AIMS Mathematics, 11(9): 30886–30903.
DOI: 10.3934/math.20261223
Received: 22 July 2026
Revised: 11 September 2026
Accepted: 16 September 2026
Published: 21 September 2026

https://www.aimspress.com/journal/Math

Research article

A physics-informed neural network approach for the one-dimensional wave equation: data-driven solution with Fibonacci polynomial activation functions

Ali Algan[1] and Faruk Düşünceli[2,*]

[1] Department of Mathematics, Institute of Graduate Studies, Mardin Artuklu University, Türkiye
[2] Department of Economics, Mardin Artuklu University, Türkiye

* **Correspondence:** Email: farukdusunceli@artuklu.edu.tr.

Abstract: In this study, a data-driven computational framework based on Physics-Informed Neural Networks (PINNs) was proposed for the numerical solution of the one-dimensional wave equation. Conventional numerical methodologies, including finite difference and finite element schemes, necessitated intricate discretization processes and became computationally onerous for high-resolution simulations. The proposed PINN framework reinterprets the problem as a supervised learning task that learns the underlying physical behavior directly from data in a mesh-free manner, using training sets generated from the governing equation together with the initial and boundary conditions. The central contribution of this study is the integration of Fibonacci polynomials as activation functions within the network architecture. The continuous and structurally rich mathematical properties of Fibonacci polynomials enhance the representational capacity of the network relative to conventional activation functions such as Tanh, improving learning dynamics, training stability, and accuracy in capturing the oscillatory nature of wave phenomena. The performance of the model was evaluated through error analysis, utilizing standard metrics such as Mean Squared Error (MSE) and Relative-L2 error, in comparison to known reference solutions. The proposed Fibonacci-PINN was directly compared with a standard Tanh-based PINN under identical training conditions. The findings demonstrated that the Fibonacci-PINN consistently attains reduced prediction error in comparison to the standard Tanh-PINN across all three numerical illustrations examined, thereby evidencing enhanced approximation precision with diminished computational demands. In contradistinction to conventional discretization-based methodologies, the model engenders smooth, continuous functional outputs, thereby facilitating efficacious visualization of

wave propagation. In this study, we demonstrate the efficiency and flexibility of combining artificial neural networks with structurally rich mathematical activation functions as a viable alternative.

Keywords: data-driven modeling; partial differential equations; activation function design; Levenberg-Marquardt optimization
Mathematics Subject Classification: 65M70, 68T07, 35L05, 11B39

1. Introduction

In recent times, Physics-Informed Neural Networks (PINNs) and related neural network-based approaches have become one of the most prevalent methods for achieving impressive feats in a variety of disciplines, ranging from biomedical modeling [1] to nonlinear wave dynamics [2,3] and chemical transport phenomena [4]. One of the application areas of this method is mathematics and physics. The wave equation is a partial differential equation of the second order, which is employed in classical physics to describe the propagation of waves, including mechanical waves [5] and electromagnetic waves [6,7]. In particular, the one-dimensional wave equation has been extensively investigated in the context of vibrating systems and related physical models [8].

Ricky and Mishra presented a comprehensive theoretical framework for the analysis of approximation, generalization, and training errors of PINN-based models. Moreover, the influence of solution regularity and problem dimensionality on performance. The findings of the paper, which are supported by numerical experiments, emphasize training error as a predominant challenge hindering the overall efficiency of physics-informed machine learning methods [9].

It is evident that partial differential equations play a pivotal role in the modeling of physical systems. Wave equations, and more specifically, the modeling of ultrasonic waves, are utilized in medical imaging and treatment technologies. PINNs have become a widely used mesh-free approach for solving nonlinear partial differential equations (PDEs) governing wave phenomena. However, standard formulations often struggle with predictive accuracy and long-horizon physical consistency. Researchers have sought to address these limitations through targeted architectural refinements. A liquid residual gating mechanism was introduced to improve reconstruction accuracy while preserving the standard PINN training protocol [10]. Additionally, a Noether-theorem-derived conservation penalty was incorporated into the loss function to enforce strict conservation of particle number, momentum, and energy in nonlinear [11]. Concurrently, several researchers have employed neural network architectures as symbolic ansatz generators for deriving exact soliton solutions of nonlinear PDEs, including the Kakutani–Matsuuchi model [12], a generalized (2+1)-dimensional soliton equation [13], the (3+1)-dimensional conformable time-fractional Zakharov–Kuznetsov equation [14], and the generalized doubly dispersive equation via a neural-network-generalized Kudryashov method [15]. Collectively, these studies have yielded diverse soliton families such as lump, breather, rogue, and bright–dark solutions. A plethora of foundational studies have examined PINN performance on the canonical wave equation, evaluating forward/inverse problem settings, including boundary control and parameter identification [16], the effects of boundary-condition formulation [17], and network depth and width [18]. In addition to these physics-informed approaches, earlier data-driven methods such as genetic programming and model trees have also been applied to real-world ocean wave forecasting [19], emphasizing the broader shift from purely statistical to physics-constrained learning in wave modeling.

In this study, we propose a data-driven computational framework based on PINNs to solve the one-dimensional wave equation numerically. This framework builds on the outlined body of work. Conventional numerical methodologies, including finite difference and finite element schemes, necessitate intricate discretization processes and become computationally onerous at elevated spatial and temporal resolutions. In contrast, the proposed PINN approach reformulates the problem as a supervised learning task, learning the underlying physical behavior directly from data in a mesh-free manner. The network is trained on datasets generated from the governing equation, together with its initial and boundary conditions. This process enables the network to learn the functional relationship between the input variables and the corresponding wave response. The pivotal innovation of this study lies in the integration of Fibonacci polynomials as activation functions within the network architecture. These polynomials, renowned for their continuous and structurally rich mathematical properties, serve to enhance the model's representational capacity in comparison to standard activation functions. This enhancement leads to improvements in learning dynamics, stability, and accuracy, particularly in the capture of the oscillatory nature of wave phenomena. The performance of the model is evaluated through error analysis using standard metrics such as Mean Squared Error (MSE) against known reference solutions. This demonstrates high approximation accuracy, significantly reduced computational effort, and strong generalization across configurations. Moreover, in contradistinction to conventional discrete methodologies, the proposed model engenders smooth, continuous functional outputs, thereby facilitating efficacious visualization of wave propagation. The study under consideration demonstrates the efficiency, scalability, and flexibility of combining artificial neural networks with advanced mathematical activation functions as a powerful alternative for solving partial differential equations.

In this study, a one-dimensional wave equation with a single unknown function is considered as follows:

$$\frac{\partial^2 \mathrm{u}}{\partial \mathrm{t}^2} = \mathrm{c}^2 \frac{\partial^2 \mathrm{u}}{\partial \mathrm{x}^2} + f(x,t). \tag{1}$$

For the above equation, the initial and boundary conditions are $\mathrm{c} \neq 0$, $\mathrm{t} > 0$, $\mathrm{x} > 0$.

$$\begin{gathered} \mathrm{u}(0,\mathrm{t}) = \mathrm{g}_1(\mathrm{t}) \\ \mathrm{u}(1,\mathrm{t}) = \mathrm{g}_2(\mathrm{t}) \\ \mathrm{u}(\mathrm{x},0) = \mathrm{h}_1(\mathrm{x})\,' \\ \mathrm{u}_\mathrm{t}(\mathrm{x},0) = \mathrm{h}_2(\mathrm{x}) \end{gathered} \tag{2}$$

The structure of the article is as follows: In the second section of this study, we provide a comprehensive overview of the materials and methods employed in the research. These encompass the utilization of a physics-informed neural networks neural network technique, the Fibonacci polynomial and its associated properties, the integration of the Fibonacci-PINN into the wave equation, and the algorithmic implementation and training protocol. As outlined in Section 3, the proposed method is applied to three distinct problems, and the results are presented in graphical form. In the conclusion section, we summarize the findings from these three examples and demonstrate that the proposed method outperforms classical activation functions.

