Eurasian J. Sci. Eng., 2026, 12(1), 23-38
https://doi.org/10.23918/eajse.v12i1p2
Published by Tishk International University
Available at: https://eajse.tiu.edu.iq/

RESEARCH ARTICLE **Open Access**

Shifted Chebyshev Neural Network Method for Solving Differential Equations

Sharmeen I Hasan[1*], and Salisu Ibrahim[2]

[1*]Department of Mathematics Education, Tishk International University, Iraq
[2]Department of information Technology, Tishk International University, Iraq

Article History
Received: 10.11.2025
Revised: 24.05.2026
Accepted: 04.06.2026
Published: 14.06.2026
Communicated by: Prof. Dr. Syed Abbas
**Email address:*
sharmeen.hasan@tiu.edu.iq
**Corresponding Author*

Abstract: The Lane–Emden equations, a class of nonlinear ordinary differential equations, are fundamental in modeling various physical phenomena, particularly in astrophysics for characterizing polytropic star structures.In this study, we provide a neural network-based model by incorporating Shifted Chebyshev Polynomials (SCPs) for solving Lane-Emden type equations. The universal approximation capacity of neural networks is harnessed in this model alongside the spectral accuracy of SCPs to efficiently handle the nonlinear and singular nature of these equations. Shifted Chebyshev Polynomials are embedded into the neural network structure to better capture solution behavior over the semi-infinite domain while naturally satisfying the required boundary conditions. A physics-informed loss function, constructed from the residuals of the governing differential equations, is minimized during training. Numerical experiments on classical Lane–Emden problems validate the proposed method, demonstrating high accuracy and convergence compared to existing analytical and numerical techniques. The results confirm that the neural network-SCP framework is a robust, effective , and flexible tool for solving complex nonlinear differential equations.

Keywords: *Chebyshev Polynomials; Shifted Chebyshev Polynomial; Ordinary Differential Equation, Nonlinear Equations, Numerical Approximation*

1. Introduction

Differential equations (DEs) form the cornerstone of modern physics, providing the fundamental framework for modeling real-world phenomena—from terrestrial fluid dynamics to stellar formation—often expressed as initial/boundary value problems involving 2^{nd}-order nonlinear ODEs. In astrophysics, Lane [1] and Emden [2] independently derived a seminal equation describing the self-gravitational equilibrium of gas with a constant polytropic index or uniform isothermal temperature. Known as the Lane-Emden equation, which take the generalization form as

$$\frac{d^2y}{dx^2}+\frac{2}{x}\frac{dy}{dx}+f(x,y)=g(x) \quad x\geqslant 0 \tag{1}$$

subject to the initial conditions (ICs) $y(0)=y_0$ and $y'(0)=0$, whereas x, y are nonlinear functions expressed by $f(x,y)$. The Lane-Emden type equation previously discussed exhibit a singular behavior at $x=0$. It is therefore possible to solve this kind of problem analytically in the neighbourhood of the singular point [**?**]. Several astrophysical phenomena can be modeled by the function $f(x,y)$ in Eq. (1), including stellar structure configurations, thermodynamic properties of spherical gas clouds, and equilibrium states of isothermal gas spheres. In practical applications, $f(x,y)$ is most widely expressed as

$$f(x,y) = y^m, \;\; y(0) = 1, \quad y'(0) = 0 \quad \text{and} \quad g(x) = 0. \tag{2}$$

Consequently, the nonlinear canonical Lane-Emden equation takes of the nonlinear ODE form.

$$\frac{d^2y}{dx^2} + \frac{2}{x}\frac{dy}{dx} + y^m = 0. \tag{3}$$

$$\Rightarrow \frac{1}{x^2}\frac{d}{dx}\left(x^2\frac{dy}{dx}\right) + y^m = 0,$$

with the polytropic index m governing the system's thermodynamic properties.
The Lane-Emden equation with the given ICs $y(0) = 1$ and $y'(0) = 0$, is a dimensionless form of Poisson's equation for a Newtonian, spherically symmetric, self-gravitating polytropic fluid. Eq. (3) shows how a spherical cloud of gas behaves thermally when its molecules are attracted to one another and when the classical rules of thermodynamics are applied. An other nonlinear representation of $f(x,y)$ is given by the exponential form

$$f(x,y) = e^y. \tag{4}$$

The functional form of Eq. (4) is substituted into Eq. (3) gives

$$\frac{d^2y}{dx^2} + \frac{2}{x}\frac{dy}{dx} + e^y = 0. \tag{5}$$

This characterizes isothermal gas spheres, where no change in temperature occurs. With $f(x,y) = e^{-y}$, Eq. (3) becomes

$$\frac{d^2y}{dx^2} + \frac{2}{x}\frac{dy}{dx} + e^{-y} = 0. \tag{6}$$

This formulation provides a fundamental model for thermionic current theory, originally characterized in detail by Richardson's pioneering work [3]. For m = 0, 1, and 5, Chandrasekhar [4] and Datta [5] have obtained exact solutions of Eq. (3). [6] In the case where m = 5, the system admits just a one-parameter family of solutions. The standard Lane-Emden equations generally require numerical solution methods for arbitrary values of m. The inherent singularity at the origin $x = 0$ presents significant computational challenges, affecting both linear and nonlinear initial value problems in quantum mechanical systems and astrophysical modeling, including the Lane-Emden equations themselves. Analytical methods of the Lane–Emden equations typically rely on perturbation theory or series expansion methods to derive solutions. The authors Dehghan and Shakeri have developed the variational iteration approach for approximating the solution of a DE that arises in astrophysics [7]. In order to solve the Emden-Fowler problem, Govinder and Leach [8] suggested the Lie and Painleve analysis procedures. Singh et al. have introduced an effective analytical technique that is utilizing an an adapted homotopy analysis approach [9]. Muatjetjeja and Khalique in [10], successfully solved the generalized Lane-Emden equations of the first and second types.
Recently, Various machine learning and artificial intelligence techniques have been applied to solve initial and boundary value problems, especially Artificial Neural Networks (ANN). There are many

