NUMERICAL SOLUTION OF LINEAR SECOND ORDER ORDINARY DIFFERENTIAL EQUATIONS WITH MIXED BOUNDARY CONDITIONS BY GALERKIN METHOD

BY:

AKALU ABRIHAM

ADVISORS: ALEMAYEHU SHIFERAW (PhD)

GENANEW GOFE (PhD)

AYANA DERESSA (M.Sc.)

A THESIS SUBMITED TO THE DEPARTMENT OF MATHEMATICS IN PARTIAL FULLFILMENT FOR THE REQUIRMENTS OF THE DEGREE OF MASTERS SCIENCE IN MATHEMATICS

JIMMA ETHIOPIA

JUNE, 2014

DECLARATION

I, undersigned declare that this thesis entitled **"Numerical Solution of Linear Second Order Ordinary Differential Equations with Mixed Boundary Conditions by Galerkin Method"** is my original work and it has not been submitted to any institution elsewhere for the award of any academic degree or the like, where other sources of information have been used, and they have been acknowledged.

Name: Akalu Abriham

Signature: ……

Date: ……

The work has been done under the supervision of

The approval of Advisors Name:

Name: Alemayehu Shiferaw (PhD)

Signature: ……..

Date: ……..

Name**:** Genanew Gofe (PhD)

Signature: ……

Date: …….

Acknowledgment

Frist of all I would like to thank my omnipresent GOD for his goodwill and giving me life to do this research. Next to this, my deepest and sincere gratitude goes to my advisors Dr. Alemayehu Shiferaw, Dr. Genanew Gofe and my co-adviser Mr. Ayana Deressa for their important comments, effective cooperation and great care that enabled me to bring the study to its present form.

I am also indebted to extend my heartfelt thanks and appreciation to my parents (this is fully their prayer, wish and extensive effort to bring me to this stage) and all postgraduate students of 2013/2014 in our department for their support in sharing ideas and materials.

Last but not least, I would like to acknowledge Department of mathematics for giving me this delightful chance and for the supports rendered to me while conducting this research.

Akalu Abriham

Table of Contents

Content Pages

List of table **page**

List of figures **page**

Lists of Appendixes page

Acronym

IVP --Initial Value Problem

BVP--Boundary Value Problem

FEM--Finite Element Method

FDM--Finite Difference Methods

BC -- Boundary Condition

IC -- Initial Conditions

Abstract

The purpose of this study was to find numerical solutions of linear second order ordinary differential equations with mixed boundary condition by Galerkin method using Chebyshev polynomial as a trial function. The Galerkin method was applied after converting the given linear second order ordinary differential equation with mixed boundary condition into equivalent boundary value problem by considering a valid assumption for the independent variable and also converting mixed boundary condition in to Neumann type. The resulting system of equation was solved by direct method.

In order to check to what extent our method approximates the exact solution, a test example with known exact solution was solved and compared with the exact solution graphically as well as numerically.

CHAPTER ONE

1. Introduction

1.1. Background

The goal of numerical analysis is to find the approximate numerical solution to some real physical problems by using different numerical techniques, especially when analytical solutions are not available or very difficult to obtain. Since most of mathematical models of physical phenomena are expressed in terms of ordinary differential equations, and these equations due to their nature and further applications to use computers, it needs to establish appropriate numerical methods corresponding to the type of the differential equation and conditions that govern the mathematical model of the physical phenomena. The conditions may be specified as an initial Value (IVP) or at the boundaries of the system, Boundary Value (BVP) [1].

Many problems in engineering and science can be formulated as two-point BVPs, like mechanical vibration analysis, vibration of spring, electric circuit analysis and many others. This shows that the numerical methods used to approximate the solutions of two-point boundary value problems play a vital role in all branches of sciences and engineering [2].

Among different numerical methods used to approximate two-point boundary value problems in terms of differential equations are shooting method, finite difference methods, finite element methods (FEM), Variational methods (Weighted residual methods, Ritz method) and others have been used to solve the two-point boundary value problems [3]. Both in FEM and Variational methods the main attempts were to look an approximation solution in the form of a linear combination of suitable approximation function and undetermined coefficients [4].

For a vector space of functions V, if S=$\{\phi_i(x)\}_{i=1}^{\infty}$ be basis of V, a set of linearly independent functions, any function $f(x) \in V$ could be uniquely written as a linear combination of the basis as:

$$f(x) = \sum_{j=1}^{\infty} c_j \phi_j(x) \qquad (1.1)$$

The weighted residual methods use a finite number of linearly independent functions $\{\phi_i(x)\}_{i=1}^{n}$ as trial function.

Suppose that we seek an approximation of the solution of the differential equation,

$D(u) = L(u(x)) + f(x) = 0$, on the boundary $B(u) = [a,b]$ in the form:

$$u(x) \approx U_N(x) = \sum_{j=1}^{N} c_j \phi_j(x) + \phi_0(x) \tag{1.2}$$

where $U_N(x)$ is the approximate solution, $u(x)$ is the exact solution " L " is a differential operator, " f " is a given function, $\phi_j(x)'s$ are finite number of basis functions and c_j unknown coefficients for $j = 1,2,...,N$.

The residual $R(x,c_j)$ is defined as: $R(x,c_j) = D(U_N(x)) - (L(U_N(x)) + f(x))$ .If we determine c_j by requiring R to vanish in a "weighted-residual" sense:

$$\int_a^b w_i(x) R(x,c_j)dx = 0 \quad (i = 1,2,...,N) \tag{1.3}$$

where $w_i(x)$ are a set of linearly independent functions, called weight functions, which in general can be different from the approximation functions $\phi_j(x)$, this method is known as the weighted-residual method. If $\phi_j(x) = w_i(x)$, equation (1.3) is known as the Galerkin method. Thus Galerkin method is one of the weighted residual methods in which the approximation function is the same as the weight function and hence it is also used to find the approximate solution of two-point boundary value problems [4].

The Galerkin method was invented in 1915 by Russian mathematician Boris Grigoryevich Galerkin and the origin of the method is generally associated with a paper published by Galerkin in 1915 on the elastic equilibrium of rods and thin plates. He published his finite element method in 1915. The use of Galerkin method increased rapidly during the 1950's when it was used for analyzing dynamics of aeronautical structures [1]. Furthermore the Galerkin method approximates the solution to the BVP by a linear combination of basis functions determined by requiring that the residual be orthogonal to each of the homogeneous basis functions, i.e., those

that vanish on the boundary, and that the boundary conditions are satisfied. The Galerkin method can be used to approximate the solution to ordinary differential equations, partial differential equations and integral equations [5].

Many authors have been used the Galerkin method to find approximate solution of ordinary differential equations with boundary condition. Among this, a spline solution of two point boundary value problems introduced in [6], a method for solutions of nonlinear second order multi-point boundary value problems produced in [7], in [8] linear and non-linear differential equations were solved numerically by Galerkin method using a Bernstein polynomials basis, in [9] a numerical method is established to solved second order linear boundary value problems with Neumann and Cauchy types boundary conditions using Hermite polynomials on [0 1].

In this study we extend the result of M. M. Rahman, M.A. Hossen, M. Nuru Islam and Md. Shajib Ali by extending the domain and solving for different types of boundary conditions to find an approximate solution of second order ordinary differential equation with mixed boundary condition by Galerkin method.

1.2. Statement of the problem

Mathematical models in terms of ordinary differential equations with boundary conditions are common in many fields of science and engineering such as mechanical engineering, electrodynamics, physics and others. The increasing of desire for the numerical solutions to mathematical problems, which are more difficult or impossible to solve explicitly, has become the present day scientific research. Thus, this shows that the importance and application of numerical methods to solve problems in real life. The numerical method used to find approximate solution of boundary value problems has an impressive importance due to its wide applications in scientific and engineering researchers.

So, among methods used to find approximate solution of ordinary differential equations with boundary condition the Galerkin method is one of the weighted residual methods that approximate the solution of the BVP by a linear combination of basis functions determined by requiring that the residual is orthogonal to each of the homogeneous basis functions.

Therefore, the presented study answered the following questions:

- How can we approximate solution of linear second order differential equations with mixed boundary condition?
- To what extent our method approximates the exact solution?
- How can we extend the result obtained by M. M. Rahman.et.al in [9] into mixed boundary case?
- How to solve examples by using our method with the help of MATLAB software?

1.3. Objective of the study

1.3.1. General objective

The main objective of this study was to obtain numerical solution for linear second order ordinary differential equation with mixed boundary condition by Galerkin method.

1.3.2. Specific objective

- To determine the approximate solution of linear second order ordinary differential equations with mixed boundary condition.
- To show how the graph of the approximate solution approaches to the graph of the exact solution.

- To show how to extend the method by M.M.Rahman.et. al. [9] in to mixed BC.
- To solve examples using the presented method by the help of MATLAB software.

1.4. Significance of the study

Since the purpose of this study is to solve ODEs with mixed boundary condition, the result of this study will be used to find the approximate solution of ODEs with mixed boundary conditions using Galerkin method which avoids the restriction to only Neumann and Cauchy boundary condition in [9]. The results obtained in this paper would contribute to research activities in this area. Further, collaboration in this research project would be useful for the graduate program of the department and enhances the research skill and scientific communication of the researcher.

1.5. Delimitation of the study

This study is delimited to numerical solution. The study was also limited to linear second-order ordinary differential equations of the form (4.2) using Galerkin Method by taking Chebyshev polynomial as trial function.

CHAPTER TWO

2. Literature review

In the literature of numerical analysis solving ordinary differential equations with boundary conditions, many authors have attempted to obtain higher accuracy rapidly by using a numerous methods.

Among the methods available in the literature concerning their numerical solutions, a spline solution of two point boundary value problems introduced. [6], a method for solutions of nonlinear second order multi-point boundary value problems produced [7], a linear and non-linear differential equations were solved numerically by Galerkin method by a Bernstein polynomials basis [8], a numerical method is established to solved second order linear boundary value problems with Neumann and Cauchy types boundary conditions using Hermite polynomials [9]. That is by converting the given ODE with boundary condition on any arbitrary interval [a, b] into equivalent BVP in [0, 1]. A parametric cubic spline solution of two point boundary value problems were obtained [10], a second-order Neumann boundary value problem with singular nonlinearity for exact three positive solutions were solved [11], a Numerical solution of a singular boundary-value problem in non-Newtonian fluid mechanics were established [12], a Fourth Order Boundary Value Problems by Galerkin Method with Cubic B-splines were solved by considering different cases on the boundary condition [13] and a special successive approximations method for solving boundary value problems including ordinary differential equations were proposed.[14]

As clearly explained above, most of the authors have attempted to obtain simple and accurate methods by using different polynomials as a trail function such that the polynomial must satisfy both the differential equation and the boundary conditions. This study mainly depends on the results of [9] in which the trail function was replaced by Chebyshev polynomials and the boundary condition is mixed type. The objective of using Chebyshev polynomial is to extend the domain from [0, 1] in to [-1, 1].