2. Material and methods

In the following section, we delineate the core methodological framework of the study, commencing with the foundational concepts of the artificial neural network architecture and the

mathematical properties of Fibonacci polynomials that are utilized as activation functions. In the following section, the practical application of this integrated approach to the wave equation is detailed. This is then followed by an explanation of the learning algorithm employed during the network's training process.

2.1. Physics-informed neural networks

In this section, we have created the following neural network architecture for the purpose of applying PINNs to the wave equation. The configuration of the network architecture delineates the layers and the quantity of neurons that will be incorporated [20]. As demonstrated in Figure 1, a standard artificial neural network consists of interconnected neuron layers that receive, process, and transmit information from the input layer to the output layer.

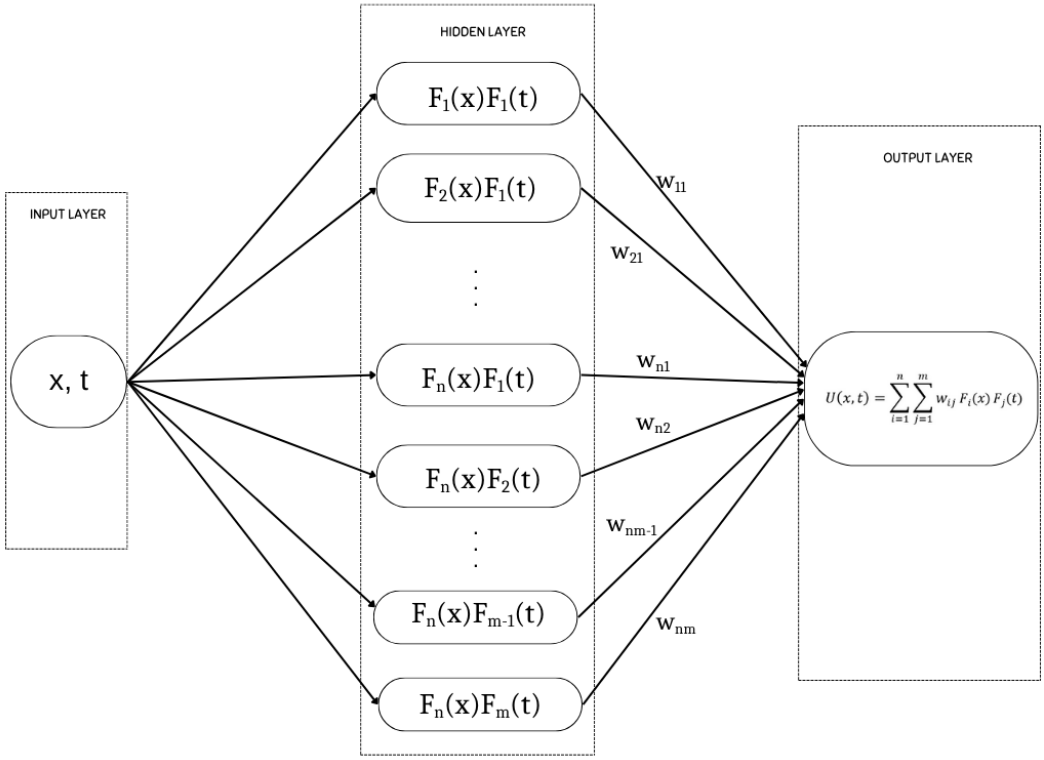

Figure 1. Artificial neural networks diagram.

Activation functions constitute a crucial component of artificial neural networks, serving to enhance the model's capacity for learning and ensuring that the optimization problem yields optimal outcomes. The primary function of activation functions in the hidden layer of artificial neural networks composed of layers is to ensure that the data from the input layer and the data in the output layer work in harmony. The Fibonacci activation function employed in the hidden layer of size nxm makes use of the Fibonacci polynomial.

The U(x, t) is the output of the PINN and is defined as follows:

$$U(x,t) = \sum_{i=1}^{n}\sum_{j=1}^{m} w_{ij}\, F_i(x)\, F_j(t).$$

In addition to hyperparameters such as the activation function, the number of iterations, the loss term weights, and the damping parameter used to improve model performance, another key design choice is the loss function. The error function is a mathematical function that calculates the extent to which the model's actual values deviate from the estimated values. Error functions have been demonstrated to play a critical role in guiding weight updates during the model training process. In this context, the most commonly used error function is the Mean Squared Error (MSE). MSE, a particularly favored metric in regression problems, calculates the mean of the squared errors in each instance by comparing the model's predictions with the actual values. The formula for MSE is as follows:

$$\mathrm{T}(\mathrm{y},\hat{\mathrm{y}}) = \frac{1}{\mathrm{N}}\sum_{\mathrm{i}=1}^{\mathrm{N}}(\mathrm{y_i} - \hat{\mathrm{y}}_\mathrm{i})^2.$$

This formula enables the model to learn more carefully and minimize large errors, approaching zero over time. This process aims to improve performance by providing feedback during model training by reducing the value of the loss function. In general, this is achieved through gradient descent [21]. The gradient represents the derivative of the loss function with respect to the model's parameters (weights and biases). Gradient descent is an iterative optimization method; in other words, it updates the model's weights step by step in the direction of the gradient until convergence is achieved. This process involves many iterations to enable the model to learn better [22]. In this study, rather than standard gradient descent, the Levenberg-Marquardt (LM) algorithm is employed, which combines the gradient descent method with the Gauss-Newton method through a damping parameter (μ), enabling faster and more stable convergence. In this approach, the gradients of the parameters are calculated separately, with respect to the weights and biases, as follows:

$$\nabla\mathrm{T}(\mathrm{w}) = \left[\frac{\partial \mathrm{T}}{\partial \mathrm{w_1}}, \frac{\partial \mathrm{T}}{\partial \mathrm{w_2}}, \ldots, \frac{\partial \mathrm{T}}{\partial \mathrm{w_n}}\right],$$

$$\nabla\mathrm{T}(\mathrm{b}) = \left[\frac{\partial \mathrm{T}}{\partial \mathrm{b_1}}, \frac{\partial \mathrm{T}}{\partial \mathrm{b_2}}, \ldots, \frac{\partial \mathrm{T}}{\partial \mathrm{b_n}}\right].$$

In this study, the bias term is set to zero, and therefore only the weight parameters are updated during the optimization process. Accordingly, the weight update rule for the LM algorithm is given by

$$w^{(k+1)} = w^{(k)} - \left(H^{(k)} + \mu \mathrm{I}\right)^{-1} \nabla T\left(w^{(k)}\right),$$

where $\nabla T\left(w^{(k)}\right)$ denotes the gradient of the loss function T with respect to the weight vector $w^{(k)}$ at the kth iteration, $H^{(k)}$ is the Hessian matrix of T with respect to $w^{(k)}$, I is the identity matrix, and μ is the damping parameter that controls the balance between the gradient descent and Gauss-Newton methods. As μ decreases, the algorithm behaves more like the Gauss-Newton method, providing faster convergence near the solution, whereas larger values of μ make the algorithm behave more like gradient descent, ensuring stability when the current estimate is far from the optimal solution.

