Research Article ISSN Online: 2783-4859 Computational Mathematics and Computer Modeling with Applications

Received 30 January 2026
Accepted 29 April 2026

DOI: 10.48308/CMCMA.5.1.1
AMS Subject Classification: 65L10; 65N35; 68T07

A Novel Hybrid Architecture Combining High-Order B-Splines and Physics-Informed Neural Networks for Solving an Astrophysical Model

Sima Naraghi[a] **and Kourosh Parand**[b,c]

In this paper, we present a novel architecture for approximating solutions to differential equations in astrophysics. Our approach introduces the innovative use of nonlinear B-spline basis functions as activation functions within a neural network. Furthermore, we develop a physics-informed B-spline neural network framework with associated control points to address the Lane–Emden equations, frequently encountered in astronomy. This new method offers enhanced accuracy while requiring fewer epochs than conventional neural networks. **Copyright © 2026 Shahid Beheshti University.**

Keywords: Physics-Informed Neural Networks; High-Order B-Spline; Astrophysical Modeling; Lane–Emden equations; Numerical Methods.

1. Introduction

Recently, neural networks have found extensive applications in areas such as facial recognition, classification, and predicting natural phenomena [4, 36]. Some real-world problems require complex modeling, and we can refer to the paper [14], which develops an early warning model for social risk related to environmental damage in large-scale construction projects in China. It uses the LSTM model to predict trends and control risks, enhancing accuracy and reducing the need for data storage. Since neural networks need to model complex phenomena, it is advisable to use non-linear activation functions [12].

Today, neural networks have shown significant potential in solving differential equations across a wide range of scientific disciplines, including astronomy, physics, and biology. Traditional methods for solving complex equations often lack the necessary computational efficiency. These methods usually require substantial computational power and perform poorly when data is scarce. In contrast, neural networks can effectively solve differential equations even in general scenarios, and they have the added advantage of solving equations in the absence of certain boundary or initial conditions. This allows for data-driven approaches to solve equations [31]. The concept of solving equations using neural networks, specifically known as physics-informed neural networks (PINNs), was pioneered by Maziar Raissi [32]. Subsequently, more advanced neural networks, such as recurrent neural networks, have been utilized to address equations involving sequential data and more intricate computations [32]. In [8], the authors introduce an innovative approach that integrates deep Q-networks with context-free grammars to symbolically solve differential equations. Artificial neural networks were utilized by [2] to solve the Lane–Emden type equation. In [9], the authors present an enhanced version of the water strider algorithm tailored for solving the inverse problem of the Burgers-Huxley equation, a nonlinear partial differential equation. They also introduce a physics-informed neural network to tackle the same problem. In [6], the authors employ a hybrid approach that combines genetic programming with physics-informed neural networks to symbolically solve the Lane–Emden equation. In [21], the researchers propose an effective neural network method for solving the fractional Duffing system by designing a multi-layer network trained using the Adam algorithm and optimized via a discretized Caputo derivative with the $L1$ scheme. In [7], the researchers present a novel method that combines Monte Carlo tree search

[a] *Department of Applied Mathematics, Faculty of Mathematical Sciences, Shahid Beheshti University, Tehran, Iran.*
[b] *Department of Computer and Data Sciences, Faculty of Mathematical Sciences, Shahid Beheshti University, Tehran, Iran.*
[c] *Department of Cognitive Modeling, Institute for Cognitive and Brain Sciences, Shahid Beheshti University, Tehran, Iran.*
[*] *Correspondence to: K. Parand. Email: kparand@ibu.ca*

with physics-informed neural networks to symbolically solve differential equations by constructing and optimizing mathematical expressions. Tested on twelve examples including a case of the Lane–Emden equation.

Recently, B-splines have garnered interest in neural networks. These non-linear basis functions show considerable promise as activation functions in neural network structures. Moreover, utilizing neural networks can improve the precision of approximating B-spline curves. For example, in paper [40], the focus is on the development and use of a neural network called Adaptive B-Spline for controlling mobile satellite antennas. By utilizing B-splines, the issues of nonlinearity and uncertainty in the system are addressed, leading to more reliable communication during movement. Additionally, in another study, the authors examine the use of deep learning techniques to improve the parametrization of B-spline curves. The goal of this research is to enhance the accuracy and efficiency of curve approximation and complex shapes in three-dimensional data [20]. Furthermore, in [33], the authors explore how transformer neural networks can be utilized for B-spline curve approximation. They show that this innovative approach can improve both the accuracy and efficiency of curve handling, particularly when dealing with complex data. Additionally, Fei et al. present a novel framework known as SplineCNN, which employs continuous B-spline kernels to efficiently process geometric data. They illustrate how this approach can speed up computations while still achieving high accuracy in recognizing and classifying shapes [16]. Finally, the authors in paper [38] present a novel approach using deep neural networks for approximating B-spline curves, achieving greater accuracy and efficiency than traditional methods. This study illustrates how deep learning techniques can improve geometric design processes.

Another application of B-splines and neural networks is in the solution of differential equations. A significant advantage of utilizing these functions for this purpose is that B-splines offer a flexible means to approximate complex functions, giving users control over the finer details of the shape. This capability is particularly beneficial in scenarios where solutions may be discontinuous [34]. Another benefit of these functions is their capacity to represent complex shapes using a small number of control points. This feature contributes to reducing computational costs while maintaining an accurate representation [30]. The local properties of B-splines allow for adaptive refinement in regions where higher accuracy is needed [15]. B-splines are highly effective for modeling complex geometries, making them invaluable in fields such as fluid dynamics and structural analysis, where accurate geometric representation is crucial [17]. These functions can easily be combined with common numerical methods, like finite element method and Fourier operator, improving the performance of PINNs in solving partial differential equations [3, 24]. The paper [18] explores a novel approach to macroscopic traffic flow modeling and collision avoidance by integrating B-splines and physics-informed neural networks.

The paper [22] introduces a deep learning framework that combines Kolmogorov-Arnold networks (KANs) with physics-informed techniques to address both forward and inverse problems. By integrating physical constraints into the model, the authors enhance the accuracy and reliability of predictions.

The Lane–Emden equation is a crucial tool in astronomy, used to model the structure of polytropic stars and various other phenomena [**?**, 13]. Solving this nonlinear, singular, second-order differential equation using traditional methods is highly challenging. Various numerical techniques have been developed for solving the Lane–Emden equation [28, 1, 26, 23]. Several efficient computational algorithms have also been proposed for nonlinear singular forms of these equations, including spectral and pseudospectral approaches, as well as function-based schemes [11] . In particular, efficient computational algorithms, generalized pseudospectral methods, and fractional-order rational Bernoulli function approaches have been successfully applied to nonlinear Lane–Emden-type models arising in astrophysics and fluid mechanics [29]. These studies demonstrate the effectiveness of advanced spectral and functional techniques in handling nonlinear singular problems [27, 10]. Recently, an approximation solution of the model has been performed using the neural block method [25]. The general form of the Lane–Emden equation is [39]:

$$y''(x) + \frac{k}{x}y'(x) + f(x, y(x)) = g(x), \tag{1}$$

with the initial conditions:

$$y(0) = l_0, \quad y'(0) = l_1. \tag{2}$$

The organization of our paper is as follows: Section 2 introduces the utilization of B-splines as activation functions in neural networks. In Section 3, we demonstrate the capability of the proposed method by solving three examples. Finally, the last section provides concluding remarks.