CHAPTER THREE

3. METHODOLOGY

The following methods and materials were used to carry out this study.

3.1. Study Design, site and period

This study used documentary review article design assisted by MATLAB. The study was conducted in Jimma University, department of mathematics from November 2013 to June 2014**.**

3.2. Study area

The study focused on numerical solution of linear second order ordinary differential equations with mixed boundary and Neumann boundary conditions by using Galerkin method.

3.3. Source of information

In this study secondary data were used as a source of information. So, the available sources of information for this study were books, journals and different related studies from internet services. In addition to this, a one week workshop on MATLAB software were designed and conducted.

3.4. Instrumentation and Administration

Important materials and data for the study were collected by the researcher using documentary analysis as an instrument. MATLAB program were coded and used to solve examples in order to compare the results obtained by our method with the exact solution.

3.5. Study procedures

In order to achieve the objectives of this study we used the standard technique used in [9] to extend the boundary conditions into a mixed type boundary condition. In addition to the standard technique used in [9], the study used the following mathematical steps.

Step 1: We check whether the boundary condition is mixed type or Neumann type.

Step 2: If the boundary condition is Neumann type go to step 4

Step 3: If the boundary condition is mixed type convert the boundary condition in to Neumann

type by considering a guess for $y'(a)$ by the concept of shooting method assuming that the given value for $y'(b)$ is satisfied.

Step 4: Convert the given linear second order ordinary differential equation with Neumann boundary condition, whose independent variable is within the interval [a, b]; into equivalent BVP with interval of independent variable [-1, 1].

Step 5: Use the technique of Galerkin method to find an approximate solution of the ODE using constant parameters $c_i's$ and Chebyshev polynomial as a trial function.

Step 6: Integrating the first term by parts with in the limit of integration -1 to 1, we get an integral equation.

Step 7: The integral equation obtained in step 6 contains unknowns $y'(-1)$, and $y'(1)$, for this we substitute values from the Neumann boundary condition with interval of independent variable [-1, 1].

Step 8: Upon substitution of these values into the integral equation obtained in step 7 and using the sequence of Chebyshev polynomial, we have a system of equation.

Step 9: The unknown parameters $c_i's$ used in step 5 are determined by solving the system of equation using direct method.

Step 10: Substituting these values of the parameters, we get the approximate solution of ODE satisfying the Neumann as well as mixed boundary conditions.

After all these steps, MATLAB software was used to visualize the graphs of the exact solution and the approximate solution in the same plane.

Finally, from the results of the study possible conclusion and recommendation were given.

3.6. Ethical issues

To be legal for collecting all the above materials it is important to have a permission letter. So the researcher has got a letter of permission from the department of mathematics and the letter was used to explain the aim of collecting of those materials.

CHAPTER FOUR

4. RESULT AND DISCUSSION

4.1. Preliminaries

A boundary value problem for a given differential equation consists of finding a solution of the given differential equation subject to a given set of boundary conditions. A boundary condition is a prescription some combinations of values of the unknown solution and its derivatives at more than one point.

Definition 4.1: Let $I = (a, b) \subseteq \mathbb{R}$ be an interval. Let P , Q , and R: $(\text{a, b}) \to \mathbb{R}$ be continuous functions. Consider the linear second order differential equation given by:

$$y'' + P(x)\, y' + Q(x)\, y = f(x) \qquad (4.1)$$

Corresponding to ODE (4.1), there are four important kinds of (linear) boundary conditions. They are given by:

i) Dirichlet or first kind : $y(a) = \eta_1,\ y(b) = \eta_2$

ii) Cauchy or first kind: $y(a) = 0,\ y'(b) = \eta_2$

iii) Neumann or Second kind : $y'(a) = \eta_1,\ y'(b) = \eta_2$

iv) Robin or third or mixed kind : $\alpha_1 y(a) + \alpha_2 y'(a) = \eta_1,\ \beta_1 y(b) + \beta_2 y'(b) = \eta_2$

v) Periodic:- $y(a) = y(b),\ y'(a) = y'(b)$, where $\eta_1, \eta_2, \alpha_1, \alpha_2, \beta_1$ and β_2 are all real constants.

Unlike IVPs, In fact, depending on the specific combination of the ODE and the boundary conditions, the BVP may have:

(i) No solutions,

(ii) One solution,

(iii) Finite number of solutions, or

(iv) Infinitely many solutions.

Possibility (iii) can take place only for nonlinear BVPs, while the other three possibilities can take place for both linear and nonlinear BVPs. Because of this, programs for solving BVP require users to provide a guess for the solution desired.

The boundary-value problem is said to be homogeneous if both the differential equation and the boundary conditions are homogeneous (i.e. $f(x) = 0 = \eta_i = \beta_i, i = 1, 2$). Otherwise the problem is nonhomogeneous.

Definition 4.2: A mixed boundary condition for a differential equation defines a boundary value problem in which the solution of the given equation is required to satisfy different boundary conditions on disjoint parts of the boundary of the domain where the condition is stated. Precisely, in a mixed boundary value problem, the solution is required to satisfy a Dirichlet or a Neumann boundary condition in a mutually exclusive way on disjoint parts of the boundary.

4.1.1. Chebyshev polynomial

The polynomials whose properties and applications are discussed in this paper were 'discovered' almost a century ago by the Russian mathematician Chebyshev. Their importance for practical computation, however, was rediscovered some thirty years ago by C. Lanczos. The coming of the digital computer gave further emphasis to this development, and at the present time the research literature of numerical mathematics abounds with papers on applications of Chebyshev polynomials and the theory and practice of Chebyshev approximation [15].

Definition 4.3. The Chebyshev polynomial is a function defined using trigonometric functions $\cos\theta$ and $\sin\theta$ for $x \in [-1,1]$. There are two types of Chebyshev polynomial.

1. Chebyshev polynomial of first kind with degree n for $x \in [-1, 1]$ defined as:

$T_n(x) = \cos n\theta$, such that $\cos\theta = x$, for $-1 \le x \le 1$ and $n \ge 0$

Thus we have $T_n(x) = \cos(n\cos^{-1} x)$,

$\Rightarrow T_0(x) = \cos(0) = 1 \; and \; T_1(x) = \cos(\cos^{-1} x) = x$

From the trigonometric identity,

$\cos(n+l)\theta + \cos(n-l)\theta = 2\cos n\theta\cos\theta - \sin(n\theta)\sin(\theta) + \sin(n\theta)\sin(\theta)$

$$\Rightarrow \cos(n+l)\theta + \cos(n-l)\theta \;=\; 2\ \cos\ n\theta\cos\theta$$

$$\Rightarrow T_{n+1}(x) \;=\; 2xT_n(x) - T_{n-1}(x).$$

Thus using the recursive relation above for n=1, 2… we have a series of Chebyshev polynomial

$T_1(x) = x$

$T_2(x) = 2x^2 - 1$

$T_3(x) = 4x^3 - 3x$

$T_4(x) = 8x^4 - 8x^2 + 1$

$T_5(x) = 16x^5 - 20x^3 + 5x$ etc.

We see that the coefficient of x^n in $T_n(x)$ is 2^{n-1}.

2. Chebyshev polynomial of second kind, $U_n(x)$ of degree n for $x \in [-1,1]$ defined as:

$$U_n(x) \;=\; \frac{\sin(n+1)\theta}{\sin\theta} \quad \text{, for } \cos\theta = x \;\Rightarrow\; \theta = \cos^{-1}(x)$$

$$\Rightarrow U_r(x) \;= \frac{\sin r\theta\cos\theta + \cos r\theta\sin\theta}{\sin\theta}$$

$$\Rightarrow U_{r+1}(x) \;= \frac{\sin(r+2)\theta}{\sin\theta}$$

$$\Rightarrow U_{r-1}(x) \;= \frac{\sin r\theta}{\sin\theta}$$

$$\Rightarrow U_{r+1}(x) \;=\; 2xU_r(x) - U_{r-1}(x)$$

Hence $T_n(x)$ is the Chebyshev polynomial of the first kind and $U_r(x)$ is the Chebyshev polynomial of the second kind.

4.1.2. Properties of Chebyshev polynomials

Chebyshev polynomial of first kind $T_n(x)$ satisfies the following properties

i. The polynomials of even order are obviously even functions of x and the polynomials of odd order are odd functions of x.

ii. All $T_n(x)$ have the value unity at $x=1$, and at $x=-1$ the value is 1 for even n and -1 for odd n.

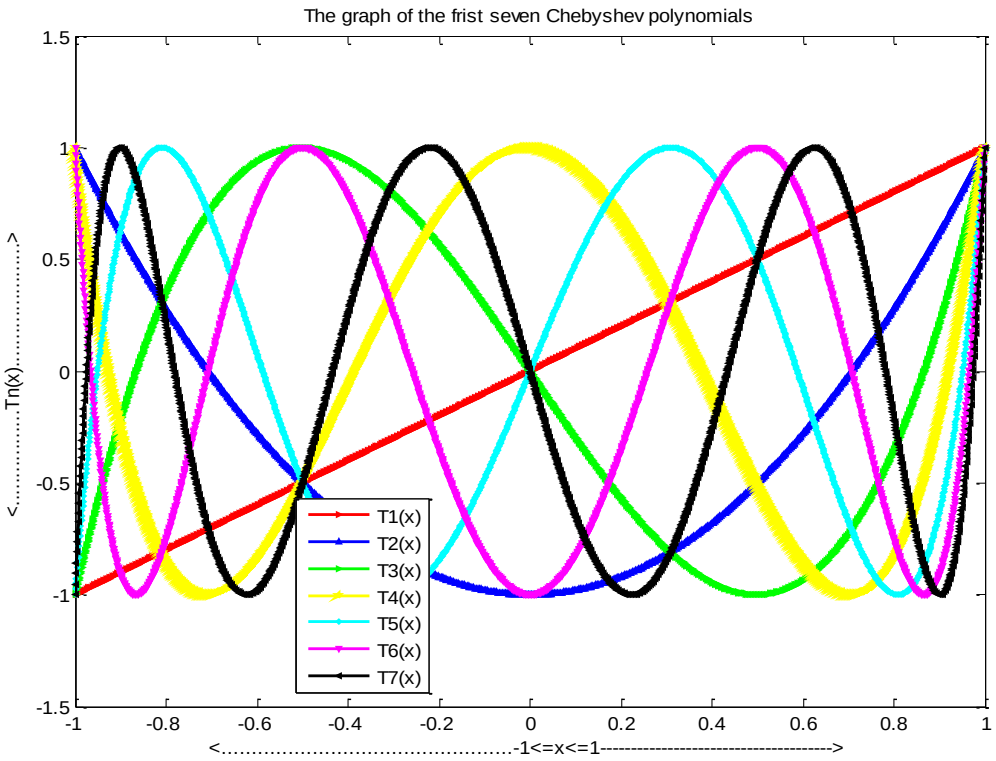
iii. The turning points of $T_n(x)$ occur at the zeros of $\sin(n\theta)/\sin\theta$, that is at the $n-1$ points

$$\theta_i = \frac{i\pi}{n}, x_i = \cos(\frac{i\pi}{n}), \quad i = 1,\ 2,\ 3,\ \ldots, n \text{ and at these points } T_n(x_i) = (-1)^i$$

iv. The turning points are separated by the n zeros, at $\theta_i = (i+\frac{1}{2})\frac{\pi}{n}$,

$$x_i = \cos(i+\tfrac{1}{2})\frac{\pi}{n}\ ,\ i = 1,\ 2,\ 3,\ \ldots, n$$

The figure 1 below shows the graph of the first seven Chebyshev polynomials for $x \in [-1,1]$.



