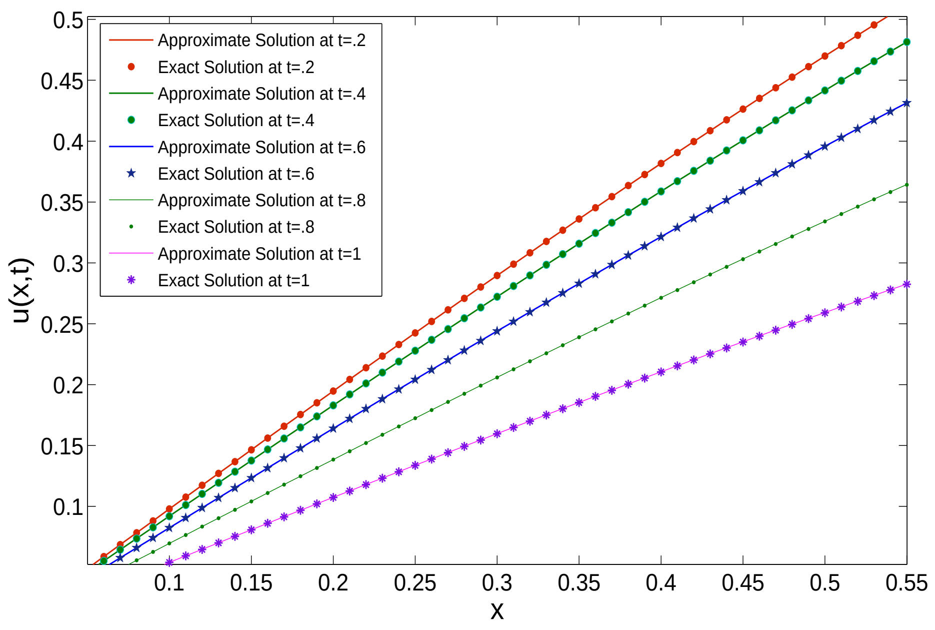


Figure 4.7: Numerical and exact solutions of Example 4.4 with $h = .01$, $\Delta t = .001$.

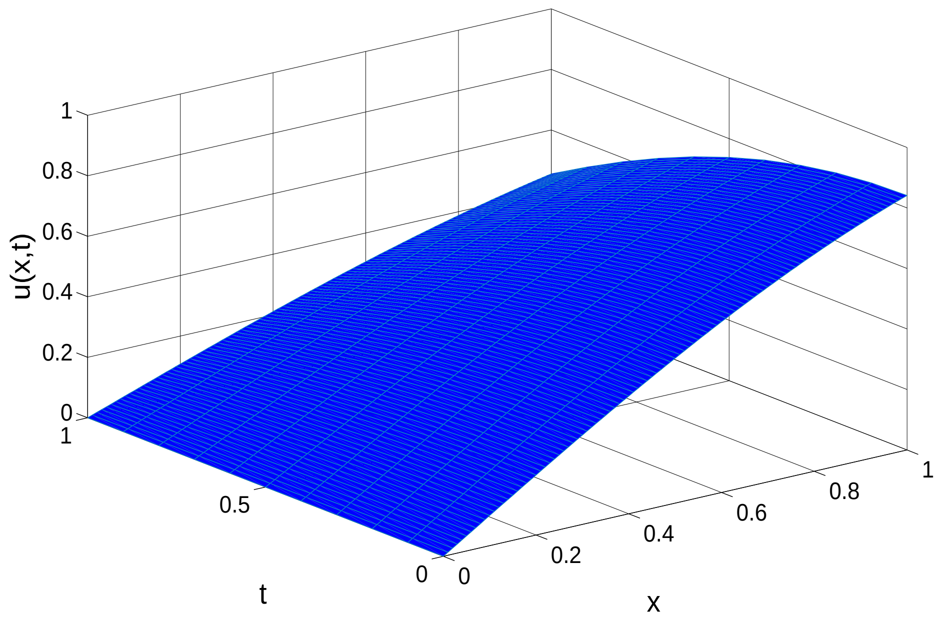


Figure 4.8: Space-time graph of numerical solutions of Example 4.4 up to $t = 1$.

Chapter 5

Numerical Solution of Two Dimensional Linear Telegraph Equation

5.1 Introduction

The study in this chapter is mainly concerned with the numerical solution of second-order linear two-space dimensional hyperbolic telegraph equation. The hyperbolic partial differential equations have a significant role in formulating fundamental equations in atomic physics [128] and are also very useful in understanding various phenomena in applied sciences like engineering, industry, aerospace as well as in chemistry and biology too. On one hand vibrations of structures can be easily analyzed and studied and on the other hand it is more convenient than ordinary diffusion equation in modeling reaction diffusion for such branches of sciences [44].
Consider the two dimensional hyperbolic telegraph equation

$$\begin{aligned} u_{tt}(x,y,t) + 2\alpha u_t(x,y,t) + \beta^2 u(x,y,t) = u_{xx}(x,y,t) + u_{yy}(x,y,t) + f(x,y,t), \\ (x,y,t) \in D \times (0,T], \end{aligned} \tag{5.1}$$

with the initial conditions

$$u(x,y,0) = u_0(x,y), \quad u_t(x,y,0) = v_0(x,y), \ (x,y) \in D. \tag{5.2}$$

The Dirichlet boundary conditions are given by

$$\begin{aligned} &u(a,y,t) = f_1(y,t), \ u(b,y,t) = f_2(y,t), \\ &u(x,c,t) = f_3(x,t), \ u(x,d,t) = f_4(x,t), \quad (x,y,t) \in \partial D \times (0,T], \end{aligned} \tag{5.3}$$

or Neumann boundary conditions are given by

$$\begin{aligned} &u_x(a,y,t)=g_1(y,t),\ \ u_x(b,y,t)=g_2(y,t),\\ &u_y(x,c,t)=g_3(x,t),\ \ u_y(x,d,t)=g_4(x,t),\quad (x,y,t)\in\partial D\times(0,T], \end{aligned} \tag{5.4}$$

where $D=\{(x,y):a\le x\le b, c\le y\le d\}$, ∂D is its boundary and $(0,T]$ is the time interval. α, β are the constants, for $\alpha>0$, $\beta=0$ equation (5.1) represents a damped wave equation and for $\alpha>0$, $\beta>0$ it is called telegraph equation.

On delving through the literature we found that much efforts have been taken for the numerical solution of hyperbolic telegraph equation. Various numerical schemes were developed for one dimensional telegraph equation such as Taylor matrix method [28], dual reciprocity boundary integral method [44], unconditionally stable finite difference scheme [72], Chebyshev tau method [186], interpolating scaling function method [128] etc. Numerical solution of linear hyperbolic telegraph equation in three space dimensions has been proposed by Mohanty et al. [156], by constructing an unconditionally stable ADI scheme. In [152], Mohanty propounded an unconditionally stable implicit difference scheme for one, two and three space dimensional telegraphic equations. Dehghan et al. [57] have used He's variational iteration method to solve linear, variable coefficient, fractional derivative and multi space dimensional telegraph equations.

In the recent past, much emphasis has been given in the literature for numerical solution of two dimensional hyperbolic telegraph equation. Bülbül and Sezer [27] have proposed a Taylor matrix method which converts the telegraph equation into the matrix equation and solutions are computed by solving matrix equation, which corresponds to a system of linear algebraic equations. Dehghan and Ghesmati [43] have explored two meshless methods, namely meshless local weak-strong (MLWS) and meshless local Petrov-Galerkin (MLPG) method for equations (5.1)-(5.4). Dehghan and Mohebbi [49] have solved equation (5.1) using higher order implicit collocation method. Ding and Zhang [58] have discussed compact finite difference scheme which is of the fourth order in both space and time. Mohanty and Jain [155] have derived an unconditionally stable alternating direction implicit scheme. A combination of boundary knot method (BKM) and analog equation method (AEM) for equation (5.1) has been proposed by Dehghan and Salehi [52]. Numerical solution of 2D telegraph equation with variable coefficients has been tackled by Dehghan and Shokri [55]. The differential quadrature method, which approximates the solution of the problem on a finite dimensional space by using polynomials as the basis of the space, has been applied to solve two dimensional telegraph equations with both Dirichlet and Neumann boundary conditions by Jiwari et al. [91].

In this chapter we have developed a differential quadrature method to solve two dimensional hyperbolic telegraph equation with Dirichlet and Neumann boundary conditions. In this method modified cubic B-spline basis functions (1.6) are used to compute the weighting coefficients. First we convert the telegraph equation (5.1) into a system of partial differential equations and further it is reduced into a system of ordinary differential equations by DQM. Then the obtained system is solved using SSP-RK43 [197] scheme. By employing DQM, accurate solutions can be obtained using fewer grid points in the spatial domain and hence computer execution time can be reduced. Seven numerical examples are solved to illustrate the accuracy and efficiency of the DQM. Stability of scheme is also examined using matrix stability analysis.

This chapter is organised as follows. In Section-5.2, process for execution of modified cubic B-spline differential quadrature method is described for equations (5.1)-(5.4). Stability of scheme is discussed in Section-5.3. To show the efficiency of the present approach computationally, seven numerical experiments are taken in Section-5.4. Finally, brief conclusions drawn from the present study are presented in Section-5.5.

5.2 Solution of Two-Dimensional Telegraph Equation

The region $a \leq x \leq b$, $c \leq y \leq d$ is discretized by taking N and M grid points in x and y direction respectively, such that $h_x = x_{i+1} - x_i$, $i = 1, 2, \ldots, N-1$ and $h_y = y_{j+1} - y_j$, $j = 1, 2, \ldots, M-1$. Then according to two-dimensional differential quadrature method stated in chapter 1, the first and second order partial derivatives of the function $u(x, y, t)$ with respect to x, at a point (x_i, y_j) can be approximated as follows

$$u_x^{(1)}(x_i, y_j, t) = \sum_{k=1}^{N} a_{ik}^{(1)} u(x_k, y_j, t), \quad i = 1, 2, \ldots, N;\ j = 1, 2, \ldots, M, \tag{5.5}$$

$$u_x^{(2)}(x_i, y_j, t) = \sum_{k=1}^{N} a_{ik}^{(2)} u(x_k, y_j, t), \quad i = 1, 2, \ldots, N;\ j = 1, 2, \ldots, M, \tag{5.6}$$

where $a_{ik}^{(1)}$ and $a_{ik}^{(2)}$ are the weighting coefficients related to first and second order partial derivatives of u w.r.to x, respectively.

Similarly, partial derivatives of the function $u(x, y, t)$ with respect to y, at a point (x_i, y_j) along line $x = x_i$ can be approximated as

$$u_y^{(1)}(x_i, y_j, t) = \sum_{k=1}^{M} b_{jk}^{(1)} u(x_i, y_k, t), \quad i = 1, 2, \ldots, N;\ j = 1, 2, \ldots, M, \tag{5.7}$$

$$u_y^{(2)}(x_i, y_j, t) = \sum_{k=1}^{M} b_{jk}^{(2)} u(x_i, y_k, t), \quad i = 1, 2, \ldots, N;\ j = 1, 2, \ldots, M, \tag{5.8}$$

where $b_{jk}^{(1)}$ and $b_{jk}^{(2)}$ are the weighting coefficients related to first and second order partial derivatives of u w.r.to y, respectively.

To compute the weighting coefficients $a_{ik}^{(1)}$ and $b_{jk}^{(1)}$, we use modified cubic B-spline differential quadrature method defined in chapter 1 (see Section-1.9). Then weighting coefficients $a_{ik}^{(2)}$ and $b_{jk}^{(2)}$ are computed using Shu's [192] recurrence formulae (1.27) and (1.28), respectively.

To solve telegraph equation (5.1), first we convert it into the coupled system using the transformation $u_t = v$

$$\begin{aligned} &u_t(x, y, t) = v(x, y, t), \\ &v_t(x, y, t) = -2\alpha v(x, y, t) - \beta^2 u(x, y, t) + u_{xx}(x, y, t) + u_{yy}(x, y, t) + f(x, y, t). \end{aligned} \tag{5.9}$$

Now the spatial derivatives of u are discretized by using the approximations (5.6) and (5.8) and finally we obtain the following system of first order ODEs in time

$$\begin{aligned} &\frac{du_{i,j}}{dt} = v(x_i, y_j, t), \\ &\frac{dv_{i,j}}{dt} = -2\alpha v(x_i, y_j, t) - \beta^2 u(x_i, y_j, t) + \sum_{k=1}^{N} a_{ik}^{(2)} u(x_k, y_j, t) + \sum_{k=1}^{M} b_{jk}^{(2)} u(x_i, y_k, t) + f(x_i, y_j, t), \\ &\qquad (x_i, y_j, t) \in D \times (0, T],\ i = 1, 2, \ldots, N;\ j = 1, 2, \ldots, M, \end{aligned} \tag{5.10}$$

with the following initial conditions

$$\begin{aligned} &u(x_i, y_j, 0) = u_0(x_i, y_j), \\ &v(x_i, y_j, 0) = v_0(x_i, y_j),\ (x_i, y_j) \in D. \end{aligned} \tag{5.11}$$

5.2.1 Treatment of Boundary Conditions

The Dirichlet boundary conditions are directly used on the boundary and do not need any further simplification. But when the boundary conditions are of Neumann type or mixed, they are discretized using the modified B-spline differential quadrature method and further simplified to get the solutions at the boundary points.
Dirichlet boundary conditions (5.3) give

$$\begin{aligned} &u_{1,j} = f_1(y_j, t), \;\; u_{N,j} = f_2(y_j, t), \\ &u_{i,1} = f_3(x_i, t), \;\; u_{i,M} = f_4(x_i, t), \quad (x_i, y_j, t) \in \partial D \times (0, T]. \end{aligned} \tag{5.12}$$

Neumann boundary conditions (5.4) along $x = a$ and $x = b$ are approximated as

$$\begin{aligned} &\sum_{k=1}^{N} a_{1k}^{(1)} u(x_k, y_j, t) = g_1(y_j, t), \\ &\sum_{k=1}^{N} a_{Nk}^{(1)} u(x_k, y_j, t) = g_2(y_j, t), \;\; j = 1, 2, \ldots, M. \end{aligned} \tag{5.13}$$

Rewriting the above system as

$$\begin{aligned} a_{11}^{(1)} u_{1,j} + a_{1N}^{(1)} u_{N,j} &= g_1 - \sum_{k=2}^{N-1} a_{1k}^{(1)} u_{k,j}, \\ a_{N1}^{(1)} u_{1,j} + a_{NN}^{(1)} u_{N,j} &= g_2 - \sum_{k=2}^{N-1} a_{Nk}^{(1)} u_{k,j}, \;\; j = 1, 2, \ldots, M. \end{aligned} \tag{5.14}$$

On solving the above system of equations, we get approximate values of u on $x = a$ and $x = b$ as

$$\begin{aligned} u_{1,j} &= \frac{A - s_1}{(a_{1,1}^{(1)} a_{N,N}^{(1)} - a_{N,1}^{(1)} a_{1,N}^{(1)})}, \\ u_{N,j} &= \frac{B - s_2}{(a_{1,1}^{(1)} a_{N,N}^{(1)} - a_{N,1}^{(1)} a_{1,N}^{(1)})}, \;\; j = 1, 2, \ldots, M, \end{aligned} \tag{5.15}$$

where

$$A = (g_1 a_{N,N}^{(1)} - g_2 a_{1,N}^{(1)}), \;\; s_1 = \sum_{k=2}^{N-1} (a_{N,N}^{(1)} a_{1,k}^{(1)} - a_{1,N}^{(1)} a_{N,k}^{(1)}) u_{k,j},$$

$$B = (g_2 a_{1,1}^{(1)} - g_1 a_{N,1}^{(1)}), \;\; s_2 = \sum_{k=2}^{N-1} (a_{1,1}^{(1)} a_{N,k}^{(1)} - a_{N,1}^{(1)} a_{1,k}^{(1)}) u_{k,j}.$$

In a similar way, Neumann boundary conditions (5.4) along $y = c$ and $y = d$ are approximated as

$$\begin{aligned}
&\sum_{k=1}^{M} b_{1k}^{(1)} u(x_i, y_k, t) = g_3(x_i, t), \\
&\sum_{k=1}^{M} b_{Mk}^{(1)} u(x_i, y_k, t) = g_4(x_i, t), \ \ i = 1, 2, \ldots, N.
\end{aligned} \tag{5.16}$$

Solution of above system will give $u_{i,1}$ and $u_{i,M}$ as

$$\begin{aligned}
u_{i,1} &= \frac{C - s_3}{(b_{1,1}^{(1)} b_{M,M}^{(1)} - b_{M,1}^{(1)} b_{1,M}^{(1)})}, \\
u_{i,M} &= \frac{D - s_4}{(b_{1,1}^{(1)} b_{M,M}^{(1)} - b_{M,1}^{(1)} b_{1,M}^{(1)})}, \ \ , i = 1, 2, \ldots, N,
\end{aligned} \tag{5.17}$$

where

$$\begin{aligned}
C &= (g_3 b_{M,M}^{(1)} - g_4 b_{1,M}^{(1)}), \ \ s_3 = \sum_{k=2}^{M-1} (b_{M,M}^{(1)} b_{1,k}^{(1)} - b_{1,M}^{(1)} b_{M,k}^{(1)}) u_{i,k}, \\
D &= (g_4 b_{1,1}^{(1)} - g_3 b_{M,1}^{(1)}), \ \ s_4 = \sum_{k=2}^{M-1} (b_{1,1}^{(1)} b_{M,k}^{(1)} - b_{M,1}^{(1)} b_{1,k}^{(1)}) u_{i,k}.
\end{aligned}$$

To find the solution, first we compute the initial vector $[u_{i,j}^0, v_{i,j}^0]^T$, $i = 1 : N; j = 1 : M$, using the initial conditions (5.11). Then SSP-RK43 scheme has been employed to solve the system of first order ODEs (5.10) with the appropriate boundary conditions and hence the approximate solution $[u_{i,j}^k, v_{i,j}^k]^T$ is computed at a time level $t = t_k$.

5.3 Stability Analysis

We consider the system (5.10) and rewrite it in compact form as

$$\frac{dW}{dt} = PW + F \tag{5.18}$$

or

$$\frac{d}{dt}\begin{bmatrix} u \\ v \end{bmatrix} = \begin{bmatrix} O & I \\ Q & -2\alpha I \end{bmatrix} \begin{bmatrix} u \\ v \end{bmatrix} + \begin{bmatrix} O_1 \\ F_i(t) \end{bmatrix}, \tag{5.19}$$

where O=null matrix,

I=identity matrix of order $(N-2)(M-2)$,

Q is a square matrix of weighting coefficients of order $(N-2)(M-2)$,
$W=[u,v]^T$ is solution vector at the interior grid points, given by
$W=[u_{2,2},u_{2,3},\ldots,u_{2,M-1},u_{3,2},\ldots,u_{3,M-1},\ldots,u_{N-1,2},\ldots,u_{N-1,M-1},v_{2,2},v_{2,3},\ldots,v_{2,M-1}$
$,v_{3,2},\ldots,v_{3,M-1},\ldots,v_{N-1,2},\ldots,v_{N-1,M-1}]^T$.
$F=[O_1,F_i(t)]^T$ is a vector containing non-homogenous part and boundary conditions, where O_1 is null and $F_i(t)$ is column vector whose entries are given by
$F_i(t)=-(a^{(2)}_{i,1}u_{1,j}+a^{(2)}_{i,N}u_{N,j})-(b^{(2)}_{j,1}u_{i,1}+b^{(2)}_{j,M}u_{i,M})+f(x_i,y_j,t), i=2:N-1; j=2:M-1.$

The stability of the system (5.18) is very important since it is related to the stability of numerical scheme for solving it. If the system of ordinary differential equations (5.18) is unstable then stable numerical scheme for temporal discretization may not generate converged solution. The stability of this system depends on the eigenvalues of coefficient matrix P, since its exact solution can be directly found using the eigenvalues. If all the eigenvalues of P are having negative real part then the system will be stable.

Let λ_P is an eigenvalue of P and X_1, X_2 be two components (each of order $(N-2)(M-2)$) of eigenvector corresponding to eigenvalue λ_P. Then we have

$$\begin{bmatrix} O & I \\ Q & -2\alpha I \end{bmatrix}\begin{bmatrix} X_1 \\ X_2 \end{bmatrix}=\lambda_P\begin{bmatrix} X_1 \\ X_2 \end{bmatrix}, \tag{5.20}$$

which gives

$$\begin{aligned} IX_2 &= \lambda_P X_1, \\ QX_1-2\alpha X_2 &= \lambda_P X_2. \end{aligned} \tag{5.21}$$

From above set of equations, we get

$$QX_1=(\lambda_P^2+2\alpha\lambda_P)X_1, \tag{5.22}$$

$\Rightarrow \lambda_P(\lambda_P+2\alpha)$ is an eigenvalue of Q.
The matrix Q is given by

$$Q=-\beta^2 I+Q_1+Q_2, \tag{5.23}$$

where Q_1 and Q_2 are the matrices of weighting coefficients $a^{(2)}_{ij}$ and $b^{(2)}_{ij}$, given by

$$Q_1=\begin{bmatrix} a^{(2)}_{22}I_{M-2} & a^{(2)}_{23}I_{M-2} & \ldots & a^{(2)}_{2,N-1}I_{M-2} \\ a^{(2)}_{32}I_{M-2} & a^{(2)}_{33}I_{M-2} & \ldots & a^{(2)}_{3,N-1}I_{M-2} \\ \vdots & & \ddots & \vdots \\ a^{(2)}_{N-1,2}I_{M-2} & a^{(2)}_{N-1,3}I_{M-2} & \ldots & a^{(2)}_{N-1,N-1}I_{M-2} \end{bmatrix}, \tag{5.24}$$

Q_2 is block diagonal matrix, given by

$$Q_2 = \begin{bmatrix} R & O & \dots & O \\ O & R & \dots & O \\ \vdots & & \ddots & \vdots \\ O & O & \dots & R \end{bmatrix}, \tag{5.25}$$

and

$$R = \begin{bmatrix} b_{22} & b_{23} & \dots & b_{2,M-1} \\ b_{32} & b_{32} & \dots & b_{2,M-1} \\ \vdots & & \ddots & \vdots \\ b_{M-1,2} & b_{M-1,2} & \dots & b_{M-1,M-1} \end{bmatrix}_{(M-2)\times(M-2)} \tag{5.26}$$

We have computed the eigenvalues of the matrix Q_1 and Q_2 by taking different grid points. In Figure-5.1 we have plotted the eigenvalues of matrix Q for 6×6, 11×11, 21×21, 31×31, 41×41 grid points in space and found that all the eigenvalues of Q are real and negative for different values of N and M.

From (5.22) it is clear that $\lambda_P(2\alpha + \lambda_P)$ will be negative and real.

Let $\lambda_P = x + iy$, where x and y are real numbers.

We have $(x + iy)(x + iy + 2\alpha)$ is real and negative.

i.e.

$$\begin{aligned} x(x + 2\alpha) - y^2 < 0, \\ y(x + \alpha) = 0. \end{aligned} \tag{5.27}$$

From above set of equations, we get the solution as

1. $x + \alpha = 0$ and y is arbitrary real number.

   $\Rightarrow$ x is negative real number, since α is real and positive.

2. if $y = 0$

   $\Rightarrow x(x + 2\alpha) < 0,$

   $\Rightarrow (x + \alpha)^2 < \alpha^2,$

   $\Rightarrow -2\alpha < x < 0$, since α is real and positive.

   $\Rightarrow$ Real part of eigenvalues of P are negative.

We conclude that the real part of eigenvalues of P are always negative, i.e. the system (5.18) is stable.

5.4 Numerical Results

To illustrate the efficiency of proposed scheme, we have solved seven examples which are also considered by other researchers. In order to measure the accuracy of numerical solutions we have computed L_2, L_∞ and relative errors.

Example 5.1

We consider equation (5.1) in the region $0 \le x,\ y \le 1$ with $\alpha = 1, \beta = 1,\ f(x,y,t) = 2(\cos t - \sin t)\sin x \sin y$ and the following initial conditions

$$\begin{aligned} &u(x,y,0) = \sin x\, \sin y, \\ &u_t(x,y,0) = 0. \end{aligned} \tag{5.28}$$

The Dirichlet boundary conditions are given by

$$\begin{aligned} &u(0,y,t) = 0, && 0 \le y \le 1,\ x = 0, \\ &u(1,y,t) = \cos t\, \sin(1)\, \sin y, && 0 \le y \le 1,\ x = 1, \\ &u(x,0,t) = 0, && 0 \le x \le 1,\ y = 0, \\ &u(x,1,t) = \cos t\, \sin x\, \sin(1), && 0 \le x \le 1,\ y = 1. \end{aligned} \tag{5.29}$$

The exact solution is given in [43] as

$$u(x,y,t) = \cos t\, \sin x\, \sin y. \tag{5.30}$$

Results are computed for $\Delta t = .01,\ h_x = h_y = .1$ and $\Delta t = .001,\ h_x = h_y = .05$ and reported in Table-5.1. A comparison of exact and numerical solutions at time $t = 1,\ 2,\ 3$ is depicted in Figure-5.2 with $\Delta t = .001$ and $h_x = h_y = .05$, which shows that the numerical solutions are in excellent agreement with the exact solutions.

Example 5.2

In this example, we consider the telegraph equation (5.1) in the region $0 \le x,\ y \le 1$ with $f(x,y,t) = (-2\alpha + \beta^2 - 1)e^{-t} \sinh x\, \sinh y$ and the following initial conditions

$$\begin{aligned} &u(x,y,0) = \sinh x\, \sinh y, \\ &u_t(x,y,0) = -\sinh x\, \sinh y. \end{aligned} \tag{5.31}$$

Dirichlet boundary conditions are given by

$$\begin{aligned}
&u(0,y,t) = 0, && 0 \le y \le 1,\ x = 0,\\
&u(1,y,t) = e^{-t}\sinh(1)\ \sinh y, && 0 \le y \le 1,\ x = 1,\\
&u(x,0,t) = 0, && 0 \le x \le 1,\ y = 0,\\
&u(x,1,t) = e^{-t}\sinh x\ \sinh(1), && 0 \le x \le 1,\ y = 1.
\end{aligned} \tag{5.32}$$

The exact solution is given in [91] as

$$u(x,y,t) = e^{-t}\sinh x\ \sinh y. \tag{5.33}$$

This example is solved for $\alpha = 10,\ \beta = 5$ and $\alpha = 10,\ \beta = 0$. In Table-5.2, results are reported at different time levels with $\Delta t = .01$ and $h_x = h_y = .1$. Table-5.3 shows the comparison of our results with those of Jiwari et al. [91] with $\Delta t = .001$ and $h_x = h_y = .05$. We observe that our results are comparable to that of [91] in terms of relative error, however the CPU time taken in our computation is very less. So we can conclude that our scheme produces good accuracy with less computational effort. Figure-5.3 depicts the comparison of numerical and exact solutions at $t = 1,\ 2,\ 3$ with $\Delta t = .001$ and $h_x = h_y = .05$.

Example 5.3

Consider the following telegraph equation in the region $0 \le x,\ y \le 1$,

$$u_{tt}+4\pi u_t+2\pi^2 u = u_{xx}+u_{yy}+2\pi\sin\pi(x+y)e^{-(x+y)t}+((x+y-2\pi)^2-2t^2)\sin(\pi x)\sin(\pi y)e^{-(x+y)t}. \tag{5.34}$$

The initial and Dirichlet boundary conditions are given by

$$\begin{aligned}
&u(x,y,0) = \sin\pi x\ \sin\pi y,\\
&u_t(x,y,0) = -(x+y)\sin\pi x\ \sin\pi y.
\end{aligned} \tag{5.35}$$

$$\begin{aligned}
&u(0,y,t) = 0, && 0 \le y \le 1,\ x = 0,\\
&u(1,y,t) = 0, && 0 \le y \le 1,\ x = 1,\\
&u(x,0,t) = 0, && 0 \le x \le 1,\ y = 0,\\
&u(x,1,t) = 0, && 0 \le x \le 1,\ y = 1.
\end{aligned} \tag{5.36}$$

The exact solution is given in [43] as

$$u(x,y,t) = e^{-(x+y)} \sin \pi x \, \sin \pi y. \tag{5.37}$$

In Table-5.4, results are reported with $\Delta t = .01$ and $h_x = h_y = .1$. Table-5.5 shows the accuracy of computed solutions in terms of L_2, L_∞ and relative errors with $\Delta t = .001$ and $h_x = h_y = .025$ and also compares the relative error at $t = 2$ with those obtained by Dehghan and Ghesmati [43]. The plots of numerical and exact solutions at different time levels are depicted in Figure-5.4.

Example 5.4

We consider the telegraph equation (5.1) in the region $0 \le x,\ y \le 1$ with $f(x,y,t) = (-3\cos t - 2\alpha \sin t + \beta^2 \cos t)\sinh x \sinh y$ and with the following initial conditions

$$\begin{aligned} &u(x,y,0) = \sinh x \, \sinh y, \\ &u_t(x,y,0) = 0. \end{aligned} \tag{5.38}$$

The Dirichlet boundary conditions (5.3) are given by

$$\begin{aligned} &u(0,y,t) = 0, && 0 \le y \le 1,\ x = 0, \\ &u(1,y,t) = \cos t \, \sinh(1) \, \sinh y, && 0 \le y \le 1,\ x = 1, \\ &u(x,0,t) = 0, && 0 \le x \le 1,\ y = 0, \\ &u(x,1,t) = \cos t \, \sinh x \, \sinh(1), && 0 \le x \le 1.\ y = 1. \end{aligned} \tag{5.39}$$

The exact solution is given in [91] as

$$u(x,y,t) = \cos t \, \sinh x \, \sinh y. \tag{5.40}$$

We solve this problem for $\alpha = 10$, $\beta = 5$ and $\alpha = 50$, $\beta = 5$ with $\Delta t = .001$, $h_x = h_y = .05$. Table-5.6 presents the computed errors and CPU time at different time levels. Relative errors are also compared with those obtained by Jiwari et al. [91]. It can be seen from the table that our results are comparable to that of [91] but CPU time used in our computation is very less. Figure-5.5 demonstrates the comparison of numerical solutions with exact solutions at $t = 1,\ 2,\ 3$.

