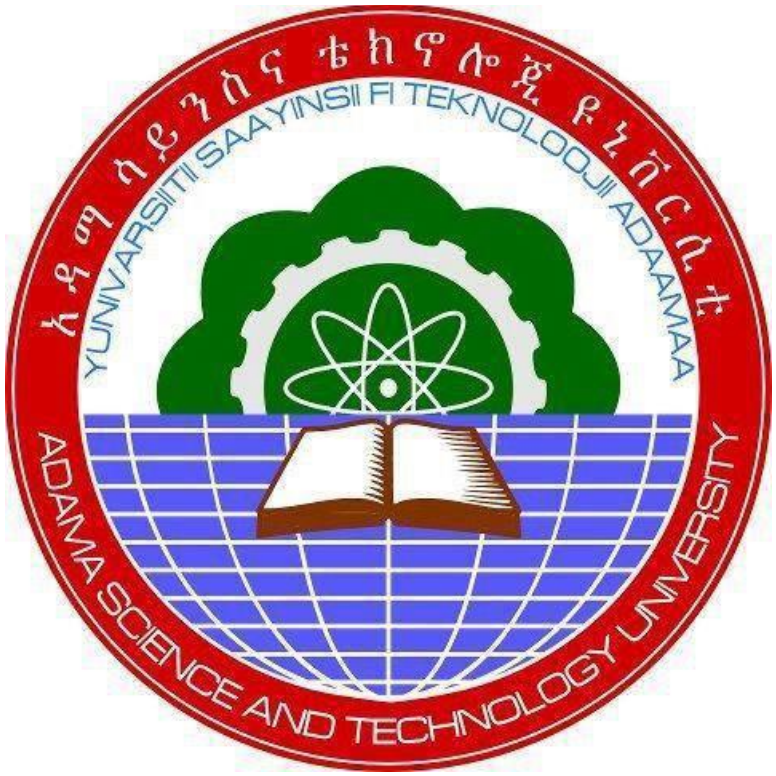
COMPARISON OF FINITE ELEMENT METHOD AND THE CUBIC SPLINE TO SOLVE BOUNDARY VALUE PROBLEMS



BY
SAMUEL GIRMA

A THESIS SUBMITTED TO ADAMA SCIENCE AND TECHNOLOGY UNIVERSITY
THE PROGRAM OF APPLIED MATHEMATICS
SCHOOL OF APPLIED NATURAL SCIENCE

IN PARTIAL FULFILLMENTS OF THE REQUIREMENTS FOR THE DEGREE OF MASTERS OF SCIENCE IN NUMERICAL ANALYSIS

Janwary, 2018
ADAMA, ETHIOPIA

Comparison of finite element method and the cubic spline to solve Boundary Value problems

By
Samuel Girma

Advisor
Dr.Tekle Gemechu (phD)

A Thesis Submitted to Adama Science and Technology University in Partial Fulfillment of the Requirement for the Degree of Master in Numerical analysis

Janwary, 2018
Adama, Ethiopia

Preface

I Samuel Girma affirm that the work on this thesis was carried out in the school of Applied Natural Science, Adama science and technology university, Adama during the period of September 2016 to August 2017. It was completed by the outer under the supervision of Dr. Tekle Gemechu.

The research contained in this thesis has not been submitted to any university nor has it been published previously. Where we used the work of others it has been duly acknowledged in the text.

SAMUEL GIRMA

Janwary, 2018

Declaration

I. Samuel Girma declare that

1. The research reported in this thesis, except where otherwise indicated, is my original research.
2. This thesis has not been submitted for any degree or examination at any other university.
3. This thesis does not contain other persons' data, pictures, graphs or other information, unless specifically acknowledge as being sourced from other persons.
4. This thesis does not contain other persons writing, unless specifically acknowledge as being sourced from other researchers. where other written sources have been quoted, then :
    a. Their words have been re-written but the general information attributed to them has been referenced.
    b. Where their exact words have been used, then their writing has been placed in italics and inside quotation marks, and referenced.
5. This thesis does not contain text, graphs or tables Copied and pasted from the internet, unless specifically acknowledged, and the source being detailed in the thesis and in the reference section.

Signed: _____

Date: _____

Abstract

Many physical problems are modeled by Boundary value problems and the solution sometimes may not be easily obtained analytically. In that case, we use different numerical methods. Among these numerical methods, in this paper, we present the Finite difference method, the Galerkins finite element method, and the cubic spline method for solving boundary value problems. They have great importance in Science and Engineering. Finite difference method is based on domain discretization. We first discretize the domain in to equal sub-intervals and then substitute the differentials with finite difference schemes. The tri-diagonal matrix system formed is solved using Thomas algorithm or any other Numerical Method. The Galerkin's finite element method for solving Boundary value differential equation is also examined. The Galerkin's finite element method needs base functions using Lagrange polynomials and it needs to define residue with weak formulation in the boundary conditions. The tri-diagonal matrix formed is carried out using Thomas algorithm or LU decomposition. We also choose cubic spline functions to develop the numerical solution of boundary value problems and use linear combination of cubic spline base functions to approximate the solution. The resulting linear systems of equations is solved using a tri-diagonal solver. To obtain a good approximation, the choice of the base function is important and to improve the approximation the number of base function is increased,. Convergence of the method shown through standard convergence analysis, finally, we compare the Galerkin's finite element method and cubic spline method with finite difference method, and then the advantage of the former is illustrated. For this, we select a second order boundary value problem and then discritize the domain of interest in to different nodal points and apply the three methods to determine the solutions. As the degree of the Lagrange polynomial increases, the maximum absolute error tends to zero, which means the Galerkins finite element method is more convergent, stable and consistent. The absolute errors for test examples are estimated, the comparison of approximate values, exact values and absolute error at the nodal points are shown using tables and graph, Further shown that the Galerkin's finite element method and the cubic spline methods give more accurate solution comparing with finite difference method.

Committee Signature Page

(APPROVAL SHEET)

Name	Signature	Date
SAMUEL GIRMA (Student)	_____	_____
Dr. TEKLE GEMECHU (Advisor)	_____	_____
_____ (External Examiner)	_____	_____
_____ (Internal Examiner)	_____	_____
_____ (Chair person)	_____	_____
_____ (HoD)	_____	_____
_____ (School Dean)	_____	_____
_____ (SGS Dean)	_____	_____

ACKNOWLEDGMENTS

I feel indebted to my supervisor, Dr Tekle Gemechu for all his reliable guidance and commitments throughout the course of my study. His approach to problem solving will definitely help me in the future. My thanks also go to my parents, for their expression of love. My irrevocable gratitude goes to my beloved wife Misrak Geremew for her love and patience throughout my study. More so, I will not forget all my lovely friends and teachers of Graduate class. My at most gratitude goes to my LORD for enabling me accomplish this dissertation.

TABLE OF CONTENTS

Pages

List of Tables page

List of Figures page

Notation

L	Linear differential operator
R	Residual
$\emptyset_i$	Basis functions
$w(x)$	Weight function
x	Domain of interest
$c(x)$	Continuous function over the domain x
y	Exact solution to differential equation
y_g	Approximate solution to Galerkins weighted residual method
e_a	Error from the approximate solution y_a
y_c	Approximate solution to cubic spline method
x_k	Interpolation nodes
l_k	Lagrange polynomial function
$\varphi(x)$	Nodal polynomials
N_e	Number of finite elements
h_i	Length of the i_{th} element

ACRONYMS

FDM: Finite Difference Method

FEM: Finite Element Method

BVP: Boundary value problem

PDE: Partial Differential Equation

ODE: Ordinary Differential Equation

CFDM: Central Finite Difference Method

BWFDM: Backward Finite Difference Method

CHAPTER 1

INTRODUCTION AND BACKGROUND

1. Introduction

In this thesis, we present numerical techniques called the Finite difference method; Galerkin's finite element method and the cubic spline method for solving second order Boundary value differential equations. The first chapter consists of the background, statement of the problem, the objectives of the study, significance of the Study, delimitation and definition of key terms. The second chapter deals with review of related literature. The third chapter focuses on methodology parts: study area and period, source of information, study design, and study procedure. The fourth chapter deals with description of the method, result and discussion. Lastly, the fifth chapter deals with conclusion and scope of future work. Each chapter gives the details of the numerical procedures used.

1.1 BACKGROUND OF THE STUDY

Differential equations used for mathematical modeling in science and engineering. However, it is apparent that there may be no known analytic solution to some differential equations and even when it does exist, it might be very difficult to obtain. In such cases, the numerical approximations are used. The theory of second-order differential equation is vast, and we focus on second-order boundary value differential equations, which have many important uses of special interest and numerical techniques that reduce excessive demand for computational time and at the same time giving a close approximation to the exact solution.

The history of differential equations traces the development of differential equations from calculus, independently invented by English physicist Isaac Newton (1665) and German mathematician Gottfried Leibnitz (1674). The term differential equation was coined by Leibnitz in 1676 for a relationship between the two differentials dx and dy for the two variables x and y. Newton solved his first differential equation in 1676 by the use of infinite series, eleven years after his discovery of calculus in 1665. Leibnitz solved his first differential equation in 1693; the year in which Newton first published his results, Hence in 1693 marks the inception for the differential equations as a distinct field in mathematics Korzybski Alfred, (1994). The different phases of 17^{th}, 18^{th}, and 19^{th} centuries played some crucial role in the history of differential equations. In the year 1695, Bernoulli proposed the problem of finding the general solution of Bernoulli's equation and Leibnitz and Jhonann Bernoulli solved it by different methods. In further development, 1724 was important to the early history of ordinary differential equations.

Ordinary differential equation acquired its significance when it was introduced in 1724; by Jacopo Francesco; count Ricatti of venice in his work in acoustics. Further, in the year of 1739 Leonhard Euler solves the general homogeneous linear ordinary differential equation with constant coefficients. L'Hospital came up with separation of variables in 1750, and it is now the physicist's handiest tool for solving partial differential equations since its introduction in 1828, Green's function have become a fundamental mathematical technique for solving boundary value problems by Poincare, H. (1890). In 20^{th} century, many quality works had done in the field of differential equations, but the major concern was the analytic and computational solutions of differential equations.

In last few decades, numerical analysis of differential equations has become the major topic of the study. The study of differential equations applied in both pure and applied mathematics. Pure mathematicians study the types and properties of differential equations, such as whether or not solutions exist, and whether they are unique. Applied mathematicians emphasize differential equations from applications, and in addition to existence/uniqueness questions. They are concerned with rigorously justifying methods for approximating solutions. Physicists and engineers are usually more interested in computing approximate solutions to differential equations. These solutions are then used to simulate celestial motions, simulate neurons, Design Bridge, automobiles, aircraft, etc. often, these equations do not have closed form solutions and are solved using numerical methods. Richard G. Rice, and Duong D. Do, (1995). Mathematicians also study weak solutions (relying on weak derivatives), which are types of solutions that do not have to be differentiable everywhere. This extension is often necessary for solutions to exist, and it results in more physically reasonable properties of solutions, such as shocks in hyperbolic (or wave) equations. Modern numerical analysis can credibly said to begin with the John von Neumann and Herman Goldstein, (1947). Although numerical analysis has a longer and richer history, modern numerical analysis, as used here is characterized by the synergy of the programmable electronic computer, mathematical analysis, and the opportunity and need to solve large and complex problems in applications. The need for advances in applications, such as ballistics prediction, neutron transport, and no steady, multidimensional fluid dynamics drove the development of the computer and depended strongly on advances in numerical analysis and mathematical modeling. Modern numerical analysis and scientific computing developed quickly and on many fronts.

Our current focus is on numerical algebra, numerical methods for differential equations, methods of approximation of functions, and the impact of these developments on science and technology. Recently, Dekema and Schultz, (1969) presented higher order methods for differential equations with large first derivative terms. These methods rewrite higher derivatives in Taylor's expansions in terms of first and second derivative term by differentiating their problem.

Boundary value problems arise in a number of different applications, which include, for example, deflection of beams and boundary value problem for partial differential equations. The problem is of considerable practical importance, for example, to determine the efficiency of solvent utilization or filtrate recovery in the washing of filter cakes Brenner, (1962).

Two-point boundary value problems are encounter in almost every branch of engineering and physical sciences. They have been widely studied and many numerical techniques have developed for their solution. Several books deal with two-point boundary value problems and numerical techniques for their solution. A comprehensive coverage of these topics can be found in Bellman and Kalaba,(1965). The Finite Element Method (FEM) is one of the most successful and dominant numerical method in the last century. It is extensively use in modeling and simulation of engineering and science due to its versatility for complex geometries of solids and structures and its flexibility for many problems. The FEM regarded as relatively accurate and versatile numerical tool for solving differential equations that model physical phenomenon Shabani and Mazahery, (2011). In finite element method (FEM), functions represented in terms of Basis functions and the ODE solved in its integral (weak) form. In this method, the domain under consideration, partitioned in a finite set of elements. In this the differential equation is discretized by using approximation methods with piecewise polynomial solution S.Surana K., A.Rajwani, (2006). The finite element method originated from the need for solving complex elasticity and structural analysis problems in civil and aeronautical engineering. Its development can be trace back to the work by Alexander Hrennikoff, (1941) and Richard Courant, (1942) while the approaches used by these pioneers are dramatically different. They share one essential characteristic: mesh discritization of a continuous domain in to a set of discrete sub domain, usually called elements. Development of the finite element method began in earliest in the middle to late 1950s for airframe and structural analysis and gathered momentum at the University of Stuttgart through the work of John Argyris and at Berkeley through the work of Ray W. Clough in the 1960s for use in civil engineering.

By late 1950s, the key concepts of stiffness matrix and element assembly existed essentially in the form used today, NASA issued request for proposals for the development of the finite element software NASTRAN in 1965. The method was provided with a rigorous mathematical foundation in 1973 with the publication of Strang and Fix's An analysis of the finite element method has since been generalized in to a branch of applied mathematics for numerical modeling of physical systems in a wide variety of engineering disciplines, like electromagnetism and fluid dynamics. The Galerkin's Finite Element method is a well-known numerical technique for the numerical solution of differential equations. Boris G. Galerkin a Russian mathematician was born on march 4 [O.S February 20, 1871] He comes from poor family and this was to mean that he had a harder time through his years of education than would otherwise have been the case, He attend secondary school in Minsk, then in 1893 he entered the petersberg Technological institute. Here he studied mathematics and engineering but He needed to make money to survive so at first he look on private tutoring then from 1896 he worked as a designer. There are linear and nonlinear problems in science and engineering, second order differential equations with various types of boundary conditions solved by either analytically or numerically. Numerical simulation in engineering science and in applied mathematics has become a powerful tool to model the physical phenomena, particularly when analytical solutions are not available or when it is very difficult to obtain the solution. Finlayson B.A**.** (1972),

Consider a linear differential equation

$$y'' = A(x)y' + B(x)y + F(x), \qquad a \le x \le b \tag{3.1}$$

With boundary conditions $y(a) = \alpha$ and $y(b) = \beta$

For the residual given by $R(x) = y'' - A(x)y' - B(x)y - F(x)$ (3.2)

Finite element methods find a piecewise polynomial (PP) approximation $u(x)$, to the solution of (3.1). A piecewise polynomial is a function defined on a partition such that on the subintervals defined by the partition is a polynomial. The PP approximation can represent by

$$u(x) = \sum_{j=1}^{m} C_j \phi_j(x) \tag{3.3}$$

Where $\{\phi_j(x)/j = 1,2,\dots,m\}$ are specified functions that are piecewise continuously differentiable, and $\{C_j/j = 1,2,\dots,m\}$ are as yet unknown constants. The functions $\phi_j(x)$, are called basis functions, satisfying the boundary conditions and $\int_a^b R(x)\phi(x)dx = 0$. The finite element methods differ only in how the unknown coefficients are determined. P.E.LEWISE, J.P.WARD, (2002). The theory of spline functions is a very active field of approximation for the boundary value problems (BVPs) when numerical aspects are considered. One of the most popular ways of dealing with these issues is to use splines.

In their most general form, splines can considered as a mathematical model that associate a continuous representation of a curve or surface with a discrete set of points in a given space. Spline fitting is an extremely popular form of piecewise approximation using various forms of polynomials of degree n, on an interval in which they are fete to the function at specified points, known as control points, nodes or knots. The polynomial used can change, but the derivatives of the polynomials are required to match up to degree $n-1$ at each side of the knots, or to meet related interpolator conditions. Boundary conditions also imposed on the ends of the interval. H.N.Caglar,S.H.Caglar and E.H.Twizell, (1999)

The heart of spline construction revolves around how the selected control points are effectively "blended" together using the polynomial function of choice. Given the various alternative forms of spline, the question is which type of spline is most applicable in any given situation naturally arises and it is inevitably difficult one to answer without clear criteria. Arguably, the most important deciding question is whether the spline is required to approximate or interpolate the control points. In other words, does the user require the curve to pass through the control points with absolute precision, or is the overall shape of the curve more important. Beyond this critical requirement, Blanc and Schlick, (2013) offer an extremely useful checklist of desirable properties that any spline model should possess. The most common spline and piecewise interpolation used are linear, quadratic and cubic respectively. To obtain a smoother curve, cubic splines are frequently recommended, because they provide the simplest representation that exhibits the desired appearance of smoothness. They are generally well behaved and continuous up to the second order derivative at the data points. Even though cubic splines are less prone to oscillation or overshooting due to instability inherent in higher order polynomial than global polynomial equations, they do not prevent it. Thus, to avoid these oscillations, it is common to divide the interval into sub-interval and approximate the function using low degree polynomial on each sub-interval. In view of this, this thesis gives a small step towards the development of computational analysis of ordinary differential equations, which have lot of utilities in the field of science and engineering.