In the flowchart above, $\mathrm{w}^{(\mathrm{k})}$, $\mathrm{T}\left(\mathrm{w}^{(\mathrm{k})}\right)$, $\nabla\mathrm{T}\left(\mathrm{w}^{(\mathrm{k})}\right)$, and $\mathrm{H}\left(\mathrm{w}^{(\mathrm{k})}\right)$ denote the quantities at the kth iteration and satisfy the following equations:

$$T(w^k) = \sum_{x_k,t_m} \left(\left(\frac{\partial^2 U(x_k,t_m)}{\partial t^2} - c^2 \frac{\partial^2 U(x_k,t_m)}{\partial x^2} \right)^2 + (U(0,t_m) - g_1(t_m))^2 + (U(1,t_m) - h_2(t_m))^2 + (U(x_k,0) - h_1(x_k))^2 + (U_t(x_k,0) - h_2(x_k))^2 \right),$$

$$w^{(k)} = \begin{bmatrix} w_{11} \\ w_{21} \\ \vdots \\ w_{n1} \\ w_{n2} \\ \vdots \\ w_{nm-1} \\ w_{nm} \end{bmatrix}_{(n\,x\,m)}, \quad \nabla T(w^{(k)}) = \begin{bmatrix} \frac{\partial T(w^{(k)})}{\partial w_{11}} \\ \frac{\partial T(w^{(k)})}{\partial w_{21}} \\ \vdots \\ \frac{\partial T(w^{(k)})}{\partial w_{n1}} \\ \frac{\partial T(w^{(k)})}{\partial w_{n2}} \\ \vdots \\ \frac{\partial T(w^{(k)})}{\partial w_{nm-1}} \\ \frac{\partial T(w^{(k)})}{\partial w_{nm}} \end{bmatrix}_{(n\,x\,m)},$$

$$H^{(k)} = \begin{bmatrix} \frac{\partial^2 T(w^{(k)})}{\partial w_{11}^2} & \cdots & \frac{\partial^2 T(w^{(k)})}{\partial w_{11}\,\partial w_{nm}} \\ \vdots & \ddots & \vdots \\ \frac{\partial^2 T(w^{(k)})}{\partial w_{nm}\,\partial w_{11}} & \cdots & \frac{\partial^2 T(w^{(k)})}{\partial w_{nm}^2} \end{bmatrix}_{(nm\,x\,nm)}.$$

2.2. Fibonacci polynomial and its properties

Within mathematics, number sequences and polynomials are frequently defined via recursive relations, with numerous well-known polynomial families constructed in this manner. In 1883, Fibonacci polynomials were introduced by the Belgian mathematician Eugene Charles Catalan and the German mathematician Ernst Erich Jacobsthal [23,24].

The polynomials:

$$F_1(x) = 1,\ F_2(x) = x, \qquad F_n(x) = xF_{n-1}(x) + F_{n-2}(x), n \geq 3.$$

The recurrence relation defined in this way is given by the following series expansion [25].

$$F_i(x) = \sum_{j=0}^{[\frac{i-1}{2}]} \binom{i-j-1}{j} x^{i-2j-1}.$$

2.3. Integration of Fibonacci-PINN to the wave equation

In this section, the application of the PINN method to solve the model under consideration is demonstrated. The equation of the output layer is defined as U(x,t).

$$U(x,t) = \sum_{i=1}^{n}\sum_{j=1}^{m} w_{ij}\, F_i(x)\, F_j(t)$$

The conversion of the output layer equation to Eqs (1) and (2) is as follows:

$$\begin{aligned} \frac{\partial^2 u}{\partial t^2} &= c^2 \frac{\partial^2 u}{\partial x^2} + f(x,t), \\ u(0,t) &= g_1(t), \\ u(1,t) &= g_2(t), \\ u(x,0) &= h_1(x), \\ u_t(x,0) &= h_2(x). \end{aligned} \tag{3}$$

By applying the equation to these transformations, the following expression is obtained:

$$\sum_{i=1}^{n}\sum_{j=1}^{m} w_{ij}\, F_i(x) \frac{\partial^2 F_j(t)}{\partial t^2} = c^2 \sum_{i=1}^{n}\sum_{j=1}^{m} w_{ij} \frac{\partial^2 F_i(x)}{\partial x^2} F_j(t) + f(x,t). \tag{4}$$

Now, using Eq (4) and condition (3), the following minimization problem is obtained:

$$\min_{w} \left\{ \sum_{x_k,t_m} \left(\left(\frac{\partial^2 U(x_k,t_m)}{\partial t^2} - c^2 \frac{\partial^2 U(x_k,t_m)}{\partial x^2} \right)^2 + (U(0,t_m) - g_1(t_m))^2 + (U(1,t_m) - g_2(t_m))^2 \right. \right.$$
$$\left. \left. + (U(x_k,0) - h_1(x_k))^2 + (U_t(x_k,0) - h_2(x_k))^2 \right) \right\}.$$

Here, the network is trained by computing the error using the training points x_k, t_m and updating the network with the LM optimization algorithm. Following the general loss function formulation introduced in Section 2.1, the problem-specific loss function $T\left(w^{(k)}\right)$ for the wave equation is constructed as the weighted sum of the residuals associated with the governing equation and the boundary/initial conditions. Each residual term is defined separately as follows:

PDE residual:

$$R_{PDE}(w) = \sum_{x_k,t_m} \left(\frac{\partial^2 U(x_k,t_m)}{\partial t^2} - c^2 \frac{\partial^2 U(x_k,t_m)}{\partial x^2} \right)^2.$$

Dirichlet residuals:

$$R_D(w) = \sum_{t_m} \left(U(0,t_m) - g_1(t_m)\right)^2 + \sum_{t_m} \left(U(1,t_m) - g_2(t_m)\right)^2 + \sum_{x_k} \left(U(x_k,0) - h_1(x_k)\right)^2.$$

Neumann residuals:

$$R_N(w) = \sum_{x_k} \left(U_t(x_k,0) - h_2(x_k)\right)^2.$$

The total loss function is then given by the weighted sum of these components:

$$\mathrm{T}(\mathrm{w}^{\mathrm{k}}) = \alpha_{PDE} R_{PDE}(w) + \alpha_D R_D(w) + \alpha_N R_N(w) \tag{5}$$

where α_{PDE}, α_D, and α_N are weighting coefficients that control the relative contribution of each residual term to the total loss.

In this study, uniform weighting is adopted for the PDE residual, boundary-condition, and initial-condition losses, i.e., $\alpha_{PDE} = \alpha_D = \alpha_N = 1$; no adaptive loss-weighting strategy is employed. This choice is maintained consistently across all numerical experiments to ensure a fair comparison. For the considered wave equations, the individual loss components decrease simultaneously during training, indicating that no severe imbalance among the PDE, boundary, and initial-condition terms is observed. This is demonstrated in Figure 6 for the examples presented in Section 3.

The process of minimizing Eq (5) using the LM algorithm to solve this optimization problem is described in the form of a flowchart, as shown in Figure 2.