advantages of using ANNs for approximate solutions. Regardless of the problem's dimensions, ANN trial solutions only use one independent variable. Throughout the whole integration domain, the approximate solutions are continuous. Additionally, most other numerical techniques are iterative in nature, the step size must be set before the computation begins. The step size must be set before the computation starts because approximate methods are iterative in nature. If, after obtaining the solution, we want to know it in between steps, we must restart the process from the beginning. One solution that could help us get past this iteration cycle is ANNs. We may also make use of it a black-box tool to generate numerical results at arbitrary points within the domain. Meade and Fernandez [11, 12], investigated by employing the use of linear B1-splines as basis functions in a feedforward neural network model, Lagaris et al. [13],combined the Broyden-Fletcher-Goldfarb-Shanno (BFGS) quasi-Newton optimization approach and neural networks in order to solve both ODEs and PDEs.Additionally, multilayer perceptrons were implemented within the network architecture to resolve boundary value problems associated with irregular geometries[14]. While the neural networks-based methods have become popular for solving ODEs and PDEs, traditional methods like modified cubic B-Spline finite element technique are still significant because of their guaranteed existence, uniqueness, and convergence properties [15], and semi-implicit finite difference method for the Cahn-Allen equation that preserves stability and convergence [16]. For solving both lower and higher order ODEs, A hybrid strategy based on ANNs and optimization techniques was presented by Maleck and Shekari Beidokhti [17]. A variation of the kernel least mean squares method that is unsupervised has been proposed by Yazid et al. [18], for the solution of ODEs of the first and second orders. New methodologies for matrix Riccati differential equation solutions were developed by Selvaraju and Abdul Samant [19]. Multilayer perceptrons and radial basis function (RBF) neural networks with a novel unsupervised training technique have been proposed in [20] as numerical solution of the nonlinear Schrodinger problem. Aarts and Van der Veer used evolutionary algorithms to evaluate PDE and IVPs in [21]. He et al. [22] used feed forward neural networks in combination with the extended back propagation algorithm to solve a class of first-order PDEs. Hoda and Nagla. [23] have proposed an alternative method for resolving mixed BVPs over irregular domains. An ANN approach for solving mixed boundary value issues with irregular domains was introduced by Mcfall and Mahan [24]. In order to solve DEs using neural networks, Leephakpreeda [25] devised a fuzzy language model. The neural network model has been investigated by Manevitz et al. [26] to resolve the mesh adaptation problem for time-dependent PDE using the finite-element method. Mai-Duy, Tran-Cong's and other in [27] used multi-quadric radial basis function neural networks to solve elliptic PDEs, ODEs and linear differential equations without the need for meshes.The work of Jianyu et al. [28] introduced a computational methodology employing radial basis neural networks to obtain numerical solutions to elliptic PDEs. To solve Fredholm integro-differential equations numerically, Jackiewicz et al. [29] developed a computational system based on neural networks. To be able to solve the problem of unstable solid-gas reactors, Parisi et al. [30] employed unsupervised neural networks. Radial basis function neural networks and multilayer perceptrons were investigated by Kumar and Yadav to solve differential equations [31]. A neural network approach that uses regression was proposed by Mall and Chakraverty [32] to solve ODEs. Recently, physics-informed neural networks (PINNs) have been proposed as a tool to solve nonlinear PDEs, but they have been shown to be ineffective for stiff problems and infinite domains. To address these challenges, spectrally adapted PINNs are introduced to incorporate adaptive spectral methods and PINNs to solve well the problems of unbounded domain efficiently [33]. Similarly, the self-adaptive PINNs (SA-PINNs) [34] incorporate trainable weights for each training point to automatically select the more challenging solution regions, resulting in improved accuracy compared to other state-of-the-art PINNs algorithms with fewer training epochs.

The experiments for harder nonlinear problems like $m = 3$ or $m = 5$ would further support the work, as would be a full comparison with modern Physics-Informed Neural Networks (PINNs), adaptive deep-learning solvers, the convergence analysis, CPU-time comparison, complexity analysis, or a robustness study; work on the latter would be helpful. In the present study, due to scope limitation, a complete implementation of PINN, an analysis of its convergence, CPU-time comparison, complexity analysis and robustness study for more challenging nonlinear cases, e.g., $m = 3$ and $m = 5$ has not been performed.

The conventional methods of ChNN are focused mainly on general function approximation; the proposed method of SChNN is used for the specific integration of shifted spectral basis functions in the structure of the neural network training for nonlinear singular boundary value problems. The hybridization not only enhances the representation capability of the network but also maintains the orthogonality and convergence properties of shifted Chebyshev polynomials. Furthermore, it is more compatible with the boundary conditions imposed on the positive domain and also increased the numerical stability around the singular points. Reducing the complexity of mapping, and transformation errors.

2. Preliminary

This section provides definitions and fundamental characteristics of the shifted Chebyshev polynomials and Shifted Chebyshev Neural Network that will be used to support the suggested method.

2.1 Shifted Chebyshev Polynomial

Modeling a variety of physical processes, such as chemical reactions and spring-mass systems requires the use of nonlinear differential equations. In order to find solutions for NDEs, reference [35] uses the Shifted Chebyshev collocation matrix approach. By applying the method, the problem is converted into a matrix equation comprising nonlinear algebraic equations with Shifted Chebyshev coefficients as unknowns. The least square methods was introduced in [36], [37]. The authors in [38] investigate the shifted Chebyshev approach to high-order ODEs. The shifted Chebyshev collocation method for nonlinear differential equations is improved in this study.

$$\sum_{k=0}^{m}\sum_{s=0}^{n} Q_{k,s}(x)y^{s}(x)y^{(k)}(x) + \sum_{k=1}^{m}\sum_{s=1}^{m} P_{k,s}(x)y^{(s)}(x)y^{(k)}(x) = f(x), \tag{7}$$

subject to conditions

$$\sum_{k=0}^{m-1}\left(a_{ik}y^{(k)}(a) + b_{ik}y^{(k)}(b) + c_{ik}y^{(k)}(c)\right) = \alpha_i, \quad i = 0,1,\ldots,m-1, \tag{8}$$

in which the functions $y^{(0)}(x) = y(x)$ and $y(x) \in C^m[0,L]$ are unknown parameters. The known values $Q_{k,s}(x), P_{k,s}(x)$ and $f(x)$ are determined on interval $[0,L]$.

