Mechanical Engineering Advances 2026, 4(1), 4125.
doi: https://doi.org/10.59400/mea4125

Article

PDFP-enhanced physics-informed neural networks for solving higher-order PDEs: Application to engineering beam problems

Shahbaz Ahmad[1], **Neha Zaman**[2], **Muhammad Asif**[2], **Muhammad Qasim**[3], **Aaqib Hussain Shah**[4], **Anam Shahzadi**[5], **Faizan Arshid**[2], **Muhammad Israr**[1,2,*]

[1] Abdus Salam School of Mathematical Sciences, Government College University, Lahore 54600, Pakistan
[2] Department of Mathematics, Government Post Graduate College Mansehra, Mansehra 21300, Pakistan
[3] Government Degree College Havelian, Abbottadad 22010, Pakistan
[4] Department of Chemistry, Government Post Graduate College Mansehra, Mansehra 21300, Pakistan
[5] Islamabad Model College for Girls, Islamabad 44000, Pakistan
* **Corresponding author:** Muhammad Israr, m.israr_22@sms.edu.pk

CITATION

Ahmad S, Zaman N, Asif M, et al. PDFP-enhanced physics-informed neural networks for solving higher-order PDEs: Application to engineering beam problems. Mechanical Engineering Advances. 2026; 4(1): 4125. https://doi.org/10.59400/mea4125

ARTICLE INFO

Received: 5 November 2025
Revised: 4 January 2026
Accepted: 10 January 2026
Available online: 27 January 2026

Abstract: This study introduces a modern technique for solving beam equations on elastic foundations, applied to Euler–Bernoulli and Timoshenko beams based on the Winkler foundation. Conventional Physics-Informed Neural Networks (PINNs) have faced major challenges in handling large space-time domains, even when closed-form solutions exist. To address these challenges, we introduced a preconditioned PINNs loss function that incorporates prior knowledge from similar problems, thereby enhancing both productivity and accuracy. This novel method improves the generalization capability of PINNs in structural engineering, specifically for beam dynamics on elastic foundations. The effectiveness of the method is shown through numerical simulations on Euler–Bernoulli beams and extended domains for Timoshenko beams. Comparing with state-of-the-art PINNs shows that our method speeds up the convergence and precisely describes the behavior of the system, surpassing current strategies under the L2-norm metric. Besides, we examine the weight and weight loss plots, and present 3D visualizations of the best weight configurations. Further, we describe the superior performance of the proposed method. The present study explores the application of Preconditioned Devidon–Fletcher–Powell (PDFP)-enhanced Physics-Informed Neural Networks (PINNs) for solving higher-order partial differential equations, with particular emphasis on their implementation in engineering beam problems. It provides an overview of the new approach for addressing complex beam dynamics, including visualizations of the neural network architectures, PINNs convergence behavior, and representative solutions for the beam problems.

Keywords: quasi newton optimization; PDFP optimizer; neural network; PINNs; euler–bernoulli beam problem; timoshenko beam problem

1. Introduction

Beams on elastic foundations serve as an intrinsic element in civil engineering because they give us strength and stability for various applications such as railway lines, deep foundations, and parallel fibers in elastomeric blends [1, 2]. These beams play a crucial role in load distribution, minimizing structural deformations and enhancing durability, analogous to the behavior observed in the analysis of asymmetric vibrations of functionally graded annular nanoplates under thermal environments using nonlocal elasticity theory with modified nonlocal boundary conditions [3,4]. To ensure the reliability of such systems, it is necessary to understand their behavior, which gives better maintenance planning, improves machinery performance, and optimizes design processes, and exact control mechanisms [5]. Although they are widely but it is difficult to predict the dynamic behavior of beams on elastic foundations using physical observations and experiments [6, 7]. Performing multiple trials with different prototypes and material conditions is time-consuming and expensive as well. In this context, software solutions based on finite element analysis (FEA) play their role as viable alternatives; however, they lack generalization, as even minor domain changes often require starting the simulation process from scratch [8, 9]. In a similar vein, the study by Saini et al. [10] on free axisymmetric vibrations of heated non-uniform bi-directional FGM Mindlin rings further highlights the importance of accurately capturing thermo-mechanical effects and material gradation in advanced structural components, reinforcing the need for robust and generalized modeling frameworks for complex elastic systems [11, 12]. Building on these advancements, recent studies have explored surrogate-assisted and hybrid optimization strategies to efficiently predict and enhance the reliability of complex structural systems [12, 13]. Such approaches offer a promising framework for integrating high-fidelity simulations with reduced computational costs, particularly for functionally graded and elastically supported beam structures. Recently, deep learning methods became popular in finding solutions to partial differential equations (PDEs), due to their simplicity, mesh-free nature, and efficiency in solving inverse problems and high-dimensional spaces [14, 15]. The results shown by methods like the deep Ritz method [16], deep Galerkin method [17], and physics-informed neural networks (PINNs) [18] were exceptional. These methods use deep neural networks (DNNs) to minimize a loss function that is comprised of domain-specific knowledge and physical laws, leading to approximate PDE solutions [19]. Specifically, PINNs incorporate PDE residuals, boundary conditions, and observational data to solve inverse boundary value problems [20, 21]. Moreover, recent studies have explored adaptive optimization strategies and hybrid modeling approaches to further enhance the performance of deep learning-based PDE solvers [22, 23]. These works demonstrate the effectiveness of informed mutation strategies, high-fidelity numerical modeling, and intelligent system management in improving convergence, accuracy, and computational efficiency for complex engineering problems [24].

It is necessary for the effective application of PINNs to upgrade DNN parameters in order to solve PDEs. Common optimizers like Adam [25], L-BFGS [26], and Stochastic Gradient Descent (SGD) [27] are extensively used for training, but they face ill-conditioned or hard problems [28]. Consequently, researchers have invented various algorithmic improvements, such as flexible weighting schemes [28], multi-dimensional approaches [29], and physics-based optimization algorithms, including Gauss-Newton methods [30], to intensify convergence and stability in difficult conditions [31].

PINNs for beam simulations have been used in various research. For example, Bazmara et al. [32] applied PINNs to estimate nonlinear bending behavior in beams under restricted domains. Kapoor et al. [33] used PINNs to solve beam problems [34]. Despite this, their approach remained restricted by limited domain boundaries. Xu et al. [35] proposed a self-adjusting PINN framework to deal with unsettled load conditions. However, these methods are sometimes difficult to compute, end into slow convergence for sequential tasks. These challenges led to the need for introducing more efficient techniques capable of reducing the growing complexity of beam simulations [36].

In this paper, we presented an innovative framework for modeling beams on elastic foundations using Euler–Bernoulli and Timoshenko beam theories. Our method combined a preconditioner-enhanced Physics-Informed Neural Network (pPINNs) framework, focused on upgrading the computational efficiency and convergence rates of optimizers during the training process. Our primary objective was to increase the rate of convergence and mitigate the consumption of computing resources by designing novel nonlinear preconditioning techniques. We illustrated the effectiveness of this methodology and carried out many numerical experiments to show its ability to solve complex beam problems more efficiently than conventional PINNs.

The content of the paper is organized as follows: Section 1 offers a review of mathematical modelling of acetone decomposition and multistep reaction. Section 2 describes the cPINNs method with a preconditioned optimizer in detail. Section 3 presents the numerical experiments performed, and Section 4 concludes the study by summarizing the key findings and contributions of this research.

2. Preconditioned physics-informed neural networks (PINNs)

We address the general problem of solving differential equations represented as:

$$\begin{aligned} \mathcal{A}(\varphi(\mathbf{x})) &= h(\mathbf{x}) && \text{for } \mathbf{x} \in \Omega, \\ \mathcal{C}^k(\varphi(\mathbf{x})) &= j^k(\mathbf{x}) \qquad \text{for } \mathbf{x} \in \Gamma^k \subseteq \partial\Omega, && k = 1, 2, \ldots, n_\Omega. \end{aligned} \tag{1}$$

Here, $\mathcal{A}$ is a nonlinear differential operator, while $\mathcal{C}^k$ corresponds to boundary condition operators defined on boundary segments Γ^k for $k = 1, 2, \ldots, n_\Gamma$. In time-independent problems, we take time as an additional component of the input space, such that the input vector $\mathbf{x} \in \mathbb{R}^d$ includes both spatial and temporal dimensions. In this case, the initial conditions are treated as a special type of boundary condition within the spatiotemporal domain. Hence, the computational domain Ω contains both spatial and temporal components.

2.1. Neural network approximation of the solution

For approximating the solution of Equation (1), we use a Physics-Informed Neural Network (PINN) framework. The aim is to show the solution as:

$$\varphi(\mathbf{x}) \approx \varphi_{\mathrm{NN}}(\alpha, \mathbf{x}), \tag{2}$$

where $\varphi_{\mathrm{NN}} : \mathbb{R}^d \to \mathbb{R}$ is a nonlinear mapping parameterized by the trainable parameters $\alpha \in \mathbb{R}$. The overall structure of the neural network defines the functional form of φ_{NN}, containing an input layer, multiple hidden layers, and an output layer, as shown by the following equation:

$$\begin{aligned} \text{Input layer:} \quad & z_0 = W_0\mathbf{x}, \\ \text{Hidden layers:} \quad & z_L = \mathcal{F}(z_{L-1}), \quad \text{for } L = 1, 2, 3, \ldots, L-1, \\ \text{Output layer:} \quad & z_L = W_L z_{L-1} + b_L. \end{aligned} \tag{3}$$

The input layer converts the input features $\mathbf{x} \in \Omega$ into the hidden layer space through the weights $W_0 \in \mathbb{R}^{d \times nh}$. Each subsequent hidden layer processes its input z_{L-1} using a nonlinear activation function $\mathcal{F} : \mathbb{R}^{nh} \to \mathbb{R}^{nh}$, applied as a skip-connected single-layer perceptron:

$$\mathcal{F}(z_{L-1}) = z_{L-1} + \sigma_L(W_L z_{L-1} + b_L). \tag{4}$$

The parameters of each layer are represented by $\alpha_L = (\text{flatten}(W_L), \text{flatten}(b_L))$, where $W_L \in \mathbb{R}^{nh \times nh}$ and $b_L \in \mathbb{R}^{nh}$ represent the weights and biases of the L-th layer. The function flatten$(\cdot)$ converts the weight matrices and bias vectors into 1D arrays. Nonlinearity introduced $\sigma_L : \mathbb{R}^{nh} \to \mathbb{R}^{nh}$ at each hidden layer by the activation function. The output layer combines the outputs from the last hidden layer linearly to get the final result.

2.2. Training data and loss function

To train the neural network and to get the approximated solution of Equation (1), we defined a set of interior collocation points $\mathcal{D}_{int} = \{\mathbf{x}_j\}_{j=1}^{n_{int}}$, where n_{int} represents the number of points in the interior of the domain Ω. These points enforce the physics of the differential equation. We also define the boundary dataset $\mathcal{D}_{bc}^k = \{(\mathbf{x}_j^k, j_j^k)\}_{j=1}^{n_{bc}^k}$, where each $\mathcal{D}_{bc}^k$ contains n_{bc}^k points for the boundary segment Γ^k.

