Laguerre and Hermite Polynomials based Galerkin Approach for Second-Order Linear ODE for Varied BCs using Shooting Method

Pavitra
Department of Basic & Applied Science
Bhagat Phool Singh Mahila Vishwavidyalaya
Khanpur Kalan, Sonipat, India
Pavitradhanda10@gmail.com

ABSTRACT
We proposed an algorithm for the finest approximating solutions of second-order ordinary linear differential equations based on the Galerkin technique by using Laguerre and Hermite polynomials. The approach is to convert Dirichlet or mixed BCs, using the shooting method has been used in conjuction with the secant and Runge-Kutta method. Accuracy and efficiency are dependent on the size of the set of polynomials and the procedure in our case is simpler as compared to the methods such as spline and Bernstein polynomials for solving differential equations. The accuracy of the three test problems is testified through $\mathbf{L_2}$ and $\mathbf{L_\infty}$ norms, wherein solutions obtained using Hermite polynomials are better than Laguerre and as such better than the solution obtained by any other numerical techniques. The visibility of solutions is depicted through tables and graphs.

Keywords − Hermite Polynomial; Laguerre Polynomial; Shooting Method; Secant Method; Runge-Kutta Method; Second Order Linear ODE; Galerkin Method

I. INTRODUCTION

Finding an approximate solution to some genuine physical issues by the use of various numerical techniques is the aim of numerical analysis, particularly in situations when analytical answers are either impossible to find or extremely difficult to achieve, an entire solution to the boundary governing equation. One may specify an initial value or a boundary value for the conditions.

Two-point Boundary Value Problems (BVPs) are a common way to define many scientific and engineering problems. Examples include mechanical vibration analysis and spring vibration. This demonstrates the critical importance that numerical techniques for approximating Two-point BVP solutions play across all scientific and engineering domains. The shooting method, finite difference method, finite element method, variational method (Weighted residual methods, Ritz method), and other numerical techniques have been used to solve the two-point boundary value problems. These techniques are among the various approaches used to approximate two-point BVPs in terms of differential equations.

The primary efforts in both variational and finite element approaches were to examine an approximate solution as a linear combination of appropriate approximation functions and unknown coefficients (2). The Laplace decomposition method was used in (3) to solve second-order differential equations using Bernstein polynomials. In (6), a parametric cubic spline solution of two-point BVPs was obtained. In (7), a Galerkin method with cubic B-Splines was used to solve fourth-order BVPs by taking into account various cases on the boundary condition. The numerical solution of second-order ODEs with Galerkin, Petrov-Galerkin, Collocation, Least-square method (9),(10),(11), and the numerical solution of RLW equation using quadratic B-Splines (12).

The weighted residual method was the most widely used approach for Galerkin; in this study, we employ the shooting method to compute the Neumann boundary conditions, incorporating the secant method as well. With the boundary condition based on the Laguerre and Hermite polynomials basis, we employ the Galerkin method methodology to provide numerical solutions for the second-order linear ordinary differential equation. The formulation is derived. In this work, two sorts of boundary conditions are currently being considered: Dirichlet boundary condition(first kind) and mixed boundary condition(third kind). The structure of this document is as follows.

We go over the fundamental ideas of the Galerkin technique in Section II. The development of the Galerkin approach is detailed in Section III, which also contains the primary results. Several numerical results and comments are provided in Section IV. Section V has the conclusion.

II. GALERKIN METHOD

Russian mathematician Boris Grigoryevich Galerkin created the Galerkin technique in 1915. The method's inception is typically linked to a 1915 paper that Galerkin wrote regarding the elastic equilibrium of rods and thin plates. Integral equations, partial differential equations, and ordinary differential equations can all have their solutions approximated using the Galerkin method.

One of the weighted residuals methods is the Galerkin Method. The primary goals of both FEM and variable techniques were to find an approximate solution in the form of a linear combination of appropriate approximations and unknown coefficients. For a vector space of functions V, if $S = \{\psi_i(x)\}_{i=1}^{\infty}$ be the basis of V, a set of linearly independent functions, any function $f(x) \in V$ could be uniquely written as a linear combination of basis as:

$$f(x) = \sum_{j=1}^{\infty} c_j \, \psi_j \tag{1}$$

Assume that the differential equation's approximate solution $D(u) = L\big(u(x)\big) + f(x) = 0$, on the boundary $B(u) = [a, b]$ is in the form:

$$u(x) \approx U_N(x) = \sum_{j=1}^{N} c_j \, \psi_j(x) + \psi_0(x) \tag{2}$$

Where L is a differential operator, f is a given function, $\psi_j(x)'s$ are finite number of basis functions, and c_j are unknown coefficients for $j = 1,2,3, \dots, N$. $U_N(x)$ is the approximate solution, while $u(x)$ is the precise answer. **Vichnevetsky** was the one who first introduced the phrase "weighted residuals method" (11). Therefore, the weighted residual approaches are shown using the generalized inner product that follows:

$$\int_a^b w_i(x) R\big(x, c_j\big) \, dx = 0 \tag{3}$$

Where, the method known as the weighted-residual method uses $R\big(x, c_j\big) = D\big(U_N(x)\big) - \Big(L\big(U_N(x)\big)\Big) + f(x)$ and $w_i(x)$ as a collection of linearly independent functions. These weight functions can differ from the approximate functions ψ_j in general.

The weighted-residual method's particular name is called the Galerkin method if, as in equation (3), $\psi_j(x) = w_i(x)$. The Galerkin technique, which finds the approximate solution of two-point boundary value problems, is one of the weighted residual methods in which the approximate function is the same as the weighted function.