Example 5.5

We consider the telegraph equation (5.1) in the region $0 \le x,\ y \le 1$ with $\alpha = 1$, $\beta = 1$, $f(x, y, t) = -2e^{x+y-t}$ and the following initial conditions

$$\begin{aligned} u(x, y, 0) &= e^{x+y}, \\ u_t(x, y, 0) &= -e^{x+y}, \end{aligned} \tag{5.41}$$

and with the following mixed boundary conditions

$$\begin{aligned} u(0, y, t) &= e^{y-t}, \quad 0 \le y \le 1,\ x = 0, \\ u(1, y, t) &= e^{1+y-t},\ 0 \le y \le 1,\ x = 1, \\ \frac{\partial u}{\partial y}(x, 0, t) &= e^{x-t},\ 0 \le x \le 1,\ y = 0, \\ u(x, 1, t) &= e^{1+x-t},\ 0 \le x \le 1.\ y = 1. \end{aligned} \tag{5.42}$$

The exact solution is given in [43] as

$$u(x, y, t) = e^{x+y-t}. \tag{5.43}$$

To get the numerical solutions, first we take $\Delta t = .01$, $h_x = h_y = .1$ and results are reported in Table-5.7. Further, we check the performance of proposed scheme on a finer grid $h_x = h_y = .05$ and with $\Delta t = .001$. Results are shown in Table-5.8 up to time $t = 10$ and compared with the results of Dehghan and Ghesmati [43], in terms of relative errors. We noticed that our results are better than that given in [43] and CPU time used in our computations is much less. Figure-5.6 shows the comparison of numerical and exact solutions at $t = 1,\ 2,\ 4$ with $\Delta t = .001$ and $h_x = h_y = .05$.

Example 5.6

Consider the telegraph equation (5.1) with $\alpha = 1$, $\beta = 1$, $f(x, y, t) = 2\pi^2 e^{-t} \sin \pi x\ \sin \pi y$ and the following initial conditions

$$\begin{aligned} u(x, y, 0) &= \sin \pi x\ \sin \pi y, \\ u_t(x, y, 0) &= -\sin \pi x\ \sin \pi y, \end{aligned} \tag{5.44}$$

and with the following mixed boundary conditions

$$\begin{aligned} &\frac{\partial u}{\partial x}(0,y,t) = \pi e^{-t}\sin \pi y, \quad 0 \le y \le 1,\ x = 0,\\ &u(1,y,t) = 0, \qquad\qquad\quad 0 \le y \le 1,\ x = 1,\\ &u(x,0,t) = 0, \qquad\qquad\quad 0 \le x \le 1,\ y = 0,\\ &\frac{\partial u}{\partial y}(x,1,t) = -\pi e^{-t}\sin \pi x,\ 0 \le x \le 1.\ y = 1. \end{aligned} \tag{5.45}$$

The exact solution is given in [43] as

$$u(x,y,t) = e^{-t}\sin \pi x\, \sin \pi y. \tag{5.46}$$

Table-5.9 reports the L_2, L_∞, relative errors and CPU time up to time level $t = 10$ with $\Delta t = .01$ and $h_x = h_y = .1$. Table-5.10 provides the numerical results with $\Delta t = .001$ and $h_x = h_y = .05$ and also shows the comparison of our results with those of Dehghan and Ghesmati [43] and Jiwari et al. [91], in term of relative errors. It is clear from the table that our scheme produces much better results in comparison with [43, 91]. In Figure-5.7, numerical and exact solutions are plotted at $t = .5,\ 1,\ 2$ and it is noticed that the numerical solutions produced by presented method exhibit correct physical behavior for different values of t.

Example 5.7

Consider (5.1) for $\alpha = 1, \beta = 1$ in the domain $0 \le x,\ y \le 1$ with the following initial conditions

$$\begin{aligned} &u(x,y,0) = \log(1+x+y),\\ &u_t(x,y,0) = \frac{1}{1+x+y}, \end{aligned} \tag{5.47}$$

and with the following mixed boundary conditions

$$\begin{aligned} &u(0,y,t) = \log(1+y+t),\ 0 \le y \le 1,\ x = 0,\\ &\frac{\partial u}{\partial x}(1,y,t) = \frac{1}{2+y+t}, \quad 0 \le y \le 1,\ \ x = 1,\\ &\frac{\partial u}{\partial y}(x,0,t) = \frac{1}{1+x+t}, \quad 0 \le x \le 1,\ \ y = 0,\\ &u(x,1,t) = \log(2+x+t),\ 0 \le x \le 1,\ y = 1. \end{aligned} \tag{5.48}$$

The exact solution is given in [43] as

$$u(x, y, t) = \log(1 + x + y + t), \tag{5.49}$$

and we extract $f(x, y, t)$ from the exact solution.

Numerical results are computed for $\Delta t = .001$ and $h_x = h_y = .05$ and reported in Table-5.11 in terms of L_2, L_∞, relative errors and CPU time. The obtained results demonstrate that the proposed scheme produces better results with less computational cost in comparison with Dehghan and Ghesmati [43]. Figure-5.8 shows the surface plots of numerical and exact solutions at $t = 1$, 3, 5, which also confirms the good accuracy of the scheme.

5.5 Conclusions

The following observations have been made based on present study:

1. A differential quadrature method based on modified cubic B-spline basis functions has been developed to solve two dimensional hyperbolic telegraph equation, which can easily deal with Dirichlet's or Mixed boundary conditions.

2. The presented method converts the telegraph equation into a system of first order ordinary differential equations, which has been solved using the SSP-RK43 scheme.

3. Seven numerical examples have been solved and it is observed that the scheme produces better results while taking less CPU time in comparison with other known work.

4. The stability test is performed with matrix stability analysis and scheme is found to be unconditionally stable.

5. The combination of modified cubic B-spline differential quadrature method in space and SSP-RK43 scheme in time is found to be a apposite solution procedure in terms of accuracy and computational cost for solving two dimensional telegraph equation.

Table 5.1: Errors and CPU time of Example 5.1.

t	L_2	L_∞	Relative-errors	CPU time(s)
$\Delta t = .01$, $h_x = h_y = .1$				
1	9.9722E-4	2.2746E-3	5.9762E-3	.08
2	1.0926E-3	2.8706E-3	8.5019E-3	.11
3	2.2877E-4	6.0818E-4	7.4720E-4	.14
5	1.1562E-3	2.9942E-3	1.2767E-3	.20
7	7.2867E-4	1.8781E-3	3.1572E-3	.26
10	5.8889E-4	1.5158E-3	2.2874E-3	.34
$\Delta t = .001$, $h_x = h_y = .05$				
1	9.8870E-5	2.4964E-4	6.2977E-4	.78
2	1.2148E-4	3.2296E-4	1.0025E-3	1.3
3	3.7627E-5	9.9310E-5	1.3078E-4	1.7
5	1.2762E-4	3.3205E-4	1.5411E-3	3.0
7	6.7672E-5	1.7679E-4	3.0892E-4	3.3
10	5.1764E-5	1.3521E-4	2.1245E-4	5.2

Table 5.2: Errors and CPU time of Example 5.2 with $\alpha = 10$, $\beta = 5$, $\Delta t = .01$, $h_x = h_y = .1$.

t	L_2	L_∞	Relative-errors	CPU time(s)
.5	8.3931E-4	3.3019E-3	2.8902E-3	.13
1	6.0254E-4	2.0597E-3	3.4208E-3	.16
2	2.4167E-4	7.6531E-4	3.7297E-3	.19
3	8.9534E-5	2.7920E-4	3.7937E-3	.24
5	1.2168E-5	3.7800E-5	3.8097E-3	.34

Table 5.3: Errors and CPU time of Example 5.2 with $\alpha = 10$, $\Delta t = .001$, $h_x = h_y = .05$.

t	Proposed method				Jiwari et al. [91]	
	L_2	L_∞	Rel. errors	CPU time(s)	Rel. errors	CPU time(s)
$\beta = 5$						
0.5	1.0690E-4	2.4738E-4	1.1088E-4	0.47	1.1185E-4	6
1	1.5293E-5	3.3082E-4	1.3266E-4	1.1	1.8051E-4	12
2	4.6468E-5	1.1380E-5	3.1954E-4	1.1	4.7289E-4	25
3	2.1994E-5	4.3577E-5	1.3024E-4	2.8	1.2656E-4	37
5	2.7151E-6	5.4141E-6	1.4439E-4	4.3	9.2770E-4	62
$\beta = 0$						
0.5	9.2959E-5	4.2348E-4	3.4675E-4	0.52	1.1198E-4	6
1	6.3652E-5	2.5838E-4	3.9146E-4	0.98	1.8635E-4	12
2	2.5540E-5	9.5843E-5	4.2739E-4	1.8	5.1797E-4	25
3	9.9234E-6	3.5340E-5	4.5140E-4	2.2	1.4412E-4	37
5	1.5116E-6	4.8043E-6	5.0758E-4	4.5	1.0883E-4	62

Table 5.4: Errors and CPU time of Example 5.3 with $\Delta t = .01$ and $h_x = h_y = .1$.

t	L_2	L_∞	Relative-errors	CPU time(s)
1	9.7047E-4	1.9232E-3	4.9447E-3	.05
2	3.7546E-4	1.0072E-3	4.3014E-3	.10
3	5.1383E-4	1.6753E-3	1.1908E-2	.17
5	6.8021E-4	2.6738E-3	4.8567E-2	.22

Table 5.5: Errors and CPU time of Example 5.3 with $\Delta t = .001$ and $h_x = h_y = .025$.

t	Proposed method				Dehghan and Ghesmati [43]
	L_2	L_∞	Relative-errors	CPU time(s)	Rel.-errors(MLWS)
1	1.1370E-4	1.9735E-4	5.7932E-4	1.1	-
2	1.1044E-4	3.4253E-4	1.2658E-4	1.9	4.731E-4
3	1.4336E-4	4.5378E-5	3.2997E-4	3.3	-
5	1.6269E-4	5.8801E-4	1.1501E-4	5.5	-
7	1.5769E-4	6.7607E-4	2.6368E-3	7.3	-
10	1.4145E-4	7.9495E-4	6.3276E-2	11.0	

Table 5.6: Errors and CPU time of Example 5.4 for $\beta = 5$ with $\Delta t = .001$ and $h_x = h_y = .05$.

t	Proposed method				Jiwari et al. [91]	
	L_2	L_∞	Relative errors	CPU time(s)	Relative errors	CPU time(s)
$\alpha = 10$						
0.5	1.0696E-4	3.7559E-4	1.3787E-5	.57	2.0149E-5	7
1	1.7174E-4	5.6395E-4	3.5957E-5	.92	4.6003E-5	13
2	1.6468E-4	5.1298E-4	4.4722E-5	1.2	4.4460E-5	27
3	8.9858E-6	1.9564E-5	1.0266E-6	2.3	6.4830E-6	39
5	1.7737E-4	5.5627E-4	7.0735E-5	4.1	7.3626E-5	69
7	1.4200E-4	4.7231E-4	2.1307E-5	5.4	-	-
10	1.2241E-4	4.1222E-4	1.6514E-5	7.4	2.4901E-5	139
$\alpha = 50$						
0.5	9.8800E-5	3.6962E-4	1.2735E-5	.57	2.8724E-5	7
1	1.6766E-4	5.6874E-4	3.5104E-5	.94	7.0064E-5	13
2	1.7109E-4	5.2572E-4	4.6408E-5	1.4	6.7759E-5	27
3	1.7406E-5	4.3459E-5	1.9886E-6	2.5	1.3856E-5	39
5	1.8420E-4	5.6940E-4	7.3460E-5	4.1	1.2840E-4	69
7	1.3760E-4	4.7587E-4	2.0647E-5	6.0	-	-
10	1.1691E-4	4.1396E-4	1.5772E-5	8.8	4.4202E-5	139

Table 5.7: Errors and CPU time of Example 5.5 with $\Delta t = .01$ and $h_x = h_y = .1$.

t	L_2	L_∞	Relative errors	CPU time(s)
1	1.4441E-2	2.9996E-2	1.0829E-3	.03
2	1.3898E-3	3.9711E-3	2.8333E-4	.05
3	1.3018E-3	2.2178E-3	7.2867E-4	.08
5	1.1112E-4	2.0618E-4	4.5956E-4	.11
7	1.3695E-5	3.0052E-5	4.1851E-4	.14
10	1.4408E-6	2.5354E-6	8.8440E-4	.19

Table 5.8: Errors and CPU time of Example 5.5 with $\Delta t = .001$ and $h_x = h_y = .05$.

	Proposed method				Dehghan and Ghesmati [43]			
t	L_2	L_∞	Rel. errors	CPU time(s)	Rel. err. (MLWS)	CPU time(s)	Rel. err. (MLPG)	CPU time(s)
0.5	3.4808E-3	9.5129E-3	8.4225E-5	.5	8.014E-5	7.4	2.032E-5	16.6
1	3.2351E-3	7.4749E-3	1.2906E-4	.7	2.020E-4	12.3	9.071E-5	28.9
2	2.8518E-4	1.0361E-4	3.0957E-5	1.3	9.791E-5	22.9	3.004E-4	42.3
3	3.1028E-4	5.7859E-4	9.1555E-5	1.9	7.029E-4	31.0	5.201E-4	53.7
4	9.0898E-5	2.7645E-4	7.2908E-5	2.3	1.703E-3	40.2	7.065E-4	67.4
5	2.4495E-5	6.7234E-5	5.3354E-5	3.3	-	-	-	-
7	2.5376E-6	8.2203E-6	4.0840E-5	3.9	-	-	-	-
10	3.6505E-6	8.5897E-6	1.1812E-5	5.2	-	-	-	-

Table 5.9: Errors and CPU time of Example 5.6 with $\Delta t = .01$ and $h_x = h_y = .1$.

t	L_2	L_∞	Rel. errors	CPU time(s)
1	1.6144E-3	3.6006E-3	8.7768E-4	.07
2	2.6345E-3	5.7068E-3	3.8933E-3	.09
3	5.3845E-4	1.2479E-3	2.1847E-3	.11
5	1.2418E-4	2.1003E-4	3.7232E-3	.15
7	1.3653E-5	2.6261E-5	3.0247E-3	.12
10	7.5592E-6	1.4083E-6	3.3635E-3	.20

Table 5.10: Errors and CPU time of Example 5.6 with $\Delta t = .001$ and $h_x = h_y = .05$.

t	Proposed method				Dehghan and Ghesmati [43]		Jiwari et al. [91]
	L_2	L_∞	Rel. errors	CPU time(s)	Rel. errors (MLWS)	Rel. errors (MLPG)	Rel. errors
0.5	3.5833E-4	9.5129E-4	8.4225E-5	.3	7.040E-5	3.701E-5	1.0078E-4
1	3.2351E-4	7.4749E-4	1.2906E-4	.7	9.088E-5	7.900E-5	9.1354E-4
2	2.8518E-5	1.0361E-4	3.0957E-5	1.3	4.820E-4	1.216E-4	1.1962E-4
3	3.1028E-5	5.7859E-4	9.1555E-5	1.7	1.400E-3	8.302E-4	1.3267E-5
5	2.4495E-6	6.7234E-5	5.3354E-4	2.9	-	-	1.3277E-5
7	2.5376E-7	8.2203E-7	4.0840E-5	4.1	-	-	–
10	3.6505E-9	8.5897E-8	1.1812E-5	5.4	-	-	3.3574E-6

Table 5.11: Errors and CPU time of Example 5.7 with $\Delta t = .001$ and $h_x = h_y = .05$.

t	Proposed method				Dehghan and Ghesmati [43]			
	L_2	L_∞	Rel. errors	CPU time(s)	Rel. errors (MLWS)	CPU time(s)	Rel. errors (MLPG)	CPU time(s)
0.5	1.0690E-3	2.4738E-3	1.1088E-3	.5	7.939E-5	9.2	9.991E-5	21.0
1	1.5293E-3	3.3082E-3	1.3266E-3	1.1	9.098E-5	12.9	7.198E-5	36.2
2	4.6468E-4	1.1380E-3	3.1954E-4	2.0	8.705E-4	25.7	8.784E-5	49.1
3	2.1994E-4	4.3577E-4	1.3024E-4	2.8	9.931E-4	38.1	4.801E-4	66.8
4	2.7151E-4	5.4141E-4	1.4435E-5	4.3	4.703E-3	49.8	6.091E-4	82.0
5	1.7201E-4	3.4812E-4	8.4225E-5	7.0	7.302E-3	62.0	9.498E-4	97.3
10	7.7285E-5	1.4037E-4	2.9624E-5	9.6	-	-	-	-

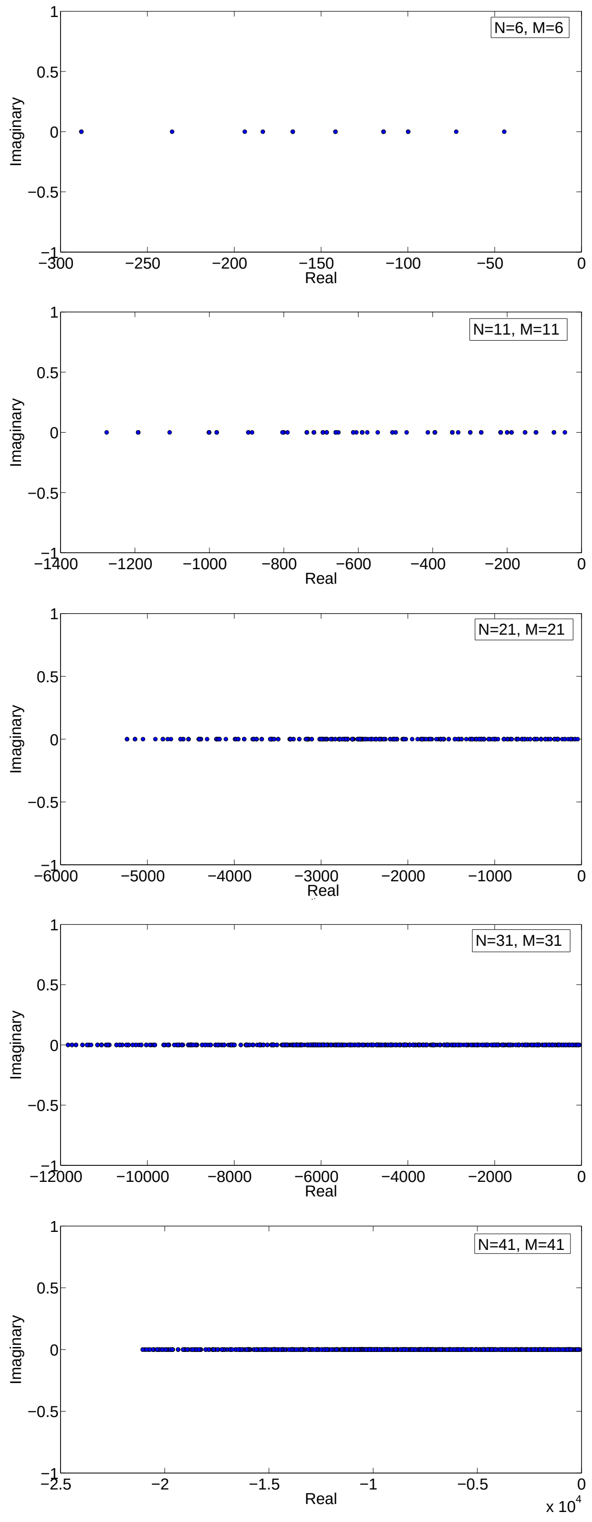


Figure 5.1: Eigenvalues of matrix Q using different grid points.

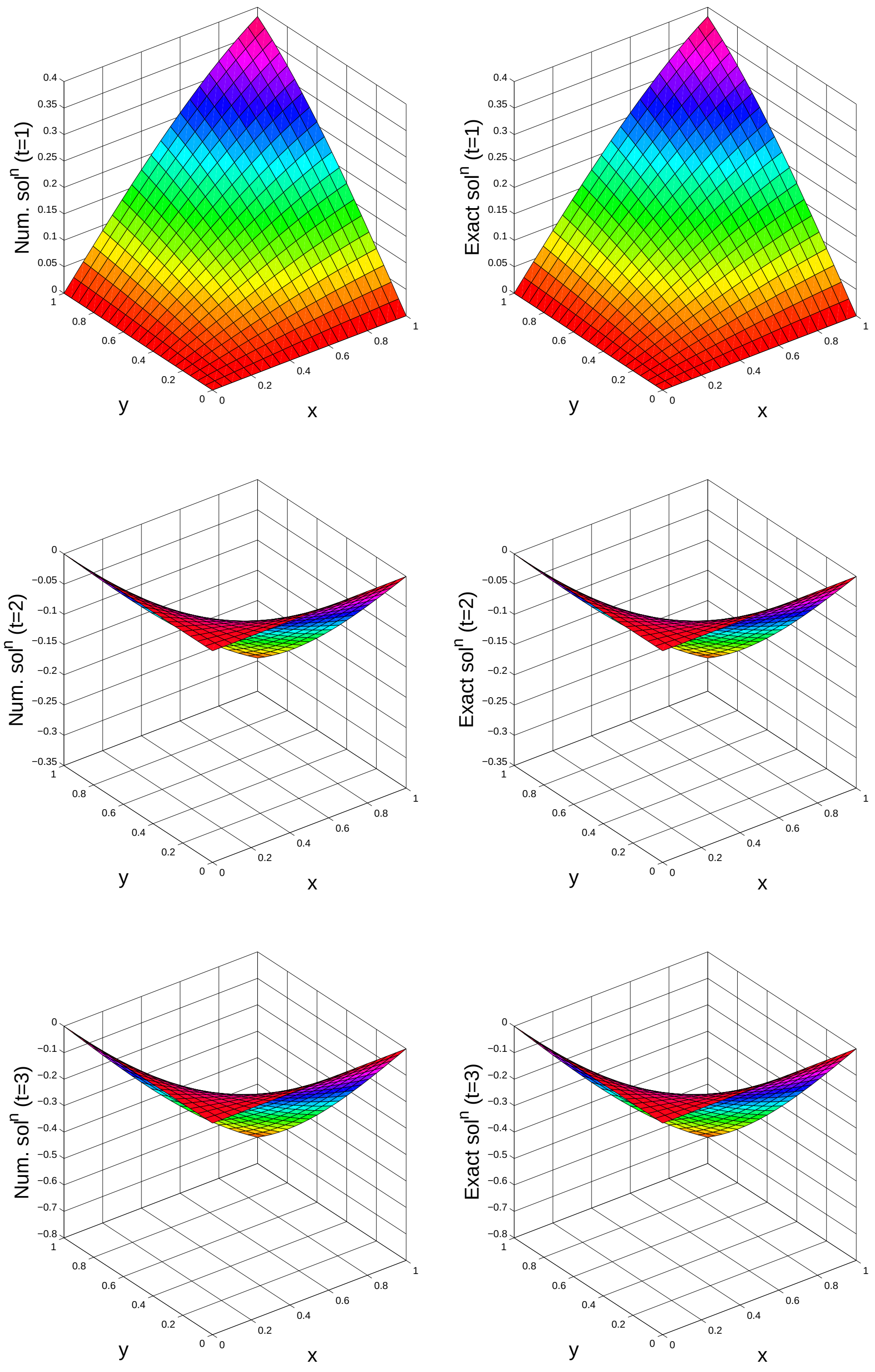


Figure 5.2: Plots of numerical and exact solutions of Example 5.1.

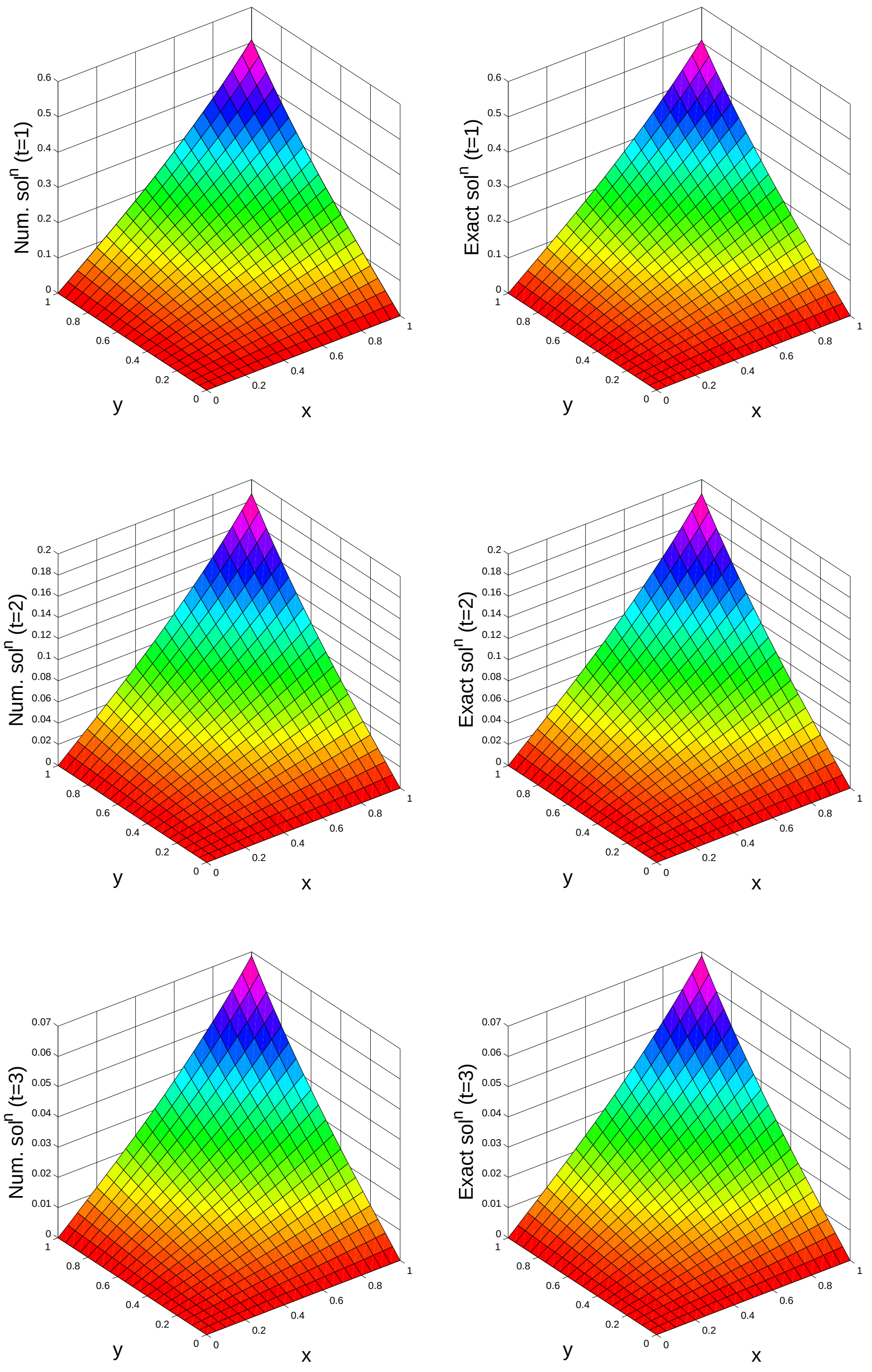


Figure 5.3: Plots of numerical and exact solutions of Example 5.2.

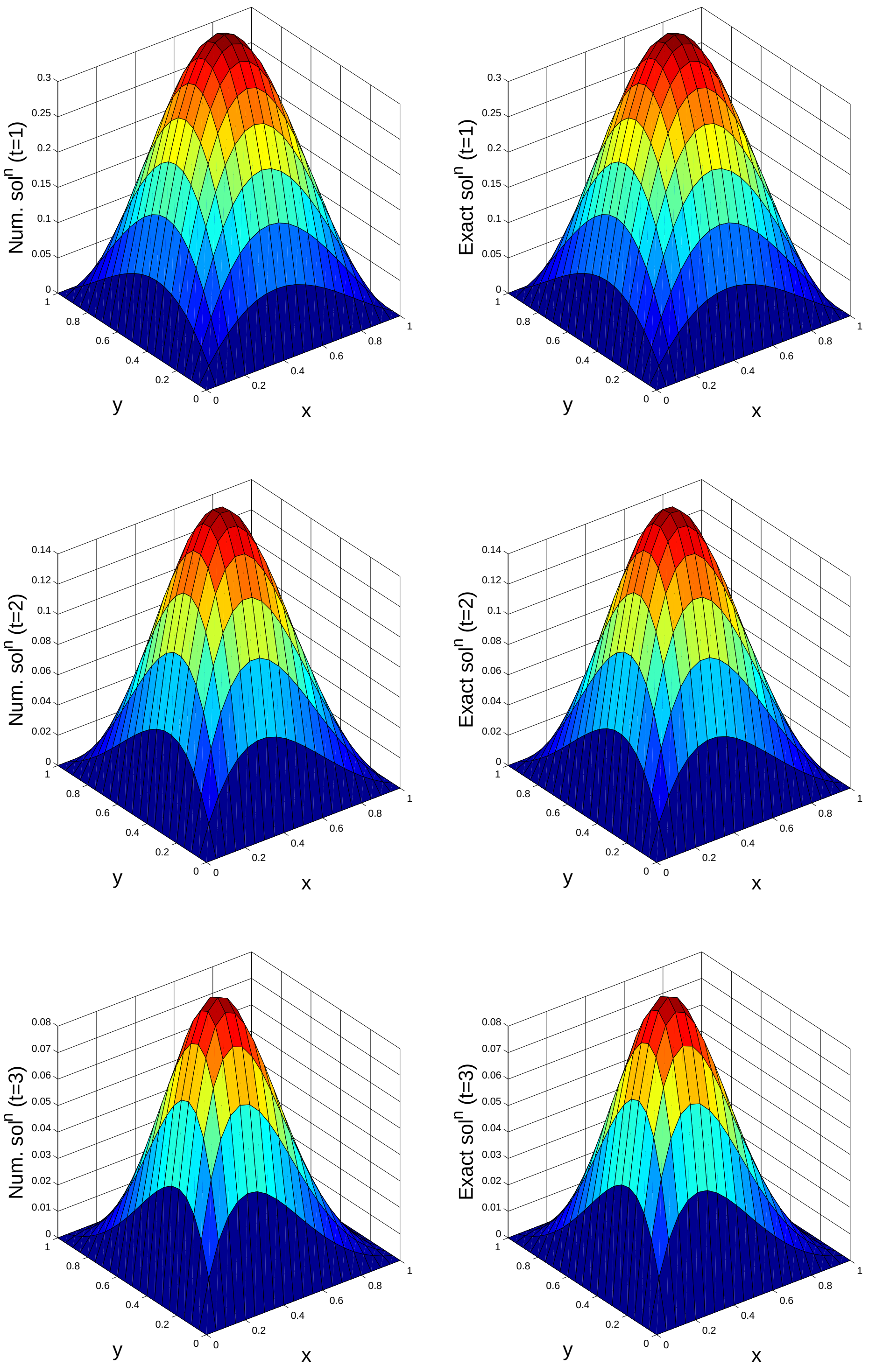


Figure 5.4: Plots of numerical and exact solutions of Example 5.3.