Many researchers have developed numerical technique to study the numerical solution of two Point boundary value problem, Dinkar Sharma et al. (2013) proposed a Comparative Study of Galerkin Finite Element and Spline Methods for two Point Boundary Value Problems. Dogan (2004) proposed the Galerkin finite element approach for the numerical solutions of Burgers' equation. Sengupta et al. (2005) carried out Gakerkin finite element methods for wave problems.

Kaneko et al.(2006) discussed the Discontinuous Galerkin-finite element method for parabolic problems. EI-Gebeily et al. (2009) studied the finite element- Galerkin method for differential equations. Sharma et al. (2011) proposed Galerkin finite Element Methods for numerical solution of advection diffusion equation. Onah (2002) proved the asymptotic convergence of the solution of a parabolic equation by using two methods namely, the Galerkin method expressed in terms of linear splines and the Finite Element method expressed by cubic spline basis functions.

In this thesis, we apply a direct method based on finite difference method, Galerkin's finite element method and the Cubic spline for the analysis of two-point boundary value problems of second-order differential equations, and we compare the numerical solutions of two-point boundary value problems. Calculate the absolute errors; check the convergence and stability of the problem for each method. Using graphs illustrate the deflection of the approximate solution with the exact solution.

1.2 Statement of the problem

In the literature of numerical analysis for solving boundary value problem (BVP) of differential equations, many authors have attempted to obtain higher accuracy rapidly by using a number of methods. Such that diffusion occurring in the presence of exothermic chemical reaction, heat conductions associated with radiation effect, A.S.Bakhvalov, (1969). Solving such type of boundary value problems analytically is possible only in a very rare cases and there is a gap in accuracy of numerical solution for different numerical methods. Although various studies were carried out Mehmet, (2009); Gentian, (2015); Lijalem, (2017), and Nuru, (2017); for the analysis of Galerkins Finite element method and the cubic spline. The present study was different from them for some reasons. Most studies, which had conducted on Galerkins Finite element method and the cubic spline method, were concerned with stability Consistency and convergence of the methods independently but in each of them there is no clear comparison on the numerical solutions of boundary value problems. This study, intended to focus analysis of cubic spline method, the Galerkin finite element method and finite difference method for solving Boundary value differential equations. Thus, compare the numerical solutions of Boundary value problems.

1.3 Objective of the Study

The study had the following general and specific objectives.

1.3.1. General Objective

The general objective of this thesis was to make comparison among the finite difference method, Galerkin's Finite Element method and the Cubic Spline methods for solving Boundary value problems.

1.3.2. Specific Objective

The specific objectives of this thesis were:

1. Calculate the convergence of finite element method, and the cubic spline method.
2. To compare solutions of finite element method and cubic spline method with exact value.
3. Compare the absolute maximum errors.
4. To study consistency and stability

1.4 Research Questions

Based on the objectives stated above, the following research questions were designed to guide the study. The research questions were:

- Which method is better convergent?
- Which method is more stable?
- Compare the numerical solutions of Boundary value problems (BVP) by using cubic spline, finite element and finite difference methods?

1.5 Significance of the study

This study may help to develop skills and Knowledge in finite difference method, Finite element method and the cubic spline method for solving boundary value problems, further may provide useful result and information for the advancement of Numerical algorithm, and also it provide a reference material for students, teachers and anyone who works on this area.

1.5 Delimitation of the study

As it was discussed earlier in this chapter, the aim of the study was to compare numerical methods. We were described and discuss a method based on the application of Finite difference method, Galerkin's Finite Element and Cubic splines. Firstly, to keep the study, it was delimited to second order differential boundary value problems. The reason the researcher had selected this was that there are many applications leading to second order Boundary Value problems, which is the most important in applied mathematics and physics.

1.6 Limitation of the study

The researcher was met with luck of internet accesses to study around his work area.

1.7 Organization of the thesis

This thesis organized into five chapters. Chapter one discusses the background, the statement of problem, objectives and research questions. It also discusses the significance of the study, its delimitation, and limitations of the study, organization of the study and definition of terms. The second chapter deals with review of related literature. The third chapter focuses on methodology parts: study area and period, source of information, study design, and study procedure. The fourth chapter deals with description of the method, result and discussion. Lastly, the fifth chapter deals with conclusion and scope of future work.

1.8 Definition of key terms

Boundary value problem (BVP): A problem, typically an ODE or a PDE, which has values assigned on the physical boundary of the domain in which the problem is specified, is called a boundary value problem (BVP).

I. An Ordinary differential equation (ODE) is a differential equation in which the unknown function is a function of a single independent variable.

II. A partial differential equation (PDE) is a differential equation in which the unknown function is a function of multiple independent variables and their partial derivatives.

CHAPTER TWO

REVIEW OF RELATED LITERATURES

2. Introduction

This section concerned with presentation of ideas that forwarded by different scholars of Numerical analysis about the historical background of finite difference method, Galerkins finite element method, and the cubic spline methods to solve Boundary value problems along with its definition and mathematical formulation.

2.1 Historical Background of Finite Difference Methods (FDM)

Finite difference method (FDM) used to solve ordinary differential equations that have conditions imposed on the boundary rather than at the initial points. The derivatives appearing in the differential equation and the boundary conditions replaced by their finite difference approximations and the resulting linear systems of equations solved by any standard procedure. The accuracy of finite difference operators can be improved by using information from more grid points. The weights for the grid points can be obtained by using Taylor's series. R.J. LeVeque, (1998). In finite difference method (FDM), functions represented by their values at certain grid, points and derivatives are approximate through differences in these values. For the finite difference method, the domain under consideration is represented by a finite subset of points. These points are called nodal points or the grids. This grid is almost always arranged in (uniform or non uniform) rectangular manner. The differential equation is replaced by a set of difference equations, which can be solved by direct or iterative method. These roots are the values of the required solution at the pivotal points R. E. Mickens (2002), R.J. LeVeque, (1998).

The finite difference method can be difficult to analyze, in part because it is quite general in its applicability. Much of the existing stability and convergence analysis is restricted to special cases, particularly to linear differential equations with constant coefficients. These results are then used to predict the behavior of difference methods for more complicated equations.

2.2 The Galerkin's Finite element method

The Galerkin's Finite element formulation of the Galerkin method with Galerkin's (or "weak ") differential equations problem statement form are known all over the world. Now a day they provide a foundation for algorithms in the field of mechanics, thermodynamics, electromagnetism, hydrodynamics and money others.

Galerkin's method belongs to a wider class of methods called the weighted residual methods. In this method, an approximating function called the trial function (which satisfies all the boundary conditions) is substituted in the given differential equation and the result is called the residual (the result will not be zero since we have substituted an approximating function). S. Iqbala et. al. (2010). the residual is then weighted and the integral of the product, taken over the domain, is then set to zero. If the Euler - Lagrange equation corresponding to a functional coincides with the differential equation of the problem, then both the Rayleigh – Ritz and Galerkin methods yield the same system of equations. The method of weighted residuals used to find approximate solutions to differential equations before the finite element method come in to existence, Finlyson B. A. (1980). The method was introduced by Crandal S.H, (1956). Other scientists like Finlayson and Scriven (1966), Vichnevetsky(1969) and Finlayson (1972) have also used this method. The method used to solve boundary value problems arising from fluid flow, structural mechanics and heat transfer. It is well known because of the interactive nature of the first step. The user provides an initial guess for the solution which is then forced to satisfy the governing differential equations and its boundary conditions. The guessed solution is chosen to satisfy the boundary conditions; however, it does not necessarily satisfy the differential equations Grandin H. (1991) and Richard G.Rice (1995). The method of Galerkin's weighted residual requires two types of functions namely the trial functions and the weighted functions. The former is used to construct the trial solution while the latter is used as a criterion to minimize the residual Finalyson B.A, (1980). Various methods are available in the literature concerning their numerical solutions Dennis G.zill, (2013). Khan in G.D. Smith, (1993) obtained a parametric cubic spline solution of boundary value problems, in H.N Caglar, et.al. (1999). Feng (2003) Solved second-order Neumann boundary value problem with singular nonlinearity for exact three positive solutions, Lima and Carpentier, (1998) Obtained a Numerical solution of a singular boundary-value problem in non-Newtonian fluid mechanics, Rashidinia and Jalilian introduced a spline solution of two point boundary value problems, Viswanadham et al.(1998) obtained a numerical solution of a fourth order boundary value problems by Galerkin method with splines basis, Das .et. al,(1999) Produced a method for solutions of nonlinear second order multi-point boundary value problems and recently Bhatti and Bracken, (1968) solved linear differential equations numerically by Galerkin method. However, in this paper a very simple and efficient Galerkin numerical method is proposed with lagrange polynomials as trial basis functions. The formulation is derived to solve second order boundary value problem in detail.

2.3 Cubic spline method

In the mathematical field of numerical analysis, interpolation is a method of constructing new data point within the range of a discrete set of known data points. In a more informal language, interpolation means a guess at what happens between two values already known. In Engineering and Environmental sciences application, data collected from the field are usually discrete, therefore a more analytically controlled function that fits the field data is desirable and the process of estimating the outcomes in between these desirable data points is achieved through interpolation Crochiere and Rabiner, (1983).

There are two main uses of interpolation. The first use is in reconstructing the function (x) when it is not given explicitly and/or only the values of (x) and certain order derivatives at a set of points, called nodes are known. Secondly interpolation is used to replace the function $f(x)$ by an interpolating polynomial $p(x)$ so that many common operations like the determination of roots, differentiation, integration etc. which are intended for the function $f(x)$ may be performed using the interpolant $p(x)$ Jain *et al*. (2007)

Spline interpolation is a form of interpolation where the interpolant is a special type of piecewise polynomial called spline. Spline interpolation is often preferred over polynomial interpolation because the interpolation error can be made small even when using low degree polynomials for the spline Hazewinkel, (2001). Spline interpolation avoids the problem of Runge's phenomenon, in which oscillation occurs between points when interpolating using high degree polynomials (Kim, 2005; Turner, 1989). This method gives an interpolating polynomial that is smoother and has smaller error than other interpolating polynomials such as Lagrange polynomial and Newton polynomial (Jain *et al*.; 2007).

Current literatures exist showing the application of Spline and cubic interpolation. The optimal point for a given group of oil wells was found by Jamal (2001) using the application of Lagrange multiplier, an approach that depend on the functional relationship between oil production and gas injection rates. Jamal (2001) used two type of function (quadratic and rational functions) to fit the gas injection against oil production rate. Ntherful (2013)

Worked on optimal Spline based Gas-left Allocation using Lagrange's multiplier in an oil production field. After fitting the gas injection and oil output rates of the wells with cubic spline functions, the optimal rates of gas injection and oil output in each of the wells are determined using the Lagrange multiplier method. It was found that the total optimum oil production rate for data fitting with spline based function was higher than the total optimum oil production rate for

data fitting with rational function. The optimal value of the spline based function was found to be twice that of rational function. Nelson *et al.* (1997), worked on parametric studies on thermally stratified chilled water storage system. In their work, they analyze the stratification decay in thermally stratified vertical cylindrical cool storage system using a one dimensional conjugate heat conduction model. They found out that the thermo clines degrade due to the heat transfer from the ambient, thermal diffusion in the storage tank, axial wall conduction and mixing due to admission of the fluid in the storage tank during charging and discharging. Another application of heat transfer was found in the construction of a natural cubic spline of the heat capacity of gadolinium using experimental measurement at fixed values of magnetic induction and the least square curve fitting was used to obtain the approximation function of the heat capacity of gadolinium.

The concept of interpolation is the selection of a function $p(x)$ from a given class of functions in such a way that the graph of $y = p(x)$ passes through a finite set of given data points. In this thesis, we limit the interpolating function p (x) to being a Cubic polynomial interpolation. Its primary use is to furnish some mathematical tools used in developing methods in the areas of approximation theory, and the numerical solution of Boundary value differential equations. Interpolation is used to estimate the value of a function between known data points without knowing the actual function. Piecewise interpolation method uses a polynomial of low degree between each pair of known data points. If a first degree polynomial is used, it is called linear interpolation, for second and third degree polynomial; it is called quadratic and cubic spline respectively. The higher the degree of the spline, the smoother the curve spline of degree m, will have continuous derivative up to $(m - 1)$ at the data points.

As noted in the introduction, interpolation theory is a foundation for the development of methods in numerical integration and differentiation, approximation theory, and the numerical solution of differential equations. Additional results on interpolation theory are given in de Boor (1978), Davis (1963), Henrici (1982, chaps. 5 and 7), and Hildebrand (1956). For an historical account of many of the topics of this Chapter, see Goldstine (1977). The introduction of digital computers produced a revolution in numerical analysis, including interpolation theory. Before the use of digital computers, hand computation was necessary, which meant that numerical methods were used that minimized the need for computation. Such methods were often more complicated than the methods now used on computers, taking special advantage of the unique mathematical characteristics of each problem.

These methods also made extensive Use of tables, to avoid repeating calculations done by others; interpolation formulas based on finite differences was used extensively. A large subject was created, called the *finite difference calculus,* and it was used in solving problems in several areas of numerical analysis and applied mathematics. For a general introduction to this approach to numerical analysis, see Hildebrand (1956) and the references contained. The use of digital computers has changed the needs of other areas for interpolation theory, vastly reducing the need for finite difference based interpolation formulas. But there is still an important place for both hand computation and the use of mathematical tables, especially for the more complicated functions of mathematical physics. The National Bureau of Standards tables of Abramowitz and Stegun (1964) are an excellent reference for non elementary functions. The availability of sophisticated hand calculators and microcomputers makes possible a new level of hand (or personal) calculation. Piecewise polynomial approximation theory has been very popular since the early 1960s, and it is finding use in a variety of fields. Most of the interest in piecewise polynomial functions has centered on spline functions. The beginning of the theory of spline functions is generally credited to Schoenberg in his 1946 papers, and he has been prominent in helping to develop the subject [e.g., see Schoenberg (1973)]. There is now an extensive literature on spline functions, involving many individuals and groups. For general surveys, see Ahlberg et al. (1967), de Boor (1978), and Schumaker (1981). Some of the most widely used computer software for using spline functions is based on the programs in de Boor (1978). Interpolating a set of data points can be done using polynomial or spline functions. However polynomial interpolation is commonly used and many numerical methods are based on polynomial approximations. For a given set of $(N + 1)$ data points (x_0 , y_0), (x_1, y_1), … , (x_N , y_N), it is of interest to find N^{th} order polynomial function that can match these data points. The N^{th} polynomial function is given as

$$P_N = a_o + a_1x + a_2x^2 + \dots + a_Nx^N$$

The coefficients can be obtained by solving a set of algebraic equations

$$\begin{cases} a_0 + a_1x_0 + a_2x_0^2 + a_3x_0^3 + \quad . \quad . \quad . \quad + a_Nx_0^N = y_0 \\ a_0 + a_1x_1 + a_2x_1^2 + a_3x_1^3 + \quad . \quad . \quad . \quad + a_Nx_1^N = y_1 \\ \vdots \\ a_0 + a_1x_N + a_2x_N^2 + a_3x_N^3 + \quad . \quad . \quad . \quad + a_Nx_N^N = y_N \end{cases}$$

As the number of data points increases, so also that of the unknown variables and equations, the resulting system of equations may not be so easy to solve. However, there are a number of alternative forms of expressing an interpolating polynomial beyond the familiar format stated above. Among them are Lagrange, Newton's forward and backward difference, and Hermite interpolations. According to Singiresu (2002), the errors of a single polynomial tend to increase drastically as its order n becomes large. The higher order polynomial often introduces unnecessary oscillation and wiggles and due to this, polynomial interpolation will not be always accurate. To avoid this, information more data points will be used and at the same time keeping the function true to the data behavior is the objective of spline interpolation.

CHAPTER THREE

RESEARCH DESIGN AND METHODOLOGY

3. Introduction

This chapter describes the research methodology employed in order to achieve the objective of the study. First, it presents the research design of the study. Then, procedures used for the study. Finally, it presents the methods employed for final conclusion.

3.1 Study design

The main objective of this study was to compare the finite difference method, the Galerkins finite element method and the Cubic spline method for solving Boundary value problems. In line with this aim of the study, our design was documentary and experimental. Solving the discritized ordinary differential equation using weighted function for Galerkin's method, tri-diagonal matrix system for FDM, and a system of linear equations and obtain a function value at each node for cubic spline method. The study involves numerical implementation with MATLAB.

3.2 Source of information

We use previous study documents and uses different reference books, and documents from internet on Analysis of finite difference method, Galerkin's Finite element method and the Cubic spline method.

3.3 Instrumentation

The thesis is introduces the numerical solution of Boundary Value Ordinary differential equation using finite difference method, Galerkin's Finite element method and the Cubic spline method . To compare the numerical solutions of each method use table of comparison for the exact and approximate solution and use MATLB to graph the problem for each method.

3.4 Study procedures

In order to achieve the stated objectives, the study will follow the following procedures

- Define the problem
- Describe the methods
- Discritize the given domain (or interval)
- Solving a system of Algebraic equations formed for FEM, FDM and cubic spline applications.
- Writing Code for the problem by using MATLAB language
- Presenting results by tables and graphs

CHAPTER FOUR
RESULTS AND DISCUSSION

4. Introduction

This chapter deals with the analysis of the finite difference method, the Galerkins finite element method and the Cubic spline methods to solve Boundary value problems. The detail of Discussion is given below.

4.1 Finite difference method (FDM)

A finite difference method typically involves the following steps

Consider a second order differential equation given by

$y'' = A(x)y' + B(x)y + F(x), \quad a \le x \le b$

$$y(a) = \alpha \ and \quad y(b) = \beta \tag{4.1}$$

Grid: Suppose $a = x_0 < x_1 < x_2 < \cdots < x_{n-1} < x_n = b$ represent a regular partition of the internal $[a, b]$ as shown in fig (1.1), below and substitute the derivatives in the ordinary differential equation with finite difference schemes.