III. PROPOSED METHOD

This section covers the Galerkin method's application to second-order linear differential equations, as well as the shooting method's use of secant and Runge-Kutta methods to convert Dirichlet or mixed boundary conditions to Neumann boundary conditions.

Consider a second-order linear differential equation of the form

$$\frac{-d}{dx}\left(p(x) \frac{du}{dx}\right) + q(x)u - r(x) = 0, a \le x \le b \tag{4}$$

With the boundary conditions,

$$\alpha_0 u(a) + \alpha_1 u'(a) = c_1 \tag{5}$$

$$\beta_0 u(b) + \beta_1 u'(b) = c_2 \tag{6}$$

Where $\alpha_0, \beta_0, \alpha_1, \beta_1, c_1, c_2$ are constant and $p(x), q(x), r(x)$ are continuous functions.

The solution of the differential equation $(4 - 6)$ is approximated as

$$u(x) \approx U_N(x) = \sum_{i=1}^{N} c_i \, \psi_i, N \ge 1 \tag{7}$$

Substituting (1) into (4), the Galerkin weighted residual equations are:

$$\int_a^b \left[\frac{-d}{dx}\left(p(x)\frac{dU_N}{dx}\right) + q(x)U_N(x) - r(x)\right]\psi_i(x)dx = 0 \tag{8}$$

Simplifying, we obtain

$$\sum_{j=1}^{N}\left[\left[\int_a^b p(x)\frac{d\psi_i}{dx}\frac{d\psi_j}{dx} + q(x)\psi_i(x)\psi_j(x)\right]dx = \int_a^b r(x)\psi_i(x)\,dx + \psi_i(b)p(b)U'_N(b) - \psi_i(a)p(a)U'_N(a)\right.$$

Or in matrix notations,

$$\sum_{j=1}^{N} K_{ij}\,c_j = F_i \tag{9}$$

Where, $K_{ij} = \int_a^b \left[p(x)\frac{d\psi_i}{dx}\frac{d\psi_j}{dx} + q(x)\psi_i(x)\psi_j(x)\right]dx$

$$F_i = \int_a^b r(x)\psi_i(x)\,dx + \psi_i(b)p(b)U'_N(b) - \psi_i(a)p(a)U'_N(a)$$

K_{ij} gives the stiffness matrix, we obtain the values of the parameters $c_i's$ by solving the system(9) and then substitute into (7) to get the approximate solution $U_N(x)$ of the desired BVP $(4-6)$.
The values of $U'_N(a)$ and $U'_N(b)$, which are roughly equal to $u'(a)$ and $u'(b)$, respectively— u being the exact solution of the BVP—must be known in order to solve equation (9) above.
Consider BVP with
Mixed boundary condition:

$$u(a) = c_1, u'(b) = c_2 \tag{10}$$

And
Dirichlet boundary condition:

$$u(a) = c_1, u(b) = c_2 \tag{11}$$

Since $u'(a)$ is not supplied in the mixed type boundary condition and $u'(a)$ and $u'(b)$ are not given in the Dirichlet boundary condition, it is not possible to apply the aforementioned approach directly in this situation. The BVP must be transformed into a boundary value problem of the Neumann type. Several numerical techniques are used to do the conversion.

Consider solving the BVP

$$\frac{-d}{dx}\left(p(x)\frac{du}{dx}\right) + q(x)u - r(x) = 0, a \le x \le b \tag{12}$$

With Dirichlet boundary condition

$$u(a) = c_1, u(b) = c_2$$

In order to solve for $u'(a)$ and $u'(b)$, we must assume that $u(b) = c_2$. To determine $u'(a)$ such that $u(b) = c_2$, assume $u'(a) = u_n$ and use the R-K technique for second order to solve for $u(b)$. Denote the estimated solution u_{u_n} in the ODE after obtaining a value using the guess, and expect that $u_{u_n}(b) = c_2$. If not, try solving using the R-K technique with a different estimate for $u'(a)$. This procedure can be methodically repeated until the option fulfills $u(b)$.

The following algorithm will do this.
First, choose u_n such that $u_{u_n}(b) = c_2$. $\psi(u_n) = u_{u_n}(b) - c_2$.
The estimate for u_n
Step 2: Since the goal at this point is to just find $\psi(u_n) = 0$, the secant approach can be applied.
Step 3: u_n computation
Assume that guesses u_0 and u_1 yield the answers $u_{u_0}(b)$ and $u_{u_1}(b)$, respectively.
Step 4: Now, determine z_2 provided by utilizing the secant approach.

$$u_{k+1} = \frac{u_{k-1}\psi(u_k) - u_k\psi(u_{k-1})}{\psi(u_k) - \psi(u_{k-1})}, k = 1,2,3, \ldots$$

Following this sequence of iteration $\exists\, u_k$ such that

$u_{u_k}(b) = u(b)$ and $u'_{u_k}(b) = u'^{(b)} = \chi_1(say)$

Thus the Neumann condition

$$u'(a) = u_k$$
$$u'(b) = \chi_1$$

Conversion of the Domain of the BVP

An analogous BVP that is defined on $[0,1]$ must be created from the given BVP that is defined on the arbitrary interval $[a, b]$. Therefore, on $[0,1]$ the approximation polynomial is defined. It is feasible to apply the Hermite polynomial after transforming the BVP defined on the arbitrary interval $[a, b]$ into an equivalent BVP defined on $[0,1]$ because the Hermite polynomial is defined on $[0,1]$.