2. B-spline Deep Neural Networks

In this section, we introduce the B-spline Deep Neural Network (BDNN) framework employed to solve the Lane–Emden equation. A knot vector is expressed as

$$E = \{\xi_1, \xi_2, \ldots, \xi_{n+p+1}\},$$

where p represents the polynomial degree and n denotes the number of basis functions used to generate the B-spline curve. The recursive definition begins with the piecewise constant basis functions:

$$N_{i,0}(\xi) = \begin{cases} 1 & \text{if } \xi_i \le \xi < \xi_{i+1}, \\ 0 & \text{otherwise.} \end{cases} \qquad i = 1, 2, \ldots, n+p+1, \tag{3}$$

For higher-order B-splines, the Cox–de Boor recursion is given by

$$N_{i,p}(\xi) = \frac{\xi - \xi_i}{\xi_{i+p} - \xi_i} N_{i,p-1}(\xi) + \frac{\xi_{i+p+1} - \xi}{\xi_{i+p+1} - \xi_{i+1}} N_{i+1,p-1}(\xi). \tag{4}$$

B-spline basis functions satisfy key properties such as partition of unity, affine invariance, and local support (convex hull property); see [5, 30, 34]. In particular, the partition of unity property implies that, for any ξ,

$$\sum_{i=1}^{n} N_{i,p}(\xi) = 1.$$

In this work, cubic B-spline polynomials (order 4) are employed as activation functions, defined by

$$B_{i,4}(u) = \begin{cases} \frac{u^3}{6} & \text{for } 0 \le u < 1, \\ \frac{-3u^3+3u^2+3u+1}{6} & \text{for } 1 \le u < 2, \\ \frac{3u^3-6u^2+4}{6} & \text{for } 2 \le u < 3, \\ \frac{(1-u)^3}{6} & \text{for } 3 \le u < 4. \end{cases} \tag{5}$$

To solve the Lane–Emden equation on the prescribed interval, we use the uniform cubic B-spline basis matrix

$$B = \frac{1}{6} \begin{bmatrix} -1 & 3 & -3 & 1 \\ 3 & -6 & 3 & 0 \\ -3 & 0 & 3 & 0 \\ 1 & 4 & 1 & 0 \end{bmatrix}, \tag{6}$$

with the control-point vector $\mathbf{b} = \begin{bmatrix} b_0 & b_1 & b_2 & b_3 & b_4 \end{bmatrix}^T$. The corresponding cubic B-spline basis functions are illustrated in Figure 1.

bspline.png bspline.png bspline.bb bspline.bb

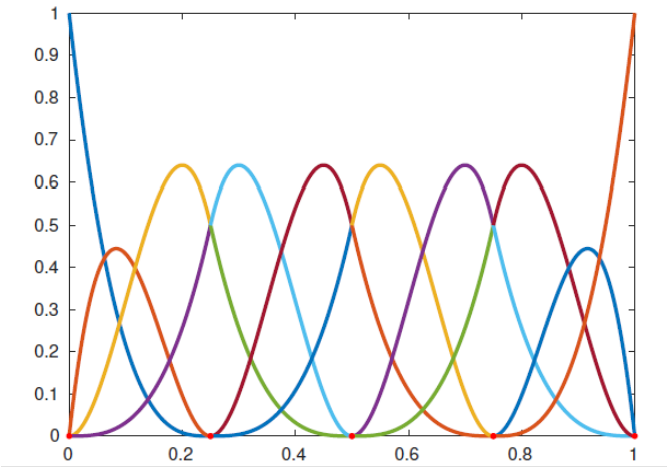


Figure 1. Cubic B-spline basis functions.

The presented approach employs a multi-layer fully-connected neural network in which the network output is regarded as the unknown function $y(x)$ in the Lane–Emden equation. Based on the Lane–Emden model (1) together with the initial conditions (2), the governing equation is rewritten as

$$x\,y''(x) + k\,y'(x) + x\,f(x, y(x)) - xg(x) = 0. \tag{7}$$

We enforce (**??**) at collocation points $\{x_i\}_{i=1}^n$ and incorporate the initial conditions through the following loss terms:

$$\begin{aligned} L_e &= \frac{1}{n} \sum_{i=1}^{n} \Big(x_i\,y''(x_i) + k\,y'(x_i) + x_i\,f(x_i, y(x_i)) - x_i g(x_i) \Big)^2, \\ L_b &= (y(0) - I_0)^2 + \big(y'(0) - I_1\big)^2. \end{aligned} \tag{8}$$

and the total loss is defined by

$$Loss_{general} = L_e + L_b. \tag{9}$$

Let the l-layer network be represented as

$$\psi = [(w^{(1)}, b^{(1)}), \ldots, (w^{(l)}, b^{(l)})],$$

where $w^{(j)}$ and $b^{(j)}$ denote the weight matrix and bias vector in the j-th layer, respectively. The forward propagation is defined by

$$z^{(1)} = w^{(1)}x^{(1)} + b^{(1)}, \qquad y^{(1)} = B_{i,4}(z^{(1)}), \tag{10}$$

and for the hidden layers $j = 2, \ldots, l-1$,

$$\begin{aligned} z^{(j)} &= w^{(j)}y^{(j-1)} + b^{(j)}, \\ y^{(j)} &= B_{i,4}(z^{(j)}). \end{aligned} \tag{11}$$

In practice, the activation functions are mapped to a prescribed interval $[-T, T]$ by appropriately scaling the input variables and the associated control points. This allows the activation function shape to adapt during training. The positions of the control points along the input axis remain fixed, while the network learns the weights and the activation parameters through optimization. As training proceeds, different layers may implicitly develop distinct activation shapes via their learned parameters. The loss function (9) is minimized using the Adam optimization algorithm, yielding an accurate approximation of the Lane–Emden solution over the desired domain.

3. Numerical Results and Discussion

In this section, we evaluate the performance and effectiveness of the proposed neural network architecture by applying it to a range of Lane–Emden equations. We present results for three examples.

For some cases, exact solutions are available, and these are provided for comparison. In instances where exact solutions are not available, we present graphs of the approximate solutions obtained. To assess the accuracy of our network, we calculate the absolute difference between the approximate and exact solutions. When an exact solution is not provided, we use the values from references [19] as a benchmark. The architecture used for both the BDNN and the conventional neural network consists of three hidden layers, with the first hidden layer having 10 nodes and the second and third hidden layers having 20 nodes each. Furthermore, the input layer has 1 node as input variable x and finally, the output layer has one node producing the value of $y(x)$. The activation function for the conventional network is $\tanh(x)$, and Adam optimization with a learning rate of 0.01 is used. To solve the Lane–Emden equation, the number of training epochs is 2500 for the conventional network, while it is reduced to 1400 for the BDNN, as shown in the numerical results. This reduction in the number of training epochs for the BDNN indicates better performance of this architecture compared to the conventional neural network, which may be due to the use of B-splines, aiding in more accurate modeling.

3.1. Example 1

The standard form of the Lane–Emden equation is given by [35]:

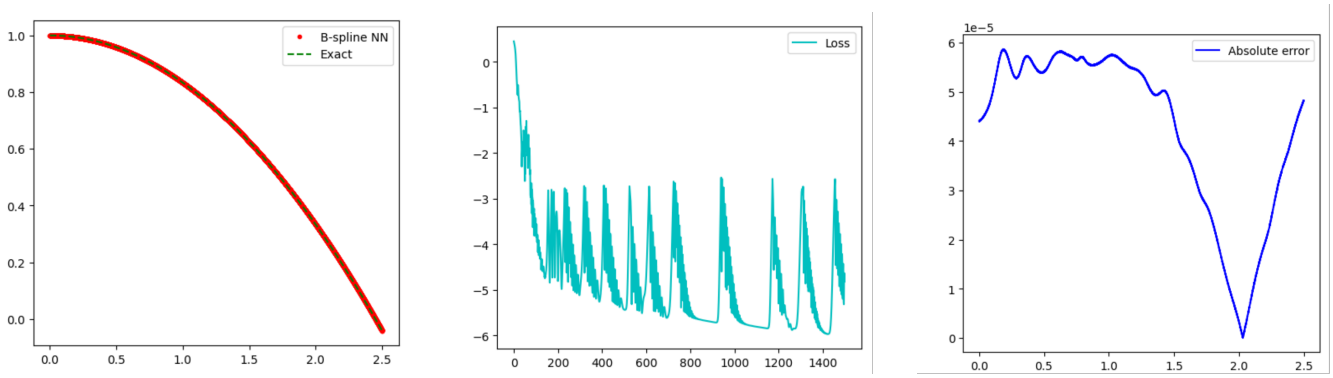
$$\begin{aligned} &y''(x) + \frac{2}{x}y'(x) + y^m(x) = 0, \\ &y(0) = 1, \quad y'(0) = 0. \end{aligned} \tag{12}$$

For specific values of m, the exact solutions are:

$$y(x) = \begin{cases} 1 - \frac{x^2}{6} & \text{for } m = 0, \\ \frac{\sin(x)}{x} & \text{for } m = 1, \\ \left(1 + \frac{x^2}{3}\right)^{-\frac{1}{2}} & \text{for } m = 5. \end{cases} \tag{13}$$

In cases where an exact analytical solution exists, we compare the results obtained by the presented method with the exact solution. However, for $m = 2, 3, 4$, no exact analytical solution is available. In these cases, some researchers have provided values at specific interior points within a given interval. Therefore, we compare the results obtained from our proposed method with those values.