The truncated shifted Chebyshev series will be used to display the numerical solution of Eq. (7) from Eq. (8).

$$y_N(x) = \sum_{r=0}^{N} a_r T^{*}_{L,r}(x), \quad x \in [0,L]. \tag{9}$$

With $N > m$ and N might be any positive integer. However, the following recurrence relationship can also be used to produce $r = 0,1,\ldots,N$ for shifted Chebyshev polynomials, we denote $T^{*}_{L,r}(x)$.:

$$T^{*}_{L,r+1}(x) = 2\left(\frac{2x}{L} - 1\right)T^{*}_{L,r}(x) - T^{*}_{L,r-1}(x), \;\; r = 1,2,\ldots$$

When $T^{*}_{L,0}(x) = 1,\; T^{*}_{L,1}(x) = \frac{2x}{L} - 1$. The shifted Chebyshev polynomilas $T^{*}_{L,r}(x)$ of degree r has the following exact form:

$$T^*_{L,r}(x) = r\sum_{p=0}^{r}(-1)^{r-p}\ \frac{(r+p-1)!(2)^{2p}\ x^p}{(r-p)!(2p)!Lp} \tag{10}$$

For $T^*_{L,r}(L) = 1$ and $T^*_{L,r}(0) = (-1)^r$. The condition of orthogonality shown as

$$\int_0^L T^*_{L,j}(x)T^*_{L,i}(x)w_L(x)dx = h_k\delta_{ji},$$

where $h_i = b_i\pi/2, b_0 = 2, b_i = 1, k \geq 1$, and $w_L(x) = \left(Lx - x^2\right)^{-1/2}$. using Eq. (7),the k-th derivatives of $T^*_{L,r}(x)$ are obtained .

$$\left(T^*_{L,r}\right)^{(k)}(x) = T^{*,\ k}_{L,r}(x) = r\sum_{p=m}^{r}(-1)^{r-p}\ p(p-1)\dots(p-k+1)\ \frac{(r+p-1)!(2)^{2p}\ x^{p-k}}{(r-p)!(2p)!L^k}, \tag{11}$$

where $p \geq m-1$.

2.2 Shifted Chebyshev polynomial Method

To determine the k-th derivatives of the approximate solution $y_N(x)$, we use Eq. (9).

$$y_N^{(k)}(x) = \sum_{r=0}^{N} a_r\left(T^*_{L,r}\right)^{(k)}(x) = \sum_{r=k}^{N} a_r T^{*,k}_{L,r}(x). \tag{12}$$

From (7), (9) and (12), we get

$$\begin{aligned}&\sum_{k=0}^{m}\sum_{s=0}^{n} Q_{k,s}(x)\left(\sum_{r=0}^{N} a_r T^*_{L,r}(x)\right)^s\left(\sum_{r=k}^{N} a_r T^{*,k}_{L,r}\right)\\ &\quad+\sum_{k=1}^{m}\sum_{s=1}^{m} P_{k,s}(x)\left(\sum_{r=s}^{N} a_r T^{*,s}_{L,r}\right)\left(\sum_{r=k}^{N} a_r T^{*,k}_{L,r}\right) = f(x).\end{aligned} \tag{13}$$

Collocating Eq. (13) at $N-m+1$ points $x_p, p = 0,1,\dots,N-m$ leads to

$$\begin{aligned}&\sum_{k=0}^{m}\sum_{s=0}^{n} Q_{k,s}(x_q)\left(\sum_{r=0}^{N} a_r T^*_{L,r}(x_q)\right)^s\left(\sum_{r=k}^{N} a_r T^{*,k}_{L,r}(x_q)\right)\\ &\quad+\sum_{k=1}^{m}\sum_{s=1}^{m} P_{k,s}(x_q)\left(\sum_{r=s}^{N} a_r T^{*,s}_{L,r}(x_q)\right)\left(\sum_{r=k}^{N} a_r T^{*,k}_{L,r}(x_q)\right) = f(x_q).\end{aligned} \tag{14}$$

Where x_p is the root of $T^*_{L,m}(x)$. Furthermore, the following k equations are obtained by applying Eq. (12) to the conditions in Eq. (8):

$$\sum_{k=0}^{m-1}\left(a_{ik}\sum_{r=k}^{N} a_r T^{*,k}_{L,r}(a) + b_{ik}\sum_{r=0}^{N} a_r T^{*,k}_{L,r}(b) + c_{ik}\sum_{r=0}^{N} a_r T^{*,k}_{L,r}(c)\right) = \alpha_i, \quad i = 0,1,\dots,m-1. \tag{15}$$

Eq. (14), together with ks- equations from the conditions Eq. (15) give ($N+1$) nonlinear algebraic equations (NAEs). The NAEs are solved to get the unknown shifted Chebyshev coefficients. N a_r ($r = 0,1,\dots$). Consequently, With the use of calculated $y_N(x)$ we can have an estimate of $y(x)$ given in Eq. (7).

