

Physics-Informed Neural Networks for Solving Forward and Inverse Problems in Complex Beam Systems

Taniya Kapoor¹, Hongrui Wang¹, *Member, IEEE*, Alfredo Núñez², *Senior Member, IEEE*, and Rolf Dollevoet¹

Abstract—This article proposes a new framework using physics-informed neural networks (PINNs) to simulate complex structural systems that consist of single and double beams based on Euler–Bernoulli and Timoshenko theories, where the double beams are connected with a Winkler foundation. In particular, forward and inverse problems for the Euler–Bernoulli and Timoshenko partial differential equations (PDEs) are solved using nondimensional equations with the physics-informed loss function. Higher order complex beam PDEs are efficiently solved for forward problems to compute the transverse displacements and cross-sectional rotations with less than $1e - 3\%$ error. Furthermore, inverse problems are robustly solved to determine the unknown dimensionless model parameters and applied force in the entire space–time domain, even in the case of noisy data. The results suggest that PINNs are a promising strategy for solving problems in engineering structures and machines involving beam systems.

Index Terms—Complex system, double-beam system, Euler–Bernoulli beam, physics-informed neural networks (PINNs), Timoshenko beam.

I. INTRODUCTION

COMPLEX engineering issues in real-life scenarios are often characterized by the connection between various subsystems and uncertainty in behavior caused by internal and external variables and their interactions. Furthermore, the design and maintenance of complex systems, such as engineering structures and machines, are made challenging by the unpredictable collective behaviors and properties of these concurrently operating and interacting components. These issues are typically difficult to analyze through conventional methods [1]. Most of these complex engineering systems are continuous, and partial differential equation (PDE) models are used to characterize and understand their behavior. These PDE models are used to simulate a wide range of engineering phenomena, ranging from multiple beam systems in suspension bridge cables (Timoshenko beam equations) [2] to

catenary–pantograph interactions in railways (damped beam equations) [3] to simulating air turbulence that disrupts flight (Navier–Stokes equations) [4], [5], among many others [6], [7], [8], [9], [10], [11], [12], [13]. Solutions to governing PDEs enable real challenges such as structural health monitoring [14], [15], [16] and optimal structural design [17], [18] to be addressed.

The development of algorithms for diagnostics and prognosis is an issue in maintaining complex engineering systems [1]. Insights could be obtained by solving the forward and inverse problems for the governing PDEs of interest to forecast the system’s behavior and minimize unexpected downtimes of complex systems. These equations range in complexity from being extremely nonlinear (Navier–Stokes equation [19]) to incorporating intricate higher order boundary conditions (fourth-order beam equations [20]). In practice, these equations are too complicated to be solved analytically and must be solved numerically. Numerical methods such as the finite-difference and finite-element methods have been used to approximate the solutions of these PDEs. Despite their success in practice, these methods encounter some difficulties, such as mesh creation, which is more difficult for complex geometries in higher dimensions [21], [22].

In recent years, scientific machine learning, which combines scientific computing with machine learning methodologies to estimate PDEs solutions, has made remarkable developments and has emerged as a viable alternative to the aforementioned numerical methods. The review papers [21], [23], [24] extensively discuss the state-of-the-art breakthroughs in scientific machine learning, including works on real-world engineering problems. However, data-driven methods require a large amount of data, which are possibly computationally expensive and susceptible to noise in some engineering systems [25]. One possible way to mitigate the effects of these problems is to use the known physical knowledge of the underlying system in the learning procedure [26], [27], [28]. Prior physical knowledge could be incorporated into the learning procedure by collocating the PDE residual at training points, similar to leveraging the physical equation in the training process. The underlying neural networks proposed in [25] are called physics-informed neural networks (PINNs).

PINNs use neural networks’ universal function approximation property [29] and embed the well-posed physical equations modeled by PDEs in the loss function. Prior knowledge of physical principles works as a regularization agent

Manuscript received 24 February 2023; revised 19 May 2023 and 21 July 2023; accepted 23 August 2023. Date of publication 19 September 2023; date of current version 3 May 2024. This work was supported in part by ProRail and Europe’s Rail Flagship Project under Grant IAM4RAIL and in part by the Holistic and Integrated Asset Management for Europe’s RAIL System under Grant 101101966. (Corresponding author: Hongrui Wang.)

The authors are with the Section of Railway Engineering, Department of Engineering Structures, Delft University of Technology, 2628 CN Delft, The Netherlands (e-mail: t.kapoor@tudelft.nl; h.wang-8@tudelft.nl; a.a.nunezvicencio@tudelft.nl; r.p.b.j.dollevoet@tudelft.nl).

Digital Object Identifier 10.1109/TNNLS.2023.3310585

in neural network training, restricting the space of admissible solutions and improving the function approximation accuracy. As a result, given some knowledge of the physical features of the problem and some training data, PINN can be used to identify a high-fidelity solution. PINNs have already proven to be a very effective paradigm for approximating solutions of PDEs for real-world problems [30], [31], as discussed in the review papers [21], [23].

However, several challenges for PINNs have also been found [32]. One such challenge for PINNs is to learn relevant physical phenomena for more complex problems with large coefficients in the physical equation [33]. A sequence-to-sequence learning task was proposed in [33] as a remedy to this problem. However, this can be computationally expensive when the scale is large. In [34], the importance of using nondimensional equations in the PINN framework was highlighted for cardiovascular blood flow. We build on these works and address the challenge of multiscale complex beam systems. Accordingly, this article uses nondimensional PDEs instead of dimensional PDEs in the loss function. This provides a way to simulate realistic physical equations with computational tractability.