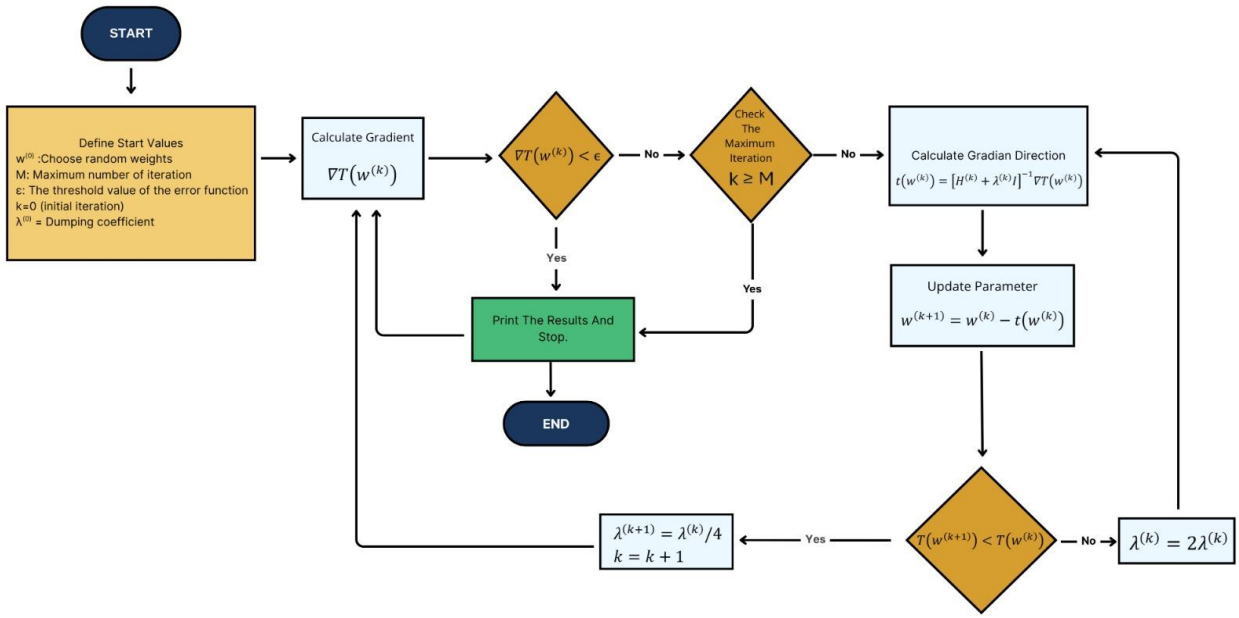

Figure 2. Flowchart of the physics-informed neural network (PINN).

3. Numerical examples

In this section, three numerical examples are presented to demonstrate the applicability and effectiveness of the proposed PINN approach. For each example, the solution obtained with the proposed model is compared against the solution obtained using the classical hyperbolic tangent (tanh) activation function. This enables a direct assessment of the improvement introduced by the proposed activation scheme. The predicted and exact solutions are further visualized by means of three-dimensional surface plots, providing a qualitative comparison of the solution behavior over the spatiotemporal domain. In addition to the relative L_2 error, the predictive accuracy of the proposed model is quantitatively evaluated for selected cases using the maximum absolute error, mean absolute error, and mean squared error metrics, thus providing a comprehensive assessment of the model's performance.

Example 1. The time-location wave equation is to be considered as follows:

$$\mathrm{u_{tt}} = \mathrm{u_{xx}}, t > 0, 0 < x < 1$$

subject to the following initial and boundary conditions:

$$u(0,t) = t^2,\ u(1,t) = 1 + t^2, \qquad t > 0,$$
$$u(x,0) = x^2,\ u_t(x,0) = 0, \qquad x\epsilon[0,1].$$

The exact solution to the equation is given by $\mathrm{u(x,t)} = x^2 + t^2$.

The data and graphical results for Example 1, for which the exact solution is known, are presented below. The corresponding results are provided in Tables 1–3 and Figure 3.

Table 1. Error values and the number of iterations required to obtain the solution for Example 1.

n = m	Number of iteration	Max. Abs. Error	Mean Abs. Error	Mean Squared error	Relative-L2
3	13	1.73×10^{-12}	7.94×10^{-13}	7.97×10^{-25}	1.12×10^{-12}
5	19	1.26×10^{-10}	3.92×10^{-11}	2.38×10^{-21}	6.16×10^{-11}
10	15	8×10^{-7}	2.7×10^{-7}	1×10^{-13}	4×10^{-7}
12	18	5.4×10^{-5}	9.2×10^{-6}	1.4×10^{-10}	1.5×10^{-5}
15	25	5.4×10^{-4}	5.2×10^{-5}	4.7×10^{-9}	8.6×10^{-5}

As demonstrated in Table 1, the proposed Fibonacci-PINN method accurately approximates the analytical solution for all discretization levels that are tested. The maximum absolute error and the relative L_2 error remain sufficiently small, indicating high numerical accuracy. As anticipated, although the approximation capacity increases with increasing polynomial degree $n = m$, the error exhibits a marked increase beyond a certain degree, owing to numerical ill-conditioning arising from the growth of the condition number of the collocation matrix. This behavior is a well-known consequence of the exponential growth of the condition number of Vandermonde matrices with increasing polynomial degree, a phenomenon closely related to the Runge phenomenon observed in high-degree polynomial interpolation at equispaced points [26,27]. To assess the improvement achieved by the proposed Fibonacci-PINN approach, the same problem is also solved using a single-hidden-layer PINN with the classical tanh activation function, trained via the Levenberg–Marquardt algorithm. The corresponding results are presented in Table 2.

Table 2. Single-layer PINN results obtained using the tanh activation function for Example 1.

H (Neuron)	Number of parameters	Number of iteration	Max. Abs. Error	Mean Abs. Error	Mean squared error	Relative-L2
4	17	200	4.89×10^{-2}	6.09×10^{-3}	6.22×10^{-5}	9.89×10^{-3}
8	33	200	6.81×10^{-2}	1.11×10^{-2}	1.89×10^{-4}	1.72×10^{-2}
12	49	200	1.04×10^{-2}	6.56×10^{-4}	9.34×10^{-7}	1.21×10^{-3}
16	65	200	1.21×10^{-2}	8.73×10^{-4}	1.49×10^{-6}	1.53×10^{-3}
20	81	200	1.03×10^{-2}	6.85×10^{-4}	9.98×10^{-7}	1.25×10^{-3}

To facilitate a direct comparison between the two approaches in terms of accuracy and computational cost, the results from both methods are summarized together in Table 3.

Table 3. Comparative performance of the Fibonacci-PINN approach and the tanh- PINN for Example 1.

Method	Model complexity	Computational cost (iteration)	Max. Abs. Error	Mean Abs. Error	MSE	Relative-L2
Fibonacci-PINN	n = m = 3	3	1.73×10^{-12}	7.94×10^{-13}	7.97×10^{-25}	1.12×10^{-12}
Fibonacci-PINN	n = m = 15	25	5.4×10^{-4}	5.2×10^{-5}	4.7×10^{-9}	8.6×10^{-5}
Tanh-PINN	H = 12 (49 parameter)	200	1.04×10^{-2}	6.56×10^{-4}	9.34×10^{-7}	1.21×10^{-3}
Tanh-PINN	H = 20 (81 parameter)	200	1.03×10^{-2}	6.85×10^{-4}	9.98×10^{-7}	1.25×10^{-3}

The visual comparisons presented in Figure 3 further confirm the excellent agreement between the exact and numerical solutions, while the corresponding error distributions reveal that the approximation errors remain negligible throughout the computational domain.

