



Received: 10-09-2026

Accepted: 20-10-2026

## International Journal of Advanced Multidisciplinary Research and Studies

ISSN: 2583-049X

### Boundary Value Problems for Ordinary Differential Equations Using MATLAB

<sup>1</sup> Medhat Mohammad Roueheb, <sup>2</sup> Khaled Mohamed AbdElkhalek, <sup>3</sup> Taha Gabaireldar Elradi

<sup>1,2</sup> Department of Mathematics, Faculty of Science, Sudan University of Sciences and Technology, Sudan

<sup>3</sup> Department of Mathematics, Faculty of Science, University of Albutana, Sudan

Corresponding Author: **Medhat Mohammad Roueheb**

#### Abstract

This study deals with establishing existence of solution to a boundary value problem of second order ordinary differential equation with constant delay of the type  $y''(t) = f(t; y(t); y(t - r); y'(t))$  also Numerical solution is found for the boundary value problem using MATLAB, and we give a survey of the solutions of second-order ordinary differential equation. The study followed applied analytical approach. The study reached the most important results, the numerical results that are obtained using MATLAB converges to the exact solutions strictly, the amount of

computation time required may become excessive, and good accuracy may be difficult to achieve, MATLAB automatically adjusts and increases mesh density in regions where second – order solutions experience rapid variations, sharp curvatures, or steep boundary layers. We recommend that researchers conduct further research on solutions to these Boundary value problems for ordinary differential equations using MATLAB and other software to solve and plot them.

**Keywords:** MATLAB, Boundary Value Problems (BVPs), DEDA

#### Introduction

This study of boundary value problems (BVPs) for differential equations with deviating arguments (DEDA) is important because of their applications in various fields such as production problems in Economics. Biological systems. Physical models. optimal control theory etc. Thus given a DEDA the most natural question that arises is of existence of its solution for the fixed point theory which is a main source of proving many existence results [3].

Fractional calculus is much old mathematical concept dating back to 17th century and involves integration and differentiation of arbitrary order. In a later dated 30th September 1695. L' Hospital wrote to Leibniz asking him about the differentiation of order 1/2. Leibniz response "an apparent paradox from which one day useful consequences will be drawn". In the following centuries fractional calculus developed significantly within pure mathematics. However, the applications of fractional calculus just emerged in last few decades in various areas of physics and engineering. namely in signal processing. control engineering, electromagnetism, biosciences, fluid mechanics, electrochemistry, diffusion processes, dynamic of viscoelastic materials, continuum and statistical mechanics, and propagation of spherical flames [2].

With important applications. orthogonal polynomials theory was developed during the 20th century. Krall gave necessary and sufficient conditions for orthogonal polynomial systems satisfying an  $N$ th order linear differential equation of spectral type with polynomial coefficients. Furthermore, he showed that the differential equation must have even order as an orthogonal polynomial system exists as a solution set [4].

Ordinary and partial differential equation have long played important roles in the history of modelling physical phenomena; they continue to serve as indispensable tools in future investigations. In the theory and application of ordinary and partial differential equations. one may be interested to find a solution to a differential equation satisfying certain defined conditions. If the conditions are given at only one point of the independent variable. we have initial conditions; where as if the conditions are given at more than one point of the independent variable. they are called boundary conditions (BC) [5].

This study deals with the existence and uniqueness of solutions for boundary value problems (BVP for short). for fractional order differential equations:

$${}^c D^\alpha y(t) = f(t, y). \text{ for each } t \in J = [0, T]. 2 < \alpha \leq 3 \quad (1)$$

$$y(0) = y_0, y'(0) = y_0^*, y''(T) = y_T \quad (2)$$

Where  ${}^c D^\alpha$  is the Caputo fractional derivative.  $f: J \times \mathbb{R} \rightarrow \mathbb{R}$  is a continuous function and  $y_0, y_0^*, y_T$  are real constants [6]. The general second order linear ordinary differential equation [1] is given as:

$$y_{(2)} + B(x)y_{(1)} + C(x)y = R(x) \quad (3)$$

Where  $y_{(k)}$  represents  $k^{\text{th}}$  derivative of the dependent variable -  $y$  with respect to the independent variable  $x$  i.e.

$$y_{(k)} = \frac{d^k y}{dx^k} \quad (4)$$

$B(x)$ ,  $C(x)$  and  $R(x)$  are functions of the independent variable  $x$  [7].

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 [8].

**Definition 1:** An equation containing the derivative of one or more dependent variables with respect to one or more independent variables is said to be differential equation.

**Definition 2:** A differential equation is said to be an ordinary differential equation if it contains only ordinary derivatives of one or more dependent variables with respect to a single independent variable.