Accurate prediction of the dynamics of structures [35] and structural elements, such as plates [36], and beams [37], [38], is crucial in the field of structural engineering. However, measuring quantities of interest in beam systems through laboratory experiments can prove to be difficult, as it necessitates specialized prototypes, training, and safety during the testing process, increasing the overall cost of the experiment. PINNs offer a simulation-based solution as a mesh-free method that does not require discretizing the domain into a finite number of elements, making it computationally inexpensive compared with numerical methods. PINNs can effectively integrate incomplete or noisy information with prior physical knowledge. The proposed framework converts dimensionalized PDEs into a nondimensionalized form, increasing the suitability for neural networks and enabling the prediction of deflections and rotations for any material, resulting in a more generalizable method.

This article provides a framework to simulate complex structural systems consisting of two or more basic structural systems connected by an elastic layer. In particular, the forced vibration of two elastically connected beams is studied, which is commonly encountered in the mechanical, construction, and aeronautical industries [6]. These double-beam systems in engineering structures have received significant attention in the scientific community and are considered complex systems. Studies have been conducted to predict the dynamics of these systems under various loading and force conditions, such as those found in papers [39], [40], [41], [42], [43], [44], [45], [46], [47], among others. These studies include the use of analytical and closed-form solutions [43], [48], [49], [50], [51]; however, analytical methods have limitations in applicability, as they may be useful only for specific types of problems and can become complex for systems with many variables or nonlinear equations. Other approaches, such as the state-space method presented in [45] and [52], may also be computationally expensive for systems with a large number of states. In addition, modal analysis methods as presented in [6]

and [53] have been used to study the natural frequencies and modes of vibration, but they do not provide information on the full response of the system and cannot be used to predict the time-domain response at any instant.

The considered governing equations are modeled using Euler–Bernoulli and Timoshenko theories. In addition to solving the forward problem and computing the physical quantities of interest, we also solve the inverse problem. For the inverse problem, one may not necessarily have complete information about the inputs to the PDEs, such as initial or boundary data, coefficients [54], [55], [56] or applied forces. This lack of knowledge makes the forward problem ill-posed, and subsequently, the forward problem cannot be solved uniquely. In this article, access to data for quantities of interest is leveraged to determine the PDEs’ unknown inputs, for instance, the model parameters and applied forces.

The main contributions of the current article are as follows.

- 1) To the best of the authors’ knowledge, this is the first work to use physics-informed machine learning to solve the forward and inverse problems of Euler–Bernoulli and Timoshenko complex beam models.
- 2) We address a challenge for PINNs in solving multiscale complex beam PDEs and propose a framework for using nondimensional equations in the loss function.
- 3) The proposed nondimensional PINN framework is used to address ill-posed inverse problems for complex systems and to identify the unknown model parameters and the applied force on the beam components. This is achieved using data from indirect measurements such as the displacement and cross-sectional rotations of the beams.
- 4) The presented methodology is robust to noise and can accommodate potential uncertainty in the measurement data, making it well-suited for real-world applications where data are incomplete or uncertain.

The rest of this article is organized as follows. In Section II, the PINN method is presented to simulate the dimensional Euler–Bernoulli beam equation. Due to the limitations of PINNs in simulating the dimensional Euler–Bernoulli beam equation, an alternative approach of using nondimensional equations in the PINN’s loss function is proposed and successfully used to solve the dimensionless Euler–Bernoulli equation in Section III. Section IV first applies the proposed framework to simulate the Timoshenko beam model for solving forward and inverse problems. The forward problem of the Euler–Bernoulli double-beam equation is then solved. In addition, Section IV covers forward and inverse Timoshenko double-beam system problems. Section V concludes this article.¹

II. PINNs FOR DIMENSIONAL PDES

In this section, the method of PINNs to simulate PDEs is presented in brief using an abstract dimensional PDE. The method is then used to simulate the dimensional Euler–Bernoulli equation. The following abstract dimensional PDE is considered with implicit initial and boundary

¹Collection of code and data intended for replicating the experiments presented in this paper can be found here: https://github.com/taniyakapoor/PINNs_beam/tree/master

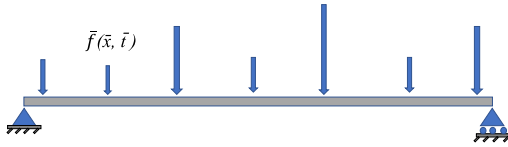


Fig. 1. Simply supported beam with varying transverse force.

conditions:

$$\begin{aligned} \bar{K}(\bar{x}, \bar{t}) &:= \mathcal{D}[\bar{u}](\bar{x}, \bar{t}; \bar{\lambda}) - \bar{f}(\bar{x}, \bar{t}) \\ \forall(\bar{x}, \bar{t}) &\in \bar{\Omega} \times \bar{T} \subset \mathbb{R}^d \times \mathbb{R} \end{aligned} \quad (1)$$

where $\mathcal{D}[\cdot]$ denotes the differential operator, \bar{u} is the quantity of interest, $\bar{x} \in \bar{\Omega} \subset \mathbb{R}^d$, $\bar{t} \in \bar{T} \subset \mathbb{R}$ for $d \geq 1$, $\bar{\Omega}$ denotes the spatial boundary contained in the d -dimensional Cartesian spatial space, \bar{T} denotes the temporal domain, $\bar{\lambda} \in \mathbb{R}$ is the model parameter, $\bar{f}(\bar{x}, \bar{t})$ is the external force, and \bar{K} is the notation for the abstract physical equation.

Deep neural networks are the core for PINNs in which inputs (\bar{x}, \bar{t}) map to output (\bar{u}) through an iterative composition of hidden layers. The composition consists of weights (w), biases (b), and linear or nonlinear activation function(s) (σ). The inputs undergo a linear composition within a neuron, where they are multiplied by respective weights and summed along with a bias term. Subsequently, this combined input is passed through a nonlinear activation function (σ). This allows the neural network to introduce nonlinearity, enabling the network to capture intricate relationships between inputs and outputs.