That is $x_i = a + ih\,, where\ i = 0, 1, 2, 3\ .\ .\ .\ n\ and\ h = \frac{b-a}{n}$

The points $take\ x_1 = a + h\,,\ x_2 = a + 2h\,,\quad x_3 = a + 3h\quad .\ .\ .\quad x_{n-1} = a + (n-1)h$

This points are called interior mesh pints of the interval $[a, b]$.

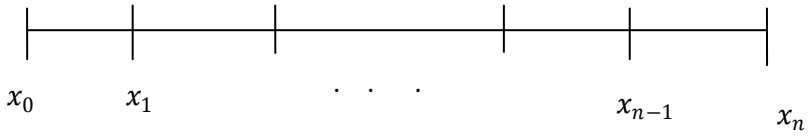


Fig4.1 (1-D discritzation)

If $y(x)$ and its derivatives are single - Valued continuous functions of x then by Taylor's expansion, we have:

$$\mathrm{y(x + h)} = y(x) + hy'(x) + \frac{h^2}{2!}y''(x) + \frac{h^3}{3!}y'''(x) + \ .\ .\ . \tag{4.2}$$

$$y(x - h) = y(x) - hy'(x) + \frac{h^2}{2!}y''(x) - \frac{h^3}{3!}y'''(x) + \quad .\quad .\quad . \tag{4.3}$$

Equation (4.2).gives

$y'(x) = \frac{1}{h}[y(x + h) - y(x)] - \frac{h}{2}y''(x) + \ldots$

$$y'(x) = \frac{1}{h}[y(x + h) - y(x)] + O(h) \tag{4.4}$$

This is the forward difference approximation for $y(x)$ with an error of the order h.

Similarly (4.3) gives

$$y'(x) = \frac{1}{h}[y(x) - y(x-h)] + O(h) \tag{4.5}$$

This is the backward difference approximation for $y'(x)$ with error of the order h.

Adding (4.4) and (4.5), we obtain

$$y'(x) = \frac{y(x+h)+y(x-h)}{2h} + O(h^2) \tag{4.6}$$

This is the central difference approximation for $y'(x)$ with an error of the order h^2.

It is better than the forward and backward difference approximations,

Hence Adding (4.2) and (4.3) we get:

$$y''(x) = \frac{1}{h^2}[y(x+h) - 2y(x) + y(x-h)] + O(h^2) \tag{4.7}$$

This is the central difference approximation for $y''(x)$.

Similarly, we can derive central difference approximations for higher derivatives. Hence the working expressions for the central difference approximations to the first four derivatives of y_i are by negating $0(h)$ or $0(h^2)$ (truncation errors) when h is small.

$$y'_i = \frac{y_{i+1}-y_{i-1}}{2h} \tag{4.7a}$$

$$y''_i = \frac{y_{i+1}-2y_i+y_{i-1}}{h^2} \tag{4.7b}$$

$$y'''_i = \frac{y_{i+2}-2y_{i+1}2y_{i+1}+y_{i-2}}{2h^3} \tag{4.7c}$$

$$y^{iv}{}_i = \frac{y_{i+2}-4y_{i+1} + 6y_i + 4y_{i-1} + y_{i-2}}{h^4} \tag{4.7d}$$

The accuracy of this method depends on the size of the sub – interval h and on the order of approximation. As we reduce h, the accuracy improves but the number of equations solved also increases.

We consider second order boundary value differential problem (4.1)

If y'' and y' are replaced by the central deference approximations (4.7a) and (4.7b), we get

$$\frac{y_{i+1}-2y_i+y_{i-1}}{h^2} = A_i\left(\frac{y_{i+1}-y_{i-1}}{2h}\right) + B_i y_i + F_i \tag{4.8}$$

After simplifying:

$$\left(1+\frac{h}{2}A_i\right)y_{i-1} - (2+h^2B_i)y_i + \left(1-\frac{h}{2}A_i\right)y_{i+1} = h^2F_i \tag{4.9}$$

This is called a finite difference equation approximation to the differential equation. Letting, $i = 1, 2, 3 \ . \ . \ . \ , n-1$. The ordinary differential equation then becomes a linear system of algebraic equations.

That is we obtain $n-1$ equations in the $n-1$ unknowns $y_1,\ y_2,\ y_3\ .\ .\ .\ y_{n-1}$, bear in mind that we know $y_0\ and\ y_n$ since these are the prescribed boundary conditions.

$$y_0 = y(x_0) = y(a) = \propto\ and\ \ y_n = y(x_n) = y(b) = \beta$$

Since $A(x),\ B(x)\ and\ F(x)$ are functions of x.

We also obtain the tri-diagonal system

$$\begin{bmatrix} 2+h^2B(x_1) & -1+\frac{h}{2}A(x_1) & 0 & & \\ -1-\frac{h}{2}A(x_2) & 2+h^2B(x_2) & -1+\frac{h}{2}A(x_2) & & \\ & \ddots & \ddots & \ddots & \\ & & & -1+\frac{h}{2}A(x_{n-1}) & \\ & 0 & -1-\frac{h}{2}A(x_n) & 2+h^2B(x_n) & \end{bmatrix} \begin{bmatrix} y_1 \\ y_2 \\ \\ \vdots \\ \\ y_{n-1} \\ y_n \end{bmatrix} = \begin{bmatrix} -h^2F(x_1)+(1+\frac{h}{2}A(x_1))\alpha \\ -h^2F(x_2) \\ \\ \vdots \\ \\ -h^2F(x_{n-1}) \\ -h^2F(x_n)+(1-\frac{h}{2}A(x_n))\beta \end{bmatrix}$$

This tri-diagonal system can be solved using Algebraic methods such as Thomas algorithm, LU decomposition or any other methods.

4.1.1 Error Analysis for Finite Difference Method

Consider the second order ordinary differential equation;

$$y''(x) = f(x)\ \ for,\ a \le x \le b \tag{4.10}$$

With some given boundary conditions

$$y(a) = \alpha,\quad and\quad y(b) = \beta \tag{4.11}$$

We were attempted to compute a grid function consisting of values $Y_0, Y_1, \dots, Y_n, Y_{n+1}$ where Y_j is our approximation to the solution $y(x_j)$. Here $x_j = a + jh$ and $h = \Delta x = \frac{b-a}{n+1}$ is the mesh width, the distance between grid points. From the boundary conditions we know that $Y_0 = \alpha$ and $Y_{n+1} = \beta$ Therefore, we have n unknown values $Y_1, Y_2, \dots., Y_n$ to compute. If we replace $y''(x)$ in equation (4.10) by the centered difference approximation (4.7b)

$$y_j'' = \frac{1}{h^2}\left(Y_{j+1} - 2Y_j + Y_{j-1}\right) \tag{4.12}$$

Then we get a set of algebraic equations

$$\frac{1}{h^2}\left(Y_{j+1} - 2Y_j + Y_{j-1}\right) = f(x_j),\ \ for\ \ j = 1,2,3, \dots n \tag{4.13}$$

Note that the first equation $(j = 1)$ involves the value $Y_0 = \alpha$ and the last equation $(j = n)$ involves the value $Y_{n+1} = \beta$. we have a linear system of n equations for the n unkowns. This can be written in the form

$$AY = f$$

Where Y is the vector of unknowns $Y = [Y_1, Y_2, \cdots, Y_n]^T$ and

$$A = \frac{1}{h^2}\begin{bmatrix} -2 & 1 & 0 & 0 & 0 & & \cdots \\ 1 & -2 & 1 & 0 & 0 & & \cdots \\ 0 & 1 & -2 & 1 & 0 & & \cdots \\ 0 & 0 & & & & & \\ 0 & 0 & & \ddots & \ddots & & \\ 0 & 0 & \cdots & & 1 & -2 & 1 \\ 0 & 0 & 0 & & \cdots & 1 & -2 \end{bmatrix} \qquad f = \begin{pmatrix} f(x_1) - \frac{\alpha}{h^2} \\ f(x_2) \\ f(x_3) \\ \\ \vdots \\ \\ f(x_{n-1}) \\ f(x_n) - \frac{\beta}{h^2} \end{pmatrix} \qquad (4.14)$$

This tri-diagonal linear system is non singular and can be easily solved for Y from any right hand side f. We know that the centered difference approximation $y''(x)$, when applied to a known smooth function $y(x)$, gives a second order accurate approximation to $y''(x)$. the error in the discrete values $y_1, y_2, \ldots, y_n$, in relation to the true solution $y(x)$, which is a function .since y_j is supposed to approximate $y(x_j)$ it is natural to use the point wise errors Y_j of $y(x_j)$.

If we let $\tilde{Y}$, be the vector of true values.

$$\tilde{Y} = \begin{cases} y(x_1) \\ y(x_2) \\ y(x_3) \\ \vdots \\ y(x_n) \end{cases}$$

Then the error vector E defined by $E = Y - \tilde{Y}$, contains the errors at each grid points. To obtain a bound on the magnitude of this vector showing that it is $O(h^2)$ as $h \to 0$. To measure the magnitude of this vector we must use some norm, for example, the max-norm.

$\|E\|_\infty = max_{1 \le j \le n} |E_j| = max_{1 \le j \le n} |Y_j - y(x_j)|$

We first compute the local truncation error (LTE) of the method and then use some form of stability to show that the global can be bounded in terms of the LTE. The global error simply refers to the error $Y - \tilde{Y}$ that we are attempting to bound. The LTE refers to the error in our finite difference approximation of derivatives and hence is something that can be easily estimated using Taylor series expansions.

4.1.2 Local Truncation error for finite difference method

The local truncation error is defined by replacing Y_j with the true solution $y(x_j)$ in the finite difference formula (4.13). In general the true solution $y(x_j)$ won't satisfy this equation exactly and the discrepancy is the LTE, which we denote by η_j.

$$\eta_j = \frac{1}{h^2}\left(y(x_{j+1}) - 2y(x_j) + y(x_{j-1})\right) - f(x_j) \tag{4.15}$$

For $j = 1, 2, \ldots, n.$ of course in practice we don't know what the true solution $y(x)$ is, but if we assume it is smooth, then by the Taylor series expansions we know that

$$\eta_j = \left[y''(x_j) + \frac{1}{12}h^2 y''''(x_j) + O(h^4)\right] = f(x_j) \tag{4.16}$$

Using our original differential equation (4.10) this becomes

$$\eta_j = \frac{1}{12}h^2 y''''(x_j) + O(h^4) \tag{4.17}$$

If we define η to be the vector with components η_j, then

$$\eta = A\tilde{Y} - F \tag{4.18}$$

Where $\tilde{Y}$ is the vector of true solution values equation (4.14), and so

$$A\tilde{Y} = F + \eta \tag{4.19}$$

4.2 Galerkin's Finite Element Method

The Galerkin's finite element method described here is one of the most important numerical method. In this method the domain is sub divided in to a number of smaller regions called elements and over each of these elements the continuous function is approximated by a suitable piecewise polynomial. To obtain a better approximation one need not use higher order polynomials but only use a finer sub division, $i.e$ increase the number of elements.

To explain the Galerkin's finite element method, we consider the boundary value problem defined by (4.1). It is a linear two point boundary value problem. To find an approximate solution of the problem, first we need to discritize the domain in to sub intervals of size $h = x_{i+1} - x_i$ and then choose base functions $\phi_i(x)$ using Lagrange polynomials, Approximate solution $u(x)$ assumed to be a linear combination of the $\phi_i(x)$

We write the $\boldsymbol{i^{th}}$ term of the Lagrange polynomial as

$$\phi_i(x) = \frac{(x - x_0)(x - x_1)\ldots(x - x_{i-1})(x - x_{i+1})\ldots(x - x_n)}{(x_i - x_0)(x_i - x_1)\ldots(x_i - x_{i-1})(x_i - x_{i+1})\ldots(x_i - x_n)}$$

$$i.e \quad u(x) = \phi_0(x) + \sum_{i=1}^{n} C_i\phi_i(x) \tag{4.20}$$

Where the function $\phi_0(x)$ is chosen to satisfy the given boundary conditions of the problem

$$\phi_0(x) = y(a) + (y(b) - y(a))\frac{(x - a)}{(b - a)}$$

Functions $\phi_1(x),\ \phi_2(x),\ \cdots, \phi_n(x)$ must each satisfy the corresponding homogeneous form of the boundary conditions. Now, $u(x)$ will not satisfy equation(4.1), but produces a residual or discrepancy. This is equal to the difference between the left hand side and the right hand side of equation (4.1).

If $R(x)$ is the residual, $$R(x) = y'' + A(x)y' + B(x)y - F(x) \tag{4.21}$$

There are n equations to solve the residual

$$\int_a^b R(x)\phi_i(x)dx = 0, \qquad \text{For} \quad i = 1, 2, 3, \ . \ . \ . \ \ n\,, \tag{4.22}$$

Taking the weight function as, $\phi_i(x)$ we write

$$\int_a^b \psi_i(x)R(u)dx = 0\,.$$

And it is called the weak form of (4.1).

This yields a system of equations for the unknown parameters C_i and can be solved using Galerkin's method, we usually take $\psi_i(x) = \phi_i(x)$. Also the trial function can be calculated by using Lagrange interpolating polynomial.

4.2.1 Lagarange Interpolation polynomial

We write the $\boldsymbol{i^{th}}$ term of the Lagrange polynomial as

$$l_i(x) = \frac{(x-x_0)(x-x_1)\ldots(x-x_{i-1})(x-x_{i+1})\ldots(x-x_n)}{(x_i-x_0)(x_i-x_1)\ldots(x_i-x_{i-1})(x_i-x_{i+1})\ldots(x_i-x_n)} \tag{4.23}$$

Using product notation, we may also write

$$l_i(x) = \prod_{j=0, i\neq j}^{n} \left(\frac{x-x_i}{x_i-x_j}\right) \tag{4.24}$$

The Lagrange form of the interpolating polynomial may be written as

$$y_n(x) = l_0(x)f_0 + l_1(x)f_1 + l_2(x)f_2 + \cdots + l_n(x)f_n$$

$$\boldsymbol{y_n(x) = \sum_{i=0}^{n} l_i(x)f_i} \tag{4.25}$$

4.2.2 Lagrange polynomial Error formula

Theorem 4.1: Suppose $x_0 < x_1 < \cdots < x_n$ are numbers in the interval $[a, b]$ and $f \in C^{n+1}[a, b]$. Then, for each x in $[a, b]$, a number $\xi(x)$ (generally unknown) between $x_0, x_1, x_2, \ldots, x_n,$ and hence in (a, b), exists with

$$f(x) = R(x) + \frac{f^{(n+1)}(\xi(x))(x-x_0)(x-x_1)\ldots(x-x_n)}{(n+1)!} \tag{4.26}$$

$R(x)$ is the interpolating polynomial.

proof: Note that if $x = x_k$ for any $k = 0,1, \dots, n$ then $f(x_k) = R(x_k)$ and choosing $\xi(x_k)$ arbitrary in (a, b). If $x \neq x_k$, for all $k = 0, 1, 2, \dots, n$ define the function g for s in (a, b) by

$$g(s) = f(s) - R(s) - [f(x) - R(x)] \frac{(s - x_0)(s - x_1) \dots (s - x_n)}{(x - x_0)(x - x_1) \dots (x - x_n)}$$

$$= f(s) - R(s) - [f(x) - R(x)] \Pi_{i=0}^{n} \frac{(s - x_i)}{(x - x_i)}$$

Since $f \in C^{n+1}[a, b]$ and $R \in C^{\infty}[a, b]$, it follows that, $g \in C^{n+1}[a, b]$.