**Definition 3:** An  $n^{\text{th}}$  order ordinary differential equation is said to be linear in  $y$  if it can be written in the form:

$$a_n(x)y^{(n)} + a_{n-1}(x)y^{(n-1)} + \dots + a_0(x)y = f(x) \quad (5)$$

Where  $a_n(x)$  are continuous function.

The differential equation and the boundary condition are linear (these are called BVP) and for which the boundary conditions are based on only two points say  $x_1 = a$  and  $x_2 = b$  where  $a < b$ . The differential equation may be written in the form:

$$a_n(x)y^{(n)} + a_{n-1}(x)y^{(n-1)} + \dots + a_0(x)y = f(x)$$

With boundary conditions:

$$y(x_1) = a, y(x_2) = b \quad (6)$$

The differential equation is homogeneous when  $f(x) = 0$  in  $[a, b]$  otherwise non-homogeneous. The boundary value problem is called homogeneous when the differential and all the boundary condition are homogeneous. That is:

$$a_n(x)y^{(n)} + a_{n-1}(x)y^{(n-1)} + \dots + a_0(x)y = f(x)$$

With boundary conditions  $y(x_1) = y(x_2) = \alpha$ .

**Definition 4:** A set of function  $f_1(x), f_2(x), \dots, f_n(x)$  is said to be linearly independent on an interval  $I$  if the only constants for which:

$\alpha_1 f_1(x) + \alpha_2 f_2(x) + \dots + \alpha_n f_n(x) = 0$  for every  $x$  in the interval  $I$  are  $\alpha_1 = \alpha_2 = \dots = \alpha_n = 0$  [5].

Let  $a, b \in \mathbb{R}, T > 0, I = [0, T], 0 < r \leq T$  and  $f: I \times \mathbb{R}^3 \rightarrow \mathbb{R}$  be continuous function. Then the BVP under investigation is given by.

$$y''(t) = f(t, y(t), y(t-r), y'(t)), t \in I \quad (7)$$

$$y(t) = a, t \in [-r, 0] \quad (8)$$

$$y(T) + y'(T) = b \quad (9)$$

**Definition 5:** A function  $u: [-r, T] \rightarrow \mathbb{R}$  is said to be a solution of BVP(7), (8), and (9) if:

1.  $u \in C[-r, T]$  and  $u \in C^2[0, T]$ .
2.  $u''(t) = f(t, u(t), u(t-r), u'(t)), t \in I$ .
3.  $u(t) = a, t \in [-r, 0]$ .
4.  $u(T) + u'(T) = b$ .

We need the following lemma, proof of which is straightforward and hence omitted [3].

**Definition 6:** Consider the second-order linear differential operator:

$$L_2 = a_0(x) \frac{d^2}{dx^2} + a_1(x) \frac{d}{dx} + a_2(x) \quad (10)$$

Assume that  $a_k$  ( $k = 0,1,2$ ) are all continuous and have continuous  $(n-k)$ th derivatives on an interval  $a \leq x \leq b$  and that  $a_0(x) \neq 0$  on  $a \leq x \leq b$ . The corresponding second-order linear differential equation is:

$$L_2 y = a_0(x) \frac{d^2 y}{dx^2} + a_1(x) \frac{dy}{dx} + a_2(x) y = 0 \quad (11)$$

The adjoint equation to equation (11) is:

$$\overline{L}_2 y = (-1)^2 \frac{d^2}{dx^2} [a_0(x)y] + (-1)^1 \frac{d}{dx} [a_1(x)y] + a_2(x)y = 0 \quad (12)$$

This reduces to:

$$\overline{L}_2 y = a_0(x) \frac{d^2 y}{dx^2} + [2a_0'(x) - a_1(x)] \frac{dy}{dx} + [a_0''(x) - a_1'(x) + a_2(x)] y = 0 \quad (13)$$

Where the primes denotes differentiation w.r.t.  $x$ . Thus the adjoint equation to the second-order equation (11) is equation (12). If  $L_2 = \overline{L}_2$  the second-order differential operator is called self-adjoint operator [4].

**Theorem 1:** The sufficient and necessary condition to make the equation (5) self-adjoint is:

$$\frac{d}{dx} [a_0(x)] = a_1(x) \text{ on the interval } a \leq x \leq b$$

**Proof:** If  $\frac{d}{dx} [a_0(x)] = a_1(x)$  satisfies then we get:

$$2a_0'(x) - a_1(x) = a_1(x) \quad . \quad a_0''(x) - a_1'(x) + a_2(x) = a_2(x)$$

Substitute in equation (12) we get:

$$\overline{L}_2 y = a_0(x) \frac{d^2 y}{dx^2} + a_1(x) \frac{dy}{dx} + a_2(x) y = 0 \quad \text{then } L_2 = \overline{L}_2$$

That is equation (11) is self-adjoint.

