Global Scientific JOURNALS

GSJ: Volume 7, Issue 6, June 2019, Online: ISSN 2320-9186

www.globalscientificjournal.com

On Analyzing Numerical solution of time-Independent Schrodinger equation

Akalu Abriham Anulo

Department of Mathematics, Dire Dawa University, Dire Dawa, Ethiopia

Abstract

In this study numerical solutions of time-independent Schrodinger wave equation (TISWE) under infinite potential well were analyzed. The TISWE is reduced to computationally tractable form by using Galerkin method and then the approximate solution is analyzed on the interval $[-1,1]$ *and Chebysheve polynomials were used as a trial function. The approximate solutions generated using this numerical scheme is highly accurate and physically acceptable. Finally the result is compared with the analytic solution which shows that this numerical method is one of the finest numerical methods to find approximate solutions of TISWEs.*

Key-Words: - Time-independent Schrodinger equation, Galerkin Method, Chebysheve Polynomial

1. Introduction

The Schrodinger equation is essential for many non-relativistic problems and gives an accurate description of the true nature of the microscopic world in a probabilistic sense which can be applied quantum mechanics, nuclear physics, theoretical physics and chemistry, physical chemistry, chemical physics and in many other scientific areas to describe the quantum system through a well-defined wave function determined by applying suitable analytic or numerical method. [1, 2]

This important equation was formulated by Erwin Schrodinger in 1926, which showed that it is the equivalent of the Newtonian laws but for quantum systems [3]. Based on the use of time as an independent variable, the Schrödinger wave equation can be categorized as time-independent Schrödinger equation (TISE) and time-dependent Schrödinger equation (TDSE). There are various ways for which Schrodinger equations can be solved analytically, for instance the harmonic oscillator and the hydrogen atom methods are widely used methods [3]. However, in most cases of practical interest (like atomic, molecular, and

solid-state physics) exact or approximate numerical methods must be employed [4].

Many authors have done analytic as well as numerical solutions of both TISE and TDSE.

Some of the recent analytic solutions of Schrodinger equation include Elzaki decomposition method [5], homotopy perturbation and Adomian decomposition methods [6, 7]. On the other hand many authors developed approximate, numerical solutions of both TISE and TDSE, to mention some; numerical solution for time-independent Schrodinger wave equation using Numerov algorithm for the square well, harmonic and linear potentials [8], time-independent Schrodinger equation using wavelet method by determining the wave function using the harmonic multi-resolution analysis [9], numerical solution of time-independent Schrodinger equation is analyzed using imaginary time propagation method [10], a direct way to the exact controllability of the 1D Schrödinger equation with Dirichlet boundary control is derived using the flatness approach which involves in parameterizing the solution and the control by the derivatives of a flat output [11].

In addition to the authors listed above; analytical techniques for the generation of wide classes of exact solutions of the nonlinear Schrödinger equation containing an external potential are proposed [12], a numerical solution of TISE is developed without making any approximation by applying simple procedure to some quantum mechanical problems in one dimension [13], approximate solution of one-dimensional TISE using symplectic schemes [14], numerical solution of one dimensional TISE in quantum-well structures presented based on the transfer matrix method [15] , a MATLAB code is developed by applying finite difference method on time-independent Schrodinger equation in which a potential well is taken (particle in a box) and the wave-function of the particle is calculated by solving Schrodinger equation[16], authors in [17,18,20,21 22,23,24] established numerical solution for TDSE and in [25,26,27,28] numerical methods presented for nonlinear Schrodinger equations.

In this paper numerical solution of TISE is analyzed and accurate numerical solution is obtained using the algorithm formulated in [29], by avoiding Runge-Kutta and secant methods used in [29]. Hence this paper presents numerical solution of TISE by Galerkin Method using Chebysheve polynomials as a trial function.

This Paper is outlined as; section II, brief explanation of mathematical formulation of the method will be clarified which comprises; the conditions that the

approximate solution should satisfy, derivatives of the wave function and description and implementation of Galerkin method. Section III, numerical results and examples by comparing with the exact solution will be discussed and finally in section IV possible conclusion will be presented.

2. Mathematical formulation of the Method

In this study we consider the time-independent Schrodinger equation for a particle bounded by infinitely high potential barriers within a region $0 < x < r$;

$$\frac{d^2\psi(x)}{dx^2} + k^2\psi(x) = 0 \quad \ldots\ldots\ldots\ldots (1)$$

Where, $\psi(x)$ - is the wave function,

$$k = \frac{\sqrt{2Em}}{\hbar}$$

E-Energy of the particle

m- Mass of the particle and

$$\hbar = \frac{h}{2\pi} \quad \&$$

$h = 6.62607015 \times 10^{-34}\, J.s$

called Planck's constant.

Since the probability of the particle being found at $x = 0$ and $x = d$ is zero, we have boundary conditions given by;

$$\psi(0) = 0 \text{ and } \psi(r) = 0 \quad \text{---------- (2)}$$

The general solution of equation (1) is given by;

$\psi(x) = A\sin(kx) + B\cos(kx)$, where A and B are constants that we should determine.

Now using the boundary conditions in (2) we get;

$$\psi(0) = B\cos((k)(0)) = 0 \Rightarrow B = 0$$

$\psi(r) = A\sin(kr) = 0 \Rightarrow$ *either* ***A=0 or sin(kr)=0***

If we consider A=0, $\psi(x) = 0$, which is not useful solution. Thus now considering

$$\boldsymbol{\sin(kr)=0 \Rightarrow kr=n\pi}$$

$$\boldsymbol{\Rightarrow k=n\pi/r \text{ for } n=0,\pm 1,\pm 2,\ldots} \quad \ldots\ldots\ldots\ldots (3)$$

For $n = 0 \Rightarrow \psi(x) = 0$, this is again a trivial solution of (1).

Considering $n = 1, 2, 3, \ldots$ $k_n = n\pi / r$ and the negative side of n yields the same wave function with negative sign.

Now equating the equations for k in equation (1) and (3) we get;

$$E_n = \frac{\pi^2 n^2 \hbar^2}{2r^2 m} \quad \text{---------- (4)}$$

Thus for different values of n there is corresponding energy, E and for different choice of E we have different wave function $\psi(x)$. To emphasize this fact we write the wave function as $\psi_E(x)$ representing a wave function for corresponding value of energy, E.

2.1. The conditions $\bar{\psi}_n(x)$ should satisfy

To have physically acceptable the approximate solution, $\boldsymbol{\bar{\psi}_n(x)}$, obtained by this method must satisfy two conditions.