2.3 The architectural framework of Shifted Chebyshev Neural Network

The single-layer Shifted Chebyshev Neural Network (SChNN) model is made up of a Shifted Chebyshev polynomial expansion block, an input node and an output node. To improve the input structure, shifted Chebyshev polynomials are applied within the numerical transformation and learning segments of the model. Considering the input data as $x = (x_1, x_2, \ldots, x_h)^T$, where x consists of h data points and Shifted Chebyshev polynomials, which are orthogonal and also obtained as solutions to the shifted Chebyshev DEs. we referred the first two pair of Shifted Chebyshev polynomials as

$$T_0^*(x) = 1,$$
$$T_1^*(x) = x.$$

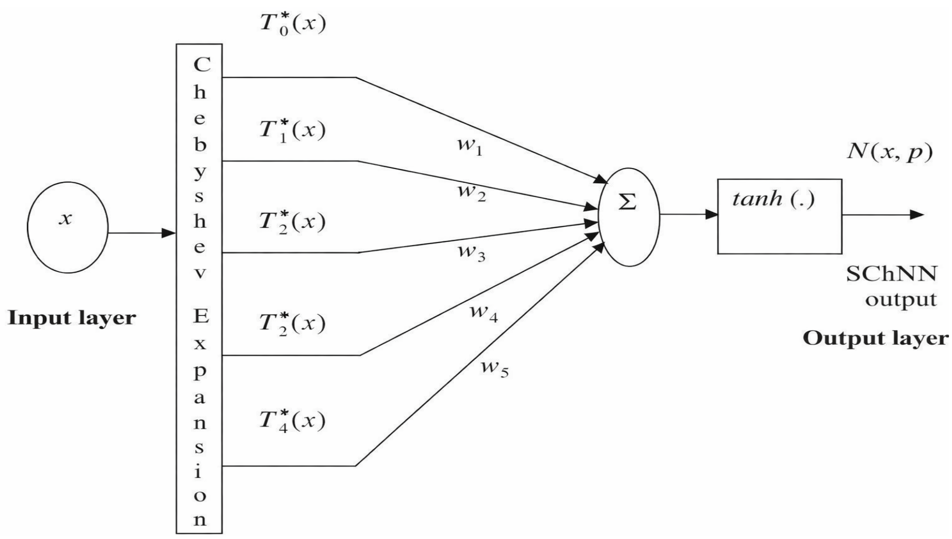

Figure 1: architectural framework of Shifted Chebyshev Neural Network.

The well-known recursive formula can be used to construct the higher order Shifted Chebyshev polynomials.

$$T_{r+1}^*(x) = 2xT_r^*(x) - T_{r-1}^*(x), \tag{16}$$

when $T_r^*(x)$ represents a Shifted Chebyshev polynomial of rth order. Here, m dimensional enhanced Shifted Chebyshev polynomials are created by expanding the n dimensional input pattern. The SChNN's advantage is that it uses a single layer network to produce the desired outcome. Even if the Shifted Chebyshev polynomial is used to increase the input's size.
Figure 1 illustrates the network architecture featuring a single-node input layer, a single-node output layer, and the first five Shifted Chebyshev polynomials.

The input variable x represented as the start point of the proposed network expanded with Shifted Chebyshev polynomials to come up with a set of transformed features. Then a corresponding weight multiplied with each of these features and added through a weighted summation in the $\sum$ node. The resulting value is transmitted by use of a hyperbolic tangent. *tanh* activation, which is the addition of nonlinearity to the network, and allows the network to approximate complicated relationship structure. The result of this activation function, which is denoted as $N(x, p)$, is the last output of the network, or the SChNN output. In this architecture, Shifted Chebyshev expansion can be integrated with neural processing in an effective way in order to improve approximation and learning.

Note: The notations has the same meaning since they are all referred to the "shifted chebyshev polynomial". The notation $T^*_{L,r}(x)$ and $T^*_r(x)$, are the definition of shifted chebyshev polynomial, while the notation $T^*_{j-1}(x)$ is the definition of "shifted chebyshev polynomial involving the neural network".

3. Shifted Chebyshev Neural Network for ODE Solutions

In this section, we aim to apply the SChP alongside with the neural network techniques to come up with a numerical neural method that can be used to solve nonlinear DEs. Moreover, we will analogously going to consider the L2 norm to find the minimum error.

3.1 SChNN learning algorithm

The learning algorithm minimizes the error function through gradient-based optimization of network parameters p. Using error back propagation, the SChNN iteratively calculates the error gradient with respect to p in order to update its weights. $(\tanh(z))$ is the activation function, which is the tangent hyperbolic function.
With input data x and parameters (weights) p, the network output can be calculated as following

$$N(x,p) = \tanh(z). \tag{17}$$

The linear sum z in this case is given as

$$z = \sum_{j=1}^{m} w_j T^*_{j-1}(x), \tag{18}$$

where $T^*_{j-1}(x)$ is the SChP of the neural network, $x = (x_1, x_2, \ldots, x_h)^T$ represent the input data, and the weight vectors is the w_j with $j = \{1,2,3,\ldots,m\}$.
Now, the back propagation principle can be used to modify the weights of SChNN.

$$w_j^{k+1} = w_j^k + \Delta w_j^k = w_j^k + \left(-\eta \frac{\partial E(x,p)^k}{\partial w_j^k}\right), \tag{19}$$

where k is the iteration step that is used to update the weights in an ANN as usual, $E(x,p)$ is the error function, and η is the learning parameter.
Following this, we focus on establishing ANN architectures for general DE systems, after which we investigate a general DE that can be used to express partial or ordinary differential equations.

$$G\left(x, y(x), \nabla y(x), \nabla^2 y(x), \ldots, \nabla^n y(x)\right) = 0, \quad x \in \bar{D} \subseteq R^n, \tag{20}$$

where $\bar{D}$ is the discretized domain over a finite set of points, ∇ is the differential operator, $y(x)$ is the solution, and G is the function that establishes the structure of the DE. For ODEs, $x \in \bar{D} \subset R$, and for partial differential equations, $x = (x_1, x_2, \ldots, x_n) \in \bar{D} \subset R^n$. The following general DE takes on a different form where the trial solution with modifiable parameters (weights) p is denoted by $y_t(x,p)$.