Revers that if the equation (11) self-adjoint we get the following:

$$2a_0'(x) - a_1(x) = a_1(x);$$

$$a_0''(x) - a_1'(x) + a_2(x) = a_2(x).$$

From the second equation we get  $a_0'(x) = a_1(x) + c$  where  $c$  is arbitrary constant. and from the first equation we get  $a_0'(x) = a_1(x)$  that is  $c = 0$ . thus the sufficient and necessary condition to make the equation (5) self-adjoint is  $\frac{d}{dx} [a_0(x)] = a_1(x)$  [4].

**Lemma 1:** Denote by  $A$  the matrix of the finite difference method for  $q \geq 0$  and by  $A_0$  the corresponding matrix for  $q = 0$ .

$$A_0 = \begin{bmatrix} 2 & -1 & 0 & . & . & 0 \\ -1 & 2 & -1 & 0 & . & 0 \\ 0 & -1 & 2 & -1 & . & . \\ . & . & . & . & . & . \\ 0 & . & . & -1 & 2 & -1 \\ 0 & . & . & . & -1 & 2 \end{bmatrix}$$

Then  $0 \leq A^{-1} \leq A_0^{-1}$ .

That is. All the components of  $A^{-1}$  are nonnegative and smaller than or equal to the corresponding components of  $A_0^{-1}$ .

**Proof:** The columns of the inverse  $A^{-1} = (a_1, a_2, \dots, a_n)$  satisfy  $Aa_j = e_j$  for  $j = 1, \dots, n$  with the canonical unit vectors  $e_1, e_2, \dots, e_n$  in  $\mathbb{R}^n$ . The Jacobi iterations for the solution of  $Az = e_j$  starting with  $z_0 = 0$  are given by:

$$z_{n+1} = -D^{-1}(A_L + A_R)z_n + D^{-1}e_j. n = 0,1, \dots$$

With the usual splitting  $A = D + A_L + A_R$  of  $A$  into its diagonal, lower and upper triangular parts. Since the entries of  $D^{-1}$  and  $-D^{-1}(A_L + A_R)$  are all nonnegative, it follows that  $A^{-1} \geq 0$ . Analogously, the iterations:

$$z_{n+1} = -D^{-1}(A_L + A_R)z_n + D^{-1}e_j, n = 0, 1, \dots$$

Yields the columns of  $A_0^{-1}$ . Therefore, from  $D_0^{-1} \geq D^{-1}$  we conclude that:

$$A_0^{-1} \geq A^{-1} [5].$$

**Corollary 1:** Let  $\mu_1 > 0, \mu_2 > 0, \mu_3 > 0, \mu = \max\{\mu_1 + |\theta|, \mu_2, \mu_3 + |\gamma|\}$  such that  $6\mu T(2T + 1) + 9\mu T < 1$ . Let  $f: I \times \mathbb{R}^3 \rightarrow \mathbb{R}$  be continuous and satisfy  $|f(t, x_1, y_1, z_1) - f(t, x_2, y_2, z_2)| \leq \mu_1|x_1 - x_2| + \mu_2|y_1 - y_2| + \mu_3|z_1 - z_2|$   $\forall (t, x_1, y_1, z_1), (t, x_2, y_2, z_2) \in I \times \mathbb{R}^3$ . Then the BVP (7), (8) and (9) has a unique solution.

**Proof.** Let  $F^*(t, u, v, w) = f(t, u, v, w) - \theta u - \gamma w, \forall (t, u, v, w) \in I \times \mathbb{R}^3$ . Consider the BVP given by.

$$y''(t) = F^*(t, y(t), y(t-r), y'(t)), t \in I \quad (14)$$

$$y(t) = a, t \in [-r, 0]$$

$$y(T) + y'(T) = b.$$

Let  $(t, u_1, v_1, w_1), (t, u_2, v_2, w_2) \in I \times \mathbb{R}^3$  be arbitrary. Then

$$\begin{aligned} & |F^*(t, u_1, v_1, w_1) - F^*(t, u_2, v_2, w_2)| \\ &= |f(t, u_1, v_1, w_1) - \theta u_1 - \gamma w_1 - f(t, u_2, v_2, w_2) + \theta u_2 + \gamma w_2| \\ &\leq |f(t, u_1, v_1, w_1) - f(t, u_2, v_2, w_2)| + |\theta||u_1 - u_2| + |\gamma||w_1 - w_2| \\ &\leq (\mu_1 + |\theta|)|u_1 - u_2| + \mu_2|v_1 - v_2| + (\mu_3 + |\gamma|)|w_1 - w_2| \end{aligned}$$

Therefore, by applying Corollary 1 we conclude that the BVP (7), (8) and (9) has a unique solution  $\mathcal{Y}$ . So we have:

$$y''(t) = f(t, y(t), y(t-r), y'(t)) - \theta y(t) - \gamma y'(t), t \in I.$$

$$y(t) = a, t \in [-r, 0].$$

$$y(T) + y'(T) = b.$$

Hence  $\mathcal{Y}$  is the unique solution of the BVP (7), (8) and (9) [3].