The training process lessens the following total loss function, which involves the interior loss and boundary loss:

$$\alpha^* = \arg \min_{\alpha \in \mathbb{R}^n} \left(\mathcal{L}_{int}(\alpha) + \mathcal{L}_{bc}(\alpha) \right). \tag{5}$$

The interior loss $\mathcal{L}_{int}(\alpha)$ enforces the PDE at the collocation points, whereas the boundary loss $\mathcal{L}_{bc}(\alpha)$ enforces the boundary conditions:

$$\begin{aligned} \mathcal{L}_{int}(\alpha) &= \frac{w_{int}}{|\mathcal{D}_{int}|} \sum_{\mathbf{x}_j \in \mathcal{D}_{\text{int}}} ||\mathcal{A}(\varphi_{\text{NN}}(\alpha, \mathbf{x}_j)) - h(\mathbf{x}_j)||^2, \\ \mathcal{L}_{bc}(\alpha) &= \sum_{k=1}^{n_\Gamma} \frac{w_{bc}}{|\mathcal{D}_{bc}^k|} \sum_{(\mathbf{x}_j^k, j_j^k) \in \mathcal{D}_{bc}^k} ||(\varphi_{\text{NN}}(\alpha, \mathbf{x}_j^k) - j_j^k)||^2. \end{aligned} \tag{6}$$

In this equation, the weights w_{int} and w_{bc} balanced the interior and boundary losses. We can determine their optimal values through hyperparameter tuning or by using adaptive strategies as discussed by Raissi et al. [18]. The boundary conditions can also be managed directly using the length factor function $\ell^k(\mathbf{x})$:

$$\varphi_{\text{NN}}(\alpha, \mathbf{x}) = \sum_{k=1}^{n_\Gamma} j^k(\mathbf{x}) \frac{\tilde{\ell}^k(\mathbf{x})}{\max_{\mathbf{x} \in \Omega} \tilde{\ell}(\mathbf{x})} \tag{7}$$

where

$$\tilde{\ell}^k(\mathbf{x}) = \frac{\prod\limits_{\substack{i=1 \\ i\neq k}}^{n_\Gamma} \ell^i(\mathbf{x})}{\sum\limits_{i=1}^{n_\Gamma} \prod\limits_{\substack{j=1 \\ j\neq i}}^{n_\Gamma} \ell^j(\mathbf{x})} \tag{8}$$

The length factor function $\ell^k(\mathbf{x})$ is constructed such that:

$$\begin{aligned} &\ell^k(\mathbf{x}) = 0, \quad \text{for } \mathbf{x} \in \Gamma^k \\ &\ell^k(\mathbf{x}) = 1, \quad \text{for } \mathbf{x} \in \Gamma^j, \quad j \neq k \\ &\ell^k(\mathbf{x}) \in (0,1), \quad \text{otherwise.} \end{aligned} \tag{9}$$

This length factor function helps in handling complex geometries, as discussed by Sukumar and Srivastava [37].

The final optimization problem is shown as:

$$\alpha^* = \arg\min_{\alpha\in\mathbb{R}^n} \mathcal{L}(\alpha) = \frac{1}{|\mathcal{D}_{int}|} \sum_{\mathbf{x}_j\in\mathcal{D}_{int}} ||\mathcal{A}(\varphi_{\text{NN}}(\alpha, \mathbf{x}_j)) - h(\mathbf{x}_j)||^2. \tag{10}$$

After training the model, the optimal parameters α^* are used to estimate the solution φ_{NN} to the differential equation. The error in the approximation is given as:

$$\epsilon(\varphi_{\text{NN}}, \varphi^*) = ||\varphi_{\text{NN}} - \varphi^*||. \tag{11}$$

This error has three components: the discretization error, the network approximation error, and the optimization error [38]. The discretization error depends on the placement of collocation points, the network approximation error relies on the representational power of the neural network, and the optimization error is linked to the effectiveness of the optimization algorithm applied.

This work focused on improving the training methods to minimize the optimization error and improve the overall performance of the PINN framework in solving complex PDEs.

2.3. Physics-Informed Neural Networks (PINN) for beam problems

Two separate Physics-Informed Neural Network (PINN) frameworks are created to solve the beam equations representing Euler–Bernoulli and Timoshenko beams. Each system has independent training with its own residual-based loss formulation and initial conditions. Automatic differentiation guarantees that all temporal and spatial derivatives used in the governing equations are formulated exactly within the neural network graph.

2.3.1. Model 1: Euler–Bernoulli beam on elastic foundation

The first model represents the Euler–Bernoulli beam on a Winkler foundation, led by the following equation:

$$u_{tt} + u_{xxxx} + p(x,t) = f(x,t), \quad x \in [0, 8\pi],\ t \in [0,1], \tag{12}$$

where $u(x,t)$ is the vertical displacement of the beam. The foundation response, $p(x,t)$, is given by:

$$p(x,t) = ku(x,t), \tag{13}$$

with the boundary conditions:

$$u(0,t) = u(8\pi,t) = 0, \quad u_x(0,t) = u_x(8\pi,t) = 0. \tag{14}$$

The external loading function is:

$$f(x,t) = (2-\pi^2)\sin(x)\cos(\pi t), \tag{15}$$

and the initial condition is:

$$u(x,0) = \sin(x), \quad u_t(x,0) = 0. \tag{16}$$

The PINN approximates the displacement field as:

$$u(x,t) \approx u_{\text{NN}}(x,t;\alpha),$$

where $\alpha \in \mathbb{R}^n$ denotes the trainable parameters of the neural network. The optimal parameters are obtained from the following loss minimization problem:

$$\alpha^* = \arg\min_{\alpha\in\mathbb{R}^n} \mathcal{L}(\alpha). \tag{17}$$

The total loss function is defined as:

$$\mathcal{L}(\alpha) = \mathcal{L}_{\text{int}}(\alpha) + \mathcal{L}_{\text{bc}}(\alpha) + \mathcal{L}_{\text{ic}}(\alpha). \tag{18}$$

Interior loss:

$$\mathcal{L}_{\text{int}}(\alpha) = \frac{1}{|\mathcal{D}_{\text{int}}|} \sum_{(x_j,t_j)\in\mathcal{D}_{\text{int}}} \left| \frac{\partial^2 u_{\text{NN}}}{\partial t^2} + \frac{\partial^4 u_{\text{NN}}}{\partial x^4} - R_u(x_j,t_j) \right|^2 \tag{19}$$

Boundary condition loss:

$$\mathcal{L}_{\text{bc}}(\alpha) = \frac{1}{|\mathcal{D}_{\text{bc}}|} \sum_{(x_j,t_j)\in\mathcal{D}_{\text{bc}}} |u_{\text{NN}}(x_j,t_j)|^2 \tag{20}$$

Initial condition loss:

$$\mathcal{L}_{\text{ic}}(\alpha) = \frac{1}{|\mathcal{D}_{\text{ic}}|} \sum_{x_j\in\mathcal{D}_{\text{ic}}} |u_{\text{NN}}(x_j,0) - u(x_j,0)|^2 \tag{21}$$

The optimal parameter vector α^* is obtained by minimizing the composite loss function $\mathcal{L}(\alpha)$, ensuring that the neural network solution satisfies the governing Euler–Bernoulli beam equation along with the prescribed boundary and initial conditions.

The flowchart in **Figure 1** shows the architecture and the steps used in solving beam dynamics using the Physics-Informed Neural Networks (PINNs) approach. The network architecture has the following key components:

The input layer consists of collocation points (x,t), which represent the spatial and temporal variables. These are carried through hidden layers, and the result is compared with the target function $f = N(u)$, where u is the displacement. A set of optimizers, like Adam, LBFGS, and PDFP, is applied to minimize the loss function. The training loop finds out the Mean Squared Error (MSE) between the approximated and exact solutions, including residuals and boundary conditions, until convergence is achieved $(MSE < \epsilon)$.

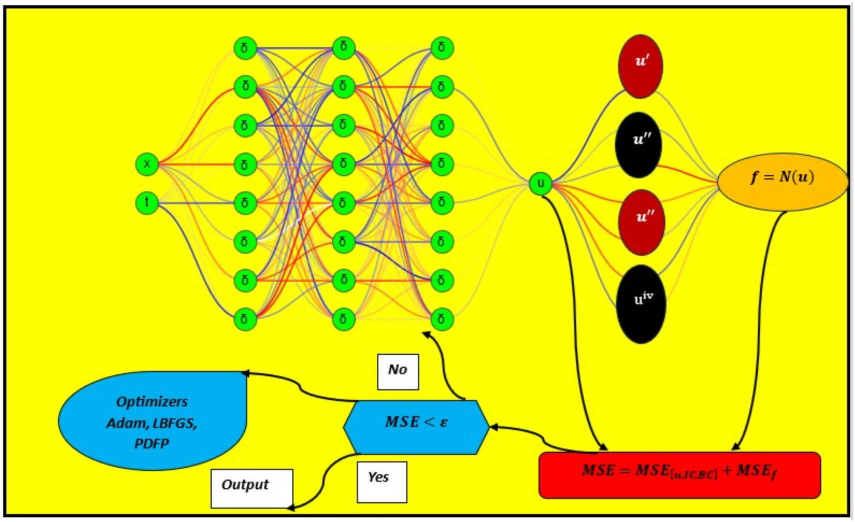


Figure 1. Neural network architecture for PINN solution of beam dynamics.

This structure makes the network capable of approximating the beam's displacement and velocity under various conditions.

2.3.2. Model 2: Timoshenko beam on elastic foundation

The second model expresses the Timoshenko beam theory on a Winkler foundation, headed by the following coupled equations:

$$\phi_{tt} - \phi_{xx} + (\phi - u_x) = 0, \tag{22}$$

$$u_{tt} + (\phi - u_x)_x + ku = h(x,t), \tag{23}$$

where $\phi(x,t)$ is the cross-sectional rotation, $u(x,t)$ is the vertical displacement, and $h(x,t)$ represents the external loading, which in this case is supposed to be $\cos(t)$. The boundary conditions are:

$$\phi(0,t) = \phi(3\pi,t) = u(0,t) = u(3\pi,t) = 0. \tag{24}$$

The initial conditions for ϕ and u are shown as:

$$\phi(x,0) = \frac{3\pi}{2}\cos(x) + \left(x - \frac{3\pi}{2}\right), \quad u(x,0) = \frac{3\pi}{2}\sin(x), \tag{25}$$

and $\phi_t(x,0) = u_t(x,0) = 0$.

The PINN approximates both the displacement and rotation fields as:

$$u(x,t) \approx u_{\mathrm{NN}}(x,t;\alpha), \quad \phi(x,t) \approx \phi_{\mathrm{NN}}(x,t;\alpha), \tag{26}$$

where $\alpha \in \mathbb{R}^n$ represents the trainable parameters of the neural networks. The optimal parameters are obtained by minimizing the total loss in Equation (18).

Interior loss:

$$\mathcal{L}_{\text{int}}(\alpha) = \frac{1}{|\mathcal{D}_{\text{int}}|} \sum_{(x_j,t_j)\in\mathcal{D}_{\text{int}}} \Big\{ \Big[\frac{\partial^2 \phi_{\text{NN}}}{\partial t^2} - \frac{\partial^2 \phi_{\text{NN}}}{\partial x^2} + \big(\phi_{\text{NN}} - \frac{\partial u_{\text{NN}}}{\partial x}\big) - R_\phi(x_j,t_j) \Big]^2 + \Big[\frac{\partial^2 u_{\text{NN}}}{\partial t^2} + \frac{\partial^2 u_{\text{NN}}}{\partial x^2} + k\, u_{\text{NN}} - R_u(x_j,t_j) \Big]^2 \Big\} \tag{27}$$

Boundary condition loss:

$$\mathcal{L}_{\text{bc}}(\alpha) = \frac{1}{|\mathcal{D}_{\text{bc}}|} \sum_{(x_j,t_j)\in\mathcal{D}_{\text{bc}}} \big[|\phi_{\text{NN}}(x_j,t_j)|^2 + |u_{\text{NN}}(x_j,t_j)|^2 \big] \tag{28}$$

Initial condition loss:

$$\mathcal{L}_{\text{ic}}(\alpha) = \frac{1}{|\mathcal{D}_{\text{ic}}|} \sum_{x_j\in\mathcal{D}_{\text{ic}}} \big[|\phi_{\text{NN}}(x_j,0) - \phi(x_j,0)|^2 + |u_{\text{NN}}(x_j,0) - u(x_j,0)|^2 \big] \tag{29}$$

The optimal parameters α^* are obtained by minimizing $\mathcal{L}(\alpha)$, ensuring that the neural network solutions satisfy the Timoshenko beam equations along with the prescribed boundary and initial conditions.

Figure 2 represents an expanded version of the previous neural network architecture where higher-order derivatives such as displacement, velocity, acceleration, and higher derivatives of the displacement are included in the model. The main components of this figure are:

The network included not only the displacement but also its derivatives, such as $\partial u/\partial x$, $\partial u/\partial t$, and $\partial^2 u/\partial x^2$, $\partial^2 u/\partial t^2$ to model complicated behaviors. Same as the previous flowchart, optimizers like Adam, LBFGS, and PDFP are used, but now the model also accounts for the computation of higher-order derivatives, improving the accuracy and adaptability of the simulation.