$$G\left(x, y_t(x,p), \nabla y_t(x,p), \nabla^2 y_t(x,p), \ldots, \nabla^n y_t(x,p)\right) = 0. \tag{21}$$

The following minimization problem is obtained from the corresponding problem.

$$\min_{p} \sum_{x \in D} \frac{1}{2} \left(G\left(x, y_t(x,p), \nabla y_t(x,p), \nabla^2 y_t(x,p), \ldots, \nabla^n y_t(x,p)\right)\right)^2. \tag{22}$$

3.2 The proposed Shifted Chebyshev Neural Network (SChNN) algorithm

In the SChNN framework, the parameterized trial function $y_t(x,p)$ with parameters p is formulated as

$$y_t(x,p) = A(x) + F(x, N(x,p)). \tag{23}$$

The initial/boundary conditions are firmly enforced by the first term $A(x)$ without tunable parameters, while the second term $F(x,N(x,p))$ represents the SChNN's parameterized output that processes the input x through adjustable weights p. As previously discussed a single-layer SChNN, characterized by input x and parameters p, comprises one input and one output node, expressed as $N(x,p)$.

$$N(x,p) = \tanh(z), \tag{24}$$

whereas the $T^*_{j-1}(x)$ indicates the corresponding set of Shifted Chebyshev polynomials as well as $z = \sum_{j=1}^{m} w_j T^*_{j-1}(x)$ and w_j is the weight vectors of the SChNN.
The general form of the ODE's associated error function can be expressed by

$$E(x,p) = \sum_{i=1}^{h} \frac{1}{2} \left\{ \frac{d^n y_t(x_i,p)}{dx^n} - f\left(x_i, y_t(x_i), \frac{dy_t(x_i,p)}{dx}, \ldots, \frac{d^{n-1} y_t(x_i,p)}{dx^{n-1}}\right)\right\}^2. \tag{25}$$

To minimize the error function $E(x,p)$, we calculate its derivative concerning the network parameters and update the weights accordingly. As a result, the network's output gradient in relation to its inputs is calculated.
As our focus is to address the Lane–Emden type second-order DEs, let's now turn our attention to the general formulation of second-order ODEs.
The DE that has been focused on can be expressed as

$$\frac{d^2y}{dx^2} = f\left(x, y, \frac{dy}{dx}\right) x \in [a,b].$$

The solution for the SChNN trail with the ICs $y(a) = A, \;\; y'(a) = A'$
can be expressed as

$$y_t(x,p) = A + A'(x-a) + (x-a)^2 N(x,p), \tag{26}$$

at which $N(x,p)$ is the SChNNs output along with x and p are its only inputs. In terms of ICs, the trial solution $y_t(x,p)$ is satisfied.
In order to reduce the error function, it is represented as

$$E(x,p) = \sum_{i=1}^{h} \frac{1}{2} \left(\frac{d^2 y_t(x_i,p)}{dx^2} - f\left[x_i, y_t(x_i,p), \frac{dy_t(x_i,p)}{dx}\right]\right)^2. \tag{27}$$

The following rule determines how the weights of the input and output layers are changed.

$$w_j^{k+1} = w_j^k + \Delta w_j^k = w_j^k + \left(-\eta \frac{\partial E(x,p)^k}{\partial w_j^k}\right). \tag{28}$$

Here

$$\frac{\partial E(x,p)}{\partial w_j} = \frac{\partial}{\partial w_j}\left(\sum_{i=1}^{h}\frac{1}{2}\left(\frac{d^2 y_t(x_i,p)}{dx^2} - f\left[x_i, y_t(x_i,p), \frac{dy_t(x_i,p)}{dx}\right]\right)^2\right). \tag{29}$$

Lastly, the approximate solutions can be obtained by plugging the converged SChNN results into Eq. (26).

4. Evaluation and Interpretation of the results

In this part of the work, we study both the homogeneous and the non-homogeneous Lane-Emden equations to show how well our new method performs on each one. The model is trained with different sets of training points such as, 10, 15 and 20 because the speed at which it converges really depends on the problem in front of it. For every example we present exactly how many points we chose, since that choice is what balances accuracy with the overall quality of the results.
Homogeneous Lane-Emden Equations

$$\frac{d^2y}{dx^2} + \frac{2}{x}\frac{dy}{dx} + y^m = 0 \quad x \geqslant 0,$$

subject to $y(0) = 1, y'(0) = 0$ conditions.

4.1 Example 1

For $m = 0$, the equation becomes linear ODE

$$\frac{d^2y}{dx^2} + \frac{2}{x}\frac{dy}{dx} + 1 = 0, \tag{30}$$

subject to $y(0) = 1,\ \ y'(0) = 0$ conditions.
The Analytical solution of Eq. (30) can be defined as

$$y(x) = 1 - \frac{x^2}{6}. \tag{31}$$

The SChNN trial solution is written as

$$y_t(x,p) = 1 + x^2 N(x,p) \tag{32}$$

The following weight function was obtained after solving the nonlinear neural network equations.

(33)
$$w_1 = -0.166667,\ \ w_2 = 0,\ \ w_3 = -8.32667\times10^{-17},\ \ w_4 = -8.32667\times10^{-17},\ \ w_5 = 5.5511\times10^{-17}.$$

And the approximate solution is given as

$$y_N = 1 + x^2\text{Tanh}\left[0.166667 + 2.6090\times10^{-15}x - 1.22125\times10^{-14}x^2 + 1.68754\times10^{-14}x^3 - 7.10543\times10^{-15}x^4\right]. \tag{34}$$

The training process utilizes ten points, equally distributed over the interval $[0,1]$ with the first five Shifted Chebyshev polynomials. Comparison between analytical and Chebyshev neural solutions and SChNN solutions with random weights is provided in Table 1. Comparison between analytical and

SChNN results is illustrated in Figure 2. Finally, plots of error between analytical and SChNN results are illustrated.