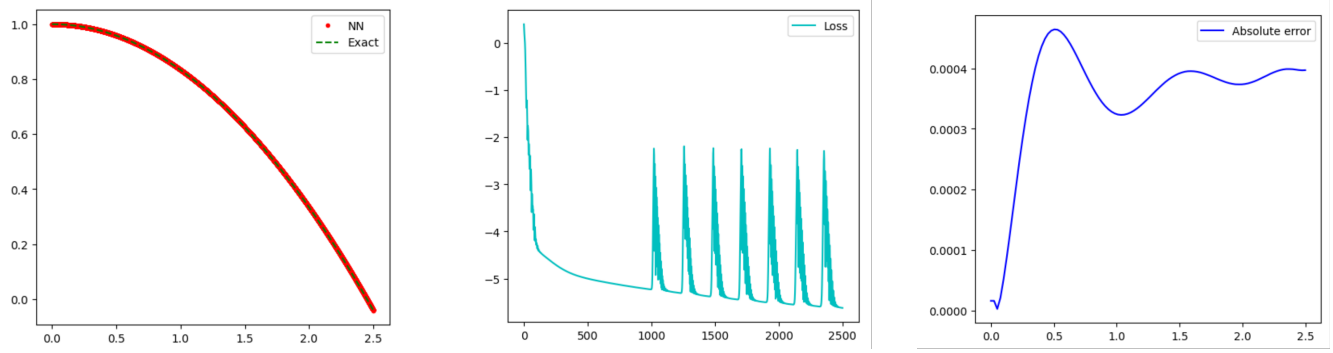
3.1.1. Result for case $m = 0$ of Eq. (12) In this subsection we provide the result obtained from the proposed BDNN and NN for case $m = 0$ of Eq. (12). Since an exact analytical solution exists for this problem, $y(x) = 1 - x^2/6$, we compare the results derived from our proposed method with this solution. Figures 2 and 3 present the outcomes of the proposed method alongside those from the standard network. The computed MSE in this example is $2.31e{-}09$.



(a) Exact vs. BDNN (b) BDNN loss (c) Absolute error

Figure 2. Results obtained by BDNN with the exact analytical solution for Example 1 with $m = 0$.

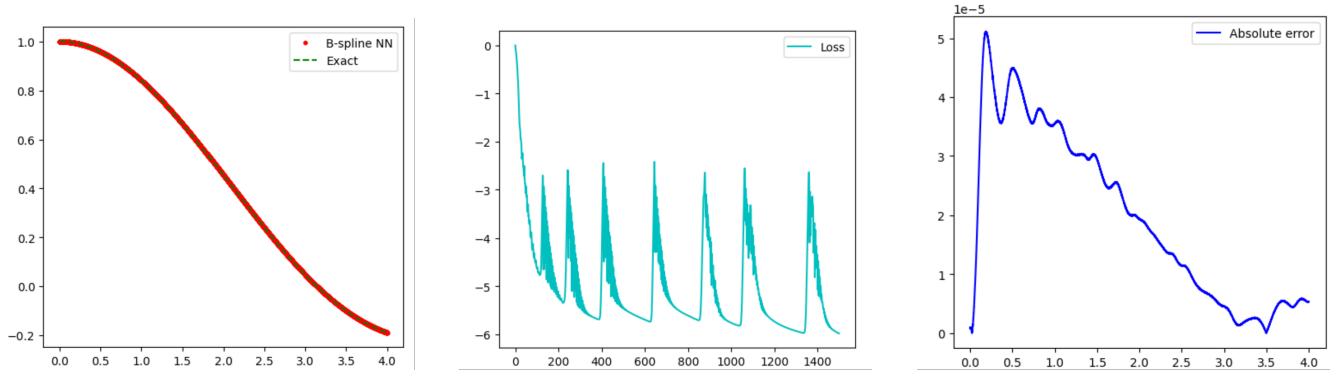
(1).png (1).png (1).bb (1).bb



(a) Exact vs. PINN (b) PINN loss (c) Absolute error

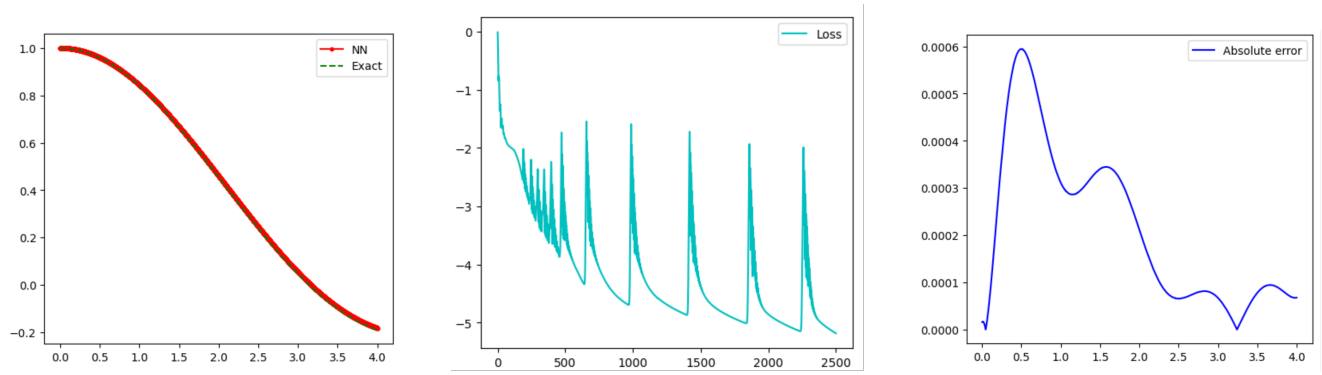
Figure 3. Results obtained by conventional PINN with the exact analytical solution for Example 1 with $m = 0$.

3.1.2. Result for case $m = 1$ of Eq. (12) With an exact analytical solution available for this problem, $y(x) = \sin(x)/x$, we compare the outcomes of BDNN with the conventional PINN. The computed MSE in this case is $6.04e{-}10$.



(a) Exact vs. BDNN (b) BDNN loss (c) Absolute error

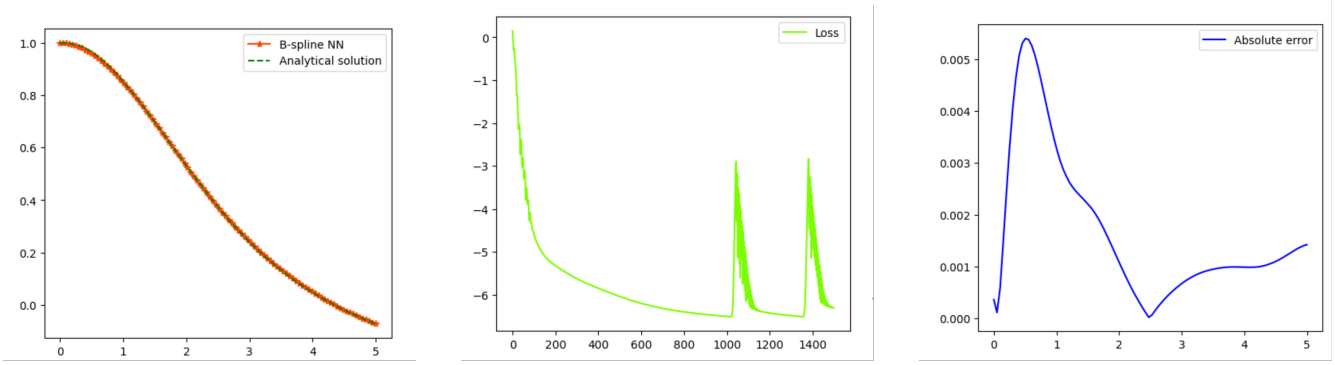
Figure 4. Results obtained by BDNN with the exact analytical solution for Example 1 with $m = 1$.