Example 2. The time-location wave equation is considered as follows:

$$\mathrm{u_{tt}} = \mathrm{u_{xx}}, t > 0, 0 < x < 1$$

subject to the following initial and boundary conditions:

$$\mathrm{u_x}(0,\mathrm{t}) = 0,\ u_x(1,t) = 0,\ t > 0,$$

$$\mathrm{u}(\mathrm{x},0) = \cos(\pi\mathrm{x}),\ \mathrm{u_t}(\mathrm{x},0) = 0,\ x\,\epsilon\,[0,1].$$

The exact solution to the equation is given by $u(x,t) = cos(\pi x)cos(\pi t)$.

The data and graphical results for Example 2, for which the exact solution is known, are presented below. The corresponding results are provided in Tables 4–6 and Figure 4.

As demonstrated in Table 4, the proposed Fibonacci-PINN method generates highly precise numerical approximations for the wave equation with Neumann boundary conditions. In the context of the experimental investigation, it is observed that among the discretization levels, which are subjected to rigorous testing, case $n = m = 10$ produces the smallest maximum absolute error, mean absolute error, mean squared error, and relative L_2 error, while requiring only 27 training iterations.

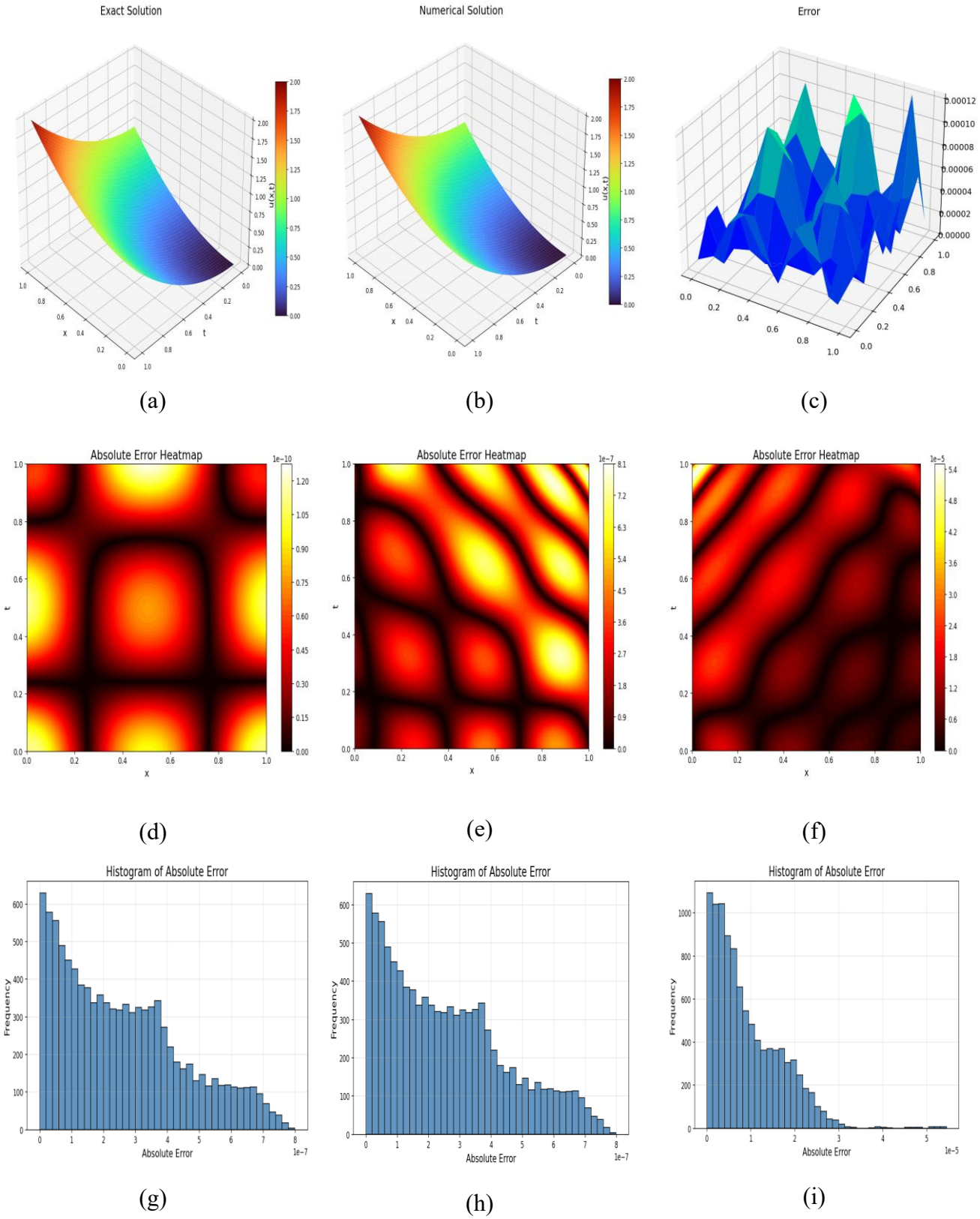

Figure 3. Numerical and error analysis for Example 1: (a) Exact solution, (b) numerical solution,and (c) absolute error 3D surface for n = m = 12. (d–f) Absolute error heatmaps for n = m = 5, 10, and 12, respectively. (g–i) Error distribution histograms for n = m = 5, 10, and 12, respectively.

Table 4. Error values and the number of iterations required to obtain the solution for Example 2.

n = m	Number of iteration	Max. Abs. Error	Mean Abs. Error	Mean squared error	Relative-L2
8	100	1.03×10^{-4}	3.81×10^{-5}	2.16×10^{-9}	9.16×10^{-5}
10	27	1.17×10^{-5}	3.22×10^{-6}	1.53×10^{-11}	7.76×10^{-6}
11	56	6.44×10^{-5}	1.78×10^{-5}	4.67×10^{-10}	4.28×10^{-5}
12	100	9.60×10^{-5}	3.29×10^{-5}	1.60×10^{-9}	7.92×10^{-5}

The corresponding results obtained using the tanh-PINN model for the same problem are presented in Table 5.

Table 5. Single-layer PINN results obtained using the tanh activation function for Example 2.

H (Neuron)	Number of parameters	Number of iteration	Max. Abs. Error	Mean Abs. Error	Mean squared error	Relative-L2
4	17	200	1.44×10^{-1}	5.95×10^{-2}	4.91×10^{-3}	1.38×10^{-1}
8	33	200	6.78×10^{-2}	2.51×10^{-2}	9.37×10^{-4}	6.02×10^{-2}
12	49	200	5.79×10^{-3}	1.92×10^{-3}	5.55×10^{-6}	4.63×10^{-3}
16	65	200	6.21×10^{-3}	1.91×10^{-3}	5.50×10^{-6}	4.61×10^{-3}
20	81	200	1.57×10^{-3}	2.98×10^{-4}	1.46×10^{-7}	7.51×10^{-4}

A comparative summary of the two methods is presented in Table 6.

Table 6. Comparative performance of the Fibonacci-PINN and the tanh- PINN for Example 2.

Method	Model complexity	Computational cost (iteration)	Max. Abs. Error	Mean Abs. Error	MSE	Relative-L2
Fibonacci-PINN	n = m = 10	27	1.17×10^{-5}	3.22×10^{-6}	1.53×10^{-11}	7.76×10^{-6}
Fibonacci-PINN	n = m = 12	100	9.60×10^{-5}	3.29×10^{-5}	1.60×10^{-9}	7.92×10^{-5}
Tanh-PINN	H = 20 (81 parameter)	200	1.57×10^{-3}	2.98×10^{-4}	1.46×10^{-7}	7.51×10^{-4}
Tanh-PINN	H = 12 (49 parameter)	200	5.79×10^{-3}	1.92×10^{-3}	5.55×10^{-6}	4.63×10^{-3}

The results in Figure 4 show excellent agreement between exact and numerical solutions, with negligible approximation errors across the domain.