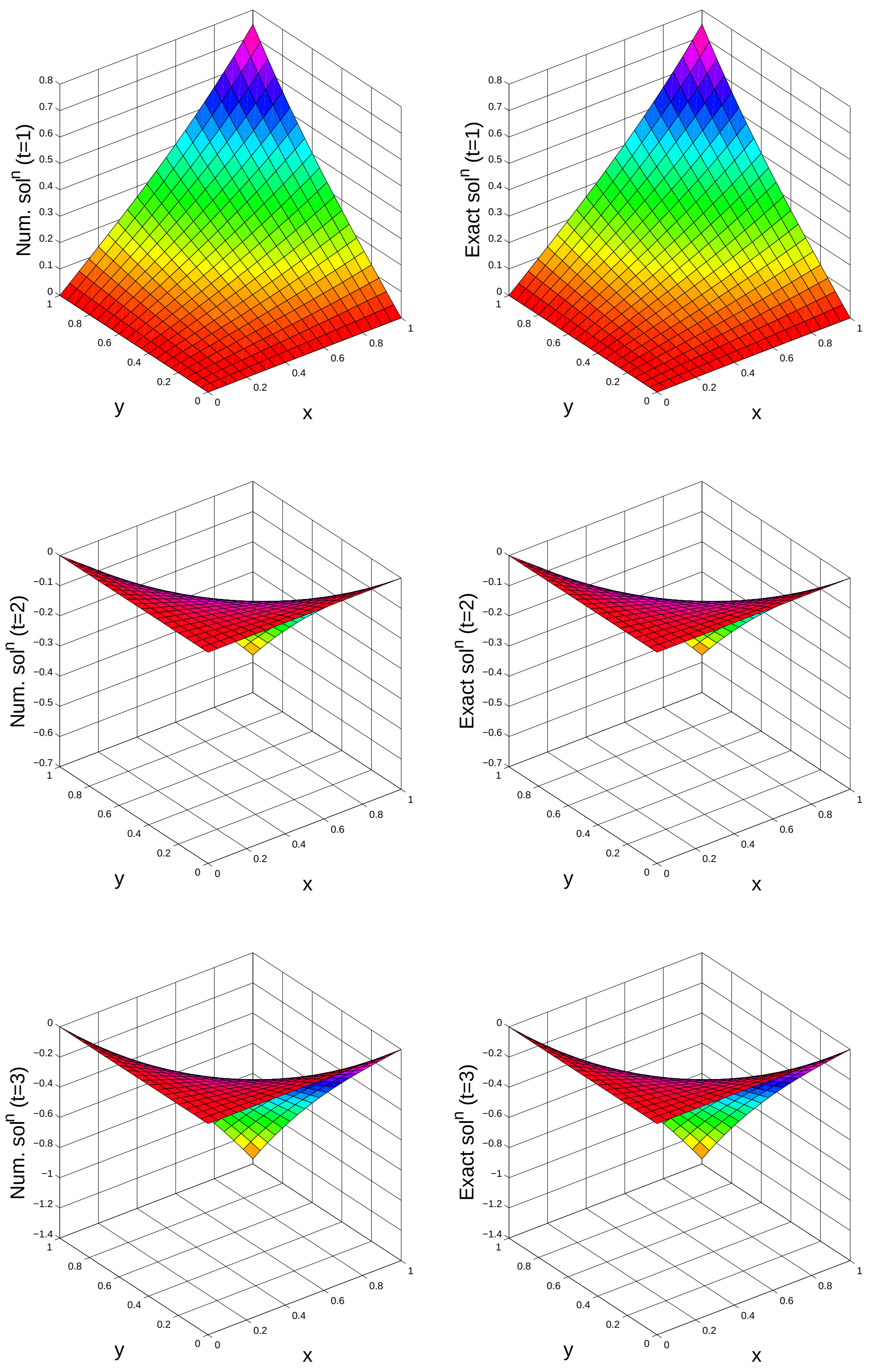


Figure 5.5: Plots of numerical and exact solutions of Example 5.4.

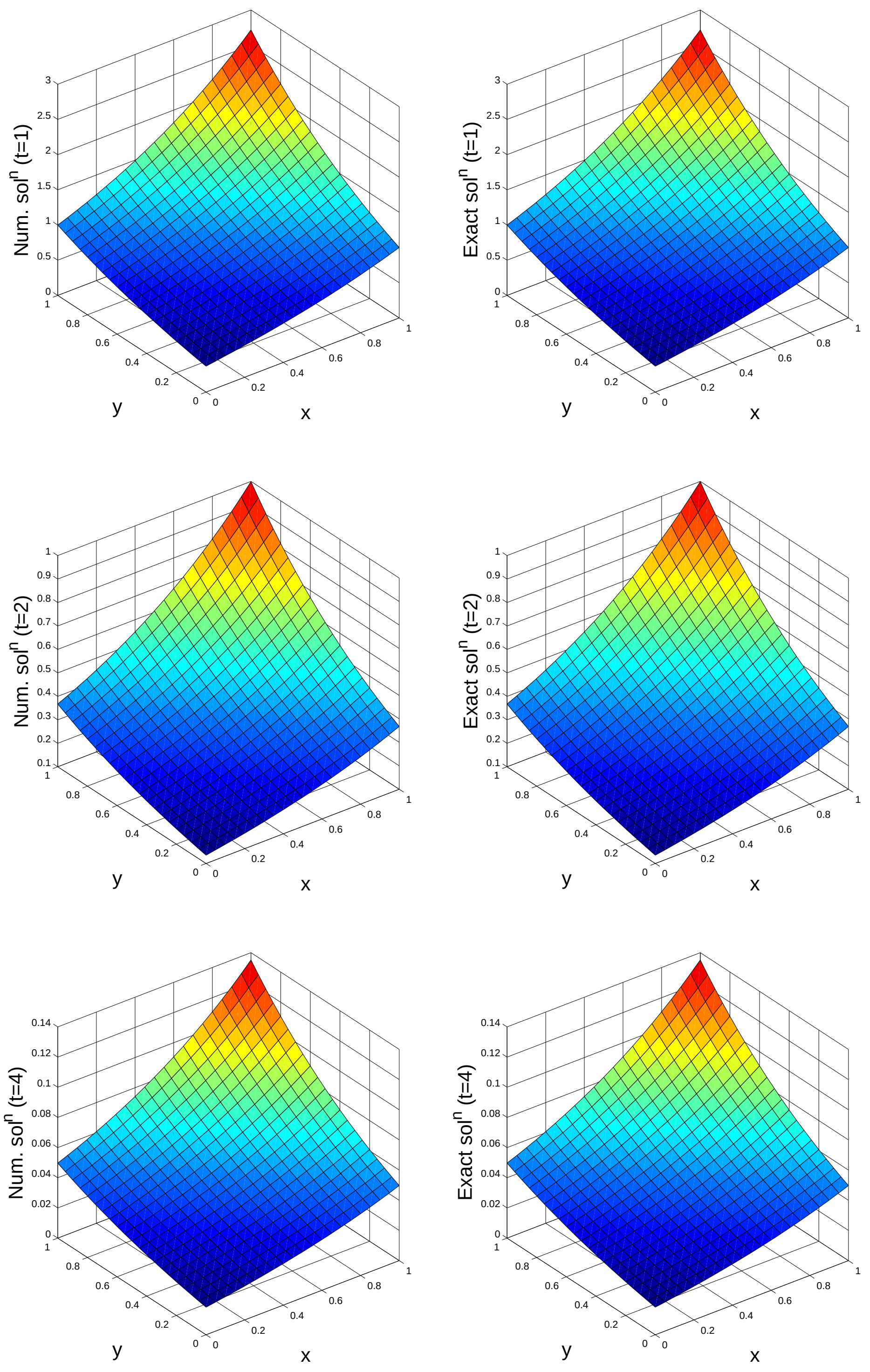


Figure 5.6: Plots of numerical and exact solutions of Example 5.5.

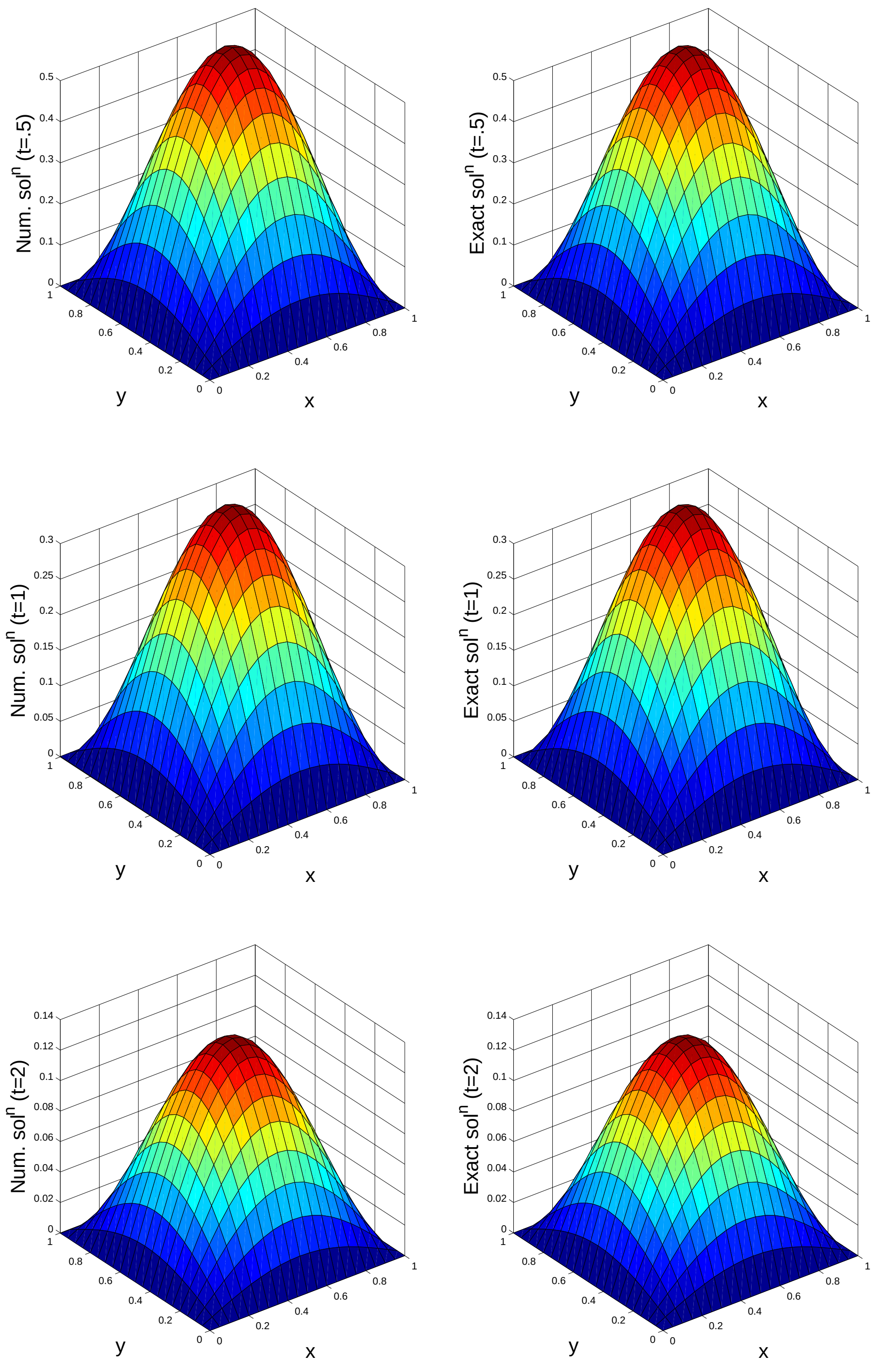


Figure 5.7: Plots of numerical and exact solutions of Example 5.6.

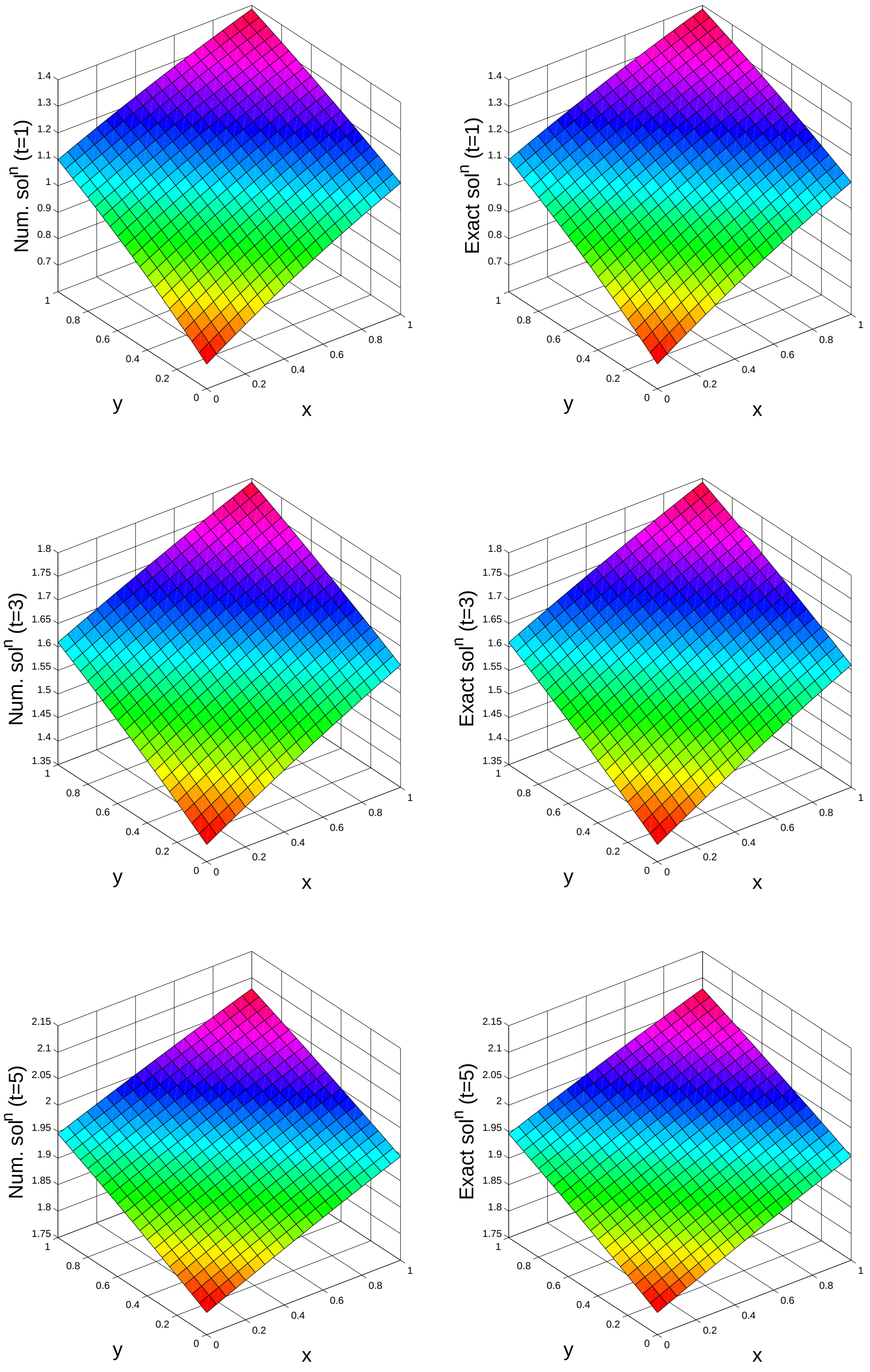


Figure 5.8: Plots of numerical and exact solutions of Example 5.7.

Chapter 6

Numerical Solutions of Some Nonlinear Wave Equations

6.1 Introduction

Wave equations arise in many physical and engineering applications such as continuum physics, mixed models of transonic flows, fluid dynamics and many other fields of science and engineering. These equations have been studied by various numerical techniques in past several decades. In this chapter, we are mainly concerned with the numerical solutions of one and two dimensional wave equations.

6.1.1 One Dimensional Nonlinear Wave Equations

We consider one dimensional nonlinear wave equations of the form

$$u_{tt} = u_{xx} + f(x,t,u,u_x,u_t),\ \ x \in (a,b),\ t > 0, \tag{6.1}$$

with the following initial and Dirichlet boundary conditions

$$u(x,0) = f_1(x),\ \ u_t(x,0) = f_2(x),\ \ x \in [a,b], \tag{6.2}$$

$$u(a,t) = g_1(t),\ \ u(b,t) = g_2(t),\ \ t \geq 0. \tag{6.3}$$

6.1.2 Two Dimensional Nonlinear Wave Equations

We also consider two dimensional nonlinear wave equations of the form

$$u_{tt} = u_{xx} + u_{yy} + f(x,y,t,u,u_x,u_y,u_t),\ \ (x,y,t) \in D \times (0,T], \tag{6.4}$$

with the following initial conditions

$$u(x,y,0) = u_0(x,y),\ \ u_t(x,y,0) = v_0(x,y),\quad (x,y) \in D, \tag{6.5}$$

and the following Dirichlet boundary conditions

$$\begin{aligned} &u(a,y,t) = h_1(y,t),\ \ u(b,y,t) = h_2(y,t),\\ &u(x,c,t) = h_3(x,t),\ \ u(x,d,t) = h_4(x,t),\quad (x,y,t) \in \partial D \times (0,T], \end{aligned} \tag{6.6}$$

where $D = \{(x,y) : a \le x \le b, c \le y \le d\}$, ∂D is its boundary and $(0,T]$ is the time interval.

Many authors have studied the numerical solutions of one dimensional linear wave equations by using various techniques. Some are unconditionally stable difference schemes [72, 134, 150], Chebyshev tau method [186], Quartic B-spline collocation method [64], dual reciprocity boundary integral function method [44], Bernoulli matrix method [199] etc. Ding and Zhang [59] and Rashidinia et al. [176] have solved the second order linear hyperbolic equation with mixed boundary conditions using parameter cubic spline method and non-polynomial cubic spline method, respectively.

Mohanty et al. [158] have proposed a higher order difference method for one dimensional nonlinear hyperbolic equations with variable coefficients. Implicit finite difference methods for quasilinear hyperbolic equations have been proposed by [154, 159]. A three level implicit nine point compact finite difference scheme for one dimensional nonlinear wave equations has been proposed by Mohanty and Gopal [153]. Mohanty and Singh [160] have proposed a Numerov type three-level implicit compact discretization scheme for nonlinear hyperbolic equations. The study of the solution of two dimensional nonlinear wave equations is also an attractive area of research in the literature. Jain et al. [82] have proposed an implicit finite difference method for two dimensional wave equations. Iyengar and Mittal [81] have proposed an unconditionally stable ADI scheme of $O(h^2+k^2)$ and $O(h^4+k^4)$ for two dimensional nonlinear wave equations with variable coefficients. Three level implicit conditionally stable difference scheme of order $O(k^4+h^2k^2+h^4)$ for two dimensional nonlinear wave equations with constant coefficients has been proposed by Mohanty et al. [157]. Mohanty and Jain [155] have proposed an unconditionally stable ADI scheme for two dimensional linear wave equations with constant coefficients and Mohanty [149] has presented an unconditionally stable three level implicit operator splitting method for two dimensional linear hyperbolic equation with variable coefficients. Dehghan and Mohebbi [48] have used the compact finite difference approximations of orders two and four, for discretizing the spatial derivatives and used collocation method for the time component. Dehghan and Mohebbi [49] have developed an unconditionally stable implicit collocation method for two dimensional linear hyperbolic equations. Dehghan and Shokri [55] have proposed a meshless method using collocation points with radial basis functions for two dimensional hyperbolic equations with variable coefficients. Jiwari et al. [91] and Kumar et al. [118] have solved two dimensional linear hyperbolic telegraph and two dimensional quasi-linear hyperbolic equations, respectively using polynomial differential quadrature method.

In this chapter, we have applied a differential quadrature method based on modified cubic B-spline basis functions for solving one and two dimensional wave equations with Dirichlet boundary conditions. The equations are converted into a system of partial differential equations and further discretized spatially by using DQM. Finally, we obtain a system of first order ODEs, which has been solved using SSP-RK43 [197] scheme. This chapter is organized as follows. Section-6.2 and Section-6.3 give details of execution of present method for solving equations (6.1) and (6.4), respectively. In Section-6.4, seven numerical test examples are considered to show the efficiency and accuracy of the proposed method, computationally. Conclusions are presented in Section-6.5, that briefly sum up the numerical outcomes.

6.2 Solution of One Dimensional Wave Equations

For one dimensional DQM, the domain $a \leq x \leq b$ is divided into a mesh of uniform length $h_x = x_{i+1} - x_i = \frac{b-a}{N-1}$ by the knot x_i, where $i = 1, 2, ., N-1$. Then the first and second order partial derivatives of the function $u(x,t)$ at any time at a knot $x = x_i$ can be approximated as follows:

$$u_x^{(1)}(x_i,t) = \sum_{j=1}^{N} a_{ij}^{(1)} u(x_j,t), \quad i = 1, 2, \ldots, N, \tag{6.7}$$

$$u_x^{(2)}(x_i,t) = \sum_{j=1}^{N} a_{ij}^{(2)} u(x_j,t), \quad i = 1, 2, \ldots, N, \tag{6.8}$$

where $a_{ij}^{(1)}$ and $a_{ij}^{(2)}$ are the weighting coefficients of first and second order derivatives of $u(x,t)$ at the point $x = x_i$. The weighting coefficients $a_{ij}^{(1)}$ are computed by using modified cubic B-spline basis functions (1.6) in the approximations (6.7) as stated in chapter 1 (see Section-1.9). To compute $a_{ij}^{(2)}$, we have used Shu's recurrence formulae (1.19).

To solve equation (6.1), we convert it into a system of partial differential equations using the transformation $u_t = v$, as

$$\begin{aligned} u_t &= v, \\ v_t &= u_{xx} + f(x,t,u,u_x,v). \end{aligned} \tag{6.9}$$

Now spatial derivatives of u at the grid points x_i, for $1 \leq i \leq N$ are discretized using approximations (6.7) and (6.8) and finally we get the following system of first order ordinary differential equations

$$\begin{aligned} \frac{du_i}{dt} &= v(x_i,t), \\ \frac{dv_i}{dt} &= \sum_{j=1}^{N} a_{ij}^{(2)} u(x_j,t) + f(x,t,u(x_i,t), \sum_{j=1}^{N} a_{ij}^{(1)} u(x_j,t), v(x_i,t)), \quad i = 1,\ldots,N, \end{aligned} \tag{6.10}$$

with the following initial conditions

$$u(x_i,0) = f_1(x_i), \; v(x_i,0) = f_2(x_i), \;\; i = 1,2,\ldots,N. \tag{6.11}$$

System (6.10) represents a system of $2N$ nonlinear first order ordinary differential equations. Once the initial vector $[\mathbf{u}_i^0, \mathbf{v}_i^0]^T$, $i = 1,\ldots,N$ is computed using initial conditions (6.11), we solve the system (6.10) with boundary conditions (6.3) using SSP-RK43 scheme and consequently the approximate solution $\left[\mathbf{u}_i^k, \mathbf{v}_i^k\right]^T$ is computed at time level $t = t_k$.

6.3 Solution of Two Dimensional Wave Equations

For two dimensional DQM, the region $a \leq x \leq b$, $c \leq y \leq d$ is discretized by taking N and M grid points in x and y directions such that $h_x = x_{i+1} - x_i$ and $h_y = y_{j+1} - y_j$. Then the first and second order partial derivatives of the function $u(x,y,t)$ with respect to x at a point (x_i, y_j) can be approximated as follows:

$$u_x^{(1)}(x_i,y_j,t) = \sum_{k=1}^{N} a_{ik}^{(1)} u(x_k,y_j,t), \quad i = 1,2,\ldots,N; \; j = 1,2,\ldots,M, \tag{6.12}$$

$$u_x^{(2)}(x_i, y_j, t) = \sum_{k=1}^{N} a_{ik}^{(2)} u(x_k, y_j, t), \quad i = 1, 2, \ldots, N; \; j = 1, 2, \ldots, M. \tag{6.13}$$

Similarly, the partial derivatives of the function $u(x, y, t)$ with respect to y at a point (x_i, y_j) are approximated as follows:

$$u_y^{(1)}(x_i, y_j, t) = \sum_{k=1}^{M} b_{jk}^{(1)} u(x_i, y_k, t), \quad i = 1, 2, \ldots, N; \; j = 1, 2, \ldots, M, \tag{6.14}$$

$$u_y^{(2)}(x_i, y_j, t) = \sum_{k=1}^{M} b_{jk}^{(2)} u(x_i, y_k, t), \quad i = 1, 2, \ldots, N; \; j = 1, 2, \ldots, M, \tag{6.15}$$

where $a_{ik}^{(1)}$, $a_{ik}^{(2)}$, $b_{jk}^{(1)}$ and $b_{jk}^{(2)}$ represent the weighting coefficients, which are computed using modified cubic B-spline basis functions in approximations (6.12) and (6.14) (see chapter 1, Section-1.9.1.2) and using Shu's recurrence formulae (1.27) and (1.28).

Now to solve equation (6.4), first we convert it into the following coupled system using the transformation $u_t = v$, as

$$\begin{aligned} u_t &= v, \\ v_t &= u_{xx} + u_{yy} + f(x, y, t, u, u_x, u_y, v). \end{aligned} \tag{6.16}$$

The first and second order spatial derivatives of u are discretized using approximations (6.12-6.15) and finally system of equations (6.16) reduces into the following system of nonlinear first order ordinary differential equations

$$\begin{aligned} \frac{du_{i,j}}{dt} &= v(x_i, y_j, t), \\ \frac{dv_{i,j}}{dt} &= \sum_{k=1}^{N} a_{ik}^{(2)} u(x_k, y_j, t) + \sum_{k=1}^{M} b_{jk}^{(2)} u(x_i, y_k, t) + f(x_i, y_j, t, u(x_i, y_j, t), \sum_{k=1}^{N} a_{ik}^{(1)} u(x_k, y_j, t), \\ &\qquad \sum_{k=1}^{M} b_{jk}^{(1)} u(x_i, y_k, t), v(x_i, y_j, t)), \\ &\qquad (x_i, y_j, t) \in D \times (0, T], \; i = 1, 2, \ldots, N; \; j = 1, 2, \ldots, M. \end{aligned} \tag{6.17}$$

Initial vector is computed using the following initial conditions

$$\begin{aligned} u(x_i, y_j, 0) &= u_0(x_i, y_j), \\ v(x_i, y_j, 0) &= v_0(x_i, y_j), \; i = 1, 2, \ldots, N; \; j = 1, 2, \ldots, M. \end{aligned} \tag{6.18}$$

The system of first order nonlinear ordinary differential equations (6.17) with the boundary conditions (6.6) have been solved using SSP-RK43 scheme and consequently the approximate solution $[u_{i,j}^k, v_{i,j}^k]$ is computed at time level $t = t_k$.

6.4 Numerical Results

The proposed scheme is applied on some linear and nonlinear wave equations and accuracy of the scheme is tested by computing L_2 error, maximum absolute error (MAE) and root mean square error (RMSE).

Example 6.1

We consider Vander Pol type nonlinear wave equation

$$u_{tt} = u_{xx} + \gamma(u^2 - 1)u_t + \left(\pi^2 + \gamma^2 e^{-2\gamma t}\sin^2(\pi x)\right) e^{-\gamma t}\sin(\pi x), \;\; x \in (0,1) \;\; t > 0. \quad (6.19)$$

The initial and boundary conditions are given by

$$u(x,0) = \sin(\pi x), \;\; u_t(x,0) = -\gamma\sin(\pi x), \;\; x \in (0,1), \quad (6.20)$$

$$u(0,t) = 0, \quad u(1,t) = 0, \;\; t \geq 0. \quad (6.21)$$

The exact solution is given in [153] as

$$u(x,t) = e^{-\gamma t}\sin(\pi x). \quad (6.22)$$

The maximum absolute errors and CPU time are tabulated in Table-6.1 for $\gamma = 1, 2, 3$ at $t = 2$ with $h = .05$ and for $\Delta t = .01$, .001, .0001. It is observed that the results are close to the exact solutions. Table-6.2 compares the maximum absolute errors for various values of h with numerical method of order $(k^2 + h^2)$ given in Mohanty and Gopal [153]. It can be seen from this table that the errors given by our method are better than those given by the method of [153]. It has been also noticed that by decreasing the space and time steps the errors get improved. A comparison of exact and approximate solutions at $t = 2$ for $\gamma = 1, 2, 3$ is depicted in Figure-6.1.

Example 6.2

In this example we consider Dissipative nonlinear wave equation

$$u_{tt} = u_{xx} - 2uu_t + \big(\pi^2 - 1 - 2\sin(\pi x)\sin t\big)\sin(\pi x)\cos t,\ x \in (0,1),\ \ t > 0. \tag{6.23}$$

The initial and boundary conditions are given by

$$u(x,0) = \sin(\pi x),\ \ u_t(x,0) = 0,\ x \in (0,1), \tag{6.24}$$

$$u(0,t) = 0, \qquad\qquad u(1,t) = 0,\ \ t \geq 0. \tag{6.25}$$

The exact solution is given in [153] as

$$u(x,t) = \sin(\pi x)\cos t. \tag{6.26}$$

Table-6.3 presents the numerical results at different time t with $h = .05$ and $\Delta t = .01, .001$. In Table-6.4 we report the maximum absolute errors for various values of h and compare them with numerical method of order $(k^2 + h^2)$ given in Mohanty and Gopal [153]. Our results are found to be better in comparison with that in [153]. It has been also noticed from tables that as we decrease the space and time steps the approximate solution approaches to the exact one. In Figure-6.2, the comparisons of approximate and exact solutions are depicted at $t = 1,\ 2\ ,3$ with $h = .05$ and $\Delta t = .001$.