(a) Exact vs. PINN (b) PINN loss (c) Absolute error

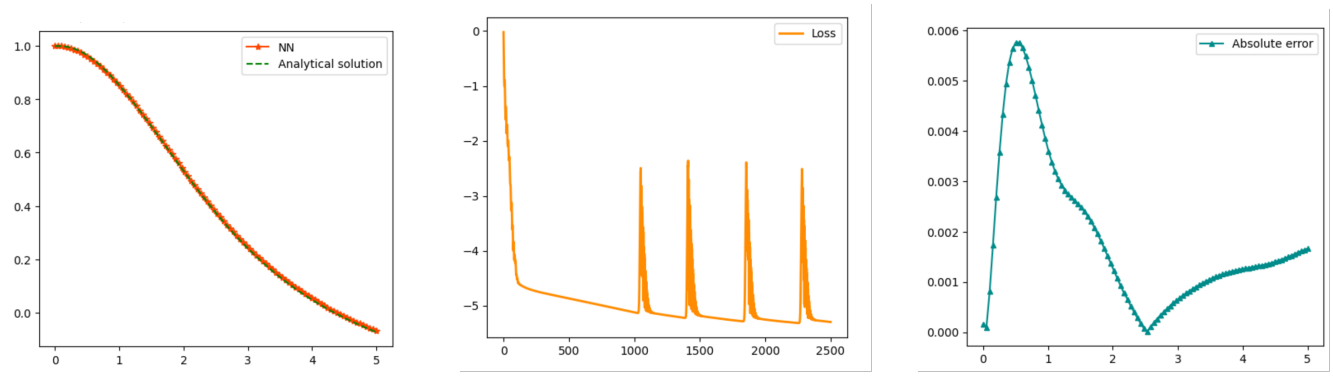
Figure 5. Results obtained by conventional PINN with the exact analytical solution for Example 1 with $m = 1$.

3.1.3. Result for case $m = 2$ of Eq. (12) For this example, an analytical solution was recently found in [6] on the interval $[0, 5]$. Figures 6 and 7 show BDNN and conventional PINN results. Table 1 reports pointwise comparisons. The computed MSE is $4.79e{-}6$.



(a) BDNN vs. ref. (b) BDNN loss (c) Absolute error

Figure 6. Comparison of BDNN results with [6] for Example 1 with $m = 2$.



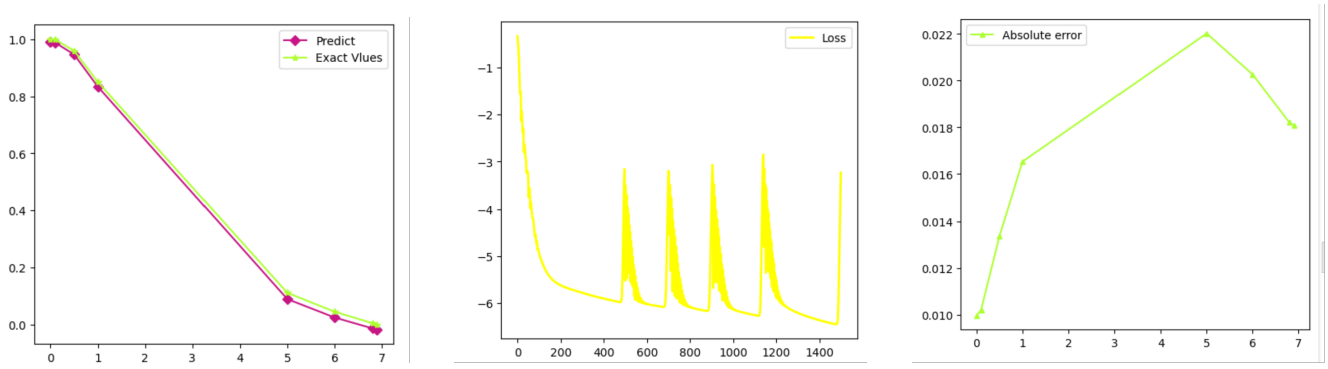
(a) PINN vs. ref. (b) PINN loss (c) Absolute error

Figure 7. Comparison of conventional PINN results with [6] for Example 1 with $m = 2$.

Table 1. Comparison of BDNN predictions with exact values and associated error in Example 1 with $m = 2$.

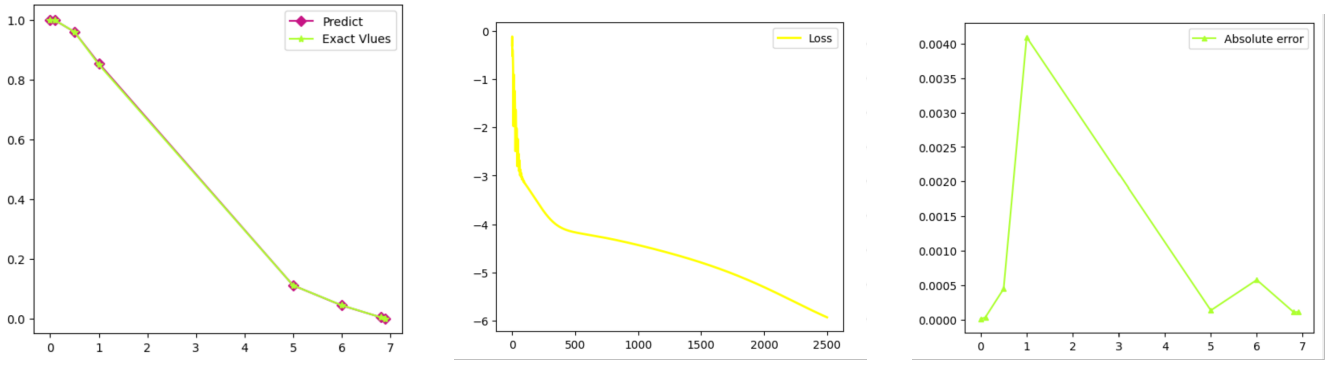
x	BDNN	Exact value	Absolute Error
0.45	0.966890	0.972591	0.005701
0.90	0.875184	0.879376	0.004192
1.36	0.740726	0.743436	0.002710
1.81	0.592484	0.594289	0.001805
2.26	0.446885	0.447484	0.000599
2.71	0.315885	0.315620	0.000264
3.16	0.204630	0.203815	0.000815
3.61	0.113439	0.112381	0.001058

3.1.4. Result for case $m = 3$ of Eq. (12) Since an exact analytical solution is not available for this problem, we assess the accuracy of the proposed method using values reported in [19, 26]. Figures 8 and 9 illustrate the results. The MSE is $2.20e{-}02$.



(a) BDNN vs. ref. (b) BDNN loss (c) Absolute error

Figure 8. BDNN results for Example 1 with $m = 3$.



(a) PINN vs. ref. (b) PINN loss (c) Absolute error

Figure 9. Conventional PINN results for Example 1 with $m = 3$.