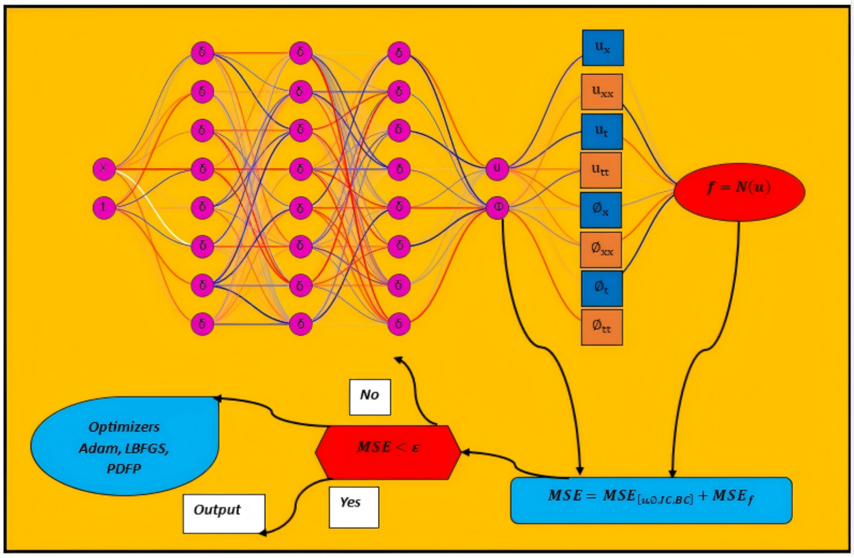


Figure 2. Flowchart for extended PINN-based beam dynamics with higher-order derivatives.

This approach can model more complex dynamic behaviors in beam problems by integrating higher-order spatial and temporal derivatives into the PINN framework.

2.4. Preconditioned DFP optimizer

To train the causal PINN architectures for the above chemical model efficiently, we aaply the Preconditioned Davidon–Fletcher–Powell (PDFP) optimizer [39]. This quasi-Newton (QN) optimizer gathers the curvature approximation with a nonlinear preconditioning strategy, which makes it suitable for rigid, nonlinear ODE systems such as chemical reaction kinetics.

Conventional optimizers like Adam or AdaGrad have poor performance for stiff kinetics because their step sizes are not scaled according to local curvature. The PDFP approach avoids this by creating an approximate inverse Hessian $\mathbb{B}^{(k+1)}$ that satisfies the secant condition

$$\mathbb{B}^{(k+1)} s^{(k)} = y^{(k)}, \tag{30}$$

where $s^{(k)} = \theta^{(k+1)} - \theta^{(k+1/2)}$ and $y^{(k)} = \nabla\mathcal{L}(\theta^{(k+1)}) - \nabla\mathcal{L}(\theta^{(k+1/2)})$. The intermediate iterate $\theta^{(k+1/2)}$ is obtained through a nonlinear preconditioning step that rescales gradients according to the PDE stiffness.

Since the secant condition alone can not be uniquely determined $\mathbb{B}^{(k+1)}$, additional constraints (symmetry and positive definiteness) yield the DFP update rule. The general Broyden family of quasi-Newton updates is

$$B^{(k+1)} = B^{(k)} - \frac{B^{(k)} s^{(k)} (s^{(k)})^T B^{(k)}}{(s^{(k)})^T B^{(k)} s^{(k)}} + \frac{y^{(k)} (y^{(k)})^T}{(y^{(k)})^T s^{(k)}} + \phi^{(k)} \left((s^{(k)})^T B^{(k)} s^{(k)} \right) v^{(k)} (v^{(k)})^T, \tag{31}$$

where

$$v^{(k)} = \frac{y^{(k)}}{(y^{(k)})^T s^{(k)}} - \frac{B^{(k)} s^{(k)}}{(s^{(k)})^T B^{(k)} s^{(k)}}. \tag{32}$$

Different values of $\phi^{(k)}$ produce the BFGS ($\phi^{(k)} = 0$) or DFP ($\phi^{(k)} = 1$) methods. A convex combination form:

$$B^{(k+1)} = (1 - \phi^{(k)}) B^{(k+1)}_{\text{BFGS}} + \phi^{(k)} B^{(k+1)}_{\text{DFP}}. \tag{33}$$

$B^{(k+1)}$ remains positive definite for $\phi^{(k)} \in [0, 1]$, confirming numerical stability.

The compact matrix form of the preconditioned DFP update is

$$\begin{aligned} B^{(k+1)} = B^{(o)} &- \begin{bmatrix} B^{(o)} s^{(k)} & y^{(k)} \end{bmatrix} \begin{bmatrix} s^{(k)} B^{(o)} s^{(k)} & L^{(k)} \\ (L^{(k)})^T & -D^{(k)} \end{bmatrix}^{-1} \begin{bmatrix} s^{(k)} B^{(o)} \\ (y^{(k)})^T \end{bmatrix} \\ &+ \phi^{(k)} \begin{bmatrix} A^T & B^T \end{bmatrix} \begin{bmatrix} A \\ B \end{bmatrix}, \end{aligned} \tag{34}$$

where $B^{(o)}$ is the initial Hessian approximation, $A = (s^{(k)})^T B^{(k)} s^{(k)}$, and $B = v^{(k)} (v^{(k)})^T$. The scalar coefficient

$$\phi^{(k)} = \frac{1}{1 + e^{-\|y^{(k)}\|}} \tag{35}$$

alters automatically between DFP and BFGS behavior, increasing curvature sensitivity when there are large changes in gradient.

The model parameters are then modified as

$$\theta^{(k+1)} = \theta^{(k+1/2)} - \alpha^{(k+1/2)} (\mathbb{B}^{(k+1)})^{-1} \nabla \mathcal{L}(\theta^{(k+1/2)}), \tag{36}$$

where $(\mathbb{B}^{(k+1)})^{-1}$ is efficiently calculated using the Sherman–Morrison–Woodbury identity. Momentum is incorporated as

$$v^{(k+1/2)} = (1-\mu) v^{(k-1/2)} + \mu p^{(k+1/2)}, \tag{37}$$

and the global update becomes

$$\theta^{(k+1)} = \theta^{(k+1/2)} + \alpha^{(k+1/2)} v^{(k+1/2)}. \tag{38}$$

3. Numerical experiments

This section analyzes the dynamic response of the Euler–Bernoulli beam model with the help of a series of numerical experiments conducted with the proposed preconditioned Physics-Informed Neural Network (pPINN) framework. To illustrate the effectiveness of the method, its performance is measured against several established PINN variants, including self-adaptive PINNs (SA-PINNs) [40], gradient-enhanced PINNs (gPINNs) [6], adaptive activation PINNs (Adap. PINNs) [27] standard PINNs [18], wavelet-based PINNs (Wav. PINNs) [41], and causal PINNs (cPINNs) [42].

In the primary network configuration, there are four fully connected hidden layers with 20 neurons each. For comparison, a larger architecture having the same depth but 200 neurons in each hidden layer is also analyzed. A combination of the Adam optimizer and a preconditioned limited-memory Broyden–Fletcher–Goldfarb–Shanno (LBFGS) algorithm is used to perform model training, with learning rates nominated to ensure stable and efficient convergence.

The computational domain is segmented using $N_t = 100$ temporal points, $N_i = 500$ initial-condition points, $N_b = 1{,}000$ boundary-condition points, and $N_{\text{int}} = 10{,}000$ interior collocation points. The weighting coefficients related to the interior and residual loss terms, w_{int} and w_{res}, are both fixed at 1. Model accuracy is computed using the relative L^2 error, represented by $\mathcal{R}$, defined as

$$\mathcal{R} = \frac{\|u_{NN} - u_{exact}\|_2}{\|u_{exact}\|_2} \times 100, \tag{39}$$

where u_{exact} indicates the analytical solution and u_{NN} represents the neural network prediction.

To make a comparison, all alternative PINN models are trained for 500 epochs, starting with the learned parameters from the main pPINN model. The Hyperbolic tangent activation function is used by the principal network. With a momentum coefficient of 0.9, global learning rate of 0.1 and local learning rate of 0.01 are adopted. In this setup, the preconditioned LBFGS plays a role as the global optimizer, while Adam is used for local updates.

These experiments give us a detailed analysis of the pPINN structure and highlight its productivity as compared to other modern PINNs methodologies for modeling and simulation of beam dynamics on elastic foundations.

3.1. Solution of the Euler–Bernoulli beam equation

Here we present the numerical solution of the Euler–Bernoulli beam Equation (12) using different variants of PINN-based strategies, and the use of nonlinear preconditioning within the optimization process. The preconditioned PINN (pPINN) approach is calculated against several standard variants of PINNs. This includes the methods, i.e., gradient-enhanced PINNs (gPINNs), adaptive-activation PINNs (Adap. PINNs), vanilla PINNs (vPINNs), wavelet-based PINNs (Wav. PINNs), causal PINNs (cPINNs), and self-adaptive PINNs (SA-PINNs).

For the prototype of a beam with stiffness parameter $k = 1$, we make the quantitative comparisons which are summarized in **Table 1**. In order to predict the beam displacement at time $t = 1$ [42], the concurrent methods show comparatively large relative errors, which reflect the drawbacks in their convergence behavior and solution accuracy for this advanced structural PINNs model.

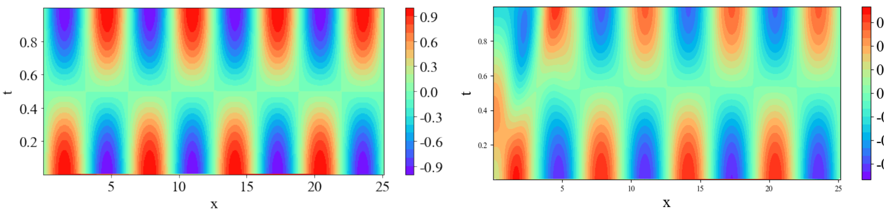
On the other side, the pPINN method achieves a high level of accuracy on fewer training iterations. As reported in **Table 1**, the relative percentage error obtained with pPINNs is 1.04%, illustrating its capability in capturing the beam's displacement response. These results highlight the benefit of nonlinear preconditioning in stabilized training and enhanced predictive performance for complex beam dynamics problems.

Table 1. Relative L^2 errors ($\mathcal{R}$) and number of training epochs for different PINN-based methods solving the Euler–Bernoulli beam at $t = 1$ with stiffness $k = 1$.

Method	**PINNs**	**SA-PINNs**	**gPINNs**	**Adap. PINNs**	**Wav. PINNs**	**cPINNs**	**pPINNs**
$\mathcal{R}$ (%)	5.34	5.25	3.53	5.35	4.48	0.07	1.01
Epochs	10,000	10,000	10,000	10,000	10,000	10,000	500

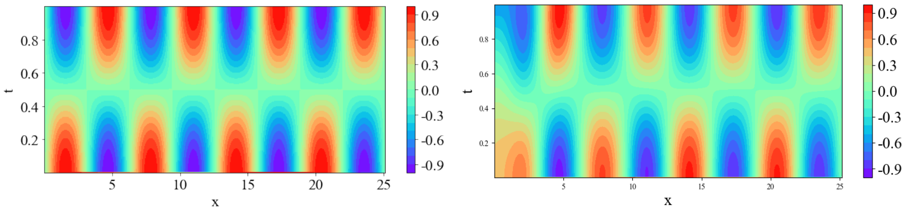
3.2. Benchmarking PINN approaches for the Euler–Bernoulli beam

Figure 3 illustrates a comparison of different PINNs-based solutions for the Euler–Bernoulli beam problem at a specific time. The results in **Figure 3b–g** show the predictions from several PINNs methods, including vanilla PINNs (**Figure 3b**), self-adaptive PINNs (SA-PINNs) (**Figure 3c**), gradient-enhanced PINNs (gPINNs) (**Figure 3d**), adaptive PINNs (Adap. PINNs) (**Figure 3e**), wavelet PINNs (Wav. PINNs) (**Figure 3f**), and causal PINNs (cPINNs) (**Figure 3g**). As shown in the figure, these methods fail to predict the displacement of the beam accurately, especially at the initial simulation stages. The errors are highlighted in the initial phase, where the displacement predictions diverge significantly from the exact solution.

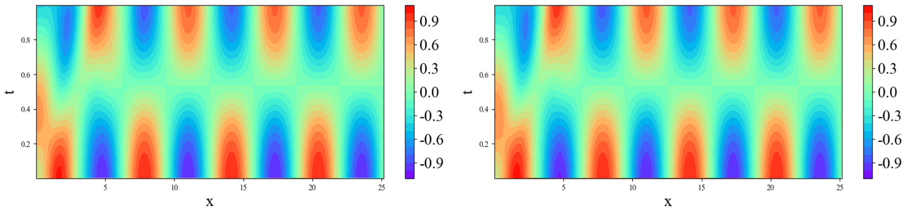


(a) Exact solution. **(b)** Standard PINNs.