By letting $x = (b - a)x + a$, the BVP can be transformed into an equivalent issue on $[0,1]$. Equation (4) then corresponds to the BVP

$$\frac{-d}{dx}\left(\frac{1}{(b-a)}p_1(x)\frac{du}{dx}\right) + q_1(x)u - r_1(x) = 0, 0 \le x \le 1 \tag{13}$$

Subject to the boundary conditions

$$\alpha_0 u(0) + \frac{1}{b-a}\alpha_1 u'(0) = c_1 \tag{14}$$

$$\beta_0 u(1) + \frac{1}{b-a}\beta_1 u'(1) = c_2 \tag{15}$$

Where, $p_1(x) = p\big((b-a)x + a\big)$, $q_1(x) = q\big((b-a)x + a\big)$ and $r_1(x) = r\big((b-a)x + a\big)$.

IV. NUMERICAL RESULTS AND DISCUSSION

We presented three numerical experiments to demonstrate the flexibility of the numerical algorithm. The accuracy and efficiency of the method are tested by the normed error of the above Galerkin approach, L_2 and L_∞ error (15) measured based on the following formulae:

$$L_2 = \sqrt{\sum_{j=0}^{N}\left|u_{exact}(x_j) - u_{approx.}(x_j)\right|}$$

$$L_\infty = \max_{0\le j\le N}\left|u_{exact}(x_j) - u_{approx.}(x_j)\right| \tag{16}$$

The numerical outcomes are compared with the exact or approximate solutions. The results are reported in tables and figure where computations are carried out on MATLAB R2018a.

Problem 1: Consider a one-dimensional heat conduction/convection equation (4)

$$\frac{-d}{dx}\left(a\frac{du}{dx}\right) + cu = q; 0 < x < 1$$

$$u(0) = u_0, \left[a\frac{du}{dx} + \beta(u - u_\infty)\right]_{x=1} = Q_0 \text{ at } x = 1$$

Where a and q are functions of x, and β, c, u_∞ and Q_0 are constants.

Case 1 By taking, $a = 1, c = 1, u_0 = 1, Q_0 = \beta = 0$

$$\frac{-d^2u}{dx^2} + u = x^2; 0 < x < 1$$

Subject to the boundary condition

$$u(0) = 1, u'^{(1)} = 0$$

The exact solution is

$$u(x) = x^2 - \frac{e^x(2e+1)}{(e^2+1)} + \frac{e^{-x}(2e-e^2)}{(e^2+1)} + 2$$

The aforementioned problem requires the use of a mixed boundary condition. Let $u_0 = u'(0) = 0$ be the initial guess, and suppose that $u'(1) = 0$. Now, make a guess based on the value of $u'(1) = 0$. The second step is to solve the second-order differential equation (4) using the Runge-Kutta method, where $u''(x) = f(x, u, u')$. Therefore, $u''(x) = f(x, u)$ for this. Given that f is not reliant on u'.
Given that $x_0 = 0, x_n = 1$ and $u(0) = 1$ and take step size $h = 0.05, u'(0) = 0$.

R-K Method for the linear second-order ordinary differential equation:

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

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

Where
$K_1 = \frac{h^2}{2} f(x_i, u_i), K_2 = \frac{h^2}{2} f\left(x_i + \frac{2}{3}h, u_j + \frac{2}{3}hu'_j + \frac{4}{9}K_1\right)$, for $j = 0, 1, 2, 3, \ldots., 20$

This displays the outcome for the initial iteration in Table 2. Where in the ith step $x = x_i, u = u(x_i)$ and $u' = u'(x_i)$. Referring to table 2, take $u'_{u_0}(1) = 0.82480$. But $u'_{u_0}(1) \neq u'(1)$

$\psi(u_0) = u'_{u_0}(0) - 0 = 0.82480.$

Now we guess another value $u_1 = 1$. Referring to table 3, $u'_{u_1}(1) = 2.36787$.

$\psi(u_1) = u'_{u_1}(0) - 0 = 2.36787.$

Then find u_2 $(By\ the\ secant\ method)$

$$u_2 = \frac{u_0\psi(u_1) - u_1\psi(u_0)}{\psi(u_1) - \psi(u_0)} = -0.534518$$

Referring to table 4, $u'_{u_2}(1) = 0.00000,\ u'(0) \approx -0.534518$.

Thus, the Neumann boundary value problem is given by

$$\frac{-d^2u}{dx^2} + u = x^2; 0 \leq x \leq 1$$

$$u'(0) = -0.534518,\ u'(1) = 0. \tag{17}$$

Assume for the moment that U_N is the approximate solution to equation (17) provided by the linear combination of basis functions and unknown parameters.

Results have been shown for different values of x in Figure 6 showing the approximate solution with Hermite and Laguerre polynomials.