For $s = x_k$, we have $g(x_k) = f(x_k) - R(x_k) - [f(x) - R(x)] \Pi_{i=0}^{n} \frac{(x_k - x_i)}{(x - x_i)} = 0 - [f(x) - R(x).0 = 0]$

Moreover, $g(x) = f(x) - R(x) - [f(x) - R(x)] \Pi_{i=0}^{n} \frac{(x - x_i)}{(x - x_i)} = f(x) - R(x) - [f(x) - R(x)] = 0$

Thus $g \in C^{n+1}[a, b]$, and g is zero at the $n + 2$ distinict numbers $x_0, x_1, \dots, x_n$. by generalized Rolles Theorem there exists a number ξ in $[a, b]$ for which $g^{n+1}(\xi) = 0$. so

$$g^{(n+1)}(\xi) = f^{(n+1)}(\xi) - R^{(n+1)}(\xi) - [f(x) - R(x)] \frac{d^{n+1}}{ds^{n+1}} \left[\prod_{i=0}^{n} \frac{(s - x_i)}{(x - x_i)} \right]_{s=\xi} = 0 \qquad (4.27)$$

However $R(x)$ is polynomial of degree at most n, so the $(n + 1)^{st}$ derivative , $R^{(n+1)}(x)$ is identically zero. Also $\prod_{i=0}^{n} \frac{(s - x_i)}{(x - x_i)}$ is a polynomial of degree $(n + 1)$, so

$\prod_{i=0}^{n} \frac{(s - x_i)}{(x - x_i)} = \left[\frac{1}{\prod_{i=0}^{n} (x - x_i)} \right] s^{n+1} + (lower\ degree\ term\ in\ s)$ and
$\frac{d^{n+1}}{ds^{n+1}} \prod_{i=0}^{n} \frac{(s - x_i)}{(x - x_i)} = \frac{(n+1)!}{\prod_{i=0}^{n} (x - x_i)}$ Equation (4.27) now becomes

$$f^{(n+1)}(\xi) - 0 - [f(x) - R(x)] \frac{(n+1)!}{\prod_{i=0}^{n} (x - x_i)} = 0 \qquad (4.28)$$

And upon solving for $f(x)$, we have

$$f(x) = R(x) + \frac{f^{(n+1)}}{(n+1)!} \prod_{i=0}^{n} (x - x_i) \qquad (4.29)$$

Where $\frac{f^{(n+1)}(\xi)}{(n+1)!} \prod_{i=0}^{n} (x - x_i)$ is the truncation error of $f(x)$ denoted by E_n

Hence the truncation error of the Lagrange polynomial of degree n is given by

$$E_n = \frac{f^{(n+1)}(\xi)}{(n+1)!} \prod_{i=0}^{n} (x - x_i) = f(x) - R(x) \qquad (4.30)$$

4.3 Cubic spline Interpolation

Definition (4.3.1): Cubic spline interpolation

Given a function f on $[a,b]$ and nodes $a = x_0 < x_1 < x_2 < x_3 < \ . \ . \ . \ < x_n = b$

A cubic spline interpolation f satisfied the following conditions (Kim. 2005 and Jain et al. 2007)

1. $S(x)$ is composed of cubic polynomial pieces $S_i(x)$ if $x \in [x_i, x_{i+1}]$, $i = 1,2, \dots, n-1$.

$$S(x) = \begin{cases} S_1(x) = y_1 + b_1(x-x_1) + c_1(x-x_1)^2 + d_1(x-x_1)^3, & on\ [x_1, x_2] \\ S_2(x) = y_2 + b_2(x-x_2) + c_2(x-x_2)^2 + d_2(x-x_2)^3, & on\ [x_2, x_3] \\ \vdots \\ \vdots \\ S_{n-1}(x) = y_{n-1} + b_{n-1}(x-x_{n-1}) + c_{n-1}(x-x_{n-1})^2 + d_{n-1}(x-x_{n-1})^3, & on\ [x_{n-1}, x_n] \end{cases}$$

2. $S_i(x_i) = f_i(x_i)$, for $i = 0,1,2, \ . \ . \ . \ , n$ ($i.e$ the spline matches function values)
3. $S_i(x_{i+1}) = S_{i+1}(x_{i+1}) = f_{i+1}$, for $i = 0, 1, 2, . \ . \ ., n-2$ ($i.e$ the spline is continuous)
4. $S_i^{'}(x_{i+1}) = S_{i+1}^{'}(x_{i+1})$, for $i = 0, \ 1, . \ . \ . \ , n-2$ ($i.e$ the spline $C^1[a,b]$)
5. $S_i^{''}(x_{i+1}) = S_{i+1}^{''}(x_{i+1})$, for $i = 0, 1, \ . \ . \ . , \ n-2$ ($i.e$ the spline $C^2[a,b]$)
6. Two end conditions: examples

   $i). S_i''(x_0) = S_i^{''}(x_n) = 0$ ($i.e$ Natural spline resulting from free boundary conditions)

   $ii). S_i^{'}(x_0) = f_i^{'}(x_0)$ and $S_i^{'}(x_n) = f_i^{'}(x_n)$ ($i.e$ complete or clamped end condition Resulting from clamped boundary condition)

   $iii). S_i^{'''}(x_0) = S_{n-1}^{'''}(x_{n-1}) = 0$ ($i.e$ Parabolically terminated boundary condition)

   $iv). S_i^{'''}(x_{i+1}) = S_{i+1}^{'''}(x_{i+1})$ ($i.e$ Not – a – Knot boundary condition)

4.3.1 Construction of Cubic spline

In the derivation of the cubic spline interpolation polynomial, we follow the above conditions one after the other. The objective of the cubic spline is to derive a third-order polynomial for each interval between knots as represented by a general cubic spline polynomial function given below

$$S_i(x) = a_i + b_i(x - x_i) + c_i(x - x_i)^2 + d_i(x - x_i)^3 \quad (4.31)$$

The first condition is that the cubic spline must pass through all data points. Hence we have

$$f_i = a_i + b_i\,(x - x_i\,) + c_i\,(x - x_i\,)^{\,2} + d_i\,(x - x_i\,)^3 \quad (4.31a)$$

$$\Longrightarrow f_i = a_i + b_i\,(x_i - x_i) + c_i\,(x_i - x_i\,)^{\,2} + d_i\,(x_i - x_i\,)^3 \quad (4.31b)$$

Hence $f_i = a_i$ (4.32)

Therefore the constant in each cubic must be equal to the value of the dependent variable at the beginning of the interval. Substituting (4.32) in (4.31), we have

$$S_i\,(x) = f_i + b_i\,(x - x_i\ \ ) + c_i\,(x - x_i\,)^2 + d_i\,(x - x_i\,)^3 \quad (4.33)$$

Next we apply the third condition that each of the cubic spline must join the knot For $(i + 1)$ knot, we have $_{+1}(x) = f_i + b_i\,(x_{i+1} - x_i) + c_i\,(x_{i+1} - x_i\,)^2 + d_i\,(x_{i+1} - x_i\,)^3$

$$\Longrightarrow f_{i+1} = f_i + b_i h_i + c_i\,h_i{}^2 + d_i h_i^3 \tag{4.34}$$

$where\ h_i = x_{i+1} - x_i$

Applying the fourth condition, the first derivation at the interior nodes must be equal.

Hence differentiating equation (4.33) yield, $'(x) = b_i + 2c_i\,(x - x_i\,) + 3d_i\,(x - x_i\,)^2$ (4.35)

The equivalent of the derivative at the interior node $(i + 1)$ is given as

$$b_i + 2c_i h_i + 3d_i\,h_i{}^2 = b_{i+1} \tag{4.36}$$

Applying the fifth condition, the second derivative at the interior nodes must also be equal. Hence differentiating equation (4.35) yield, $_i''(x) = 2c_i + 6d_i\,(x - x_i\,)$ (4.37)

The equivalent of the second derivative at the interior node $(i + 1)$ is given as

$$c_i + 3\,d_i h_i = c_{i+1} \tag{4.38}$$

Solving (4.38) for d_i, yields

$$d_i = \frac{C_{i+1} - C_i}{3h_i} \tag{4.39}$$

Substituting (4.39) into (4.34) yields

$$f_{i+1} = f_i + b_i\,h_i + \frac{h_i^2}{3}(2c_i + c_{i+1}) \tag{4.40}$$

Substituting (4.39) into (4.36) yields

$$b_{i+1} = b_i + h_i\,(c_i + c_{i+1}) \tag{4.41}$$

Equation (4.40) can be solved for b_i to get

$$b_i = \frac{f_{i+1} - f_i}{h_i} - \frac{h_i}{3}(2c_i + c_{i+1}), \tag{4.42}$$

Reducing the index by 1, equation (4.41) and (4.42) can be rewritten as

$$b_{i-1} = \frac{f_i - f_{i-1}}{h_{i-1}} - \frac{h_{i-1}}{3}\,(2c_{i-1} + c_i\,), \tag{4.43}$$

$$b_i = b_{i-1} + h_{i-1}(c_{i-1} + c_i\,) \tag{4.44}$$

Substituting equation (4.42) and (4.43) in (4.44) yields

$$\frac{f_{i+1} - f_i}{h_i} - \frac{h_i}{3}\,(2c_i + c_{i+1}) = \frac{f_i - f_{i-1}}{h_{i-1}} - \frac{h_{i-1}}{3}\,(2c_{i-1} + \ c_i) + h_{i-1}(c_{i-1} + c_i\,)$$

Further simplification yields

$$h_{i-1}c_{i-1} + 2(h_{i-1} + h_i\,) + h_i\,c_{i+1} = 3\,(\frac{f_{i+1} - f_i}{h_i}) - 3\,(\frac{f_i - f_{i-1}}{h_{i-1}}) \tag{4.45}$$

Using divided difference notation, $f\,[\,x_i, x_j\,] = \frac{f_i - f_j}{x_i - x_j}$

Equation (4.45) can be written as

$$h_{i-1}c_{i-1} + 2(h_{i-1} + h)\, c_i + h_i\, c_{i+1} = 3(f[x_{i+1}, x_i] - f[x_i, x_{i-1}]) \quad (4.46)$$

Equation (4.46) can be written for the interior knots, $i = 2, 3, \ldots, n-2$,This results in $(n-3)$ Simultaneous tri-diagonal system of equations with $(n-1)$ unknown coefficient $c_1, c_2, \ldots, c_{n-1}$. Using the last condition (boundary condition), the second derivative at the first node (that is equation (4.37) can be set to zero, given as

$$Si''(x) = 2c_i + 6di\,(x - x_i) = 0 \Longrightarrow c_i = 0$$

Similarly the same evaluation can be made at the last node, given as

$$S''_{n-1}(x_n) = 2c_{n-1} + 6d_{n-1}h_{n-1} = 0 \quad (4.47)$$

Recalling (4.38), equation (4.37) became

$$c_{n-1} + 3d_{n-1}h_{n-1} = c_n = 0 \quad (4.48)$$

Thus to impose a zero second derivative at the last node, we set $c_n = 0$

Rewriting equation (4.46) in matrix form, we have

$$\begin{pmatrix} 1 & . & . & . . & . & . & . \\ h_1 & 2(h_1+h_2) & h_2 & . . & . & . & . \\ . & h_2 & 2(h_1+h_2) & h_3 . & . & . & . \\ . & . & . & . . & . & . & . \\ . & . & . & . . & . & . & . \\ . & . & . & . . & . & . & . \\ . & . & . & . . & h_{n-2} & 2(h_{n-2}+h_{n-1}) & h_{n-1} \\ . & . & . & . . & . & . & 1 \end{pmatrix} \begin{pmatrix} c_1 \\ c_2 \\ c_3 \\ . \\ . \\ . \\ c_{n-1} \\ c_n \end{pmatrix} = \begin{pmatrix} 0 \\ 3(f[x_3,x_2] - f[x_2,x_1]) \\ . \\ . \\ . \\ . \\ 3(f[x_n,x_{n-1}] - f[x_{n-1},x_{n-2}]) \\ 0 \end{pmatrix} \quad (4.49)$$

In matrix vector form $H.C = F$

Where H is tri diagonal symmetric and diagonal dominant

$$2|h_{i-1} + h_i| > |h_i| + |h_{i-1}|$$

From (4.48) we can now solve for c's. Equation (4.39) and (4.42) can then be used to determine the remaining coefficient b and d.

Alternatively the above scheme for a_i, b_i, c_i and d_i can be found by solving a total of $(4n-4)$ equations with $(4n-4)$ unknowns using matrix method of solving a system of equations. This gotten from the condition 1 through 6 in Definition 4.3.1 From conditions 1 and 2, the spline function goes through the first and last point of the interval, yielding $2(n-1)$ equations as shown in equations (4.32) and (4.34) respectively.Also from condition 4 of Definition 4.3.1, the first derivative is continuous at each interior point, yielding $(n-2)$ equations as shown in equation 4.35. Another condition (the fifth condition) of Definition 4.3.1 requires that the second derivative is continuous at each interior point which leads to $(n-2)$ equations as shown in equation (4.37). The sixth condition of Definition 4.3.1, that the second derivative at the end knots is zero yields $(\frac{n}{4})$ equations. Finally this gives a total of $(4n-4)$ equations with $(4n-4)$ unknown. This can be solve using $Ax = b$

4.3.2 Cubic spline method for solving Boundary value problems

Consider problem (4.1) and let's solve by using cubic spline method, a finite set of grid points, $x_i,\ where\ i = 0,1,2,\cdots,n-1,n;$ established in the interval $[a,b]$ to $n+1$ uniformly sub intervals $x_i = a + ih,\ x_0 = a,\ x_n = b,\ and\ h = \frac{b-a}{n+1}$.

Let $S(x)$ be the cubic spline approximating the function $y(x)$ and let $S''(x_i) = M_i$. to derive the governing equation of $S(x)$, we observe that second derivative of a third order polynomial is a linear equation. This means that within each spline the second derivative is a linear function of x. For the interval $[x_{i-1}, x_i]$, this linear function can be written in the lgrange form:

$$S''_i(x) = \frac{x-x_i}{x_{i-1}-x_i}S_i''(x_{i-1}) + \frac{x-x_{i-1}}{x_i-x_{i-1}}S_i''(x_i) = \frac{1}{h}[(x_i - x)M_{i-1} + (x - x_{i-1})M_i] \qquad (4.50)$$

Where the values of the second derivative of the third order polynomial at the end points of the i^{th} interval are $S_i''(x_{i-1})$ and $S_i''(x_i)$. The third order polynomial in interval i can be determined by integrating equation (4.50) twice. The resulting expression contains two constants of integration. These two constants can be determined from the condition that the values of the polynomial at the knots are known. $S_i''(x_{i-1}) = y_{i-1}$ And $S_i''(x_i) = y_i$ once the constants of integration are determined, the equation of the third-order polynomial in interval i is given by:

$$S_i(x) = \frac{1}{h}\left[\frac{(x_i-x)^3}{6}M_{i-1} + \frac{(x-x_{i-1})^3}{6}M_i + \left(y_{i-1} - \frac{h^2}{6}M_{i-1}\right) + \left(y_i - \frac{h^2}{6}M_i\right)(x - x_{i-1})\right] \qquad (4.51)$$

In equation 4.51 the spline second derivatives M_i, are still not known. To determine them, we use the condition of first derivative continuity. From equation 4.53, we obtain by differentiation:

$$S_i'(x) = \frac{1}{h}\left[\frac{-3(x_i-x)^2}{6}M_{i-1} + \frac{3(x-x_{i-1})^2}{6}M_i - \left(y_{i-1} - \frac{h^2}{6}M_{i-1}\right) + \left(y_i - \frac{h^2}{6}M_i\right)\right] \qquad (4.52)$$

Setting $x = x_i$ in equation 4.52, we obtain the left hand derivative:

$$\begin{aligned} S_i'(x_i-) &= \frac{h}{2}M_i - \frac{1}{h}\left(y_{i-1} - \frac{h^2}{6}M_{i-1}\right) + \frac{1}{h}\left(y_i - \frac{h^2}{6}M_i\right) \\ &= \frac{1}{h}(y_i - y_{i-1}) + \frac{h}{6}M_{i-1} + \frac{h}{6}M_i \end{aligned} \qquad (4.53)$$

For ; $i = 1,2,\cdots,n;$

And the right hand derivative becomes:

$$S_i'(x_i+) = \frac{1}{h}(y_{i-1} - y_i) + \frac{h}{6}M_i + \frac{h}{6}M_{i+1} \qquad (4.54)$$

For; $i = 0,1,2,\cdots,n-1$

Equality of equation 4.52 and equation 4.53 produces the recurrence relation:

$$M_{i-1} + 4M_i + M_{i+1} = \frac{6}{h^2}(y_{i+1} - 2y_i + y_{i-1}), \qquad i = 1,2,3,\cdots,n-1 \qquad (4.55)$$

Then at $x = x_i$, the differential equation 4.1 gives

$$M_i + A_iS_i'(x_i) + B_iy_i = F_i \qquad (4.56)$$

But from (4.53) and (4.54), we have

$$S'(x_i -) = \frac{h}{6}(2M_i + M_{i-1}) + \frac{1}{h}(y_i - y_{i-1}) \quad (4.57)$$

And

$$S'(x_i +) = -\frac{h}{6}(2M_i + M_{i+1}) + \frac{1}{h}(y_{i+1} - y_i) \quad (4.58)$$

Substituting equation 4.57 and equation 4.58 in to equation 4.57, we obtain the equations

$$M_i + A_i\left[\frac{h}{6}(2M_i + M_{i-1}) + \frac{1}{h}(y_i - y_{i-1})\right] + B_i y_i = F_i \quad (4.59)$$

And

$$M_i + A_i\left[\frac{h}{6}(2M_i + M_{i+1}) + \frac{1}{h}(y_{i+1} - y_i)\right] + B_i y_i = F_i \quad (4.60)$$

Simplifying (4.59) and (4.60) we get

$$a_i M_{i-1} + b_i M_i + c_i y_{i-1} + d_i y_i = F_i, \qquad i = 1,2,3,\cdots,n \quad (4.61)$$

And

$$e_i M_i + g_i M_{i+1} + h_i y_i + k_i y_{i+1} = F_i, \qquad i = 0,1,2,3,\cdots,n-1 \quad (4.62)$$

Where, $a_i = \frac{h}{6}A_i, \quad b_i = 1 + \frac{h}{3}A_i, \quad c_i = -\frac{A_i}{h}, \; d_i = B_i + \frac{A_i}{h}$

$$e_i = 1 - \frac{h}{3}A_i, \quad g_i = -\frac{h}{6}A_i, \quad h_i = B_i - \frac{A_i}{h}, \quad and \quad k_i = \frac{A_i}{h} \quad (4.63)$$

Since from the boundary conditions $y_0 \; and \; y_n$ are known, equations (4.61) and (4.62) consists a system of $2n$ equations of $2n$ unknowns, these unknowns are $M_0, M_1, \cdots, M_n, y_1, y_2, \cdots, y_{n-1}$.

In matrix vector form

$$A.M = F$$

Where $M = [M_0, M_1, \ldots, M_n, y_1, \ldots, y_{n-1}]^T$

$$F = [F_1 - c_1, F_2, \ldots, F_n - d_n, F_0 - h_0, F_1, \ldots, F_{n-1} - k_{n-1}]^T$$

Where; A is $(2n) \times (2n)$ dimensional tri-diagonal symmetric and diagonal dominant.