To train the neural network, one needs training set (Δ) , consisting of spatial boundary points (Δ_b), temporal boundary points (Δ_t), and interior points (Δ_{int}). As a result, the training set can be written as $\Delta = \Delta_t \cup \Delta_b \cup \Delta_{\text{int}}$. In this work, Δ_t , Δ_b , and Δ_{int} are considered to have N_t , N_b , and N_{int} training points, respectively. The total number of training points is denoted by N_{train} . To approximate the quantity of interest \bar{u} , one needs to minimize the loss function containing the physical model in the form of a PDE with initial and boundary conditions of (1). No additional data are required in the loss function for forward problems. The loss function $\bar{\mathcal{L}}$ is defined as follows:

$$\bar{\mathcal{L}}(\theta) = \text{Min}_{\theta} \left(\frac{1}{N_{\text{train}}} \sum_{n=1}^{N_{\text{train}}} \|\bar{K}(\bar{x}_n, \bar{t}_n)\|^2 \right) \quad (2)$$

where (\bar{x}_n, \bar{t}_n) represents the training tuple for each n . Minimizing this loss function using a suitable optimization algorithm provides optimal parameters $\theta = \{w, b\}$.

Now, we use the PINN algorithm for the dimensional Euler–Bernoulli beam equation and evaluate the corresponding performance. The dynamic Euler–Bernoulli beam equation is given by

$$\rho A \bar{u}_{\bar{t}\bar{t}} + EI \bar{u}_{\bar{x}\bar{x}\bar{x}\bar{x}} = \bar{f}(\bar{x}, \bar{t}) \quad \bar{x} \in [0, \bar{l}], \bar{t} \in [0, t_{\text{end}}]. \quad (3)$$

Here, \bar{l} and t_{end} refer to the length of the beam and final time, respectively. This equation models the transverse displacement of beam \bar{u} in the space–time domain subject to the external transverse force \bar{f} as shown in Fig. 1. This work considers a uniform cross-sectioned beam with constant

material properties throughout the beam. The parameters ρ and A denote the density and cross-sectional area of the beam, respectively. The parameters E and I are Young’s modulus and the moment of inertia of the beam, respectively. The external force \bar{f} acts nonuniformly on the body, and \bar{u} is the transverse displacement of the beam, which is the only unknown in the governing PDE. In addition, u_{tt} represents the second-order partial derivative of u with respect to t , and u_{xxxx} represents the fourth-order partial derivative of u with respect to x . The goal of the forward problem is to compute the transverse displacement of the beam supplemented with the initial and boundary conditions. For this study, simply supported beams are considered, which rest on two supports and are free to move horizontally. Real-world applications of simply supported beams include railway tracks and bridges, to name a few. Mathematically, the simply supported boundary condition for (3) is given by

$$\bar{u}(0, \bar{t}) = \bar{u}(\bar{l}, \bar{t}) = \bar{u}_{\bar{x}\bar{x}}(0, \bar{t}) = \bar{u}_{\bar{x}\bar{x}}(\bar{l}, \bar{t}) = 0.$$

For the numerical experiment, the parameter values of aluminum-like material are considered in the physical equation, which are widely used for making beams. The parameter values taken for the problem are $\rho = 2 \times 10^3 \text{ kg/m}^3$, $A = 5 \times 10^{-2} \text{ m}^2$, $E = 10^{10} \text{ N/m}^2$, and $I = 4 \times 10^{-4} \text{ m}^4$. In addition, the beam is taken to be $\pi^2 \text{ m}$ long, and the external force \bar{f} is taken to be $EI(1 - 16\pi^2) \sin(\bar{x}/\pi) \cos(4c\bar{t}/\pi)/\bar{l}^3 \text{ N}$, where $c = (EI/\rho A)^{1/2}$. Taking the final time to be $\pi^2/200$, the PDE to be solved takes the form

$$\begin{aligned} 10^2 \bar{u}_{\bar{t}\bar{t}} + 4 \times 10^6 \bar{u}_{\bar{x}\bar{x}\bar{x}\bar{x}} \\ = 4 \times 10^6 (1 - 16\pi^2) \sin(\bar{x}/\pi) \cos(800\bar{t}/\pi)/\pi^3 \end{aligned} \quad (4)$$

in the domain $\bar{x} \in [0, \pi^2]$ and $\bar{t} \in [0, \pi^2/200]$. For (4) to be well-posed, the initial condition of the beam is taken to be $\sin(\bar{x}/l)$ with zero initial velocity, where $l = \sqrt{\bar{l}}$.

For training the neural network, 16000 random training points are generated with the distribution $N_t = 2000$, $N_b = 4000$, and $N_{\text{int}} = 10000$. The neural network consists of four hidden layers with 20 neurons in each hidden layer. The tanh activation function, which is one of the most commonly used activation functions in the PINN literature, as described in the review paper [23], is chosen. The loss function (2) consists of the initial condition, boundary condition, and PDE. The PDE is regularized in the loss function with the residual parameter 0.1 [57]. The L-BFGS optimizer, which is again one of the most commonly used optimizers in the PINN literature [23], is used to minimize the loss function. As shown in Fig. 2 15000 epochs are performed. However, the figure clearly illustrates that the optimizer does not converge to the solution, and a vast training loss of 10^{14} is obtained. In addition, the graph shows that the optimizer is stuck in the local minima and hence will not converge even if the number of epochs is increased for the same neural network configuration.