Example 6.3

Now we consider following nonlinear wave equation in the domain $(0,1)$

$$u_{tt} = u_{xx} + \gamma u(u_x + u_t) + (x^2\sinh t - 2\sinh t - \gamma x^2 \sinh t(2x\sinh t + x^2\cosh t)). \tag{6.27}$$

The initial and boundary conditions are given by

$$u(x,0) = 0,\ \ u_t(x,0) = x^2,\ x \in (0,1), \tag{6.28}$$

$$u(0,t) = 0, \quad\ u(1,t) = \sinh t,\ \ t \geq 0. \tag{6.29}$$

The exact solution is given in [154] as

$$u(x,t) = x^2\sinh t. \tag{6.30}$$

Table-6.5 reports the maximum absolute errors for $\gamma = 1, 5$ and 10 with $h = .05$ and $\Delta t = .01,\ .001,\ .0001$ respectively. From table it may be seen that our results are in good agreement with exact solutions. Convergence study is also carried out in Table-6.6 with different space step sizes and $\Delta t = .0001$. From this table it is observed that the difference between the exact and numerical solutions decreases as the mesh size is reduced. Figure-6.3 depicts the comparison of exact and numerical solution at $t = 1$ with $h = .05$, $\gamma = 1$ and $\Delta t = .001$.

Example 6.4

In this example we consider the one dimensional telegraph equation

$$u_{tt} + 2\alpha u_t + \beta^2 u = u_{xx} + (2 - 2\alpha + \beta^2)e^{-t} \sin x, \;\; x \in (0, \pi), \;\; t > 0. \tag{6.31}$$

The initial and boundary conditions are given by

$$u(x,0) = \sin x, \;\; u_t(x,0) = -\sin x, \;\; x \in (0,\pi), \tag{6.32}$$

$$u(0,t) = 0, \qquad\qquad u(\pi,t) = 0, \;\; t \geq 0. \tag{6.33}$$

The exact solution is given in [54] as

$$u(x,t) = e^{-t} \sin x. \tag{6.34}$$

This example has been solved for $\alpha = 2$ and $\beta = \sqrt{2}$. Table-6.7 shows the L_2 and L_∞ errors with $\Delta t = .01$ and $h = .06$, $h = .04$, respectively. In Table-6.8 results are presented for more finer space step size $h = .02$ and $\Delta t = .0001$ and compared with those given in Dehghan and Shokri [54]. Table-6.7 and Table-6.8, indicate that our scheme gives better numerical results while using less number of grid points in comparison to [54]. CPU time is also very less in our computations. Further, the absolute errors are computed with $h = \frac{\pi}{30}$ and $\Delta t = .1$. Table-6.9 compares absolute errors with those of Gao and Chi [72]. It may be noticed that numerical solutions obtained by present method are accurate than in [72]. Comparisons of approximate and exact solutions at $t = 1, 2, 3$ with $h = .04$ and $\Delta t = .01$ are presented in Figure-6.4.

Example 6.5

We consider the following one dimensional hyperbolic telegraph equation

$$u_{tt} + 2\alpha u_t + \beta^2 u = u_{xx} + \alpha\left(1 + \tan^2\left(\frac{x+t}{2}\right)\right) + \beta^2 \tan\left(\frac{x+t}{2}\right),\ x \in (0,2),\ t > 0. \tag{6.35}$$

The initial and boundary conditions are given by

$$u(x,0) = \tan\left(\frac{x}{2}\right),\ u_t(x,0) = \frac{1}{2}\left(1 + \tan^2\left(\frac{x}{2}\right)\right),\ x \in (0,2), \tag{6.36}$$

$$u(0,t) = \tan\left(\frac{t}{2}\right),\quad u(2,t) = \tan\left(\frac{2+t}{2}\right),\ t \geq 0. \tag{6.37}$$

The exact solution is given in [64] as

$$u(x,t) = \tan\left(\frac{x+t}{2}\right). \tag{6.38}$$

For this example we take $\alpha = 10$, $\beta = 5$. To measure the accuracy of proposed method we compute L_2 and L_∞ error norms with $h = .04$, $\Delta t = .0001$, which are presented in Table-6.10. It has been noticed that our results are in excellent agreement with the exact solutions. A comparison of our results with those of Dosti and Nazemi [64] has been reported in Table-6.11 with $h = .04$, $\Delta t = .001$. We noticed that our results are compatible with [64], but grid points used in our computation are very less. CPU time is also reported in Table-6.10 and 6.11. A graph comparing the exact and approximate solutions at time $t = .2, .4, .6, .8, 1$ is presented in Figure-6.5.

Example 6.6

We consider Vander Pol type two dimensional nonlinear wave equation

$$u_{tt} = u_{xx} + u_{yy} + \gamma(u^2 - 1)u_t + (2\pi^2 + \gamma^2 e^{-2\gamma t} \sin^2 \pi x \sin^2 \pi y)e^{-\gamma t} \sin \pi x \sin \pi y,$$
$$0 \leq x, y \leq 1,\ t > 0, \tag{6.39}$$

with the following initial conditions

$$\begin{aligned} u(x,y,0) &= \sin \pi x\ \sin \pi y, \\ u_t(x,y,0) &= -\gamma \sin \pi x\ \sin \pi y. \end{aligned} \tag{6.40}$$

The Dirichlet boundary conditions are given by

$$\begin{aligned} u(0,y,t) &= 0, \;\; 0 \le y \le 1, \\ u(1,y,t) &= 0, \;\; 0 \le y \le 1, \\ u(x,0,t) &= 0, \;\; 0 \le x \le 1, \\ u(x,1,t) &= 0, \;\; 0 \le x \le 1. \end{aligned} \tag{6.41}$$

The exact solution is given in [157] as

$$u(x,y,t) = e^{-\gamma t} \sin \pi x \, \sin \pi y. \tag{6.42}$$

The results of this example are given in Table-6.12 for $\gamma = 1, 2$ and 3 with $\Delta t = .001$ and $h_x = h_y = .1, .05$, respectively and are found close to the exact solutions. In Table-6.13 we report the comparison of maximum absolute errors with those given in Mohanty et al. [157] for $\gamma = .01$ and $\gamma = .001$. Order of convergence is also computed at $t = 1$. The physical behaviour of numerical and exact solutions at $t = 1$ and $t = 3$ is presented in Figure-6.6. From the tables and figures we conclude that the obtained results show very good accuracy and efficiency of proposed scheme.

Example 6.7

We consider two dimensional Dissipative nonlinear wave equation

$$u_{tt} = u_{xx}+u_{yy}-2uu_t+(2\pi^2-1+2\sin \pi x \sin \pi y \cos t) \sin \pi x \sin \pi y \sin t, \;\; 0 \le x, y \le 1, \;\; t > 0, \tag{6.43}$$

with the following initial and boundary conditions

$$\begin{aligned} u(x,y,0) &= 0, \\ u_t(x,y,0) &= \sin \pi x \, \sin \pi y, \end{aligned} \tag{6.44}$$

$$\begin{aligned} u(0,y,t) &= 0, \;\; 0 \le y \le 1, \\ u(1,y,t) &= 0, \;\; 0 \le y \le 1, \\ u(x,0,t) &= 0, \;\; 0 \le x \le 1, \\ u(x,1,t) &= 0, \;\; 0 \le x \le 1. \end{aligned} \tag{6.45}$$

The exact solution is given in [157] as

$$u(x, y, t) = \sin \pi x \, \sin \pi y \, \sin t. \tag{6.46}$$

The maximum absolute errors, root mean square errors and CPU time are computed up to time $t = 10$ with grid sizes $h_x = h_y = .1, .05$ and $\Delta t = .001$ and reported in Table-6.14. It has been observed that the obtained numerical solutions agree well with the exact solutions. Table-6.15 shows the maximum absolute errors for different grid sizes and order of convergence at $t = 1$. We compare obtained errors with those given in Mohanty et al. [157] and found them better. Finally, in Figure-6.7 exact and numerical solutions are plotted at $t = 1$ and $t = 3$.

6.5 Conclusions

1. In this chapter modified cubic B-spline differential quadrature method has been successfully applied to find the solutions of some linear and nonlinear partial differential equations such as the 1D Vander Pol equation, 1D Dissipative nonlinear wave equation, 2D Vander Pol equation, 2D Dissipative nonlinear wave equation and some linear telegraph equations.

2. The obtained numerical results are found to be very good in comparison with the existing solutions in the literature. From the numerical results given in Table-6.7, Table-6.8 and Table-6.11, we conclude that the scheme gives more accurate results than earlier works with smaller grid points, larger time steps and with less computational cost.

3. The main advantage of this scheme is that it does not use any transformation or linearization process to solve nonlinear equations.

4. The scheme is easy and very suitable for computer implementation.

Table 6.1: MAE of Example 6.1 at $t = 2$ with $h = .05$.

Δt	$\gamma = 1$	$\gamma = 2$	$\gamma = 3$	CPU time(s)
.01	1.650E-3	1.834E-3	1.444E-3	.02
.001	1.994E-4	1.962E-4	1.483E-4	.13
.0001	5.697E-5	3.683E-5	2.188E-5	1.23

Table 6.2: MAE of Example 6.1 at $t = 2$ with $\Delta t = .0001$.

Proposed method					Mohanty and Gopal [153]		
h	$\gamma = 1$	$\gamma = 2$	$\gamma = 3$	CPU time(s)	$\gamma = 1$	$\gamma = 2$	$\gamma = 3$
$\frac{1}{8}$	7.633E-4	3.484E-3	1.445E-4	1.05	3.390E-3	1.580E-3	6.629E-4
$\frac{1}{16}$	1.023E-4	5.640E-5	3.007E-5	1.10	8.434E-4	3.986E-4	1.662E-4
$\frac{1}{32}$	2.578E-5	2.246E-5	1.598E-5	1.30	2.103E-4	9.976E-5	4.156E-5
$\frac{1}{64}$	1.718E-5	1.852E-5	1.439E-5	2.16	5.254E-5	2.494E-5	1.039E-5

Table 6.3: Errors of Example 6.2 with $h = .05$.

t	$\Delta t = .01$			$\Delta t = .001$		
	L_2	L_∞	CPU time(s)	L_2	L_∞	CPU time(s)
1	3.046E-3	4.274E-3	.02	4.613E-4	6.086E-4	.07
2	3.251E-3	4.625E-3	.02	2.117E-4	3.339E-4	.14
3	5.737E-5	9.782E-5	.03	8.708E-5	1.120E-4	.21

Table 6.4: MAE of Example 6.2 with $\Delta t = .0001$.

Proposed method					Mohanty and Gopal [153]	
h	$t = 1$	CPU time(s)	$t = 2$	CPU time(s)	$t = 1$	$t = 2$
$\frac{1}{8}$	3.259E-3	.54	2.465E-3	1.1	1.741E-2	1.323E-2
$\frac{1}{16}$	4.296E-4	.57	2.826E-4	1.2	4.366E-3	3.374E-3
$\frac{1}{32}$	8.866E-5	.69	2.788E-5	1.4	1.092E-3	8.476E-3
$\frac{1}{64}$	4.689E-5	1.1	4.301E-5	2.3	2.731E-4	2.121E-4

Table 6.5: MAE of Example 6.3 at $t = 1$ with $h = .05$.

Δt	$\gamma = 1$	$\gamma = 5$	$\gamma = 10$
.01	6.769E-3	7.080E-3	1.095E-2
.001	4.906E-4	5.188E-4	9.601E-4
.0001	1.788E-4	4.540E-4	1.456E-3

Table 6.6: MAE of Example 6.3 at $t = 1$ with $\Delta t = .0001$.

h	$\gamma = 1$	$\gamma = 5$	$\gamma = 10$	CPU time(s)
$\frac{1}{8}$	1.404E-3	3.572E-3	1.174E-2	.67
$\frac{1}{16}$	3.041E-4	7.604E-4	2.427E-3	.71
$\frac{1}{32}$	4.782E-5	1.415E-4	4.861E-4	.77
$\frac{1}{64}$	5.573E-5	6.405E-5	1.076E-4	1.23

Table 6.7: Errors of Example 6.4 at different time levels with $\Delta t = .01$.

t	$h = .06$			$h = .04$		
	L_2	L_∞	CPU time(s)	L_2	L_∞	CPU time(s)
.5	3.820E-6	5.153E-6	.036	1.142E-7	1.550E-7	.07
1.0	2.989E-6	3.183E-6	.039	8.872E-7	9.486E-7	.07
1.5	2.126E-6	1.952E-6	.041	6.278E-7	5.796E-7	.08
2.0	1.542E-6	1.195E-6	.043	4.535E-7	3.537E-7	.10
2.5	1.135E-6	7.236E-7	.047	3.333E-7	2.136E-7	.11
3.0	8.350E-7	4.985E-7	.054	2.449E-7	1.452E-7	.12

Table 6.8: L_2 and L_∞ errors of Example 6.4 with $h = .02$, $\Delta t = .0001$.

	Proposed method			Dehghan and Shokri [54]		
t	L_2	L_∞	CPU time(s)	L_2	L_∞	CPU time(s)
.5	1.416E-7	1.920E-7	1.2	7.949E-5	8.372E-6	5
1.0	1.099E-7	1.171E-7	2.0	1.455E-4	1.568E-5	12
1.5	7.797E-8	7.129E-8	2.7	1.590E-4	1.741E-5	19
2.0	5.644E-8	4.337E-8	3.5	1.418E-4	1.581E-5	28
2.5	4.176E-8	2.638E-8	4.1	-	-	-
3.0	3.068E-8	1.827E-8	4.9	-	-	-

Table 6.9: Comparison of absolute errors of Example 6.4 with $h = \frac{\pi}{30}$ and $\Delta t = .1$.

x	$t = 1$		$t = 2$	
	Present method	Gao and Chi [72]	Present method	Gao and Chi [72]
h	1.546E-5	9.04E-6	5.601E-6	8.84E-6
$8h$	1.230E-6	6.479E-5	2.410E-7	6.337E-5
$15h$	8.203E-6	8.928E-5	2.098E-6	8.731E-5
$22h$	1.230E-6	7.069E-5	2.410E-7	6.913E-5
$29h$	1.546E-5	9.04E-6	5.601E-6	8.84E-6

Table 6.10: L_2 and L_∞ errors of Example 6.5 with $h = .04$ and $\Delta t = .0001$.

t	L_2	L_∞	CPU time(s)
.2	2.5347E-5	8.1533E-5	.21
.4	6.1875E-5	2.1482E-4	.39
.6	1.6704E-4	6.6819E-4	.56
.8	6.9870E-4	3.0221E-3	.74
1.0	8.4574E-3	3.9648E-2	.91

Table 6.11: L_2 and L_∞ errors of Example 6.5 with $\Delta t = .001$.

t	Proposed method			Dosti and Nazemi [64]
	$h = .04$			$h = .001$
	L_2	L_∞	CPU time(s)	L_∞
.2	3.1922E-4	8.3002E-4	.05	2.774E-4
.4	6.4498E-4	1.2832E-3	.07	7.0782E-4
.6	1.1111E-3	2.0374E-3	.09	1.3848E-3
.8	1.9570E-3	3.5872E-3	.11	3.0930E-3
1.0	4.4368E-3	1.1975E-2	.13	1.3424E-2

Table 6.12: Errors and CPU time of Example 6.6 at different time.

t	$N = M = 11$, $\Delta t = .001$			$N = M = 21$, $\Delta t = .001$		
	MAE	RMSE	CPU time	MAE	RMSE	CPU time(s)
	$\gamma = 1$					
1	8.8267E-4	5.0855E-4	.24	2.4756E-4	1.4750E-4	.45
2	1.0663E-3	4.5911E-4	.45	3.6623E-4	1.7121E-4	.78
3	2.8078E-4	1.1910E-4	.64	7.8208E-5	3.7667E-5	1.1
5	2.7463E-4	8.8053E-5	.93	7.6234E-5	3.1852E-5	1.8
	$\gamma = 2$					
1	4.2033E-4	2.3093E-4	.19	1.3672E-4	7.2942E-5	.38
2	4.0214E-4	1.8060E-4	.36	2.0909E-4	9.8179E-5	.73
3	9.5439E-5	4.4826E-5	.53	4.6129E-5	2.1882E-5	1.1
5	2.5087E-5	8.7846E-5	.87	1.0318E-5	4.6889E-6	1.8
	$\gamma = 3$					
1	2.0988E-4	1.0927E-4	.18	5.3468E-5	2.5444E-5	.37
2	1.6754E-4	7.5001E-5	.36	1.0867E-4	5.0905E-5	.73
3	3.4819E-5	1.6319E-5	.53	2.1649E-5	1.0096E-5	1.1
5	1.9699E-6	7.7289E-7	.87	1.1101E-6	3.1852E-7	1.8

Table 6.13: MAE of Example 6.6 with $\Delta t = .001$.

	Present scheme			Mohanty et al. [157]	
h	$t = 0.5$	$t = 1$	Order of Conv.(at $t = 1$)	$t = 0.5$	$t = 1$
	$\gamma = .01$				
.25	2.733E-2	2.527E-2		5.946E-2	3.547E-2
.125	3.668E-3	3.426E-3	2.8828	1.591E-2	6.722E-3
.0625	5.196E-4	4.712E-4	2.8621	4.108E-3	1.426E-3
	$\gamma = .001$				
.25	2.736E-2	2.544E-2		5.951E-2	3.550E-2
.125	3.669E-3	3.448E-3	2.8832	1.592E-2	6.723E-3
.0625	5.135E-4	4.732E-4	2.8652	4.113E-3	1.462E-3

Table 6.14: Errors and CPU time of Example 6.7 with $\Delta t = .001$.

t	$N = M = 11$			$N = M = 21$		
	MAE	RMSE	CPU time(s)	MAE	RMSE	CPU time(s)
1	8.9440E-4	5.5389E-4	.29	2.5162E-4	9.6896E-5	.46
2	1.0571E-3	7.2458E-4	.49	1.6671E-4	1.1439E-4	.86
3	7.1048E-4	3.7267E-4	.69	6.0546E-4	2.9335E-4	1.2
5	1.3813E-3	8.9369E-4	1.1	5.5131E-4	2.9705E-4	2.1
7	7.4138E-4	5.0181E-4	1.5	9.8372E-5	6.8085E-5	2.8
10	4.8445E-4	2.6003E-4	2.1	4.6837E-5	2.0792E-5	3.9

Table 6.15: MAE of Example 6.7 with $\Delta t = .001$.

h	Present scheme			Mohanty et al. [157]	
	$t = 0.5$	$t = 1$	Order of Conv.(at $t = 1$)	$t = 0.5$	$t = 1$
.25	1.797E-2	4.903E-3		5.763E-2	6.217E-2
.125	3.080E-3	7.637E-4	2.6825	1.542E-2	1.373E-2
.0625	1.255E-4	1.423E-4	2.4240	3.972E-3	3.269E-3

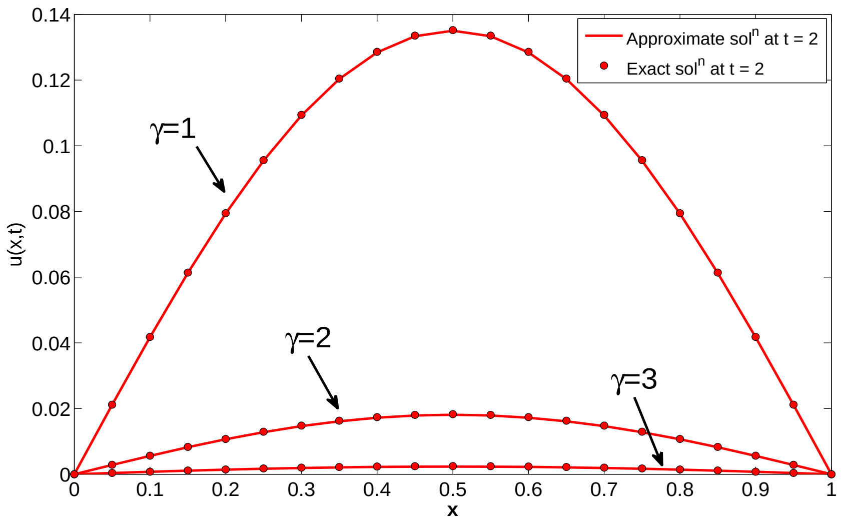


Figure 6.1: Approximate and exact solutions of Example 6.1.

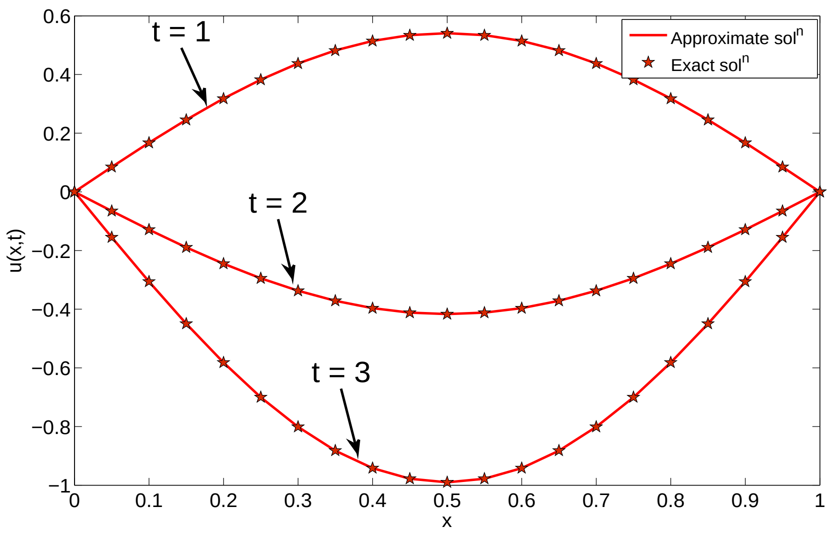


Figure 6.2: Approximate and exact solutions of Example 6.2.

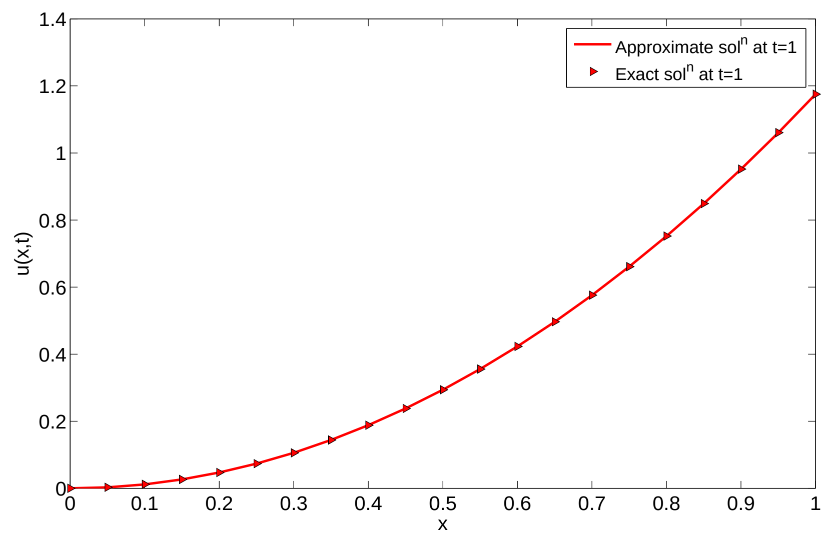


Figure 6.3: Approximate and exact solutions of Example 6.3.

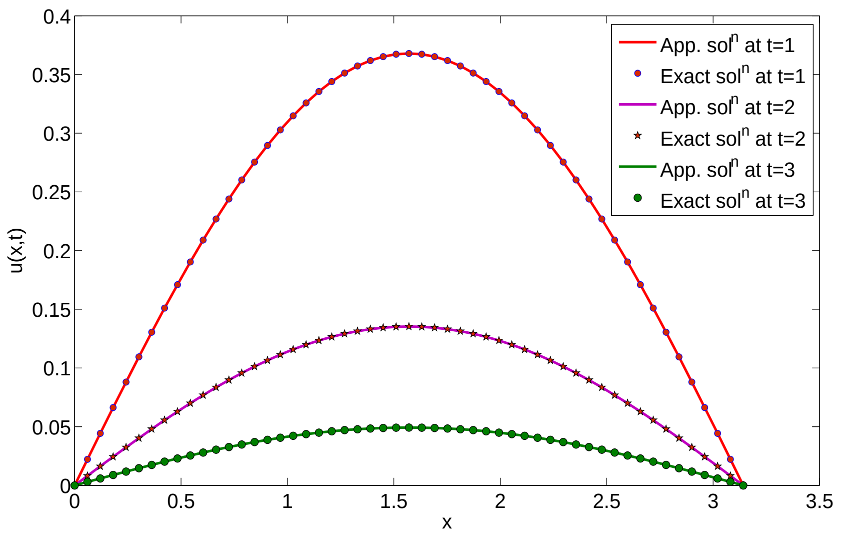


Figure 6.4: Approximate and exact solutions of Example 6.4.

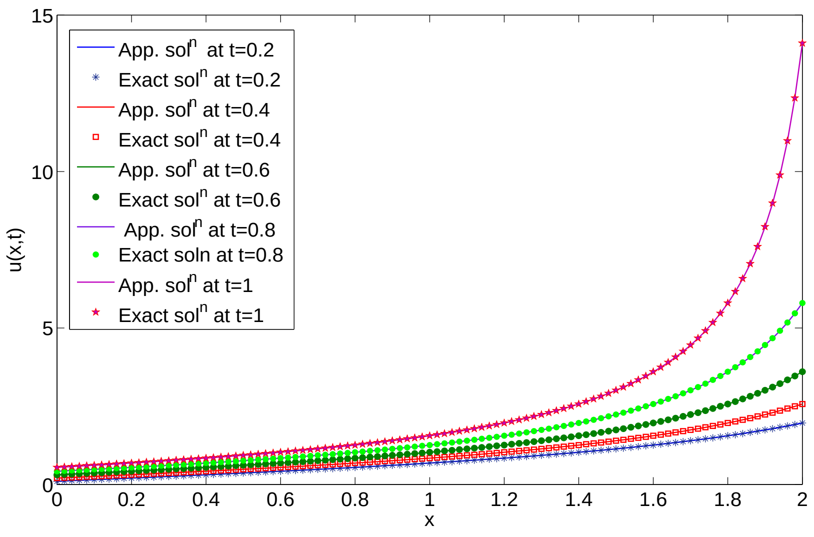


Figure 6.5: Approximate and exact solutions of Example 6.5.

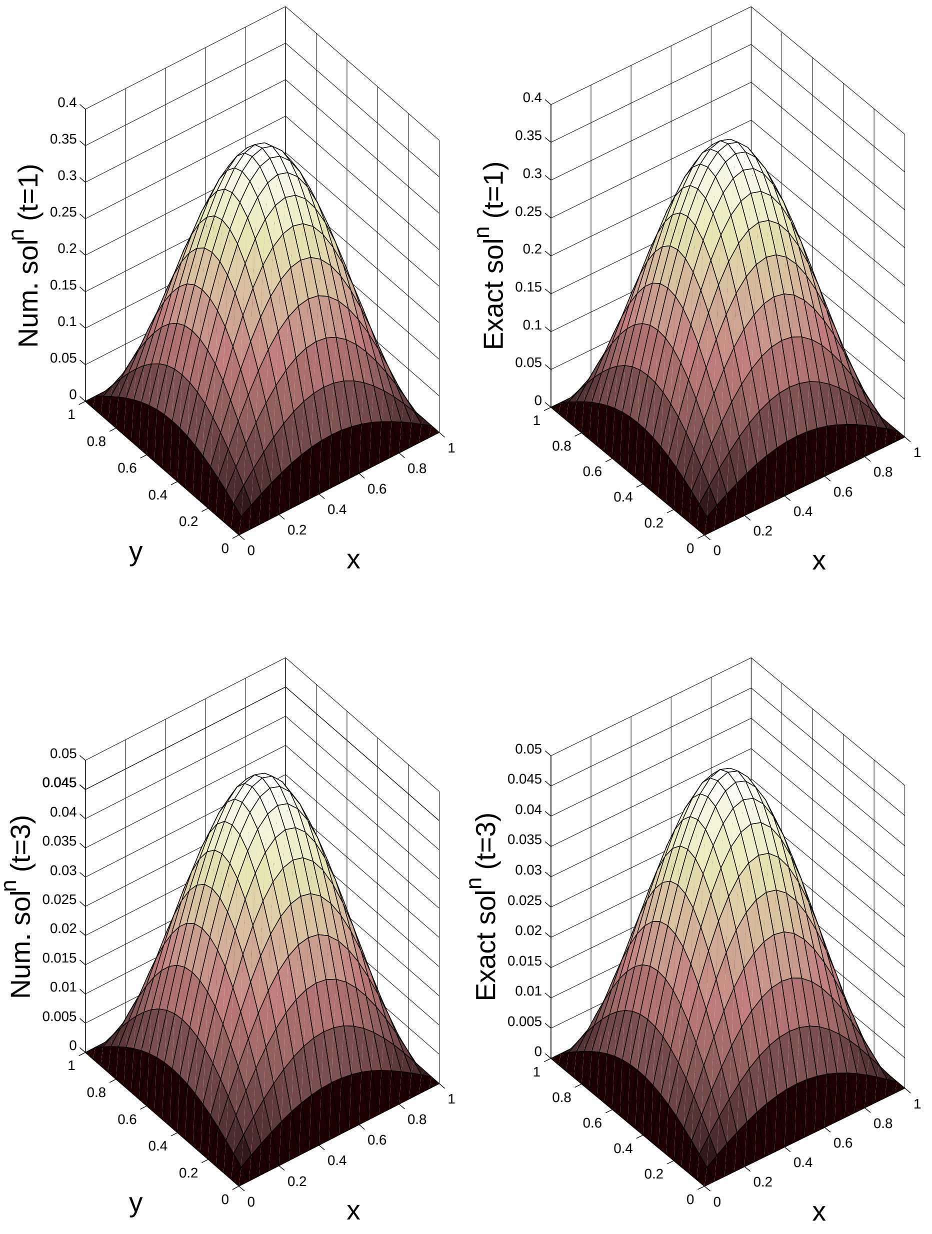


Figure 6.6: Surface plots of numerical and exact solutions of Example 6.6.