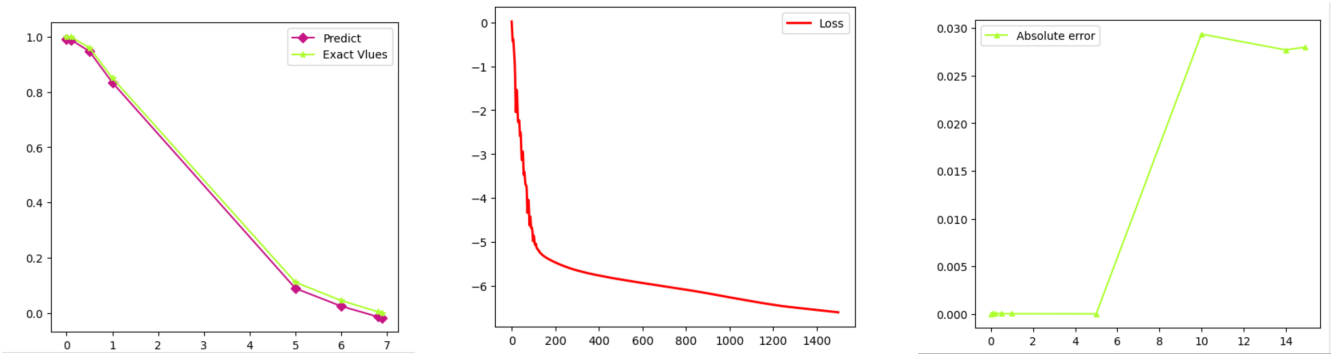
Table 2. Comparison of BDNN predictions with exact values and associated error in Example 1 with $m = 3$.

x	BDNN	Exact value	Absolute error
0.000	0.999999	1.000000	0.000001
0.100	0.998330	0.998336	0.000006
0.500	0.959904	0.959839	0.000065
1.000	0.855093	0.850576	0.004517
5.000	0.110824	0.110820	0.000005
6.000	0.043741	0.044274	0.000533
6.800	0.004107	0.004168	0.000061
6.896	-0.000027	0.000036	0.000063

3.1.5. Result for case $m = 4$ of Eq. (12) Due to the absence of an exact analytical solution for this case, we use reported values in [19, 26]. The MSE is $9.74e{-}05$.

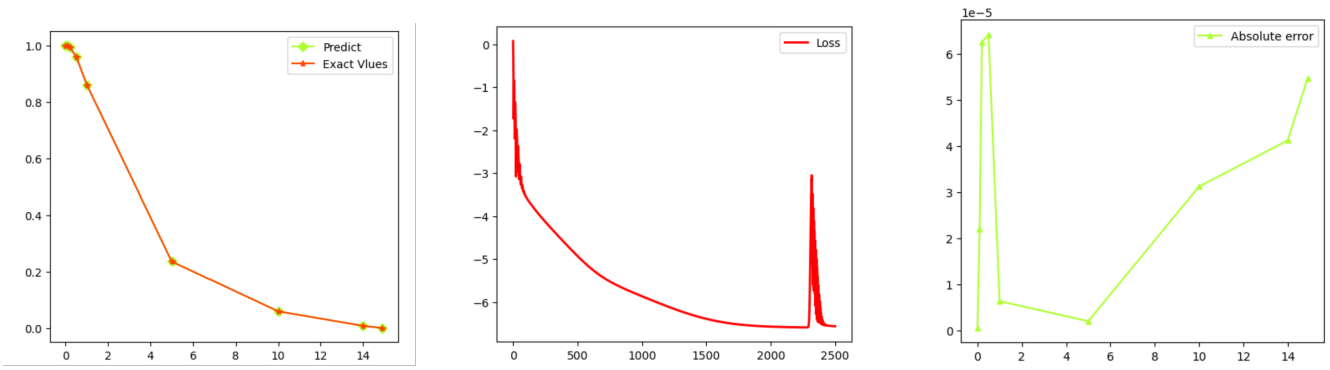
Table 3. Comparison of BDNN predictions with exact values and the associated error in Example 1 with $m = 4$.

x	BDNN	Exact value	Absolute error
0.0	0.999999	1.000000	7.748604e-07
0.1	0.998397	0.998337	6.061792e-05
0.2	0.993492	0.993386	1.059175e-04
0.5	0.960386	0.960311	7.480383e-05
1.0	0.860860	0.860814	4.613400e-05
5.0	0.235899	0.235923	2.416968e-05
10.0	0.060288	0.059673	6.154850e-04
14.0	0.007279	0.008330	1.051223e-03
14.9	0.002148	0.000576	1.571193e-03



(a) Exact vs. BDNN (b) BDNN loss (c) Absolute error

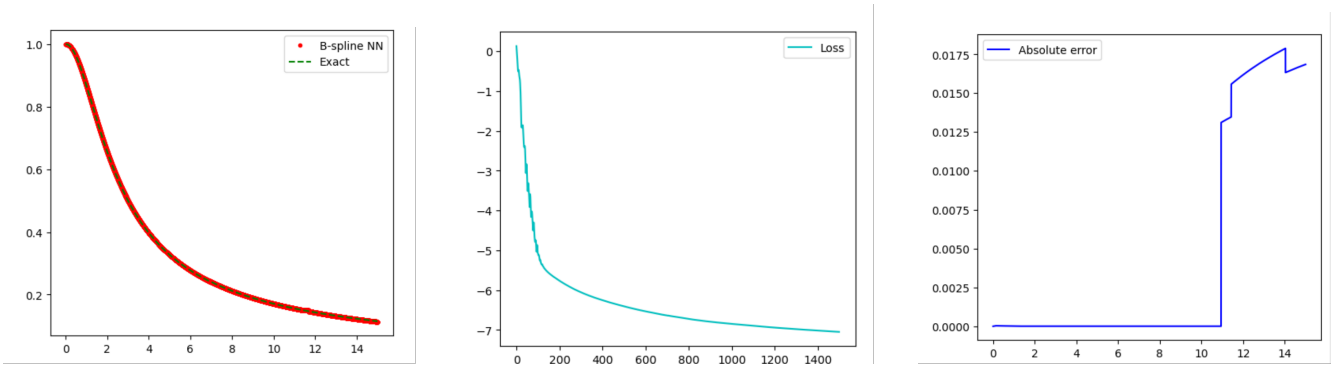
Figure 10. BDNN results for Example 1 with $m = 4$.



(a) Exact vs. PINN (b) PINN loss (c) Absolute error

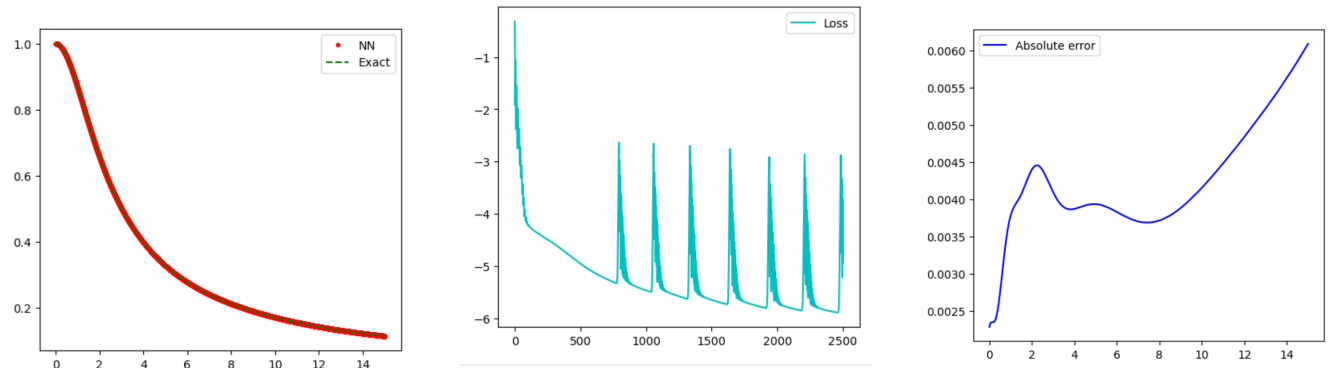
Figure 11. Conventional PINN results for Example 1 with $m = 4$.

3.1.6. Result for case $m = 5$ of Eq. (12) Since an exact analytical solution exists for this case, we compare BDNN and conventional PINN results. The MSE is $7.26e{-}05$.



(a) Exact vs. BDNN (b) BDNN loss (c) Absolute error

Figure 12. BDNN results for Example 1 with $m = 5$.



(a) Exact vs. PINN (b) PINN loss (c) Absolute error

Figure 13. Conventional PINN results for Example 1 with $m = 5$.