Problem 2: Consider the second-order linear ODE

$$\frac{d^2u}{dx^2} = u + x$$

With Dirichlet boundary condition

$$u(0) = 1, u(1) = 2$$

The exact solution is given by:

$$u(x) = \frac{3e - 1}{e^2 - 1}e^x + \frac{e(e - 3)}{e^2 - 1}e^{-x} - x$$

In order to use the aforementioned method, one must use the shooting method to change the provided boundary condition into a Neumann boundary condition.
Assume now that $u(1) = 2$ will determine the outcome. Let $u_0 = u'(0) = 0$ be the initial guess, and hope that $u(1) = u_{u_0}(1) = 2$. Applying the Runge-Kutta method to the second-order differential equation is the next step. Assuming $u(0) = 1, x_0 = 0, x_n = 1$, and step size $h = 0.05$.
This displays the outcome for the initial iteration in Table 5. Using Table 5 as a reference, $u_{u_0}(1) = 1.7183$.But $u_{u_0}(1) \neq u(1)$

$\psi(u_0) = u_{u_0}(1) - 2 = 1.7183 - 2 = -0.2817$

Now we guess another value $u_1 = 1$. Referring to Table 6, $u_{u_1}(1) = 2.8935$. But $u_{u_1}(1) \neq u(1)$

$\psi(u_1) = u_{u_1}(1) - 2 = 2.8935 - 2 = 0.8935$.

Then find $u_2(by\ secant\ method)$

$$u_2 = \frac{u_0\psi(u_1) - u_1\psi(u_0)}{\psi(u_1) - \psi(u_0)} = 0.2397$$

Referring to Table 7, $u_{u_2}(1) = 2.0000$

$\psi(u_2) = u_{u_2}(1) - 2 = 0.0000$
$\Rightarrow u'(0) \approx 0.2397$ and $u'(1) \approx 2.08812$

Thus the Neumann boundary value problem

$$\frac{d^2u}{dx^2} = u + x; 0 \leq x \leq 1$$

$$u'(0) = 0.23972, u'(1) = 2.08812 \tag{20}$$

Assume for the moment that U_N is the approximate solution to equation (20) provided by the linear combination of basis functions and unknown parameters.
Results have been shown for different values of x in Table 10 for $n = 4$ and $n = 6$. Also, Figure 7 and Figure 8 show the exact and approximate solution with Hermite and Laguerre polynomials.
In Problem 1 and Problem 2, we have given mixed and Dirichlet boundary conditions. According to our Galerkin approach, mixed and Dirichlet boundary condition needs to convert into Neumann boundary condition. A comparison table and graph have been shown for error analysis. After comparison, we see that Galerkin approach with Hermite basis function gives a better result than Laguerre basis functions. There is a drawback of this method with Laguerre basis function that, sometimes stiffness matrix close to singular as increases the degree of basis function, then does not work well.

In Table 1 the maximum error occurred in Problem 1 and Problem 2 with Laguerre basis function that Hermite basis functions.

Table 1: Computed L_∞-error and L_2-error

Problems	L_∞	L_2	L_∞	L_2
	(Laguerre poly.)	(Laguerre poly.)	(Hermite poly.)	(Hermite poly.)
Problem 1 (Case 1)	5×10^{-4}	2.5×10^{-7}	3×10^{-4}	1.2×10^{-7}
Problem 1 (Case 2)	8×10^{-4}	3.55×10^{-6}	1×10^{-4}	2.24×10^{-8}
Problem 2	9×10^{-4}	2.10×10^{-2}	4.6×10^{-3}	1.12×10^{-2}

Table 2

x	u	u'
0.00000	1.00000	0.00000
0.05000	1.00125	0.04998
0.10000	1.00500	0.09983
0.15000	1.01123	0.14944
0.20000	1.01993	0.19866
0.25000	1.03109	0.24739
0.30000	1.04466	0.29548
0.35000	1.06062	0.34281
0.40000	1.07893	0.38925
0.45000	1.09953	0.43466
0.50000	1.12237	0.47891
0.55000	1.14740	0.52185
0.60000	1.17453	0.56335
0.65000	1.20371	0.60325
0.70000	1.23483	0.64142
0.75000	1.26782	0.67768
0.80000	1.30256	0.71190
0.85000	1.33897	0.74389
0.90000	1.37691	0.77348
0.95000	1.41627	0.80052
1.00000	1.45692	0.82480

Table 3

x	u	u'
0.00000	1.00000	1.00000
0.05000	1.05127	1.05123
0.10000	1.10516	1.10484
0.15000	1.16179	1.16071
0.20000	1.22127	1.21873
0.25000	1.28370	1.27880
0.30000	1.34918	1.34082
0.35000	1.41781	1.40469
0.40000	1.48968	1.47032
0.45000	1.56487	1.53763
0.50000	1.64347	1.60653
0.55000	1.72555	1.67695
0.60000	1.81119	1.74881
0.65000	1.90045	1.82204
0.70000	1.99341	1.89658
0.75000	2.09013	1.97236
0.80000	2.19067	2.04933
0.85000	2.29508	2.12741
0.90000	2.40343	2.20657
0.95000	2.51576	2.28674
1.00000	2.63212	2.36787

Table 4

x	u	u'
0.00000	1.00000	-0.53452
0.05000	0.97451	-0.48521
0.10000	0.95145	-0.43736
0.15000	0.93075	-0.39111
0.20000	0.91232	-0.34658
0.25000	0.89606	-0.30392
0.30000	0.88189	-0.26327
0.35000	0.86970	-0.22478
0.40000	0.85937	-0.18860
0.45000	0.85080	-0.15490
0.50000	0.84384	-0.12383
0.55000	0.83837	-0.09557
0.60000	0.83423	-0.07030
0.65000	0.83128	-0.04821
0.70000	0.82935	-0.02949
0.75000	0.82827	-0.01434
0.80000	0.82786	-0.00299
0.85000	0.82791	0.00437
0.90000	0.82822	0.00748
0.95000	0.82858	0.00611
1.00000	0.82875	0.00000