1) Quantization of Energy

2) Continuity

2.1.1. The Quantization of Energy

As the Schrodinger equation describes the probability of the existence of a particle in a given region, $0 < x < r$, then we must remain aware of one further requirement of a wave function which comes from its probability interpretation. So we have the amazing result that the probability interpretation of the wave function forces us to conclude that the allowed energies of a particle moving in a potential $V(x)$ are restricted to certain discrete values, these values determined by the nature of the potential. This is the phenomenon known as the quantization of energy, a result of quantum mechanics which has vast significance for determining the structure of atoms or generally to go further on the properties of matter overall [3].

Therefore to become physically acceptable a wave function must satisfy the normalization condition; That is,

$$\int_0^r |\psi(x)|^2 \, dx = 1 \quad \ldots\ldots\ldots\ldots\ldots\ldots \quad (5)$$

From the general solution of equation (1) and the boundary condition in (2) the value of $B = 0$, but to find the value of A the solution of equation (1) should be normalized.

$$\psi(x) = A\sin(kx) = A\sin(\frac{n\pi x}{r})$$

$$\Rightarrow \int_0^d \left| A\sin(\frac{n\pi x}{r}) \right|^2 dx = 1 \Rightarrow A = \sqrt{\frac{2}{r}}$$

$$\Rightarrow \psi_n(x) = \sqrt{\frac{2}{r}}\sin(\frac{n\pi x}{r}), \quad for \quad 0 < x < r \text{ ----------} \quad (6)$$

Using this equation a number of solutions can be obtained for different values of n.

2.1.2. Continuity

On the other hand to be physically acceptable, the wave function and its derivative should be continuous on $0 < x < r$ .Thus, the approximate solution obtained using this method and its derivative should be continuous on $0 < x < r$

The analytic solution in equation (6) above and its derivative in equation (7) below, both $\psi(x)$ ***and*** $\psi_n'(x)$ are continuous on $0 < x < r$.

$$\psi_n'(x) = \frac{n\pi}{r}\sqrt{\frac{2}{r}}\cos(\frac{n\pi}{r}x) \ldots\ldots\ldots\ldots\ldots \quad (7).$$

2.2. Derivative of the wave function

The derivative of the wave function given in equation (7) is continuous function $\forall x \in (0, r)$ and at the end point of this interval the value of $\psi_n'(x)$ are determined as follows.

$$\psi_n'(0) = \frac{n\pi}{r}\sqrt{\frac{2}{r}} \; for \; n = 1, 2, \ldots \quad \text{----------} \quad (8)$$

$$\psi_n'(r) = (-1)^n \frac{n\pi}{r}\sqrt{\frac{2}{r}} \; for \; n = 1, 2, \ldots \text{------} \quad (9)$$

From equations (8) and (9) can be concluded as;

$\psi_n'(0) = \psi_n'(r)$ *for n even and*

$\psi_n'(0) = -\psi_n'(r)$ *for n odd*

Table1. Analyzing the values of $\psi_n'(0)$ *&* $\psi_n'(r)$ *using equations (8) and (9) for* $r=1\&2\&n=1,2,3$

r	n	$\psi_n'(0)$	$\psi_n'(r)$
1	1	$\pi\sqrt{2}$	$-\pi\sqrt{2}$
	2	$2\pi\sqrt{2}$	$2\pi\sqrt{2}$
	3	$3\pi\sqrt{2}$	$-3\pi\sqrt{2}$
2	1	$\pi/2$	$-\pi/2$
	2	π	π
	3	$3\pi/2$	$-3\pi/2$

Now up on applying the mathematical formulations in [3], the Schrodinger equation given in the form;

$\frac{d^2\psi(x)}{dx^2}+k^2\psi(x)=0$ for $0<x<r$ is $\psi(0)=0=\psi(r)$

transformed in to equivalent Schrodinger equation on $-1<x<1$ given by;

$$\frac{d^2\psi(x)}{dx^2}+\left(\frac{kr}{2}\right)^2\psi(x)=0$$
$$\psi(-1)=0=\psi(1) \qquad \ldots\ldots\ldots\ldots\ldots (8)$$

2.3. Galerkin Method

In this section we introduce the Galerkin Method to solve equation [8]. Galerkin method is one of the weighted residual methods in which the approximation function is the same as the weight function [29].

Consider an operator equation

$$\boldsymbol{A}u=f \quad \boldsymbol{in}\ \Omega \qquad \ldots\ldots\ldots\ldots\ldots\ldots (9)$$

$\boldsymbol{A}$ is an operator, often a differential operator acting on u.

The approximate solution of (10) is given by;

$U_N(\boldsymbol{x})=\sum_{i=0}^{N} d_i\phi_i(\boldsymbol{x})$ where $\phi_i(\boldsymbol{x})$ is the approximating function.

The difference $\boldsymbol{A}(U_N)-f$, called the residual of the approximation is nonzero;

$$\boldsymbol{R}\equiv \boldsymbol{A}(\boldsymbol{U}_N)-f=\boldsymbol{A}(\sum_{i=0}^{N} d_i\phi_i(\boldsymbol{x}))-f\neq 0$$

In the weighted residual method, as the name suggests, the parameters $d_i's$ are determined by requiring the residual R to vanish in the weighted-integral sense; That is;

$$\int_\Omega \phi_i(\boldsymbol{x})\boldsymbol{R}(\boldsymbol{x},\boldsymbol{d}_i)dxdy=0 \qquad (i=1,2,\ldots,N)$$

…… (10).

Where $\phi_i(\boldsymbol{x})$ is the weighted function which is the same as the approximating function and Ω is a two dimensional domain [30].

Thus, the Galerkin method to solve equation (8) is to find the values of the constants, $d_i's$ by assuming that the approximate solution of (8) is given by;

$$\bar{\psi}(x)=\sum_{i=0}^{m} d_iN_i(x) \qquad \ldots\ldots\ldots\ldots\ldots\ldots\ldots (11)$$

Where $N_i(x)$ for $i=1,2,3,\ldots,m$ are Chebysheve polynomial.