In [16] and [58], the problem of free vibrations in the Euler–Bernoulli single-beam equation was successfully solved by PINNs, where the coefficients of the PDE were taken to be unity. This shows that PINNs can simulate the beam equations,

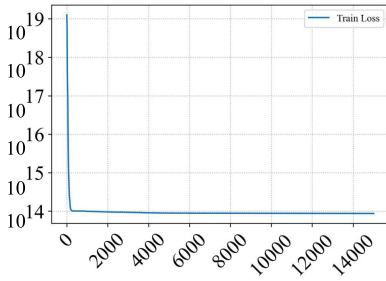


Fig. 2. L-BFGS training loss versus the number of epochs for the dimensional Euler-Bernoulli beam equation.

and the challenge lies in the multiscale coefficient values that arise when dealing with a real-life physical equation. The nonconvergence in our case is due to the high value of coefficients, which is due to the dimensional equation. Consequently, a pressing need arises to transform the dimensional form of the equation into a nondimensional form. It may be possible that for some configurations containing hundreds of hidden layers and neurons, this problem may be solved without the need to nondimensionalizing the PDE. However, nondimensionalization aims to provide computational tractability.

III. PINNs FOR NONDIMENSIONAL PDES

This section presents the proposed framework of using nondimensional equations in the PINN loss function. The method for nondimensionalizing the governing PDE is described first. Then, the algorithms for forward and inverse problems using dimensionless equations in PINNs are presented. To nondimensionalize the abstract PDE given by (1), the following transformations are performed:

$$\bar{x} = \xi_1(x); \quad \bar{t} = \xi_2(t); \quad \bar{u} = \xi_3(u); \quad \bar{f} = \xi_4(f) \quad (5)$$

where ξ_1 , ξ_2 , ξ_3 , and ξ_4 are suitable functions that map the dimensional quantities \bar{x} , \bar{t} , \bar{u} , and \bar{f} to the corresponding nondimensional quantities. After substituting the above transformations in (1) and introducing the dimensionless parameter λ , one obtains

$$\mathcal{K}(x, t) := \mathcal{D}[u](x, t; \lambda) - f(x, t) \\ \forall (x, t) \in \Omega \times T \subset \mathbb{R}^d \times \mathbb{R}. \quad (6)$$

The proposed framework uses dimensionless equations to simplify and stabilize the problem computationally. By nondimensionalizing the variables and parameters, they are kept within a specific range, resulting in improved performance and generalization of the neural network. Furthermore, dimensionless equations generate more interpretable solutions by eliminating the units of measure, making it easier to understand the underlying physical phenomena and to compare the results across different physical systems in the form of ratios and parameters. Hence, using dimensionless equations in PINNs can enhance the neural network's computational stability, generalization, and interpretability.

A. PINN Framework for Forward Problems

\mathcal{K} , the nondimensional PDE corresponding to the dimensional PDE $\bar{\mathcal{K}}$, is now used in the loss function \mathcal{L} defined as

follows:

$$\mathcal{L}(\theta) = \text{Min}_{\theta} \left(\frac{1}{N_{\text{train}}} \sum_{n=1}^{N_{\text{train}}} \|\mathcal{K}(x_n, t_n)\|^2 \right). \quad (7)$$

A schematic representation of the proposed PINN-based framework is illustrated in Fig. 4.

B. Nondimensional Euler-Bernoulli Beam Equation

We now test the nondimensional equation in the PINN framework and evaluate the corresponding performance. To nondimensionalize (3), the following transformations are used:

$$u = \bar{u}/l; \quad x = \bar{x}/l; \quad t = c\bar{t}/l^2; \quad f = \bar{f}l^3/(EI). \quad (8)$$

Upon substituting these values in (3), one obtains

$$u_{tt} + u_{xxxx} = f(x, t) \quad x \in [0, \pi], t \in [0, 1] \quad (9)$$

where $f(x, t) = (1 - 16\pi^2) \sin(x) \cos(4\pi t)$, with the initial and boundary conditions

$$u(x, 0) = \sin(x), \quad u_t(x, 0) = 0 \\ u(0, t) = u(\pi, t) = u_{xx}(0, t) = u_{xx}(\pi, t) = 0.$$

For the error estimation, the relative percentage error (\mathcal{R}) used in [57] is chosen. Here, u^* is the prediction and u is the analytical solution

$$\mathcal{R} = \frac{\|u^* - u\|_2}{\|u\|_2} \times 100.$$

The same neural network architecture as the previous case is chosen to solve this resulting nondimensional PDE. A low training loss is obtained, indicating that the PINN is trained successfully. The analytical solution for this case is $u(x, t) = \sin(x) \cos(4\pi t)$, which is used to quantify the error in the approximated solution. The nondimensional displacement of the Euler-Bernoulli beam is computed within $\mathcal{R} = 5.3e - 4\%$. The nondimensional displacement prediction using PINN is shown in Fig. 3(a). Fig. 3(b) shows the absolute error between the exact and predicted solutions.

The contour plot for the approximate solution shows the dynamics of a simply supported beam under a force, where the x -axis represents the time, the y -axis represents the position along the length of the beam, and the colors represent the displacement of the beam. In Fig. 3(b), the red regions indicate high displacement, while the blue regions indicate low displacement. There is a strong displacement at the position of the beam when a substantial force is applied, which is consistent with the known physics of this system. The network accurately captures the displacement behavior of the beam, which is evident by the smooth and continuous transition of colors across the plot.

The contour plot for the error in Fig. 3(b) shows the difference between the approximate solution obtained from the network and the true solution. The x -axis represents the time, the y -axis represents the position along the length of the beam, and the colors represent the error. The red regions indicate high error, while the blue regions indicate low error. The areas where the training point concentration is low account for more