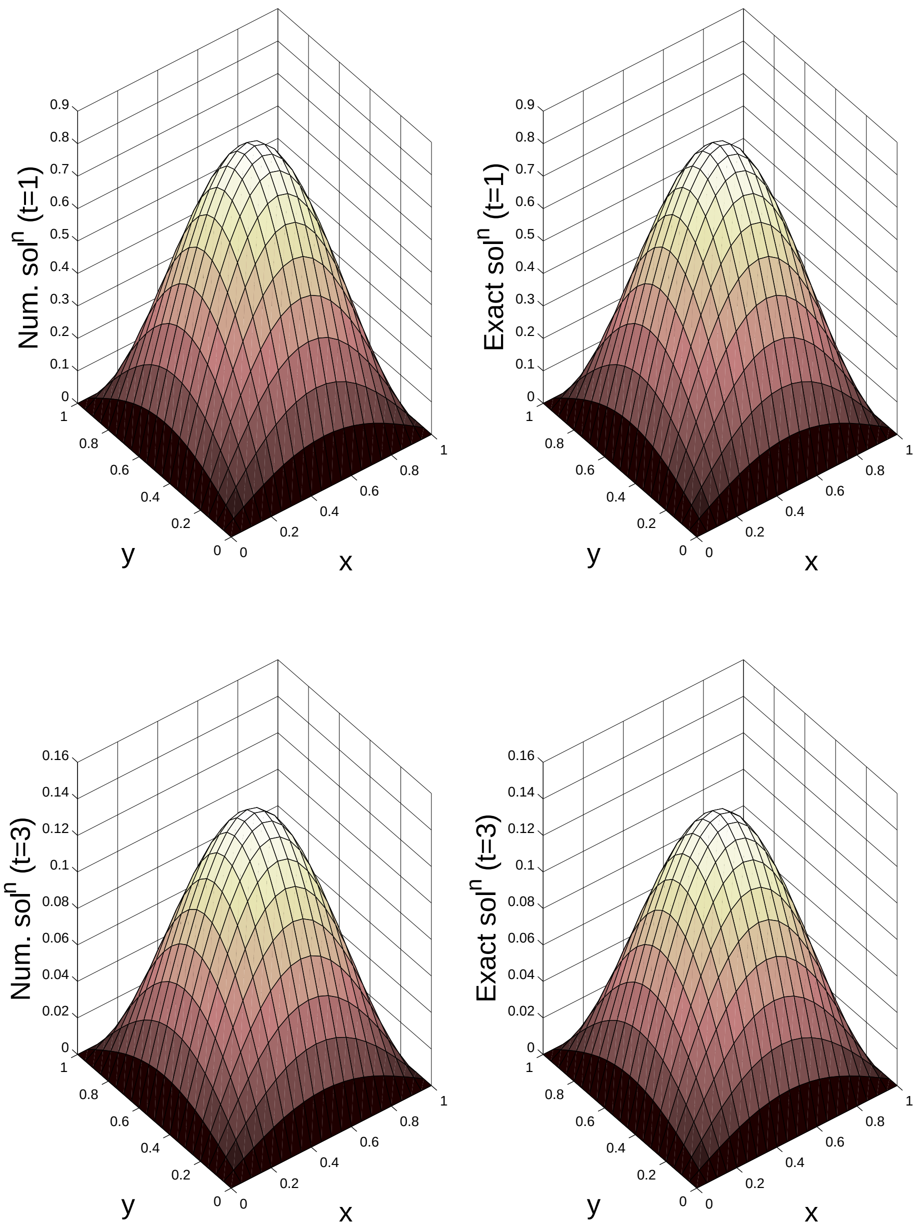


Figure 6.7: Surface plots of numerical and exact solutions of Example 6.7.

Chapter 7

Numerical Solution of Two Dimensional Coupled Burgers' Equation

7.1 Introduction

This chapter focuses on the numerical solution of two-dimensional coupled Burgers' equation. It is an important nonlinear parabolic partial differential equation in evolution theory and is named for the great Physicist Johannes Martinus Burgers' [29]. Burgers' equation has a convection term, a viscosity term and a time-dependent term, and is very similar to the Navier-Stokes equations except pressure gradient term.

Consider the following two-dimensional coupled Burgers' equation

$$\begin{aligned} \frac{\partial u}{\partial t}+u\frac{\partial u}{\partial x}+v\frac{\partial u}{\partial y} &= \frac{1}{R}\left(\frac{\partial^2 u}{\partial x^2}+\frac{\partial^2 u}{\partial y^2}\right), \\ \frac{\partial v}{\partial t}+u\frac{\partial v}{\partial x}+v\frac{\partial v}{\partial y} &= \frac{1}{R}\left(\frac{\partial^2 v}{\partial x^2}+\frac{\partial^2 v}{\partial y^2}\right), \quad (x,y,t)\in D\times(0,T], \end{aligned} \tag{7.1}$$

with initial conditions

$$\begin{aligned} u(x,y,0) &= \phi(x,y), \\ v(x,y,0) &= \psi(x,y), \ \ (x,y)\in D \end{aligned} \tag{7.2}$$

and the following boundary conditions

$$\begin{aligned} u(x,y,t) &= \phi_1(x,y,t), \\ v(x,y,t) &= \psi_1(x,y,t), \quad (x,y,t)\in \partial D\times(0,T], \end{aligned} \tag{7.3}$$

where D=$\{(x,y) : a \le x \le b, c \le y \le d\}$ denotes the rectangular domain, ∂D is its boundary and $(0,T]$ is the time interval. R is the Reynolds number, $u(x,y,t)$ and $v(x,y,t)$

are the unknown velocity components and $\phi(x, y)$, $\psi(x, y)$, $\phi_1(x, y, t)$ and $\psi_1(x, y, t)$ are the known sufficient smooth functions.

This system models a large number of physical phenomena such as traffic flow, flow of a shock wave traveling in a viscous fluid [35], phenomena of turbulence [29], interaction between the nonlinear convection process and the diffusive viscous process [71], sedimentation of two kinds of particles in fluid suspensions under the effect of gravity, etc.

To study the general properties and the solutions of Burgers' equation, the first attempt was made by Bateman [14], who derived the analytical solution for one dimensional case and later modified by Burger [29] to model the turbulence behavior. Subsequently, various methods have been developed by researchers to find the approximate analytical solutions of one and two dimensional Burgers' equations such as Variational iteration method [2, 20], Tanh-function method [196], Lie point symmetry method [3], modified Adomian decomposition method [1] etc. In [79], one dimensional Burgers' equation has been solved analytically for arbitrary initial condition. Fletcher [71] has used the Hopf-Cole transformation to find an analytical solution for the two dimensional system of Burgers' equations (7.1).

Numerical solutions of one and two dimensional Burgers' equations were studied by many schemes. Kutluay and Esen [120] have developed a linearized implicit finite-difference scheme to find numerical solutions of one-dimensional Burgers-like equations. Mittal and Jain [144] have developed a modified cubic B-spline collocation method and Korkmaz and Dağ [110] have presented a cubic B-spline functions based differential quadrature method for solving one dimensional Burgers' equation. Mantri et al. [137] have developed a qualocation method for one dimensional Burgers' equation, which is a quadrature based modification of the collocation approximations. Fletcher [70] has presented a comparisons of linear, quadratic and cubic finite element methods and three, five and seven points finite difference methods for the numerical solutions of one and two dimensional Burgers' equations. In [76, 77], local discontinuous Galerkin (LDG) finite element methods have been developed for one and two dimensional Burgers' equations. These methods were based on the Hopf-Cole transformation and transforms the Burgers' equation into linear equation. Rong-Pei et al. [184] have developed local discontinuous Galerkin method to solve modified Burgers' equation. Jain and Holla [86] have presented cubic spline function based two algorithms for solving Burgers' equation in one dimension and coupled Burgers' equation in two dimensions. Wei et al. [208] have used distributed approximating functionals (DAFs) in space direction and Taylor expansion in time direction for solving one and two dimensional Burgers' equations. Radwan [172] has derived two higher order finite difference schemes namely the fourth-order two-point compact scheme and the fourth-order accurate Du Fort Frankel scheme for two dimensional Burgers' equations. In [8], a meshfree solution of coupled Burgers' equation has been presented by applying the combination of collocation method using the radial basis functions (RBFs) with first-order accurate forward difference approximation. Zhang et al. [211] have proposed a meshfree method based on characteristic and Galerkin method for solutions of one and two dimensional Burgers' equations. Bahadir [12] has proposed a fully implicit finite difference scheme, which leads to a system of nonlinear difference equations to be solved at each time step. A discrete Adomian decomposition method has been proposed by Zhu et al. [213], to solve two dimensional coupled Burgers' equation. Mittal and Jiwari [147] have developed a polynomial based differential quadrature method for two dimensional coupled Burgers' equation. Recently, Goyal and Mehra [74] have developed a fast adaptive wavelet method for equation (7.1). Doha et al. [62] have proposed a J-GL-C method based on Jacobi polynomials and Gauss-Lobatto quadrature integration, in combination with the implicit Runge-Kutta-Nystrom (IRKN) scheme, for solving equation (7.1). Many other numerical methods have been developed to solve Burgers' equation such as finite element methods [10, 32, 34, 121, 164, 166], finite difference methods [132, 165], lattice Boltzmann method [65, 133], spectral method [148], spectral collocation methods [61, 66] etc.

In this chapter, we present a differential quadrature method based on modified cubic B-spline functions for finding the numerical solutions of two dimensional coupled Burgers' equation. The coupled Burgers' equation is discretized spatially by modified cubic B-spline differential quadrature method, which results in a system of nonlinear ODEs. The system of differential equations obtained over time is solved using SSP-RK43 scheme given by Spiteri and Ruuth [197]. The accuracy of the scheme is tested by solving five numerical experiments. The computed numerical solutions are compared with the analytical and other numerical solutions available in the literature.

7.2 Solution of Coupled Burgers' Equation

The region $a \leq x \leq b$, $c \leq y \leq d$ is discretized by taking N and M grid points in x and y direction respectively, such that $h_x = x_{i+1} - x_i$ and $h_y = y_{j+1} - y_j$.
Then according to two dimensional DQM stated in chapter 1, the first and second order partial derivatives of the function $u(x, y, t)$ can be approximated as follows:

$$\begin{aligned} u_x^{(1)}(x_i, y_j, t) &= \sum_{k=1}^{N} a_{ik}^{(1)} u(x_k, y_j, t), \quad u_y^{(1)}(x_i, y_j, t) = \sum_{k=1}^{M} b_{jk}^{(1)} u(x_i, y_k, t), \\ u_x^{(2)}(x_i, y_j, t) &= \sum_{k=1}^{N} a_{ik}^{(2)} u(x_k, y_j, t), \quad u_y^{(2)}(x_i, y_j, t) = \sum_{k=1}^{M} b_{jk}^{(2)} u(x_i, y_k, t), \\ &\qquad i = 1, 2, \ldots, N;\ j = 1, 2, \ldots, M. \end{aligned} \tag{7.4}$$

Similarly, for the function $v(x, y, t)$ we have

$$\begin{aligned} v_x^{(1)}(x_i, y_j, t) &= \sum_{k=1}^{N} a_{ik}^{(1)} v(x_k, y_j, t), \quad v_y^{(1)}(x_i, y_j, t) = \sum_{k=1}^{M} b_{jk}^{(1)} v(x_i, y_k, t), \\ v_x^{(2)}(x_i, y_j, t) &= \sum_{k=1}^{N} a_{ik}^{(2)} v(x_k, y_j, t), \quad v_y^{(2)}(x_i, y_j, t) = \sum_{k=1}^{M} b_{jk}^{(2)} v(x_i, y_k, t), \\ &\qquad i = 1, 2, \ldots, N;\ j = 1, 2, \ldots, M. \end{aligned} \tag{7.5}$$

First we compute the weighting coefficients $a_{ik}^{(1)}$, $a_{ik}^{(2)}$, $b_{jk}^{(1)}$ and $b_{jk}^{(2)}$. Then the spatial derivatives of u and v are discretized using the approximations (7.4) and (7.5), and finally the equation (7.1) is reduced into the following system of nonlinear first order ODEs

$$\begin{aligned} \frac{du_{i,j}}{dt} &= -u_{i,j} \sum_{k=1}^{N} a_{ik}^{(1)} u_{k,j} - v_{i,j} \sum_{k=1}^{M} b_{jk}^{(1)} u_{i,k} + \frac{1}{R} \left(\sum_{k=1}^{N} a_{ik}^{(2)} u_{k,j} + \sum_{k=1}^{M} b_{jk}^{(2)} u_{i,k} \right), \\ \frac{dv_{i,j}}{dt} &= -u_{i,j} \sum_{k=1}^{N} a_{ik}^{(1)} v_{k,j} - v_{i,j} \sum_{k=1}^{M} b_{jk}^{(1)} v_{i,k} + \frac{1}{R} \left(\sum_{k=1}^{N} a_{ik}^{(2)} v_{k,j} + \sum_{k=1}^{M} b_{jk}^{(2)} v_{i,k} \right), \\ &\qquad i = 1, 2, \ldots, N;\ \ j = 1, 2, \ldots, M, \end{aligned} \tag{7.6}$$

with the initial conditions

$$\begin{aligned} u(x_i, y_j, 0) &= \phi(x_i, y_j), \\ v(x_i, y_j, 0) &= \psi(x_i, y_j), \quad (x_i, y_j) \in D. \end{aligned} \tag{7.7}$$

Initial vector $[u_{i,j}^0, v_{i,j}^0]^T$ is computed using the initial conditions (7.7). Finally, system of nonlinear first order ordinary differential equations (7.6) is solved with the boundary conditions (7.3), using SSP-RK43 scheme and hence the approximate solution $[u_{i,j}^k, v_{i,j}^k]^T$ at a time level $t = t_k$ is computed.

7.3 Numerical Results

Example 7.1

We consider the Burgers' equation (7.1) in the computational domain $D=\{(x, y) : 0 \leq x \leq 1, 0 \leq y \leq 1\}$ with the following exact solutions, which have been obtained by Fletcher [71] using the Hopf-Cole transformation:

$$u(x,y,t) = \frac{3}{4} - \frac{1}{4\left(1+e^{\frac{R(-4x+4y-t)}{32}}\right)}, \tag{7.8}$$

$$v(x,y,t) = \frac{3}{4} + \frac{1}{4\left(1+e^{\frac{R(-4x+4y-t)}{32}}\right)}. \tag{7.9}$$

The initial and boundary conditions are taken from the exact solutions.

The numerical solutions are computed with a mesh size $h_x = h_y = .05$ and time step size $\Delta t = .001$.

In Table-7.1, numerical solutions and corresponding absolute errors are reported for $R = 100$ at $t = .01$ at some mesh points. We see that the numerical results are close to the exact solutions. Comparisons are also made with the results given by Bahadir [12], Mittal and Jiwari [147], Zhu et al. [213] and results of the proposed method are found better.

Table-7.2 and Table-7.3 present the numerical solutions and corresponding absolute errors for $R = 100$ at $t = 0.5$ and $t = 2.0$, respectively. These tables also depict the comparisons of our results with those given in Bahadir [12], Mittal and Jiwari [147], Zhu et al. [213] and Liu and Shi [133]. It is found that our results are better as compare to [12, 133, 147, 213]. To demonstrate the accuracy of proposed approach for higher time level, maximum absolute errors (MAE), root mean square errors (RMSE) and relative errors are computed up to time $t = 12$ and reported in Table-7.4. We observe that as calculations are carried out for higher time level the approximate solution approaches to the exact ones.

In addition, the performance of the method is also tested for a larger value of Reynolds number $R = 1000$ and 1200 at $t = 2$ with $\Delta t = .001$ and for finer grid size $h_x = h_y = .025$. Numerical solutions at some grid points are given in Table-7.5 and compared with the exact solutions. From these results it is concluded that the proposed method solves the system for large values of R with remarkable accuracy.

Approximate velocity profiles of u and v are plotted at $t = .5$, 2, 5 and 12, and are shown in Figure-7.1.

Example 7.2

We consider equation (7.1) in the computational domain $D = \{(x, y) : 0 \leq x \leq 0.5, 0 \leq y \leq 0.5\}$ with the following initial conditions as given in [86]

$$\begin{aligned} u(x, y, 0) &= \sin(\pi x) + \cos(\pi y), \\ v(x, y, 0) &= x + y. \end{aligned} \tag{7.10}$$

The boundary conditions are given by

$$\begin{aligned} u(0, y, t) &= \cos(\pi y), && 0 \leq y \leq 0.5, \\ u(.5, y, t) &= 1 + \cos(\pi y), && 0 \leq y \leq 0.5, \\ u(x, 0, t) &= 1 + \sin(\pi x), && 0 \leq x \leq 0.5, \\ u(x, .5, t) &= \sin(\pi x), && 0 \leq x \leq 0.5. \end{aligned} \tag{7.11}$$

$$\begin{aligned} v(0, y, t) &= y, && 0 \leq y \leq 0.5, \\ v(.5, y, t) &= 0.5 + y, && 0 \leq y \leq 0.5, \\ v(x, 0, t) &= x, && 0 \leq x \leq 0.5, \\ v(x, .5, t) &= x + 0.5, && 0 \leq x \leq 0.5. \end{aligned} \tag{7.12}$$

The analytic solution of this problem is not known.

It has been mentioned in [86] that time to reach the steady state solution does not depend on the Reynolds number and the steady state solution is reached at $t = .625$. Therefore, numerical computations are performed with $h_x = h_y = .05$, $\Delta t = .001$ up to $t = .625$. Numerical values of velocity u and v are reported in Table-7.6 and 7.7 at some mesh points for $R = 50$ and $R = 500$ and compared with the solutions obtained in Jain and Holla [86], Bahadir [12], and Mittal and Jiwari [147]. We have noticed that our results are in excellent agreement with those obtained in [12, 86, 147]. Numerical solutions are also computed for higher Reynolds number $R = 1200$ and 1500 with mesh size $h_x = h_y = .025$, $\Delta t = .001$ and are shown in Table-7.8. Steady state profiles of u and

v for $R = 50$ at $t = .625$ are shown in Figure-7.2. Similar figures have been obtained in [12].

Example 7.3

In this example, we consider 2D Burgers' equation (7.1) in the computational domain $D = \{(x, y) : 0 \le x \le 0.5, 0 \le y \le 0.5\}$ with the following initial conditions

$$\begin{aligned} u(x, y, 0) &= x + y, \\ v(x, y, 0) &= x - y. \end{aligned} \tag{7.13}$$

The exact solution is given in [213] as

$$\begin{aligned} u(x, y, t) &= \frac{x + y - 2xt}{1 - 2t^2}, \\ v(x, y, t) &= \frac{x - y - 2yt}{1 - 2t^2}. \end{aligned} \tag{7.14}$$

The boundary conditions can be obtained from the exact solution.

Absolute errors are reported in Table-7.9 at $t = 0.1$ and 0.4 for Reynolds number $R = 500$, $h = .05$ and $\Delta t = .001$. In order to compare our results with Zhu et al. [213], calculations are performed for smaller time step $\Delta t = .0001$ and obtained results are presented in Table-7.10. From tables it is clear that our method produces much better numerical results than [213]. In Table-7.11, we show the L_2 and L_∞ norms of error and CPU time at different levels of time. The physical behavior of numerical solutions at $t = 0.1$ and $t = 0.4$ is also presented in Figure-7.3 and 7.4, respectively. Similar figures were obtained in [213] at $t = 0.1$.

Example 7.4

We consider the following 2D Burgers' equation in the computational domain D=$\{(x, y) : 0 \le x \le 1, 0 \le y \le 1\}$

$$u_t + uu_x + uu_y = \frac{1}{R}(u_{xx} + u_{yy}), \;\; (x, y, t) \in D \times (0, T], \tag{7.15}$$

with the following initial conditions

$$u(x, y, 0) = \frac{1}{1 + e^{R(x+y)/2}}, \;\; (x, y) \in D. \tag{7.16}$$

The boundary conditions are given by

$$\begin{aligned} u(0,y,t) &= \frac{1}{1+e^{R(-t+y)/2}}, \\ u(1,y,t) &= \frac{1}{1+e^{R(-t+1+y)/2}}, \\ u(x,0,t) &= \frac{1}{1+e^{R(-t+x)/2}}, \\ u(x,1,t) &= \frac{1}{1+e^{R(-t+1+x)/2}}, \ \ (x,y,t) \in \partial D \times (0,T]. \end{aligned} \tag{7.17}$$

The exact solution is given by

$$u(x,y,t) = \frac{1}{1+e^{R(-t+x+y)/2}}. \tag{7.18}$$

The numerical solutions are computed at time $t = 0.25$ for Reynolds number $R = 1, 10$ and 100. Table-7.12 presents the L_2, L_∞ error norms and CPU time with $\Delta t = .001$ and for different grid sizes. These results are compared with those obtained in Duan and Liu [65]. It is found that our results are more accurate than [65] and also errors increases with the increasing Reynolds numbers. Therefore, it can be concluded that we must choose finer grids for high Reynolds number. Velocity profiles of u are depicted in Figure-7.5 at $t = .25, 3, 5, 12$ for $R = 1$. In order to compare our results with Young et al. [209], numerical experiments are also conducted for $R = 20$. Absolute errors are computed at different grid points at time $t = 0.5$, 0.75, 1.0, 1.25 and are shown in Table-7.13. The comparison shows that proposed scheme gives better accuracy, while using very less number of grid points in comparison with [209].

Example 7.5

We consider the Burgers' equation (7.1) in the computational domain $D=\{(x,y) : 0 \leq x \leq 1, 0 \leq y \leq 1\}$, with the following initial conditions as given in [10, 208]

$$\begin{aligned} u(x,y,0) &= \sin \pi x \ \ \sin \pi y, \\ v(x,y,0) &= (\sin \pi x + \sin 2\pi x) \ (\sin \pi y + \sin 2\pi y), \quad (x,y) \in D, \end{aligned} \tag{7.19}$$

and the homogenous boundary conditions

$$\begin{aligned} u(x,y,t) &= 0, \\ v(x,y,t) &= 0, \quad (x,y,t) \in \partial D \times (0,T]. \end{aligned} \tag{7.20}$$

The exact solution of this problem is not available in the literature.

In our first computation, we take $h_x = h_y = .05$, $\Delta t = .001$. First we compute the velocities u and v at some mesh points with $R = 1$ at $t = .01$. Computed results are given in Table-7.14 and are compared with those given in Arminjon and Beauchamp [10], Radwan [171], Goyal and Mehra [74], and Wei et al. [208]. It can be seen that our results are in excellent agreement with their results and the time step Δt is much bigger in our case. In addition, to check the performance of the method for higher value of R, computations are performed for $R = 100$, $h_x = h_y = .01$ and $\Delta t = .001$. Results are tabulated in Table-7.15 and compared with those given by Wei et al. [208] at $t = 0.5, 1.0$. It is observed that our results are very similar to the results given in [208]. To measure the computational efficiency of proposed scheme, the CPU time is also calculated in Table-7.16 at different time levels and compared with those obtained in Radwan [171]. It is apparent that CPU time used in our computation is much less. The physical behaviors of u and v are depicted in Figure-7.6 at $t = .01$ for $R = 1$. Similar trends have been observed in [8]. Figure-7.7 represents the contour plots of u and v at $t = 0.5$, 1.0 for $R = 100$. Same figures have been also obtained by Wei et al. [208].

7.4 Conclusions

1. Differential quadrature method has been used for space discretization of nonlinear coupled Burgers' equation and SSP-RK54 scheme is used to solve resulting system of first order nonlinear ordinary differential equations.

2. The accuracy of the approach is tested by taking five test problems. The results of computations indicate that modified cubic B-spline differential quadrature method gives more accurate results than previous works with larger time steps and with less computational cost.

3. The numerical results are computed for higher Reynolds number up to $R = 1500$.

4. The proposed method solves the nonlinear Burgers' equation without using any transformation or quasi-linearization approach.

Table 7.1: Numerical values of velocity u and v with $h_x = h_y = .05$, $R = 100$ at $t = .01$ of Example 7.1.

	Numerical solutions			Exact sol^n	Absolute errors	
(x, y)	Proposed method $\Delta t = .001$	Bahadir [12] $\Delta t = .0001$	Mittal and Jiwari [147] $\Delta t = .001$		Proposed method $\Delta t = .001$	Zhu et al. [213] $\Delta t = .0001$
$u(x, y, t)$						
(.1,.1)	0.6230428970	0.62310	0.62305	0.6230470339	4.1369E-6	5.9137E-5
(.5,.1)	0.5016217544	0.50161	0.50162	0.5016220674	3.1302E-7	4.8403E-6
(.9,.1)	0.5000110010	0.50000	0.50001	0.5000110003	7.4652E-10	3.4100E-8
(.3,.3)	0.6230488712	0.62311	0.62305	0.6230470339	1.8373E-6	5.9137E-5
(.7,.3)	0.5016221262	0.50162	0.50162	0.5016220674	5.8757E-8	4.8403E-6
(.1,.5)	0.7482743425	0.74827	0.74827	0.7482740405	3.0205E-7	1.6429E-6
(.5,.5)	0.6230488682	0.62311	0.62305	0.6230470339	1.8343E-6	5.9137E-5
(.9,.5)	0.5016225333	0.50162	0.50162	0.5016220673	4.6588E-7	-
(.3,.7)	0.7482740496	0.74827	0.74827	0.7482740405	9.0754E-9	-
(.7,.7)	0.6230488357	0.62311	0.62305	0.6230470339	1.8017E-6	-
(.1,.9)	0.7499882881	0.74998	0.74999	0.7499882902	2.1421E-9	-
(.5,.9)	0.7482734318	0.74827	0.74827	0.7482740405	6.0872E-7	-
(.9,.9)	0.6230392844	0.62311	0.62305	0.6230470339	7.7495E-6	-
$v(x, y, t)$						
(.1,.1)	0.8769571030	0.87688	0.87695	0.8769529661	4.1369E-6	5.9137E-5
(.5,.1)	0.9983782456	0.99837	0.99838	0.9983779326	3.1302E-7	4.8403E-6
(.9,.1)	0.9999889990	0.99998	0.99999	0.9999889997	7.4652E-10	3.4100E-8
(.3,.3)	0.8769511288	0.87689	0.87695	0.8769529661	1.8373E-6	5.9137E-5
(.7,.3)	0.9983778738	0.99838	0.99838	0.9983779326	5.8757E-8	4.8403E-6
(.1,.5)	0.7517256574	0.75172	0.75172	0.7517259595	3.0205E-7	1.6429E-6
(.5,.5)	0.8769511317	0.87689	0.87695	0.8769529661	1.8343E-6	5.9137E-5
(.9,.5)	0.9983774667	0.99838	0.99838	0.9983779326	4.6588E-7	-
(.3,.7)	0.7517259504	0.75173	0.75173	0.7517259595	9.0754E-9	-
(.7,.7)	0.8769511643	0.87689	0.87695	0.8769529661	1.8017E-6	-
(.1,.9)	0.7500117119	0.75001	0.75001	0.7500117097	2.1421E-9	-
(.5,.9)	0.7517265682	0.75173	0.75172	0.7517259595	6.0872E-7	-
(.9,.9)	0.8769607156	0.87689	0.87695	0.8769529661	7.7495E-6	-

Table 7.2: Numerical values of velocity u and v with $h_x = h_y = .05$, $R = 100$ at $t = 0.5$ of Example 7.1.

	Numerical solutions			Exact sol^n	Absolute errors		
(x, y)	Proposed method $\Delta t = .001$	Bahadir [12] $\Delta t = .0001$	Mittal and Jiwari [147] $\Delta t = .001$		Proposed method $\Delta t = .001$	Liu and Shi [133] $\Delta t = .005$	Zhu et al. [213] $\Delta t = .0001$
$u(x, y, t)$							
(.1,.1)	0.54418555	0.54235	0.54322	0.54332205	8.6350E-4	2.20E-3	2.7766E-4
(.5,.1)	0.50036870	0.49964	0.50035	0.50035259	1.6119E-5	2.92E-5	4.5208E-4
(.9,.1)	0.50000246	0.49931	0.50000	0.50000238	8.0102E-8	5.67E-7	3.3740E-6
(.3,.3)	0.54394874	0.54207	0.54321	0.54332205	6.2669E-4	8.29E-4	2.7766E-4
(.7,.3)	0.50036551	0.49961	0.50035	0.50035259	1.2917E-5	1.30E-4	4.5208E-4
(.1,.5)	0.74197686	0.74130	0.74219	0.74221404	2.3718E-4	2.02E-3	2.8655E-4
(.5,.5)	0.54348877	0.54222	0.54329	0.54332205	1.6672E-4	5.19E-4	2.7766E-4
(.9,.5)	0.50035023	0.49997	0.50035	0.50035259	2.3552E-6	1.86E-4	-
(.3,.7)	0.74211463	0.74146	0.74221	0.74221404	9.9416E-5	2.93E-3	-
(.7,.7)	0.54328516	0.54243	0.54332	0.54332205	3.6892E-5	7.85E-4	-
(.1,.9)	0.74994386	0.74913	0.74995	0.74994566	2.0073E-6	2.30E-5	-
(.5,.9)	0.74219639	0.74201	0.74221	0.74221404	1.7654E-5	2.86E-3	-
(.9,.9)	0.54334979	0.54232	0.54332	0.54332205	2.7741E-5	1.05E-3	-
$v(x, y, t)$							
(.1,.1)	0.95581444	0.95577	0.95678	0.95667795	8.6350E-4	1.72E-3	2.7766E-4
(.5,.1)	0.99963129	0.99827	0.99965	0.99964741	1.6119E-5	1.59E-4	4.5208E-4
(.9,.1)	0.99999754	0.99861	1.00000	0.99999762	8.0102E-8	2.63E-6	3.3740E-6
(.3,.3)	0.95605126	0.95596	0.95679	0.95667795	6.2669E-4	2.20E-4	2.7766E-4
(.7,.3)	0.99963449	0.99827	0.99964	0.99964741	1.2917E-5	2.44E-4	4.5208E-4
(.1,.5)	0.75802314	0.75699	0.75780	0.75778596	2.3718E-4	5.31E-4	2.8655E-4
(.5,.5)	0.95651123	0.95685	0.95671	0.95667795	1.6672E-4	5.62E-4	2.7766E-4
(.9,.5)	0.99964976	0.99903	0.99965	0.99964741	2.3552E-6	2.39E-4	-
(.3,.7)	0.75788537	0.75723	0.75779	0.75778596	9.9416E-5	2.05E-3	-
(.7,.7)	0.95671484	0.95746	0.95668	0.95667756	3.6892E-5	6.06E-4	-
(.1,.9)	0.75005614	0.74924	0.75005	0.75005414	2.0073E-6	3.04E-5	-
(.5,.9)	0.75780361	0.75781	0.75779	0.75778596	1.7654E-5	2.49E-3	-
(.9,.9)	0.95665021	0.95777	0.95667	0.95667795	2.7741E-5	1.21E-3	-

Table 7.3: Numerical values of velocity u and v with $h_x = h_y = .05$, $R = 100$ at $t = 2.0$ of Example 7.1.