Applying Galerkin method to equation (8) we have;

$$\int_{-1}^{1}\left(\frac{d^2\bar{\psi}}{dx^2}+\left(\frac{kr}{2}\right)^2\bar{\psi}(x)\right)N_j(x)dx=0 \quad \ldots (12)$$

Up on integrating equation (10) and applying integration by parts on the first term on the left hand side;

$$\int_{-1}^{1}[-\frac{d\bar{\psi}(x)}{dx}N_j'(x)+\left(\frac{kr}{2}\right)^2\bar{\psi}(x)N_j(x)]dx=\bar{\psi}'(-1)N_j(-1)-\bar{\psi}'(1)N_j(1) \quad \ldots\ldots\ldots\ldots\ldots\ldots (13)$$

Substituting equation (9) into equation (11) and after some rearrangement;

$$\sum_{i=0}^{m}d_i\int_{-1}^{1}\left[-N_i'(x)N_j'(x)+N_i(x)N_j(x)\right]dx=\bar{\psi}'(-1)N_j(-1)-\bar{\psi}'(1)N_j(1) \ldots\ldots\ldots\ldots\ldots\ldots (14)$$

Rewriting equation (14) in short form;

$$\sum_{i=0}^{m}d_iR_{ij}=G_i \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots (15)$$

Where $R_{ij}=\int_{-1}^{1}\left[-N_i'(x)N_j'(x)+N_i(x)N_j(x)\right]dx$ and $G_i=\bar{\psi}'(-1)N_j(-1)-\bar{\psi}'(1)N_j(1)$

Now before beginning to determine the constants d_i's, it needs a way out to replace values for $\bar{\psi}'(-1)\,\&\,\bar{\psi}'(1)$. But in Table 1 above we have values for $\psi'(0)=\bar{\psi}'(-1)$ and $\psi'(r)=\bar{\psi}'(1)$ which corresponds to each r and n values. Substituting these values into G_i and evaluating the integral for R_{ij}, we have a system of equations with m unknowns and m equations and the only variables left unknown are the values of d_i's. Using direct method of solving system of linear equation the values of d_i's are obtained and substituting these values into equation (11) the approximate solution of time-independent Schrodinger wave equation in equation (1) $\forall x\in(-1,1)$ will be generated.

3. Numerical Result

In this section we deal with the numerical result of the method compared to the analytical solution using tables and graphs as a tool to compare. Finally we will check whether the generated approximate solution of time-independent Schrodinger wave equation is physically acceptable (whether it satisfies quantization of energy and continuity properties).

Consider the time-independent Schrodinger wave equation given in equation (1) with boundary condition given in below.

$$\frac{d^2\psi(x)}{dx^2}+k^2\psi(x)=0, \quad \psi(0)=0,\psi(2)=0 \quad \text{with exact solution } \psi(x)=\sin(n\pi(x+1)/2),\ \textit{where } n=1,2,3,\ldots$$

The corresponding equivalent time-independent Schrodinger wave equation is given as;

$$\frac{d^2\psi(x)}{dx^2}+\left(\frac{kr}{2}\right)^2\psi(x)=0$$
$$\psi(\text{-}1)=0=\psi(1)$$

But the value of r is given which is $r=2 \Rightarrow k=n\pi/2$

$$\Rightarrow \frac{d^2\psi(x)}{dx^2}+k^2\psi(x)=0$$
$$\psi(\text{-}1)=0=\psi(1)$$

Now find the values of column vector G_i and $m\times m$ coefficient matrix R_{ij} given by;

$$R_{ij}=\int_{-1}^{1}\left[-N_i'(x)N_j'(x)+N_i(x)N_j(x)\right]dx \text{ and } G_i=\bar{\psi}'(-1)N_j(-1)-\bar{\psi}'(1)N_j(1)$$

The approximate solutions for the above Schrodinger equation using the first four Chebysheve polynomials for $n=1,n=2\,\&\,n=3$ are shown below together with their corresponding graphical comparison with analytic solution.

Thus the resulting approximate solutions of the given Schrodinger equation for $n=1,2\,\&\,3$ are;

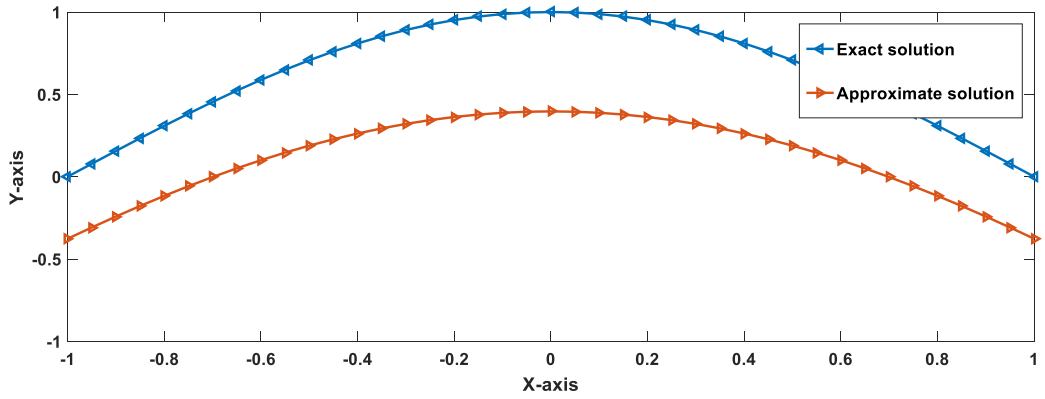
$$\bar{\psi}_1(x)=\frac{6960880021566711883670933313945 60}{22238096227716583542872639283363058 3}\pi(8x^4-8x^2+1)-(\frac{2737315788716042759425262119550976 0\pi}{22238096227716583542872639283363058 3}(2x^2-1)$$

$$\bar{\psi}_2(x)=\frac{-6842067937608804259860158873 60}{38302555212077620602527211014 59}\pi(-4x^3+3x)-(\frac{7840969636784191131966882447 36}{38302555212077620602527211014 59}\pi x)$$

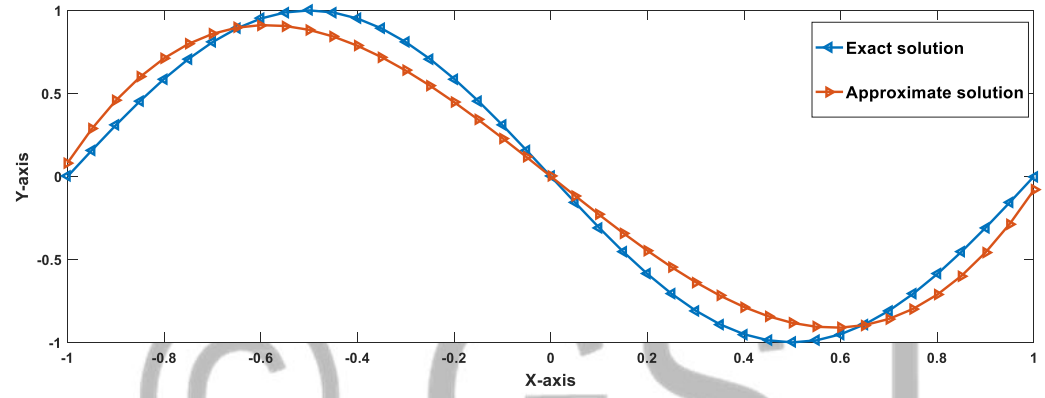
$$\bar{\psi}_3(x)=\frac{4672001744007950987795744214220 80}{60833448046247826530405098738370 09}\pi(2x^2-1)-(\frac{8177564209417042861009693310976 00}{60833448046247826530405098738370 09}\pi(8x^4-8x^2+1)$$