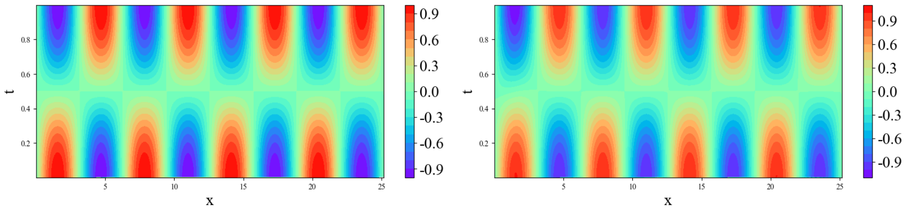
Figure 3. *Cont.*



(c) SA-PINNs. **(d)** Gradient-enhanced PINNs (gPINNs).



(e) Adaptive PINNs. **(f)** Wavelet PINNs (Wav. PINNs).



(g) Causal PINNs (cPINNs). **(h)** Preconditioned PINNs (pPINNs).

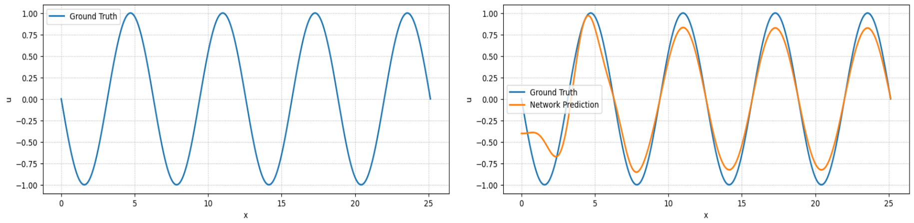
Figure 3. Comparison of predicted displacement solutions for the Euler–Bernoulli beam using different PINN-based methods.

However, the results shown in **Figure 3h**, which correspond to the pPINNs method, illustrate a considerable improvement. With the integration of a preconditioning-respecting loss function, pPINNs achieve rapid convergence and displacement predictions with accuracy. The displacement patterns obtained using pPINNs reach the exact solution closely, highlighting that the performance of the PINNs model greatly improves due to the preconditioned optimizer.

This figure indicates the efficacy of the preconditioned PINNs approach (pPINNs) in improving the accuracy and convergence rate compared to other conventional PINNs methods, which face challenges in producing precise results in the same time frame.

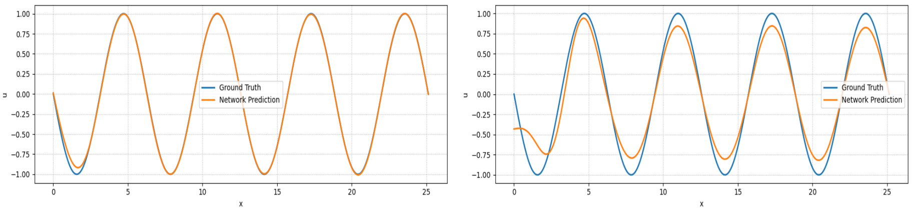
3.3. Different PINNs approaches for Euler–Bernoulli beam at $t = 1.00$

Figure 4 further shows the performance of different PINNs-based methods for the Euler–Bernoulli beam problem in the next time step, specifically at $t = 1.00$. This figure directly compared the exact solution (**Figure 4a**) with the predictions obtained using different PINNs approaches (**Figure 4b–g**).



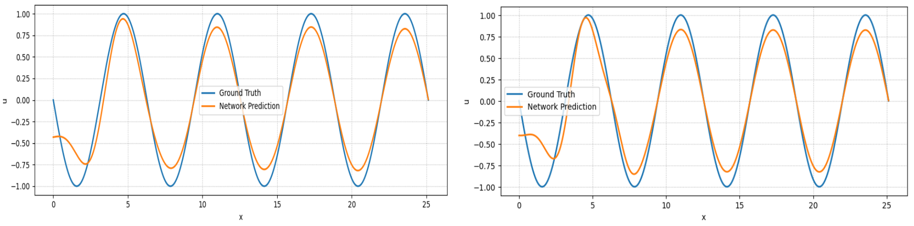
(a) Exact Solution. **(b)** PINNs.

Figure 4. *Cont.*



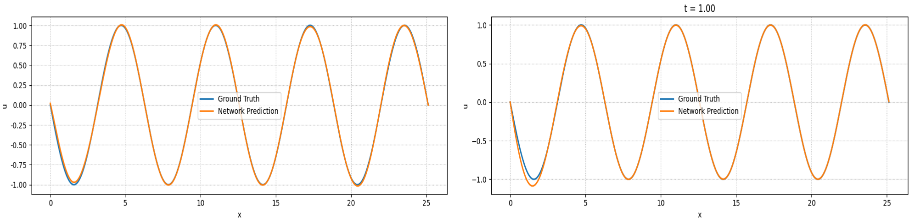
(c) SA-PINNs.

(d) Gradient-enhanced PINNs (gPINNs).



(e) Adaptive PINNs.

(f) Wavelet PINNs (Wav. PINNs).



(g) Causal PINNs (cPINNs).

(h) Preconditioned PINNs (pPINNs).

Figure 4. Visual comparison of predicted displacement fields for the Euler–Bernoulli beam at $t = 1.00$ using various PINN-based approaches.

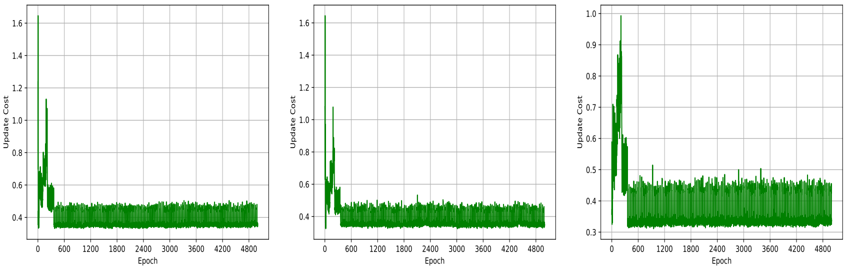
Figure 4b–g shows the results from PINNs, SA-PINNs, gPINNs, Adap. PINNs, Wav. PINNs, and cPINNs. As with the previous figure, the other models (except pPINNs) show noticeable differences when compared to the exact solution. The displacement profiles in these panels indicate that the models fail to correspond with the exact beam displacement, specifically in the initial stages of the simulation, where the errors remain significant.

On the other hand, **Figure 4h** illustrates the assumptions obtained using pPINNs, which produce much more accurate displacement profiles. The pPINNs method, which consists of a preconditioned optimizer, successfully reduces the error and produces results that matches closely with the exact solution. The rapid convergence and accuracy of displacement predictions shown in **Figure 4h** further comprehend the efficacy of the preconditioning strategy in upgrading the performance of the PINNs framework.

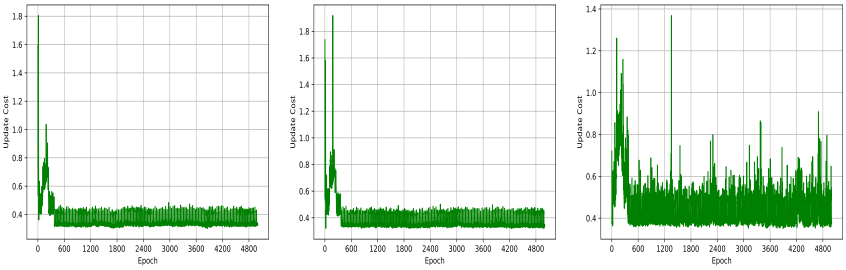
In conclusion, **Figure 4** focuses the benefits of using the pPINNs approach, particularly at $t = 1.00$, where it performs superior than the other methods with respect to both accuracy and convergence speed.

The **Figure 5** demonstrates the modified cost of different PINNs solutions for the Euler–Bernoulli beam problem. It indicates the convergence of the Mean Squared Error (MSE) during network training using different PINN methods. The Ideal convergence is shown by the exact solution, where the error is observed to be decreased rapidly in the initial iterations. On the other hand, standard PINNs exhibit a slower convergence rate and several noticeable oscillations, particularly in later stages. The Self-Adaptive

PINNs (SA-PINNs) show fast convergence compared to standard PINNs, although there remain some oscillations in reducing of error. Gradient-Enhanced PINNs show an enhanced rate of convergence, exhibiting the effectiveness of improving the slope during the optimization process. Adaptive PINNs further rectify the training process, generating smoother convergence curvatures. Wavelet PINNs (Wav. PINNs) show slower convergence and are pereiled to higher error fluctuations, especially in the later stages of training. Causal Physics-Informed Neural Networks (cPINNs) show the most consistent and steady decrease in error across iterations, indicating stability. Lastly, preconditioned PINNs (pPINNs) obtain a satisfactory solution along with fastest and most accurate convergence with few iterations. This indicates the productivity of the preconditioning technique in the optimizing the training process.



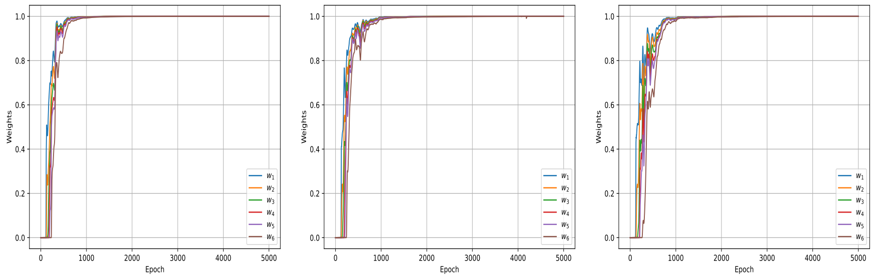
(a) PINNs. **(b)** SA-PINNs. **(c)** Gradient-Enhanced PINNs (gPINNs).



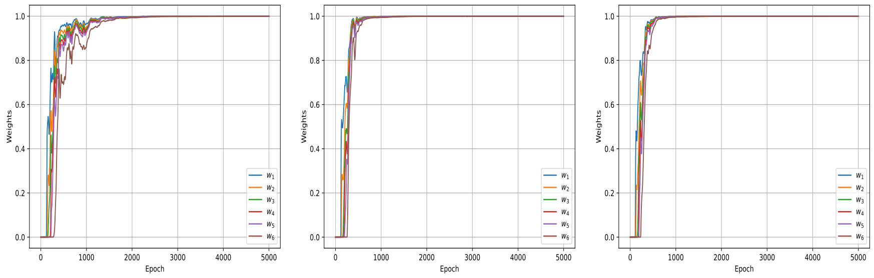
(d) Adaptive PINNs. **(e)** Wavelet PINNs (Wav. PINNs). **(f)** Causal PINNs (cPINNs).

Figure 5. Comparison of modified rates during training for various PINN-based methods applicable on the Euler–Bernoulli beam problem.

The **Figure 6** highlights the consistent weight growth during training for various PINNs-based methods applied to the Euler–Bernoulli beam problem. The plot demonstrates the variation in network weights as the model reaches the feasible solution. In case of the exact solution, the weights rapidly gain stability, due to the ideal convergence of the system. For traditional PINNs, the weights display noticeable fluctuations, especially in the initial iterations, reflecting a slower convergence rate. SA-PINNs demonstrate a more systematic and controlled weight update with a gain of smoother convergence. Gradient-Enhanced PINNs show rapid and better weight convergence then the previous ones, showing enhanced optimization. Adaptive PINNs are leading to achieve even smoother weight transitions, which help in obtaining more upgraded solutions. Wavelet PINNs, on the other hand, display slower and ill-conditioned weight updates, majorly due to the built-in noise in the wavelet-based approach. cPINNs exhibit the most compatible and controlled weight adjustments, demonstrating a resilient learning process. Lastly, pPINNs show the fastest and most accurate weight convergence, focusing the efficiency of the preconditioning technique in achieving optimal solutions quickly. The evaluation of the weights across these methods indicates that the introduction of adjustable strategies, especially pPINNs, significantly improves both the speed and accuracy of the training process.



(a) PINNs. **(b)** SA-PINNs. **(c)** Gradient-Enhanced PINNs (gPINNs).