	Numerical solutions			Exact sol^n	Absolute errors	
(x, y)	Proposed method $\Delta t = .001$	Bahadir [12] $\Delta t = .0001$	Mittal and Jiwari [147] $\Delta t = .001$		Proposed method $\Delta t = .001$	Liu and Shi [133] $\Delta t = .005$
$u(x, y, t)$						
(.1,.1)	0.50050084	0.49983	0.50048	0.50048018	2.0654E-5	1.08E-5
(.5,.1)	0.50000336	0.49930	0.50000	0.50000324	1.2254E-7	2.67E-7
(.9,.1)	0.49999996	0.49930	0.50000	0.50000000	6.0147E-8	6.68E-8
(.3,.3)	0.50050125	0.49977	0.50048	0.50048018	2.1070E-5	1.40E-4
(.7,.3)	0.50000429	0.49930	0.50000	0.50000324	1.0462E-6	2.78E-6
(.1,.5)	0.55640420	0.55461	0.55540	0.55567503	7.2916E-4	2.34E-3
(.5,.5)	0.50049593	0.49973	0.50048	0.50048018	1.5747E-5	2.91E-4
(.9,.5)	0.50000572	0.49931	0.50000	0.50000324	2.4752E-6	2.87E-6
(.3,.7)	0.55605555	0.55429	0.55540	0.55567503	3.8051E-4	7.52E-4
(.7,.7)	0.50053624	0.49970	0.50048	0.50048018	5.6056E-5	3.72E-4
(.1,.9)	0.74406456	0.74340	0.74422	0.74425566	1.9109E-4	1.61E-3
(.5,.9)	0.55583135	0.55413	0.55541	0.55567503	1.5631E-4	4.80E-4
(.9,.9)	0.50051858	0.50001	0.50048	0.50048168	3.6898E-5	4.63E-4
$v(x, y, t)$						
(.1,.1)	0.99949916	0.99826	0.99952	0.99951982	2.0654E-5	6.88E-5
(.5,.1)	0.99999664	0.99860	1.00000	0.99999676	1.2254E-7	2.96E-6
(.9,.1)	1.00000000	0.99861	1.00000	0.99999998	6.0147E-8	2.18E-8
(.3,.3)	0.99949875	0.99820	0.99952	0.99951982	2.1070E-5	2.97E-4
(.7,.3)	0.99999571	0.99860	1.00000	0.99999676	1.0462E-6	3.52E-6
(.1,.5)	0.94359580	0.94393	0.94460	0.94432496	7.2916E-4	6.42E-4
(.5,.5)	0.99950407	0.99821	0.99952	0.99951982	1.5747E-5	4.48E-4
(.9,.5)	0.99999428	0.99862	1.00000	0.99999676	2.4752E-6	8.60E-6
(.3,.7)	0.94394445	0.94409	0.94460	0.94432496	3.8051E-4	2.11E-3
(.7,.7)	0.99946376	0.99823	0.99952	0.99951982	5.6056E-5	6.14E-4
(.1,.9)	0.75593543	0.75500	0.75578	0.75574434	1.9109E-4	5.01E-4
(.5,.9)	0.94416865	0.94441	0.94459	0.94432497	1.5631E-4	3.73E-3
(.9,.9)	0.99948142	0.99846	0.99952	0.99951832	3.6898E-5	6.81E-4

Table 7.4: Error norms with $h_x = h_y = .05$ and $R = 100$ of Example 7.1.

t	u			v			
	MAE	RMSE	Rel. err.	MAE	RMSE	Rel. err.	CPU time(s)
.01	1.7043E-4	1.8412E-5	2.7680E-5	1.7043E-4	1.8412E-5	1.9881E-5	.03
.5	9.8298E-4	2.2796E-4	3.5719E-4	9.8297E-4	2.2796E-4	2.3930E-4	.25
2.0	8.7692E-4	2.1645E-4	3.8067E-4	8.7692E-4	2.1645E-4	2.1337E-4	.91
3.0	9.0788E-4	1.6910E-4	3.1329E-4	9.0788E-4	1.6910E-4	1.6295E-4	1.38
5.0	9.8970E-5	8.1480E-6	1.5516E-5	9.8990E-5	8.1480E-6	7.7609E-6	2.30
7.0	1.9995E-7	1.6383E-8	3.1206E-8	1.9995E-7	1.6383E-8	1.5603E-8	3.23
9.0	3.8603E-10	3.1628E-11	6.0244E-11	3.8603E-10	3.1631E-11	3.0125E-11	4.17
12.0	6.5614E-14	2.3960E-14	4.5638E-14	2.0850E-13	6.9216E-14	6.5920E-14	5.36

Table 7.5: Numerical results for $R = 1000$, 1200 at $t = 2$ with $\Delta t = .001$ and 41×41 grid of Example 7.1.

	$R = 1000$		$R = 1200$			
(x, y)	Num. u	Num. v	Num. u	Num. v	Exact u	Exact v
(.1,.1)	0.49964	1.00034	0.49715	1.00285	0.50000	1.00000
(.5,.1)	0.50000	1.00000	0.49996	1.00000	0.50000	1.00000
(.9,.1)	0.49990	1.00001	0.49965	1.00035	0.50000	1.00000
(.3,.3)	0.50016	0.99984	0.50025	0.99975	0.50000	1.00000
(.7,.3)	0.50022	0.99978	0.50049	0.99951	0.50000	1.00000
(.1,.5)	0.49875	1.00124	0.49637	1.00362	0.50000	1.00000
(.5,.5)	0.50000	1.00000	0.50004	0.99996	0.50000	1.00000
(.9,.5)	0.50001	0.99999	0.50072	0.99993	0.50000	1.00000
(.3,.7)	0.49896	1.00102	0.49629	1.00371	0.50000	1.00000
(.7,.7)	0.50013	0.99987	0.50069	0.99931	0.50000	1.00000
(.1,.9)	0.74997	0.75003	0.74991	0.75009	0.75000	0.75000
(.5,.9)	0.49386	1.00613	0.49469	1.00531	0.50000	1.00000
(.9,.9)	0.50000	1.00000	0.50000	1.00000	0.50000	1.00000

Table 7.6: Numerical results at $t = .625$ for $R = 50$ and $N = M = 21$ of Example 7.2.

(x, y)	Proposed method	Jain and Holla [86]	Mittal and Jiwari [147]	Bahadir [12]
	Numerical values of velocity u			
(.1,.1)	0.970585	0.97258	0.96921	0.96688
(.3,.1)	1.151574	1.16214	1.14940	1.14827
(.2,.2)	0.862447	0.86281	0.86195	0.85911
(.4,.2)	0.980827	0.96483	0.97972	0.97637
(.1,.3)	0.663353	0.66318	0.66340	0.66019
(.3,.3)	0.772274	0.77030	0.77201	0.76932
(.2,.4)	0.582738	0.58070	0.58266	0.57966
(.4,.4)	0.761842	0.74435	0.76065	0.75678
	Numerical values of velocity v			
(.1,.1)	0.098428	0.09773	0.09787	0.09824
(.3,.1)	0.141090	0.14039	0.14014	0.14112
(.2,.2)	0.167321	0.16660	0.16708	0.16681
(.4,.2)	0.172248	0.17397	0.17181	0.17065
(.1,.3)	0.263803	0.26294	0.26362	0.26261
(.3,.3)	0.226536	0.22463	0.22630	0.22576
(.2,.4)	0.329367	0.32402	0.32873	0.32745
(.4,.4)	0.328893	0.31822	0.32731	0.32441

Table 7.7: Numerical results at $t = .625$ for $R = 500$ of Example 7.2.

	Proposed method			Jain and Holla [86]	Mittal and Jiwari [147]	Bahadir [12]
(x, y)	$N = M = 21$	$N = M = 31$	$N = M = 41$	$N = M = 41$	$N = M = 25$	$N = M = 21$
	Numerical values of velocity u					
(.15,.1)	0.964465	0.959430	0.960778	0.96066	0.96761	0.96650
(.3,.1)	1.029908	0.980835	0.970515	0.96852	1.04403	1.02970
(.1,.2)	0.840304	0.844443	0.844390	0.84104	0.85449	0.84449
(.2,.2)	0.880544	0.870308	0.869170	0.86866	0.91481	0.87631
(.1,.3)	0.675582	0.678624	0.678688	0.67792	0.68668	0.67809
(.3,.3)	0.810984	0.778471	0.774280	0.77254	0.82894	0.79792
(.15,.4)	0.547864	0.547063	0.547239	0.54543	0.55412	0.54601
(.2,.4)	0.594760	0.587874	0.587587	0.58564	0.61024	0.58874
	Numerical values of velocity v					
(.15,.1)	0.088681	0.085847	0.086529	0.08612	0.08770	0.09020
(.3,.1)	0.107883	0.082916	0.077526	0.07712	0.11439	0.10690
(.1,.2)	0.177108	0.178964	0.178906	0.17828	0.18128	0.17972
(.2,.2)	0.169529	0.163332	0.162646	0.16202	0.18443	0.16777
(.1,.3)	0.260731	0.261823	0.261778	0.26094	0.26514	0.26222
(.3,.3)	0.244151	0.219622	0.216409	0.21542	0.25498	0.23497
(.15,.4)	0.323536	0.315426	0.314824	0.31360	0.31375	0.31753
(.2,.4)	0.313240	0.299906	0.299016	0.29776	0.31462	0.30371

Table 7.8: Numerical results at $t = .625$ with $N = M = 41$, $\Delta t = .001$ of Example 7.2.

	$R = 1200$		$R = 1500$	
(x, y)	u	v	u	v
(.15,.1)	0.964486	0.087661	0.966157	0.088398
(.3,.1)	0.994682	0.089244	1.009743	0.096525
(.1,.2)	0.845572	0.178998	0.845184	0.178769
(.2,.2)	0.874949	0.165032	0.878667	0.167067
(.1,.3)	0.679879	0.261733	0.679638	0.261586
(.3,.3)	0.788421	0.225640	0.798285	0.232718
(.15,.4)	0.549387	0.316921	0.549934	0.319252
(.2,.4)	0.590868	0.302111	0.593049	0.305828

Table 7.9: Absolute errors for $R = 500$, $N = M = 21$ and $\Delta t = .001$ of Example 7.3.

	$t = 0.1$		$t = 0.4$	
(x, y)	u	v	u	v
(.1,.1)	6.4392E-9	9.0914E-8	3.4051E-5	1.0982E-4
(.3,.1)	9.6570E-7	1.3957E-7	7.5609E-5	6.8940E-5
(.2,.2)	1.3726E-7	5.2305E-8	1.7559E-5	6.1213E-5
(.4,.2)	3.4424E-5	1.2368E-5	1.3142E-5	1.9016E-6
(.1,.3)	2.1993E-8	2.2153E-7	4.0438E-4	1.1973E-3
(.3,.3)	6.9693E-6	4.7591E-6	4.3935E-5	2.0305E-4
(.2,.4)	1.0468E-6	6.6090E-7	4.6586E-4	1.4875E-3
(.3,.4)	1.1901E-5	1.1638E-5	3.5245E-4	1.3163E-3

Table 7.10: Comparison of absolute errors for $R = 500$, $N = M = 21$ and $\Delta t = .0001$ of Example 7.3.

	$u(x, y, t)$			$v(x, y, t)$		
(x, y)	Numerical sol^n	Absolute errors		Numerical sol^n	Absolute errors	
	Proposed method	Proposed method	Zhu et al. [213]	Proposed method	Proposed method	Zhu et al. [213]
$t = 0.1$						
(.1,.1)	0.1836734700	6.3332E-10	3.3075E-6	-0.0204081723	9.0041E-9	1.0538E-6
(.3,.1)	0.3469388729	9.7371E-8	5.5616E-6	0.1836734832	1.3868E-8	3.3077E-6
(.2,.2)	0.3673469526	1.3801E-8	6.6152E-6	-0.0408163213	5.2570E-9	2.1077E-6
(.4,.2)	0.5306156878	3.4429E-6	8.8694E-6	0.1632665427	1.2365E-6	2.2540E-6
(.1,.3)	0.3877551038	1.8096E-9	7.6693E-6	-0.2653060967	2.5758E-8	7.5234E-6
(.3,.3)	0.5510211070	6.9885E-7	9.9233E-6	-0.0612240123	4.7750E-7	3.1615E-6
(.2,.4)	0.5714286771	1.0564E-7	1.0977E-5	-0.2857142160	6.9678E-8	8.5770E-6
(.3,.4)	0.6530624175	1.1931E-6	1.2104E-5	-0.1836723007	1.1686E-6	6.3960E-6
$t = 0.4$						
(.1,.1)	0.17647929	3.4207E-6	1.0194E-4	-0.11769314	1.1025E-5	3.5483E-4
(.3,.1)	0.23526882	7.5755E-6	5.5872E-4	0.17647579	6.9199E-6	1.0195E-4
(.2,.2)	0.35296364	1.7706E-6	2.0389E-4	-0.23540217	6.1659E-6	7.0967E-4
(.4,.2)	0.41174526	1.3128E-6	6.6067E-4	0.05877873	1.8901E-7	4.5678E-4
(.1,.3)	0.47062902	4.0564E-5	1.5094E-4	-0.64717936	1.2002E-4	1.3174E-3
(.3,.3)	0.52944368	4.4423E-6	3.0584E-4	-0.35309204	2.0471E-5	1.0645E-3
(.2,.4)	0.64710554	4.6745E-5	4.8996E-5	-0.76485444	1.4911E-4	1.6722E-3
(.3,.4)	0.67650617	3.5387E-5	1.7939E-4	-0.61777818	1.3193E-4	1.5458E-3

Table 7.11: Errors of Example 7.3 for $R = 500$, $N = M = 21$ and $\Delta t = .0001$.

t	L_2	L_∞	CPU time(s)
.01	2.4371E-5	5.9231E-6	.06
.05	7.9137E-5	2.2580E-5	.23
.1	1.0597E-4	3.2506E-5	.43
.2	1.5193E-4	3.6833E-5	.80
.3	2.7067E-4	4.0271E-5	1.20
.4	5.8403E-4	7.9467E-5	1.61

Table 7.12: Errors of Example 7.4 at $t = .25$ with $\Delta t = .001$.

		Proposed method			Duan and Liu [65]	
R	Grid points	L_2	L_∞	CPU time(s)	L_2	L_∞
1	4× 4	1.7293E-5	4.3726E-5	.03	1.914E-5	4.423E-5
	10×10	4.9798E-5	6.5233E-5	.05	2.016E-5	7.882E-5
	20×20	5.6196E-5	6.6368E-5	.08	2.652E-4	1.483E-4
10	4 × 4	2.7636E-3	7.3628E-3	.03	–	–
	10× 10	3.1799E-4	1.9106E-3	.05	2.565E-3	1.061E-2
	20× 20	1.8545E-4	8.4678E-4	.08	4.536E-4	3.077E-3
	30× 30	1.9381E-4	6.8501E-4	.20	–	–
100	30×30	4.0574E-3	4.8182E-2	.20	–	–
	40×40	2.0214E-3	2.0001E-2	.25		
	70×70	9.8851E-4	9.2900E-3	.78		
	80×80	9.0412E-4	8.3590E-3	.98	7.610E-3	5.893E-2

Table 7.13: Absolute errors for $R = 20$, $\Delta t = 0.001$ of Example 7.4.

	Proposed scheme (N=M=21)	Young et al. [209] (N=M=441)	Proposed scheme (N=M=21)	Young et al. [209] (N=M=441)	Proposed scheme (N=M=21)	Young et al. [209] (N=M=441)	Proposed scheme (N=M=21)	Young et al. [209] (N=M=441)
(x, y)	$t = .5$		$t = .75$		$t = 1.0$		$t = 1.25$	
(.1,.1)	6.6933E-4	6.07E-4	6.7605E-5	1.70E-4	5.8709E-6	6.08E-6	4.8975E-7	3.29E-5
(.5,.1)	9.5131E-4	4.66E-4	1.7840E-3	1.05E-3	2.5518E-4	1.95E-4	2.3660E-5	4.77E-5
(.9,.1)	4.0780E-5	1.59E-6	9.9030E-5	6.60E-5	2.1692E-3	9.45E-4	8.3510E-4	6.38E-4
(.3,.3)	1.4629E-3	3.79E-4	1.5480E-3	7.88E-5	2.1742E-4	9.11E-5	2.1238E-5	5.43E-5
(.7,.3)	1.5613E-5	9.98E-7	3.3513E-4	2.55E-4	2.3642E-3	1.75E-3	7.2443E-4	2.99E-4
(.1,.5)	9.5131E-4	4.66E-4	1.7840E-3	1.05E-3	2.5518E-4	1.95E-4	2.3660E-5	4.77E-5
(.5,.5)	6.6873E-6	8.07E-6	4.6822E-4	5.18E-4	2.3528E-3	2.99E-3	6.8496E-4	6.45E-4
(.9,.5)	3.2373E-7	2.00E-7	1.7676E-7	3.93E-6	9.0495E-5	1.18E-4	1.1458E-3	2.13E-3
(.3,.7)	1.5614E-5	9.98E-7	3.3512E-4	2.55E-4	2.3642E-3	1.75E-3	7.2443E-4	2.99E-4
(.7,.7)	1.2747E-7	1.95E-8	2.8078E-6	3.16E-6	1.1739E-4	1.82E-4	1.3416E-3	3.03E-3
(.1,.9)	4.0880E-5	1.59E-6	9.9030E-5	6.60E-5	2.1692E-3	9.45E-4	8.3510E-4	6.38E-4
(.5,.9)	3.2373E-7	2.00E-7	1.7676E-7	3.93E-6	9.0495E-5	1.18E-4	1.1458E-3	2.13E-3
(.9,.9)	2.0738E-12	3.92E-8	1.1810E-8	2.40E-7	8.8739E-7	1.96E-6	2.3162E-5	5.04E-5

Table 7.14: Numerical results of Example 7.5 at $t = .01$ with $R = 1$.

	Present method		Goyal and Mehra [74]		Wei et al. [208]		Arminjon and Beauchamp [10]		Radwan [171]	
(x, y)	$\Delta t = .001$ $h_x = h_y = \frac{1}{20}$		$\Delta t = .0002$ $h_x = h_y = \frac{1}{51}$		$\Delta t = .0002$ $h_x = h_y = \frac{1}{20}$		$\Delta t = .00033$ $h_x = h_y = \frac{1}{80}$		$\Delta t = .0001$ $h_x = h_y = \frac{1}{40}$	
	u	v	u	v	u	v	u	v	u	v
(.1,.1)	0.0725445	0.4317723	0.07231	0.43091	0.0725087	0.4311635	0.0725139	0.431320	0.07273	0.43448
(.2,.8)	0.2775990	-0.1245408	0.27757	-0.12447	0.2775800	-0.1243656	0.288334	-0.121317	0.27800	-0.13148
(.4,.4)	0.7216912	1.6525725	0.72156	1.65321	0.7216922	1.6525602	0.721716	1.65270	0.72285	1.65917
(.7,.1)	0.2048236	0.0668107	0.20253	0.06672	0.2047829	0.0668145	0.201048	0.0671123	0.20497	0.06417
(.9,.9)	0.0794961	0.0135064	0.07939	0.01341	0.0794635	0.0134250	0.0794639	0.0134409	0.07953	0.01476

Table 7.15: Results of Example 7.5 with $h_x = h_y = .01$, $\Delta t = .001$ and $R = 100$.

	$t = 0.5$				$t = 1.0$			
	Present method		Wei et al. [208]		Present method		Wei et al. [208]	
(x, y)	u	v	u	v	u	v	u	v
(.1,.1)	0.01509454	0.1216257	0.0150939	0.1216201	0.0072645	0.0554206	0.0072642	0.0554181
(.2,.8)	0.1583942	0.9865354	0.1583939	0.9865377	0.0807578	0.5817098	0.0807567	0.5817142
(.4,.4)	0.1282258	0.7002100	0.1282258	0.7002100	0.0704515	0.3690012	0.0704512	0.3690011
(.7,.1)	0.1335357	0.0999821	0.1335301	0.0999871	0.0681663	0.0744596	0.0681616	0.0744617
(.8,.8)	0.5637785	1.1851217	0.5637791	1.1851231	0.2957085	0.6967747	0.2957080	0.6967743
(.9,.9)	0.2812878	0.2219794	0.2812818	0.2219593	0.3665171	0.7524211	0.3664864	0.7523752

Table 7.16: Comparison of CPU time of Example 7.5.

R	t	Grid points	CPU time(s)	
			Proposed method ($\Delta t = .001$)	Radwan [171] ($\Delta t = .00125$)
1	.01	10× 10	0.014	4
		20× 20	0.024	14
100	0.1	100×100	0.98	–
	0.3	100×100	2.3	–
	0.5	100×100	3.9	–
	0.7	100×100	5.3	–
	1.0	100×100	7.2	–

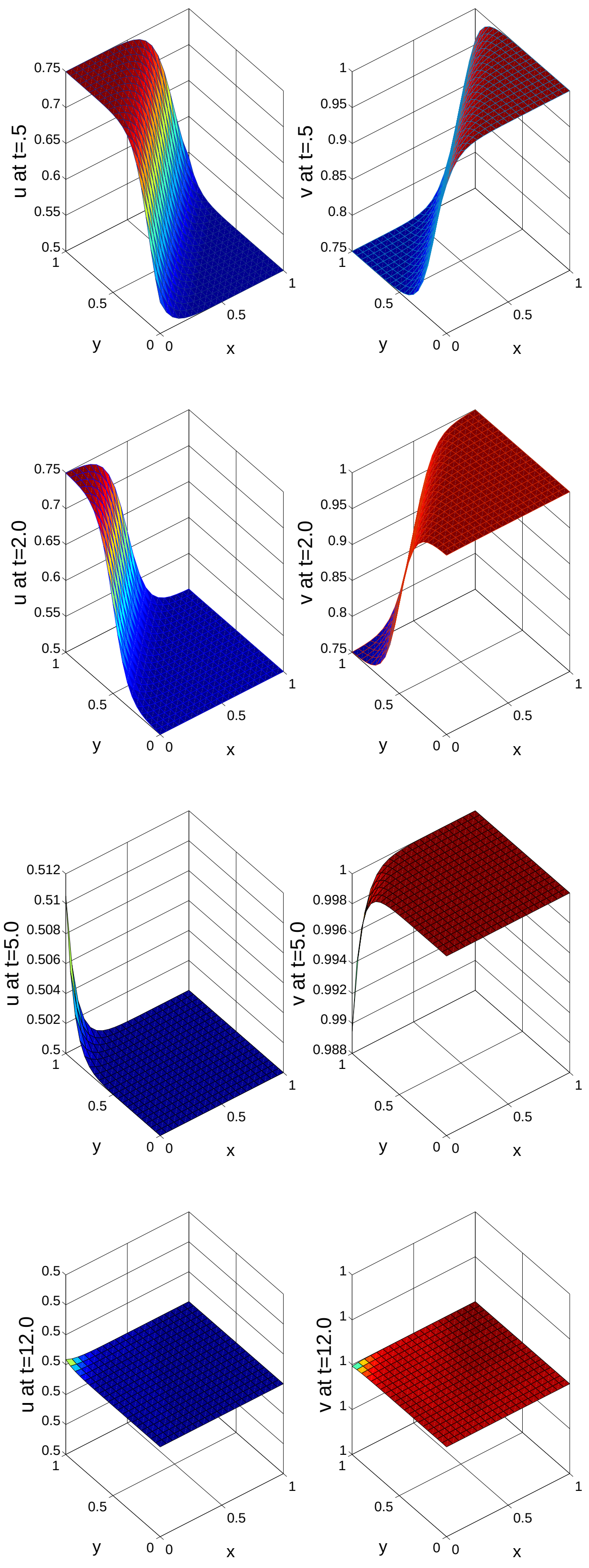


Figure 7.1: Profiles of u and v of Example 7.1 with $\Delta t = .001$ and $h_x = h_y = .05$.

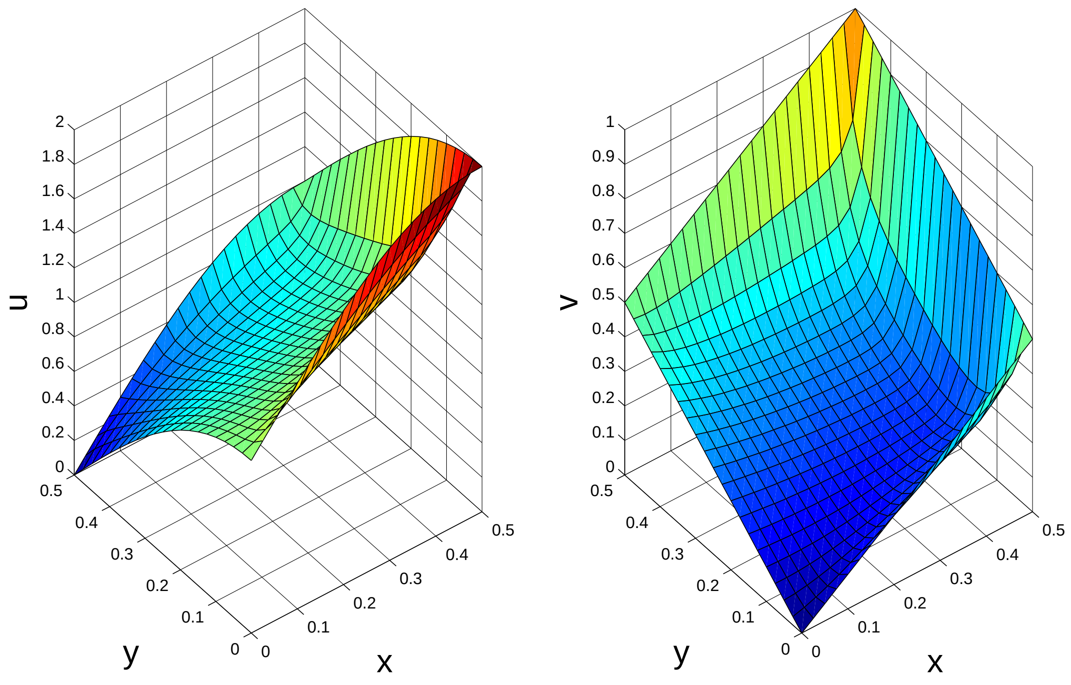


Figure 7.2: Profiles of u and v of Example 7.2 at $t = .625$ for $R = 50$.

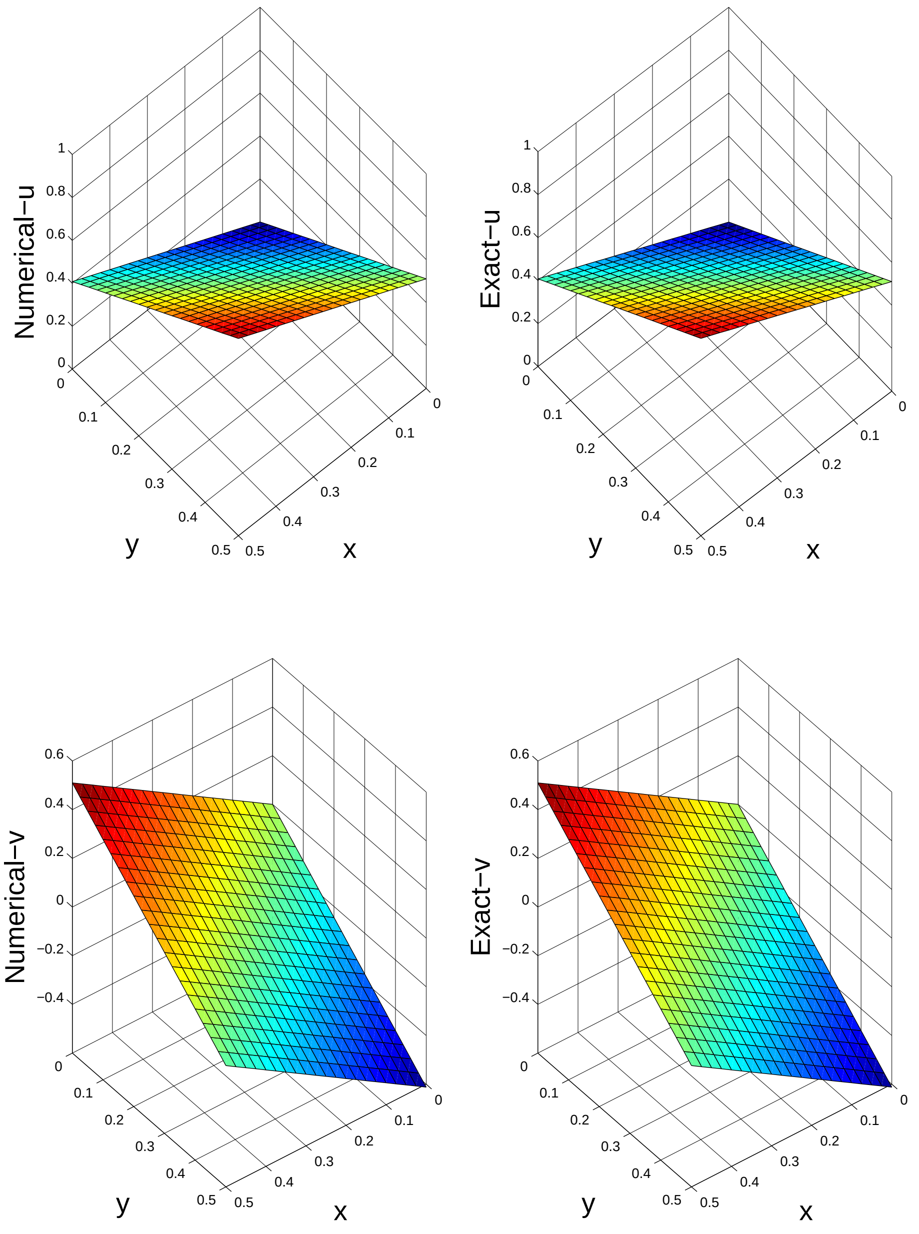


Figure 7.3: Profiles of u and v of Example 7.3 at $t = 0.1$ with $\Delta t = .001$ and $h_x = h_y = .05$.