…….(*)

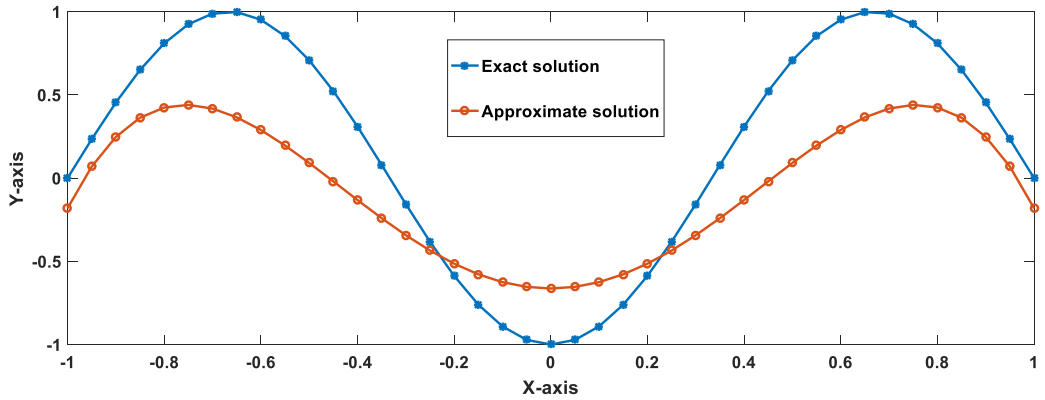
The graphs of the above approximate solution and their corresponding exact solution for $n=1,2\,\&\,3$ are shown below



(a)



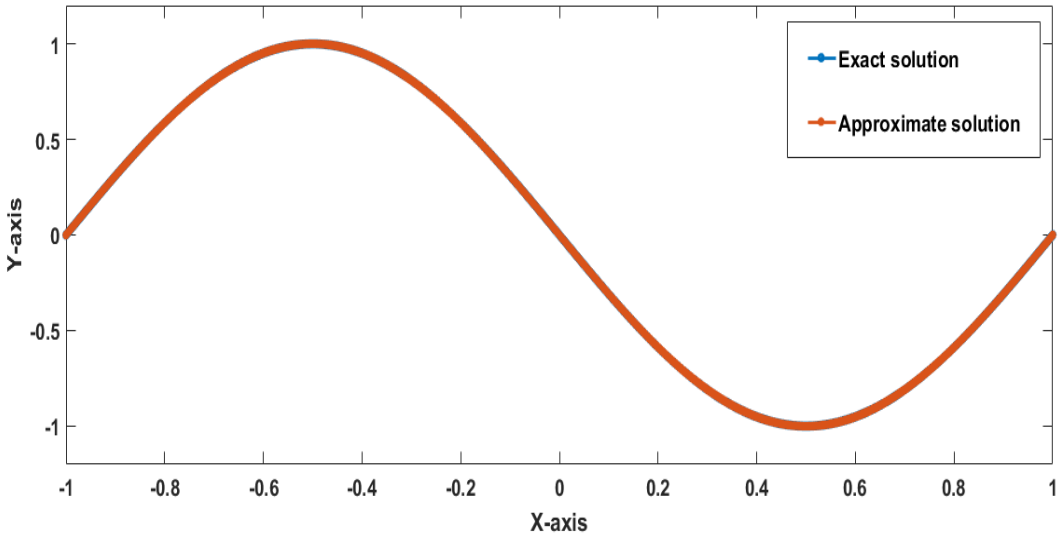
(b)



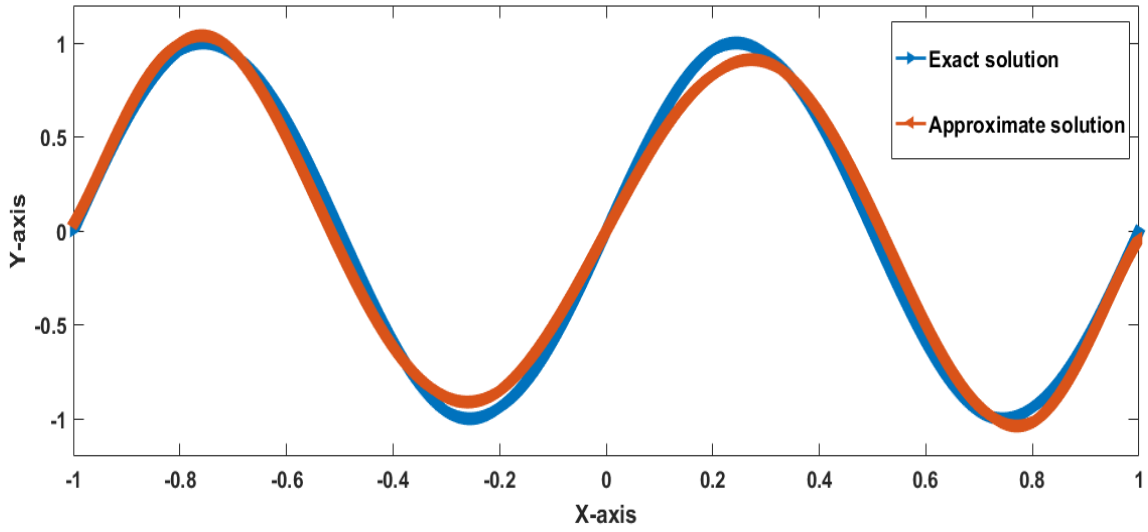
(c)

Figure 1 : ***(a)*** *the graphs of* $\bar{\psi}_1(x)$ *for n=1.* ***(b)*** *The graphs of* $\bar{\psi}_2(x)$ *for n=2* ***(c)*** *the graphs of* $\bar{\psi}_3(x)$ *for n=3.*

Similarly, but increasing the number of Chebysheve polynomials used , the associated approximate solutions for $n=2 \& n=4$ and their corresponding analytic solutions are presented graphically as well as numerically as follows.