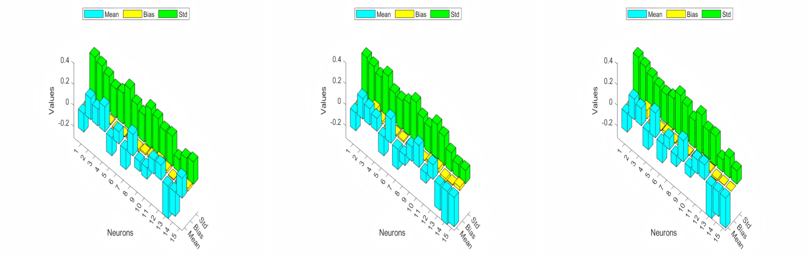


(d) Adaptive PINNs. **(e)** Wavelet PINNs (Wav. PINNs). **(f)** Causal PINNs (cPINNs).

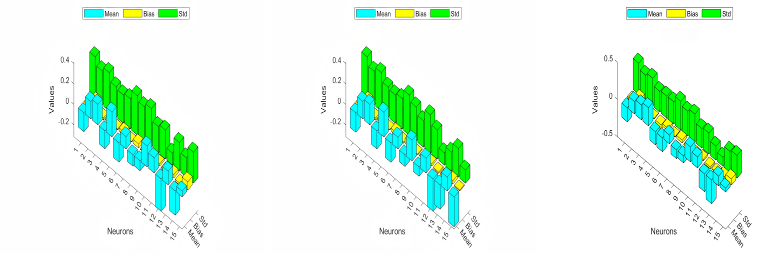
Figure 6. Comparison of the most paramount weights during training for different PINN-based techniques applied to the Euler–Bernoulli beam problem.

3.4. Analysis of best weights in PINNs for Euler–Bernoulli's beam

Figure 7 shows the best weights that play a major role to get the solution of the Euler–Bernoulli beam using various PINNs methods. Each panel in this figure demonstrates the contribution of weights of different PINNs methods to obtain the solution at different stages of training.



(a) Exact Solution. **(b)** PINNs. **(c)** SA-PINNs.



(d) Gradient-Enhanced PINNs (gPINNs). **(e)** Adaptive PINNs. **(f)** Wavelet PINNs (Wav. PINNs).

Figure 7. Visual representation of the most powerful learned weights for various PINN-based methods applied to the Euler–Bernoulli beam problem.

In **Figure 7a**, the exact solution illustrates the ideal weight distribution, serving as a reference for comparison with all numerical methods. **Figure 7b** shows the results obtained using the standard PINNs method. The weights exhibit noticeable fluctuations, indicating slower convergence and difficulty in accurately approximating the exact solution within a limited number of training epochs.

In **Figure 7c**, the SA-PINNs method demonstrates improvement over standard PINNs, where the weights converge more rapidly and show reduced oscillations. However, the accuracy still remains below that of the exact solution. In **Figure 7d**, the Gradient-Enhanced PINNs (gPINNs) method achieves further improvement, with smoother weight evolution and faster convergence, demonstrating better training efficiency compared to both PINNs and SA-PINNs.

Figure 7e presents the Adaptive PINNs method, where the weights exhibit stable and rapid convergence. This highlights the effectiveness of the adaptive weighting strategy in guiding the optimization process toward an improved solution.

Finally, **Figure 7f** shows the Wavelet PINNs (Wav. PINNs) method. Although this approach performs better than the standard PINNs, the convergence behavior is less smooth compared to Adaptive PINNs, indicating that further refinement of the wavelet-based mechanism may be beneficial.

3.5. Solution of Timoshenko beam model

The Timoshenko beam model (Equations (22)–(25)) presents a more comprehensive illustration of beam deflections and rotations, responsible for shear deformations and rotational inertia effects that were not observed in the Euler–Bernoulli model. This increases its efficacy for beams with high aspect ratios or those exposed to significant shear forces. On solution of the Timoshenko beam model, engineers can accurately and precisely identify the deflections and rotations of the beam, which is beneficial for evaluating the structural stability of a system, which also ensures the design specifications are met, and avoids failures in the system's operation.

In this study, we solved the Timoshenko beam equations by applying various PINN-based methods and compared their performance with respect to the displacement (u^*) and rotation (θ^*) at $t = 1$ and $k = 1$. The results which are obtained from these methods are shown in **Table 2**, which shows the comparison among the accuracy of different methods, and includes the PINNs, self-adaptive PINNs (SA-PINNs), gradient-enhanced PINNs (gPINNs), adaptive PINNs (Adap. PINNs), wavelet PINNs (Wav. PINNs), causal PINNs (cPINNs), and the preconditioned PINNs (pPINNs).

Table 2. Relative errors ($\mathcal{R}$) for displacement (u^*) and rotation (θ^*), with the number of training phases, for various PINN-based methods solution of the Timoshenko beam at $t = 1$ with stiffness $k = 1$.

Method	**PINNs**	**SA-PINNs**	**gPINNs**	**Adap. PINNs**	**Wav. PINNs**	**cPINNs**	**pPINNs**
u^*	118.17	133.15	112.17	230.78	228.46	1.6×10^{-6}	1.15×10^{-4}
θ^*	8.58	7.76	39.73	9.16	9.56	7.12×10^{-6}	2.01×10^{-5}
Epochs	10,000	10,000	10,000	10,000	10,000	10,000	500

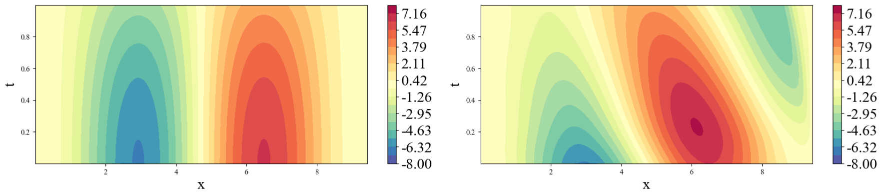
As shown in **Table 2**, the displacement (u^*) and rotation (θ^*) predictions differ noticeably across all the methods. For example, the displacement predictions for the PINNs method are 119.17, while the SA-PINNs and gPINNs methods yield 137.15 and 119.17, respectively. Moreover, the pPINNs method reaches an accuracy with a minor displacement error of 1.1×10^{-4}, which shows that the integration of preconditioning leads to a remarkable improvement in accuracy. Similarly, the rotation values show that methods like SA-PINNs and gPINNs give higher error values (6.56 and 38.63, respectively), whereas the pPINNs give us a smaller error (2.1×10^{-5}), again, this shows the effectiveness of preconditioning in improving accuracy.

The table also illustrates the epochs required for each method to achieve convergence. While all methods (except pPINNs) need 10,000 epochs, pPINNs need 500 epochs to reach convergence, reflecting the efficiency of the preconditioned approach to accelerate training and produce accurate results in fewer iterations.

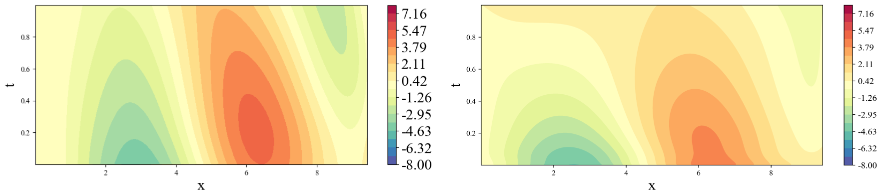
In conclusion, **Table 2** illustrates that the pPINNs approach produces superior results than other methods in forecasting displacement and rotation, while also providing a significant decrease in the number of epochs required for convergence. This highlights the effectiveness of the preconditioning technique to improve the PINNs performance in order to solve the Timoshenko beam model, making it a functional tool for engineers in structural analysis.

3.6. PINN-based solution comparison for Timoshenko beam displacement

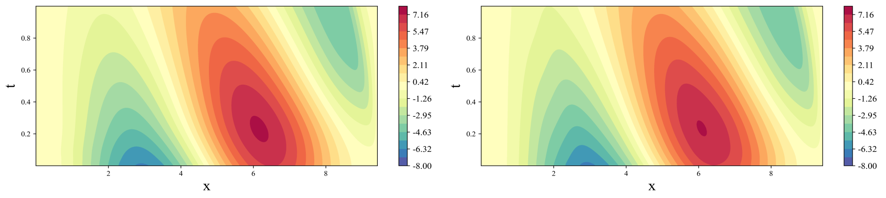
Figure 8 develops a comparison among different PINNs-based solutions for the displacement of the Timoshenko beam. In **Figure 8a**, the exact solution shown for the displacement of the Timoshenko beam, sets as the standard for comparison with the other methods. The exact solution give us a smooth and precise displacement plot across the beam with distinct patterns.



(a) Exact solution. **(b)** PINNs.

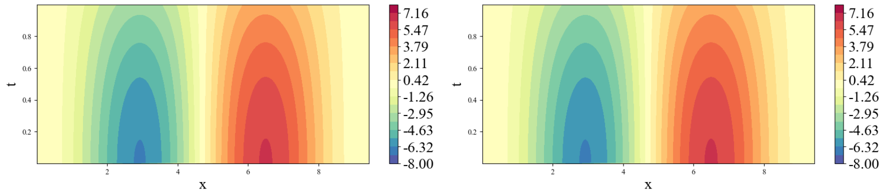


(c) SA-PINNs. **(d)** Gradient-enhanced PINNs (gPINNs).



(e) Adaptive PINNs. **(f)** Wavelet PINNs (Wav. PINNs).

Figure 8. *Cont.*



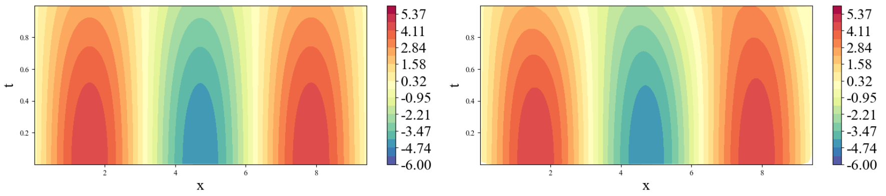
(h) Preconditioned PINNs (pPINNs). **(g)** Causal PINNs (cPINNs).

Figure 8. Visual comparison of predicted displacement fields for the Timoshenko beam at $t = 1$ using multiple PINN-based methods.

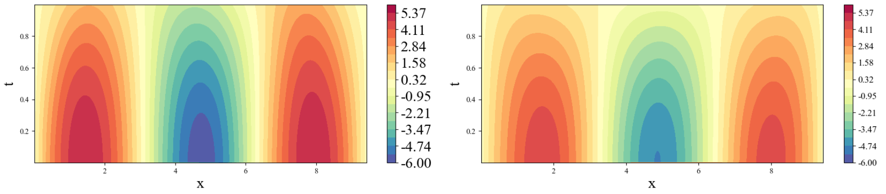
The **Figure 8b** shows the vanilla PINNs approach for the prediction of displacement. Although this solution captures the overall shape of the displacement, errors can be observed, particularly in those regions where sharp changes in displacement occur. The predictions we obtained from vanilla PINNs are less accurate than the exact solution, which shows that the standard PINNs method faces challenges in accurately capturing the beam's behavior. In **Figure 8c** we predicted the displacement using self-adaptive PINNs (SA-PINNs). The method shows a little improvement over vanilla PINNs, but the accuracy is still constrained, specially in those regions wherre gradients are sharp. Despite the improvement in the solution using SA-PINNs, there still observed significant deflections from the exact solution, particularly in the regions with higher displacement. **Figure 8d** shows the results from gradient-enhanced PINNs (gPINNs). The incorporation of gradient information helps to improve the solution, reducing errors in areas where sharp changes occur in the curves. Although, gPINNs show better performance than SA-PINNs and vanilla PINNs, the displacement plot still shows noticeable deviations in comparison to the exact solution, especially in areas with sophisticated variations. **Figure 8e** shows the displacement predicted using adaptive PINNs (Adap. PINNs). In this method, adaptive mechanism shows noticeable improvement in capturing the displacement profile more precisely. Despite of this, there are still some areas where the solution diverges from the exact result, particularly where the displacement has higher slope. In **Figure 8f**, the results obtained from the displacement predicted using wavelet PINNs (Wav. PINNs) are displayed. Wavelet PINNs improved the refinement of the displacement profile, reducing fluctuations and providing a clearer representation of the beam's displacement. While the results are better than the results obtained from SA-PINNs, gPINNs, and Adap. PINNs, Wav. PINNs still illustrate minor deviations compared to the exact solution. **Figure 8g** presents the results obtained from causal PINNs (cPINNs). The use of causal relationships in this method notably improves the displacement prognosis. The solution obtained from cPINNs is closely reaching the exact solution than those from the previous models, depicting that causal PINNs can effectively catch the displacement with a higher rate of accuracy. Finally, in **Figure 8h**, the results from preconditioned PINNs (pPINNs) are shown. This method use the preconditioned optimizer, which contributes in the most accurate displacement prediction. The solution from pPINNs is nearly closed to the exact solution, which shows the effectiveness of preconditioning in improving both the accuracy and efficiency of the PINNs model. The preconditioned method achieves the fastest convergence rate and more accurate displacement predictions as compared to all other methods, which illustrate the precedence of pPINNs in solving the Timoshenko beam problem. In conclusion, **Figure 8** clearly shows that the conventional PINNs-based methods like vanilla PINNs, SA-PINNs, gPINNs, Adap. PINNs, Wav. PINNs, and cPINNs provide useful solutions; they all still struggle with a lack of accuracy, as compared to the accuracy achieved by pPINNs. The pPINNs method, which contains a preconditioning strategy, yields highly accurate displacement predictions that closely reach the exact solution, which represents its superior performance in solving beam problems with high accuracy.