From the above graph of Chebyshev polynomials we observe that for $x \in [-1,1]$ the corresponding value of the polynomial is also in the range $[-1,1]$.

4.1.3. Runge-Kutta Method for second order ODE

Runge-Kutta method is a numerical method used to find approximate solution for initial value problems. In order to use Runge-Kutta method to find an approximate solution of second order ODE we convert in to a system of two first order ODEs. For two evaluation of f the method is given by

$$y_{j+1} = y_j + hy'_j + \frac{1}{2}(K_1 + K_2)$$

$$y'_{j+1} = y_j + hy'_j + \frac{1}{2h}(K_1 + 3K_2)$$

where

$$K_1 = \frac{h^2}{2} f(x_j, y_j)$$

$$K_2 = \frac{h^2}{2} f(x_j + \frac{2}{3}h, y_j + \frac{2}{3}hy'_j + \frac{4}{9}K_1)$$

4.1.4. Secant Method

In this method, we approximate the graph of the function $y = f(x)$ in the neighborhood of the root by a straight line (secant) passing through the points $(x_{k-1}, f(x_{k-1}))$ and $(x_k, f(x_k))$, where $f_k = f(x_k)$ and take the point of intersection of this line with the *x*-axis as the next iterate. We thus obtain

$$x_{k+1} = x_k - \frac{x_k - x_{k-1}}{f_k - f_{k-1}} f_k, \text{ k=1,2, ...} \quad \text{Or}$$

$$x_{k+1} = \frac{x_{k-1} f_k - x_k f_{k-1}}{f_k - f_{k-1}}, \text{ k=1,2, ...}$$

where x_{k-1} and x_k are two consecutive iterates. In this method, we need two initial approximations x_0 and x_1. This method is also called the *chord* method. The order of the method is 1.8.

4.2 Mathematical formulation of the method

We consider a general linear second order differential equation with two type boundary conditions

Type I: $\alpha(x^*)\frac{d^2y}{dx^{*2}} + \beta(x^*)\frac{dy}{dx^*} + \delta(x^*)y = g(x^*)\ ;\ a \le x^* \le b$ (4.2)

with boundary condition $\begin{matrix} y(a) = \mu_1 \\ y'(b) = \mu_2 \end{matrix}$ (Mixed type)

Type II: $\alpha(x^*)\frac{d^2y}{dx^{*2}} + \beta(x^*)\frac{dy}{dx^*} + \delta(x^*)y = g(x^*)\ ;\ a \le x^* \le b$ (4.3)

with boundary condition $\begin{matrix} y'(a) = \beta_0 \\ y'(b) = \beta_1 \end{matrix}$ (Neumann type)

where $\alpha(x^*), \beta(x^*), \delta(x^*)$, $g(x^*)$ are given continuous functions for $a \le x^* \le b$ where β_0, β_1, μ_0 and μ_1 are given constants and $y(x^*)$ is unknown function or exact solution of the boundary value problem which is to be determined.

In a BVP with mixed boundary condition, the solution is required to satisfy a Dirichlet or a Neumann boundary condition in a mutually exclusive way on disjoint parts of the boundary.

Suppose that we have a BVP of type II (Neumann type). To use an approximating polynomial defined for $x \in [-1,1]$, we have to convert the given BVP defined on arbitrary interval [a, b] into an equivalent BVP defined on [-1, 1]. So we need an approximating polynomial defined on [-1, 1]. Since Chebyshev polynomial is defined on [-1, 1], it is possible to use Chebyshev polynomial after converting the BVP defined on arbitrary interval [a, b] into an equivalent BVP defined on [-1, 1].

4.1.5. Conversion of the domain of BVP

In order to use Chebyshev polynomial, we convert the BVP in (4.3) to an equivalent BVP on [-1, 1]. The differential equation in (4.3) together with the Neumann boundary condition can be converted to an equivalent problem on [-1, 1] by letting

$$x^* = \frac{b-a}{2}x + \frac{b+a}{2},\ for -1 \le x \le 1\ and\ a \le x^* \le b$$

Then equation (4.2) with boundary condition is equivalent to the BVP given by

$$\tilde{\alpha}(x)\frac{d^2y}{dx^2} + \tilde{\beta}(x)\frac{dy}{dx} + \tilde{\delta}(x)y = \tilde{g}(x); \quad ; -1 \le x \le 1 \tag{4.4}$$

Subject to the boundary condition,

$$\begin{aligned} y'(-1) &= d_0 \\ y'(1) &= d_1 \end{aligned} \tag{4.5}$$

where

$\tilde{\alpha}(x) = \frac{4}{(b-a)^2}\alpha\left(\frac{b-a}{2}x + \frac{b+a}{2}\right)$ since for $x^* = \frac{b-a}{2}x + \frac{b+a}{2}$,

we have, $\frac{dy}{dx^*} = \frac{2}{(b-a)}\frac{dy}{dx}$ *and* $\frac{d^2y}{dx^{*2}} = \frac{4}{(b-a)^2}\frac{d^2y}{dx^2}$ (4.6)

If we equate the D.E in (4.2) with (4.4), we have

$$\alpha(x^*)\frac{d^2y}{dx^{*2}} + \beta(x^*)\frac{dy}{dx^*} + \delta(x^*)y - g(x^*) = \tilde{\alpha}(x)\frac{d^2y}{dx^2} + \tilde{\beta}(x)\frac{dy}{dx} + \tilde{\delta}(x)y - \tilde{g}(x)$$

$$\Rightarrow \tilde{\alpha}(x)\frac{d^2y}{dx^2} = \alpha(x^*)\frac{d^2y}{dx^{*2}}, \tilde{\beta}(x)\frac{dy}{dx} = \beta(x^*)\frac{dy}{dx^*}, \tilde{\delta}(x) = \delta(x^*) \text{ and } \tilde{g}(x) = g(x^*) \tag{4.7}$$

Therefore, the DE in (4.3) with Neumann boundary condition is an equivalent BVP with the BVP of the DE in (4.4) with the boundary condition in (4.5).

Up on substitution of (4.6) into (4.7), we get

$$\frac{4}{(b-a)^2}\alpha(x^*)\frac{d^2y}{dx^2} = \tilde{\alpha}(x)\frac{d^2y}{dx^2}, \qquad \frac{2}{(b-a)}\beta(x^*)\frac{dy}{dx} = \tilde{\beta}(x)\frac{dy}{dx},$$

$$\tilde{\delta}(x)y = \delta(x^*)y \text{ and } \tilde{g}(x) = g(x^*)$$

$$\Rightarrow \tilde{\alpha}(x) = \frac{4}{(b-a)^2}\alpha(\frac{b-a}{2}x + \frac{b+a}{2}),\quad \tilde{\beta}(x) = \frac{2}{(b-a)}\beta(\frac{b-a}{2}x + \frac{b+a}{2}),$$

$$\tilde{\delta}(x)=\delta(\frac{b-a}{2}x + \frac{b+a}{2}) \text{ and } \tilde{g}(x)=g(\frac{b-a}{2}x + \frac{b+a}{2}).$$

4.1.6. Applying Galerkin method

To apply the technique of Galerkin method to find an approximate solution of (4.2), say $\bar{y}(x)$, written as a linear combination of base functions and unknown constants. That is;

$$\bar{y}(x) = \sum_{i=0}^{n} c_i T_i(x) \tag{4.8}$$

where $T_i(x)$ are piecewise polynomial, namely Chebyshev polynomials of degree *i* and c_i'*s* are unknown parameters, to be determined.

Now applying Galerkin method with the basis function $T_i(x)$
We get

$$\int_{-1}^{1}[\tilde{\alpha}(x)\frac{d^2\bar{y}}{dx^2} + \tilde{\beta}(x)\frac{d\bar{y}}{dx} + \tilde{\delta}(x)\bar{y}]T_j(x)dx = \int_{-1}^{1}\tilde{g}(x)T_j(x)dx \tag{4.9}$$

Integrating the first term by parts on the left hand side of (4.9), that is

$$\int_{-1}^{1} T_j(x)\tilde{\alpha}(x)\frac{d^2\bar{y}}{dx^2}dx,\quad u = T_j(x)\tilde{\alpha}(x) \text{ and } dv = \frac{d^2\bar{y}}{dx^2}dx$$

$$\Rightarrow du = \frac{d}{dx}\Big[T_j(x)\tilde{\alpha}(x)\Big]dx \text{ and } v = \frac{d\bar{y}}{dx}$$

$$\Rightarrow \int_{-1}^{1}\tilde{\alpha}(x)\frac{d^2\bar{y}}{dx^2}dx = uv - \int vdu$$

$$= \frac{d\bar{y}}{dx}T_j(x)\tilde{\alpha}(x)\Big|_{-1}^{1} - \int_{-1}^{1}\frac{d\bar{y}}{dx}\frac{d}{dx}\Big[T_j(x)\tilde{\alpha}(x)\Big]dx \tag{4.10}$$

Upon substitution of (4.10) into (4.9), we get

$$\int_{-1}^{1}[-\frac{d\bar{y}}{dx}\frac{d}{dx}\Big(\tilde{\alpha}(x)T_j(x)\Big) + \tilde{\beta}(x)\frac{d\bar{y}}{dx}T_j(x) + \tilde{\delta}(x)\bar{y}(x)T_j(x)]dx$$

$$= \int_{-1}^{1}\tilde{g}(x)T_j(x)dx + \tilde{\alpha}(-1)\bar{y}'(-1)T_j(-1) - \tilde{\alpha}(1)\bar{y}'(1)T_j(1) \tag{4.11}$$