4.3.3 Error Analysis for cubic spline

Theorem 4.3.1

If $y \in C^2[a,b]$, $a = x_0 < x_1 < x_2 < \cdots < x_n = b$, and if $s(x)$ is a cubic spline for which $s(x_i) = y_i, \quad i = 0,1,2,\ldots,n$ Then $max_{x_0 \le x \le x_n}|y(x) - s(x)| \le \frac{1}{2}Mh^2$, Where $h = x_{i+1} - x_i$, $i = 0,1,2,\ldots.,n$ And $M = max|y''(x)|$, $x_0 \le x \le x_n$. the errors in the spline derivatives can be obtained by using the operator notation. To find the errors in the first derivatives we start with the recurrence relation

$$M_{i-1} + 4M_i + M_{i+1} = \frac{3}{h}(y_{i+1} - y_{i-1}),$$

That is $s'(x_{i-1}) + 4s'(x_i) + s'(x_{i+1}) = \frac{3}{h}(y_{i+1} - y_{i-1}),$

Using the operator notation, the above equation can be written as

$$(E^{-1} + 4 + E)s'(x_i) = \frac{3}{h}(E - E^{-1})y_i. \tag{4.64}$$

Since $E = e^{hD}, where\ D = \frac{d}{dx}\ equation\ (4.64) becomes,$

$$(e^{-hD} + 4 + e^{hD})s'(x_i) = \frac{3}{h}(e^{hD} - e^{-hD})y_i \tag{4.65}$$

Now, $e^{hD} = 1 + hD + \frac{h^2D^2}{2!} + \frac{h^3D^3}{3!} + \frac{h^4D^4}{4!} + \frac{h^5D^5}{5!} + \cdots$

And $e^{-hD} = 1 - hD + \frac{h^2D^2}{2!} - \frac{h^3D^3}{3!} + \frac{h^4D^4}{4!} - \frac{h^5D^5}{5!} + \cdots$

Hence $e^{hD} + e^{-hD} = 2\left(1 + \frac{h^2D^2}{2} + \frac{h^4D^4}{24} + \frac{h^6D^6}{720} + \cdots\right)$

And $e^{hD} - e^{-hD} = 2\left(hD + \frac{h^3D^3}{6} + \frac{h^5D^5}{120} + \cdots\right)$

Using the above expression in (4.64), we obtain

$$\left[2\left(1 + \frac{h^2D^2}{2} + \frac{h^4D^4}{24} + \cdots\right) + 4\right]s'(x_i) = \frac{3}{h} \times 2\left(hD + \frac{h^3D^3}{6} + \frac{h^5D^5}{120} + \cdots\right)y_i = 6\left(D + \frac{h^2D^3}{6} + \frac{h^4D^5}{120} + \cdots\right)$$

The above equation simplifies to

$$s'(x_i) = \frac{6\left(D + \frac{h^2D}{6} + \frac{h^4D^5}{120} + \cdots\right)}{6 + h^2D^2 + \frac{h^4D^4}{12} + \cdots}y_i = \frac{D + \frac{h^2D^3}{6} + \frac{h^4D^5}{120} + \cdots}{1 + \frac{h^2D^2}{6} + \frac{h^4D^4}{72} + \cdots}y_i$$

$$= \left(D + \frac{h^2D^3}{6} + \frac{h^4D^5}{120} + \cdots\right)\left[1 + \left(\frac{h^2D^2}{6} + \frac{h^4D^4}{72} + \cdots\right)\right]^{-1} y_i$$

$$= \left(D + \frac{h^2D^3}{6} + \frac{h^4D^5}{120} + \cdots\right)\left[1 - \left(\frac{h^2D^2}{6} + \frac{h^4D^4}{72} + \cdots\right) + \left(\frac{h^2D^2}{6} + \frac{h^4D^4}{72} + \cdots\right)^2 - \cdots\right]y_i$$

$$= \left(D + \frac{h^2D^3}{6} + \frac{h^4D^5}{120} + \cdots\right)\left(1 - \frac{h^2D^2}{6} - \frac{h^4D^4}{72} - \cdots + \frac{h^4D^4}{36} + \cdots\right)y_i$$

$$= \left(D + \frac{h^2D^3}{6} + \frac{h^4D^5}{120} + \cdots\right)\left(1 - \frac{h^2D^2}{6} + \frac{h^4D^4}{72} - \cdots\right)y_i$$

$$= \left(D - \frac{h^2D^3}{6} - \frac{h^4D^5}{72} - \cdots + \frac{h^2D^3}{6} - \frac{h^4D^5}{36} + \frac{h^4D^5}{120}\right)y_i$$

$$= \left(D - \frac{1}{180}h^4D^5 + \cdots\right)y_i$$

Hence $s'(x_i) = y'_i - \frac{1}{180}h^4y_i^{(5)} + O(h_6)$ (4.66)

In a similar manner, we can derive the relations:

$$s''(x_i) = y''(x_i) - \frac{1}{12}h^2y^{iv}(x_i) + \frac{1}{360}h^4y^{vi}(x_i) + O(h^6) \tag{4.67}$$

4.4 Consistency, Stability and Convergence analysis

Let y be the exact analytic solution of the differential equation and Y be the numerical solution obtained by finite difference method. The quantity $\|Y - y\|$ is the value of the error.

We have the following definitions:

4.4.1 Stability

Definition 4.4.1.1: A numerical method is said to be stable if it produces a bounded solution which imitates the exact solution. Otherwise it is said to be unstable. When the difference between the computed numerical solution and the exact solution of the discrete equation remains bounded as time grows.

Definition 4.4.1.2: A numerical method is called Stable if there is a constant C independent of the time step or the number of time steps such that:

$$\|y^n\| < C \|y^0\|$$

Definition 4.4.1.3: The finite difference method for a linear BVP gives a sequence of matrix equations of the form $A^h y^h = F^h$ where h is the mesh width. We say that the method is stable if $(A^h)^{-1}$ exists for all h sufficiently small (for $h < h_0$ say) and if there is a constant C independent of h, such that

$$\|(A^h)^{-1}\| \leq C \ for \ all \ h < h_0$$

4.4.2 Consistency

Consistency of A linear system of Equations

Consider the system of *m* linear equations in *n* unknowns:

$$\begin{cases} a_{11}x_1 + a_{12}x_2 + \cdots + a_{1n}x_n = b_1 \\ a_{21}x_1 + a_{22}x_2 + \cdots + a_{2n}x_n = b_2 \\ \vdots \\ \vdots \\ a_{m1}x_1 + a_{m2}x_2 + \cdots + a_{mn}x_n = b_m \end{cases} \tag{4.68}$$

The Matrix

$$A = \begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \vdots & \vdots & \vdots & \vdots \\ a_{m1} & a_{m2} & \cdots & a_{mn} \end{bmatrix}$$

Is called the coefficient matrix, and the matrix defined by

$$(A,b)=\begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1n} & b_1 \\ a_{21} & a_{22} & \cdots & a_{2n} & b_2 \\ \vdots & \vdots & \vdots & \vdots & \\ a_{m1} & a_{m2} & \cdots & a_{mn} & b_m \end{bmatrix}$$

A Is called the augmented matrix. If $r(A)$ the rank of A and $r(A,b)$ that of (A,b), then the following theorem holds true

Theorem 4.1 If $r(A) < r(A,b)$, then the equations defined by (4.68) are inconsistent and there will be no solution; if $r(A) = r(A,b)$, the equations are consistent and there exists at least one solution to the system.

Definition 4.4.2.1: The finite difference approximation is called Consistent with the differential equation if the truncation error goes to zero when the numerical parameters are made arbitrarily small. A numerical method is said to be consistent if

$$lim_{h\to 0}\|\eta_n\| = 0.$$

Where η_n is the discretization error.

According to the interpolation of the discretization error stated after that equation, any consistent numerical method closely matches the original differential equation when the step size h is sufficiently small.

4.4.3 Convergent

If the approximate solution Y_n approaches the exact solution $Y(x_n)$ as $h = \Delta x$ tends to zero provided the rounding errors arising from the initial conditions approaches zero. Which means as a method is continually refined by taking smaller and smaller step sizes, the sequence of approximate solutions must converge to the exact solution.

Definition 4.4.3.1: A scheme is called converge to $O(\Delta t^p)\ if\ \|Y-y\| = O(\Delta t^p)\ as\ \Delta t \to 0$, Where p is a positive constant

Definition 4.4.3.2: A numerical method is called convergent if its global error computed up to a given x satisfies:

$$\lim_{h\to 0}\|\xi\| = \lim_{h\to 0}\|y-Y\| = \lim_{h\to 0} max_n\|y_n - Y_n\| = 0$$

Theorem 4.2 (Lax-Theorem):

A consistent finite difference scheme for a partial differential equation for which, the initial value problem is convergent if and only if it is stable.

Consistency + Stability ⟺ Convergence

proof:

Consistency of the method means that the local truncation error at each step , hT_n, is sufficiently small so that the accumulated ($i.e$, global) error, which is on the order of T_n tends to zero as h is decreased. Thus:

$$Consistency \implies \|y - Y\| \text{ Is small} \qquad 4.68$$

Where, as above, Y is the ideal solution of the numerical scheme and y is the analytic solution. Stability of the method means that if at any given step the actual solution U_n slightly deviates from the ideal solution Y_n due to the round-off error, then this deviation remains small and does not converges as h increases. Thus

$$Stability \implies \|Y - U\|, \text{ is small.} \qquad 4.69$$

Equations 4.68 and equation 4.69 together imply that the maximum difference between the actual computed solution u_n and the exact solution y_n also remains small.

$$\|y - U\| = \|(y - Y) + (Y - U)\| \leq \|(y - Y)\| + \|(Y - U)\| \qquad 4.70$$

This must be small because each term of the right hand side is small. The fact that the left hand side of the above equation is small which means by definition (4.4.3.1), that the method is convergent.

4.5. Numerical Example and result

In this study we select a second order boundary value problem with mesh size $h = 0.25$ and $h = 0.125$ respectively. The exact solution, the approximate solution and absolute errors are tabulated in table (4.5.4) and comparison are shown in figures (4.5.4) and (4.5.5).the result is compared with exact solution of the problem and Galerkin finite element method, and cubic spline methods with finite difference method.

Example 1

Solve the following boundary - value problem given by

$$\begin{cases} \dfrac{d^2y}{dx^2} - \dfrac{dy}{dx} = -1\,, & 0 \leq x \leq 1 \\ with\ y(0) = 0\ \ and & y(1) = 0 \end{cases}$$

4.5.1 Solution using finite difference method

First, the discrete form of the Domain is as follows

Choose: $h = \Delta x = \frac{1}{4} = 0.25$

$$x_0 = 0\,,\ x_1 = 0.25\,,\ x_2 = 0.5\,,\ x_3 = 0.75\,,\ x_4 = 1$$

The given differential equation is

$$y'' - y' = -1 \tag{4.5.1}$$

Then substitute the differentials by finite difference schemes given in (4.7)

$$\left(\frac{y_{i+1}-2y_i+y_{i-1}}{h^2}\right) - \left(\frac{y_{i+1}-y_{i-1}}{2h}\right) = -1, \qquad i = 1, 2, \ldots, n-1 \tag{4.5.2}$$

Using equation (4.5.2) for $i = 1\ 2, 3$ we get

$$\begin{cases} 32y_1 - 14y_2 = 1 \\ -18y_1 + 32y_2 - 14y_3 = 1 \\ -18y_2 + 32y_3 = 1 \end{cases} \tag{4.5.3}$$

The linear system of equations (4.5.3) can be writing as a tri-diagonal matrix of the form

$$\begin{pmatrix} 32 & -14 & 0 \\ -18 & 32 & -14 \\ 0 & -18 & 32 \end{pmatrix}\begin{pmatrix} y_1 \\ y_2 \\ y_3 \end{pmatrix} = \begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix} \tag{4.5.4}$$

By using Thomas algorithm method from the above tri-diagonal matrix form

Step 1: Let $[a_2\,, a_3] = [-18\,, -18]$

$[b_1, b_2, b_3] = [32\,, 32\,, 32]$

$[c_1, c_2] = [-14\,, -14]$

First put $u_1 = b_1 = 32$

And $u_i = b_i - \frac{a_i c_{i-1}}{u_{i-1}},\ \ i = 2,3$

$$u_2 = b_2 - \frac{a_2 c_1}{u_1} = 32 - \frac{-18 \times -14}{32} = 24.125$$

$$u_3 = b_3 - \frac{a_3 c_2}{u_2} = 32 - \frac{-18 \times -14}{24.125} = 10.445595855$$

Step 2: Let $[d_1, d_2, d_3] = [1\,, 1\,, 1]$

Put $z_1 = \frac{d_1}{b_1} = \frac{1}{32} = 0.03125$

And $z_i = \frac{d_i - a_i z_{i-1}}{u_i},\ i = 2, 3$

$$z_2 = \frac{d_2 - a_2 z_1}{u_2} = \frac{1 - (-18 \times 0.03125)}{24.125} = 0.064766839$$

$$z_3 = \frac{d_3 - a_3 z_2}{u_3} = \frac{1 - (-18 \times 0.064766839\,)}{10.445595855} = 0.207341269$$

Step 3: If $y_i = [y_1, y_2, y_3]$ then

Put $y_i = z_i - \frac{c_i y_{i+1}}{u_i}, \quad i = 1, 2, 3$

$$y_3 = z_3 - \frac{c_3 y_4}{u_3} = 0.100480769230769$$

$$y_2 = z_2 - \frac{c_2 y_3}{u_2} = 0.064766839 - \frac{(-14 \times 0.207341269)}{24.125} = 0.123076923076923321$$

$$y_1 = z_1 - \frac{c_1 y_2}{u_1} = 0.03125 - \frac{(-14 \times 0.127897394)}{32} = 0.085096153846154163$$

Therefore, the numerical solutions for the system of equations given by:

$$y_1 = 0.085096153846154163$$

$$y_2 = 0.123076923076923321$$

$$y_3 = 0.100480769230769178$$

The exact solution is (by using the principle of superposition): $Y(x) = x + \frac{1-e^x}{e-1}$

$$Y_1 = Y(x_1) = 0.084703823328880$$

$$Y_2 = Y(x_2) = 0.122459331201855$$

$$Y_3 = Y(x_3) = 0.099932008758773$$

Therefore the absolute error for FDM, given by:

$$|Y_1 - y_1| = 0.000392330517274163$$

$$|Y_2 - y_2| = 0.000617591875068321$$

$$|Y_3 - y_3| = 0.000548760471996178$$

The absolute maximum error is $E = 0.000617591875068321$

Table (4.5.1) *shows the comparison of analytical solutions with the finite difference method for Example 1*

x_i	$exact\ solution$	$FDM\ solution$	$Error(E)$
0.0000	0.000000000000000	0.00000000000000000	0.000000000000000000
0.2500	0.084703823328880	0.085096153846154163	0.000392330517274163
0.5000	0.122459331201855	0.123076923076923321	0.000617591875068321
0.7500	0.099932008758773	0.100480769230769178	0.000548760471996178
1.0000	0.000000000000000	0.000000000000000000	0.000000000000000000

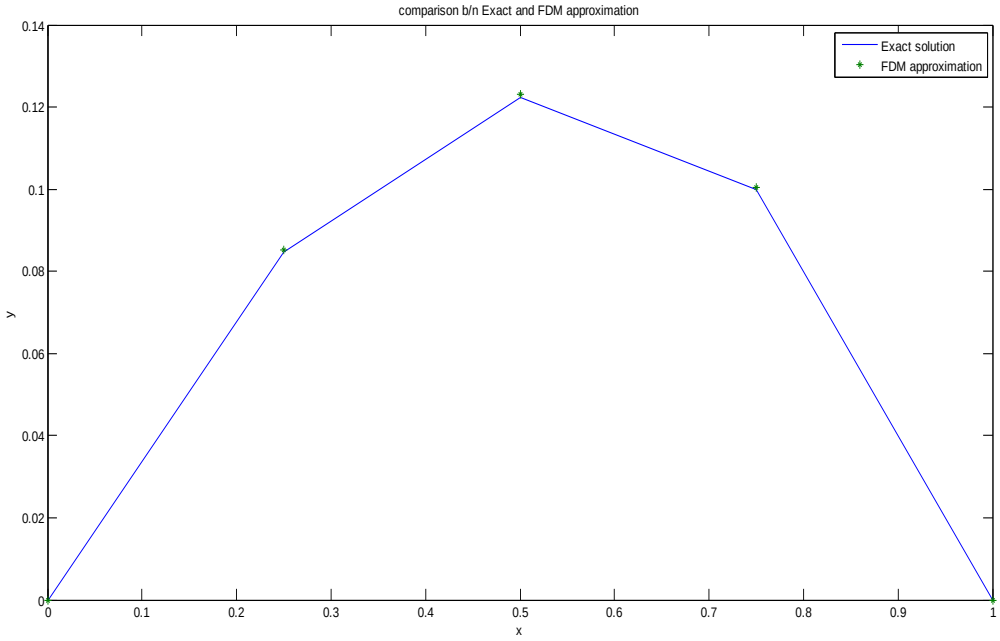


Figure (4.5.1): The *comparison between exact solutions with FDM approximation for example 1*

4.5.2 Solution using Galerkin's finite element method

We consider again the problem in Example 1

$$y'' - y' = -1, \quad 0 \le x \le 1$$
$$y(0) = y(1) = 0$$

First, discretise the interval [0, 1] in to four equal parts

$$x_0 = 0 < x_1 = 0.25 < x_2 = 0.5 < x_3 = 0.75 < x_4 = 1$$

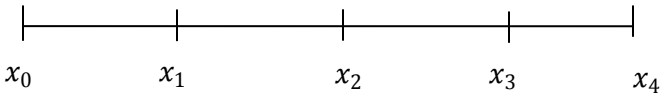


fig. 4.2 Discritisation of 1-D for $h = 0.25$

Apply the Galerkin's finite element method to determine the value of $y(x_i), \quad where \quad i = 1,2,3$