The graph is shown in the figures below is the analytical with approximate solutions, and error for $N = 6$.

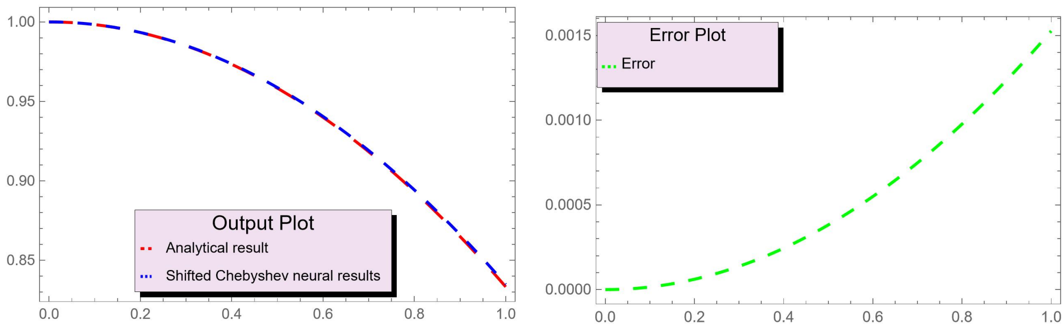

Figure 2: Plot illustrating the exact and approximate solutions and corresponding error for $N = 6$.

Table 1: Error of Example 1 for $N = 6$.

	Analytical result	ChNN result	Shifted ChNN result	ChNN Errors	Shifted ChNN Errors
0	1	1.	1	0	0
0.1	0.99833	0.9993	0.99835	0.0010	0.0000152625
0.2	0.99333	0.9901	0.99339	0.003	0.0000610501
0.3	0.985	0.9822	0.98514	0.0028	0.000137363
0.4	0.9733	0.9766	0.97358	0.0033	0.000244201
0.5	0.95833	0.9602	0.95972	0.0019	0.000381563
0.6	0.94	0.9454	0.94055	0.0054	0.000549451
0.7	0.91833	0.9139	0.9191	0.0044	0.000747864
0.8	0.89333	0.8892	0.8943	0.0041	0.000976802
0.9	0.865	0.8633	0.86624	0.0017	0.00123627
1	5/6	0.8322	0.83486	0.0011	0.00152625 .

In order to prove the efficiency of the proposed SChNN method, the results obtained are compared with those of the previous studies that adopted other numerical methods. The proposed approach gives much smaller errors than those in ChNN method [39] as illustrated in Table 1. It means that the hybrid Shifted Chebyshev polynomial neural network methodology is better at resolving the specified ODE problems and demonstrate the effectiveness and reliability of the proposed technique.

4.2 Example 2

Let us consider Lane-Emden equation for $m = 1$

$$\frac{d^2y}{dx^2}+\frac{2}{x}\frac{dy}{dx}+y=0, \tag{35}$$

with ICs $y(0) = 1,\ y'(0) = 0$.
The Analytical solution of Eq. (35) can be given as

$$y(x)=\frac{\sin(x)}{x}. \tag{36}$$

The trial solution of SChNN is written as

$$y_t(x,p)=1+x^2N(x,p). \tag{37}$$

The following weight function was obtain after solving the nonlinear neural network equations.

$$\begin{aligned} w_1=-0.1635953,\ \ w_2=0.0040809,\ \ w_3=0.0009989\times10^{-17}\\ w_4=1.4036326,\ \ w_5=1.4036325513833159\times10^{-6}. \end{aligned} \tag{38}$$

And the approximate solution is given as

$$\begin{aligned} y_N=1-x^2\text{Tanh}\,\big[0.166664+0.000094x+0.00880x^2\\ +0.00074x^3-0.0001796x^4\big]\,. \end{aligned} \tag{39}$$

The training process utilizes ten points, equally distributed over the interval $[0,1]$ with the first five Shifted Chebyshev polynomials. Comparison between analytical and Chebyshev neural solutions and SChNN solutions with random weights is provided in Table 2. Comparison between analytical and

SChNN results is illustrated in Figure 3. Finally, plots of error between analytical and SChNN results are illustrated.
The graph is shown in the figures below is the analytical with approximate solutions, and error $N = 5$.

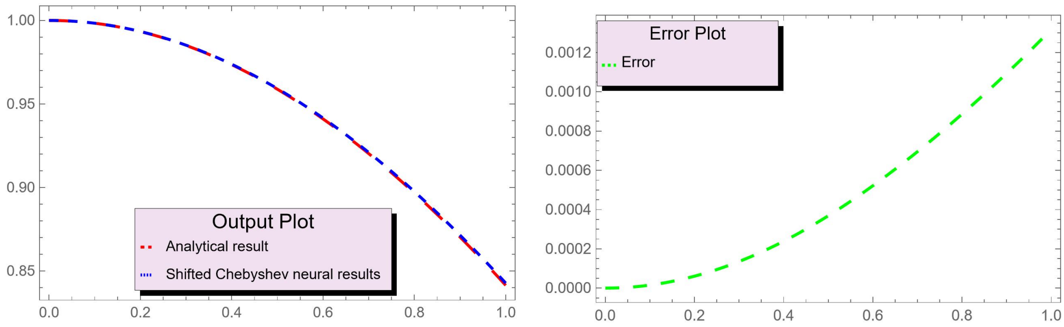

Figure 3: Plot of Example 2 of exact with approximate solutions, and corresponding error of $N = 5$

Table 2: Error of Example 2 for $N = 5$.