Table 5

x	u	u'
0.0000	1.0000	0.0000
0.0500	1.0013	0.0513
0.1000	1.0052	0.1052
0.1500	1.0118	0.1618
0.2000	1.0214	0.2214
0.2500	1.0340	0.2840
0.3000	1.0499	0.3499
0.3500	1.0691	0.4191
0.4000	1.0918	0.4918
0.4500	1.1183	0.5683
0.5000	1.1487	0.6487
0.5500	1.1833	0.7332
0.6000	1.2221	0.8221
0.6500	1.2655	0.9155
0.7000	1.3138	1.0137
0.7500	1.3670	1.1170
0.8000	1.4255	1.2255
0.8500	1.4896	1.3396
0.9000	1.5596	1.4596
0.9500	1.6357	1.5857
1.0000	1.7183	1.7183

Table 6

x	u	u'
0.0000	1.0000	1.0000
0.0500	1.0513	1.0525
0.1000	1.1053	1.1102
0.1500	1.1624	1.1731
0.2000	1.2227	1.2415
0.2500	1.2866	1.3154
0.3000	1.3544	1.3952
0.3500	1.4263	1.4809
0.4000	1.5026	1.5729
0.4500	1.5837	1.6713
0.5000	1.6698	1.7763
0.5500	1.7614	1.8883
0.6000	1.8588	2.0076
0.6500	1.9623	2.1343
0.7000	2.0723	2.2689
0.7500	2.1893	2.4117
0.8000	2.3136	2.5630
0.8500	2.4458	2.7232
0.9000	2.5861	2.8927
0.9500	2.7352	3.0719
1.0000	2.8935	3.2613

Table 7

x	u	u'
0.0000	1.0000	0.2397
0.0500	1.0133	0.2913
0.1000	1.0292	0.3461
0.1500	1.0479	0.4043
0.2000	1.0697	0.4659
0.2500	1.0946	0.5313
0.3000	1.1229	0.6004
0.3500	1.1547	0.6736
0.4000	1.1903	0.7510
0.4500	1.2299	0.8327
0.5000	1.2736	0.9190
0.5500	1.3218	1.0101
0.6000	1.3747	1.1063
0.6500	1.4326	1.2077
0.7000	1.4956	1.3146
0.7500	1.5641	1.4274
0.8000	1.6384	1.5461
0.8500	1.7188	1.6713
0.9000	1.8057	1.8031
0.9500	1.8993	1.9420
1.0000	2.0000	2.0882

Table 8: Compute absolute error in the scientific notation of Case 1

x	Exact solution	Absolute error(n=4) (Laguerre poly.)	Absolute error(n=6) (Laguerre poly.)	Absolute error(n=4) (Hermite poly.)	Absolute error(n=6) (Hermite poly.)
0.0	1.0000	0×10^{-5}	0×10^{-5}	6×10^{-3}	0×10^{-5}
0.1	0.9555	3×10^{-4}	4×10^{-4}	4.9×10^{-3}	0×10^{-5}
0.2	0.9151	1×10^{-4}	3×10^{-4}	4.2×10^{-3}	1×10^{-4}
0.3	0.8838	0×10^{-5}	1×10^{-4}	2.7×10^{-3}	2×10^{-4}
0.4	0.8605	3×10^{-4}	1×10^{-4}	9.7×10^{-3}	0×10^{-5}
0.5	0.8445	3×10^{-4}	1×10^{-4}	1.35×10^{-2}	1×10^{-4}
0.6	0.8345	3×10^{-4}	1×10^{-4}	1.23×10^{-2}	0×10^{-5}
0.7	0.8294	1×10^{-4}	2×10^{-4}	7×10^{-3}	2×10^{-4}
0.8	0.8279	2×10^{-4}	3×10^{-4}	3×10^{-4}	3×10^{-4}
0.9	0.8282	7×10^{-4}	5×10^{-4}	3.2×10^{-3}	1×10^{-4}
1.0	0.8288	0×10^{-4}	4×10^{-4}	2×10^{-3}	2×10^{-4}

Table 9: Compute absolute error in the scientific notation of Case 2

x	Exact solution	Absolute error(n=4) (Laguerre poly.)	Absolute error(n=6) (Laguerre poly.)	Absolute error(n=4) (Hermite poly.)	Absolute error(n=6) (Hermite poly.)
0.0	1.0000	6×10^{-4}	1.1×10^{-3}	1×10^{-3}	0×10^{-5}
0.1	1.5547	1.2×10^{-3}	8×10^{-4}	6.6×10^{-3}	1×10^{-4}
0.2	2.1451	1×10^{-3}	5×10^{-4}	6.2×10^{-3}	0×10^{-5}
0.3	2.7013	3×10^{-4}	3×10^{-4}	2.6×10^{-3}	0×10^{-5}
0.4	3.2163	4×10^{-4}	1×10^{-4}	1.3×10^{-3}	0×10^{-5}
0.5	3.6834	8×10^{-4}	0×10^{-5}	3.4×10^{-3}	1×10^{-4}
0.6	4.0961	6×10^{-4}	3×10^{-4}	2.9×10^{-3}	1×10^{-4}
0.7	4.4486	0×10^{-5}	4×10^{-4}	1×10^{-4}	0×10^{-5}
0.8	4.7353	7×10^{-4}	5×10^{-4}	3.6×10^{-4}	0×10^{-5}
0.9	4.9510	1.1×10^{-3}	6×10^{-4}	5.5×10^{-3}	1×10^{-4}
1.0	5.0912	0×10^{-5}	7×10^{-4}	1.8×10^{-3}	1×10^{-4}