But from equation (4.8) the approximate solution is given by

$$\bar{y}(x) = \sum_{i=0}^{n} c_i T_i(x)$$

Substituting this into equation (4.11) above we get

$$\int_{-1}^{1} [-\sum_{i=0}^{n} c_i T_i'(x) \frac{d}{dx}\left(\tilde{\alpha}(x) T_j(x)\right) + \tilde{\beta}(x) \sum_{i=0}^{n} c_i T_i'(x) T_j(x) + \tilde{\delta}(x) \sum_{i=0}^{n} c_i T_i(x) T_j(x)] dx$$

$$= \int_{-1}^{1} \tilde{g}(x) T_j(x) dx + \tilde{\alpha}(-1)\tilde{u}'(-1) T_j(-1) - \tilde{\alpha}(1)\tilde{u}'(1) T_j(1)$$

$$\Rightarrow \sum_{i=0}^{n} c_i \int_{-1}^{1} [-T_i'(x) \frac{d}{dx}\left(\tilde{\alpha}(x) T_j(x)\right) + \tilde{\beta}(x) T_i'(x) T_j(x) + \tilde{\delta}(x) T_i(x) T_j(x)] dx$$

$$= \int_{-1}^{1} \tilde{g}(x) T_j(x) dx + \tilde{\alpha}(-1)\bar{y}'(-1) T_j(-1) - \tilde{\alpha}(1)\bar{y}'(1) T_j(1) \qquad (4.12)$$

In the left hand side of equation (4.12) above we need to know the values of $\bar{y}'(-1)$ and $\bar{y}'(1)$ which approximately equal to $y'(-1)$ and $y'(1)$ respectively, where y is the exact solution of the DE in (4.4) with the boundary condition in equation (4.5).

4.1.7. The resulting system of equation

Now we have the values of $y'(-1)$ and $y'(1)$ from the boundary condition. After substituting these values into (4.12), we get a system of $n \times n$ equations to solve the parameters c_i's thus equation (4.12) in matrix form becomes:

$$\sum_{i=1}^{n} c_i K_{ij} = F_i \qquad (4.13)$$

Where $K_{ij} = k_{ij}^{(1)} + k_{ij}^{(2)} + k_{ij}^{(3)}$ and $F_i = f_i^{(1)} + f_i^{(2)}$ such that

$$k_{ij}^{(1)} = \int_{-1}^{1} [-T_i'(x) \frac{d}{dx}\left(\tilde{\alpha}(x) T_j(x)\right) dx \qquad k_{ij}^{(2)} = \int_{-1}^{1} \tilde{\beta}(x) T_i'(x) T_j(x) dx$$

$$k_{ij}^{(3)} = \int_{-1}^{1} \tilde{\delta}(x) T_i(x) T_j(x) dx \qquad f_i^{(1)} = \int_{-1}^{1} \tilde{g}(x) T_j(x) dx$$

$$f_i^{(2)} = \tilde{\alpha}(-1)\bar{y}'(-1)T_j(-1) - \tilde{\alpha}(1)\bar{y}'(1)T_j(1)$$

Now, the unknown parameters c_i's are determined by solving the system of equation in (4.13) by direct method and substituting these values into (4.8), we get the approximate solution $\bar{y}(x)$ of the DE in (4.4) satisfying the given boundary conditions in (4.5).

Suppose we have a BVP of type I. In this case it is impossible to use the above method directly; since $y'(a)$ is not given and hence instead we convert the BVP in to type II. In order to convert from type (II) to type (I) we use different numerical methods.

We consider solving the following boundary-value problem:

$$\alpha(x)\frac{d^2y}{dx^2} + \beta(x)\frac{dy}{dx} + \delta(x)y = g(x)\ ;\ a \le x \le b \qquad (4.14)$$

with boundary condition $\begin{matrix} y(a) = \mu_1 \\ y'(b) = \mu_2 \end{matrix}$ (Mixed type)

The idea of shooting method for (4.14) is to solve for $y'(a)$ hoping that $y'(b) = \mu_2$ .In order to find $y'(a)$ such that $y'(b) = \mu_2$, we guess $y'(a) = z$ and solve for $y'(b)$ using Runge-Kutta method for second order ODE, After we get a value using the guess we denote this approximate solution y_z and hope $y_z'(b) = \mu_2$. If not, we use another guess for $y'(a)$, and try to solve using the Runge-Kutta method. This process is repeated and can be done systematically until this choice satisfy $y'(b)$.

To do this, let us follow the steps below.

Step1:- select z_0 so that $y_z'(b) = \mu_2$, let $\psi(z) = y_z'(b) - \mu_2$. The guess for z_0

Step 2:- Now the objective is simply to solve for $\psi(z) = 0$, hence secant method can be used.

Step 3:- How to compute z

Suppose we have solutions $y'_{z_0}(b)$ and $y'_{z_1}(b)$ obtained from guesses z_0 and z_1 respectively.

Step 4:- Now we use secant method to find z_2 given by;

$$z_{k+1} = z_k - \frac{z_k - z_{k-1}}{y_{z_k} - y_{z_{k-1}}} y_{z_k}, \text{ k=1,2, ...}$$

Following this sequence of iteration we reach at z such that

$y'_z(b) = y'(b)$

Thus, we have a Neumann type boundary condition for the DE in (4.14) given by

$$y'(a) = z$$
$$y'(b) = \mu_2 \qquad (4.15)$$

Now to solve the DE with boundary condition in (4.15) it is convenient to use the (4.12).

4.1.8. Numerical illustration of the method

In order to show the importance and applicability of the method, we need to take problems and solve using this method. After obtaining the approximate solution we compare the result with the exact solution. We have solved one second order boundary value problems whose exact solution is known.

Problem 1: Consider the linear boundary value problem

$\frac{d^2y}{dx^2} + y = x^2e^{-x}$; $0 \le x \le 10$, subject to the boundary condition $y(0) = -0.2678$, $y'(10) = 0$

Whose exact solution is:-

$y = -11/5000sin(x)(349sin(10)e^{(10)} - 22500)/cos(10)/e^{(10)} - 3839/5000cos(x) + 1/2(x+1)^2e^{(-x)}$

Solution: - The above problem is a mixed boundary condition or (type II); to apply the above method we need to convert the given boundary condition in to Neumann boundary condition. Now we have a guess depending on the value of $y'(10) = 0$, let $y'(0) = -1$ be the first guess and hoping that $y'(10) = 0$.The next step is using Rugekutta method for second order differential equation, where $y''(x) = f(x, y, y')$. But for this problem, $y''(x) = f(x, y)$ since f is independent of y'

$x_0 = 0$, and $x_{end} = 10$ if we take step size h=0.5, $y = y(x)$ = -0.2678 *for* $x = 0 \Rightarrow y_0 = -0.2678$

$y'(x) = -1$ *for* $x = 0 \Rightarrow y'_0 = -1$

$$y_{j+1} = y_j + hy'_j + \frac{1}{2}(K_1 + K_2)$$

$$y'_{j+1} = y_j + hy'_j + \frac{1}{2h}(K_1 + 3K_2)$$

$$K_1 = \frac{h^2}{2} f(x_j, y_j)$$

$$K_2 = \frac{h^2}{2} f(x_j + \frac{2}{3}h, y_j + \frac{2}{3}hy'_j + \frac{4}{9}K_1) \quad for\ j = 0,1,2...20$$

We get the result in table 4.1 for the first iteration, where in the i^{th} step $x = x_i$, $y = y(x_i)$ and $y' = y'(x_i)$

Referring to table 4.1 in the next page, now we have $y'_z = 0.5818$. But, $y'_z(10) \neq y'(10)$, so that we need to guess another value for $y'(0)$. Let $y'(0) = 1$ hoping that $y'(10) = 0$. Using Runge-Kutta method for $x = 0$ *to* $x = 10$ and taking $h = 0.5$, $y = y(x) = -0.2678$ *for* $x = 0$ and $y' = y'(x) = 1$ *for* $x = 0$ we get the following result for the second iteration, where in the i^{th} step $x = x_i$, $y = y(x_i)$ and $y' = y'(x_i)$

Table 4.1

	x	y	y’
1^{st} iteration	0.00	–0.2670	–1.0000
	0.50	–0.6741	–0.7212
	1.00	–0.8563	–0.1822
	1.50	–0.7336	0.4641
	2.00	–0.3198	1.0127
	2.50	0.2808	1.2968
	3.00	0.9088	1.2345
	3.50	1.3967	0.8435
	4.00	1.6149	0.2289
	4.50	1.5043	–0.4500
	5.00	1.0893	–1.0227
	5.50	0.4691	–1.3505
	6.00	–0.2095	–1.3596
	6.50	–0.7891	–1.0566
	7.00	–1.1407	–0.5233
	7.50	–1.1939	0.1056
	8.00	–0.9514	0.6776
	8.50	–0.4858	1.0594
	9.00	0.0803	1.1685
	9.50	0.6059	0.9909
	10.00	0.9659	0.5818

Table 4.2

	x	y	y’
2^{nd} iteration	0.00	-0.2678	1.0000
	0.50	0.2445	1.0344
	1.00	0.7231	0.9181
	1.50	1.1004	0.6723
	2.00	1.3183	0.3161
	2.50	1.3383	-0.1000
	3.00	1.1561	-0.4984
	3.50	0.8081	-0.7958
	4.00	0.3664	-0.9276
	4.50	-0.0772	-0.8663
	5.00	-0.4321	-0.6297
	5.50	-0.6297	-0.2761
	6.00	-0.6397	0.1101
	6.50	-0.4754	0.4395
	7.00	-0.1892	0.6393
	7.50	0.1407	0.6697
	8.00	0.4302	0.5327
	8.50	0.6095	0.2697
	9.00	0.6385	-0.0497
	9.50	0.5154	-0.3458
	10.00	0.2754	-0.5480

We have $y'_{z_1}(10) = -0.5480$ *where* $z_1 = 1$. Now we have z_0 and z_1 to find z_2

$$z_2 = z_1 - \frac{z_1 - z_0}{y_{z_1} - y_{z_0}} y_{z_1} = 1 - \frac{1-(-1)}{-0.5480 - 0.5818}(-0.5480) = 0.0299$$

Using $z_2 = 0.0299$ applying Runge-Kutta method where h=0.5, $y = y(x) = -0.2678$ *for* $x = 0$ and $y' = y'(x) = 0.0299$ *for* $x = 0$ we get the following result for the 3rd iteration.