3.2. Example 2

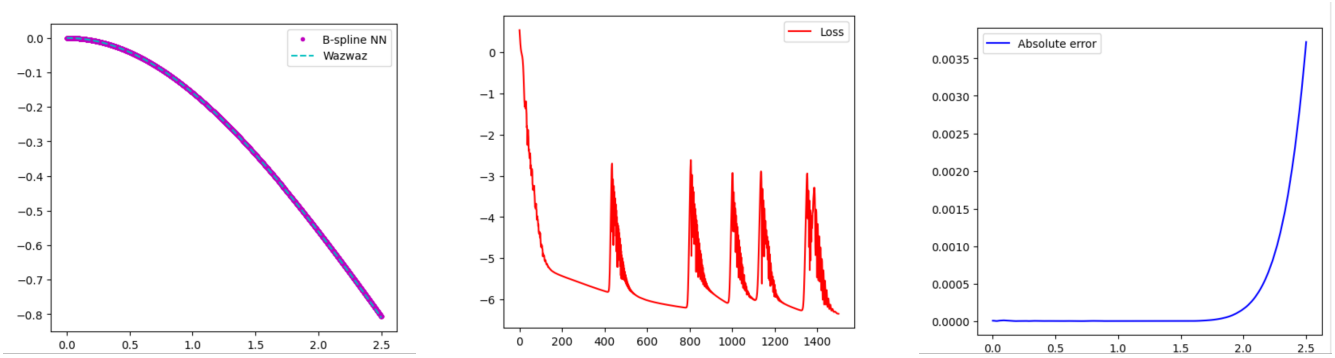
In this example, we address a different form of the Lane–Emden equation using the proposed BDNN with a B-spline of order 4 as the activation function for the model:

$$\begin{aligned} &y''(x) + \frac{2}{x}y'(x) + e^{y(x)} = 0, \\ &y(0) = 0, \quad y'(0) = 0. \end{aligned} \tag{14}$$

The approximation solution for this model is given by:

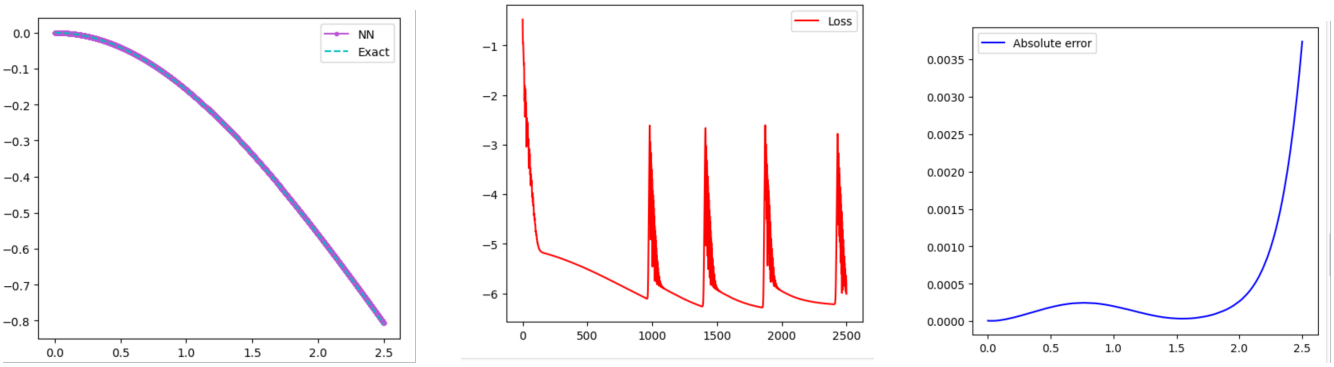
$$y(x) = -\frac{x^2}{6} + \frac{x^4}{5 \times 4!} - \frac{8x^6}{21 \times 6!} + \frac{122x^8}{81 \times 8!} - \frac{4087x^{10}}{495 \times 10!}. \tag{15}$$

Figures illustrate the outcomes of solving this Example using BDNN and conventional PINN approaches. For accuracy comparison, we reference the values provided in [37]. The resulting MSE was found to be $5.46e{-}05$.



(a) Exact vs. BDNN (b) BDNN loss (c) Absolute error

Figure 14. Results obtained by BDNN for Example 2.



(a) Exact vs. PINN (b) PINN loss (c) Absolute error

Figure 15. Results obtained by conventional PINN for Example 2.

Table 4. Comparison of BDNN predictions with exact values and the associated error in Example 2.

x	BDNN	Exact value	Absolute error
0.0	0.000005	0.000000	5.334616e-06
0.1	-0.001675	-0.001666	9.414041e-06
0.2	-0.006653	-0.006653	8.190982e-07
0.5	-0.041155	-0.041154	1.333654e-06
1.0	-0.158828	-0.158827	1.117587e-06
1.5	-0.338013	-0.338013	8.940697e-08
2.0	-0.559801	-0.559963	1.618862e-04
2.5	-0.806300	-0.810020	3.719270e-03

3.3. Example 3

In this example, we solve the following form of the Lane–Emden equation using the proposed BDNN with a B-spline of order 4 as the activation function:

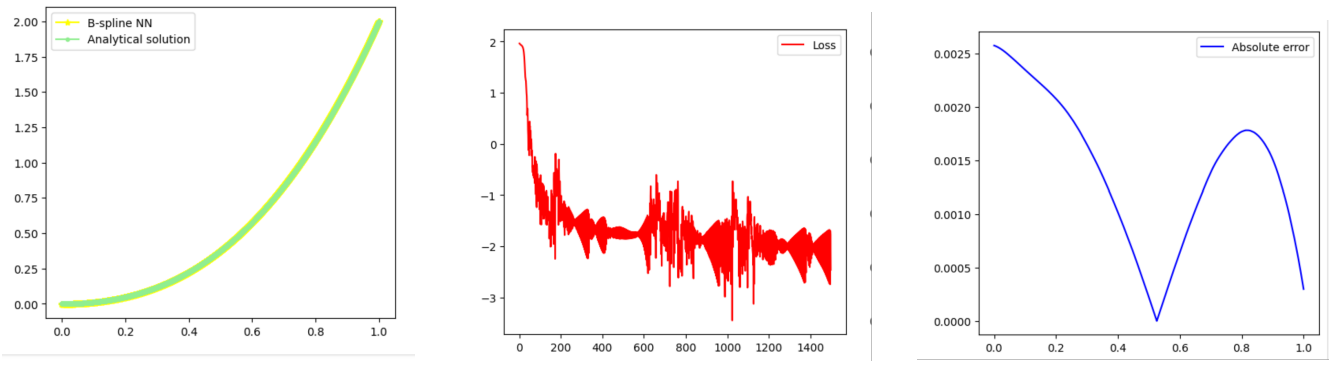
$$\begin{aligned} &x\,y''(x) + 2\,y'(x) + x\,y(x) = x(6 + 12x + x^2 + x^3), \\ &y(0) = 0, \qquad y'(0) = 0. \end{aligned} \tag{16}$$

This equation has been solved by Wazwaz in [37] and the analytical solution for this model is given by:

$$y(x) = x^2 + x^3. \tag{17}$$

Table 5. Comparison of BDNN predictions with exact values and the associated error in Example 3.