Table 10: Compute absolute error in the scientific notation of Problem 2

x	Exact solution	Absolute error(n=4) (Laguerre poly.)	Absolute error(n=6) (Laguerre poly.)	Absolute error(n=4) (Hermite poly.)	Absolute error(n=6) (Hermite poly.)
0.0	1.0000	4×10^{-3}	8×10^{-3}	0×10^{-5}	0×10^{-5}
0.1	1.0258	3.1×10^{-3}	5.6×10^{-3}	3×10^{-4}	3×10^{-4}
0.2	1.0697	1.9×10^{-3}	9×10^{-4}	2×10^{-4}	4×10^{-3}
0.3	1.1229	1.114×10^{-1}	3.5×10^{-3}	4.3×10^{-3}	4.5×10^{-3}
0.4	1.1903	8.1×10^{-3}	5.5×10^{-3}	4.5×10^{-3}	4.6×10^{-3}
0.5	1.2736	9.1×10^{-3}	6.9×10^{-3}	4.6×10^{-3}	4.4×10^{-3}
0.6	1.3747	7.7×10^{-3}	7.8×10^{-3}	4.4×10^{-3}	4.3×10^{-3}
0.7	1.4956	4.5×10^{-3}	8.1×10^{-3}	4×10^{-3}	4.1×10^{-3}
0.8	1.6384	5×10^{-4}	7.7×10^{-3}	3.1×10^{-3}	3.2×10^{-3}
0.9	1.8057	1.9×10^{-3}	6.7×10^{-3}	1.8×10^{-3}	1.8×10^{-3}
1.0	2.0000	3×10^{-4}	5×10^{-3}	0×10^{-5}	0×10^{-5}

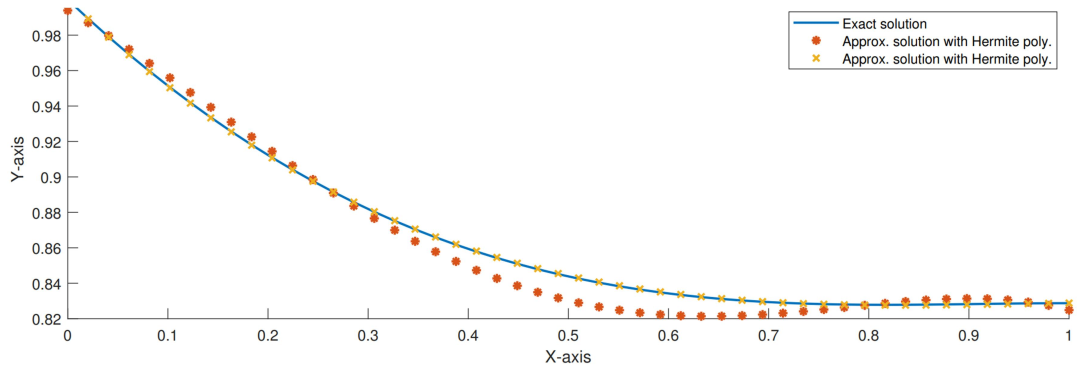

Figure1. Graph of exact and approximate solution of case 1 with Hermite and Laguerre polynomials (n=4)

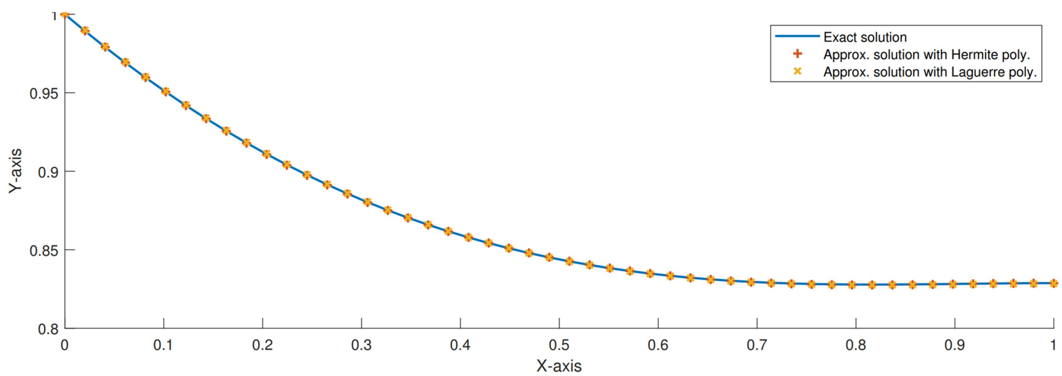

Figure2. Graph of exact and approximate solution of case 1 with Hermite and Laguerre polynomials (n=6)

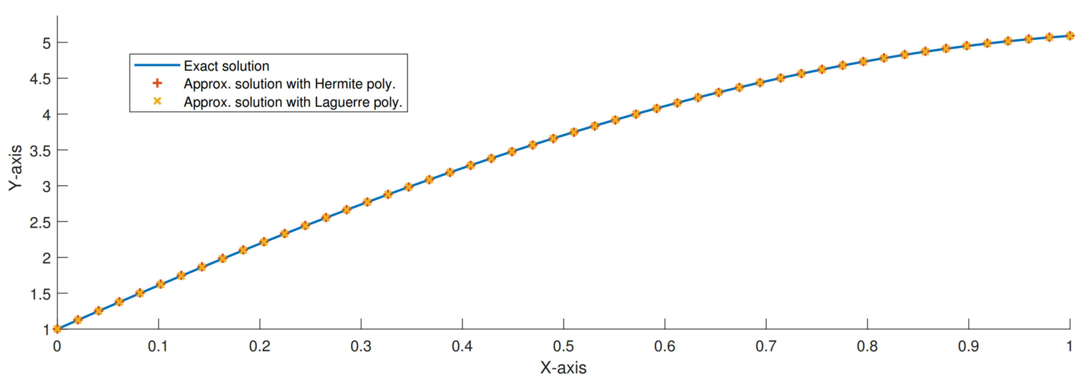

Figure3. Graph of exact and approximate solution of case 2 with Hermite and Laguerre polynomials (n=4)