(a)



(b)

*Figure 2 :**(a)** the graphs of $\bar{\psi}_2(x)$ for n=2. **(b)** The graphs of $\bar{\psi}_2(x)$ for n=4*

Table 2.The absolute error $|\psi_n(x)-\bar{\psi}_n(x)|$ for $n=2$

x	$\psi_2(x)$	$\bar{\psi}_2(x)$	$\lvert\psi_n(x)-\bar{\psi}_n(x)\rvert$
-1	0	0.000002219800211	$2.21980021135500\times10^{-6}$
-0.8	0.587785252292473	0.587512277196491	$27.2975095982342\times10^{-6}$
-0.6	0.951056516295154	0.951315357695305	$25.8841400151200\times10^{-6}$
-0.4	0.951056516295154	0.951164590915684	$10.8074620530640\times10^{-6}$
-0.2	0.587785252292473	0.587392421522651	$39.2830769822550\times10^{-6}$
0	0.000000000000000	0.000000000000122	$0.00000012200000\times10^{-6}$
0.2	-0.587785252292473	-0.587392421522651	$39.2830769822328\times10^{-6}$
0.4	-0.951056516295154	-0.951164590915684	$10.8074620530640\times10^{-6}$
0.6	-0.951056516295154	-0.951315357695305	$25.8841400151089\times10^{-6}$
0.8	-0.587785252292473	-0.587512277196491	$27.2975095982675\times10^{-6}$
1	-0.000000000000000	-0.000002219800211	$2.21980021111000\times10^{-6}$

As presented graphically and analyzed numerically, the results obtained using this method are clearly agrees with the exact solution. But to be physically acceptable the approximate solution should satisfy the condition known as quantization of energy which is briefly explained in section 2.1. Thus now we will check whether these approximate solutions satisfy this property.

Considering the approximate solutions obtained above, the following result is generated in order to show whether they satisfy the condition;

$\int_{-1}^{1}|\bar{\psi}_n(x)|^2\,dx=1$, for n=1 and 2

Table 3.*The value of* $\int_{-1}^{1}|\bar{\psi}_n(x)|^2\,dx$ *for n=1, m=4 and n=2, m=8 and the associated absolute error,* $\left|1-\int_{-1}^{1}|\bar{\psi}_n(x)|^2\,dx\right|$ *corresponding to each approximate solution.*

Exact Solution	$\int_{-1}^{1}\lvert\psi_E(x)\rvert^2\,dx$	**Approximate Solutions**	$\int_{-1}^{1}\lvert\bar{\psi}_n(x)\rvert^2\,dx$	**Absolute error**
$\psi_1(x)$	1.00	$\bar{\psi}_1(x)$	$9.999895631488434\times10^{-1}$	$1.043685115664772\times10^{-5}$
$\psi_2(x)$	1.00	$\bar{\psi}_2(x)$	$9.999895631488434\times10^{-1}$	$1.043685115664772\times10^{-5}$

As shown on table 3 above, the approximate solutions generated through this method clearly satisfied the quantization property with the specified absolute error.

In section 2.1.2, other condition that the approximate solution should satisfy is continuity property. As we have seen from the graphs of the approximate solution, the approximate function, $\bar{\psi}_n(x)$ is continuous $\forall x \in (-1,1)$. Accordingly, these approximate solutions are physically acceptable.

4. Conclusion

In this paper Galerkin method is formulated and employed to solve time-independent Schrodinger wave equation under infinite potential well. The approximate solution obtained through this method is in a good agreement with the exact solution. More importantly the approximate solution obtained via this method is physically acceptable. To test the accuracy of this method an example of TISE is carried out and shown the error numerically and graphically. Thus this method is an alternative way to solve TISE under infinite well.

References

1. H.J.W. Muller-Kirsten, Introduction to Quantum Mechanics: Schrodinger Equation and Path Integral, second ed., World Scientific, Sigapore, 2013.
2. T.E.SIMOS. A fourth algebraic order exponentially-fitted Runge-Kutta method for the numerical solution of the Schrodinger equation, IMA Journal of Numerical Analysis (2001) 21, 919-931.
3. R.A.Bertlmann, and N.Friis, "Theoretical physics T2. Quantum Mechanics", T2-Script of Summer Semester 2008
4. Anders W. Sandvik, Numerical Solutions of the Schrodinger Equation, PY 502, Computational Physics, Fall 2018
5. R. I. Nuruddeen, "Elzaki decomposition method and its applications in solving linear and nonlinear Schrodinger equations," Sohag Journal of Mathematics, vol. 4, no. 2, pp. 1–5, 2017.
6. J. Biazar, R. Ansari, K. Hosseini, and P. Gholamin, "Solution of the linear and non-linear schrodinger equations using homotopy perturbation and Adomian decomposition methods," International Mathematical Forum. Journal for Theory and Applications, vol. 3, no. 37-40, pp. 1891–1897, 2008.
7. A. Sadighi and D. D. Ganji, "Analytic treatment of linear and nonlinear schrodinger equations: a study with homotopy-perturbation and Adomian decomposition methods," Physics Letters A, vol. 372, no. 4, pp. 465–469, 2008.
8. Danny Bennett, Numerical Solutions to the Time-Independent 1-D Schrodinger Equation December 15, 2015
9. S.BIBIC, E. MALUREANU, Wavelet Solution of the Time Independent Schrodinger Equation for a Rectangular Potential Barrier, The 8th International Symposium On Advanced Topics In Electrical Engineering May 23-25, 2013 Bucharest, Romania , IEEE (2013)
10. L. Lehtovaara , J.Toivanen, J.Eloranta, solution of time-independent Schrodinger equation by imaginary time propagation method, Journal of Computational Physics 221 (2007) 148–157
11. Philippe Martin ,Lionel Rosier , Pierre Rouchon, Controllability of the 1D Schrödinger equation using flatness, Automatica, Elsevier, 91 (2018)208–216.
12. B. A. Malomed and Y.A.Stepanyants, Localized nonlinear optical modes and the corresponding support structures: Exact solutions to the nonlinear Schrödinger equation with external potentials
13. H. H. Erbil. A Simple Solution of the Time-Independent Schrödinger Equation in One Dimension. Ege University, Science Faculty, Physics Department Bornova - IZMIR 35100, TURKEY
14. Xue-Shen Liu, Xiao-Yan Liu, Zhong-Yuan Zhou, Pei-Zhu Ding, Shou-Fu Pan. Numerical Solution of One-Dimensional Time-Independent Schrödinger Equation by

Using Symplectic Schemes. International Journal of Quantum Chemistry, Vol. 79, 343–349 (2000)