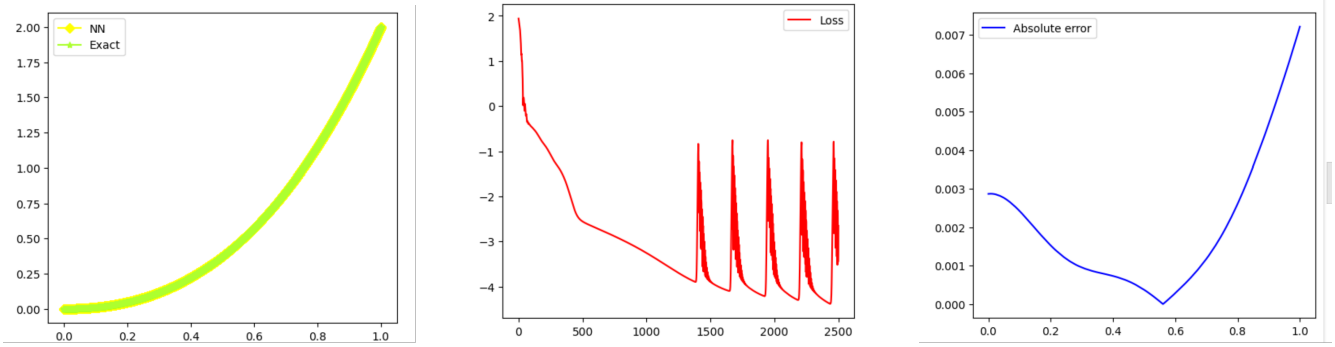
x	BDNN	Exact value	Absolute error
0.0	0.003089	0.000	0.003089
0.2	0.052760	0.048	0.004760
0.4	0.229675	0.224	0.005675
0.6	0.580881	0.576	0.004881
0.8	1.152782	1.152	0.000782
0.9	1.536234	1.539	0.000782
1.0	1.992652	2.000	0.007348



(a) Exact vs. BDNN (b) BDNN loss (c) Absolute error

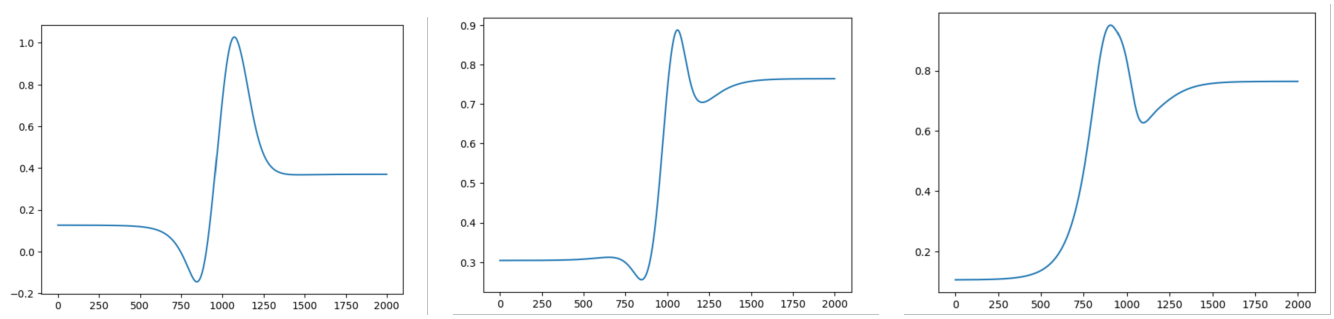
Figure 16. Results obtained by BDNN for Example 3.

Figures 16 and 17 illustrate the results obtained by solving the Lane–Emden equation with the B-spline and standard PINN approaches. We compare error values of our method with those obtained in [37] in Table 5. The MSE for this example is $2.00e{-}05$.



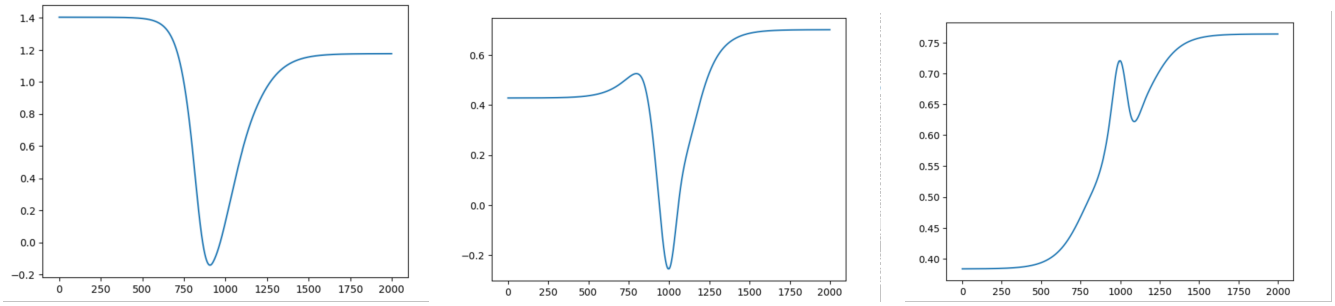
(a) Exact vs. PINN (b) PINN loss (c) Absolute error

Figure 17. Results obtained by conventional PINN for Example 3.



(a) Example 1, $m = 0$. (b) Example 1, $m = 1$. (c) Example 1, $m = 2$.

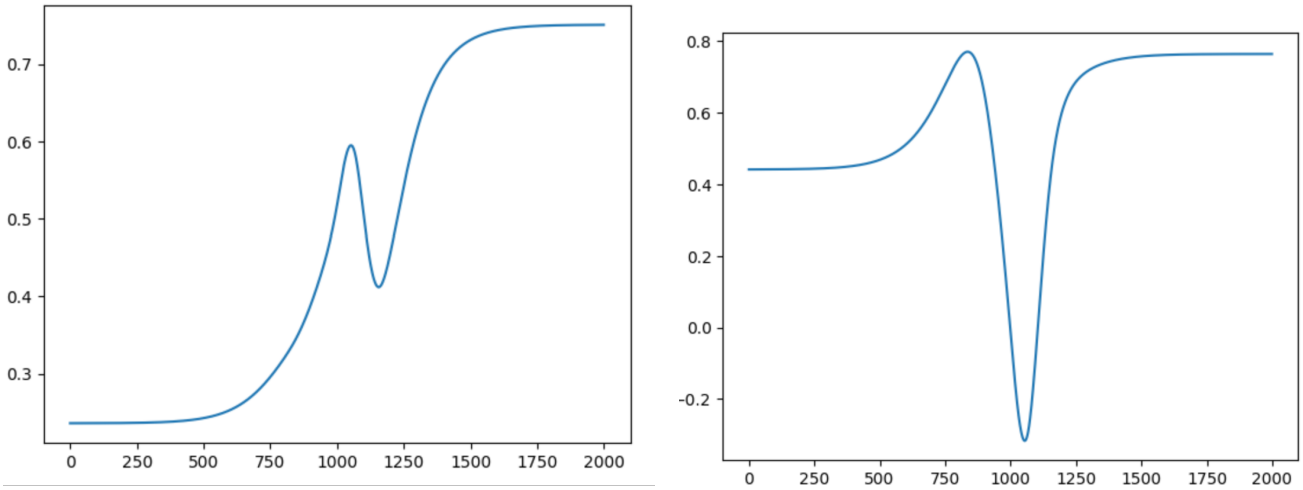
.png .png .bb .bb



(d) Example 1, $m = 3$. (e) Example 1, $m = 4$. (f) Example 1, $m = 5$.

Figure 18. Sample of activation functions gained for Example 1.

Figures 18 and 19 show several examples of the activation functions derived from B-spline curves.



(a) Example 2. (b) Example 3.

Figure 19. Sample of activation functions gained for Examples 2 and 3.

4. Conclusions

In this paper, we introduce the B-spline deep neural network as an innovative method for addressing the Lane–Emden equation, a commonly used model in astrophysics. The main goal of this method is to improve accuracy while decreasing the number of training epochs. By using B-splines as activation functions, the BDNN enhances both efficiency and precision in approximating solutions to the Lane–Emden equation. The incorporation of B-spline basis functions as activation functions highlights their flexibility in neural networks. This flexibility allows the network to better model complex behaviors in systems. Due to their local control and natural smoothness, B-splines are particularly well-suited for neural networks, especially in solving challenging problems such as nonlinear ordinary differential equations (NODEs). B-splines offer natural smoothness and continuous derivatives, crucial for solving NODEs with high-quality, stable solutions. Their mathematical properties provide robustness in approximation, ensuring consistent performance even for high-dimensional or NODEs.

We evaluated the performance of the BDNN by comparing it with a conventional neural network that uses standard activation functions. The results show that the BDNN consistently delivers superior outcomes. Specifically, the BDNN's use of higher-order B-spline polynomials significantly improves its accuracy. The numerical results from both the training and testing stages confirm the effectiveness and high performance of the BDNN approach.

The proposed architecture can be extended to handle equations involving partial derivatives and integral equations. Using higher-order B-splines as activation functions presents a promising research direction. B-splines can be integrated into the finite element method to approximate solutions for the Lane–Emden equation, enhancing both accuracy and stability. Their smoothness allows for a detailed analysis of stellar density and pressure profiles. B-splines also improve the visualization of solutions, providing clearer insights into stellar structures.