Applicable to Boundary value problems for ordinary differential equations.

Yields

A numerical approximation of the solution.

Idea

Using finite differences, the solution to a boundary value problem is determined (simultaneously) everywhere on the interval of interest.

Procedure

We will illustrate the procedure on the general second order linear ordinary differential equation. The same technique can be used, with only slight modifications, to systems of higher order ordinary differential equations, with the boundary data given virtually anywhere in the interval of interest.

Given the second order linear ordinary differential equation:

$$a(x)y'' + b(x)y' + c(x)y = d(x)$$

$$y(x_L) = y_L, y(x_U) = y_U$$

We introduce the variable  $z(x) = y'(x)$  and write equation (15) as the system:

$$\frac{d}{dx} \begin{bmatrix} y \\ z \end{bmatrix} = \begin{bmatrix} z \\ d - cy - bz \end{bmatrix} \quad (16)$$

Now, we choose a grid, not necessarily uniform, on the interval  $(x_L, x_U)$  say  $x_L = x_1 < x_2 < \dots < x_N = x_U$ . At each one of the grid points, some finite difference scheme is chosen to approximate the equations in equation (16). The scheme used can vary from point to point. For instance, if Euler's method is used for every point. Then:

$$\begin{bmatrix} y \\ z \end{bmatrix}_{k+1} = \begin{bmatrix} y \\ z \end{bmatrix}_k + (x_{k+1} - x_k) \begin{bmatrix} z \\ d - cy - bz \end{bmatrix}_k \quad (17)$$

To first order, where  $y_k = y(x_k)$ ,  $z_k = z(x_k)$  and similarly for  $\{a_k, b_k, c_k, d_k\}$ . From equation (15). The values  $y_1 = y_L$  and  $y_N = y_U$  are known.

To determine all of the  $\{z_k\}$  and the remaining  $\{y_k\}$  all of the relations in equation (17) (i.e., for  $k = 1, 2, \dots, N$ ) should be combined into one large matrix equation. First, for ease of notation. Define  $h_k = x_{k+1} - x_k$ ,  $e_k = d_k/a_k$ ,  $f_k = c_k/a_k$  and  $g_k = b_k/a_k$ . In these new variables, equation (17) may be written as:

$$\begin{aligned} y_{k+1} &= y_k + h_k z_k \\ z_{k+1} &= z_k + h_k (e_k - f_k y_k - g_k z_k) \end{aligned} \quad (18)$$

Combining all of the equations in (18) results in:

$$\begin{bmatrix} 1 & h_1 & -1 & 0 & 0 & 0 & \dots \\ h_1 f_1 & -1 + h_1 g_1 & 0 & 1 & 0 & 0 & \dots \\ 0 & 0 & 1 & h_2 & -1 & 0 & \dots \\ 0 & 0 & h_2 f_2 & -1 + h_2 g_2 & 0 & 1 & \dots \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \ddots \end{bmatrix} \begin{bmatrix} y_1 \\ z_1 \\ y_2 \\ z_2 \\ y_3 \\ z_3 \\ \vdots \\ z_N \end{bmatrix} = \begin{bmatrix} 0 \\ h_1 e_1 \\ 0 \\ h_2 e_2 \\ 0 \\ h_2 e_2 \\ \vdots \\ h_N e_N \end{bmatrix} \quad (19)$$

To this matrix equation should be added two more rows, one corresponding to  $y_1 = y_L$  and one corresponding to  $y_N = y_U$ . With these two rows, there results an  $2N \times 2N$  matrix equation. This equation can be solved to determine a numerical approximation to the solution at all of the grid points.

### Example 1:

The second order linear ordinary differential equation:

$$\begin{aligned} y'' + y &= 3 \\ y(0) &= 3, y\left(\frac{\pi}{2}\right) = 2 \end{aligned} \quad (20)$$

Has the solution  $y = 3 - \sin x$ . We use the box method to numerically approximate this solution. Writing equation (20) as a system results in:

$$\frac{d}{dx} \begin{bmatrix} y \\ z \end{bmatrix} = \begin{bmatrix} z \\ 3 - y \end{bmatrix} \quad (21)$$

We choose a uniform grid:  $x_n = (n-1)h$  for  $n = 1, 2, 3, 4$  with  $h = \pi/6$ . Defining  $y_n = y(x_n)$  and  $z_n = z(x_n)$  and using Euler's method, equation (21) may be approximated as:

$$\begin{aligned} y_{n+1} &= y_n + h z_n \\ z_{n+1} &= z_n + h(3 - y_n) \end{aligned} \quad (22)$$