3.7. Comparison of PINN methods for Timoshenko beam rotation

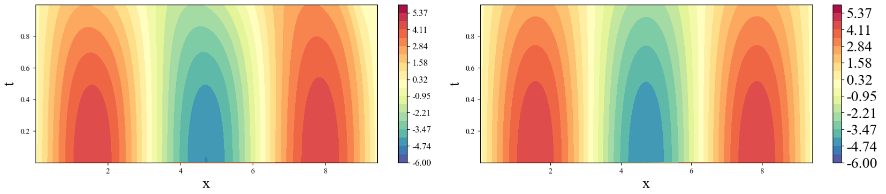
Figure 9 shows a comparison of different PINNs-based solutions for Timoshenko beam rotation at a given time. **Figure 9a** show the exact solution for the beam's rotation, which serves as a guide for evaluating the performance of various PINNs-based methods. The exact solution show the smooth and distinct variations in the rotation across the beam, with no significant divergence.



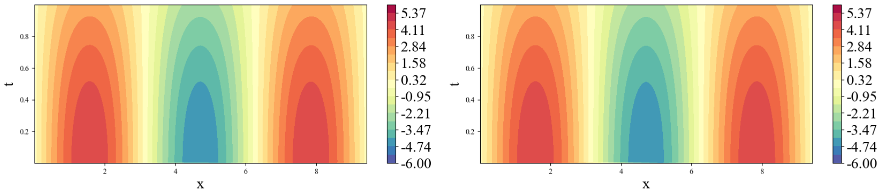
(a) Exact solution. **(b)** PINNs.



(c) SA-PINNs. **(d)** Gradient-enhanced PINNs (gPINNs).



(e) Adaptive PINNs. **(f)** Wavelet PINNs (Wav. PINNs).



(h) Preconditioned PINNs (pPINNs). **(g)** Causal PINNs (cPINNs).

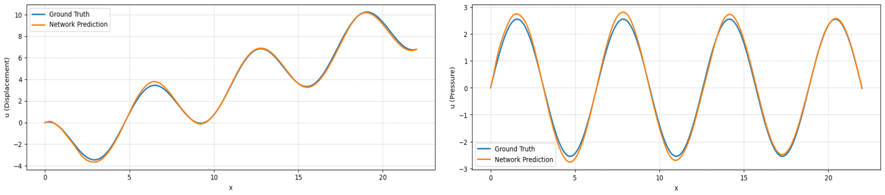
Figure 9. Visual comparison of predicted rotational fields for the Timoshenko beam at $t = 1$ using different PINN-based methods.

In **Figure 9b**, the results from vanilla PINNs (PINNs) are shwon. While this approach expresses the general tendency of the beam's rotation but fails to predict the displacement accurately in regions where sharp curves occur, which observed in noticeable errors when compared to the exact solution. The inaccuracies observed in the rotation plot show that standard PINNs struggle while handling such complex beam dynamics. **Figure 9c** shows the results from self-adaptive PINNs (SA-PINNs). The versatile features in this method ultimately improve the accuracy of the solution in comparison to vanilla PINNs, specially in regions with higher rotational variations. However, the prediction still shows notable differences when compared to the exact solution, which reflects the inability of SA-PINNs for not capturing the beam's behavior completely. In **Figure 9d**, the results of gradient-enhanced PINNs (gPINNs) are shown. The incorporation of gradient information helps in minimizing the error in the rotational profile, which provides a more accurate depiction of the beam's rotation. Nevertheless, the solution still shows observable discrepancies from the exact solution, particularly in regions with unpredicted rotation changes. The performance of gPINNs shows an improvement over vanilla and SA-PINNs but still fails to meet the desired accuracy. **Figure 9e** presents the results from adaptive PINNs (Adap. PINNs), which further refined the rotation prediction. The adaptive mechanism makes the model flexible to variations in the beam's rotation. Although the solution is more accurate as compared to the previous models, but show the still minor errors, especially in regions with high curves. In **Figure 9f**, wavelet PINNs (Wav. PINNs) are applied to predict the rotation. The use of wavelets in this approach helps to gather the small details of the rotation profile, which provides an improved solution. The wavelet method reduces deviations and provides a clearer representation of the beam's rotation, but small deflections from the exact solution still persist, specially in regions of abrupt rotational change. In **Figure 9g**, the results from causal PINNs (cPINNs) are displayed. This inclusion of causal relationships, considerably improve the model's ability to forecast the beam's rotation. The solution from cPINNs is closely reaching the exact solution compared to the other methods, with the smoother and accurate rotation profile, especially in the areas with sharp curves. Finally, in **Figure 9h**, the preconditioned PINNs (pPINNs) results are shown. The assimilation of preconditioning into the loss function shows accuracy in the solution for the beam's rotation. The rotation profile predicted by pPINNs is identical to the exact solution, which exhibites that the preconditioned approach highly improves both the accuracy and convergence rate of the model. The results from pPINNs show the high rate of precision in this comparison, which highlights its dominant performance over all other methods. In conclusion, **Figure 9** shows the relative performance of different PINNs approaches with respect to the rotation of the Timoshenko beam. While other methods such as vanilla PINNs, SA-PINNs, gPINNs, Adap. PINNs, Wav. PINNs, and cPINNs improve upon each other; the preconditioned PINNs (pPINNs) show the most accurate and efficient solution. The use of preconditioning in pPINNs remarkably accelerates the convergence rate and enhances the accuracy of the predicted rotation, which makes it the most superior method for solving the Timoshenko beam problem in this study.

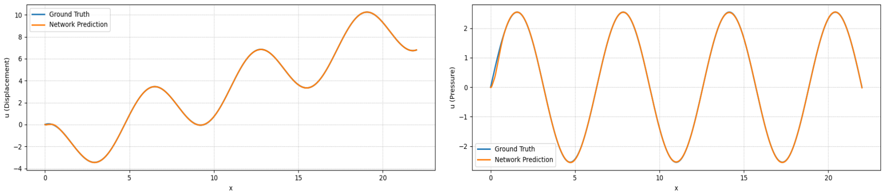
3.7.1. Extended spatial domain

Figure 10 presents the predicted solutions for the Timoshenko beam model at time $t = 1$. In **Figure 10a,b**, the results obtained using causal PINNs (cPINNs) are shown, while **Figure 10c,d** shows the results from preconditioned PINNs (pPINNs).

These solutions correspond to the beam's displacement and rotation, with both methods compared to the exact solution to analyze their performance.



(a) Causal PINNs (cPINNs) for ϕ. **(b)** Causal PINNs (cPINNs) for u.



(c) Preconditioned PINNs (pPINNs) for ϕ. **(d)** Preconditioned PINNs (pPINNs) for u.

Figure 10. Predicted solutions of the Euler–Bernoulli beam at $t = 1$ obtained using causal PINNs (cPINNs) and preconditioned PINNs (pPINNs).

In **Figure 10a,b**, the displacement and rotation predicted by cPINNs are shown. The solution's approximation to the beam's behavior, some errors are prominent, especially in regions with sharp gradients. The results indicate that cPINNs can record the general behavior of the beam but do not completely reach the exact solution, especially where the beam undergoes abrupt changes. These disparities highlight the short comings of cPINNs when handling more complex beam dynamics, especially in the case of high shear forces or large aspect ratios.

In **Figure 10c,d**, the displacement and rotation predicted by pPINNs are shown. The use of preconditioning in the loss function improves the model's accuracy significantly. The pPINNs solution aligns much more with the exact solution, which shows the less error, specifically in the regions with sharp displacement and rotation variations. This highlights the efficiency of the preconditioning approach in enhancing the accuracy and productivity of the model, also pPINNs improve convergence by reducing error, which make it a valid method for solution of beam problems, particularly for complex structural frameworks.

Table 3 outlines the comparison of the relative errors $\mathcal{R}$ for both cPINNs and pPINNs within various spatial domains. The table represents the displacement (u^*) and rotation (θ^*) values for both methods at three different spatial domain: $[0, 4\pi]$, $[0, 5\pi]$, and $[0, 7\pi]$. For cPINNs, the displacement and rotation predictions vary, with u^* values decreasing as the spatial domain increases, while θ^* shows an elevation. On the other hand, the pPINNs method produces more accurate results, with significantly smaller errors in both displacement and rotation. For example, at the domain $[0, 4\pi]$, cPINNs produce a displacement of 0.52 and a rotation of 5.07, while pPINNs give a much smaller error with $u^* = 0.06$ and $\theta^* = 2.38$. These results clearly shows the efficiency of pPINNs over cPINNs, especially with respect to accuracy.

Table 3. Relative errors ($\mathcal{R}$) for displacement (u^*) and rotation (θ^*) of the Timoshenko beam at 500 training epochs, evaluated over different spatial domains. Comparison between causal PINNs (cPINNs) and preconditioned PINNs (pPINNs).

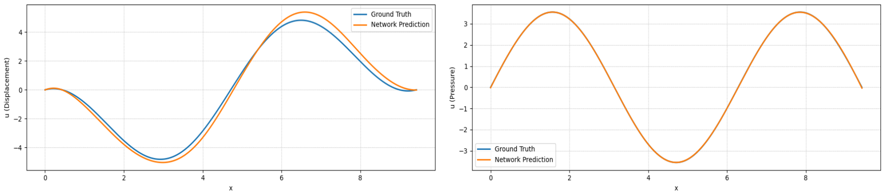
x	**cPINNs**		**pPINNs**	
	u^*	θ^*	u^*	θ^*
$[0, 4\pi]$	0.42	5.17	0.09	2.48
$[0, 5\pi]$	0.34	5.28	0.054	2.02
$[0, 7\pi]$	0.25	6.37	0.075	2.56

The table also show that the performance of both cPINNs and pPINNs improves with the increase of spatial domain. However, the difference in accuracy between the two methods becomes more recognizable as the domain expands, with pPINNs consistently yielding more accurate predictions across all domains.

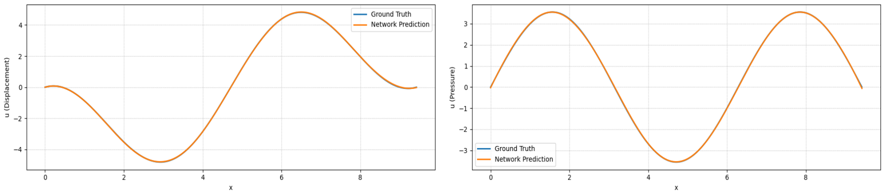
In conclusion, **Figure 10** and **Table 3** show the effectiveness of preconditioned PINNs (pPINNs) while solution of the Timoshenko beam problem. The preconditioning method significantly improves the accuracy of the displacement and rotation predictions, specially in complex situations such as large spatial domains with high aspect ratios or shear forces.

3.7.2. Temporal domain extension

Figure 11 show the predicted solutions for the Timoshenko beam at time $t = 7$. The solutions are obtained using two different PINNs-based methods which includes the causal PINNs (cPINNs) in **Figure 11a,b**, and preconditioned PINNs (pPINNs) in **Figure 11c,d**. The displacement and rotation plots at $t = 7$ are displayed for both methods, that shows the performance of each method in the approximation of the exact solution.