Table 4.3

	x	y	y'
3rd iteration	0.00	-0.2678	0.0299
	0.50	-0.2014	0.1830
	1.00	-0.0432	0.3847
	1.50	0.2108	0.5717
	2.00	0.5239	0.6543
	2.50	0.8256	0.5778
	3.00	1.0365	0.3422
	3.50	1.0939	-0.0008
	4.00	0.9722	-0.3669
	4.50	0.6899	-0.6647
	5.00	0.3058	-0.8206
	5.50	-0.0970	-0.7975
	6.00	-0.4313	-0.6029
	6.50	-0.6278	-0.2861
	7.00	-0.6510	0.0755
	7.50	-0.5068	0.3963
	8.00	-0.2399	0.6032
	8.50	0.0784	0.6530
	9.00	0.3680	0.5413
	9.50	0.5596	0.3025
	10.00	0.6105	-0.0002

Now calculate the next guess

$$z_3 = z_2 - \frac{z_2 - z_1}{y_{z_2} - y_{z_1}} y_{z_2} = 0.0299 - \frac{0.0299 - 1}{-0.0002 + 0.5480}(0.0299) = 0.031$$

Applying Runge-Kutta method for $y = 0.031$ we have the following result for the 4th iteration.

Table 4.4

	x	y	y'
4th iteration	0.00	-0.2670	0.0301
	0.50	-0.2006	0.1828
	1.00	-0.0427	0.3842
	1.50	0.2110	0.5710
	2.00	0.5237	0.6536
	2.50	0.8252	0.5772
	3.00	1.0358	0.3419
	3.50	1.0933	-0.0007
	4.0	0.9716	-0.3665
	4.50	0.6897	-0.6642
	5.00	0.3058	-0.8200
	5.50	-0.0967	-0.7969
	6.00	-0.4308	-0.6026
	6.50	-0.6272	-0.2861
	7.00	-0.6504	0.0753
	7.50	-0.5064	0.3959
	8.00	-0.2398	0.6027
	8.50	0.0782	0.6524
	9.00	0.3676	0.5409
	9.50	0.5591	0.3024
	10	0.6101	0.0000

As we have seen from table 4.4 above the guess for $y'(0) \approx 0.030$. Thus, we have a Neumann boundary value problem given by

$$\frac{d^2y}{dx^2} + y = x^2e^{-x}; \quad 0 \le x \le 10$$

$y'(0) = 0.030$, and $y'(10) = 0$

The next step is converting the BVP into equivalent BVP defined for $-1 \le x \le 1$ by letting

$x = \frac{(b-a)}{2}x + \frac{(b+a)}{2} = 5x + 5$, since $a = 0$ and $b = 10$.

The equivalent BVP for the above problem on $-1 \le x \le 1$ becomes,

$$\frac{1}{25}\frac{d^2y}{dx^2} + y = (5x+5)^2 e^{-(5x+5)}, \text{ for } -1 \le x \le 1$$

with boundary condition (4.1.1)

$$y'(-1) = 0.030$$
$$y'(1) = 0$$

Now, suppose that $\bar{y}$ be the approximate solution of (4.1.1), given by a linear combination of constants $c_i's$ and an approximating polynomial, called Chebyshev polynomial, thus we have,

$$\bar{y} = \sum_{i=1}^{n} c_i T_i(x) \qquad (4.1.2)$$

Upon substitution of $\bar{y}$, the approximate solution, into the differential equation in (4.1.1) we get an equation called residue given by:

$$R(x, c_i) = \frac{1}{25}\frac{d^2\bar{y}}{dx^2} + \bar{y} - (5x+5)^2 e^{-(5x+5)} \approx 0, \text{ for } -1 \le x \le 1 \qquad (4.1.3)$$

Applying Galerkin method we get;

$$\int_{-1}^{1} R(c_i, x)T_j(x)dx = 0$$

$$\Rightarrow \int_{-1}^{1} [\frac{1}{25}\frac{d^2\bar{y}}{dx^2} + \bar{y} - (5x+5)^2 e^{-(5x+5)}]T_j(x)dx \approx 0$$

$$\Rightarrow \int_{-1}^{1} [\frac{1}{25}\bar{y}"T_j(x) + \bar{y}T_j(x)]dx = \int_{-1}^{1} (5x+5)^2 e^{-(5x+5)}T_j(x)dx \qquad (4.1.4)$$

Using integration by parts to simplify the first term in the right hand side, we let

$u = T_j(x) \Rightarrow du = T_j'(x)dx$ and $dv = \bar{y}''(x)dx \Rightarrow v = \bar{y}'(x)$, hence we get

$$\int_{-1}^{1} \frac{1}{25}\bar{y}''T_j(x)dx = \frac{1}{25}\left(\bar{y}'(x)T_j(x)|_{-1}^{1} - \int_{-1}^{1} \bar{y}'T_j'(x)dx \right)$$

Therefore equation (4.1.4) becomes

$$\int_{-1}^{1} \bar{y}T_j(x)dx - \frac{1}{25}\int_{-1}^{1} \bar{y}'T_j'(x)dx = -\frac{1}{25}\bar{y}'(x)T_j(x)|_{-1}^{1} + \int_{-1}^{1} (5x+5)^2 e^{-(5x+5)}T_j(x)dx \qquad (4.1.5)$$

Substituting the approximate solution $\bar{y} = \sum_{i=1}^{n} c_i T_i(x)$ into (4.1.5) we obtain,

$$\sum_{i=1}^{n} c_i \left(\int_{-1}^{1} T_i(x)T_j(x)dx - \frac{1}{25}\int_{-1}^{1} T_i'(x)T_j'(x)dx \right) = -\frac{1}{25}\bar{y}'(x)T_j(x)|_{-1}^{1} + \int_{-1}^{1} (5x+5)^2 e^{-(5x+5)}T_j(x)dx$$

$$\sum_{i=1}^{n} c_i \left(\int_{-1}^{1} T_i(x)T_j(x)dx - \frac{1}{25}\int_{-1}^{1} T_i'(x)T_j'(x)dx \right) = \frac{1}{25}\bar{y}'(-1)T_j(-1) - \frac{1}{25}\bar{y}'(1)T_j(1) \qquad (4.1.6)$$

$$+ \int_{-1}^{1} (5x+5)^2 e^{-(5x+5)}T_j(x)dx$$

Now, using the given boundary condition in to (4.1.6), we get

$$\sum_{i=1}^{n} c_i \left(\int_{-1}^{1} T_i(x)T_j(x)dx - \frac{1}{25}\int_{-1}^{1} T_i'(x)T_j'(x)dx \right) = \int_{-1}^{1} (5x+5)^2 e^{-(5x+5)}T_j(x)dx + \frac{1}{25}(0.030)T_j(-1) \qquad (4.1.7)$$

For equation (4.1.7) we have a system of equation given by:

$$\sum_{i=1}^{n} c_i K_{ij} = F_i \qquad (4.1.8)$$

where $K_{ij} = k_{ij}^{(1)} + k_{ij}^{(2)}$ and $F_i = f_i^{(1)} + f_i^{(2)}$

such that $k_{ij}^{(1)} = \int_{-1}^{1} T_i(x) T_j(x) dx$

$$k_{ij}^{(2)} = -\frac{1}{25} \int_{-1}^{1} T_i'(x) T_j'(x) dx$$

$$f_i^{(1)} = \frac{1}{25}(0.030) T_j(-1)$$

$$f_i^{(2)} = \int_{-1}^{1} \left(5x+5\right)^2 e^{-(5x+5)} T_j(x) dx$$

In order to find the value of c_i's we take n trial functions defined for $x \in [-1,1]$, as mentioned earlier we use Chebyshev polynomials as trial function. For $n = 6$, we get:

$$T' = \begin{bmatrix} x \\ 2x^2 - 1 \\ 4x^3 - 3x \\ 8x^4 - 8x^2 + 1 \\ 16x^5 - 20x^3 + 5x \\ 32x^6 - 48x^4 + 18x^2 - 1 \end{bmatrix}$$, where $T = [T_1 \;\; T_2 \;\; T_3 \;\; T_4 \;\; T_5 \;\; T_6]$ and T' is the transpose of T.

$$k_{ij}^{(1)} = \int_{-1}^{1} T_i(x) T_j(x) dx = \begin{bmatrix} 2/3 & 0 & -2/5 & 0 & -2/21 & 0 \\ 0 & 14/15 & 0 & -38/105 & 0 & -26/315 \\ -2/5 & 0 & 34/35 & 0 & -22/63 & 0 \\ 0 & -38/105 & 0 & 62/63 & 0 & -34/99 \\ -2/21 & 0 & -22/63 & 0 & 98/99 & 0 \\ 0 & -26/315 & 0 & -34/99 & 0 & 142/143 \end{bmatrix}$$

$$k_{ij}^{(2)} = -\frac{1}{25}\int_{-1}^{1} T_i'(x)T_j'(x)dx = \begin{bmatrix} 2/25 & 0 & 2/25 & 0 & 2/25 & 0 \\ 0 & 32/75 & 0 & 128/375 & 0 & 288/875 \\ 2/25 & 0 & 138/125 & 0 & 142/175 & 0 \\ 0 & 128/375 & 0 & 5632/2625 & 0 & 3968/2625 \\ 2/25 & 0 & 142/175 & 0 & 1126/315 & 0 \\ 0 & 288/875 & 0 & 3968/2625 & 0 & 52064/9625 \end{bmatrix}$$

To find the value of the coefficient matrix in equation (4.1.8) we use

$$K_{ij} = k_{ij}^{(1)} + k_{ij}^{(2)}$$

So, when we express the coefficient matrix $K_{i,j}$ and the unknown coefficient $c_i's$ in a system of equation in matrix form, we get:

$K_{ij} = k_{ij}^{(1)} + k_{ij}^{(2)}$

$$\begin{bmatrix} 44/75 & 0 & -12/25 & 0 & -92/525 & 0 \\ 0 & 38/75 & 0 & -1846/2625 & 0 & -3242/7875 \\ -12/25 & 0 & -116/875 & 0 & -1828/1575 & 0 \\ 0 & -1846/2625 & 0 & -9146/7875 & 0 & -160694/86625 \\ -92/525 & 0 & -1828/1575 & 0 & -8956/3465 & 0 \\ 0 & -3242/7875 & 0 & -160694/86625 & 0 & -552582/125125 \end{bmatrix} \begin{bmatrix} c_1 \\ c_2 \\ c_3 \\ c_4 \\ c_5 \\ c_6 \end{bmatrix} = [f_i] \quad (4.1.9)$$