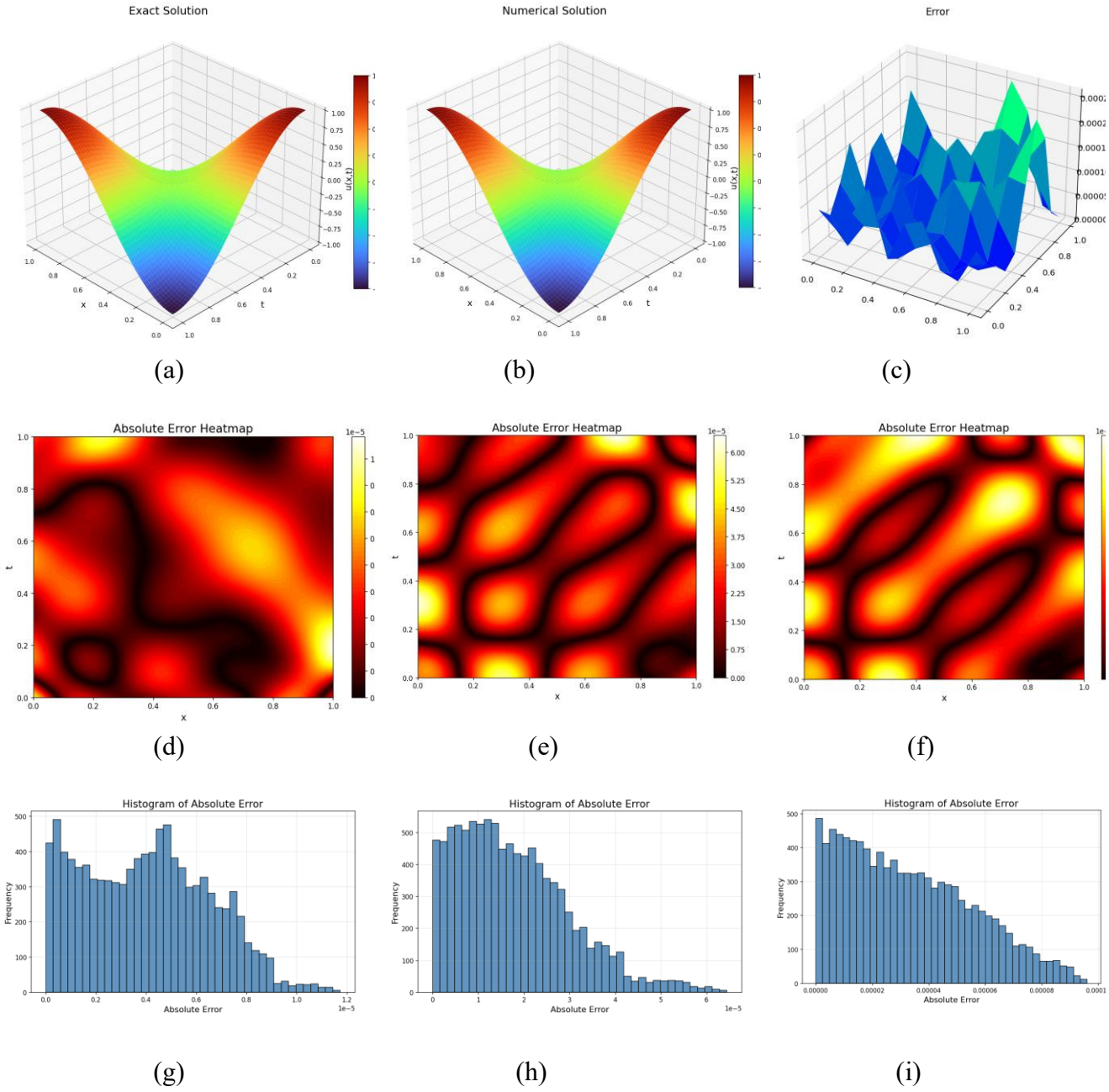

Figure 4. Numerical and error analysis for Example 2: (a) Exact solution, (b) numerical solution, and (c) absolute error 3D surface for n = m = 12. (d–f) Absolute error heatmaps for n = m = 5, 10, and 12, respectively. (g–i) Error distribution histograms for n = m = 5, 10, and 12, respectively.

Example 3: The time-location wave equation with variable coefficients is considered as follows:

$$u_{tt} - \frac{\partial}{\partial x}[(1 + x^2)u_x = f(x,t), \qquad 0 \le x \le 1, \qquad 0 \le t \le 1,$$
$$f(x,t) = e^{-t}[(1 + \pi^2(1 + x^2))\sin(\pi x) - 2\pi x\cos(\pi x)$$

subject to the following initial and boundary conditions:

$$u(x,0) = \sin(\pi x), \qquad u_t(x,0) = -\sin(\pi x),$$
$$u(0,t) = u(1,t) = 0.$$

The exact solution to the equation is given by $u(x,t) = e^{-t}sin(\pi x)$.

The data and graphical results for Example 3, for which the exact solution is known, are presented below. The corresponding results are provided in Tables 7–9 and Figure 4.

Table 7. Error values and the number of iterations required to obtain the solution for Example 3.

n = m	Number of iteration	Max. Abs. Error	Mean Abs. Error	Mean Squared Error	Relative-L2
8	100	1.81×10^{-3}	5.04×10^{-4}	3.97×10^{-7}	1.41×10^{-3}
9	21	2.44×10^{-5}	8.35×10^{-6}	1.13×10^{-10}	2.36×10^{-5}
11	26	6.84×10^{-5}	1.96×10^{-5}	6.14×10^{-10}	5.52×10^{-5}
12	24	6.54×10^{-5}	1.89×10^{-5}	6.92×10^{-10}	5.86×10^{-5}

The corresponding results obtained using the tanh-PINN model for the same problem are presented in Table 8.

Table 8. Single-layer PINN results obtained using the tanh activation function for Example 3.

H (Neuron)	Number of parameters	Number of iteration	Max. Abs. Error	Mean Abs. Error	Mean squared error	Relative-L2
4	17	200	1.44×10^{-1}	5.95×10^{-2}	4.91×10^{-3}	1.38×10^{-1}
8	33	200	6.78×10^{-2}	2.51×10^{-2}	9.37×10^{-4}	6.02×10^{-2}
12	49	200	5.79×10^{-3}	1.92×10^{-3}	5.55×10^{-6}	4.63×10^{-3}
16	65	200	6.21×10^{-3}	1.91×10^{-3}	5.50×10^{-6}	4.61×10^{-3}
20	81	200	1.57×10^{-3}	2.98×10^{-4}	1.46×10^{-7}	7.51×10^{-4}

A comparative summary of the two methods is presented in Table 9.

Table 9. Comparative performance of the Fibonacci polynomial approach and the tanh-based PINN for Example 2.