	Analytical result	ChNN result	Shifted ChNN result	ChNN Error	Shifted ChNN Errors
0	1	1.	1	0	0
0.1	0.998334	0.9975	0.998349	0.0035	0.000015216
0.2	0.99347	0.9905	0.993407	0.0028	0.000060591
0.3	0.98507	0.9839	0.985203	0.0012	0.00013551
0.4	0.97355	0.9734	0.973785	0.0001	0.000238753
0.5	0.95885	0.9598	0.959219	0.0009	0.000368309
0.6	0.94107	0.9417	0.941592	0.0006	0.000521456
0.7	0.92031	0.9210	0.921006	0.0007	0.000695116
0.8	0.896695	0.8925	0.897582	0.0042	0.000886457
0.9	0.87036	0.8700	0.871457	0.0004	0.00109376
1	Sin[1]	0.8431	0.842789	0.0016	0.00131754 .

An analogous comparison is performed with regards to the Example 2, as referred to in Table 2. The obtained numerical results of the proposed SChNN method are compared with the ones provided in [39] of the same problem. As it can be seen, the proposed approach has smaller numerical errors, which once again testifies to the validity and efficacy of the suggested method.

5. Concluding remarks

In this study, we leverage the powerful approximation properties of Shifted Chebyshev Polynomials to develop a model based on neural networks for solving Lane-Emden type equations.The proposed approach effectively combined the learning capability of neural networks with the spectral accuracy of Chebyshev polynomials, resulting in a highly accurate, stable, and efficient computational method. Through several benchmark examples, the model demonstrated excellent performance in approximating solutions to nonlinear Lane–Emden equations, often outperforming traditional numerical methods. The results confirm that this hybrid technique not only handles the singular behavior at the origin but also maintains high precision across the domain. Future work could extend this framework to address more complex boundary conditions, higher-dimensional problems, and systems of differential equations, further enhancing its applicability in mathematical physics and astrophysics

6. Conclusion and Discussion

The authors are asked to write a section at the end of the research that summarizes their findings, explains the contribution to novel science in the field, and explains why this research is more important than other recent research. Moreover, we encourage the authors to present some concrete open problems at the end of the conclusion for readers and young researchers.

Author's Contribution

There should be a declaration of authorship, originality, approval by the authors, and the authors' contribution loads. The example can be used instead.
"We confirm that all named authors have read and approved the manuscript. We also confirm that each author has the same contribution to the paper. We further confirm that all authors have approved the order of authors listed in the manuscript."

Conflict of Interest

The journal also asks for a declaration about conflict of interest as *"There is no conflict of interest for this paper."*

Acknowledgment

Authors are also asked to include an acknowledgment section for external contributors, such as an external supervisor, reviewers, and the journal's editorial team. Authors are not allowed to acknowledge each other.

Use of AI Tools Declaration

The authors declare that any AI tools used in the preparation of this manuscript were limited to language and readability improvement only, and were not used to generate scientific content, data, analyses, or conclusions, with full responsibility retained by the authors
or
The authors declare that no AI tools were used in any part of the preparation of this manuscript

References

[1] Lane HJ. On the theoretical temperature of the sun, under the hypothesis of a gaseous mass maintaining its volume by its internal heat, and depending on the laws of gases as known to terrestrial experiment. American Journal of Science. 1870 Jul 1;2(148). https://doi.org/10.2475/ajs.s2-50.148.57

[2] Emden R. Gaskugeln: Anwendungen der mechanischen Warmetheorie auf kosmologische und meteorologische Probleme. BG Teubner; 1907.

[3] Richardson OW. The emission of electricity from hot bodies. Longmans, Green and Company; 1921.

[4] Chandrasekhar S. Principles of stellar dynamics. Courier Corporation; 2005 May 13.

[5] Datta BK. Analytic solution to the Lane–Emden equation. Nuovo Cimento B. 1996;111(11):1385–8.

[6] Dresner L. Similarity solutions of nonlinear partial differential equations; 1983 Jan.

[7] Dehghan M, Shakeri F. Approximate solution of a differential equation arising in astrophysics using the variational iteration method. New Astronomy. 2008 Jan 1;13(1):53-9. https://doi.org/10.1016/j.newast.2007.06.012

[8] Govinder KS, Leach PG. Integrability analysis of the Emden-Fowler equation. Journal of Nonlinear Mathematical Physics. 2007 Jan 1;14(3):443-61. https://doi.org/10.2991/jnmp.2007.14.3.10

[9] Singh OP, Pandey RK, Singh VK. An analytic algorithm of Lane–Emden type equations arising in astrophysics using modified homotopy analysis method. Computer Physics Communications. 2009 Jul 1;180(7):1116-24. https://doi.org/10.1016/j.cpc.2009.01.012

[10] Muatjetjeja B, Khalique CM. Exact solutions of the generalized Lane–Emden equations of the first and second kind. Pramana. 2011 Sep;77(3):545-54. https://doi.org/10.1007/s12043-011-0174-4

[11] Wazwaz AM. The modified decomposition method for analytic treatment of differential equations. Applied mathematics and computation. 2006 Feb 1;173(1):165-76. https://doi.org/10.1016/j.amc.2005.02.048

[12] Shawagfeh NT. Nonperturbative approximate solution for Lane-Emden equation. Journal of Mathematical Physics. 1993 Sep 1;34(9):4364. https://doi.org/10.1063/1.530005

[13] Lagaris IE, Likas A, Fotiadis DI. Artificial neural networks for solving ordinary and partial differential equations. IEEE transactions on neural networks. 1998 Sep 30;9(5):987-1000. https://doi.org/10.1109/72.712178

[14] Lagaris IE, Likas AC, Papageorgiou DG. Neural-network methods for boundary value problems with irregular boundaries. IEEE transactions on neural networks. 2000 Sep 30;11(5):1041-9. https://doi.org/10.1109/72.870037

[15] O Hussen B, A Sabawi Y. B-Spline Cubic Finite Element Method for Solving Ordinary Differential Equations. Eurasian Journal of Science and Engineering. 2022;8(3). https://doi.org/10.23918/eajse.v8i3p39