As our approximation, choose trial function

$$u(x) = \phi_0(x) + \sum_{i=1}^{3} C_i \phi_i(x)$$

Where, $\phi_1(0) = \phi_1(1) = 0 \ and \quad y(0) = y(1) = 0$,

$$\phi_0(x) = y(0) + \big(y(1) - y(0)\big)\left(\frac{x-0}{1-0}\right)$$

$$\phi_0(x) = 0$$

Using Lagrange polynomial as a base

$$u(x) = \frac{(x-x_0)(x-x_2)(x-x_3)(x-x_4)}{\left(\frac{1}{4}-0\right)\left(\frac{1}{4}-\frac{1}{2}\right)\left(\frac{1}{4}-\frac{3}{4}\right)\left(\frac{1}{4}-1\right)} y_1 + \frac{(x-x_0)(x-x_1)(x-x_3)(x-x_4)}{\left(\frac{1}{2}-0\right)\left(\frac{1}{2}-\frac{1}{4}\right)\left(\frac{1}{2}-\frac{3}{4}\right)\left(\frac{1}{2}-1\right)} y_2 + \frac{(x-x_0)(x-x_1)(x-x_2)(x-x_4)}{\left(\frac{3}{4}-0\right)\left(\frac{3}{4}-\frac{1}{4}\right)\left(\frac{3}{4}-\frac{1}{2}\right)\left(\frac{3}{4}-1\right)} y_3$$

Simplifying the above equation, we get

$$u(x) = C_1 x(8x^3 - 18x^2 + 13x - 3) + C_2 x(16x^3 - 32x^2 + 19x - 3) + C_3 x(8x^3 - 14x^2 + 7x - 1)$$

Then

$$u'(x) = C_1(32x^3 - 54x^2 + 26x - 3) + C_2(64x^3 - 96x^2 + 38x - 3) + C_3(32x^3 - 42x^2 + 14x - 1)$$

$$u''(x) = C_1(96x^2 - 108x + 26) + C_2(192x^2 - 192x + 38) + C_3(96x^2 - 84x + 14)$$

Base functions are:

$$\phi_1(x) = x(8x^3 - 18x^2 + 13x - 3)$$
$$\phi_2(x) = x(16x^3 - 32x^2 + 19x - 3)$$
$$\phi_3(x) = x(8x^3 - 14x^2 + 7x - 1)$$

Where $\phi_1(0) = \phi_2(0) = \phi_3(0) = 0 \ and \ \phi_1(1) = \phi_2(1) = \phi_3(1) = 0$

The residual is, $R = y'' - y' + 1$

$$R(u) = y''(u) - y'(u) + 1 = 0$$

Using the Galerkin's weak formulation

$$\int_0^1 \phi_i(x) R(x) dx = 0, \qquad i = 1, 2, 3$$

If we substitute for i we get three equations containing three unknowns given below

$$\begin{cases} -0.619047619047619C_1 - 0.652380952380947C_2 - 0.199999999999995C_3 = 0.066666666666667 \\ -0.757142857142799C_1 - 1.476190476190376C_2 - 0.652380952380910C_3 = -0.033333333333333 \\ -0.238095238095295C_1 - 0.757142857142966C_2 - 0.619047619047671C_3 = 0.066666666666666 \end{cases}$$

In matrix form

$$\begin{pmatrix} -0.619047619047619 & -0.652380952380947 & -0.199999999999995 \\ -0.757142857142799 & -1.476190476190376 & -0.652380952380910 \\ -0.238095238095295 & -0.757142857142966 & -0.619047619047671 \end{pmatrix} \begin{pmatrix} C_1 \\ C_2 \\ C_3 \end{pmatrix} = \begin{pmatrix} 0.066666666666667 \\ -0.033333333333333 \\ 0.066666666666666 \end{pmatrix}$$

Solving this system of equations we get

$[C_1, \quad C_2, \quad C_3] = [-0.451671511627908, \quad 0.489825581395349, \quad -0.533066860465116]$

Therefore the trial solution is

$$u(x) = -0.451671511627908x(8x^3 - 18x^2 + 13x - 3) + 0.489825581395349x(16x^3 - 32x^2 + 19x - 3) - 0.533066860465116x(8x^3 - 14x^2 + 7x - 1)$$

Therefore the numerical solutions obtained by using Galerkin's method are

$$y_1 = 0.084688408430233$$
$$y_2 = 0.122456395348837$$
$$y_3 = 0.099950036337209$$

The exact solution: $Y(x) = x - \frac{1-e^x}{e-1}$

$$Y_1 = Y(x_1) = 0.084703823328880$$
$$Y_2 = Y(x_2) = 0.122459331201855$$
$$Y_3 = Y(x_3) = 0.099932008758773$$

Therefore the absolute error for Galerkin's FEM are calculated as

$$|Y_1 - y_1| = 0.000015414898647$$
$$|Y_2 - y_2| = 0.000002935853018$$
$$|Y_3 - y_3| = 0.000018027578436$$

The absolute maximum error is $E = 0.000018027578436$

Table (4.5.2) *shows the comparison of analytical solution with Galerkin's finite element method*

x_i	$exact\ solution$	$Galerkins\ solution$	$Error(E)$
0.0000	0.00000000000000	0.00000000000000	0.000000000000000
0.2500	0.084703823328880	0.084688408430233	0.000015414898647
0.5000	0.122459331201855	0.122456395348837	0.000002935853018
0.7500	0.099932008758773	0.099950036337209	0.000018027578436
1.0000	0.00000000000000	0.00000000000000	0.000000000000000

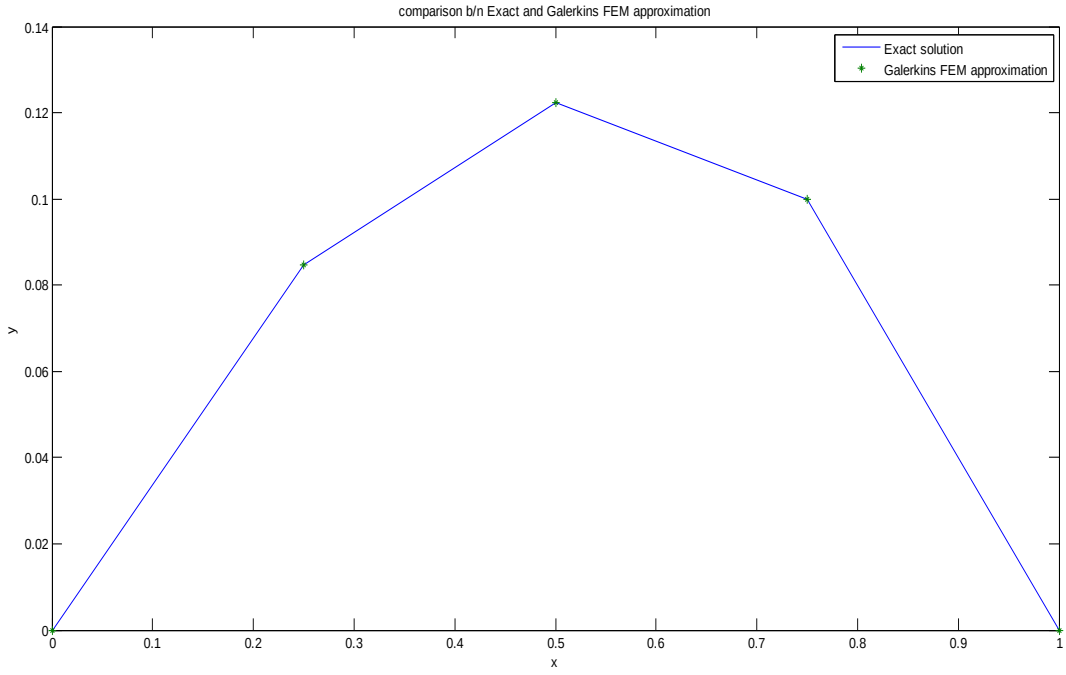


Figure 4.5.2: The *relationship between exact solutions with Galerkins FEM solution of example1 for h=0.25*

4.5.3 Numerical Solution by the cubic spline Method

We consider problem of Example 1

$$y'' = y' - 1, \quad 0 \le x \le 1$$
$$y(0) = y(1) = 1$$

First we discritize the interval [0, 1] in to four equal parts

$$x_0 = 0 < x_1 = 0.25 < x_2 = 0.5 < x_3 = 0.75 < x_4 = 1$$

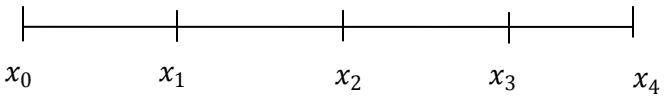


fig. 4.5.3 Discritisation of 1-D for $h = 0.25$

Setting $y'' = M_i$ the above equation become

$$M_i = y' - 1$$

Using the expressions in (4.57) and (4.58), we obtain

$$M_i = -\frac{h}{3}M_i - \frac{h}{6}M_{i+1} + \frac{y_{i+1}-y_i}{h} - 1 \qquad i = 0, 1, 2, 3$$

And

$$M_i = \frac{h}{3}M_i + \frac{h}{6}M_{i-1} + \frac{y_i-y_{i-1}}{h} - 1 \qquad i = 1, 2, 3, 4$$

These two equations gives 8×8 system of equations

$$26M_0 + M_1 - 96y_1 = -24$$
$$26M_1 + M_2 + 96y_1 - 96y_2 = -24$$
$$26M_2 + M_3 + 96y_2 - 96y_3 = -24$$
$$26M_3 + M_4 + 96y_3 = -24$$
$$M_0 - 22M_1 + 96y_1 = 24$$
$$M_1 - 22M_2 - 96y_1 + 96y_2 = 24$$
$$M_2 - 22M_3 - 96y_2 + 96y_3 = 24$$
$$M_3 - 22M_4 - 96y_3 = 24$$

In matrix form

$$\begin{pmatrix} 26 & 1 & 0 & 0 & 0 & -96 & 0 & 0 \\ 0 & 26 & 1 & 0 & 0 & 96 & -96 & 0 \\ 0 & 0 & 26 & 1 & 0 & 0 & 96 & -96 \\ 0 & 0 & 0 & 26 & 1 & 0 & 0 & 96 \\ 1 & -22 & 0 & 0 & 0 & 96 & 0 & 0 \\ 0 & 1 & -22 & 0 & 0 & -96 & 96 & 0 \\ 0 & 0 & 1 & -22 & 0 & 0 & -96 & 96 \\ 0 & 0 & 0 & 1 & -22 & 0 & 0 & -96 \end{pmatrix} \begin{pmatrix} M_0 \\ M_1 \\ M_2 \\ M_3 \\ M_4 \\ y_1 \\ y_2 \\ y_3 \end{pmatrix} = \begin{pmatrix} -24 \\ -24 \\ -24 \\ -24 \\ 24 \\ 24 \\ 24 \\ 24 \end{pmatrix}$$

Therefore, by solving the tri-diagonal matrix we get

$$S_1 = y_1 = 0.085096153846154$$

$$S_2 = y_2 = 0.123076923076923$$

$$S_3 = y_3 = 0.100480769230769$$

The analytic solutions given by

$$Y_1 = \ 0.084703823328880$$

$$Y_2 = 0.122459331201855$$

$$Y_3 = 0.099932008758773$$

Therefore the absolute error for cubic spline are calculated as

$$|Y_1 - y_1| = 0.000392330517274163$$

$$|Y_2 - y_2| = 0.000617591875068321$$

$$|Y_3 - y_3| = 0.000548760471996178$$

The absolute maximum error is $E = 0.000617591875068321$

Table (4.5.3) *shows the comparison of analytical solution with Cubic spline method for h=0.25*

x_i	***exact solution***	***cubic spline appro.***	$Error(E)$
0.0000	0.0000000000000000	0.000000000000000000	0.000000000000000000
0.2500	0.084703823328880	0.085096153846154163	0.000392330517274163
0.5000	0.122459331201855	0.123076923076923000	0.000617591875068000
0.7500	0.099932008758773	0.100480769230769178	0.000548760471996178
1.0000	0.0000000000000000	0.000000000000000000	0.000000000000000000

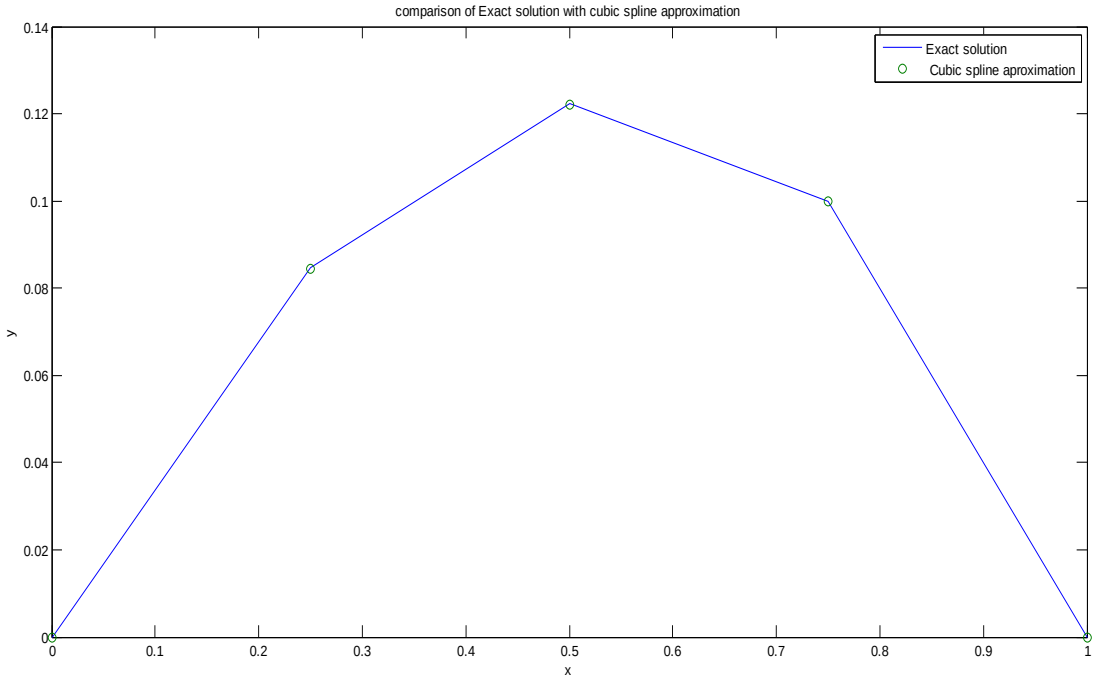


Figure (4.5.3): The *relationship between exact solutions with Cubic spline solution of example 1 for h=0.25*

Table (4.5.4) *shows the maximum absolute errors of Galerkins, the cubic spline and FDM,with Respect to the exact solution for* $h = 0.25$

Δx	**Methods**	**Absolute max. error**
0.25	Galerkins method	0.000018027578436000
	Cubic spline method	0.000617591875068321
	FDM method	0.0006175918750683210

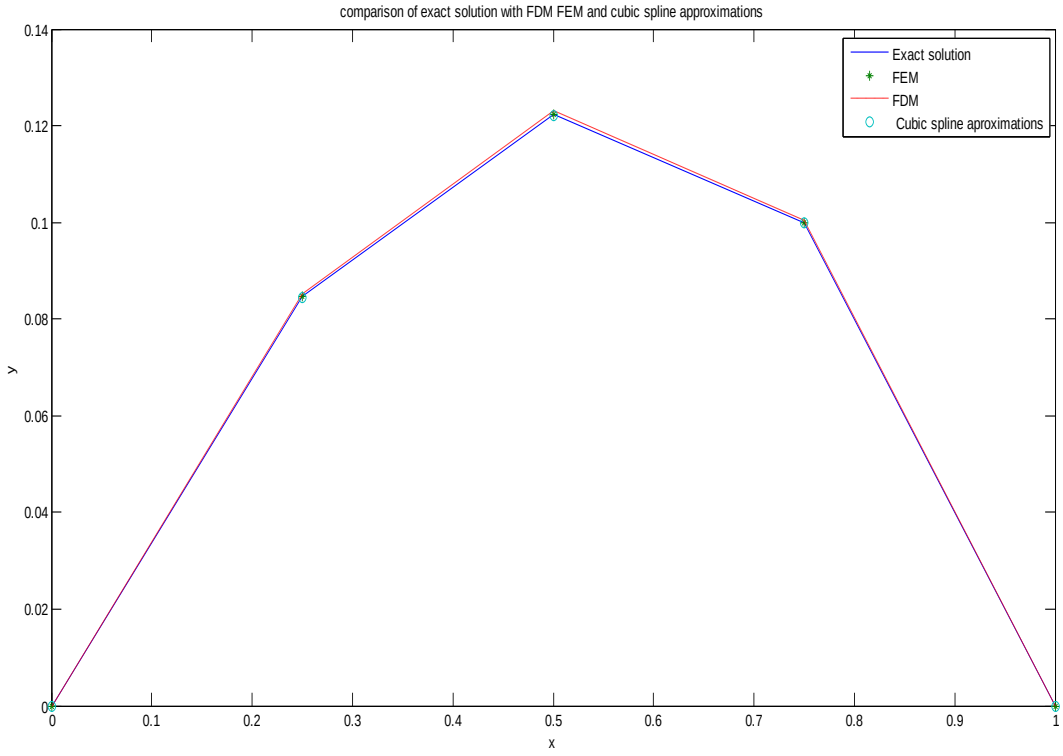


Figure (4.5.4): The *comparison of exact solutions with FDM,Galetkin's FEM and Cubic spline Approximations of Example (1) for h=0.25*

Table (4.5.5) *shows the comparison of analytical solutions with the finite difference method (FDM) of Example 1 for h=0.125*

x_i	***exact solution***	***FDM solution***	$Error(E)$
0.0000	0.000000000000000	0.00000000000000	0.000000000000000
0.1250	0.047510701759423	0.047563094639292	0.000052392879869
0.2500	0.084703823328880	0.084801268563823	0.000097445234943
0.3750	0.110205594866683	0.110337865678292	0.000132270811609
0.5000	0.122459331201855	0.122612675741356	0.000153344539501
0.6250	0.119701076940990	0.119857460479496	0.000156383538506
0.7500	0.099932008758773	0.100068216516054	0.000136207757281
0.8750	0.060887163096167	0.060973740024154	0.000086576927987
1.0000	0.000000000000000	0.000000000000000	0.000000000000000