Where f_i is 6×1 a column vector given by:-

$$f_i = \begin{bmatrix} -4/25 - 756/25e^{(-10)} \\ -22/125 - 6658/125e^{(-10)} \\ 28/125 - 14708/125e^{(-10)} \\ -38/625 - 189682/625e^{(-10)} \\ 28/3125 - 2774708/3125e^{(-10)} \\ 2818/15625 - 45459898/15625e^{(-10)} \end{bmatrix}$$

Now we have a 6×6 coefficient matrix which is symmetric, 6×1 unknown column vector that represent $c_i's$ and 6×1 column vector that represents f_i. So we have six equations with six unknowns. When we use $c_i = \left(K_{i,j}\right)^{-1} f_i$ to solve (4.1.9), we get:

$c_1 = 0.361550$

$c_2 = -0.002596$

$c_3 = 0.939452$

$c_4 = 0.360917$

$c_5 = -0.449830$

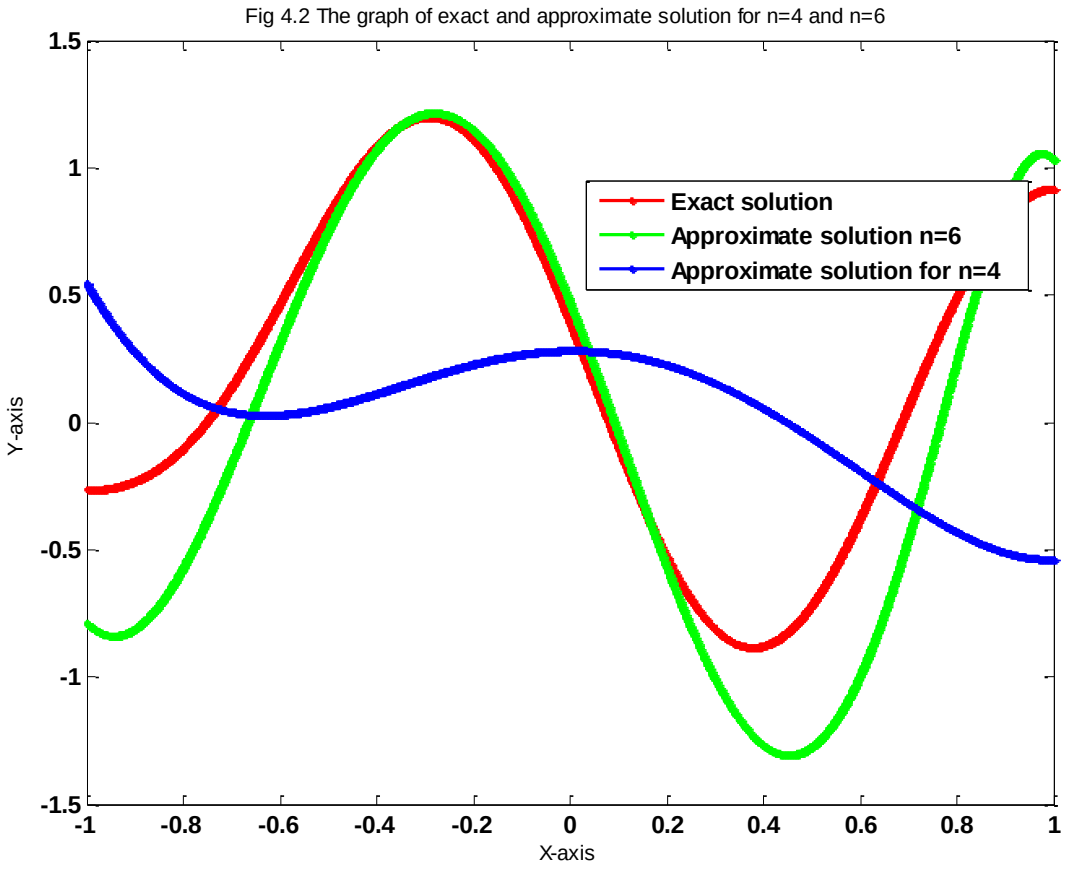
$c_6 = -0.192200$

Now we can express the approximate solution as a linear combination of constants c_i's and an approximating polynomial. So when we substitute c_i's and Chebyshev polynomials for n=6 we get:

$$\bar{y} = 0.361550x - 0.002596(2x^2 - 1) + 0.939452(4x^3 - 3x) + 0.360917(8x^4 - 8x^2 + 1)$$
$$- 0.449830(16x^5 - 20x^3 + 5x) - 0.192200(32x^6 - 48x^4 + 18x^2 - 1)$$

When we draw the graph of the exact and approximate solution to look the convergence the approximate solution to the graph of the exact solution we get the following graph between-1&1.

Fig 4.2



The above graph shows that the graph of the approximate solution approaches the graph of the exact solution for the differential equation with boundary condition in problem 1.

When we consider n=8, then the approximate solution and the Chebyshev polynomials are given by:

$$\bar{y} = \sum_{i=1}^{8} c_i T_i(x) \qquad (4.1.10)$$

$$T' = \begin{bmatrix} x \\ 2x^2 - 1 \\ 4x^3 - 3x \\ 8x^4 - 8x^2 + 1 \\ 16x^5 - 20x^3 + 5x \\ 32x^6 - 48x^4 + 18x^2 - 1 \\ 64x^7 - 112x^5 + 56x^3 - 7x \\ 1 - 32x^2 + 160x^4 - 256x^6 + 128x^8 \end{bmatrix}$$, where $T = [T_1 \;\; T_2 \;\; T_3 \;\; T_4 \;\; T_5 \;\; T_6 \;\; T_7 \;\; T_8]$ and T' is the transpose of T.

Using Matlab code on Appendix we have

$$K^{(1)}_{ij} = \begin{bmatrix} 2/3 & 0 & -2/5 & 0 & -2/21 & 0 & -2/45 & 0 \\ 0 & 14/15 & 0 & -38/105 & 0 & -26/315 & 0 & -134/3465 \\ -2/5 & 0 & 34/35 & 0 & -22/63 & 0 & -38/495 & 0] \\ 0 & -38/105 & 0 & 62/63 & 0 & -34/99 & 0 & -158/2145 \\ -2/21 & 0 & -22/63 & 0 & 98/99 & 0 & -146/429 & 0 \\ 0 & -26/315 & 0 & -34/99 & 0 & 142/143 & 0 & -22/65 \\ -2/45 & 0 & -38/495 & 0 & -146/429 & 0 & 194/195 & 0 \\ 0 & -134/3465 & 0 & -158/2145 & 0 & -22/65 & 0 & 254/255 \end{bmatrix}$$

$$K_{ij}^{(2)} = \begin{bmatrix} 2/25 & 0 & 2/25 & 0 & 2/25 & 0 & 2/25 & 0 \\ 0 & 32/75 & 0 & 128/375 & 0 & 288/875 & 0 & 512/1575 \\ 2/25 & 0 & 138/125 & 0 & 142/175 & 0 & 286/375 & 0 \\ 0 & 128/375 & 0 & 5632/2625 & 0 & 3968/2625 & 0 & 120832/86625 \\ 2/25 & 0 & 142/175 & 0 & 1126/315 & 0 & 6086/2475 & 0 \\ 0 & 288/875 & 0 & 3968/2625 & 0 & 52064/9625 & 0 & 1376768/375375 \\ 2/25 & 0 & 286/375 & 0 & 6086/2475 & 0 & 1232966/160875 & 0 \\ 0 & 512/1575 & 0 & 120832/86625 & 0 & 1376768/375375 & 0 & 11657216/1126125 \end{bmatrix}$$

$$f_i = \begin{bmatrix} -4/25 - 756/25e^{(-10)} \\ -22/125 - 6658/125e^{(-10)} \\ 28/125 - 14708/125e^{(-10)} \\ -38/625 - 189682/625e^{(-10)} \\ 28/3125 - 2774708/3125e^{(-10)} \\ 2818/15625 - 45459898/15625e^{(-10)} \\ 35404/78125 - 827348644/78125e^{(-10)} \\ 151442/78125 - 3321109962/78125e^{(-10)} \end{bmatrix}$$

Thus the unknown parameters $c_i's$ are

$c_1 = 0.375249$

$c_2 = -0.069868$

$c_3 = 0.959056$

$c_4 = 0.286293$

$c_5 = -0.454061$

$c_6 = 0.117209$

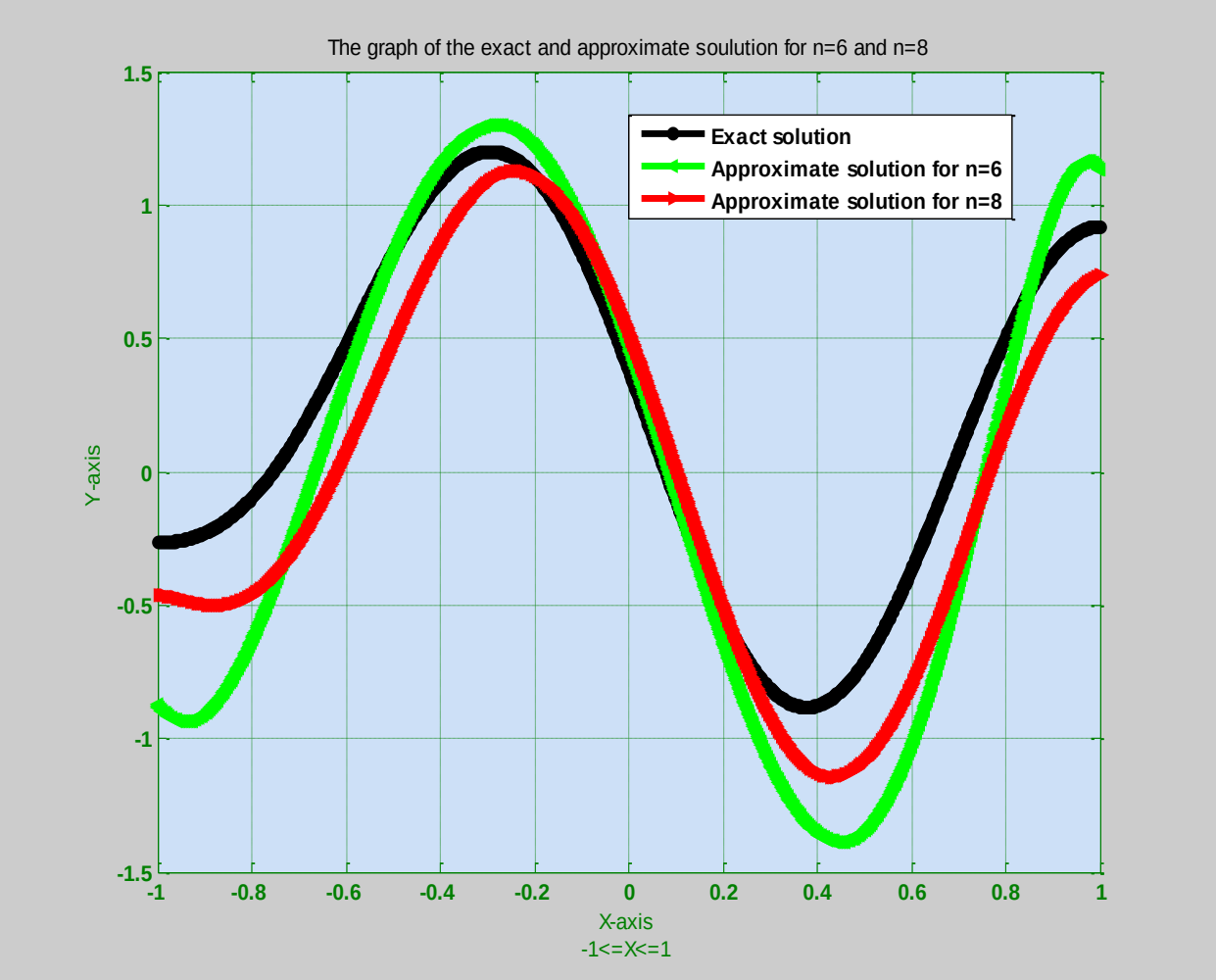
$c_7 = -0.005079$

$c_8 = -0.299613$

Now substituting the unknown parameters and eight Chebyshev polynomials in to (4.1.10) we obtain,