References

1. W. Adel and Z. Sabir. Solving a new design of nonlinear second-order Lane–Emden pantograph delay differential model via Bernoulli collocation method. *The European Physical Journal Plus*, 135(5):427, 2020.
2. H. Baty. Modelling Lane–Emden type equations using physics-informed neural networks. *Astronomy and Computing*, 44:100734, 2023.
3. S. Cen, B. Jin, Q. Quan, and Z. Zhou. Hybrid neural-network FEM approximation of diffusion coefficient in elliptic and parabolic problems. *IMA Journal of Numerical Analysis*, 44(5):3059–3093, 2024.
4. K. R. Chowdhary. Natural language processing. In *Fundamentals of Artificial Intelligence*, pages 603–649. Springer, 2020.
5. J. A. Cottrell, T. J. R. Hughes, and Y. Bazilevs. *Isogeometric Analysis: Toward Integration of CAD and FEA*. John Wiley & Sons, 2009.
6. H. Dana Mazraeh and K. Parand. GEPINN: An innovative hybrid method for a symbolic solution to the Lane–Emden type equation based on grammatical evolution and physics-informed neural networks. *Astronomy and Computing*, 48:100846, 2024.
7. H. Dana Mazraeh and K. Parand. Approximate symbolic solutions to differential equations using a novel combination of Monte Carlo tree search and physics-informed neural networks approach. *Engineering with Computers*, 2025.
8. H. Dana Mazraeh and K. Parand. An innovative combination of deep Q-networks and context-free grammars for symbolic solutions to differential equations. *Engineering Applications of Artificial Intelligence*, 142:109733, 2025.

9. H. Dana Mazraeh, K. Parand, M. Hosseinzadeh, J. Lansky, and V. Nulíček. An improved water strider algorithm for solving the inverse Burgers-Huxley equation. *Scientific Reports*, 14(1):28797, 2024.
10. M. Delkhosh, K. Parand, and H. Yousefi. Accurate numerical solution for a type of astrophysics equations using three classes of Euler functions. *Bulletin Mathématique de la Société des Sciences Mathématiques de Roumanie*, 61(1):39–49, 2018.
11. M. Delkhosh, A. Rahmanzadeh, and S.-F. Shafiei. Solving nonlinear differential equations in astrophysics and fluid mechanics using the generalized pseudospectral method. *SeMA Journal*, 78(4):457–474, 2021.
12. K. Doległo, A. Paszyńska, M. Paszyński, and L. Demkowicz. Deep neural networks for smooth approximation of physics with higher order and continuity B-spline base functions. arXiv preprint arXiv:2201.00904, 2022.
13. R. Emden. *Gaskugeln: Anwendungen der mechanischen Wärmetheorie auf kosmologische und meteorologische Probleme*. B. G. Teubner, 1907.
14. J. Fang, D. Huang, and J. Xu. Social risk early warning of environmental damage of large-scale construction projects in China based on network governance and LSTM model. *Complexity*, 2020(1):8863997, 2020.
15. L. Fang and L. Stals. Data-based adaptive refinement of finite element thin plate spline. *Journal of Computational and Applied Mathematics*, 451:115975, 2024.
16. M. Fey, J. E. Lenssen, F. Weichert, and H. Müller. SplineCNN: Fast geometric deep learning with continuous B-spline kernels. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 869–877. IEEE, 2018.
17. S. Ganga and Z. Uddin. Exploring physics-informed neural networks: From fundamentals to applications in complex systems. arXiv preprint arXiv:2410.00422, 2024.
18. M. Haddioui, Y. Qaraai, S. Bouarafa, S. Agoujil, and A. Bouhamidi. A macroscopic traffic flow modelling and collision avoidance using B-Spline and physics-informed neural network approaches. *International Journal of Intelligent Networks*, 5:196–203, 2024.
19. G. P. Horedt. *Polytropes: Applications in Astrophysics and Related Fields*. Kluwer Academic Publishers, 2004.
20. P. Laube, M. O. Franz, and G. Umlauf. Deep learning parametrization for B-spline curve approximation. In *Proceedings of the International Conference on 3D Vision (3DV)*, pages 691–699. IEEE, 2018.
21. G.-Q. Liu and G.-C. Wu. Parallel computing and a multi-layer neural network algorithm for solving the fractional Duffing system. *Romanian Journal of Physics*, 69(5–6):107, 2024.
22. Z. Liu, Y. Wang, S. Vaidya, F. Ruehle, J. Halverson, M. Soljačić, T. Y. Hou, and M. Tegmark. KAN: Kolmogorov-Arnold Networks. arXiv preprint arXiv:2404.19756, 2024.
23. J. Malele, P. Dlamini, and S. SimeLane. Solving Lane–Emden equations with boundary conditions of various types using high-order compact finite differences. *Applied Mathematics in Science and Engineering*, 31(1):2214303, 2023.
24. N. Margenberg, F. X. Kärtner, and M. Bause. Optimal dirichlet boundary control by fourier neural operators applied to nonlinear optics. *Journal of Computational Physics*, 482:112725, 2023.
25. K. Parand, A. A. Aghaei, S. Kiani, T. Ilkhas Zadeh, and Z. Khosravi. A neural network approach for solving nonlinear differential equations of Lane–Emden type. *Engineering with Computers*, 40(2):953–969, 2024.
26. K. Parand, M. Dehghan, A. R. Rezaei, and S. M. Ghaderi. An approximation algorithm for the solution of the nonlinear Lane–Emden type equations arising in astrophysics using Hermite functions collocation method. *Computer Physics Communications*, 181(6):1096–1108, 2010.
27. K. Parand, A. Ghaderi-Kangavari, and M. Delkosh. Two efficient computational algorithms to solve the nonlinear singular Lane–Emden equations. *Astrophysics*, 63(1):133–150, 2020.
28. K. Parand and S. Khaleqi. The rational Chebyshev of second kind collocation method for solving a class of astrophysics problems. *The European Physical Journal Plus*, 131(2):24, 2016.
29. K. Parand, H. Yousefi, and M. Delkhosh. A numerical approach to solve Lane–Emden-type equations by the fractional order of rational Bernoulli functions. *Romanian Journal of Physics*, 62:104, 2017.
30. L. Piegl and W. Tiller. *The NURBS Book*. Springer, 1995.
31. M. Raissi. Deep hidden physics models: Deep learning of nonlinear partial differential equations. *Journal of Machine Learning Research*, 19(25):1–24, 2018.
32. M. Raissi, P. Perdikaris, and G. E. Karniadakis. Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations. *Journal of Computational Physics*, 378:686–707, 2019.
33. M. Saillot, D. Michel, and A. Zidna. B-spline curve approximation with transformer neural networks. *Mathematics and Computers in Simulation*, 223:275–287, 2024.
34. L. Schumaker. *Spline Functions: Basic Theory*. Cambridge University Press, Cambridge, 2007.
35. Z. Shafinejad, M. Zarebnia, and M. Lakestani. Numerical solution of Lane-Emden type equations using flatlet oblique multiwavelets collocation approach. *International Journal of Nonlinear Analysis and Applications*, 14(10):199–215, 2023.
36. S. Uchida. Image processing and recognition for biological images. *Development, Growth & Differentiation*, 55(4):523–549, 2013.
37. A.-M. Wazwaz. A new algorithm for solving differential equations of Lane–Emden type. *Applied Mathematics and Computation*, 118(2–3):287–310, 2001.
38. Z. Wen, J. Luo, and H. Kang. The deep neural network solver for B-Spline approximation. *Computer-Aided Design*, 169:103668, 2024.
39. Y. H. Youssri, W. M. Abd-elhameed, and E. H. Doha. Ultraspherical wavelets method for solving Lane-Emden type equations. *Romanian Journal of Physics*, 60(9–10):1298–1314, 2015.
40. X. Zhang, Y. Zhao, K. Guo, G. Li, and N. Deng. An adaptive B-spline neural network and its application in terminal sliding mode control for a mobile satcom antenna inertially stabilized platform. *Sensors*, 17(5):978, 2017.