15. Bjorn Jonsson and Sverre T. Eng. Solving the Schrodinger Equation in Arbitrary Quantum-Well Potential Profiles Using the Transfer Matrix Method. IEEE Journal of Quantum Electronics, Vol. 26, No. 11, November 1990.
16. Sathyanarayan Rao. Numerical Solution of 1D Time Independent Schrodinger Equation using Finite Difference Method. Mathworks version 1.0.0.0, (2015).
17. Preuss Robinson, Mark. (1991). Numerical Solution of Schrodinger Equations Using Finite Element Methods.
18. Koch, Othmar. (2004). Numerical Solution of the Time-Dependent Schrodinger Equation in Ultrafast Laser Dynamics. WSEAS Transactions on Mathematics. 3.
19. Dubeibe, F. (2010). Solving the Time-Dependent Schrödinger Equation with Absorbing Boundary Conditions and Source Terms in Mathematica 6.0. International Journal of Modern Physics C. 21. 1391-1406. 10.1142/S0129183110015919.
20. Christoph Wachter, Numerical Solution of the Time-Dependent 1D-Schro¨dinger Equation using Absorbing Boundary Conditions, 2017
21. Lina Viklund, Louise Augustsson , Jonas Melander, Numerical approaches to solving the time-dependent Schrödinger equation with different potentials, 2016
22. N. A. M. Amin and B. R. Wong, A study of numerical solutions of the time-dependent Schrödinger equation , AIP Conference Proceedings 1682, 020042 (2015); doi: 10.1063/1.4932451
23. Vanilse S. Araujo, Escola , Engenharia da Fundac ¸ F. A. B. Coutinho, Faculdade , TheTime-Dependent Schrodinger Equation: The Need for the Hamiltonian to be Self-Adjoint, 2007
24. Jing Shen1,3, Wei E.I. Sha21 , Zhixiang Huang , Mingsheng Chen3, Xianliang Wu1, The Stability and Numerical Dispersion Analyses of High-Order Symplectic FDTD Scheme for Solving Time-Dependent Schrödinger Equation, IEEE, 2012.
25. Shehu Maitama, Mahmoud S. Rawashdeh and Surajo Sulaiman, An Analytical Method for Solving Linear and Nonlinear Schrödinger Equations, Palestine Journal of Mathematics, Vol. 6(1) (2017) , 59-67
26. Ismail, M.S. and Alaseri, S.H. (2016) Computational Methods for Three Coupled Nonlinear Schrödinger Equations. Applied Mathematics , 7, 2110- 2131.
27. Sarun Phibanchon , Michael A. Allen, Numerical solutions of the nonlinear Schrodinger equation with a square root nonlinearity, 2010 International Conference on Computational Science and Its Applications , 978-0-7695-3999-7/10 IEEE DOI 10.1109/ICCSA.2010.68 293
28. Jaradat, Emad & Alomari, Omar & Abudayah, Mohammad & M. Al-Faqih, Ala'a. (2018). An Approximate Analytical Solution of the Nonlinear Schrödinger Equation with Harmonic Oscillator Using Homotopy Perturbation Method and Laplace-

Adomian Decomposition Method. Advances in Mathematical Physics. 2018. 1-11. 10.1155/2018/6765021.

29. Akalu Abriham Anulo, Alemayehu Shiferaw Kibret, Genanew Gofe Gonfa, Ayana Deressa Negassa. Numerical Solution of Linear Second Order Ordinary Differential Equations with Mixed Boundary Conditions by Galerkin Method. Mathematics and Computer Science. Vol. 2, No. 5, 2017, pp. 66-78. doi: 10.11648/j.mcs.20170205.12
30. J. N. Reddy: An Introduction to the finite element method, 3rd edition, McGraw-Hill, (Jan 2011) 58-98.

Appendix

MATLAB codes used to find the approximate solutions and to sketch their graphs together with their exact solutions.