Combining all of the equations in equation (22) for  $n = 1, 2, 3, 4$  results in:

$$\begin{bmatrix} 1 & h & -1 & 0 & 0 & 0 & 0 & 0 \\ h & -1 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & h & -1 & 0 & 0 & 0 \\ 0 & 0 & h & -1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & h & -1 & 0 \\ 0 & 0 & 0 & 0 & h & -1 & 0 & 1 \end{bmatrix} \begin{bmatrix} y_1 \\ z_1 \\ y_2 \\ z_2 \\ y_3 \\ z_3 \\ y_4 \\ z_4 \end{bmatrix} = \begin{bmatrix} 0 \\ 3h \\ 0 \\ 3h \\ 0 \\ 3h \\ 0 \\ 3h \end{bmatrix} \quad (23)$$

Then the following two rows are added, to incorporate the known values of  $y(0)$  and  $y(\pi/2)$

$$\begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} y_1 \\ z_1 \\ y_2 \\ z_2 \\ y_3 \\ z_3 \\ y_4 \\ z_4 \end{bmatrix} = \begin{bmatrix} 3 \\ 2 \end{bmatrix} \quad (24)$$

The Fortran program numerically approximates the solution to the above equation. Note that this program uses a linear equation solver LSOLVE, whose source code is not listed. The output of the program is:

Here is the approximate solution:

3.000 -0.701 2.633 -0.701 2.266 -0.509 2.000 -0.124

Here is the exact solution

3.000 -1.000 2.500 -0.866 2.134 -0.500 2.000 0.000

The values for  $y_n$  are only accurate to one decimal place in this example. Putting more points in the interval would decrease the error, as would using a higher order method in place of Euler's method.

### Some Notes

1. In our example, if the two rows corresponding to the boundary terms were added to the matrix equation at the correct locations, the resulting matrix would be banded.
2. This technique is recommended for stiff boundary value problems because many points can be added where the solution undergoes large changes and different discretization schemes may be used in different regions.
3. For nonlinear equations or nonlinear boundary conditions, this method can be used iteratively by linearizing the nonlinear terms at each step.

```

DIMENSION ARRAY(8.18),SOLN(8),RHS(8),NROW(100)
PI=3.1415926
NPOINT=8
H=PI/2.*2./FLOAT(NPOINT-2)
DO 10 J=1,NPOINT
DO 10 K=1,NPOINT
10 ARRAY(J,K)=0.0
C Create the matrix
ARRAY(1,1)=1.0
RHS(1)=3.0
ARRAY(NPOINT,NPOINT-1)=1.0
RHS(NPOINT)=2.0
J=1
20 J=J+1
IF( J .GE. NPOINT ) GOTO 30
C Here is the Y-equation
ARRAY(J,J-1)=1
ARRAY(J,J)=H
ARRAY(J,J+1)=-1
RHS(J)=0
J=J+1
C Here is the Z-equation
ARRAY(J,J-2)=H
ARRAY(J,J-1)=1
ARRAY(J,J+1)=1
RHS(J)=3.0*H
GOTO 20
C Solve the matrix system
30 CALL LSOLVE(NPOINT,ARRAY,SOLN,RHS,NROW,IFSING,NPOINT)
WRITE(6,5) (SOLN(J),J=1,NPOINT)
5 FORMAT(' Here is the approximate solution:'.//8(1x.F8.3) )
C Compute the exact solution for comparison
J=1
DO 40 JJ=1,NPOINT/2
SOLN(J)=3.0-SIN( H*FLOAT(JJ-1) )
SOLN(J+1)= -COS( H*FLOAT(JJ-1) )
40 J=J+2
WRITE(6,15) (SOLN(J),J=1,NPOINT)
15 FORMAT(' Here is the exact solution:'.//8(1x.F8.3) )
END

```

Fortran program for box method <sup>[1]</sup>.

## Numerical Solution

**Table 1:** Presents the comparison between the exact solution  $y(x)$  and the finite difference numerical solution  $y_{\_num}$  for  $N = 10$  subintervals ( $h = 0.1$ )

| Node (x <sub>i</sub> ) | Exact Solution y(x) | Numerical Solution y <sub>_num</sub> | Absolute Error  y <sub>_ex</sub> - y <sub>_num</sub> |
|------------------------|---------------------|--------------------------------------|------------------------------------------------------|
| 0.0                    | 0.000000            | 0.000000                             | 0.000000e+00                                         |
| 0.1                    | -0.014766           | -0.014755                            | 1.098519e-05                                         |
| 0.2                    | -0.028680           | -0.028658                            | 2.136971e-05                                         |
| 0.3                    | -0.040878           | -0.040848                            | 3.053978e-05                                         |
| 0.4                    | -0.050483           | -0.050446                            | 3.785518e-05                                         |
| 0.5                    | -0.056591           | -0.056548                            | 4.263553e-05                                         |
| 0.6                    | -0.058260           | -0.058216                            | 4.414591e-05                                         |
| 0.7                    | -0.054507           | -0.054466                            | 4.158176e-05                                         |
| 0.8                    | -0.044295           | -0.044260                            | 3.405252e-05                                         |
| 0.9                    | -0.026518           | -0.026498                            | 2.056416e-05                                         |
| 1.0                    | 0.000000            | 0.000000                             | 0.000000e+00                                         |