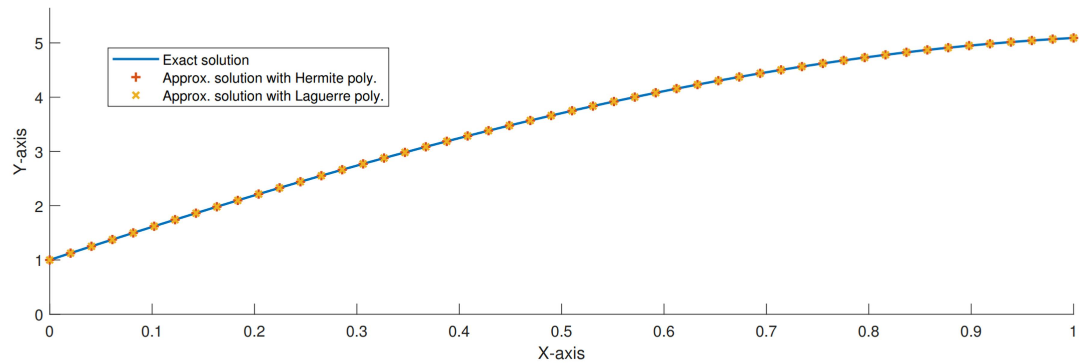

Figure4. Graph of exact and approximate solution of case 2 with Hermite and Laguerre polynomials (n=6)

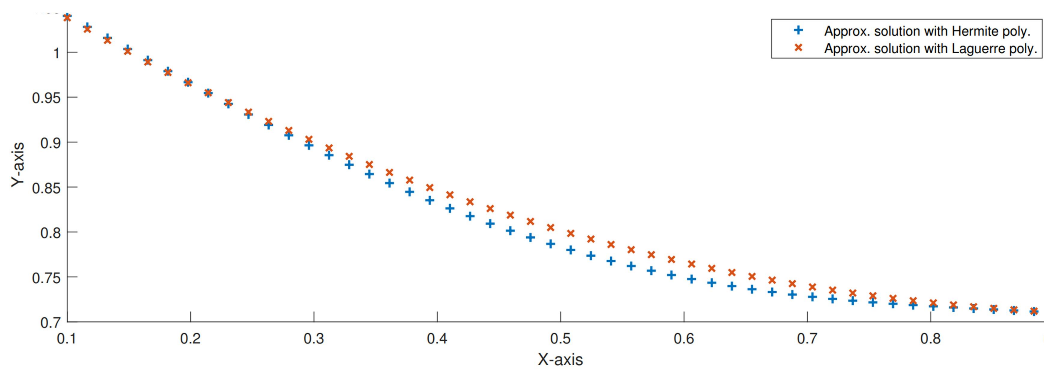

Figure5. Graph of exact and approximate solution of case 3 with Hermite and Laguerre polynomials (n=4)

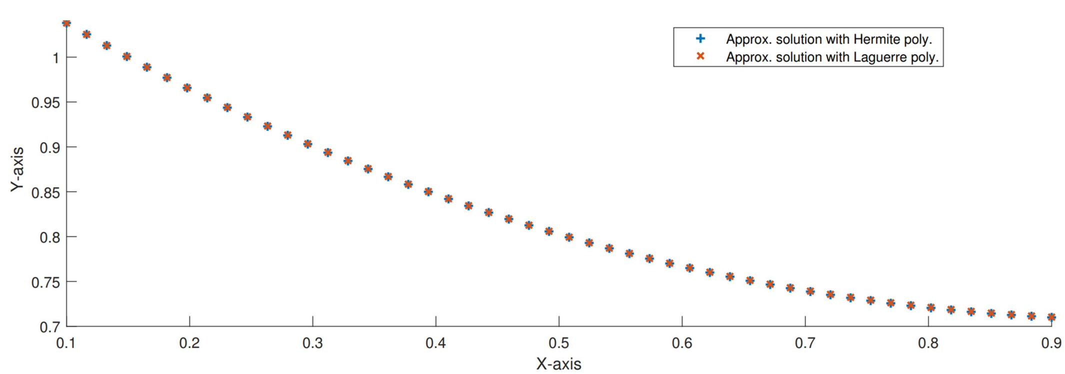

Figure6. Graph of exact and approximate solution of case 3 with Hermite and Laguerre polynomials (n=6)