(a) Causal PINNs (cPINNs) for ϕ. **(b)** Causal PINNs (cPINNs) for u.



(c) Preconditioned PINNs (pPINNs) for ϕ. **(d)** Preconditioned PINNs (pPINNs) for u.

Figure 11. Predicted outcomes of the Timoshenko beam at $t = 7$ using causal PINNs (cPINNs) and preconditioned PINNs (pPINNs).

In **Figure 11a,b**, the results from cPINNs are shown. The displacement and rotation predictions which is provided by cPINNs show noticeable alignment with the exact solution, as highlighted by the close match between the predicted and rectified data curves. However, minor discrepancies can still be observed, especially in the regions with more unpredictable changes in the beam's displacement and rotation.

These small errors show the problems that are faced by cPINNs in accurately capturing the behavior of the Timoshenko beam in the temporal domain extension.

Figure 11c,d presents the results from pPINNs. The inclusion of the preconditioned PINNs method considerably improves the accuracy of the displacement and rotation predictions. The curves in these plots match the exact solution more closely with less deviation in the regions with abrupt changes in displacement and rotation. The preconditioning technique blended into the pPINNs model accelerates the convergence and enhances the accuracy of the predictions, as observed in the much smaller error compared to cPINNs.

Table 4 summarizes the comparison of the relative errors $\mathcal{R}$ for both cPINNs and pPINNs over different temporal domains, with 500 epochs used for training. The table shows the displacement (u^*) and rotation (θ^*) values for both methods at three different temporal intervals: $[0, 4]$, $[0, 6]$, and $[0, 7]$. The values of u^* and θ^* for cPINNs are prominently higher compared to those for pPINNs. For example, at $t = 4$, cPINNs produce $u^* = 8.61$ and $\theta^* = 6.04$, whereas pPINNs achieve much smaller values of $u^* = 3.59$ and $\theta^* = 2.84$. This trend is continuous across the other time intervals as well, with pPINNs constantly providing more accurate predictions for both displacement and rotation.

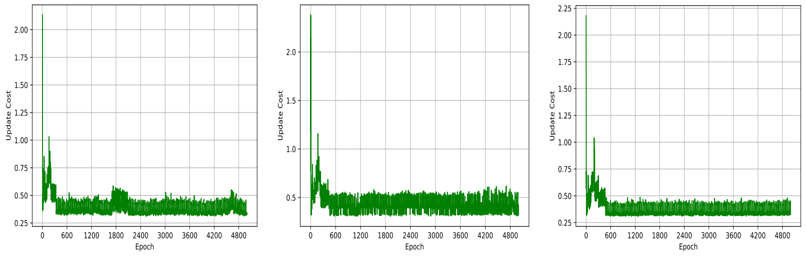
Table 4. Relative errors ($\mathcal{R}$) for displacement (u^*) and rotation (θ^*) of the Timoshenko beam over various temporal domains at 500 training epochs. Comparison between causal PINNs (cPINNs) and preconditioned PINNs (pPINNs).

x	**cPINNs**		**pPINNs**	
	$\boldsymbol{u^*}$	$\boldsymbol{\theta^*}$	$\boldsymbol{u^*}$	$\boldsymbol{\theta^*}$
$[0, 4]$	7.64	4.06	5.49	1.93
$[0, 6]$	6.03	6.09	6.67	4.47
$[0, 7]$	5.42	2.76	4.75	1.15

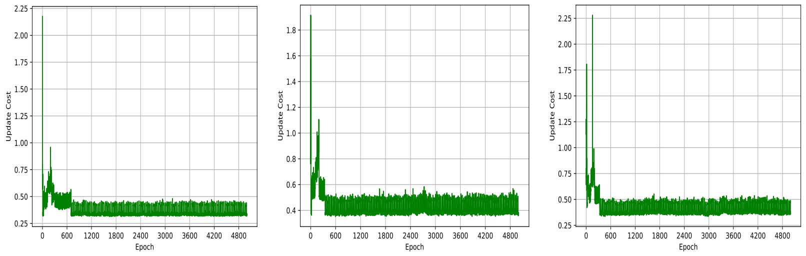
The table also illustrates the improvement in accuracy in the extension of the temporal domain. While the error in cPINNs remains relatively large, the pPINNs method shows smaller errors in both displacement and rotation, demonstrating the effectiveness of the preconditioning approach in improving the model's long-lasting performance.

In conclusion, **Figure 11** and **Table 4** provide a noticeable comparison between cPINNs and pPINNs for the Timoshenko beam problem over an expansion temporal domain. The results clearly show that pPINNs surpass cPINNs with respect to both accuracy and efficiency, as indicated by the minor errors in the displacement and rotation predictions and the reduced number of epochs required for convergence. The preconditioning strategy incorporated in pPINNs shows its effectiveness for solving beam problems over extended temporal domains.

In **Figure 12**, the update cost related to different PINNs solutions is presented for the Timoshenko beam. **Figure 12a** presents the exact solution, providing a standard for the error reduction over time. **Figure 12b** illustrates the cost function for standard PINNs, which shows a typical inconsistent behavior during the initial training stages, showing fluctuations in the error before gradually getting more stable. In **Figure 12c**, SA-PINNs demonstrate a slightly higher convergence rate compared to standard PINNs, reducing deviations in the error over time. The gradient-enhanced PINNs (**Figure 12d**) show further improved stability, with a smoother and more uniform decline in the update cost as the solution improves. The adaptive PINNs (**Figure 12e**) show even greater accuracy with fewer fluctuations, leading to faster convergence rates. Wavelet PINNs (**Figure 12f**) show slower convergence due to more fluctuations compared to other methods.



(a) PINNs. **(b)** SA-PINNs. **(c)** Gradient-Enhanced PINNs (gPINNs).

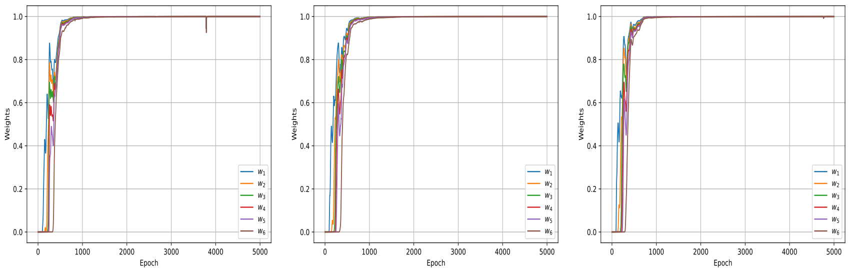


(d) Adaptive PINNs. **(e)** Wavelet PINNs (Wav. PINNs). **(f)** Preconditioned PINNs (pPINNs).

Figure 12. Update cost comparison of different PINN-based solutions for the Timoshenko beam at 500 training epochs.

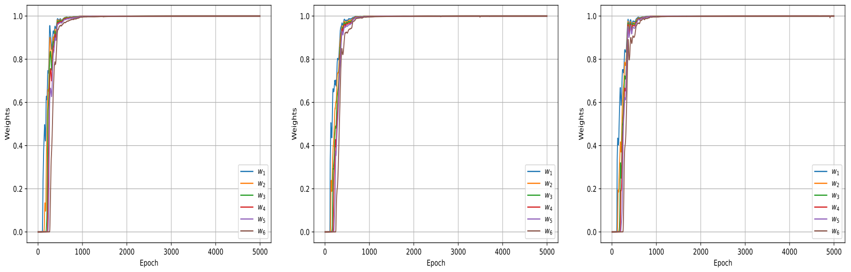
Note: Each subplot represents a different PINN-based approach.

Figure 13 presented the progress of the weights in the network during training for various PINNs methods applied to the Timoshenko beam. In **Figure 13a**, the exact solution shows the ideal weight distribution, with smooth transformations between training iterations. On the other hand, **Figure 13b** shows the weights for standard PINNs, which show more inconsistencies and deviations compared to the exact solution. As the training progresses, the weights in SA-PINNs (**Figure 13c**) achieve stability, and steeper PINNs (**Figure 13d**) show an evolution toward the optimal weights. Adaptive PINNs (**Figure 13e**) demonstrate the most reliable weight updates, leading to smoother convergence in the solution. Wavelet PINNs (**Figure 13f**) show slower and more unstable weight updates, reflecting the inefficiency of wavelet-based approaches for this specific problem.



(a) Exact solution. **(b)** Standard PINNs. **(c)** SA-PINNs.

Figure 13. *Cont.*



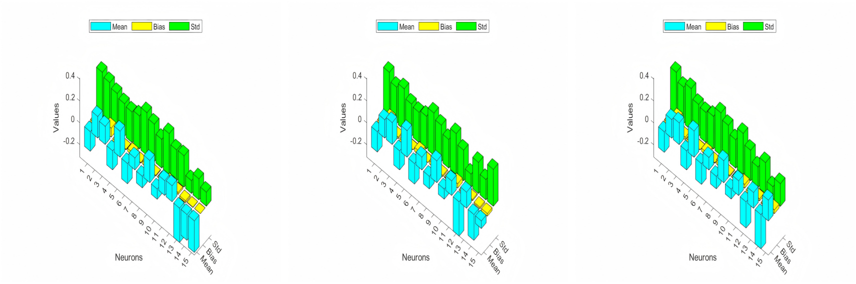
(d) Gradient-Enhanced PINNs (gPINNs). **(e)** Adaptive PINNs. **(f)** Wavelet PINNs (Wav. PINNs).

Figure 13. Learned weight distributions for various PINN-based solutions of the Timoshenko beam.

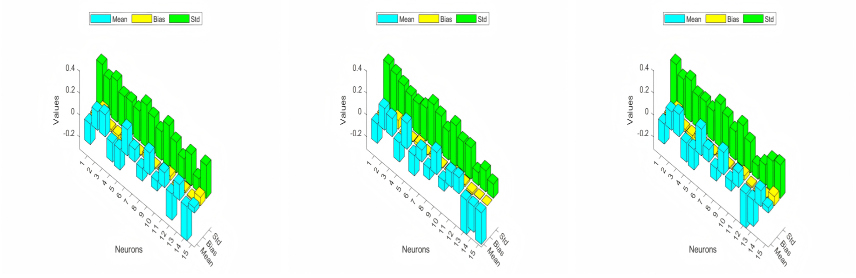
Note: Each subplot corresponds to one specific PINN method.

3.8. Analysis of best weights in PINNs for Timoshenko beam

Figure 14 introduces the best weights that grant most to the solution of different PINNs techniques for the Timoshenko beam. In this figure, each panel shows the contribution of the weights at different steps of training for various PINNs methods.



(a) Exact solution. **(b)** Standard PINNs. **(c)** SA-PINNs.



(d) Gradient-Enhanced PINNs (gPINNs). **(e)** Adaptive PINNs. **(f)** Wavelet PINNs (Wav. PINNs).

Figure 14. Most powerful learned weights contributing to the Timoshenko beam solution for different PINN-based methods.

Note: Each subplot corresponds to one specific method.

Figure 14a displays the exact solution, showing the weights in the beginning, contributing to the solution. This sets a standard for analyzing the performance of the other PINNs methods. **Figure 14b** shows that the results from the standard PINNs method demonstrate the weights' distribution, reflecting a relatively slower convergence towards the optimal weights compared to the exact solution. The distribution of weights is abrupt and shows more fluctuations during training, depicting the method's difficulty in achieving the best solution within a short time period time.

In **Figure 14c**, the SA-PINNs approach shows an improved weight distribution. Despite of the fast convergence than standard PINNs, consistency of the weight contributions could be improved. **Figure 14d** illustrates the results for Gradient-Enhanced PINNs, which show a more cincentrated weight distribution. It has observed that the weights converge more efficiently toward their optimal values, demonstrating the ability of method to enhance rate of convergence over the standard PINNs approach.

Figure 14e shows the results for Adaptive PINNs. This panel shows that the weights converge rapidly with a more stable distribution, reflecting the effectiveness of adaptive mechanism to guide the network toward an accurate solution with minimal deviations in weight values. In **Figure 14f**, the Wav. PINNs method represents a slightly slower convergence of weights compared to the Adaptive PINNs. Despite this slow convergence, a noticeable improvement over the standard PINNs approach has observed.