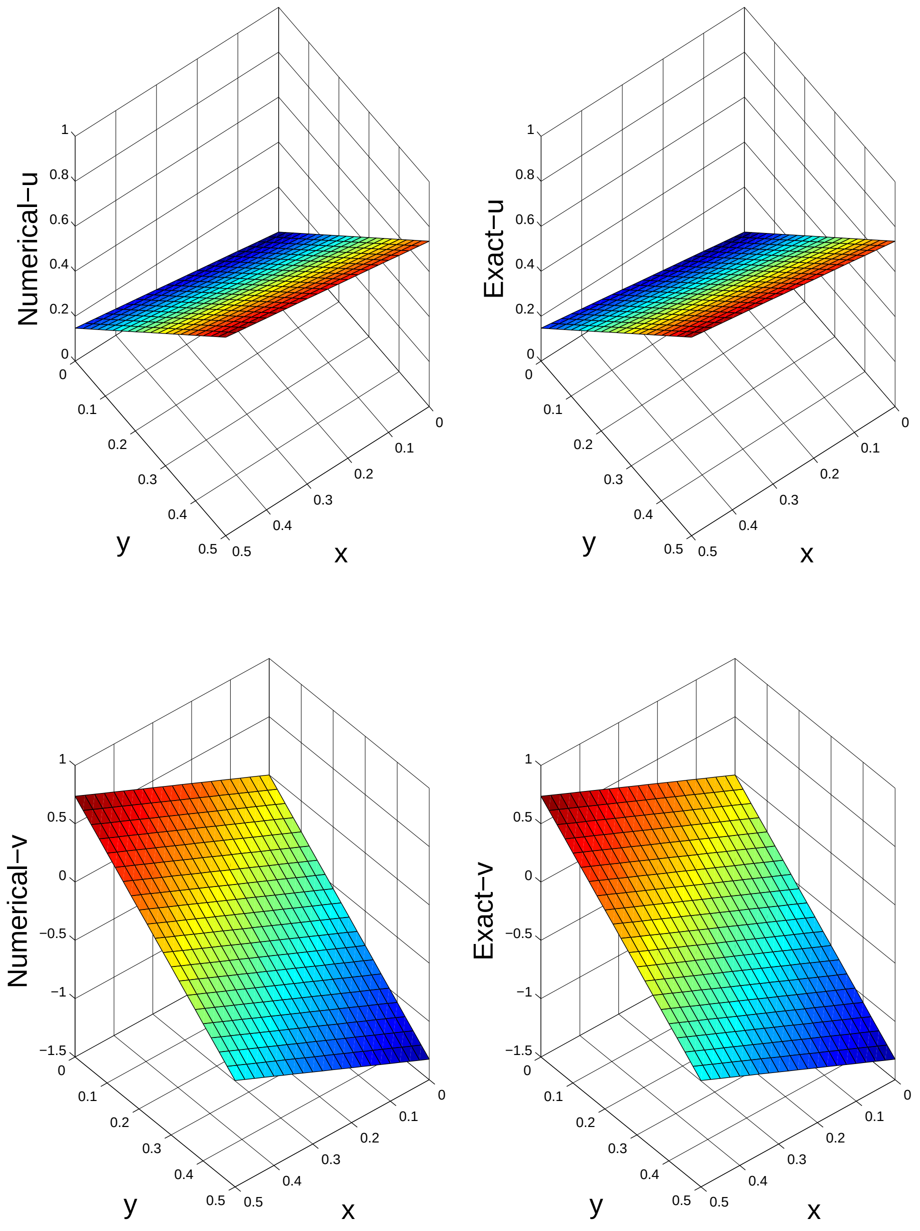


Figure 7.4: Profiles of u and v of Example 7.3 at $t = 0.4$ with $\Delta t = .001$ and $h_x = h_y = .05$.

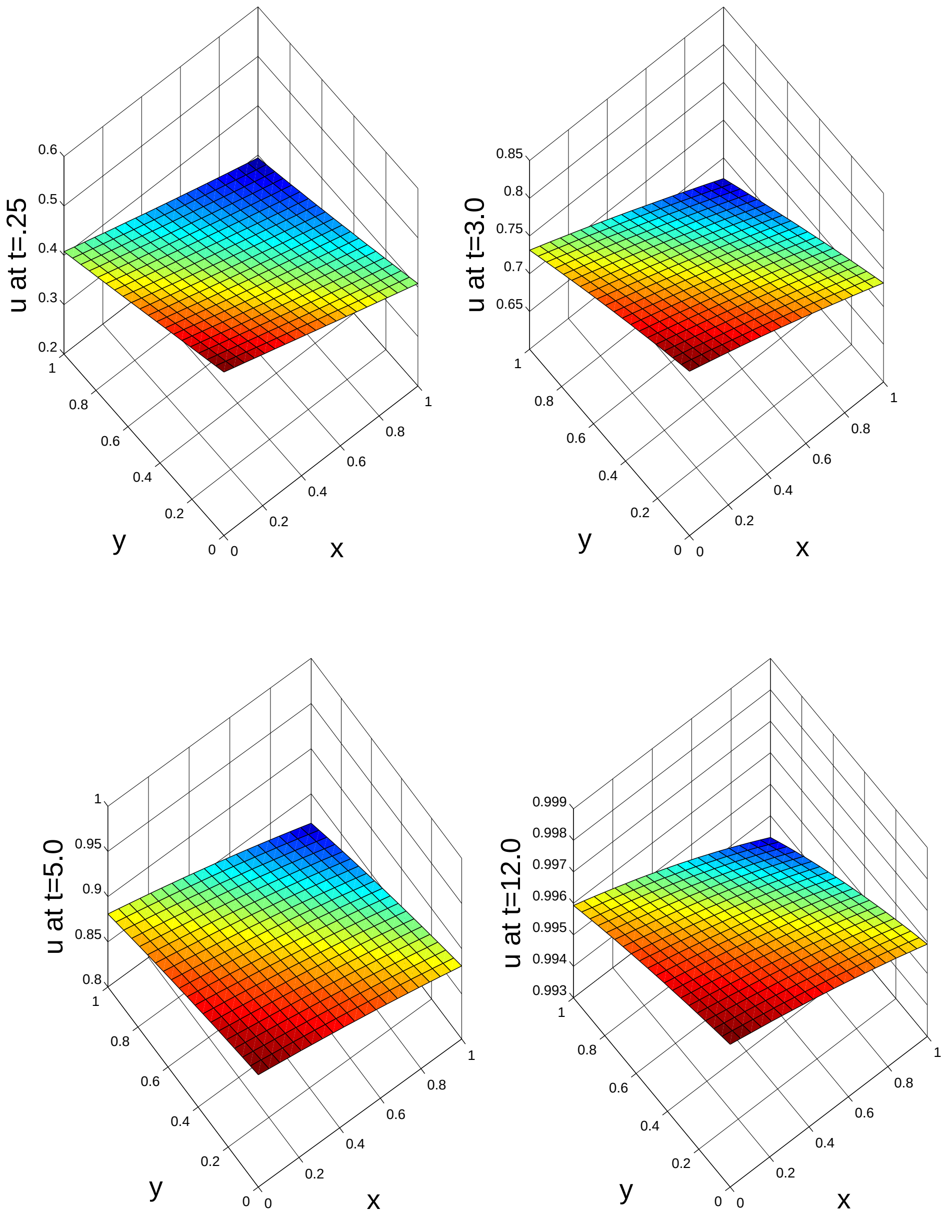


Figure 7.5: Profile of u of Example 7.4 at different time levels with $\Delta t = .001$ and $N = M = 20$.

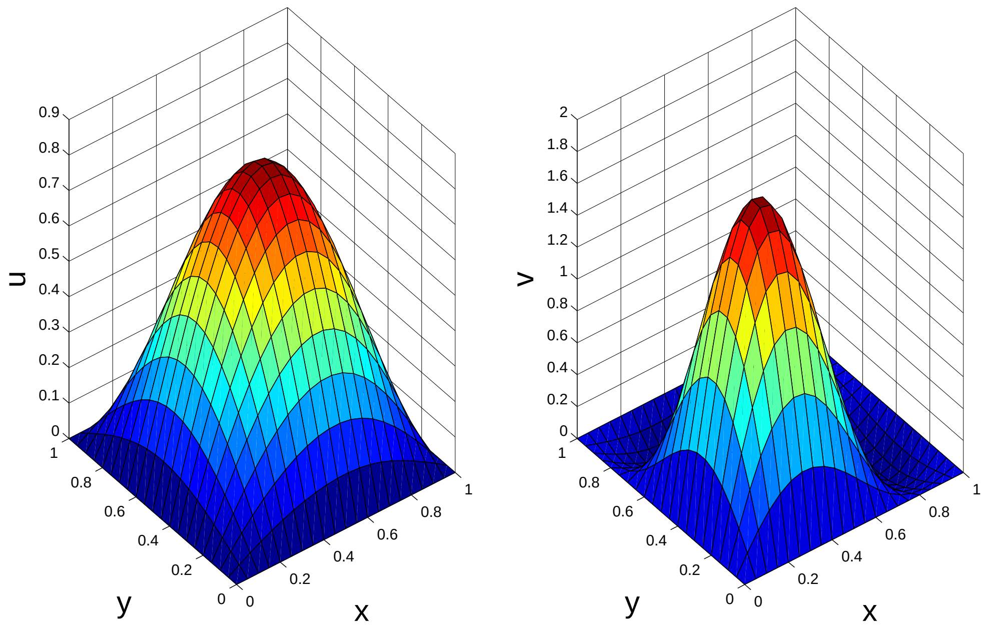


Figure 7.6: Profiles of u and v of Example 7.5 at $t = .01$ with $h_x = h_y = .05$ and $\Delta t = .001$.

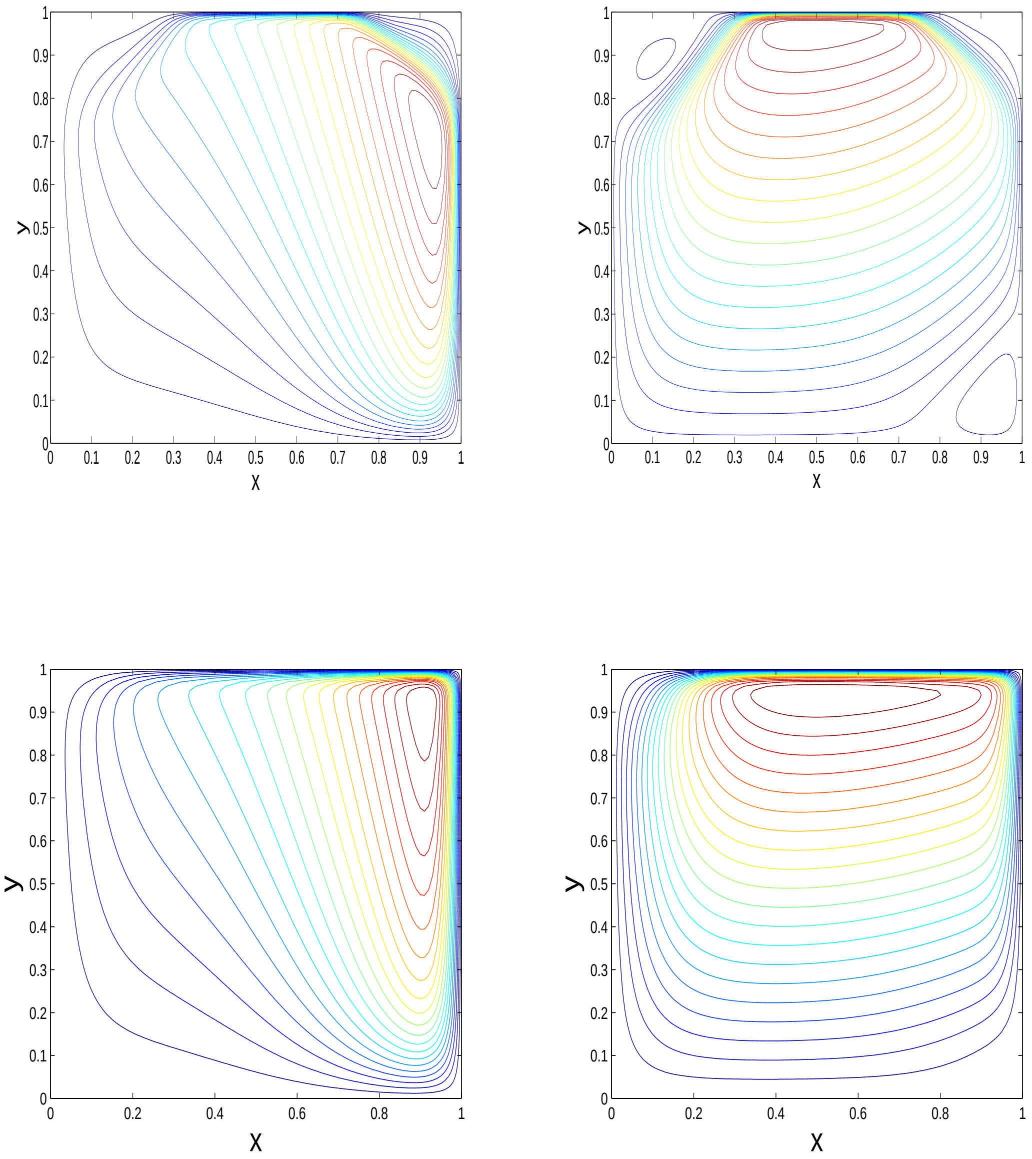


Figure 7.7: Contour plots of u(left) and v(right):first row at $t = 0.5$, second row at $t = 1.0$, for $R = 100$, $h_x = h_y = .01$ and $\Delta t = .001$ of Example 7.5.

Chapter 8

Conclusions and Future Scope

8.1 Conclusions

In this thesis, we have developed B-splines collocation and differential quadrature methods to solve some linear and nonlinear partial differential equations. These methods are designed in such a way that they reduce the given partial differential equation into a system of first order ordinary differential equations. Strong stability preserving Runge-Kutta (SSP-RK) methods of different stages and order have been used to solve resulting system of first order ordinary differential equations. SSP-RK methods provide an efficient explicit solution with high accuracy and minimal computational effort. The presented methods do not require any extra effort to tackle the nonlinearity i.e. the numerical solutions can be obtained without using the process of the transformation and linearization. Therefore, the equations are easily solved with the help of the developed techniques.

The problems considered in chapters two to seven have applications in many fields such as quantum physics, differential geometry, stability of fluid motion, atomic physics, in wave phenomena, continuum physics, mixed models of transonic flows, fluid dynamics, flow of a shock wave, phenomena of turbulence etc. The chapter wise conclusions are as follows:

In chapter 2, we have proposed a collocation method with cubic B-splines for solving Klein-Gordon and coupled Klein-Gordon-Schrödinger equations. For Neumann boundary conditions, we have used cubic B-splines and for Dirichlet boundary conditions, we have used modified cubic B-splines. The presented method approximates both the equations without using any transformation and quasi-linearization approach. The accuracy of the method is tested by taking six numerical experiments known in the literature. It has been proved that the proposed method produces better results with less CPU time in comparison to those available in the literature. This scheme not only provides solutions to grid points but at any point in the solution domain.

In chapter 3, we have presented a modified cubic B-spline collocation method for solving one dimensional sine-Gordon equation with Dirichlet boundary conditions. The approximate solutions of nonlinear sine-Gordon equation have been obtained without using any transformation and linearization process. The efficiency of the method is demonstrated by applying it on four examples. The numerical results obtained by it are quite accurate and better in comparison with the existing solutions in the literature. Order of convergence of the method is also calculated and found to be approaching two. This method produces a spline function, which may be used to obtain the solution at any point in the range of interest.

In chapter 4, cubic B-spline collocation method has been developed to solve one dimensional hyperbolic telegraph equation with Dirichlet as well as Neumann boundary conditions. The use of B-spline basis functions for spatial variable and its derivatives, results in an amenable system of differential equations. The resulting system has been solved by SSP-RK54 scheme. The accuracy of the scheme is measured by solving four numerical examples. It has been observed that the proposed method provides better results in comparison to those available in the literature and also CPU time taken is relatively less in our case. The stability of the scheme is tested with matrix stability analysis and found to be unconditionally stable. The scheme is simple and require less computational effort.

In chapter 5, a differential quadrature method has been introduced to solve two dimensional hyperbolic telegraph equation that can easily deal with Dirichlet or Mixed boundary conditions. Modified cubic B-spline basis functions were used to determine the weighting coefficients of the differential quadrature method. The telegraph equation is reduced to a system of ordinary differential equations using DQM and the resulting system is solved by SSP-RK43 scheme. In seven numerical test problems, the performance of this method is shown by computing L_2, L_∞ and relative error norms, for different time levels. The results show that the numerical solutions are very close to the exact solutions and better compared to the existing solutions found in literature with less computational cost. The stability test is also performed with matrix stability analysis and scheme is found to be unconditionally stable.

In chapter 6, we have applied a modified cubic B-spline functions based differential quadrature method to solve some linear and nonlinear wave equations with Dirichlet boundary conditions. The numerical solutions of nonlinear wave equations have been computed without using any transformation and linearization process. In numerical testing, the presented method is implemented on seven test problems such as 1D Vander Pol equation, 1D Dissipative nonlinear wave equation, 2D Vander Pol equation, 2D Dissipative nonlinear wave equation and some linear telegraph equations. The results show that the scheme gives more accurate results than earlier works with smaller grid points, larger time steps and with less computational cost. The order of convergence of the method is also calculated and found to be two. The scheme is simple and very suitable for computer implementation.

In chapter 7, the application of modified cubic B-spline differential quadrature method is discussed by solving the coupled system of Burgers' equation with appropriate initial and boundary conditions. The accuracy of the approach is tested on five test problems. The results of computations indicate that the modified cubic B-spline differential quadrature method gives more accurate results than previous works with larger time steps and with less computational efforts. The system has been solved for large Reynolds number $R = 1500$. The proposed method solves the nonlinear Burgers' equation without using a transformation or quasi-linearization approach.

The following observations have been made with respect to the B-spline collocation and differential quadrature methods:

- The B-spline collocation and differential quadrature methods are introduced along with SSP-RK schemes that are capable for solving nonlinear PDEs, without using a transformation and linearization.

- These methods are easy to use compared to other numerical methods such as finite difference and finite element methods.

- The obtained results are compared with the results available in the literature and found better or in very good agreement.
- Differential quadrature method yields higher accuracy in a smaller number of grid points.
- The order of convergence of the derived schemes are found to be close to two or more, which is quite good.

8.2 Future Scope

For the future work, the following is suggested for consideration:

- The collocation method with B-spline functions is developed for one dimensional problems. This approach can be extended to solve two or higher dimensional problems.
- The modified cubic B-spline differential quadrature method can be extended to solve higher dimensional partial differential equations appearing in various applications of science and engineering.
- Higher degree B-spline basis functions can be also modified to solve the higher order PDEs.

Bibliography

[1] S. Abbasbandy and M. T. Darvishi. A numerical solution of Burgers' equation by modified Adomian method. *Appl. Math. Comput.*, 163(3):1265–1272, 2005.

[2] M. A. Abdou and A. A. Soliman. Variational iteration method for solving Burgers' and coupled Burgers' equations. *J. Comput. Appl. Math.*, 181(2):245–251, 2005.

[3] M. A. Abdulwahhab, A. H. Bokhart, A. H. Kara, and F. D. Zaman. On the Lie point symmetry analysis and solutions of the inviscid Burgers' equation. *Pramana*, 77(3):407–414, 2011.

[4] J. H. Ahlberg and T. Ito. A collocation method for two-point boundary value problems. *Math. Comp.*, 29:761–776, 1975.

[5] J. H. Ahlberg, E. N. Nilson, and J. L. Walsh. *The theory of Splines and their applications*. Academic Press, New York-London, 1967.

[6] G. Akram and S. S. Siddiqi. Nonic spline solutions of eighth order boundary value problems. *Appl. Math. Comput.*, 182(1):829–845, 2006.

[7] E. L. Albasiny and W. D. Hoskins. Cubic spline solutions to two-point boundary value problems. *Comput. J.*, 12:151–153, 1969/1970.

[8] A. Ali, S. ul Islam, and H. Sirajul. A computational meshfree technique for the numerical solution of the two-dimensional coupled Burgers' equations. *Int. J. Comput. Methods Eng. Sci. Mech.*, 10(5):406–422, 2009.

[9] A. H. A. Ali, G. A. Gardner, and L. R. T. Gardner. A collocation solution for Burgers' equation using cubic B-spline finite elements. *Comput. Methods Appl. Mech. Engrg.*, 100(3):325–337, 1992.

[10] P. Arminjon and C. Beauchamp. Numerical solution of Burgers' equations in two space dimensions. *Comput. Methods Appl. Mech. Engrg.*, 19(3):351–365, 1979.

[11] G. Arora and B. K. Singh. Numerical solution of Burgers' equation with modified cubic B-spline differential quadrature method. *Appl. Math. Comput.*, 224:166–177, 2013.

[12] A. R. Bahadır. A fully implicit finite-difference scheme for two-dimensional Burgers' equations. *Appl. Math. Comput.*, 137(1):131–137, 2003.

[13] A. Barone, F. Esposito, C. J. Magee, and A. C. Scott. Theory and applications of the sine-Gordon equation. *La Rivista del Nuovo Cimento*, 1(2):227–267, 1971.

[14] H. Bateman. Some recent researches on the motion of fluids. *Mon. Wea. Rev.*, 43:163–170, 1915.

[15] B. Batiha, M. S. M. Noorani, and I. Hashim. Numerical solution of sine-Gordon equation by variational iteration method. *Phys. Lett. A*, 370(5-6):437–440, 2007.

[16] R. Bellman, B. Kashef, E. S. Lee, and R. Vasudevan. Solving hard problems by easy methods: differential and integral quadrature. *Comput. Math. Appl.*, 1(1):133–143, 1975.

[17] R. Bellman, B. G. Kashef, and J. Casti. Differential quadrature: a technique for the rapid solution of nonlinear partial differential equations. *J. Computational Phys.*, 10:40–52, 1972.

[18] G. Ben-Yu, P. J. Pascual, M. J. Rodríguez, and L. Vázquez. Numerical solution of the sine-Gordon equation. *Appl. Math. Comput.*, 18(1):1–14, 1986.

[19] A. H. Bhrawy and M. A. Alghamdi. Approximate solutions of Fisher's type equations with variable coefficients. *Abstr. Appl. Anal.*, 2013:Article ID 176730, 10 pages, 2013.

[20] J. Biazar and H. Aminikhah. Exact and numerical solutions for non-linear Burger's equation by VIM. *Math. Comput. Modelling*, 49(7-8):1394–1400, 2009.

[21] W. G. Bickley. Piecewise cubic interpolation and two-point boundary problems. *Comput. J.*, 11:206–208, 1968/1969.

[22] G. Birkhoff and H. L. Garabedian. Smooth surface interpolation. *J. Math. and Phys.*, 39:258–268, 1960.

[23] A. G. Bratsos. A fourth order numerical scheme for the one-dimensional sine-Gordon equation. *Int. J. Comput. Math.*, 85(7):1083–1095, 2008.

[24] A. G. Bratsos. On the numerical solution of the Klein-Gordon equation. *Numer. Methods Partial Differential Equations*, 25(4):939–951, 2009.

[25] A. G. Bratsos and E. H. Twizell. The solution of the sine-Gordon equation using the method of lines. *Int. J. Comput. Math.*, 61(3-4):271–292, 1996.

[26] A. G. Bratsos and E. H. Twizell. A family of parametric finite-difference methods for the solution of the sine-Gordon equation. *Appl. Math. Comput.*, 93(2-3):117–137, 1998.

[27] B. Bülbül and M. Sezer. A Taylor matrix method for the solution of a two-dimensional linear hyperbolic equation. *Appl. Math. Lett.*, 24(10):1716–1720, 2011.

[28] B. Bülbül and M. Sezer. Taylor polynomial solution of hyperbolic type partial differential equations with constant coefficients. *Int. J. Comput. Math.*, 88(3):533–544, 2011.

[29] J. M. Burgers. A mathematical model illustrating the theory of turbulence. 1:171–199, 1948.

[30] H. N. Caglar, S. H. Caglar, and E. H. Twizell. The numerical solution of fifth-order boundary value problems with sixth-degree B-spline functions. *Appl. Math. Lett.*, 12(5):25–30, 1999.

[31] H. N. Caglar, S. H. Caglar, and E. H. Twizell. The numerical solution of third-order boundary-value problems with fourth-degree B-spline functions. *Int. J. Comput. Math.*, 71(3):373–381, 1999.

[32] J. Caldwell, P. Wanless, and A. E. Cook. A finite element approach to Burgers' equation. *Appl. Math. Model.*, 5(3):189–193, 1981.

[33] P. J. Caudrey, J. C. Eilbeck, and J. D. Gibbon. The sine-Gordon equation as a model classical field theory. *Il Nuovo Cimento B (11)*, 25(2):497–512, 1975.

[34] H. Chen and Z. Jiang. A characteristics-mixed finite element method for Burgers' equation. *Korean J. Comput. Appl. Math.*, 15(1-2):29–51, 2004.

[35] J. D. Cole. On a quasi-linear parabolic equation occurring in aerodynamics. *Quart. Appl. Math.*, 9:225–236, 1951.

[36] J. Crank and R. S. Gupta. A method for solving moving boundary problems in heat-flow using cubic splines or polynomials. *J. Inst. Math. Appl.*, 10:296–304, 1972.

[37] İ. Dağ, A. Doğan, and B. Saka. B-spline collocation methods for numerical solutions of the RLW equation. *Int. J. Comput. Math.*, 80(6):743–757, 2003.

[38] İ. Dağ and M. N. Özer. Approximation of the RLW equation by the least square cubic B-spline finite element method. *Appl. Math. Model.*, 25(3):221–231, 2001.

[39] İ. Dağ and B. Saka. A cubic B-spline collocation method for the EW equation. *Math. Comput. Appl.*, 9(3):381–392, 2004.

[40] C. de Boor. *A practical guide to splines*, volume 27 of *Applied Mathematical Sciences*. Springer-Verlag, New York, revised edition, 2001.

[41] E. Y. Deeba and S. A. Khuri. A decomposition method for solving the nonlinear Klein-Gordon equation. *J. Comput. Phys.*, 124(2):442–448, 1996.

[42] M. Dehghan and A. Ghesmati. Application of the dual reciprocity boundary integral equation technique to solve the nonlinear Klein-Gordon equation. *Comput. Phys. Comm.*, 181(8):1410–1418, 2010.

[43] M. Dehghan and A. Ghesmati. Combination of meshless local weak and strong (MLWS) forms to solve the two dimensional hyperbolic telegraph equation. *Eng. Anal. Bound. Elem.*, 34(4):324–336, 2010.

[44] M. Dehghan and A. Ghesmati. Solution of the second-order one-dimensional hyperbolic telegraph equation by using the dual reciprocity boundary integral equation (DRBIE) method. *Eng. Anal. Bound. Elem.*, 34(1):51–59, 2010.

[45] M. Dehghan and M. Lakestani. Numerical solution of nonlinear system of second-order boundary value problems using cubic B-spline scaling functions. *Int. J. Comput. Math.*, 85(9):1455–1461, 2008.

[46] M. Dehghan and M. Lakestani. The use of Chebyshev cardinal functions for solution of the second-order one-dimensional telegraph equation. *Numer. Methods Partial Differential Equations*, 25(4):931–938, 2009.

[47] M. Dehghan and D. Mirzaei. The boundary integral equation approach for numerical solution of the one-dimensional sine-Gordon equation. *Numer. Methods Partial Differential Equations*, 24(6):1405–1415, 2008.

[48] M. Dehghan and A. Mohebbi. The combination of collocation, finite difference, and multigrid methods for solution of the two-dimensional wave equation. *Numer. Methods Partial Differential Equations*, 24(3):897–910, 2008.

[49] M. Dehghan and A. Mohebbi. High order implicit collocation method for the solution of two-dimensional linear hyperbolic equation. *Numer. Methods Partial Differential Equations*, 25(1):232–243, 2009.

[50] M. Dehghan, A. Mohebbi, and Z. Asgari. Fourth-order compact solution of the nonlinear Klein-Gordon equation. *Numer. Algorithms*, 52(4):523–540, 2009.

[51] M. Dehghan and A. Nikpour. Numerical solution of the system of second-order boundary value problems using the local radial basis functions based differential quadrature collocation method. *Appl. Math. Model.*, 37(18-19):8578–8599, 2013.

[52] M. Dehghan and R. Salehi. A method based on meshless approach for the numerical solution of the two-space dimensional hyperbolic telegraph equation. *Math. Methods Appl. Sci.*, 35(10):1220–1233, 2012.

[53] M. Dehghan and A. Shokri. A numerical method for one-dimensional nonlinear sine-Gordon equation using collocation and radial basis functions. *Numer. Methods Partial Differential Equations*, 24(2):687–698, 2008.

[54] M. Dehghan and A. Shokri. A numerical method for solving the hyperbolic telegraph equation. *Numer. Methods Partial Differential Equations*, 24(4):1080–1093, 2008.

[55] M. Dehghan and A. Shokri. A meshless method for numerical solution of a linear hyperbolic equation with variable coefficients in two space dimensions. *Numer. Methods Partial Differential Equations*, 25(2):494–506, 2009.

[56] M. Dehghan and A. Shokri. Numerical solution of the nonlinear Klein-Gordon equation using radial basis functions. *J. Comput. Appl. Math.*, 230(2):400–410, 2009.

[57] M. Dehghan, S. A. Yousefi, and A. Lotfi. The use of He's variational iteration method for solving the telegraph and fractional telegraph equations. *Int. J. Numer. Methods Biomed. Eng.*, 27(2):219–231, 2011.

[58] H. Ding and Y. Zhang. A new fourth-order compact finite difference scheme for the two-dimensional second-order hyperbolic equation. *J. Comput. Appl. Math.*, 230(2):626–632, 2009.