The MATLAB codes presented below are to find the approximate solution of TISE after converting it into equivalent TISE on $-1 < x < 1$.

```
% The MATLAB code below is used to find the 1st approximate
%solutions considering different values of M, number of
%Chebysheve polynomials used, n, energy level used, L, and the
%length of the interval considered. So here M=4and L=2 and n=1
syms x   % Creating symbolic variable x
T1=x;
T2=2*x^2-1;
T3=4*x^3-3*x;
T4=8*x^4-8*x^2+1;      %Defining the first four Chebysheve
%polynomials in terms of symbolic variable x
T=[T1,T2,T3,T4];%Defining Ni(x) for i=1,2,3 and 4,the %first four
Chebysheve polynomials as 1x4 matrix
TrT=transpose(T); %Find the transpose of Ni(x)=Nj(x)
DrT=diff(T,x);    %Find the derivative of Ni(x)
DrTrT=diff(TrT,x);%Compute the derivative of Nj(x)
n=1; % Assign the value of n (energy level)
L=2; % Assign the value of r (the upper boundary of x)
K=(n*pi)/L; % Compute the value of k presented in %equation (1)
m=(K*L)/2; % Compute the coefficient in equation (8)
k1T=(m^2)*TrT*T; %Defining the term inside the integration sign
k2T=DrTrT*DrT;   %on the LHS of equation (14)
K1=int(k1T,x,-1,1); %Calculating the 1st &2nd
K2=int(k2T,x,-1,1); %terms on the LHS of Equation (14)
KK=K1-K2; %Computing LHS of equation (14)
q=[1 1 1 1]; %The value of Nj(1) for j=1,2,…in equation(14)
p=[-1 1 -1 1]; %The value of Nj(-1) for j=1,2,…,7 in
%equation(14)
N=q'; %Finding the Transpose of q
M=p'; %Compute the Transpose of p
F1=(pi)*M; %Computing the 1st term on the RHS of equation (14)
F2=(pi)*N; %Computing the 2nd term on the RHS of equation (14)
F=F1-F2; %Solving terms on the RHS of equation (14)
c1=KK\F; %Solving for the values of di's in equation (15)
App_sol=T*c1; %solving for the approximate solution using di's
display(App_sol);%Display the approximate solution in terms of x
```

```
% The MATLAB code below is used to find the 1st approximate
%solutions considering different values of M, number of
%Chebysheve polynomials used, n, energy level used, L, and the
%length of the interval considered. So here M=4and L=2 and n=2
syms x   % Creating symbolic variable x
T1=x;
T2=2*x^2-1;
T3=4*x^3-3*x;
T4=8*x^4-8*x^2+1;      %Defining the first four Chebysheve
%polynomials in terms of symbolic variable x
T=[T1,T2,T3,T4];%Defining Ni(x) for i=1,2,3 and 4,the %first four
Chebysheve polynomials as 1x4 matrix
TrT=transpose(T); %Find the transpose of Ni(x)=Nj(x)
DrT=diff(T,x);    %Find the derivative of Ni(x)
DrTrT=diff(TrT,x);%Compute the derivative of Nj(x)
n=2; % Assign the value of n (energy level)
L=2; % Assign the value of r (the upper boundary of x)
K=(n*pi)/L; % Compute the value of k presented in %equation (1)
m=(K*L)/2; % Compute the coefficient in equation (8)
k1T=(m^2)*TrT*T; %Defining the term inside the integration sign
k2T=DrTrT*DrT;   %on the LHS of equation (14)
K1=int(k1T,x,-1,1); %Calculating the 1st &2nd
K2=int(k2T,x,-1,1); %terms on the LHS of Equation (14)
KK=K1-K2; %Computing LHS of equation (14)
q=[1 1 1 1]; %The value of Nj(1) for j=1,2,…in equation(14)
p=[-1 1 -1 1]; %The value of Nj(-1) for j=1,2,…,7 in
%equation(14)
N=q'; %Finding the Transpose of q
M=p'; %Compute the Transpose of p
F1=(pi)*M; %Computing the 1st term on the RHS of equation (14)
F2=(pi)*N; %Computing the 2nd term on the RHS of equation (14)
F=F1-F2; %Solving terms on the RHS of equation (14)
c1=KK\F; %Solving for the values of di's in equation (15)
App_sol=T*c1; %solving for the approximate solution using di's
display(App_sol);%Display the approximate solution in terms of x
```

```
% The MATLAB code below is used to find the 1st approximate
%solutions considering different values of M, number of
%Chebysheve polynomials used, n, energy level used, L, and the
%length of the interval considered. So here M=4and L=2 and n=3
syms x   % Creating symbolic variable x
T1=x;
T2=2*x^2-1;
T3=4*x^3-3*x;
T4=8*x^4-8*x^2+1;      %Defining the first four Chebysheve
%polynomials in terms of symbolic variable x
T=[T1,T2,T3,T4];%Defining Ni(x) for i=1,2,3 and 4,the %first four
Chebysheve polynomials as 1x4 matrix
TrT=transpose(T); %Find the transpose of Ni(x)=Nj(x)
DrT=diff(T,x);    %Find the derivative of Ni(x)
DrTrT=diff(TrT,x);%Compute the derivative of Nj(x)
n=3; % Assign the value of n (energy level)
L=2; % Assign the value of r (the upper boundary of x)
K=(n*pi)/L; % Compute the value of k presented in %equation (1)
m=(K*L)/2; % Compute the coefficient in equation (8)
k1T=(m^2)*TrT*T; %Defining the term inside the integration sign
k2T=DrTrT*DrT;   %on the LHS of equation (14)
K1=int(k1T,x,-1,1); %Calculating the 1st &2nd
K2=int(k2T,x,-1,1); %terms on the LHS of Equation (14)
KK=K1-K2; %Computing LHS of equation (14)
q=[1 1 1 1]; %The value of Nj(1) for j=1,2,…in equation(14)
p=[-1 1 -1 1]; %The value of Nj(-1) for j=1,2,…,7 in
%equation(14)
N=q'; %Finding the Transpose of q
M=p'; %Compute the Transpose of p
F1=(pi)*M; %Computing the 1st term on the RHS of equation (14)
F2=(pi)*N; %Computing the 2nd term on the RHS of equation (14)
F=F1-F2; %Solving terms on the RHS of equation (14)
c1=KK\F; %Solving for the values of di's in equation (15)
App_sol=T*c1; %solving for the approximate solution using di's
display(App_sol);%Display the approximate solution in terms of x
```

```
%The MATLAB code below is used to find the approximate
%solutions considering different values of m, number of
%Chebysheve polynomials used, n, energy level used, and L, the
%length of the interval considered. So here M=8and L=2 and n=2
syms x   % Creating symbolic variable 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;%Defining the first four
%Chebysheve polynomials in terms of symbolic variable x
T=[T1,T2,T3,T4,T5,T6,T7,T8];%Defining Ni(x) for i=1,2,3 and 4,the
%first four Chebysheve polynomials as 1x4 matrix
TrT=transpose(T); %Find the transpose of Ni(x)=Nj(x)
DrT=diff(T,x);    %Find the derivative of Ni(x)
DrTrT=diff(TrT,x);%Compute the derivative of Nj(x)
n=2; % Assign the value of n (energy level)
L=2; % Assign the value of r (the upper boundary of x)
K=(n*pi)/L; % Compute the value of k presented in %equation (1)
m=(K*L)/2; % Compute the coefficient in equation (8)
k1T=(m^2)*TrT*T; %Defining the term inside the integration sign
k2T=DrTrT*DrT;   %on the LHS of equation (14)
K1=int(k1T,x,-1,1); %Calculating the 1st &2nd
K2=int(k2T,x,-1,1); %terms on the LHS of Equation (14)
KK=K1-K2; %Computing LHS of equation (14)
q=[1 1 1 1 1 1 1 1]; %The value of Nj(1) for j=1,2,…in
%equation(14)
p=[-1 1 -1 1 -1 1 -1 1]; %The value of Nj(-1) for j=1,2,…,7 in
%equation(14)
N=q'; %Finding the Transpose of q
M=p'; %Compute the Transpose of p
F1=(pi)*M; %Computing the 1st term on the RHS of equation (14)
F2=(pi)*N; %Computing the 2nd term on the RHS of equation (14)
F=F1-F2; %Solving terms on the RHS of equation (14)