Method	Model complexity	Computational cost (iteration)	Max. Abs. Error	Mean Abs. Error	MSE	Relative-L2
Fibonacci-PINN	n = m = 10	27	1.17×10^{-5}	3.22×10^{-6}	1.53×10^{-11}	7.76×10^{-6}
Fibonacci-PINN	n = m = 12	100	9.60×10^{-5}	3.29×10^{-5}	1.60×10^{-9}	7.92×10^{-5}
Tanh-PINN	H = 20 (81 parameter)	200	1.57×10^{-3}	2.98×10^{-4}	1.46×10^{-7}	7.51×10^{-4}
Tanh-PINN	H = 12 (49 parameter)	200	5.79×10^{-3}	1.92×10^{-3}	5.55×10^{-6}	4.63×10^{-3}

The results in Figure 5 show excellent agreement between exact and numerical solutions, with negligible approximation errors across the domain.

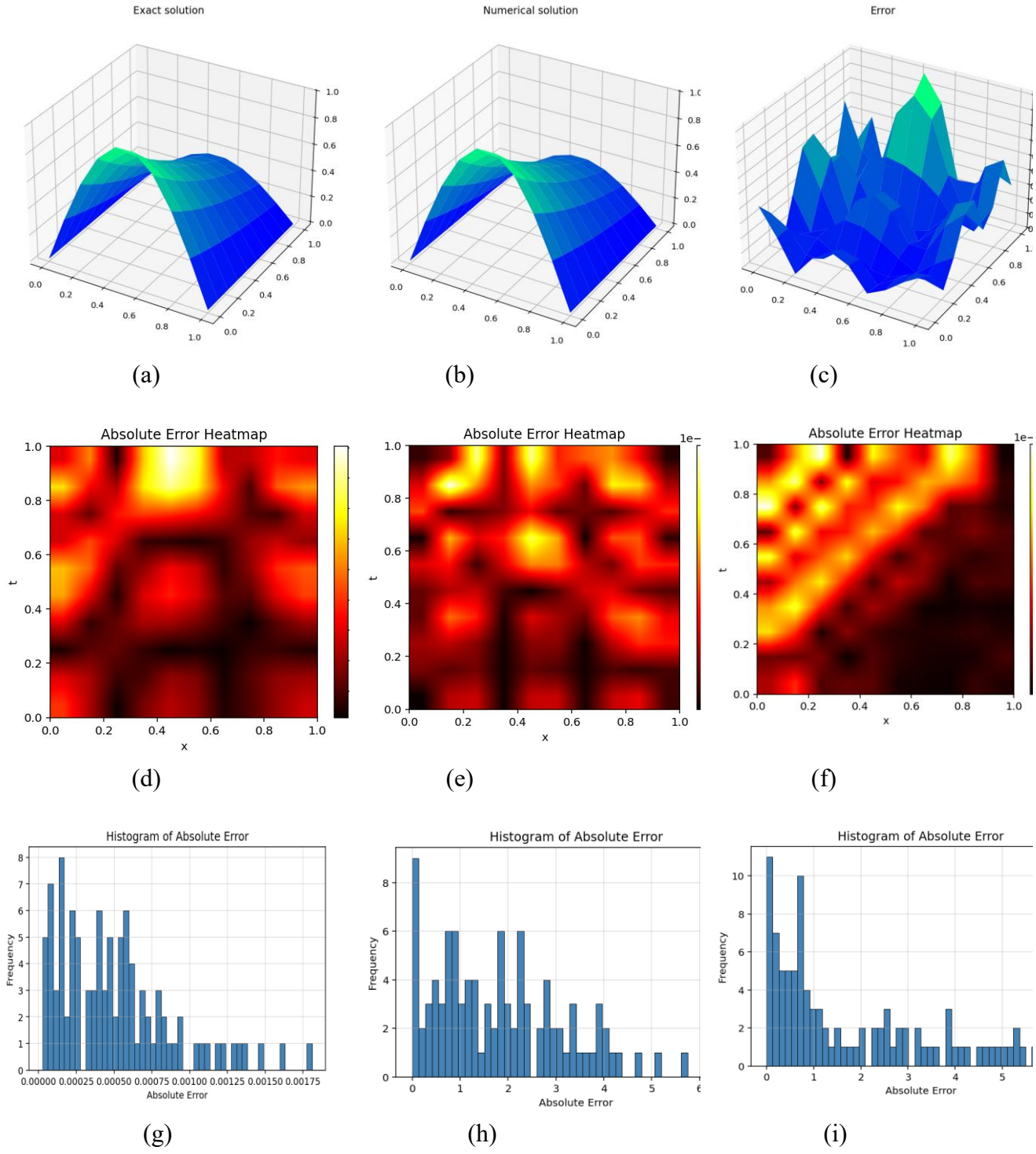

Figure 5. Numerical and error analysis for Example 3: (a) Exact solution, (b) numerical solution, and(c) absolute error 3D surface for n = m = 12. (d–f) Absolute error heatmaps for n = m=5, 10, and 12, respectively. (g–i) Error distribution histograms for n = m = 5, 10, and 12, respectively.

To further illustrate the training behavior discussed above, Figure 6 shows the evolution of the individual loss terms for each of the three examples. The simultaneous decrease of all loss

components indicates stable and balanced convergence throughout training.

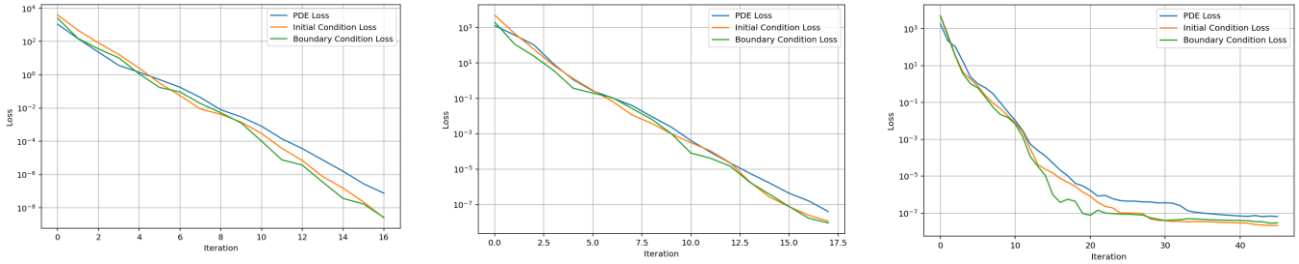

Figure 6. Training loss curves for the PDE residual, boundary-condition, and initial-condition terms for the three numerical examples: (a) Example 1, (b) Example 2, and (c) Example 3.

4. Conclusions

In this study, a Fibonacci-PINN framework was proposed for solving one-dimensional wave equations with Dirichlet and Neumann boundary conditions. The Fibonacci polynomial activation was incorporated into the PINN architecture to improve the approximation capability while preserving the physics-informed training strategy.

Three benchmark examples with known analytical solutions were considered to evaluate the effectiveness of the proposed method. The numerical results demonstrated that the Fibonacci-PINN model accurately approximates the exact solutions with very small maximum absolute errors, mean absolute errors, mean squared errors, and relative L_2 errors. Moreover, the method converged within a relatively small number of training iterations, indicating an efficient optimization process. The graphical comparisons further confirmed the excellent agreement between the exact and numerical solutions, while the corresponding error distributions showed that the approximation errors remained uniformly low over the computational domain.

The proposed Fibonacci-PINN was further validated on a linear PDE, where it achieved lower prediction error than the standard Tanh-based PINN under identical training conditions. This result suggests that the Fibonacci activation offers a practical advantage over the conventional Tanh activation, at least for the class of problems considered in this study.