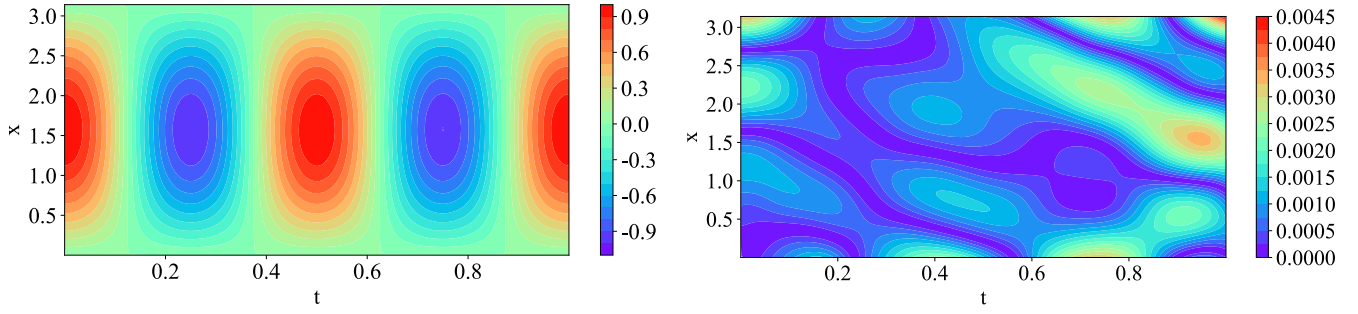


Fig. 3. Nondimensional Euler-Bernoulli beam equation color bar represents predicted solution (u^*) (left). Absolute error in prediction ($|u - u^*|$) (right).

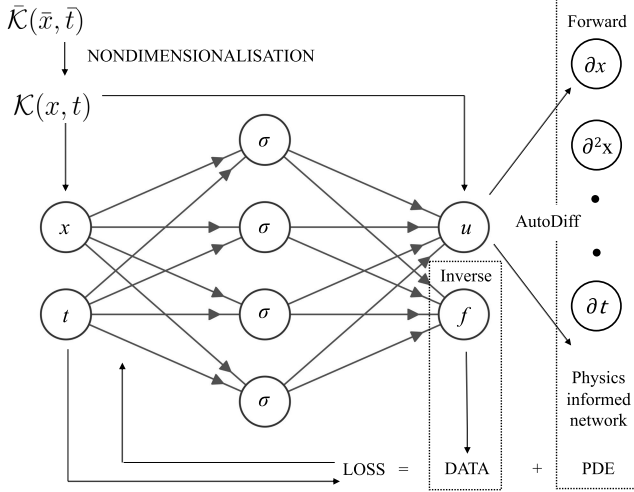


Fig. 4. PINN framework for beam systems: For forward problems, the loss function comprises the nondimensional PDEs and the boundary and initial conditions. For inverse problems, the nondimensional PDEs are supplemented with extra data and potential initial/boundary conditions.

error, and areas where the concentration of training points is more have relatively low error. One approach to reduce the error is to have more training points in the regions of high error. However, the overall error is low, which indicates that the network accurately captures the displacement behavior of the beam.

From Fig. 3(b), the PINNs are found to solve the dimensionless Euler-Bernoulli beam equation accurately, and hence, for all further experiments, nondimensional PDEs are simulated using PINNs. In addition, the nondimensional displacement is henceforth referred to as displacement for conciseness. The presented methodology predicts the dimensionless quantities, and hence, all the plots of results and their associated error plots are dimensionless. Consequently no units are mentioned in the plots of the presented results. Next, we describe the inverse problem-solving strategy using nondimensional equations.

C. PINN Framework for Inverse Problems

The abstract dimensionless PDE described by (6) is well-posed, and the forward problem can be solved uniquely. However, in the case of an inverse problem, the problem is ill-posed and either the initial/boundary conditions or the parameters/forces are unknown. Hence, the generic abstract

PDE can be rewritten as

$$\mathcal{K}'(x, t) := \mathcal{D}[u](x, t; \lambda) - f(x, t) \quad \forall (x, t) \in \Omega \times T \subset \mathbb{R}^d \times \mathbb{R}. \quad (10)$$

The algorithm for the PINN framework is presented to solve inverse problems.

Algorithm 1 Inverse PINN Algorithm

Goal: To predict the unknown parameter $\bar{\lambda}$ or function $\bar{f}(\bar{x}, \bar{t})$.

Step 1: Nondimensionalize the governing PDE to approximate the dimensionless parameter λ or function $f(x, t)$.

Step 2: Choose the training set from the space-time domain $\Omega \times T$, and augment with $(x_{\text{data}}, t_{\text{data}})$ at which additional data (u_{data}) are provided.

Step 3: Construct a feedforward deep neural network with inputs (x, t) and outputs u, λ , or $f(x, t)$.

Step 4: Minimize the loss function (11) with a suitable optimization algorithm, and find the optimal parameters.

Step 5: Use the optimal parameters to approximate the parameter λ^* or the function $f^*(x, t)$.

The aim of the inverse problem is to predict the unknown parameter λ or the force function $f(x, t)$, when data are provided for the observable u in some part of the training domain. In this article, u_{data} denotes the available data for the inverse problem at N_{data} points. The prediction of the unknown parameter requires additional information in the loss function as shown in Fig. 4. It is essential for the Jacobian matrix used in the inverse operation study using neural networks to exhibit a nonzero determinant, to be invertible, and to possess a reasonable ratio between its largest and smallest eigenvalues to guarantee a unique solution and ensure computational stability. The algorithm for the inverse problem is the same as for the forward problem with a minor modification in the loss function. In addition to the output u , the PINNs now predict the unknown parameter, force, and initial or boundary conditions of the physical problems by leveraging the known data. The loss function for the inverse problem can be defined as

$$\mathcal{L}'(\theta) = \text{Min}_{\theta} \left(\frac{1}{N_{\text{train}}} \sum_{n=1}^{N_{\text{train}}} \|\mathcal{K}(x_n, t_n)\|^2 \right)$$

$$+ \frac{1}{N_{\text{data}}} \sum_{n=1}^{N_{\text{data}}} \|u_{\text{data}}(x_n, t_n) - u_{\text{pred}}(x_n, t_n)\|^2 \Bigg). \quad (11)$$

Here, u_{pred} denotes the prediction of u by the neural network section implementing the PINN algorithm for forward and inverse problems of dimensionless beam equations.