## MATLAB Code

```

% MATLAB Code to solve BVP: y'' = y + x. y(0)=0. y(1)=0
% Method: Finite Difference Method (FDM) vs Exact Solution

clear; clc; close all;

% Parameters
N = 10;           % Number of subintervals
x0 = 0; xL = 1;   % Domain boundaries
h = (xL - x0) / N; % Step size
x = linspace(x0, xL, N+1)'; % Mesh points

```

```

% Exact Solution
y_exact = sinh(x) / sinh(1) - x;

% Finite Difference System Construction: A*y = b
% y'' - y = x ==> (y_{i+1} - 2y_i + y_{i-1})/h^2 - y_i = x_i
% ==> y_{i-1} - (2 + h^2)*y_i + y_{i+1} = h^2 * x_i

A = zeros(N+1, N+1);
b = zeros(N+1, 1);

% Boundary Conditions
A(1, 1) = 1; b(1) = 0; % y(0) = 0
A(N+1, N+1) = 1; b(N+1) = 0; % y(1) = 0

% Internal Mesh Points
for i = 2:N
    A(i, i-1) = 1;
    A(i, i) = -(2 + h^2);
    A(i, i+1) = 1;
    b(i) = (h^2) * x(i);
end

% Solve Linear System
y_num = A \ b;

% Absolute Point-wise Error
abs_error = abs(y_exact - y_num);

% Display Table of Results
T = table(x, y_exact, y_num, abs_error, ...
    'VariableNames', {'x', 'Exact_Solution', 'Numerical_Solution', 'Absolute_Error'});
disp(T);

%% Plotting (All labels in English)
figure('Color', 'w');

% Subplot 1: Comparison of Solutions
subplot(2,1,1);
plot(x, y_exact, 'b-', 'LineWidth', 2, 'DisplayName', 'Exact Solution');
hold on;
plot(x, y_num, 'ro--', 'LineWidth', 1.5, 'MarkerSize', 6, 'DisplayName', 'Numerical Solution (FDM)');
grid on;
title('Comparison between Exact and Numerical Solutions', 'FontSize', 12);
xlabel('x', 'FontSize', 11);
ylabel('y(x)', 'FontSize', 11);
legend('Location', 'best');

% Subplot 2: Point-wise Absolute Error
subplot(2,1,2);
plot(x, abs_error, 'k-s', 'LineWidth', 1.5, 'MarkerFaceColor', 'g', 'MarkerSize', 5);
grid on;
title('Point-wise Absolute Error', 'FontSize', 12);
xlabel('x', 'FontSize', 11);
ylabel('Absolute Error |y_{exact} - y_{num}|', 'FontSize', 11);

% Save Figure
saveas(gcf, 'BVP_Solution_Plot.png');