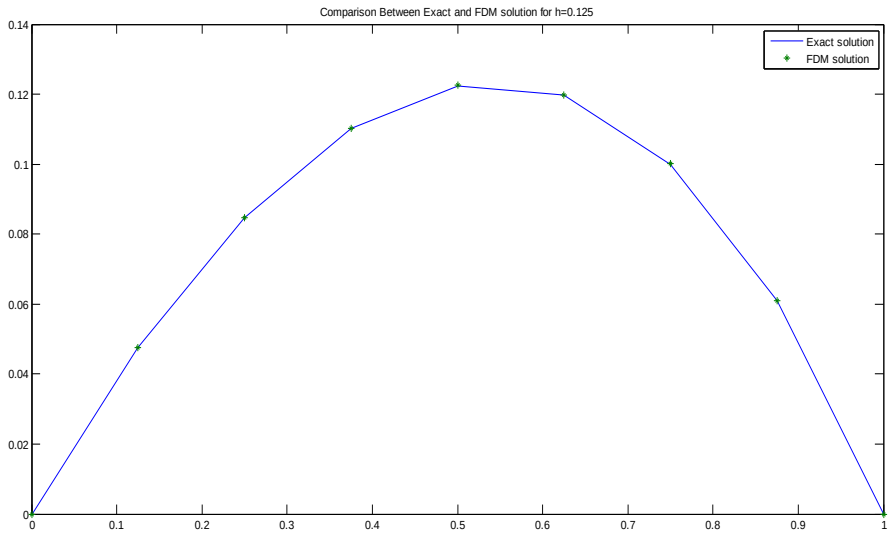


Figure (4.5.5): *The comparison between exact solutions with FDM approximation of Example 1 for h=0.125*

Table (4.5.6) *shows the comparison of analytical solution with Galerkin's finite element method for h=0.125*

x_i	***exact solution***	***Galerkins solution***	$Error(E)$
0.0000	0.000000000000000	0.0000000000000000	0.0000000000000000
0.1250	0.047510701759423	0.047507463143118	0.000003238616305
0.2500	0.084703823328880	0.084702477466908	0.000001345861972
0.3750	0.110205594866683	0.110202746157338	0.000002848709345
0.5000	0.122459331201855	0.122457971952142	0.000001359249713
0.6250	0.119701076940990	0.119699276442770	0.000001800498220
0.7500	0.099932008758773	0.09994221511607	0.000010206357297
0.8750	0.060887163096167	0.060886077948122	0.000001085148045
1.0000	0.000000000000000	0.0000000000000	0.00000000000000

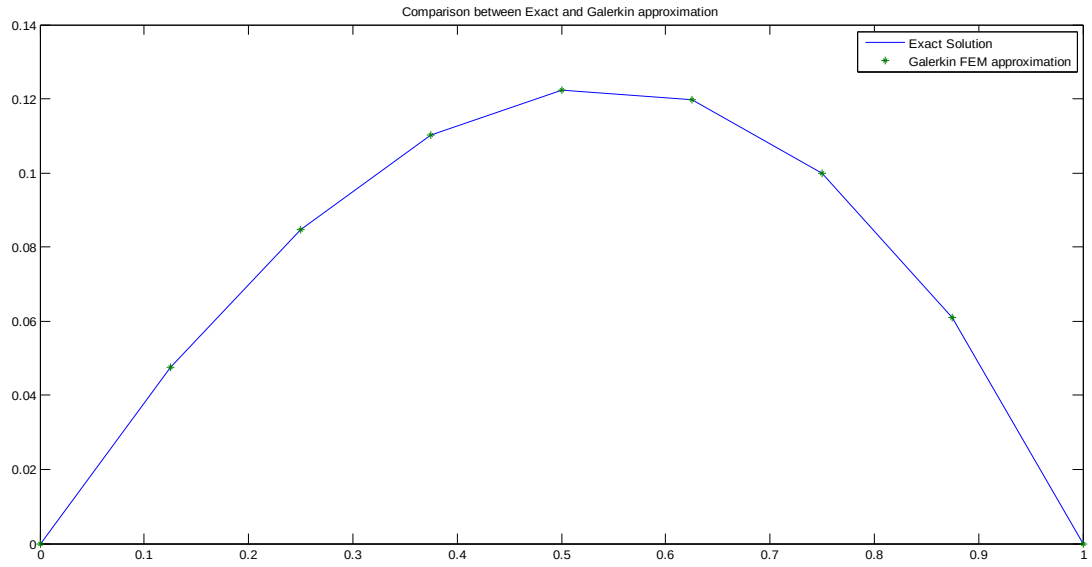


Figure 4.5.6: *The relationship between exact solutions with Galerkins FEM solution of* **E***xample1 for h=0.125*

Table (4.5.7) *shows the comparison of analytical solution with Cubic spline method for h=0.125*

x_i	***exact solution***	***cubic spline appro.***	$Error(E)$
0.0000	0.000000000000000	0.00000000000000	0.000000000000000
0.1250	0.047510701759423	0.047458308884702	0.00005239287472100
0.2500	0.084703823328880	0.084606378096339	0.000097445232541
0.3750	0.110205594866683	0.110073324056107	0.00013227081057600
0.5000	0.122459331201855	0.122305986668204	0.000153344533651
0.6250	0.119701076940990	0.119544693407895	0.00015638353309500
0.7500	0.099932008758773	0.099795801004187	0.0001362077545860
0.8750	0.060887163096167	0.060800586172098	0.000086576924069000
1.0000	0.000000000000000	0.000000000000000	0.000000000000000

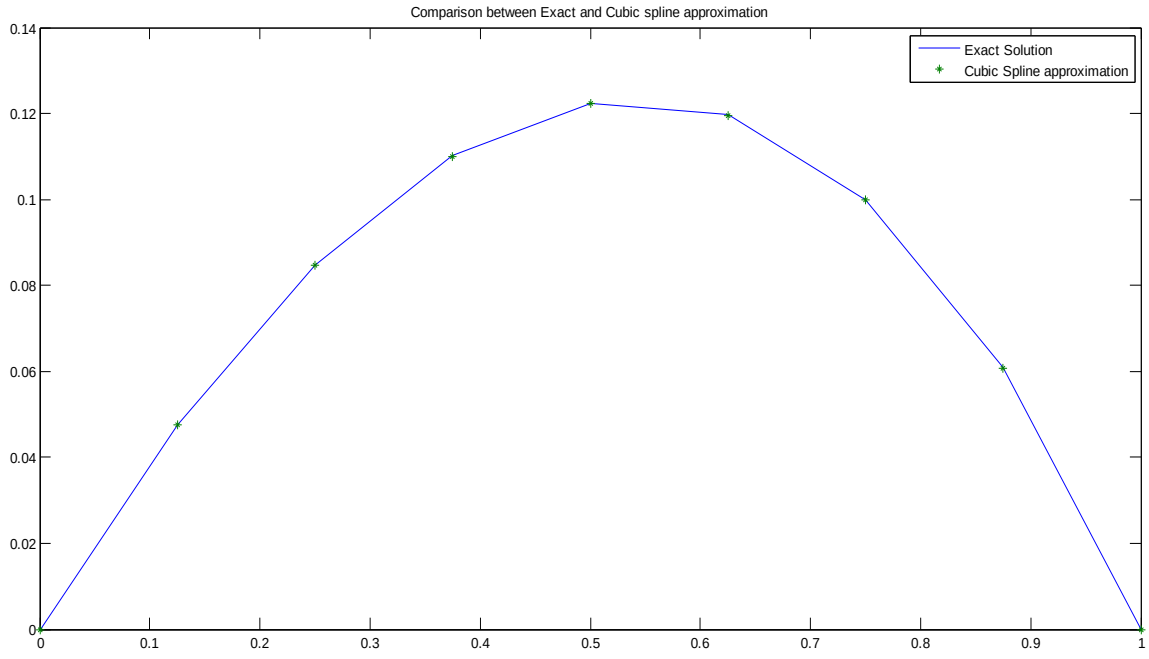


Figure (4.5.7): *The relationship between exact solutions with Cubic spline solution of example1 for h=0.125*

Table (4.5.8) *shows the maximum absolute errors of Galerkins, the cubic spline and FDM, with Respect to the exact solution for* $h = 0.125$

Δx	Methods	Absolute max. error
0.125	Galerkins method	0.000003238616305
	Cubic spline method	0.000156383533095
	FDM method	0.000156383533856

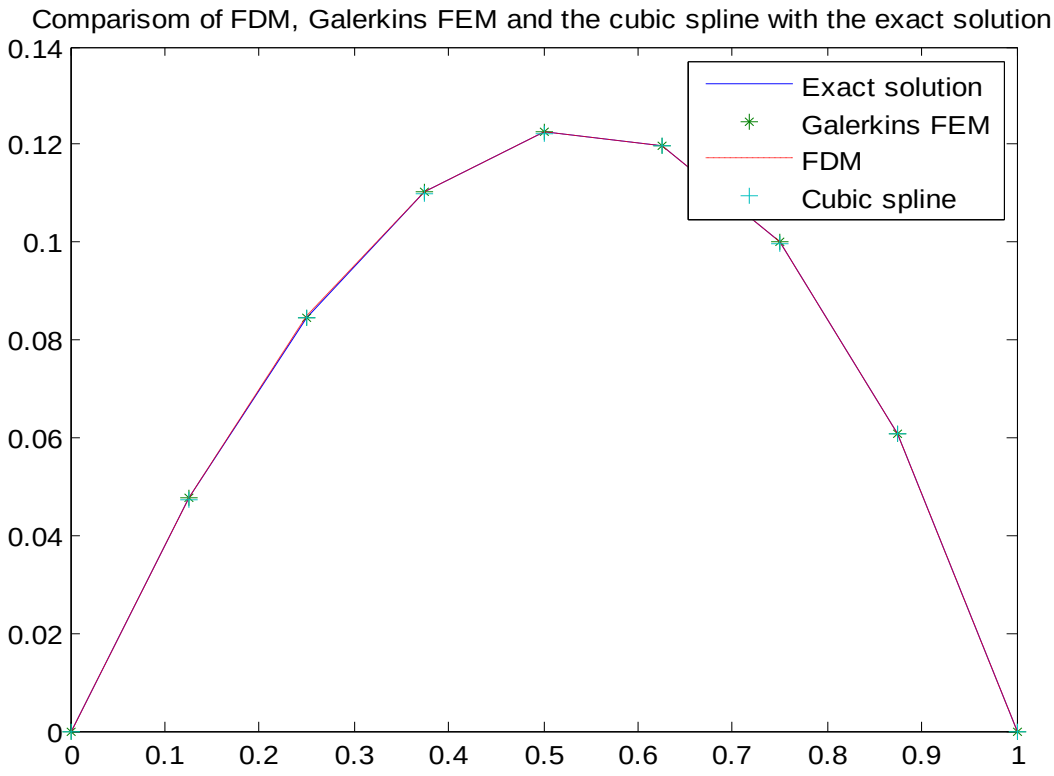


Figure (4.5.8): The *comparison of exact solutions with FDM, Galetkin's FEM and Cubic spline Approximations of Example (1) for h=0.125*

4.6 Discussion

We have solved a test example on second order boundary value problem with a given boundary conditions using finite difference method (FDM), Galerkins finite element method and the cubic spline method. To check the accuracy of the proposed numerical schemes, we first discreatize the domain in n sub intervals (mesh points) and using the methods, we construct a tri- diagonal matrix system then solve by using Thomas Algorithm or MATLAB.

From the tables 4.5.4 and 4.5.8, we can observe that the dependence of error on step sizes for finite difference method, Galerkins finite element method and the cubic spline method. As number of sub intervals, increases, the maximum absolute error for all methods decreases. Equivalently, with decreasing in the number of sub intervals (or meshes) the error increases.

Figures (4.5.1) and (4.5.5) show the behavior of approximated solutions using finite difference method. Figure (4.5.2) and (4.5.6) show the approximated solutions for different values using Galerkin's finite element method and Figure (4.5.3) and (4.5.7) show the behavior of solutions for different values using Cubic spline method.

Generally, comparisons of the three methods are shown in tables 4.5.4 and 4.5.8. As the number of discretization increases, the numerical solution becomes more stable and consistent. From tables 4.5.4 and 4.5.8 we conclude that the absolute maximum error of Galerkin's and the Cube spline methods are smaller than that of finite difference method. Therefore, Galerkin's finite element and the cubic spline methods are more accurate than that of finite difference method.

CHAPTER 5
5. Conclusion and Recommendation

5.1 Conclusion

The research done in this thesis clearly explains the Finite difference method, Galerkin's Finite element method and the cubic spline method. The main idea of FDM is to replace the differentials with finite difference schemes in the discrete domain and solving the tri-diagonal matrix system formed, Here we observe the results that obtained in tested example (1) using tables 4.5.1, 4.5.2 and 4.5.3 which show the absolute maximum errors. From table 4.5.4 and 4.5.8 as the step size approaches to zero, it produces a bounded solution, which closes to the exact solution. From definition (4.4.1.1), the absolute maximum error is smaller as the step size approaches to zero this tells us the method is stable, also by definition (4.4.1.2) and definition (4.4.3.1), as the step size approaches to zero the numerical deviation is close to zero, so the method is consistent. The consistency and the stability also show the convergence of the method by theorem (4.1). Therefore, as the step size approaches to zero, the numerical solution of finite difference method is convergent to the exact solution. Hence from the test example one the stability, consistency and convergence of the numerical solution of finite difference method depends on the conditions of the discretization of domain of the problem.

Galerkins-finite element method was also used to find the approximate solutions of two-point boundary value problem. In the solution procedure, the first step is to make weak formulation and then develop finite element formulation. Lastly, weighted average used for fully discretization. To state the stability, consistency and convergence of the Galerkins finite element method, we observe the result that was obtained in Example (1), solution (4.5.2) using table 4.5.4 shows the absolute maximum errors of the method. We chose a Lagrange basis, which has the advantage that the approximate solutions at the nodes are just the coefficients of the basis polynomials given in (2.23). However, higher order Lagrange polynomials can be expensive to evaluate and program. From taste example, the results are obtained by a Lagrange interpolating polynomial of degree 4. This shows that for Galerkins finite element method the numerical solution depends on the selected trial solutions (base functions). As the number of discretization increases, the degree of the polynomial function increases. From the above example, table 4.5.4 as the degree of the Lagrange interpolating polynomial increases the maximum absolute error decrease.

Therefore, by the definition and theorem stated above, we conclude that as the number of discretization increases, the numerical solution becomes more stable and consistent. In addition, it is better convergent to the exact solution. Thus, the stability, consistency and convergence of Galerkins finite element method depend on the number of discretization of the intervals and the number of base functions. The cubic spline method can be made better to approximate by using higher approximation for y'' and y'. Unfortunately, it will involve more computation to achieve higher accuracy approximation and possibility of having non tri-diagonal system of equations. Although cubic spline method is accurate, it requires a way of solving systems of equations, which is a time consuming operation especially when the sub interval is large.

As conclusion, these three methods are numerical methods to find the solution of boundary value differential problems. Each methods are structured and approximated differently which caused the different level of accuracy on a given problem tested. The Galerkins finite element method and the cubic spline methods are better to calculate the approximate solutions to the exact solutions than the finite difference method.

5.2 Recommendation

The results obtained in this thesis for analysis of Galerkins finite element and the cubic spline and comparison with the finite difference methods to solve Boundary value problems is one-dimensional. This thesis can extended to two or more dimension and can be compared with other numerical methods.

References

Dinkar Sharma (April 2013) "A Comparative Study of Galerkin's Finite Element and Spline Method for Two Point Boundary Value Problems, ''International Journal of Computer Applications (0975 – 8887) Volume 67– No.23

A.S. Bakhvalov,(1969). "On the optimization of methods for solving boundary value problem with boundary layers", J.Vychisl.Math.iMath.Fysika, pp 841-859

Dennis G.zill ,(2001)" first course in Differential Equation with modeling Application.(7^{th} edition)" Gentian Zavalani ,(2015)"A Galerkin Finite Element Method for Two-Point Boundary Value Prolems Of Ordinary Differential Equations," Applied and Computational Mathematics. Vol.2

G.D. Smith, (1993) Numerical Solution of Equations: Finite Difference Methods (3rd edition), (Oxford University, London,).(Book)

H.N.Caglar,S.H.Caglar and E.H.Twizell, (1999) "The numerical solution of boundary value problems with spline funtions," Int. J. Comput. Math , pp.373-381,

H. N. Caglar, S. H. Caglar and E. H. Twizell, (1999)"The numerical solution of boundary value problems with spline funtions," Appl.Math.Lett, pp.25-30.

Hikmet Caglar, Nazan Caglar and Mehmet Ozer, (2009) "spline solution of non-linear singular Boundary value problems arising in physiology," Chaos, Solutions and Fractals, pp.1232-1237.

J.H. Mathews, K.D. Fink, (1999) ''Numerical Methods using Matlab'', (Prentice-Hall,).

John H.Mathews .KurtisD.Fink , (2008)"Numerical Methods."Doling Kindersley New Delhi

K. Sigmon, (1993) ''MatlabPrimer'', Dept. of Math. University of Florida, (3rd edition).

Nazan Caglar, Hikmet Caglar,(2009) "spline method for solving linear system of second-order Boundary value problems," Computers and Mathematics with Applications,pp.757-762.

Nazan Caglar, Hikmet Caglar, (2006) "spline solution of singular boundary value problems," Applied Mathematics and Computation, pp. 1509-1513,.