These results reinforce the effectiveness of advanced PINNs approaches, such as SA-PINNs, Gradient-Enhanced PINNs, Adaptive PINNs, cPINNs, and pPINNs, with the ability of achieving faster convergence and more accurate solutions compared to the standard PINNs method. The incorporation of preconditioning in pPINNs represents a noticeable improvement in weight convergence and overall solution accuracy, leading to enhanced stability during training and superior approximation of the underlying physical solution.

4. Conclusion

This study presented a PDFP-enhanced Physics-Informed Neural Network framework for solving higher-order PDEs arising in Euler–Bernoulli and Timoshenko beam models. The results demonstrate that the proposed approach significantly improves convergence behavior and predictive accuracy compared to existing PINN variants. In particular, the method achieves lower relative error with substantially fewer training epochs, confirming its effectiveness for stiff and high-order systems.

Despite these advantages, several limitations remain. First, the performance of the proposed model is sensitive to network architecture and hyperparameter selection, which may require careful tuning for different problems. Second, the current implementation is restricted to one-dimensional spatial domains and relatively simple geometries, limiting its direct applicability to complex engineering structures. Third, although the PDFP optimizer enhances convergence, it increases computational overhead per iteration due to curvature approximation and matrix operations. Additionally, the scalability of the method to large-scale three-dimensional problems and real-time applications has not been fully explored.

Future research can address these limitations in several directions. Extending the framework to multi-dimensional and complex geometries, including irregular domains and coupled multiphysics systems, is an important next step. The integration of adaptive sampling strategies and automated hyperparameter tuning can further improve robustness and reduce user intervention. Moreover, developing lightweight or hybrid optimization strategies may reduce computational cost while maintaining fast convergence. Finally, applying the proposed framework to real-world engineering problems, such as nonlinear structural dynamics and fluid–structure interaction systems, will help validate its practical effectiveness.

Author contributions: All authors contributed equally to the conception, design, data collection, analysis, and writing of this study. All authors have read and agreed to the published version of the manuscript.

Funding: This work received no external funding.

Institutional review board statement: Not applicable.

Informed consent statement: Not applicable.

Data availability statement: No new data were generated or analyzed in this study.

Acknowledgement: All authors are extremely grateful to their parents for their continuous support and encouragement, which contributed to the successful completion of this research. In particular, the author Muhammad Israr dedicates this study to his father, Muhammad Sadiq, and his family members for their unwavering support and motivation throughout this work. He also expresses sincere gratitude to his respected teachers, Dr. Ateeq Ur Rehman and Dr. Majid Khan, for their valuable guidance, encouragement, and academic support. The authors further extend heartfelt gratitude to the Department of Mathematics, Government Post Graduate College Mansehra, for providing continuous support and serving as a strong source of academic strength.

Conflict of interest: There is no conflict of interest declared by the authors.

Abbreviation

Symbol	**Description**
$\phi(x,t)$	Cross-sectional rotation (Timoshenko beam)
$u(x,t)$	Vertical displacement of the beam
$p(x,t)$	Foundation response, given by $p(x,t) = ku(x,t)$
$f(x,t)$	External loading function
k	Stiffness parameter of the foundation
$L(u)$	Total loss function for Euler–Bernoulli beam
$L_{int}(u)$	Interior loss term in the PINN framework
$L_{bc}(u)$	Boundary condition loss term in the PINN framework
$L_{ic}(u)$	Initial condition loss term in the PINN framework
R	Relative error in the beam displacement prediction
w_{int}, w_{res}	Weighting coefficients for interior and residual loss
θ	Neural network model parameters (weights and biases)
α^*	Optimal model parameters after optimization
$\epsilon(\phi_{NN}, \phi^*)$	Approximation error of the neural network model
cPINNs	Causal Physics-Informed Neural Networks
pPINNs	Preconditioned Physics-Informed Neural Networks
SA-PINNs	Self-Adaptive Physics-Informed Neural Networks
gPINNs	Gradient-Enhanced Physics-Informed Neural Networks
Adap. PINNs	Adaptive Physics-Informed Neural Networks
Wav. PINNs	Wavelet Physics-Informed Neural Networks

PDFP	Preconditioned Davidon–Fletcher–Powell optimizer
LBFGS	Limited-memory Broyden–Fletcher–Goldfarb–Shanno optimizer
Adam	Adaptive Moment Estimation optimizer

References

1. Deng J, Shahroudi M, Liu K. Dynamic stability and responses of beams on elastic foundations under a parametric load. International Journal of Structural Stability and Dynamics. 2023; 23(2): 2350018.
2. Younesian D, Hosseinkhani A, Askari H, et al. Elastic and viscoelastic foundations: a review on linear and nonlinear vibration modeling and applications. Nonlinear Dynamics. 2019; 97(1): 853–895.
3. Saini R, Pradyumna S. Asymmetric vibrations of functionally graded annular nanoplates under thermal environment using nonlocal elasticity theory with modified nonlocal boundary conditions. Journal of Engineering Mechanics. 2023; 149(5): 04023022.
4. Israr M, Khan I. Causal Physics-Informed Neural Networks for Singular and Singularly Perturbed Boundary Value Problems in Chemical Systems. Global Integrated Mathematics. 2025; 1(1): 13–27.
5. Kabir H, Aghdam M. A robust Bézier based solution for nonlinear vibration and post-buckling of random checkerboard graphene nano-platelets reinforced composite beams. Composite Structures. 2019; 212: 184–198.
6. Yu J, Lu L, Meng X, et al. Gradient-enhanced physics-informed neural networks for forward and inverse PDE problems. Computer Methods in Applied Mechanics and Engineering. 2022; 393: 114823.
7. Saini R, Gopalakrishnan S. Nonlocal boundaries and paradoxes in thermoelastic vibrations of functionally graded Non-Uniform cantilever nanobeams and annular nanoplates. Structures. 2023; 55: 1292–1305.
8. Madenci E, Guven I. The Finite Element Method and Applications in Engineering Using ANSYS. Springer; 2015.
9. Saini R. Thermoelastic vibrations of functionally graded nonuniform nanobeams. In: Nanomaterials for Advanced Technologies. Springer; 2022. pp. 141–171.
10. Saini R, Saini R, Kumar A, et al. Free axisymmetric vibrations of heated non-uniform Bi-directional FGM Mindlin rings employing quadrature approaches. Thin-Walled Structures. 2023; 184: 110482.
11. Saini R, Pradyumna S. Effect of thermal environment on the asymmetric vibration of temperature-dependent two-dimensional functionally graded annular plate by Chebyshev polynomials. Journal of Thermal Stresses. 2022; 45(9): 740–761.
12. Meng D, Yang H, Yang S, et al. Kriging-assisted hybrid reliability design and optimization of offshore wind turbine support structure based on a portfolio allocation strategy. Ocean Engineering. 2024; 295: 116842.
13. Meng D, Yang S, De Jesus AM, et al. A novel Kriging-model-assisted reliability-based multidisciplinary design optimization strategy and its application in the offshore wind turbine tower. Renewable Energy. 2023; 203: 407–420.
14. Karniadakis GE, Kevrekidis IG, Lu L, et al. Physics-informed machine learning. Nature Reviews Physics. 2021; 3(6): 422–440.
15. Israr M, Ayub A, Alkhalaf A, et al. AI-assisted PDFP-based PINNs for heat regulation of nanofluid in industrial cooling enclosures around circular barriers. International Communications in Heat and Mass Transfer. 2026; 173: 110717.
16. E W, Yu B. The deep Ritz method: A deep learning-based numerical algorithm for solving variational problems. Communications in Mathematics and Statistics. 2018; 6(1): 1–12.
17. Sirignano J, Spiliopoulos K. DGM: A deep learning algorithm for solving partial differential equations. Journal of Computational Physics. 2018; 375: 1339–1364.
18. Raissi M, Perdikaris P, Karniadakis GE. Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations. Journal of Computational Physics. 2019; 378: 686–707.
19. Saini R. A review on artificial neural networks for structural analysis. Journal of Vibration Engineering & Technologies. 2025; 13(2): 142.
20. Wang S, Teng Y, Perdikaris P. Understanding and mitigating gradient flow pathologies in physics-informed neural networks. SIAM Journal on Scientific Computing. 2021; 43(5): A3055–A3081.
21. Saini R, Saini R. Thermal buckling analysis of nonuniform FG nanobeams: analytical and numerical solutions.

Journal of Mechanics of Materials and Structures. 2024; 19(5): 837–855.

22. Gou J, Sun L, Liu J, et al. An adaptive differential evolution with deeply informed mutation strategy and historical information for numerical optimization. Information Sciences. 2026; 739: 123146.
23. Yang Q, Liu Z, Wang Xiaokai, et al. Physical model test and numerical modeling of cross-sectional shape effect on evolution mechanism of time-delayed deformation and rockburst in deep tunnels. Rock Mechanics and Rock Engineering. 2025; 59(2): 2015–2043.
24. Su H, Liao Q, Zhang H, et al. Advanced Intelligent Pipeline Management Technology. Springer; 2023.
25. Kingma DP. Adam: A method for stochastic optimization. arXiv preprint. 2014; arXiv:1412.6980.
26. Bottou L. Stochastic gradient descent tricks. In: Neural Networks: Tricks of the Trade, 2nd ed. Springer; 2012. pp. 421–436.
27. Jagtap AD, Kawaguchi K, Karniadakis GE. Adaptive activation functions accelerate convergence in deep and physics-informed neural networks. Journal of Computational Physics. 2020; 404: 109136.
28. Wang S, Yu X, Perdikaris P. When and why PINNs fail to train: A neural tangent kernel perspective. Journal of Computational Physics. 2022; 449: 110768.
29. Dolean V, Heinlein A, Mishra S, et al. Multilevel domain decomposition-based architectures for physics-informed neural networks. Computer Methods in Applied Mechanics and Engineering. 2024; 429: 117116.
30. Botev A, Ritter H, Barber D. Practical Gauss-Newton optimisation for deep learning. In: Proceedings of the 34th International Conference on Machine Learning; 6–11 August 2017; Sydney, Australia. pp. 557–565.
31. Li W, Xiang X, Xu Y. Deep domain decomposition method: Elliptic problems. In: Proceedings of the First Mathematical and Scientific Machine Learning Conference; 20–24 July 2020; Princeton, NJ, USA. pp. 269–286.
32. Bazmara M, Silani M, Mianroodi M, et al. Physics-informed neural networks for nonlinear bending of 3D functionally graded beam. Structures. 2023; 49: 152–162.
33. Kapoor T, Wang H, Núñez A, et al. Physics-informed neural networks for solving forward and inverse problems in complex beam systems. IEEE Transactions on Neural Networks and Learning Systems. 2023; 35(5): 5981–5995.
34. Saini R, Pradyumna S. A layer-wise theory for the dynamic analysis of functionally graded composite nanobeams under thermal environment using nonlocal boundary conditions and Chebyshev collocation technique. Journal of Vibration and Control. 2024; 30(19–20): 4591–4611.
35. Xu C, Cao BT, Yuan Y, et al. Transfer learning based physics-informed neural networks for solving inverse problems in engineering structures under different loading scenarios. Computer Methods in Applied Mechanics and Engineering. 2023; 405: 115852.
36. Saini R. An Investigation for Asymmetric Vibrations of Multi-dimensional Functionally Graded Heated Non-Uniform Annular Nanoplates using Chebyshev Polynomials and Parallel Computing. Journal of Vibration Engineering & Technologies. 2025; 13(5): 289.
37. Sukumar N, Srivastava A. Exact imposition of boundary conditions with distance functions in physics-informed deep neural networks. Computer Methods in Applied Mechanics and Engineering. 2022; 389: 114333.
38. Mishra S, Molinaro R. Estimates on the generalization error of physics-informed neural networks for approximating PDEs. IMA Journal of Numerical Analysis. 2023; 43(1): 1–43.
39. Ahmad S, Israr M. A preconditioned quasi-newton optimizer for efficient training of PINNs. Machine Learning for Computational Science and Engineering. 2025; 1(2): 34.
40. McClenny LD, Braga-Neto UM. Self-adaptive physics-informed neural networks. Journal of Computational Physics. 2023; 474: 111722.
41. Uddin Z, Ganga S, Asthana R, et al. Wavelets based physics informed neural networks to solve non-linear differential equations. Scientific Reports. 2023; 13(1): 2882.
42. Kapoor T, Wang H, Nunez A, et al. Transfer learning for improved generalizability in causal physics-informed neural networks for beam simulations. Engineering Applications of Artificial Intelligence. 2024; 133: 108085.