```

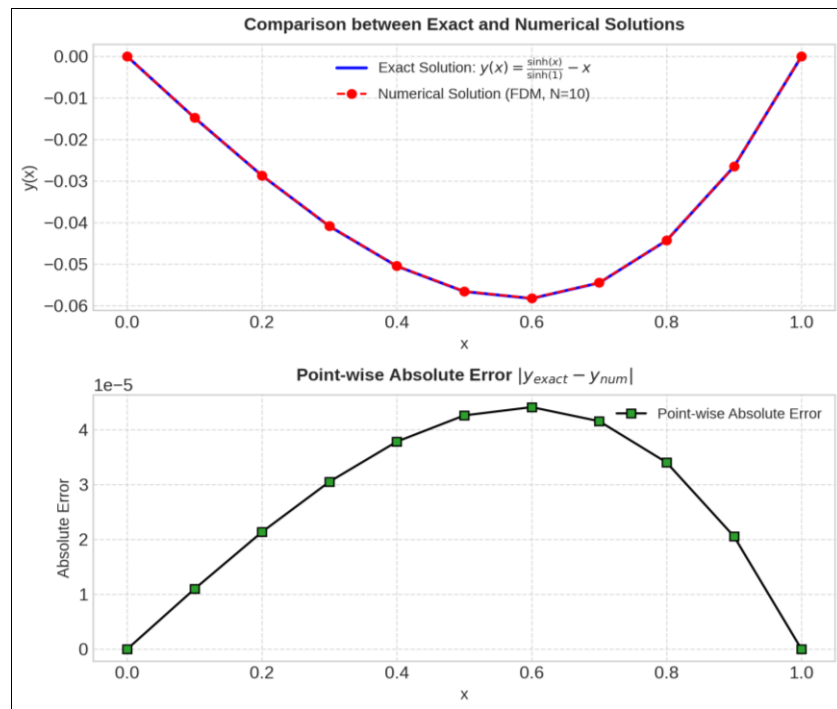


**Fig 1:** Comparison between Exact and Numerical Solutions (Top) and Point-wise Absolute Error (Bottom)

**Example 2:** Consider the boundary value problem:

$$y'' = x + y$$

With the boundary conditions  $y(0) = 0, y(1) = 0$ . The exact solution for this problem is  $y(x) = \frac{\sinh x}{\sinh 1} - x$ . The numerical solutions in terms of point wise absolute errors by comparing with the exact solutions are given in Table 1.

**Table 1:** The comparison of approximate solution with exact solution for Example 1 at different values of the step size  $h$

| Mesh size $h$ | $x_j$ | Computed value | Exact value | Error   |
|---------------|-------|----------------|-------------|---------|
| 0.25          | 0.25  | -0.03488       | -0.03505    | 0.00017 |
|               | 0.5   | -0.05632       | -0.05659    | 0.00027 |
|               | 0.75  | -0.05003       | -0.05028    | 0.00025 |
| 0.2           | 0.2   | -0.02855       | -0.02679    | 0.00176 |
|               | 0.4   | -0.05028       | -0.05048    | 0.00034 |
|               | 0.6   | -0.05804       | -0.05826    | 0.00022 |
|               | 0.8   | -0.04414       | -0.04429    | 0.00015 |

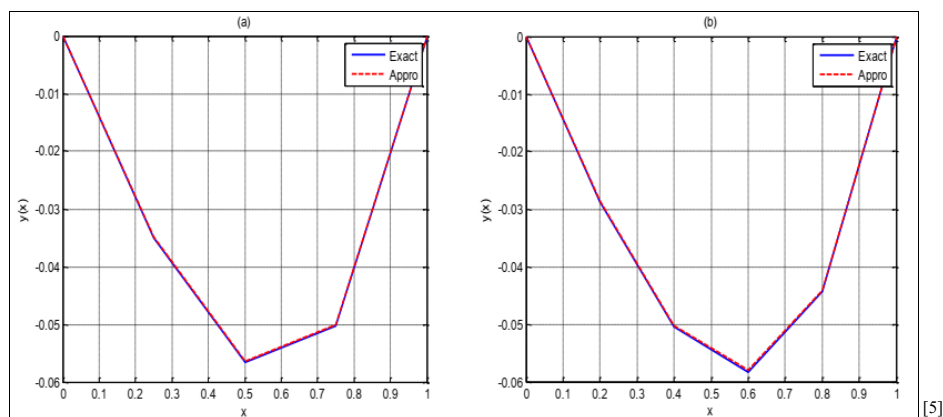


**Fig 2:** Comparison of the numerical with the exact solution for Example 1 at (a)  $h = 0.25$  and (b)  $h = 0.2$

**Method 1: Symbolic Solution (dsolve)**

```

% Clear workspace and command window
clc; clear; close all;

% Define symbolic variables
syms y(x)

% Define ODE and boundary conditions
ODE = diff(y, x, 2) + y == 3;
cond1 = y(0) == 3;
cond2 = y(pi/2) == 2;
conds = [cond1, cond2];

% Solve ODE symbolically
ySol(x) = dsolve(ODE, conds);

% Display exact solution
disp('Analytical Solution y(x):');
disp(ySol(x));

% Plot solution
figure;
fplot(ySol, [0, pi], 'LineWidth', 2, 'Color', 'b');
grid on;
title('Exact Solution of  $y'' + y = 3$ ');
xlabel('x');
ylabel('y(x)');