$$\bar{y} = 0.375249x - 0.069868(2x^2 - 1) + 0.959056(4x^3 - 3x) + 0.286293(8x^4 - 8x^2 + 1)$$
$$-0.454061(16x^5 - 20x^3 + 5x) + 0.117209(32x^6 - 48x^4 + 18x^2 - 1)$$
$$-0.005079(64x^7 - 112x^5 + 56x^3 - 7x) - 0.299613(1 - 32x^2 + 160x^4 - 256x^6 + 128x^8)$$

Fig 4.1The graph of the exact solution and approximate solution for n=6 and n=8



From the graph above the approximate solution approaches the graph of the exact solution when the number of the trial functions increases from 6 to 8.

Now we can also compare by calculating the absolute error which is given by

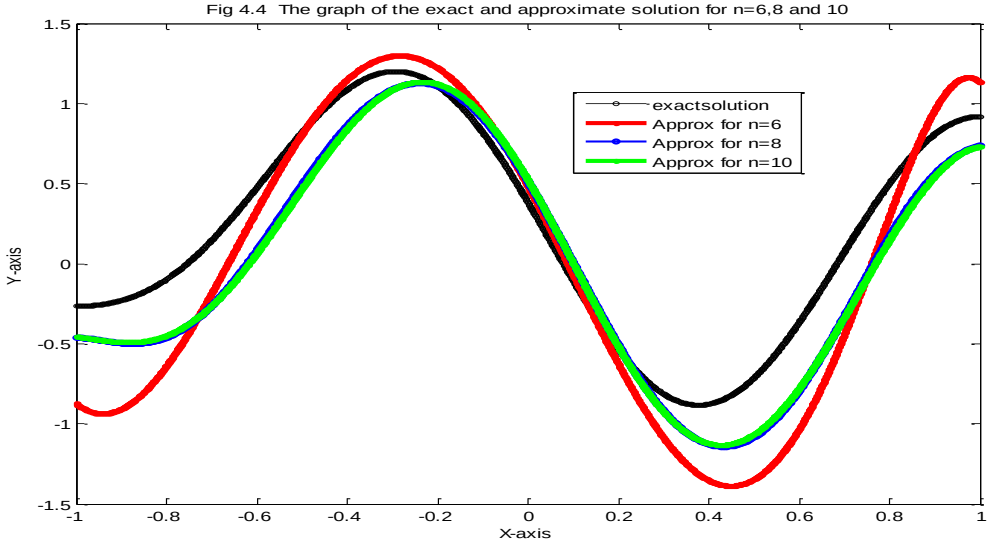
$$error = \left| y_{exact} - y_{approx} \right|$$

Table4. 5 computed the absolute error for problem 1 when n=6 and n=8

x	Exact solution	Approximate solution for n=6	Approximate solution for n=8	Absolute error for n=6	Absolute error for n=8
-1.0	-0.2670	-0.8745	-0.4640	0.6074	0.1969
-0.9	-0.2305	-0.9073	-0.4970	0.6768	0.2664
-0.8	-0.0994	-0.6369	-0.4619	0.5375	0.3625
-0.7	0.1443	-0.1813	-0.2646	0.3256	0.2089
-0.6	0.4736	0.3385	0.0794	0.1350	0.0942
-0.5	0.8180	0.8124	0.4890	0.3057	0.0291
-0.4	1.0871	1.1515	0.8573	0.0644	0.0298
-0.3	1.1994	1.2961	1.0859	0.0967	0.0136
-0.2	1.1087	1.2205	1.1089	0.1118	0.0001
-0.1	0.8185	0.9337	0.9077	0.1153	0.0892
0.0	0.3832	0.4786	0.5149	0.0954	0.0317
0.1	-0.1045	-0.0735	0.0075	0.0310	0.0120
0.2	-0.5361	-0.6303	-0.5083	0.0942	0.0277
0.3	-0.8144	-1.0896	-0.9190	0.2752	0.1046
0.4	-0.8776	-1.3536	-1.1289	0.4760	0.2513
0.5	-0.7149	-1.3457	-1.0847	0.6308	0.3698
0.6	-0.3695	-1.0312	-0.7942	0.6617	0.4247
0.7	0.0717	-0.4403	-0.3329	0.5120	0.4046
0.8	0.4990	0.3050	0.1671	0.4194	0.3319
0.9	0.8066	0.9616	0.5557	0.3155	0.2509
1.0	0.9184	1.1323	0.7374	0.2139	0.1810

If we take n=10, the graph of the corresponding approximate solution together with the graph of the approximate solution for n=6 and n=8 in the same plane with the exact solution is shown below.

Fig 4.4 The graph of the exact and approximate solution for n=6,8 and 10



As we increase the number of Chebyshev polynomial the corresponding approximate solution of the differential equation with mixed boundary condition in problem 1 approaches the graph of the exact solution.

4.3 Discussion

In this paper, Galerkin method has been used to approximate the solution of linear second order ordinary differential equations with mixed and Neumann boundary conditions. Chebyshev polynomial of first kind has been used as trial function. When we convert the boundary condition from type (I) to type (II) we use Runge-Kutta method to find the correct value of the derivative of y at the initial point. By looking at the graph of the approximate solution for n=6, n=8 and n=10 and Table 1, we observe that as we increase the number of approximating polynomial the graph approaches the exact solution very well**.**

It is observed that the approximate results converge monotonically to the exact solutions. We also notice that the approximate solutions coincide with the exact solutions as the number of trial function increases. It also observed that the accuracy of the method increases as the number of trial function increases. We realize that this method can be applied to solve other linear differential equations for the desired accuracy. In doing this we have met our objectives.

CHAPTER FIVE

5. CONCLUSION AND FUTURE SCOPE

We can conclude that by applying Galerkin method, we can find the solutions of ODE with mixed type boundaries. And the method has shown results that converge to the exact solution by increasing the number of approximate solution; and also using small step size h while converting the BVP from Mixed type to Neumann boundary condition, increases the accuracy of the approximate solution

5.1. Future Scope

This study was concentrated on numerical solution of linear second order differential equations with mixed boundary condition, by Galerkin method taking Chebyshev polynomial as a trial function. But it is also possible to use other polynomials instead of Chebyshev. We can also use this method for non- linear problems. In order to fill the Gap in terms of accuracy it is important to analyze the error of the method.

Appendix

Appendix 1 MATLAB code to find the approximate solution of problem 1 for n=6

```
Syms x
T1=x;
T2=2*x^2-1;
T3=4*x^3-3*x;
T4=8*x^4-8*x^2+1;
T5=16*x^5-20*x^3+5*x;
T6=32*x^6-48*x^4+18*x^2-1;
T=[T1,T2,T3,T4,T5,T6];
TrT=transpose(T);
DrT=diff(T,x);
DrTrT=diff(TrT,x);
%%%%%%%%%%%%%%%%%%%%%%%%
k1T=TrT*T;
k2T=DrTrT*DrT;
f1=(5*x+5)^2*exp(-5*x-5)*TrT;
%k3T=TrT*T;
K1=int(k1T,x,-1,1);
K2=(1/25)*int(k2T,x,-1,1);
F1=int(f1,x,-1,1);
KK=K1-K2;
c1=inv(KK)*F1;
yp=T*c1;
display(yp);
display(K2);
display(KK);
display(T);
display(yp);
```

Appendix 2 MATLAB code to find the approximate solution of problem 1 for n=8

```
syms x
T1=x;
T2=2*x^2-1;
T3=4*x^3-3*x;
T4=8*x^4-8*x^2+1;
T5=16*x^5-20*x^3+5*x;
T6=32*x^6-48*x^4+18*x^2-1;
T7=64*x^7-112*x^5+56*x^3-7*x;
T8=1-32*x^2+160*x^4-256*x^6+128*x^8;
%%%%%%%%%%%%%%%%%%%%%%%%%%%%
T=[T1,T2,T3,T4,T5,T6,T7,T8];
TrT=transpose(T);
DrT=diff(T,x);
DrTrT=diff(TrT,x);
%%%%%%%%%%%%%%%%%%%%%%%%
k1T=TrT*T;
k2T=DrTrT*DrT;
f1=(5*x+5)^2*exp(-5*x-5)*TrT;
%k3T=TrT*T;
K1=int(k1T,x,-1,1);
K2=(1/25)*int(k2T,x,-1,1);
F1=int(f1,x,-1,1);
```

```
KK=K1-K2;
c1=inv(KK)*F1;
yp=T*c1;
display(yp);
display(K2);
display(KK);
display(T);
display(yp);
```

Appendix 3 MATLAB code to draw the graph of problem 1for n=6

```
x=-1:0.001:1;
y=-1/2.*sin(5.*x+5)-
1/2.*cos(5.*x+5)*(cos(10)*exp(10)+99)/sin(10)/exp(10)+1/2*(5.*x+6).^2.*exp(-
5.*x-5);
yp =x*(121427607/404224000+21968890839/12632000*exp(-10))+(2.*x.^2-1)*(-
162049455981/81093058048000-1045327369091481/1267079032000*exp(-
10))+(4.*x.^3-3.*x)*(139968927/161689600+12729265479/5052800*exp(-
10))+(8.*x.^4-8.*x.^2+1)*(17673544966077/50683161280000-
996957348232923/791924395000*exp(-10))+(16.*x.^5-20.*x.^3+5.*x)*(-
331179849/808448000-22879853073/25264000*exp(-10))+(32.*x.^6-
48.*x.^4+18.*x.^2-1)*(-
76562160473799/405465290240000+8011204251751701/6335395160000*exp(-10));
plot(x,y,'r')
hold on
plot(x,yp,'g')
```