IV. NUMERICAL EXPERIMENTS AND DISCUSSION

In this section, five numerical experiments are presented. The experiments are conducted in a progressive manner, beginning with simple models such as a single-beam system and then progressing to more complex ones such as a double-beam connected to a Winkler foundation. To verify the proposed method, we first investigate forward and inverse problems for a single beam, which serves as the proof of the concept. Then, we apply the method to more intricate cases of double-beam systems to simulate forward and inverse problems.

A. Timoshenko Beam Forward Problem

The Euler–Bernoulli theory of beams is widely used in the literature and has been successfully applied in structures such as the Eiffel Tower and Ferris wheels. However, it does not consider the effects of transverse shear deformations, which are often significant in the vertical displacements of short and thick beams [59]. Timoshenko beam theory provides a mathematical framework for analyzing thick-beam bending [59]. According to the Timoshenko theory, upon the action of an external force, the beam undergoes some cross-sectional rotation in addition to transverse displacement. Mathematically, the dynamics are modeled by a coupled system of PDEs with two variables: transverse displacement and cross-sectional rotation. The model is given by

$$\begin{aligned} \rho I \bar{\theta}_{\bar{t}\bar{t}} - EI \bar{\theta}_{\bar{x}\bar{x}} - kAG(\bar{w}_{\bar{x}} - \bar{\theta}) &= 0 \\ \rho A \bar{w}_{\bar{t}\bar{t}} - kAG(\bar{w}_{\bar{x}\bar{x}} - \bar{\theta}_{\bar{x}}) &= \bar{g}(\bar{x}, \bar{t}) \end{aligned} \quad (12)$$

where ρ , A , E , and I have the usual meaning as in the case of the Euler–Bernoulli beam; k is called the Timoshenko shear coefficient; G is the shear modulus; and $\bar{g}(\bar{x}, \bar{t})$ is the external force acting on the beam. The transverse displacement is $\bar{w}(\bar{x}, \bar{t})$, and $\bar{\theta}(\bar{x}, \bar{t})$ is the cross-sectional rotation of the beam at position \bar{x} and time \bar{t} . After nondimensionalizing (12) and taking the resulting parameters [60] to be unity, the nondimensional equation can be written as follows:

$$\begin{aligned} \theta_{tt} - \theta_{xx} + (\theta - w_x) &= 0 \\ w_{tt} + (\theta - w_x)_x &= g(x, t). \end{aligned} \quad (13)$$

We consider the external force [61] to be $g(x, t) = \cos(t) - (\pi/2) \sin(x) \cos(t)$ and the computational domain to be $x \in [0, \pi]$ and $t \in [0, 1]$. To make (13) well-posed, the initial and boundary conditions are supplemented as

$$\begin{aligned} \theta(x, 0) &= \frac{\pi}{2} \cos(x) + \left(x - \frac{\pi}{2}\right), & \theta_t(x, 0) &= 0 \\ w(x, 0) &= \frac{\pi}{2} \sin(x), & w_t(x, 0) &= 0 \\ \theta(0, t) &= \theta(\pi, t) = w(0, t) = w(\pi, t) = 0. \end{aligned}$$

To estimate the error in the approximated solutions, the analytical solution for the considered problem is used, which is

$$\begin{aligned} \theta(x, t) &= \left(\frac{\pi}{2} \cos(x) + \left(x - \frac{\pi}{2}\right)\right) \cos(t) \\ w(x, t) &= \frac{\pi}{2} \sin(x) \cos(t). \end{aligned}$$

When analytical solutions are not available, there are various ways to validate the PINN solution. One approach is to compare the solutions with those obtained using numerical methods such as finite difference, finite element, finite volume, or spectral methods. This can be done by comparing the predicted solutions from the PINNs with the solutions from the numerical simulation for the same physical equation. Another approach is to compare the solutions obtained through PINNs with experimental data. One can compare the predicted solutions from the PINNs with values experimentally measured over space and time. Finally, one can validate the solutions obtained through PINNs by checking whether they satisfy the known physical constraints of the system. In summary, one can use available experimental data, numerical methods, or physical constraints to evaluate the accuracy of the solution obtained using PINNs.

The difficulty of solving a system of PDEs is greater than that solving a single PDE, but the neural network structure used for the Euler–Bernoulli equation is successful in approximating solutions for Timoshenko beams. In particular, the transverse displacement of the beam is computed within $\mathcal{R} = 3.3e - 4\%$, and the cross-sectional rotation is approximated within $\mathcal{R} = 2.8e - 3\%$. Approximated solutions and absolute errors in predicting the transverse displacement and cross-sectional rotation are presented in Figs. 5 and 6. Fig. 5 demonstrates that when a sinusoidal force is applied to a Timoshenko beam, the beam bends more than it rotates. As indicated by the scale in the figures, the maximum deflection is 1.44 and the maximum rotation is 0.32. In addition, the low error in predictions demonstrates that even with the increase in PDE complexity, the PINN successfully solves the Timoshenko PDE with comparable results to the Euler–Bernoulli equation.

We compare the results obtained from our method with three other methods. The first method we consider is the widely used numerical technique called the finite difference method (FDM). The other two methods are neural-network-based approaches, namely, physics-guided neural networks (PGNNs) [28], [62], [63], [64], [65] and gradient-enhanced PINNs (gPINNs) [66]. First, for FDM we use a central difference scheme to approximate space derivatives and a leapfrog scheme to approximate time derivatives. This approach allows us to solve problems with second-order accuracy in space and time. The results for the Timoshenko beam show that PINNs can achieve a higher level of accuracy than the FDM even with a smaller number of training points. Specifically, 30 000 points are used in the FDM scheme, while only 16 000 points were used for training with PINNs, and Table I indicates that PINNs perform better than FDM.