c1=KK\F; %Solving for the values of di's in equation (15)
App_sol=T*c1; %solving for the approximate solution using di's
display(App_sol);%Display the approximate solution in terms of x
```

```
% The MATLAB code below is used to find the approximate
%solutions considering different values of m, number of
%Chebysheve polynomials used, n, the energy level used, L, the
%length of the interval considered.  So here M=8and L=2 and n=3
syms x   % Creating symbolic variable 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;%Defining the first four
%Chebysheve polynomials in terms of symbolic variable x
T=[T1,T2,T3,T4,T5,T6,T7,T8];%Defining Ni(x) for i=1,2,3 and 4,the
%first four Chebysheve polynomials as 1x4 matrix
TrT=transpose(T); %Find the transpose of Ni(x)=Nj(x)
DrT=diff(T,x);    %Find the derivative of Ni(x)
DrTrT=diff(TrT,x);%Compute the derivative of Nj(x)
n=3; % Assign the value of n (energy level)
L=2; % Assign the value of r (the upper boundary of x)
K=(n*pi)/L; % Compute the value of k presented in %equation (1)
m=(K*L)/2; % Compute the coefficient in equation (8)
k1T=(m^2)*TrT*T; %Defining the term inside the integration sign
k2T=DrTrT*DrT;   %on the LHS of equation (14)
K1=int(k1T,x,-1,1); %Calculating the 1st &2nd
K2=int(k2T,x,-1,1); %terms on the LHS of Equation (14)
KK=K1-K2; %Computing LHS of equation (14)
q=[1 1 1 1 1 1 1 1]; %The value of Nj(1) for j=1,2,…in
%equation(14)
p=[-1 1 -1 1 -1 1 -1 1]; %The value of Nj(-1) for j=1,2,…,7 in
%equation(14)
N=q'; %Finding the Transpose of q
M=p'; %Compute the Transpose of p
F1=(pi)*M; %Computing the 1st term on the RHS of equation (14)
F2=(pi)*N; %Computing the 2nd term on the RHS of equation (14)
F=F1-F2; %Solving terms on the RHS of equation (14)
c1=KK\F; %Solving for the values of di's in equation (15)
App_sol=T*c1; %solving for the approximate solution using di's
display(App_sol);%Display the approximate solution in terms of x
```

```
% The MATLAB code below is used to Sketch the graph of the
%approximate solution obtained for n=1, L=2 and M=4 together
%with the exact solution which will give fig1 (a)
x=-4:0.05:4;      % Re-defining the value of x
y=sin((0.5)*pi*(x+1)); % The exact solution of the example
plot(x,y,'->')         % Plotting the graph of the exact solution
hold on               % to sketch on the same plane
App_sol=(69608800215667118836709333139456 0*pi*(8*x.^4 - 8*x.^2 +
1))/22238096227716583542872639283363058 3 -
(27373157887160427594252621195509760*pi*(2*x.^2 -
1))/22238096227716583542872639283363058 3;
plot(x,App_sol,'-*') %Plotting the app_sol on the same plane
ax=[-1 1];
ay=[-1 1];  %Specifying axis to display the graphs
axis([ax ay]);   %Making square axis
title('The graph of exact and approximate solutions for n=1 and
m=4') % Providing the title of the graphs
legend('Exact solution','Approximate solution')% graphs legend
xlabel('X-axis') %Labeling x axis
ylabel('Y-axis') %Labeling y axis
```

```
% The MATLAB code below is used to Sketch the graph of the
%approximate solution obtained for n=2, L=2 and M=4 together
%with the exact solution which will give fig1 (b)
x=-4:0.01:4;% Re-defining the value of x
y=sin(pi*(x+1)); % The exact solution of the example for n=2
plot(x,y,'>')    % Plotting the graph of the exact solution
hold on          % to sketch on the same plane
App_sol=-(68420679376088042598601588736 0*pi*(- 4*x.^3 +
3*x))/38302555212077620602527211014 59 -
(78409696367841911319668824473 6*pi*x)/38302555212077620602527211
01459;          % The approximate solution for n=2
plot(x,App_sol,'*') %Plotting the app_sol on the same plane
ax=[-1 1];  %Specifying axis for horizontal line (x-axis)
ay=[-1 1];  %Specifying axis to vertical line (y-axis)
axis([ax ay]); %Making square axis
title('The graph of exact and approximate solutions for n=2 and
m=4') % Providing the title of the graphs
legend('Exact solution','Approximate solution')%Legend the
graphs
xlabel('X-axis') %Labeling x axis
ylabel('Y-axis') %Labeling y axis
```

```
% The MATLAB code below is used to Sketch the graph of the
%approximate solution obtained for n=3, L=2 and M=4 together
%with the exact solution which will give fig1 (c)
x=-4:0.05:4;
y=sin(1.5*pi*(x+1));
plot(x,y,'-*')
hold on
App_sol=(46720017440079509877957442142208 0*pi*(2*x.^2 -
1))/60833448046247826530405098738370 09 -
(81775642094170428610096933109760 0*pi*(8*x.^4 - 8*x.^2 +
1))/60833448046247826530405098738370 09;
plot(x, App_sol,'->')
ax=[-1 1];
ay=[-1 1];
axis([ax ay]);
title('The graph of exact and approximate solutions for n=3 and
m=4')
legend('Exact solution','Approximate solution')
xlabel('X-axis')
ylabel('Y-axis')
```

```
% The MATLAB code below is used to Sketch the graph of the
%approximate solution obtained for n=2, L=2 and M=8 together
%with the exact solution which will give fig2 (a)
x=-4:0.05:4;
y=sin(pi*(x+1));
plot(x,y,'-*')
hold on
App_sol =-
(13772706043214508412137838344451854251702598108815732181565440*
pi*(- 4*x.^3 +
3*x))/64880741880155286782276516475805804483119065638408425630657061 -
(11756337154562238386577648113574038547960717448091817920692224*
pi*x)/64880741880155286782276516475805804483119065638408425630657061 -
(21543179198143299954938367977860848375702761065662811157299 20*pi*(16*x.^5 - 20*x.^3 +
5*x))/64880741880155286782276516475805804483119065638408425630657061 -
(13790318744505949912383477435993599869403237623164819210240 0*pi*(- 64*x.^7 + 112*x.^5 - 56*x.^3 +
7*x))/64880741880155286782276516475805804483119065638408425630657061;
```

```
plot(x, App_sol,'->')
ax=[-1 1];
ay=[-1.2 1.2];
axis([ax ay]);
title('The graph of exact and approximate solutions for n=2 and
m=8')
legend('Exact solution','Approximate solution')
xlabel('X-axis')
ylabel('Y-axis')
```

```
% The MATLAB code below is used to Sketch the graph of the
%approximate solution obtained for n=3, L=2 and M=8 together
%with the exact solution which will give fig2 (b)
```

```
x=-4:0.05:4;
y=sin(2*pi*(x+1));
plot(x,y,'-*')
hold on
App_sol=(213290524600426485660676768011162942505748161012743693926
40*pi*(- 4*x.^3 +
3*x))/834022192312786794696471798412580610246632602902698232583643 -
(116797302358892375113132939096891759784273454657158674120704*pi
*x)/834022192312786794696471798412580610246632602902698232583643
+
(196890566568135012396716343184294205381917515799169838612480*pi
*(16*x.^5 - 20*x.^3 +
5*x))/834022192312786794696471798412580610246632602902698232583643 +
(689515937225297495619173871799679993470161881158240960512000*pi*
(- 64*x.^7 + 112*x.^5 - 56*x.^3 +
7*x))/834022192312786794696471798412580610246632602902698232583643;
plot(x,App_sol,'->')
ax=[-1 1];
ay=[-1.2 1.2];
axis([ax ay]);
title('The graph of exact and approximate solutions for n=3 and
m=8')
legend('Exact solution','Approximate solution')
xlabel('X-axis')
ylabel('Y-axis')
```