[16] Sabawi YA, Ahmed SB, Hamad HQ. Numerical treatment of Allen's equation using semi implicit finite difference methods. Eurasian Journal of Science and Engineering. 2022 Jun 2;8(1):90-100. https://doi.org/10.23918/eajse.v8i1p90

[17] Malek A, Beidokhti RS. Numerical solution for high order differential equations using a hybrid neural network—optimization method. Applied mathematics and computation. 2006 Dec 1;183(1):260-71. https://doi.org/10.1016/j.amc.2006.05.068

[18] Yazdi HS, Pakdaman M, Modaghegh H. Unsupervised kernel least mean square algorithm for solving ordinary differential equations. Neurocomputing. 2011 Jun 1;74(12-13):2062-71. https://doi.org/10.1016/j.neucom.2010.12.026

[19] Samath JA, Selvaraju N. Solution of matrix Riccati differential equation for nonlinear singular system using neural networks. International Journal of Computer Applications. 2010 Feb;1(29):48-54. https://doi.org/10.5120/575-181

[20] Shirvany Y, Hayati M, Moradian R. Multilayer perceptron neural networks with novel unsupervised training method for numerical solution of the partial differential equations. Applied Soft Computing. 2009 Jan 1;9(1):20-9. https://doi.org/10.1016/j.asoc.2008.02.003

[21] Aarts LP, Van Der Veer P. Neural network method for solving partial differential equations. Neural processing letters. 2001 Dec;14(3):261-71. https://doi.org/10.1023/A:1012784129883

[22] He S, Reif K, Unbehauen R. Multilayer neural networks for solving a class of partial differential equations. Neural networks. 2000 Apr 1;13(3):385-96. https://doi.org/10.1016/S0893-6080(00)00013-7

[23] Hoda SA, Nagla HA. Neural network methods for mixed boundary value problems. International Journal of Nonlinear Science. 2011 Mar;11(3):312-6.

[24] McFall KS, Mahan JR. Artificial neural network method for solution of boundary value problems with exact satisfaction of arbitrary boundary conditions. IEEE transactions on neural networks. 2009 Jun 2;20(8):1221-33. https://doi.org/10.1109/TNN.2009.2020735

[25] Leephakpreeda T. Novel determination of differential-equation solutions: universal approximation method. Journal of computational and applied mathematics. 2002 Sep 15;146(2):443-57. https://doi.org/10.1016/S0377-0427(02)00397-7

[26] Manevitz L, Bitar A, Givoli D. Neural network time series forecasting of finite-element mesh adaptation. Neurocomputing. 2005 Jan 1;63:447-63. https://doi.org/10.1016/j.neucom.2004.06.009

[27] Mai-Duy N, Tran-Cong T. Numerical solution of differential equations using multiquadric radial basis function networks. Neural networks. 2001 Mar 1;14(2):185-99. https://doi.org/10.1016/S0893-6080(00)00095-2

[28] Jianyu L, Siwei L, Yingjian Q, Yaping H. Numerical solution of elliptic partial differential equation using radial basis function neural networks. Neural Networks. 2003 Jun 1;16(5-6):729-34. https://doi.org/10.1016/S0893-6080(03)00083-2

[29] Jackiewiez BW, Rahaman M. Numerical solution of a fredholm integro-differential equation modelling formula not shown neural networks. Applied Mathematical And Computation. 2008;195(3):2523-5363. https://doi.org/10.1016/j.amc.2007.05.031

[30] Parisi DR, Mariani MC, Laborde MA. Solving differential equations with unsupervised neural networks. Chemical engineering and processing: Process intensification. 2003 Aug 1;42(8-9):715-21. https://doi.org/10.1016/S0255-2701(02)00207-6

[31] Kumar M, Yadav N. Multilayer perceptrons and radial basis function neural network methods for the solution of differential equations: a survey. Computers & mathematics with applications. 2011 Nov 1;62(10):3796-811. https://doi.org/10.1016/j.camwa.2011.09.028

[32] Mall S, Chakraverty S. Regression-based neural network training for the solution of ordinary differential equations. International Journal of Mathematical Modelling and Numerical Optimisation. 2013 Jan 1;4(2):136-49. https://doi.org/10.1504/IJMMNO.2013.055203

[33] Xia M, Böttcher L, Chou T. Spectrally adapted physics-informed neural networks for solving unbounded domain problems. Machine Learning: Science and Technology. 2023 Jun 1;4(2):025024. 10.1088/2632-2153/acd0a1

[34] McClenny LD, Braga-Neto UM. Self-adaptive physics-informed neural networks. Journal of Computational Physics. 2023 Feb 1;474:111722. https://doi.org/10.1016/j.jcp.2022.111722

[35] Hassan SI, Ibrahim S. Shifted Chebyshev-Based Methods for Solution of Nonlinear Differential Equations. EURASIAN JOURNAL OF SCIENCE AND ENGINEERING. 2025 Aug 27;11(2):213-34. https://doi.org/10.23918/eajse.v11i2p14

[36] Ibrahim S. Numerical approximation method for solving differential equations. Eurasian Journal of Science & Engineering. 2020;6(2):157-68. https://doi.org/10.23918/eajse.v6i2p157

[37] Ibrahim S. Discrete least square method for solving differential equations. Advances and Applications in Discrete Mathematics. 2022;30. https://doi.org/10.17654/0974165822021

[38] Öztürk Y, Gülsu M. The approximate solution of high-order nonlinear ordinary differential equations by improved collocation method with terms of shifted Chebyshev polynomials. International Journal of Applied and Computational Mathematics. 2016 Dec;2(4):519-31. https://doi.org/10.1007/s40819-015-0075-1

[39] Mall S, Chakraverty S. Chebyshev neural network based model for solving Lane–Emden type equations. Applied mathematics and computation. 2014 Nov 15;247:100-14.https://doi.org/10.1016/j.amc.2014.08.085