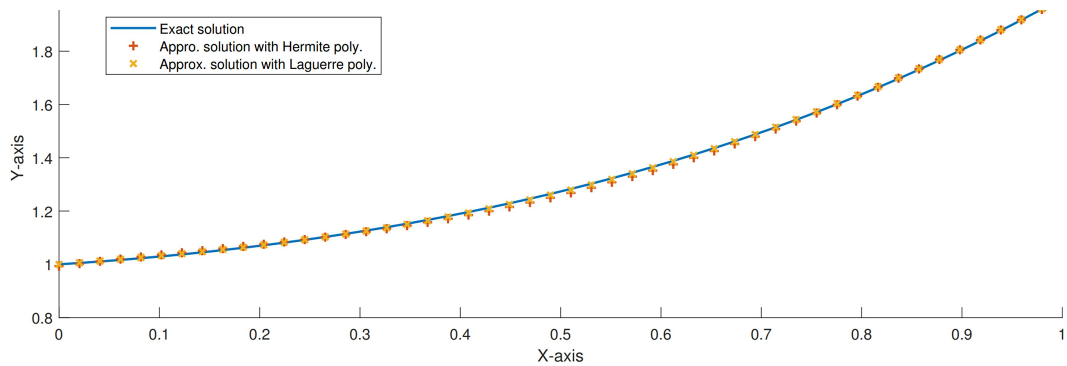

Figure7. Graph of exact and approximate solution of problem2 with Hermite and Laguerre polynomials (n=4)

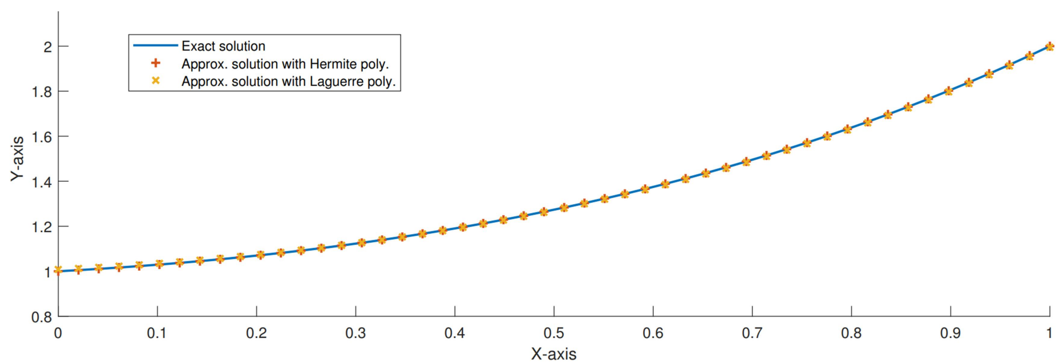

Figure8. Graph of exact and approximate solution of problem2 with Hermite and Laguerre polynomials (n=6)

V. CONCLUSION

In this work, we have developed the Galerkin approach to approximate the solution of second-order mixed and Dirichlet BVPs. It is observed that increases the accuracy of the approximate solution after converting the mixed and Dirichlet BVPs into Neumann BVPs. We also notice that the approximate solutions coincide with the exact solutions even though a few of the polynomials are used in the approximation which is shown in Table 7, Table8, and Table 9. In order for this method to produce better results as the number of Hermite polynomials increases and for the Runge-Kutta method to be used with a modest step size. Accuracy will be better as increase the value of n with the Hermite polynomial but in the case of the Laguerre polynomial, the stiffness matrix is close to singular as n increases in maximum problems.

REFERENCES

[1] T. Kim, D. Kim, K. Hwang, J.J. Sea, " Some identities of Laguerre Polynomials arising from differential equation," Advances in Difference Equations, 159(2016).

[2] J. N. Reddy, "An introduction to the finite element method," 3rd Edition, McGrew-Hill,(2011), 58-98.

[3] V. Mishra, D. Rani, "Approximate solution of boundary value problem with Bernstein polynomial Laplace decomposition method," International Journal of Pure and Applied Mathematics, 114(2017), 823-833.

[4] M. K. Jain, "Numerical method for scientific and engineering computation," Seventh Edition, New Age International, New Delhi (2019).

[5] M. M. Rahman, " Numerical solution of second order boundary value problems by Galerkin method with Hermite polynomials," Pure and Applied Math Computational, 154(2004), 175-182.

[6] A. Khan, "Parametric cubic spline solution of two point boundary value problem," Applied Math Computational, 154(2004), 175-182.

[7] K. N. S. Kasi Viswanadham, P. M. Krishnal, R. S. Koneru, "Fourth order boundary value problem by Galerkin method with cubic B-Splines," International Journal of Non-Linear Science, 10(2013), 222-230.

[8] A. Sidi, "Generalization of the secant method for non-linear equations," Applied Mathematics E-Notes, 8(2008), 115-123.

[9] A. A. Anulo, A. S. Kibret, G. G. Gonfa, and A. D. Negassa, "Numerical solution of linear second order ordinary differential euation with mixed boundary conditions by Galerkin method'" Mathematics and Computer science, 2(2017), 66-78.

[10] M. M. Rahman, M. A. Hossen and M. N. Islam, "Numerical solution of second order boundary value problem by Galerkin method with Hermite polynomials," Annals of Pure and Applied Mathematics, 1(2012), 138-148.

[11] J. Solsvik, S. Tangen, H. A. Jakobsen, "Evaluation of weighted residual method for the solution of the Pallet equations: The orthogonal collocation , Galerkin, Tau and Least-square method," Computer and chemical engineering, 58(2013), 223-259.

[12] B. Saka, I. Dag, A. Dogan, "Galerkin method for the numerical solution of RLW equation using quadratic B-Spline," International Journal of Computer Mathematics, 81(2004), 727-739.

[13] D. S. Mashat, L. A. Wazzan, M. S. Ismail, "Petrov-Galerkin method and K(2, 2) equation," International Journal of Computer Mathematics, 83(2006), 331-343.