```

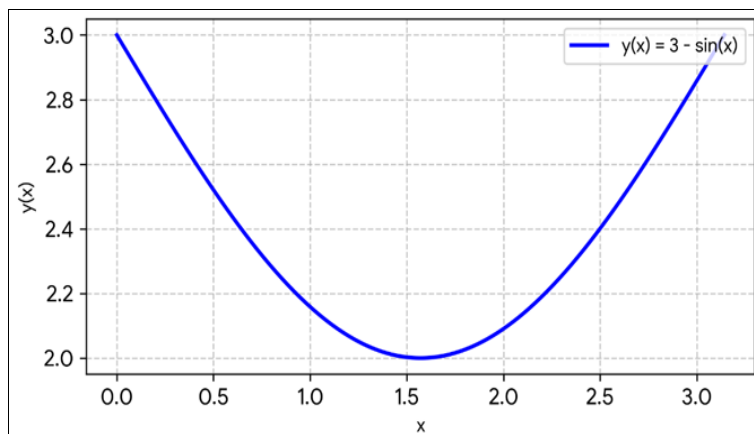


**Fig 3:** Exact solution of  $y'' + y = 3$

**Method 2: Numerical Solution (bvp4c)**

```

clc; clear; close all;
% Define initial mesh and initial guess
xmesh = linspace(0, pi/2, 100);
solinit = bvpinit(xmesh, [0 0]);

% Solve BVP using bvp4c
sol = bvp4c(@bvpfcn, @bcfcn, solinit);

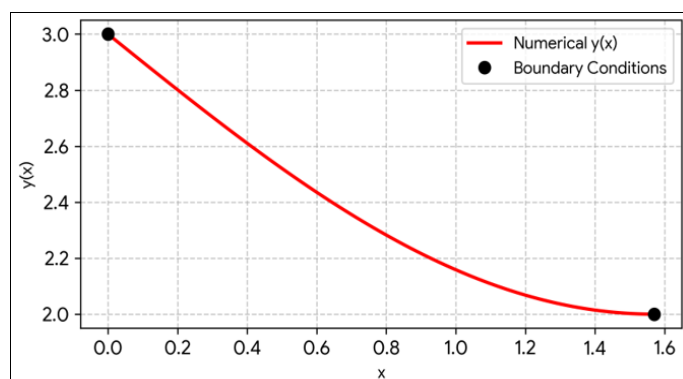
% Evaluate and plot numerical solution
x = linspace(0, pi/2, 100);
y = deval(sol, x);

figure;
plot(x, y(1,:), 'r-', 'LineWidth', 2);
hold on;
plot([0, pi/2], [3, 2], 'ko', 'MarkerFaceColor', 'k', 'MarkerSize', 8);
grid on;
title('Numerical Solution using bvp4c');
xlabel('x');
ylabel('y(x)');
legend('y(x)', 'Boundary Conditions');

% --- Helper Functions ---
function dydx = bvpfcn(x, y)
dydx = zeros(2,1);
dydx(1) = y(2);    % y1' = y2
dydx(2) = 3 - y(1); % y2' = 3 - y1
end

function res = bcfcn(ya, yb)
res = [ya(1) - 3;    % y(0) = 3
yb(1) - 2]; % y(pi/2) = 2
end

```

**Fig 4:** Numerical solution using bvp4c

## Results

1. The numerical results that are obtained using MATLAB converges to the exact solutions strictly.
2. The amount of computation time required may become excessive, and good accuracy may be difficult to achieve.
3. MATLAB automatically adjusts and increases mesh density in regions where second – order solutions experience rapid variations, sharp curvatures, or steep boundary layers.

## Recommendations

1. For complex engineering or physical models involving unknown parameters alongside the differential equation, MATLAB's unknown parameter vector capability should be utilized within the boundary condition function to simultaneously solve for both the profile and the parameter.
2. When dealing with singular BVPs at the domain boundaries (common in radial coordinate transformations), explicit analytical asymptotic expansions near the boundaries should be incorporated to avoid numerical breakdown and division-by-zero errors in the MATLAB routines.
3. Numerical solutions obtained via MATLAB solvers should be systematically benchmarked against known exact analytical solutions or highly established asymptotic approximations for simplified linear cases before extending the code to complex, non-solvable domains.

## References

1. Daniel Zwillinger. Handbook of Differential Equations 3rd edition, Academic Press, 1997.
2. Mujeeb Ur Rehman, Rahmat Ali Khan. A numerical method for solving boundary value problems for fractional differential equations, elsevier, 2012.
3. Heramb Aiya, Yeshwant Shivrai Valaulikar. On Boundary Value Problems for Second Order Ordinary Differential Equation with Constant Delay, Serdica Mathematical Journal, 2025.
4. م.م. ولدان وليد محمود. On Linear Ordinary Differential Equation with Orthogonal Polynomial Solutions. Journal of College of Education, University of Wasit.
5. Biruk Endeshaw. Finite Difference Method for Boundary Value Problems in Ordinary Differential Equations, IISTE, 2019.
6. Ravi Agarwal, Mouffak Benchohra. Boundary value problems for fractional differential equations, Researchgate, 2009.
7. Rajnish Kumar Jha. General Solution of Second Order Linear Ordinary Differential Equation, arXiv, 2021.
8. Akalu Abriham Anulo. Numerical Solution of Linear Second Order Ordinary Differential Equations with Mixed Boundary Conditions by Galerkin Method, Science Publishing Group, 2017.