Overall, the obtained results demonstrate that the proposed Fibonacci-PINN approach is an accurate, stable, and computationally efficient technique for solving wave equations. The flexibility of the proposed framework also suggests that it can be extended to more complex partial differential equations, including multidimensional, nonlinear, and fractional models, providing a promising direction for future research.

Author contributions

All authors contributed equally to this work. All authors have read and approved the final version of the manuscript for publication.

Use of Generative-AI tools declaration

The authors declare that Artificial Intelligence (AI) tools were used only for language editing. They were not used to develop the main ideas or to obtain the results of this study.

Conflict of interest

The authors state that they do not have any conflicts of interest.

References

1. A. Raina, S. Natesan, A physics-informed neural network framework for tumor-immune interactions, metastatic invasion, and haptotaxis systems, *Math. Methods Appl. Sci.*, **49** (2026), 4450–4475. https://doi.org/10.1002/mma.70355
2. H. P. Wang, Q. Ye, Z. H. Zhang, J. G. Liu, Application of multivariate bilinear neural network method to a spatial symmetric nonlinear dispersive wave model in (2+1)-dimensions, *Math. Methods Appl. Sci.*, **49** (2026), 9093–9114. https://doi.org/10.1002/mma.70516
3. J. Muhammad, A. H. Tedjani, E. Hussain, U. Younas, Exploring the exact solutions to the nonlinear systems with neural networks method, *Sci. Rep.*, **15** (2025), 36818. https://doi.org/10.1038/s41598-025-21095-2
4. F. Düşünceli, E. Çelik, Fractional approach for diffusion equations arising from oil pollution using the fractional natural decomposition method, *Int. J. Quantum Chem.*, **125** (2025), e27529. https://doi.org/10.1002/qua.27529
5. G. Seriani, S. P. Oliveira, Numerical modeling of mechanical wave propagation, *Riv. Nuovo Cimento*, **43** (2020), 459–514. https://doi.org/10.1007/s40766-020-00009-0
6. A. M. Attiya, E. M. Eldesouki, Numerical analysis for temporal and spectral responses of electromagnetic waves in spatially homogeneous time varying medium, *Sci. Rep.*, **14** (2024), 15234. https://doi.org/10.1038/s41598-024-64874-z
7. N. Bhangale, K. B. Kachhia, Fractional electromagnetic waves in plasma and dielectric media with Caputo generalized fractional derivative, *Rev. Mex. Fís.*, **66** (2020), 848–855. https://doi.org/10.31349/revmexfis.66.848
8. M. J. Alam, A. Ramady, M. S. Abbas, K. El-Rashidy, M. T. Azam, M. M. Miah, Numerical investigation of the wave equation for the convergence and stability analysis of vibrating strings, *AppliedMath*, **5** (2025), 18. https://doi.org/10.3390/appliedmath5010018
9. T. De Ryck, S. Mishra, Numerical analysis of physics-informed neural networks and related models in physics-informed machine learning, *Acta Numer.*, **33** (2024), 633–713. https://doi.org/10.1017/S0962492923000089
10. Z. Tao, H. Wang, F. Liu, Lnn-pinn: A unified physics-only training framework with liquid residual blocks, *Comput. Phys. Commun.*, **326** (2026), 110237. https://doi.org/10.1016/j.cpc.2026.110237
11. Y. Zhang, F. Liu, Noether-constrained physics-informed neural networks for the nonlinear Schrödinger equation, *Phys. Lett. A*, **589** (2026), 131802. https://doi.org/10.1016/j.physleta.2026.131802
12. M. Qasim, T. Shahzad, F. Yao, M. Z. Baber, N. Ahmed, Nonlinear water wave dynamics of data-driven soliton solutions for the Kakutani–Matsuuchi model in Ocean Engineering, *Ocean Eng.*, **365** (2026), 127401. https://doi.org/10.1016/j.oceaneng.2026.127401

13. M. Z. Baber, F. Yao, M. Qasim, Rich spectrum of data-driven soliton phenomena by using the advanced artificial neural networking, *Chaos, Soliton. Fract.*, **210** (2026), 118736. https://doi.org/10.1016/j.chaos.2026.118736
14. M. Qasim, A. Shafee, F. Yao, M. Z. Baber, Dynamics of soliton solutions for the (3+1)-dimensional CTFZK equation in plasma physics using an advanced neural networking approach, *Z. Angew. Math. Phys.*, **77** (2026), 27. https://doi.org/10.1007/s00033-025-02682-9
15. M. Qasim, A. Shafee, F. Yao, M. Z. Baber, Y. Yildirim, B. Ceesay, et al., Lump and breather interaction with different soliton solutions for the generalized doubly dispersive equation by embedded the neural networks, *Sci. Rep.*, **16** (2026), 20665. https://doi.org/10.1038/s41598-026-51557-0
16. D. Sana, *Approximating the wave equation via physics-informed neural networks: Various forward and inverse problems*, Internship report, Friedrich-Alexander-Universität Erlangen-Nürnberg, 2022. Available from: https://dcn.nat.fau.eu/wp-content/uploads/FAUMoD_DaniaSana-InternReport_PINN.pdf.
17. S. Alkhadhr, M. Almekkawy, Wave equation modeling via physics-informed neural networks: Models of soft and hard constraints for initial and boundary conditions, *Sensors*, **23** (2023), 2792. https://doi.org/10.3390/s23052792
18. A. F. Ihsan, On the neural network solution of one-dimensional wave problem, *J. RESTI (Rekayasa Sistem dan Teknologi Informasi)*, **5** (2021), 1106–1112. https://doi.org/10.29207/resti.v5i6.3565
19. A. R. Kambekar, M. C. Deo, Wave simulation and forecasting using wind time history and data-driven methods, *Ships Offshore Struct.*, **5** (2010), 253–266. https://doi.org/10.1080/17445300903439223
20. K. D. Dwivedi, Rajeev, Numerical solution of fractional order advection reaction diffusion equation with Fibonacci neural network, *Neural Process. Lett.*, **53** (2021), 2687–2699. https://doi.org/10.1007/s11063-021-10513-x
21. İ. Karabayır, O. Akbilgic, N. Taş, A novel learning algorithm to optimize deep neural networks: evolved gradient direction optimizer (EVGO), *IEEE Trans. Neural Networks Learn. Syst.*, **32** (2021), 685–694. https://doi.org/10.1109/TNNLS.2020.2979121
22. E. Seyyarer, F. Ayata, T. Uçkan, A. Karci, Applications and comparison of optimization algorithms used in deep learning, *Comput. Sci.*, **5** (2020), 90–98.
23. A. F. Horadam, E. M. Horadam, Roots of recurrence-generated polynomials, *Fibonacci Quarterly*, **20** (1982), 219–226. https://doi.org/10.1080/00150517.1982.12429991
24. M. Asci, E. Gurel, Bivariate Gaussian Fibonacci and Lucas polynomials, *Ars Comb.*, **109** (2013), 461–472.
25. E. Özkan, M. Taştan, A. Aydoğdu, Fibonacci sayılarının ailesinde 3-Fibonacci polinomları, *Erzincan Univ. J. Sci. Technol.*, **12** (2019), 926–933. https://doi.org/10.18185/erzifbed.512100
26. P. D. Brubeck, Y. Nakatsukasa, L. N. Trefethen, Vandermonde with Arnoldi, *SIAM Rev.*, **63** (2021), 405–415. https://doi.org/10.1137/19M130100X
27. L. N. Trefethen, *Approximation theory and approximation practice*, Society for Industrial and Applied Mathematics (SIAM), Philadelphia, 2013.

AIMS Press