Second, the performance of PINN is compared with a neural-network-based approach PGNN, which leverages

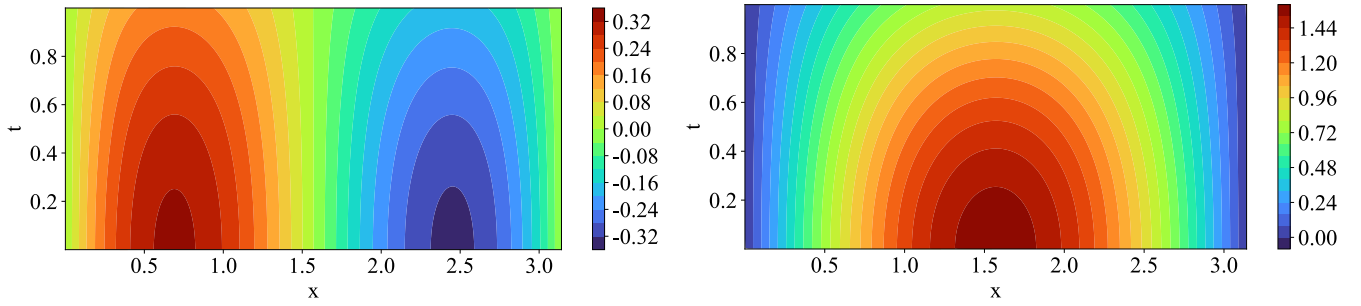


Fig. 5. Timoshenko single-beam; color bar represents cross-sectional rotation (θ^*) (left). Transverse displacement (w^*) (right).

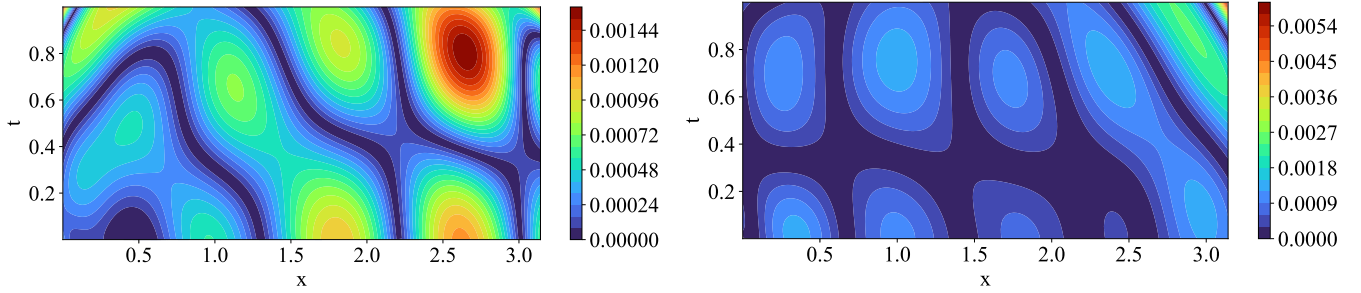


Fig. 6. Timoshenko single-beam absolute error in predictions. Left: $|\theta - \theta^*|$. Right: absolute error $|w - w^*|$.

TABLE I
TIMOSHENKO BEAM: \mathcal{R} at $t = 1$

w, θ	PINN	FDM	PGNN	gPINN
w (%)	3.3e-4	0.005615	0.002739	0.249849
θ (%)	2.8e-3	0.004733	3.486727	5.498449

physical knowledge embedded in the available data, for instance, the relationship between beam acceleration and displacement for the Timoshenko beam problem. Accelerometers can be used at discrete locations along the beam to obtain acceleration data. Acceleration data at five equidistant points along the beam are used, with 2000 data points at each location. This dataset is augmented with the boundary and initial conditions of displacement to match the training data size of PINN. PGNN is a deep neural-network-based architecture with inputs: position (x), time (t), and acceleration. Displacement (w) is taken as the output of this neural network. Training PGNN with identical hyperparameters to those used in PINN, PGNN predicts the displacement (w) with an error of approximately 0.002739%, as shown in Table I.

Furthermore, using the displacement values (w), the neural network's auto differentiation, and (13), we derived θ_x . Subsequently, a second neural network was constructed to predict θ , where θ_x is used as the input. The boundary and initial conditions for cross-sectional rotation (θ) are also used to guide the PGNN toward the optimal solution. After training the PGNN, cross-sectional rotation is predicted with approximately 3.486727% error. It can be inferred from Table I that both displacement and rotation predictions exhibited higher errors than PINN. This discrepancy can be attributed to the restricted availability of acceleration data at only discrete spatial locations within the interior domain rather than a random distribution across the entire domain. Furthermore, the second neural network, used for rotation prediction,

demonstrated inferior performance potentially due to error propagation.

Third, we perform another comparison with a neural-network-based method to simulate PDEs, gPINN [66], which differs from PINN in terms of the loss function. The acronym "gPINN" proposed in [66] is used in this work instead of "GPINN" as it is used for another method [67]. In addition to the loss function of PINN, gPINN leverages gradient information of the PDE residual and embeds the gradient into the loss function. For the Timoshenko beam problem, derivatives of the system of PDE (13) with respect to space (x) and time (t) are supplemented in the loss function. Table I shows that gPINN exhibits higher relative error percentages in learning displacement and cross-sectional rotation than PINN. The high-order derivatives of the physical equations in the loss function of gPINN make it challenging for autodifferentiation [68] and backpropagation of the loss function, resulting in poor predictions of deflection and rotation for the Timoshenko beam. Table I demonstrates that PINN outperforms FDM, PGNN, and gPINN in accurately predicting displacement and cross-sectional rotation for the Timoshenko beam, emphasizing its superior performance compared with the three alternative methods.