[59] H. Ding and Y. Zhang. A new unconditionally stable compact difference scheme of o for the 1D linear hyperbolic equation. *Appl. Math. Comput.*, 207(1):236–241, 2009.

[60] R. K. Dodd, J. C. Eilbeck, J. D. Gibbon, and H. C. Morris. *Solitons and nonlinear wave equations.* 1982.

[61] E. H. Doha, D. Baleanu, A. H. Bhrawy, and M. A. Abdelkawy. A Jacobi collocation method for solving nonlinear Burgers-type equations. *Abstr. Appl. Anal.*, 2013:12, 2013.

[62] E. H. Doha, A. H. Bhrawy, M. A. Abdelkawy, and R. M Hafez. A Jacobi collocation

approximation for nonlinear coupled viscous Burgers' equation. *Cent. Eur. J. Phys.*, 12(2):111–122, 2014.

[63] E. H. Doha, A. H. Bhrawy, D. Baleanu, and M. A. Abdelkawy. Numerical treatment of coupled nonlinear hyperbolic Klein-Gordon equations. *Rom. Journ, Phys.*, 59(3-4):247–264, 2014.

[64] M. Dosti and A. Nazemi. Quartic B-spline collocation method for solving one-dimensional hyperbolic telegraph equation. *J. Inform. Comput. Sci.*, 7(2):083–090, 2012.

[65] Y. Duan and R. Liu. Lattice Boltzmann model for two-dimensional unsteady Burgers' equation. *J. Comput. Appl. Math.*, 206(1):432–439, 2007.

[66] H. M. El-Hawary and E. O. Abdel-Rahman. Numerical solution of the generalized Burgers' equation via spectral/spline methods. *Appl. Math. Comput.*, 170(1):267–279, 2005.

[67] S. M. El-Sayed. The decomposition method for studying the Klein-Gordon equation. *Chaos Solitons Fractals*, 18(5):1025–1030, 2003.

[68] A. Esen, O. Tasbozan, and S. Kutluay. Applications of the exp-function method for the MkdV-sine-Gordon and Boussinesq-double sine-Gordon equations. *World Applied Sciences Journal*, 22(1):147–151, 2013.

[69] M. Esmaeilbeigi, M. M. Hosseini, and S. T. Mohyud-Din. A new approach of the radial basis functions method for telegraph equations. *Int. J. Phys. Sci.*, 6(6):1517–1527, 2011.

[70] C. A. J. Fletcher. A comparison of finite element and finite difference solutions of the one and two-dimensional Burgers' equations. *J. Comput. Phys.*, 51(1):159–188, 1983.

[71] C. A. J. Fletcher. Generating exact solutions of the two-dimensional Burgers' equations. *Int. J. Numer. Meth. Fl.*, 3(3):213–216, 1983.

[72] F. Gao and C. Chi. Unconditionally stable difference schemes for a one space dimensional linear hyperbolic equation. *Appl. Math. Comput.*, 187(2):1272–1276, 2007.

[73] S. Gottlieb, Chi-Wang Shu, and E. Tadmor. Strong stability-preserving high-order time discretization methods. *SIAM Rev.*, 43(1):89–112, 2001.

[74] K. Goyal and M. Mehra. A fast adaptive diffusion wavelet method for Burger's equation. *Comput. Math. Appl.*, 68(4):568–577, 2014.

[75] C. Grossmann and H. G. Roos. *Numerical treatment of partial differential equations.* Springer, Berlin, 2007.

[76] Z. Guo-Zhong, Y. Xi-Jun, and W. Di. Numerical solution of the Burgers' equation by local discontinuous Galerkin method. *Appl. Math. Comput.*, 216(12):3671–3679, 2010.

[77] Z. Guo-Zhong, Y. Xi-jun, and Z. Rongpei. The new numerical method for solving the system of two-dimensional Burgers' equations. *Comput. Math. Appl.*, 62(8):3279–3291, 2011.

[78] J. Hong, S. Jiang, L. Kong, and C. Li. Numerical comparison of five difference schemes for coupled Klein-Gordon-Schrödinger equations in quantum physics. *J. Phys. A*, 40(30):9125–9135, 2007.

[79] E. Hopf. The partial differential equation $u_t + uu_x = \mu u_{xx}$. *Comm. Pure Appl. Math.*, 3:201–230, 1950.

[80] S. R. K. Iyengar and P. Jain. Spline finite difference methods for singular two point boundary value problems. *Numer. Math.*, 50(3):363–376, 1986.

[81] S. R. K. Iyengar and R. C. Mittal. High order difference schemes for the wave equation. *Internat. J. Numer. Methods Engrg.*, 12(10):1623–1628, 1978.

[82] M. K. Jain, R. Ahuja, and S. Bhattacharyya. Difference schemes for second order hyperbolic equations. *Internat. J. Numer. Methods Engrg.*, 10(4):960–964, 1976.

[83] M. K. Jain and T. Aziz. Spline function approximation for differential equations. *Comput. Methods Appl. Mech. Engrg.*, 26(2):129–143, 1981.

[84] M. K. Jain and T. Aziz. Cubic spline solution of two-point boundary value problems with significant first derivatives. *Comput. Methods Appl. Mech. Engrg.*, 39(1):83–91, 1983.

[85] M. K. Jain, S. R. K. Iyengar, and A. C. R. Pillai. Difference schemes based on splines in compression for the solution of conservation laws. *Comput. Methods Appl. Mech. Engrg.*, 38(2):137–151, 1983.

[86] P. C. Jain and D. N. Holla. Numerical solutions of coupled Burgers' equation. *Int. J. Nonlinear Mech.*, 13(4):213–222, 1978.

[87] Zi-Wu Jiang and Ren-Hong Wang. Numerical solution of one-dimensional sine-Gordon equation using high accuracy multiquadric quasi-interpolation. *Appl. Math. Comput.*, 218(15):7711–7716, 2012.

[88] S. Jiménez and L. Vázquez. Analysis of four numerical schemes for a nonlinear Klein-Gordon equation. *Appl. Math. Comput.*, 35(1, part I):61–94, 1990.

[89] R. 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. *Appl. Math. Comput.*, 219(12):6680–6691, 2013.

[90] R. Jiwari, S. Pandit, and R. C. Mittal. A differential quadrature algorithm for the numerical solution of the second-order one dimensional hyperbolic telegraph equation. *Int. J. Nonlinear Sci.*, 13(3):259–266, 2012.

[91] R. Jiwari, S. Pandit, and R. C. Mittal. A differential quadrature algorithm to solve the two dimensional linear hyperbolic telegraph equation with Dirichlet and Neumann boundary conditions. *Appl. Math. Comput.*, 218(13):7279–7294, 2012.

[92] M. K. Kadalbajoo and V. K. Aggarwal. Fitted mesh B-spline collocation method for solving self-adjoint singularly perturbed boundary value problems. *Appl. Math. Comput.*, 161(3):973–987, 2005.

[93] M. K. Kadalbajoo and P. Arora. B-spline collocation method for the singular-perturbation problem using artificial viscosity. *Comput. Math. Appl.*, 57(4):650–663, 2009.

[94] M. K. Kadalbajoo and P. Arora. Taylor-Galerkin B-spline finite element method for the one-dimensional advection-diffusion equation. *Numer. Methods Partial Differential Equations*, 26(5):1206–1223, 2010.

[95] M. K. Kadalbajoo and V. Gupta. Numerical solution of singularly perturbed convection-diffusion problem using parameter uniform B-spline collocation method. *J. Math. Anal. Appl.*, 355(1):439–452, 2009.

[96] M. K. Kadalbajoo and A. S. Yadaw. B-spline collocation method for a two-parameter singularly perturbed convection-diffusion boundary value problems. *Appl. Math. Comput.*, 201(1-2):504–513, 2008.

[97] K. N. S. Kasi Viswanadham and S. Ballem. Numerical solution of tenth order boundary value problems by Galerkin method with sextic B-splines. *Int. J. Appl. Math. Stat. Sci.*, 3(3):17–30, 2014.

[98] K. N. S. Kasi Viswanadham and S. R. Koneru. Finite element method for one-dimensional and two-dimensional time dependent problems with B-splines. *Comput. Methods Appl. Mech. Engrg.*, 108(3-4):201–222, 1993.

[99] K. N. S. Kasi Viswanadham and P. M. Krishna. Quintic B-splines Galerkin method for fifth order boundary value problems,. *ARPN J. Eng. Appl. Sci.*, 5(2):74–77, 2010.

[100] K. N. S. Kasi Viswanadham and P. M. Krishna. Septic B-spline collocation method for sixth order boundary value problems. *ARPN J. Eng. Appl. Sci.*, 5(7):36–40, 2010.

[101] D. Kaya. A numerical solution of the sine-Gordon equation using the modified decomposition method. *Appl. Math. Comput.*, 143(2-3):309–317, 2003.

[102] D. Kaya and S. M. El-Sayed. A numerical solution of the Klein-Gordon equation and convergence of the decomposition method. *Appl. Math. Comput.*, 156(2):341–353, 2004.

[103] M. E. Khalifa and M. Elgamal. A numerical solution to Klein-Gordon equation with Dirichlet boundary condition. *Appl. Math. Comput.*, 160(2):451–475, 2005.

[104] L. Kong, J. Hong, and R. Liu. Long-term numerical simulation of the interaction between a neutron field and a neutral meson field by a symplectic-preserving scheme. *J. Phys. A: Math. Theor.*, 41(25):255207, 19, 2008.

[105] L. Kong, R. Liu, and Z. Xu. Numerical simulation of interaction between Schrödinger field and Klein-Gordon field by multisymplectic method. *Appl. Math. Comput.*, 181(1):342–350, 2006.

[106] L. Kong, J. Zhang, Y. Cao, Y. Duan, and H. Huang. Semi-explicit symplectic partitioned Runge-Kutta Fourier pseudo-spectral scheme for Klein-Gordon-Schrödinger equations. *Comput. Phys. Comm.*, 181(8):1369–1377, 2010.

[107] A. Korkmarz and İ. Dağ. A differential quadrature algorithm for simulations of non-linear Schrödinger equation. *Comput. Math. Appl.*, 56(9):2222–2234, 2008.

[108] A. Korkmaz and İ. Dağ. Polynomial based differential quadrature method for numerical solution of nonlinear Burgers' equation. *J. Franklin Inst.*, 348(10):2863–2875, 2011.

[109] A. Korkmaz and İ. Dağ. Shock wave simulations using sinc differential quadrature method. *Eng. Computations*, 28(6):654–674, 2011.

[110] A. Korkmaz and İ. Dağ. Cubic B-spline differential quadrature methods and stability for Burgers' equation. *Eng. Computation*, 30(3):320–344, 2013.

[111] A. Korkmaz and İ. Dağ. Numerical simulations of boundary-forced RLW equation with cubic B-spline-based differential quadrature methods. *Arab. J. Sci. Eng.*, 38(5):1151–1160, 2013.

[112] A. Korkmaz, A. Murat, and İ. Dağ. Quartic B-spline differential quadrature method. *Int. J. Nonlinear Sci.*, 11(4):403–411, 2011.

[113] J. X. Kuang and L. H. Lu. Two classes of finite-difference methods for generalized sine-Gordon equations. *J. Comput. Appl. Math.*, 31(3):389–396, 1990.

[114] M. Kumar. A second order spline finite difference method for singular two-point boundary value problems. *Appl. Math. Comput.*, 142(2-3):283–290, 2003.

[115] M. Kumar. Higher order method for singular boundary-value problems by using spline function. *Appl. Math. Comput.*, 192(1):175–179, 2007.

[116] M. Kumar and P. K. Srivastava. Computational techniques for solving differential equations by quadratic, quartic and octic spline. *Adv. Eng. Softw.*, 39(8):646–653, 2008.

[117] M. Kumar and P. K. Srivastava. Computational techniques for solving differential equations by cubic, quintic, and sextic spline. *Int. J. Comput. Methods Eng. Sci. Mech.*, 10(1):108–115, 2009.

[118] V. Kumar, R. Jiwari, and R. K. Gupta. Numerical simulation of two dimensional quasilinear hyperbolic equations by polynomial differential quadrature method. *Eng. Computations*, 30(7):892–909, 2013.

[119] S. Kutluay and A. Esen. A B-spline finite element method for the thermistor problem with the modified electrical conductivity. *Appl. Math. Comput.*, 156(3):621–632, 2004.

[120] S. Kutluay and A. Esen. A linearized numerical scheme for Burgers-like equations. *Appl. Math. Comput.*, 156(2):295–305, 2004.

[121] S. Kutluay, A. Esen, and İ Dağ. Numerical solutions of the Burgers' equation by the least-squares quadratic B-spline finite element method. *J. Comput. Appl. Math.*, 167(1):21–33, 2004.

[122] M. Lakestani and M. Dehghan. Numerical solution of Fokker-Planck equation using the cubic B-spline scaling functions. *Numer. Methods Partial Differential Equations*, 25(2):418–429, 2009.

[123] M. Lakestani and M. Dehghan. Collocation and finite difference-collocation methods for the solution of nonlinear Klein-Gordon equation. *Comput. Phys. Comm.*, 181(8):1392–1401, 2010.

[124] M. Lakestani and M. Dehghan. Numerical solution of Riccati equation using the cubic B-spline scaling functions and Chebyshev cardinal functions. *Comput. Phys. Comm.*, 181(5):957–966, 2010.

[125] M. Lakestani and M. Dehghan. Numerical solutions of the generalized Kuramoto-Sivashinsky equation using B-spline functions. *Appl. Math. Model.*, 36(2):605–617, 2012.

[126] M. Lakestani and M. Dehghan. Four techniques based on the B-spline expansion and the collocation approach for the numerical solution of the Lane-Emden equation. *Math. Methods Appl. Sci.*, 36(16):2243–2253, 2013.

[127] M. Lakestani, M. Dehghan, and S. Irandoust-pakchin. The construction of operational matrix of fractional derivatives using B-spline functions. *Commun. Nonlinear Sci. Numer. Simul.*, 17(3):1149–1162, 2012.

[128] M. Lakestani and B. N. Saray. Numerical solution of telegraph equation using interpolating scaling functions. *Comput. Math. Appl.*, 60(7):1964–1972, 2010.

[129] I. J. Lee. Numerical solution for nonlinear Klein-Gordon equation by collocation method with respect to spectral method. *J. Korean Math. Soc.*, 32(3):541–551, 1995.

[130] Q. Li, Z. Ji, Z. Zheng, and H. Liu. Numerical solution of nonlinear Klein-Gordon equation using lattice boltzmann method. *Applied Mathematics*, 2(12):1479–1485, 2011.

[131] M. Li-Min and W. Zong-Min. A numerical method for one-dimensional nonlinear sine-Gordon equation using multiquadric quasi-interpolation. *Chin. Phys. B*, 18(8):3099–3103, 2011.

[132] W. Liao. A fourth-order finite-difference method for solving the system of two-dimensional Burgers' equations. *Int. J. Numer. Meth. Fl.*, 64(5):565–590, 2010.

[133] F. Liu and W. Shi. Numerical solutions of two-dimensional Burgers' equations by lattice Boltzmann method. *Commun. Non-linear Sci. Numer. Simul.*, 16(1):150–157, 2011.

[134] H. W. Liu and L. B. Liu. An unconditionally stable spline difference scheme of $O(k^2 + h^4)$ for solving the second-order 1D linear hyperbolic equation. *Math. Comput. Modelling*, 49(9-10):1985–1993, 2009.

[135] Li-Bin Liu and Huan-Wen Liu. Compact difference schemes for solving telegraphic equations with Neumann boundary conditions. *Applied Mathematics and Computation*, 219(19):10112 – 10121, 2013.

[136] F. R. Loscalzo and T. D. Talbot. Spline function approximations for solutions of ordinary differential equations. *SIAM J. Numer. Anal.*, 4:433–445, 1967.

[137] P. S. Mantri, N. Nataraj, and A. K. Pani. A qualocation method for Burgers' equation. *J. Comput. Appl. Math.*, 213(1):1–13, 2008.

[138] R. C. Mittal and G. Arora. Efficient numerical solution of Fisher's equation by using B-spline method. *Int. J. Comput. Math.*, 87(13):3039–3051, 2010.

[139] R. C. Mittal and G. Arora. Quintic B-spline collocation method for numerical solution of the Kuramoto-Sivashinsky equation. *Commun. Nonlinear Sci. Numer. Simul.*, 15(10):2798–2808, 2010.

[140] R. C. Mittal and G. Arora. Numerical solution of the coupled viscous Burgers' equation. *Commun. Nonlinear Sci. Numer. Simul.*, 16(3):1304–1313, 2011.

[141] R. C. Mittal and R. K. Jain. B-splines methods with redefined basis functions for solving fourth order parabolic partial differential equations. *Appl. Math. Comput.*, 217(23):9741–9755, 2011.

[142] R. C. Mittal and R. K. Jain. Application of quintic B-splines collocation method on some Rosenau type nonlinear higher order evolution equations. *Int. J. Nonlinear Sci.*, 13(2):142–152, 2012.

[143] R. C. Mittal and R. K. Jain. Cubic B-splines collocation method for solving non-linear parabolic partial differential equations with Neumann boundary conditions. *Commun. Nonlinear Sci. Numer. Simul.*, 17(12):4616–4625, 2012.

[144] R. C. Mittal and R. K. Jain. Numerical solutions of nonlinear Burgers' equation with modified cubic B-splines collocation method. *Appl. Math. Comput.*, 218(15):7839–7855, 2012.

[145] R. C. Mittal and R. K. Jain. Redefined cubic B-splines collocation method for solving convection-diffusion equations. *Appl. Math. Model.*, 36(11):5555–5573, 2012.

[146] R. C. Mittal and R. K. Jain. Numerical solutions of nonlinear Fisher's reaction-diffusion equation with modified cubic B-spline collocation method. *Math. Sci.*, 7:Art. 12, 10, 2013.

[147] R. C. Mittal and R. Jiwari. Differential quadrature method for two-dimensional Burgers' equations. *Int. J. Comput. Methods Eng. Sci. Mech.*, 10(6):450–459, 2009.

[148] R. C. Mittal and P. Singhal. Numerical solution of Burger's equation. *Commun. Numer. Meth. En.*, 9(5):397–406, 1993.

[149] R. K. Mohanty. An operator splitting method for an unconditionally stable difference scheme for a linear hyperbolic equation with variable coefficients in two space dimensions. *Appl. Math. Comput.*, 152(3):799–806, 2004.

[150] R. K. Mohanty. An unconditionally stable difference scheme for the one-space-dimensional linear hyperbolic equation. *Appl. Math. Lett.*, 17(1):101–105, 2004.

[151] R. K. Mohanty. An unconditionally stable finite difference formula for a linear second order one space dimensional hyperbolic equation with variable coefficients. *Appl. Math. Comput.*, 165(1):229–236, 2005.

[152] R. K. Mohanty. New unconditionally stable difference schemes for the solution of multi-dimensional telegraphic equations. *Int. J. Comput. Math.*, 86(12):2061–2071, 2009.

[153] R. K. Mohanty and V. Gopal. High accuracy cubic spline finite difference approximation for the solution of one-space dimensional non-linear wave equations. *Appl. Math. Comput.*, 218(8):4234 – 4244, 2011.

[154] R. K. Mohanty and V. Gopal. A fourth-order finite difference method based on spline in tension approximation for the solution of one-space dimensional second-order quasi-linear hyperbolic equations. *Adv. Difference Equ.*, 2013(1), 2013.

[155] R. K. Mohanty and M. K. Jain. An unconditionally stable alternating direction implicit scheme for the two space dimensional linear hyperbolic equation. *Numer. Methods Partial Differential Equations*, 17(6):684–688, 2001.

[156] R. K. Mohanty, M. K. Jain, and U. Arora. An unconditionally stable ADI method for the linear hyperbolic equation in three space dimensions. *Int. J. Comput. Math.*, 79(1):133–142, 2002.

[157] R. K. Mohanty, M. K. Jain, and K. George. High order difference schemes for the system of two space second order nonlinear hyperbolic equations with variable coefficients. *J. Comput. Appl. Math.*, 70(2):231–243, 1996.

[158] R. K. Mohanty, M. K. Jain, and K. George. On the use of high order difference methods for the system of one space second order nonlinear hyperbolic equations with variable coefficients. *J. Comput. Appl. Math.*, 72(2):421–431, 1996.

[159] R. K. Mohanty, M. K. Jain, and S. Singh. A new three-level implicit cubic spline method for the solution of 1D quasi-linear hyperbolic equations. *Comput. Math. Model.*, 24(3):452–470, 2013.

[160] R. K. Mohanty and S. Singh. High order variable mesh approximation for the solution of 1D non-linear hyperbolic equation. *Int. J. Nonlinear Sci.*, 14:220–227, 2012.

[161] A. Mohebbi and M. Dehghan. High order compact solution of the one-space-dimensional linear hyperbolic equation. *Numer. Methods Partial Differential Equations*, 24(5):1222–1235, 2008.

[162] A. Mohebbi and M. Dehghan. High-order solution of one-dimensional sine-Gordon equation using compact finite difference and DIRKN methods. *Math. Comput. Modelling*, 51(5-6):537–549, 2010.

[163] K. W. Morton and D. F. Mayers. *Numerical solution of partial differential equations*. Cambridge University Press, Cambridge, second edition, 2005.

[164] T. Özis, A. Esen, and S. Kutluay. Numerical solution of Burgers' equation by quadratic B-spline finite elements. *Appl. Math. Comput.*, 165(1):237–249, 2005.

[165] K. Pandey, L. Verma, and A. K. Verma. On a finite difference scheme for Burgers' equation. *Appl. Math. Comput.*, 215(6):2206–2214, 2009.

[166] A. K. Pany, N. Nataraj, and S. Singh. A new mixed finite element method for Burgers' equation. *J. Appl. Math. Comp.*, 23(1-2):43–55, 2007.

[167] J. K. Perring and T. H. R. Skyrme. A model unified field equation. *Nuclear Phys.*, 31:550–555, 1962.

[168] P. M. Prenter. *Splines and variational methods*. Wiley-Interscience [John Wiley & Sons], New York-London-Sydney, 1975. Pure and Applied Mathematics.

[169] J. R. Quan and C. T. Chang. New insights in solving distributed system equations by the quadrature method-I. *Comput. Chem. Eng.*, 13(7):779–788, 1989.

[170] J. R. Quan and C. T. Chang. New insights in solving distributed system equations by the quadrature method-II. *Comput. Chem. Eng.*, 13(9):1017–1024, 1989.

[171] S. F. Radwan. On the fourth order accurate compact ADI scheme for solving the unsteady non-linear coupled Burgers' equations. *J. Nonlinear Math. Phys.*, 6(1):13–34, 1999.

[172] S. F. Radwan. Comparison of higher-order accurate schemes for solving the two-dimensional unsteady Burgers' equation. *J. Comput. Appl. Math.*, 174(2):383–397, 2005.

[173] S. S. Rao. *The Finite Element Method in Engineering.* Elsevier Inc., 1982.

[174] J. Rashidinia, F. Esfahani, and S. Jamalzadeh. B-spline collocation approach for solution of Klein-Gordon equation. *Inter. J. Math. Model. Comp.*, 3(1):25–33, 2013.

[175] J. Rashidinia, M. Ghasemi, and R. Jalilian. Numerical solution of the nonlinear Klein-Gordon equation. *J. Comput. Appl. Math.*, 233(8):1866–1878, 2010.

[176] J. Rashidinia, R. Jalilian, and V. Kazemi. Spline methods for the solutions of hyperbolic equations. *Appl. Math. Comput.*, 190(1):882–886, 2007.

[177] J. Rashidinia and R. Mohammadi. Tension spline approach for the numerical solution of nonlinear Klein-Gordon equation. *Comput. Phys. Comm.*, 181(1):78–91, 2010.

[178] J. Rashidinia and R. Mohammadi. Tension spline solution of nonlinear sine-Gordon equation. *Numer. Algorithms*, 56(1):129–142, 2011.

[179] K. R. Raslan. A computational method for the equal width equation. *Int. J. Comput. Math.*, 81(1):63–72, 2004.

[180] K. R. Raslan. Collocation method using quartic B-spline for the equal width (EW) equation. *Appl. Math. Comput.*, 168(2):795–805, 2005.

[181] K. R. Raslan. A computational method for the regularized long wave (RLW) equation. *Appl. Math. Comput.*, 167(2):1101–1118, 2005.

[182] A. S. V. Ravi Kanth and K. Aruna. Differential transform method for solving the linear and nonlinear Klein-Gordon equation. *Comput. Phys. Comm.*, 180(5):708–711, 2009.

[183] A. S. V. Ravi Kanth and Y. N. Reddy. Cubic spline for a class of singular two-point boundary value problems. *Appl. Math. Comput.*, 170(2):733–740, 2005.

[184] Z. Rong-Pei, Y. Xi-Jun, and Z. Guo-Zhong. Modified Burgers' equation by the local discontinuous Galerkin method. *Chin. Phys. B*, 22(3):030210, 2013.

[185] S. G. Rubin and P. K. Khosla. Higher-order numerical solutions using cubic splines. *AIAA J.*, 14(7):851–858, 1976.

[186] A. Saadatmandi and M. Dehghan. Numerical solution of hyperbolic telegraph equation using the Chebyshev tau method. *Numer. Methods Partial Differential Equations*, 26(1):239–252, 2010.

[187] B. Saka and İ. Dağ. A collocation method for the numerical solution of the RLW equation using cubic B-spline basis. *Arab. J. Sci. Eng. Sect. A Sci.*, 30(1):39–50, 2005.

[188] S. S. Sastry. Finite difference approximations to one-dimensional parabolic equations using a cubic spline technique. *J. Comput. Appl. Math.*, 2(1):23–26, 1976.

[189] I. J. Schoenberg. Contributions to the problem of approximation of equidistant data by analytic functions. Part A. On the problem of smoothing or graduation. A first class of analytic approximation formulae. *Quart. Appl. Math.*, 4:45–99, 1946.

[190] I. J. Schoenberg. Spline functions, convex curves and mechanical quadrature. *Bull. Amer. Math. Soc.*, 64:352–357, 1958.

[191] L. L. Schumaker. *Spline functions: basic theory.* John Wiley & Sons, Inc., New York, 1981. Pure and Applied Mathematics, A Wiley-Interscience Publication.

[192] C. Shu. *Differential Quadrature and Its Application in Engineering.* Springer-Verlag London Ltd., Great Britain, 2000.

[193] C. Shu and Y. L. Wu. Integrated radial basis functions-based differential quadrature method and its performance. *Int. J. Numer. Meth. Fl.*, 53(6):969–984, 2007.

[194] C. Shu and H. Xue. Explicit computation of weighting coefficients in the the harmonic differential quadrature. *J. Sound Vib.*, 204(3):549–555, 1997.

[195] S. S. Siddiqi and E. H. Twizell. Spline solutions of linear twelfth-order boundary-value problems. *J. Comput. Appl. Math.*, 78(2):371–390, 1997.

[196] A. A. Soliman. The modified extended tanh-function method for solving Burgers'-type equations. *Physica A: Statistical Mechanics and its Applications*, 361(2):394–404, 2006.

[197] R. J. Spiteri and S. J. Ruuth. A new class of optimal high-order strong-stability-preserving time discretization methods. *SIAM J. Numer. Anal.*, 40(2):469–491, 2002.

[198] A. G. Striz, X. Wang, and C. W. Bert. Harmonic differential quadrature method and applications to analysis of structural components. *Acta Mech.*, 111(1-2):85–94, 1995.

[199] F. Toutounian and E. Tohidi. A new Bernoulli matrix method for solving second order linear partial differential equations with the convergence analysis. *Appl. Math. Comput.*, 223(0):298–310, 2013.

[200] M. Uddin, S. Haq, and G. Qasim. A meshfree approach for the numerical solution of nonlinear sine-Gordon equation. *Int. Math. Forum*, 7(21-24):1179–1186, 2012.

[201] R. A. Usmani. The use of quartic splines in the numerical solution of a fourth-order boundary value problem. *J. Comput. Appl. Math.*, 44(2):187–199, 1992.

[202] R. A. Usmani and M. Sakai. Quartic spline solutions for two-point boundary problems involving third order differential equations. *J. Math. Phys. Sci.*, 18(4):365–380, 1984.

[203] R. A. Usmani and S. A. Warsi. Quintic spline solutions of boundary value problems. *Comput. Math. Appl.*, 6(2):197–203, 1980.

[204] M. Wang and Y. Zhou. The periodic wave solutions for the Klein-Gordon-Schrödinger equations. *Phys. Lett. A*, 318(1-2):84–92, 2003.

[205] Q. Wang and D. Cheng. Numerical solution of damped nonlinear Klein-Gordon equations using variational method and finite element approach. *Appl. Math. Comput.*, 162(1):381–401, 2005.

[206] S. Wang and L. Zhang. A class of conservative orthogonal spline collocation schemes for solving coupled Klein-Gordon-Schrödinger equations. *Appl. Math. Comput.*, 203(2):799–812, 2008.

[207] G. W. Wei. Discrete singular convolution for the sine-Gordon equation. *Phys. D*, 137(3-4):247–259, 2000.

[208] G. W. Wei, D. S. Zhang, D. J. Kouri, and D. K. Hoffman. Distributed approximating functional approach to Burgers' equation in one and two space dimensions. *Comput. Phys. Comm.*, 111(1-3):93–109, 1998.

[209] D. L. Young, C. M. Fan, S. P. Hu, and S. N. Atluri. The Eulerian-Lagrangian method of fundamental solutions for two-dimensional unsteady Burgers' equations. *Eng. Anal. Bound. Elem.*, 32(5):395–412, 2008.

[210] L. Zhang. Convergence of a conservative difference scheme for a class of Klein-Gordon-Schrödinger equations in one space dimension. *Appl. Math. Comput.*, 163(1):343–355, 2005.

[211] X. H. Zhang, J. Ouyang, and L. Zhang. Element-free characteristic Galerkin method for Burgers' equation. *Eng. Anal. Bound. Elem.*, 33(3):356–362, 2009.

[212] C. Zheng. Numerical solution to the sine-Gordon equation defined on the whole real axis. *SIAM J. Sci. Comput.*, 29(6):2494–2506, 2007.

[213] H. Zhu, H. Shu, and M. Ding. Numerical solutions of two-dimensional Burgers' equations by discrete Adomian decomposition method. *Comput. Math. Appl.*, 60(3):840–848, 2010.