Appendix 4 MATLAB code to draw the graph of approximate solution of problem 1 for n=6 and n=8 on the same plane

```
 x=-1:0.001:1;
y=-1/2.*sin(5.*x+5)-
1/2.*cos(5.*x+5)*(cos(10)*exp(10)+99)/sin(10)/exp(10)+1/2*(5.*x+6).^2.*exp(-
5.*x-5);
yp =x*(4567101/12632000+21968890839/12632000*exp(-10))+(2.*x.^2-1)*(-
3289084029/1267079032000-1045327369091481/1267079032000*exp(-10))+(4.*x.^3-
3.*x)*(4746861/5052800+12729265479/5052800*exp(-10))+(8.*x.^4-
8.*x.^2+1)*(285818703093/791924395000-996957348232923/791924395000*exp(-
10))+(16.*x.^5-20.*x.^3+5.*x)*(-11364507/25264000-22879853073/25264000*exp(-
10))+(32.*x.^6-48.*x.^4+18.*x.^2-1)*(-
1217665439991/6335395160000+8011204251751701/6335395160000*exp(-10));
plot(x,y,'k')
hold on
plot(x,yp,'g')
yp1 =x*(13541609277/36087040000-130881304554147/36087040000*exp(-
10))+(2.*x.^2-
1)*(29932124086807173/428409627600160000+7968961772094818 8107/107102406900040
000*exp(-10))+(4.*x.^3-3.*x)*(1430143659/1491200000-
7694658500949/1491200000*exp(10))+(8.*x.^4-
8.*x.^2+1)*(61325250251706441/214204813800080000+12895371037336941153/2677560
1725010000*exp(-10))+(16.*x.^5-
20.*x.^3+5.*x)*(7448057253/16403200000+12332010646683/16403200000*exp(10))+(3
2.*x.^6-48.*x.^4+18.*x.^2-1)*(50213535949362357/428409627600160000-
637481934194670728163/107102406900040000*exp(-10))+(64.*x.^7-
112.*x.^5+56.*x.^3-7.*x)*(-
83309031/16403200000+32633922715641/16403200000*exp(-10))+(1-
```

```
32.*x.^2+160.*x.^4-256.*x.^6+128.*x.^8)*(-
64178452117914849/214204813800080000+1871103385467719733333/26775601725010000*
exp(-10));
plot(x,yp1,'r')
```

Appendix 5 MATLAB code to draw the graph of approximate solution of problem 1 for n=10

```
x=-1:0.001:1;
y=-1/2.*sin(5.*x+5)-
1/2.*cos(5.*x+5)*(cos(10)*exp(10)+99)/sin(10)/exp(10)+1/2*(5.*x+6).^2.*exp(-
5.*x-5);

yp =x*(4567101/12632000+21968890839/12632000*exp(-10))+(2.*x.^2-1)*(-
3289084029/1267079032000-1045327369091481/1267079032000*exp(-10))+(4.*x.^3-
3.*x)*(4746861/5052800+12729265479/5052800*exp(-10))+(8.*x.^4-
8.*x.^2+1)*(285818703093/791924395000-996957348232923/791924395000*exp(-
10))+(16.*x.^5-20.*x.^3+5.*x)*(-11364507/25264000-22879853073/25264000*exp(-
10))+(32.*x.^6-48.*x.^4+18.*x.^2-1)*(-
1217665439991/6335395160000+8011204251751701/6335395160000*exp(-10));

yp1 =x*(13541609277/36087040000-130881304554147/36087040000*exp(-
10))+(2.*x.^2-1)*(-
29932124086807173/428409627600160000+796896177209481881/107102406900040000*
exp(-10))+(4.*x.^3-3.*x)*(1430143659/1491200000-
7694658500949/1491200000*exp(-10))+(8.*x.^4-
8.*x.^2+1)*(61325250251706441/214204813800080000+1289537103733694115 3/2677560
1725010000*exp(-10))+(16.*x.^5-20.*x.^3+5.*x)*(-
7448057253/16403200000+12332010646683/16403200000*exp(-10))+(32.*x.^6-
48.*x.^4+18.*x.^2-1)*(50213535949362357/428409627600160000-
637481934194670728163/107102406900040000*exp(-10))+(64.*x.^7-
112.*x.^5+56.*x.^3-7.*x)*(-
83309031/16403200000+32633922715641/16403200000*exp(-10))+(1-
32.*x.^2+160.*x.^4-256.*x.^6+128.*x.^8)*(-
64178452117914849/214204813800080000+1871103385467719733333/26775601725010000*
exp(-10));

yp2=x*(5920752912921/151218821120000+559365647026332069/151218821120000*exp(-
10))+(2.*x.^2-1)*(1092293866799506016075 43/337794642853041664000000-
267268713941848360405426062 3/337794642853041664000000*exp(-10))+(4.*x.^3-
3.*x)*(171983258337891/378047052800000+2203443335395701399/378047052800000*ex
p(-10))+(8.*x.^4-
8.*x.^2+1)*(1018836907579215757542 51/105560825891575520000000-
1526680326957237145620737811/105560825891575520000000*exp(-10))+(16.*x.^5-
20.*x.^3+5.*x)*(-49337162337177/378047052800000-
238198004283870645 3/378047052800000*exp(-10))+(32.*x.^6-48.*x.^4+18.*x.^2-
1)*(5006445016551736904559 27/337794642853041664000000-
224109859079526042289757560 47/337794642853041664000000*exp(-10))+(64.*x.^7-
112.*x.^5+56.*x.^3-7.*x)*(1164846572033859/1512188211200000-
2254911677951669064 9/1512188211200000*exp(-10))+(1-32.*x.^2+160.*x.^4-
256.*x.^6+128.*x.^8)*(525374620471390224682 29/211121651783151040000000-
114822811824697867695251346 9/211121651783151040000000*exp(-10))+(9.*x-
120.*x.^3+432.*x.^5-576.*x.^7+256.*x.^9)*(-
61863589710637/60487528448000+1348442138853559607/60487528448000*exp(-10))+(-
1+50.*x.^2-400.*x.^4+1120.*x.^6-1280.*x.^8+512.*x.^10)*(-
```

```
127380162405790781743786 53/337794642853041664000000 0+28040457614012517707818 8
183333/337794642853041664000000*exp(-10));
plot(x,y,'k')
hold on
>> plot(x,yp,'r')
>> plot(x,yp1,'b')
>> plot(x,yp2,'g')
>> error1=abs(y-yp);
>> error2=abs(y-yp1);
>> error3=abs(y-yp2);
>> [error1' error2' error3']
```

Appendix 6 MATLAB code to guess the first derivative of y at initial point

```
function rungekutta2akee
%***********************Input variables**********************************
h=input('Enter the step length :');
x=input('Enter the initial point :');
xf=input('Enter the final point :');
y=input('Enter the initial value of y: ');
y2=input('Enter the guess for the derivative of y at x=0: ');
%============================================================================
n=(xf-x)/h;
%***************formating the output*************************************
fprintf('Step 0: x = %6.2f,y = %18.4f, y2 = %18.4f\n', x,y,y2);
%============================================================================
%******************Assigning variables************************************
for i=1:n
k1 = (h^2/2)*f(x,y,y2);
k2 = (h^2/2)*f(x+(h/2), y+(h/2)*y2+(1/4)*k1,y2+(k1/h));
k3 = (h^2/2)*f(x+(h/2), y+(1/2)*h*y2+(1/4)*k1,y2+(k2/h));
k4 = (h^2/2)*f(x+h, y+h*y2+k3,y2+(2/h)*k3);
y = y + h*y2+(k1+k2+k3+k4)/3;
y2=y2+(k1+2*k2+2*k3+k4)/(3*h);
x = x + h;
%***********************Display the output values************************
fprintf('Step %d: x = %6.2f,y = %18.4f, y2 = %18.4f\n', i, x,y,y2);
end
%============================================================================
%******************* Assigning the right hand side**********************
%******************* of the differential equation***********************
function [v] = f(x,y,y2)
v = -y+x^2*exp(-x);
end
%============================================================================
end
%============================================================================
```

REFERENCES

[1]. Yattender Rishi Dubey: An approximate solution to buckling of plates by the Galerkin method, (August 2005).

[2] Tai-Ran Hsu: Mechanical Engineering 130 Applied Engineering analysis, San Jose State University, (Sept 2009).

[3] E. Suli: Numerical Solution of Ordinary Differential Equations, (April 2013).

[4] J.N. Reddy: An Introduction to the finite element method, 3rd edition, McGraw-Hill, (Jan 2011) 60-98

[5] Marcos Cesar Ruggeri: Theory of Galerkin method and explanation of MATLAB code, (2006).

[6] Jalil Rashidinia and Reza Jalilian: Spline solution of two point boundary value problems, Appl. Comput. Math 9 (2010) 258-266.

[7] S. Das, Sunil Kumar and O. P. Singh: Solutions of nonlinear second order multipoint boundary value problems by Homotopy perturbation method, Appl. Appl. Math. 05 (2010) 1592-1600.

[8] M. Idress Bhatti and P. Bracken: Solutions of differential equations in a Bernstein polynomials basis, J. Comput. Appl. Math. 205 (2007) 272-280.

[9] M. M. Rahman.et.al: Numerical Solutions of Second Order Boundary Value Problems by Galerkin Method with Hermite Polynomials, (2012).

[10] Arshad Khan: Parametric cubic spline solution of two point boundary value problems, Appl. Math. Comput. 154 (2004) 175-182.

[11] Yuqiang Feng and Guangjun Li: Exact three positive solutions to a second-order Neumann boundary value problem with singular nonlinearity, Arabian J. Sci. Eng. 35 (2010) 189-195.

[12] P. M. Lima and M. Carpentier: Numerical solution of a singular boundary-value problem in non-Newtonian fluid mechanics, Computer Phys. Communica. 126 (2000) 114-120.

[13] K.N.S. Kasi Viswanadham and Sreenivasulu Ballem : Fourth Order Boundary Value Problems by Galerkin Method with Cubic B-splines, (May 2013)

[14] Jahanshahi et al.: A special successive approximations method for solving boundary value problems including ordinary differential equations, (August 2013).

[15] L. Fox and I. B. Parker: Chebyshev Polynomials in Numerical Analysis, Oxford University Press, (1 May, 1967)