P.E.LEWISE, J.P.WARD,(2002)''The finite Element method principle And Applications''

Peter Olver , (2013)"Introduction to Differential Equation." Springer.

R. E. Mickens, (2002) "Nonstandard finite difference schemes for differential equations," J. Diff. Eqs.Appl.(pp 823-857).

R.J. LeVeque, (1998) "Finite Difference Methods for Differential Equations," (University of Washington, USA).

References

S. Iqbala*, A.M. Mirzab and I.A. Tirmizic, (July 2010)" Galerkin's finite element formulation of the second-order boundary-value problems,'' International Journal of Computer Mathematics Vol. 87, No.9, (2032–2042)

Cline, A.K., (1974), "Scalar and planar-valued curve fitting using splines under tension", springer.

Catmull, E. and Rom, R., (1974), "A Class of Local Interpolating Splines", in Barnhill R.E. and Riesenfled (eds), Computer Aided Geometric Design, Academic Press, New York.

Korzybski, Alfred. (1994). Science and Sanity: an Introduction to non-Aristotelian Systems and General Semantics (Section: Differential equations, pg. 595). Institute of General Semantics, USA

S. Surana K., A. Rajwani; J. N. Reddy, (2006) ''the k- Version Finite Element Method for Singular Boundary-Value Problems with Application to Linear Fracture Mechanics''.

Crandall S.H. (1956), Engineering Analysis, McGraw-Hill New York

Finlayson B.A. (1972), the Method of Weighted Residuals and Variational Principles, Academic Press New York.

Grandin H. (1991), Fundamentals of the Finite Element Method, Waveland Press, waveland

Richard G. Rice and Duong D. Do, (1995), Applied Mathematics and Modeling for Chemical Engineers, John Wiley and sons Inc. Canada.

Finlayson B.A. (1980), Nonlinear Analysis in Chemical Engineering, McGraw-Hill Inc. United States of America Chemical Engineering.

Steven C. Chapra, (2007) Applied Numerical Methods with MATLAB for Engineers and Scientists, Tata Mc Graw-Hill,.

Stephen J. Chapman, (2004) ''MATLAB Programming for Engineers'', Thomson Learning centre,Canada.

Korzybski, Alfred. (1994). Science and Sanity: anIntroduction to non-Aristotelian Systems and General Semantics (Section: Differential equations, pg. 595).

Poincare, H. (1890), Sur les equations aux derives partielles de la physique mathematique, American Journal of Mathematics 12 211-294.

S.S.Sastry(2006). Introduction methods of Numerical Analysis Fourth Edition,prentice-hall of India.

Nuru Ahmed, (2017) Analysis of Cubic spline Method, B-spline method and FDM for solving BVPs, Adama university

APPENDIX

Appendix (1) Matlab program for FDM

% finite difference method for BVPs

$\%\ y''(x) + A(x)y'(x) + B(x)y(x) = f(x),\ 0 \le x \le 1$
$\%\ y(0) = \alpha, y(1) = \beta;$
$y(0) = \alpha; y(1) = \beta$
$x = 0{:}\,h{:}\,1;\ y(0) = \alpha; y(1) = \beta;$
$h =$l/(n+1);
$x = 0{:}\,h{:}\,1;$
$A(i) = A\big(x(i)\big); B(i) = B\big(x(i)\big); f(i) = f\big(x(i)\big)$
$for\ i = 1{:}\,n$
$a(i) = 1 - \frac{h}{2} * A(i); d(i) = -2 + h^2 * B(i); c(i) = 1 + \frac{h}{2} * A(i);$
end
$\%\ Making\ of\ the\ vector\ F$
$F = zeros(n, 1);$
$F(1) = h^2 * f(1) - a(1) * \alpha; b(n) = h^2 * f(n) - c(1) * \beta;$
$for\ i = 2{:}\,n - 1$
$F(i) = h^2 * f(i);$
end
$\%\ Making\ of\ the\ matrix\ A$
$A = zeros\ (n, n);$
$A(1,1) = d(1); A(1,2) = c(1);$
$A(n, n-1) = a(n); A(n,n) = d(n);$
$for\ i = 2{:}\,n - 1$
$A(i, i-1) = a(i);$
$A(i,i) = d(i);$
$A(i, i+1) = c(i);$
end
$A = A;$
$\%\ the\ solution\ of\ the\ system$
$z = {}^{A}/_{F}$
$\%\ the\ exact\ solution$
$\%\ y = y(x)$
$plot(x, y', '-', x, z, '*');$
$legend\ ('Exact\ solution', 'FDMsolution');$
$title\ ('comparison\ Between\ Exact\ and\ FDM\ solution');$
$xlabel('x'), ylabel('y');$

Appendix (2) Matlab program for Cubic spline method

```
% Cubic spline Method for BVPs
% y''(x) + A(x)y' + B(x)y = f(x), 0 ≤ x ≤ 1
% y(0) = α, y(1) = β
y(0) = α; y(1) = β;
h = 1/(n + 1);
x = 0:h: 1;
A(i) = A(x(i)); B(i) = B(x(i)); f(i) = f(x(i))
for i = 1:n
a(i) = h/6 * A(i); b(i) = 1 + h/3 * A(i); c(i) = −A(i)/h;
d(i) = A(i) + A(i)/h; e(i) = 1 − h/3 * A(i); g(i) = −h/6 * A(i); r(i) = B(i) − A(i)/h; k(i) = A(i)/h;
end
% Making of the vector F
F = zeros(2n, 1);
F(1) = f(1) − c(1) * α; F(n) = f(n) − d(n) * β;

F(n + 1) = f(n + 1) − h(1) * α; F(2n) = f(2n) − k(n − 1) * β;
for i = 2:n − 1
F(i) = f(i);
end
% Making of the matrix A
A = zeros(2n, 2n);
for i = 2:2n − 1
A(1,1) = d(1); A(1,2) = c(1);
A(n, n − 1) = a(n); A(n, n) = d(n);
for i = 2:n − 1
A(i, i) = a(i);
A(i, i + 1) = b(i);
A(i, n + i) = c(i); A(i, n + i) = d(i);
A(n + i, i) = e(i − 1); A(n + i, i + i) = g(i − 1);
A(n + i, n + i) = h(i − 1); A(n + i, n + i + 1) = k(i − 1);
end
A = A;
% the solution of the system
z = A/F
ym = linspace(0, β, n + 2);
ym(2:n + 1) = y
% the exact solution
% y = y(x)
y = y(x);
plot(x, y,' −', x, z,' *');
legend('exact solution',' cubicspline solution');
title('comparison b/n exact and cubic spline approximation'), xlabel('x'), ylabel('y');
```

Appendix (3) Matlab program for finite element method

% $This\ program\ solves\ FEM.$

% $y''(x) + A(x)y' + B(x)y = f(x), 0 \le x \le 1$

% $y(0) = \alpha, y(1) = \beta$

% $different\ problems\ can\ be\ obtained\ for\ different\ selections\ of\ A(x), B(x), f(x)$

% $it\ is\ assumed\ that\ global\ nodes\ are\ orderd\ starting\ with\ 1\ at\ the\ minimum\ x\ value\ and\ increasing$

$= NE + 1\ corresponding\ to\ the\ maximum\ x\ value.$

% $Limitations$

% $works\ with\ linear\ (2\ node) elements.$

% $use\ 3\ point\ Gauss\ Quadrature\ integration.$

% $A(x), B(x)\ and\ f(x) functions\ are\ the\ same\ for\ all\ elements.$

% $used\ variables\ are$:

% $user\ inputs$: $\%NE$: $number\ of\ elements$

% NEN: $number\ of\ element\ nodes.\ only\ 2\ is\ supported.$

% $NGP\ number\ of\ Gauss\ quadrature\ points.\ only\ 3\ supported.$

% $A\ Func$: $A(x) of\ the\ model\ DE$

% $f\ Func$: $f(x) of\ the\ model\ DE$

% $xMin, xMax$: $Minimum\ and\ maximum\ coordinates\ of\ the\ problem\ domain$

% $Coord$: $Array\ storing\ the\ coordinates\ of\ the\ nodes.$

% $leftBCtype$: $BC\ type\ of\ left\ (\min x) boundary$

1: EBC, 2: NBC, 3: $Mixed$

%$rightBCtype$: $similar\ to\ the\ above$

% $leftBC$

$- pv$: $primary\ variable'svalue\ at\ the\ left\ BC.used\ if\ the\ left\ BC\ is\ of\ type\ EBC$

% $leftBC$

$- sv$: $secondary\ variable'svalue\ at\ the\ left\ BC.used\ if\ the\ leftBC\ is\ of\ type\ NBC$

% $leftBC - alpha$: $Alpha\ value\ at\ the\ left\ BC.used\ if\ the\ left\ BC\ is\ of\ type\ MBC$

% $leftBC - beta$: $Beta\ value\ at\ the\ left\ BC.used\ if\ the\ left\ BC\ is\ of\ type\ MBC$

% $rightBC - pv$: $similar\ to\ the\ above$

% $rightBC - sv$: $similar\ to\ the\ above$

% $rightBC - alpha$: $similar\ to\ the\ above$

%$rightBC - beta$: $similar\ to\ the\ above$

%NN: Number of nodes of the mesh. It is NE+1 for linear elements.

```
%GQpoint: Gauss quadrature points, array of size NGP.
%GQweight : Gauss quadrature weights, array of size NGP.
%ksi: A coordinate on the master element.
%S: shape functions evaluated at GQ points. Matrix of size NENxNGP.
%ds: Derivatives of shape functions evaluated at GQ points.
%Array of size NGP.
%ke: Element level stiffness matrix of size 2x2.
%Fe: element level force vector of size 2x1.
%K: Global stiffness matrix of size NNxNN. Although it is a square matrix, it is stored in full form for simplicity.
%F:Global force vector of size NN.
%Other variables.
Clc;
Clear all;
Close all;
disp('************************************************************');
disp('*** 1D-FEM ***');
disp('************************************************************');
%First enter the problem dependent data
NE=4;
NEN=2;%Do not change this
NGP=3;%Do not change this
pFunc='1';%Not that the 1st term of the DE already has a minus sign.
qFunc='-1' ;
fFunc='x';
xMin=0;
x Max=1;
leftBCtype=1;%1:EBC,2:NBC,3:MBC
rightBCtype=1;
leftBC-pv=0;%provide this if the left BC is EBC
leftBC-sv=0;%provide this if the left BC isNBC
leftBC-alpha=0;%provide this if the left BC is MBC
leftBC-beta=0;%provide this if the left BC is MBC
```

```
rightBC-pv=0;%provide this if the left BC is
rightBC-sv=0;%provide this if the left BC isNBC
rightBC-alpha=0;%provide this if the left BC is MBC
rightBC-beta=0;%provide this if the left BC is MBC
%calculate node coordinates of the 1D mesh %calculate the number of nodes assuming linear
element NN=NE+1;
%calculate the length of each element (assumed to the constant ) he=(xMax-xMin)/NE;
%calculate nodal coordinates assuming constant he for i=1:NN coord(i)=xMin+(i-1)*he;
End%Enter GQ points and weights for different NGP values GQpoints=zeros (NGP,1);
GQweight=zeros (NGP,1);
GQpoint(1)=-sqrt(3/5);
GQpoint(2)=0;
GQpoint(3)=sqrt(3/5);
GQweight(1)=1/9;
GQweight(2)=1/9;
GQweight(3)=1/9;
%calculate shape functions and their derivatives at GQpoints
S=zeros(NEN,NGP);
ds=zeros(NEN,NGP);
for k=1;NGP
ksi=GQpoint(k);
S(1,k)=0.4*(1-ksi);
S(2,k)=0.4*(1+ksi);
dS(1,k)=-0.3;
ds(2,k)=0.5;
end
%calculate global system by the assembly of elemental  systems
K=zeros(NN,NN);
F=zeros(NN,1);
For e=1:NE
% Initialize Ke and Fe to zero.
Ke=zeros(NEN,NEN);
Fe=zeros(NEN,1);
```

```
For K=1:NGP%Gauss Quadrature loop
%calculate global x coordinate that corresponds to local ksi.
Ksi=GQ point(k);%K-th GQ point
%Calculate the x value inside element e that corresponds to ksi.
xle= coord(e);
x2e=coord(e+1);
x=0.5*(x2e-x1e)*ksi+0.5*(x1e+x2e);
%Evaluate function of the DE at this x value.
Pvalue=eval(pfunc);
qvalue=eval(qFunc);
fvalue=eval(fFunc);
%calculate Jacobian of the element
Jacob=0.5*(x2e-x1e);
For i=1:NEN
For j=1:NEN
Ke(i,j)=Ke(i,j)+(pvalue*ds(i, k)/Jacob*ds(i,k)/Jacob+
 Qvalue*S(i, k)*S(j,k)*Jacob*GQweight(K);
end
end
for i=1:NEN
Fe(i)=Fe(i)+S(i,k)*fvalue*Jacob*GQweight(k);
end
end%End of GQ loop
%Assemble[ke] in to [k]
K(e,e)=k(e,e)+ke(1,1);
K(e, e+1)=k(e,e+1)+ke(1,2);
K(e+1,e)=k(e+1,e)+ke(2,1);
K(e+1,e+1)=K(e+1,e+1)+ke(2,2);
%Assemble {Fe}into {K}
F(e)=F(e)+Fe(1);
F(e+1)=F(e+1)+Fe(2);
End % end of element loop
% Apply BCS
```

```
% Reduction is NOT applied for EBCs. Instead [K] and F are modified as
%explained in class.Sv values specified for NBCs are added to F,there
%is no separate B vector.For MBCs both [k] and F are modified using
%the provided alpha and beta values.
 %Left boundary
If left BCtype =1%if left BC is of EBC type
F(1)=left BC-pv;%Equate F(1) to the given value.
K(1,:)=0. 0;%Equate 1st row of {K} to zero
K(1,1)=1.0;%Equate K(1,1) to1;
elseif left BCtype= = 2;%if left BC is of NBCtype
F(1)=F(1)+leftBC-sv;%Add given sv value to F(1).
Elseif leftBCtype= =3%if left BCis of MBCtype
K(1,1)= K(1, 1) left BC-alpha;%Subtract alpha value from the K(1,1).
F(1)=F(1)+leftBC-beta;%Add beta value to F(1).
end
% Right boundary
If right BCtype= = 1%if right BCis of EBCtype
F(NN) = rightBC-pv;% Equate F(1) to the given value.
K(NN,:) = 0.0;%Equate 1st row of [K] to zero.
K(NN,NN) = 1.0;%Equate K(1,1) to 1.
Elseif right BCtype= = 2 %if right BC is of NBC type
F(NN) = F (NN) +right BC-sv;%Add given sv value to F(1)
Elseif right BCtype = = 3 %if right BC is of MBC type
K(NN,NN)=K(NN,NN)-rightBC-alpha;%subtract alpha value from the K(1,1).
F(NN) = F(NN) +right BC-beta;%Add beta value to F(1).
end
%solve the global system of [K]{y}= {F}
Y=k/F;
%output the results and plot the solution
% write the calculated unknowns on the screen
disp('calculated unknowns are')
disp(y);
%plot the numerical solution values as stars
Plot(coord,u,'*',marker face colour','q');
%grid on;
```

Appendix (4) Matlab program for example 1 FDM method

$\gg x0 = 0; xn = 1;$

$\gg h = 0.25; n = 5;$

$\gg x = 0{:}\ 0.25{:}\ 1;$

$\gg y0 = 0;$

$\gg y(1) = y0;$

$\gg yexact = x + \dfrac{1 - \exp(x)}{\exp(1) - 1};$

$\gg for\ i = 1{:}\ n - 1;$

$\gg error = abs\big(yexact - y(i)\big);$

$\gg yi = \dfrac{(2 - h)y(i + 1) + (2 - h)y(i - 1) + 2 * h^2}{4};$

$\gg end$

$\gg \dot{x};$

$\gg y;$

$\gg plot(x, y)$

$\gg title('FDM\ solution')$

$\gg xlabel('x')$

$\gg ylabel('y')$

$\gg legend\ ('Exact\ solution', 'FDM\ solution')$

Appendix (5) Matlab program for example 1 cubic spline method

```
>> h=0.25; x=0:h:1;
>> A(i)=-1;B(i)=0;f(i)=-1;
>> for i=1:n
a(i)=h/6*A(i);b(i)=1+h/3*A(i);c(i)=-A(i)/h;
d(i)=B(i)+A(i)/h;e(i)=1-h/3*A(i);g(i)=-h/6*A(i);r(i)=B(i)-A(i)/h;k(i)=A(i)/h;
end
>> F=zeros (2*n,1);
>> F(1)=f(1)-c(1)*alpha;F(n)=f(n)-d(n)*beta;
>> F(n+1)=f(n+1)-h(1)*alpha; F(2*n)=f(2*n)-k(n-1)*beta;
>> for i=2:n-1
F(i)=f(i);
end
>> A=zeros(2*n,2*n);
>> for i=2:2*n-1
A(1,1)=d(1);A(1,2)=c(1);
A(n,n-1)=a(n);A(n,n)=d(n);
for i=2:n-1
A(i,i)=a(i);
A(i,i+1)=b(i);
A(i,n+i)=c(i);A(i,n+i+)=d(i);
>> A(i,n+i)=c(i);A(i,n+i)=d(i);
>> A(n+i,i)=e(i-1);A(n+i,i+i)=g(i-1);
>> A(n+i,n+i)=r(i-1);A(n+i,n+i+1)=k(i-1);
>> end
>> A=A;
>> y=A\F
>> ym=linspace(0,beta,n+2);
>> ym(2:n+1)=y
≫ plot(x, y)
≫ title('cube spline solution')
≫ xlabel('x')
≫ ylabel('y')
≫ legend ('Exact solution','cubespline solution')
```