B. Timoshenko Beam Inverse Problem

This section addresses the inverse problem for the Timoshenko beam, with the aim to determine the material properties of a beam leveraging the PDE and beam's displacement and rotation data. In structural engineering, the inverse problem of a Timoshenko beam PDE is significant for determining the beam system's structural behavior and for health monitoring. This helps engineers infer the internal material properties and unknown forces from observed responses such as displacement and rotation measurements. The PINN solves this problem by

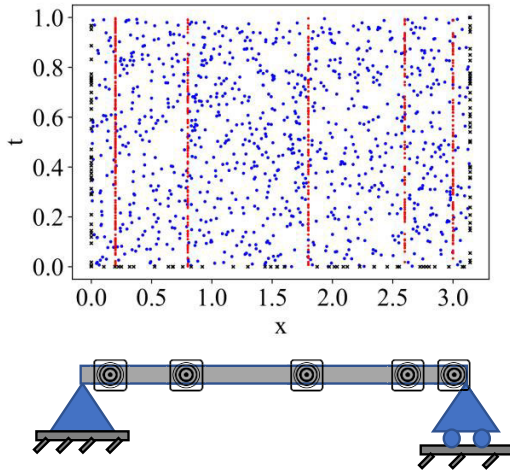


Fig. 7. Data to learn the parameters for the Timoshenko single-beam. Blue dots: collocation points. Red dots: additional data points of rotations (θ) and displacement (u). Black dots: initial and boundary points.

combining the knowledge of physics and deep learning. The PINN uses a neural network to learn the mapping between the unknown parameters of the PDE and observed data while incorporating the constraints of physics in the form of PDEs. This parameter identification aids in providing crucial information for structural diagnosis and repair and helps engineers ensure the safety and stability of structures. The Timoshenko model for parameter estimation is presented as follows:

$$\begin{aligned} \alpha \theta_{tt} - \theta_{xx} + (\theta - w_x) &= 0 \\ w_{tt} + (\theta - w_x)_x &= g(x, t). \end{aligned} \quad (14)$$

In the context of the inverse problem of the Timoshenko beam, the PINN is trained on the observed deflections and rotations of the beam, and the material properties are treated as the unknowns to be estimated. In this case, the force $g(x, t)$ applied to the beam is considered to be known, and the only unknown in the model is α . This makes the problem ill-posed, requiring additional data a priori to predict the unknown parameter. For $\alpha = 1$, the transverse displacement and cross-sectional rotation data obtained from the forward problem are supplied to approximate the parameter value. These data are not error-free and come with $10^{-3}\%$ error for transverse displacement and with $10^{-4}\%$ error for cross-sectional rotation. As shown in Fig. 7 the additional data are supplied on 5000 points (red dots) at five positions on the beam ($x = 0.2, 0.8, 1.8, 2.6, 3$). In practice, these data can be collected using sensors installed at the corresponding locations on the beam as shown in Fig. 7.

To solve the inverse problem, the neural network consists of 1600 random training points with the distribution $N_i = 200$, $N_b = 400$, and $N_{int} = 1000$. To regularize the PDE term in the loss function, a regularization parameter of 1 was chosen [25]. Using the L-BFGS optimizer, 5000 iterations are performed and the other parameters are kept the same as in the forward Timoshenko problem. At $t = 0.5$, the unknown parameter $\alpha = 1.0136$ is learned.

We perform a comparison between the PINN and DNNs, as using a numerical iterative method for inverse problems

is computationally expensive. From PINNs, at $t = 0.5$, the unknown parameter $\alpha = 1.0136$ is learned. We use DNNs to identify the parameters of a Timoshenko single-beam. We use the same architecture for DNN as used by the PINN. The predicted value of alpha is 0.6124 using DNN. PINN is more accurate than DNNs for the inverse problem of beam systems.

However, there are several issues that one may need to take care of while solving inverse problems through the presented framework. First, to avoid overfitting, the minimum training data points required to solve the problem should be determined empirically by gradually increasing the number of training points until the model's performance is satisfactory. Second, for some physical problems, noisy data may lead to nonconvergence of the optimization algorithm. Hence, suitable filtering or preprocessing of data may be required before using the PINN framework. Finally, for every run of the neural network, one may learn a different parameter or function value; due to the convergence of the optimizers at different local minima, it may be useful to find the statistics of the inverse problem solution through multiple runs.

The experimental results for single-beam equations illustrate that PINNs can efficiently solve forward and inverse problems for single beams. In this study, we investigate the ability of PINNs to handle more complex systems, specifically double-beam systems connected by a Winkler foundation, as depicted in Fig. 10.

C. Euler–Bernoulli Double-Beam Forward Problem

In this section, and for all further experiments, forced transverse vibrations of two parallel beams are studied. Structurally, two parallel beams of equal lengths joined by a Winkler massless foundation are considered. Both the beams are considered slender and have homogeneous material properties. The transverse displacement of both the beams is governed by the following system of PDEs [41]:

$$\begin{aligned} m_1 \bar{w}_{1,tt} + K_1 \bar{w}_{1,xxxx} + k(\bar{w}_1 - \bar{w}_2) &= \bar{f}_1(\bar{x}, \bar{t}) \\ m_2 \bar{w}_{2,tt} + K_2 \bar{w}_{2,xxxx} + k(\bar{w}_2 - \bar{w}_1) &= \bar{f}_2(\bar{x}, \bar{t}). \end{aligned} \quad (15)$$

Here, \bar{w}_1 and \bar{w}_2 are the beam displacements for the first and second beams, respectively. The distributed continuous forces acting transversely on the beams are \bar{f}_1 and \bar{f}_2 as shown in Fig. 10. The product of the density and the cross-sectional area of the beams is given by $m_1 = \rho_1 A_1$ for the first beam and $m_2 = \rho_2 A_2$ for the second beam. The parameters K_1 and K_2 denote the flexural rigidity of the beams and are given by $K_1 = E_1 I_1$ and $K_2 = E_2 I_2$. The stiffness modulus of the Winkler elastic layer connecting both the beams is given by k . For simplicity, we consider $m_1 = m_2$ and $K_1 = K_2$, and nondimensionalize (15). After taking all the resulting parameters to be unity, the nondimensional equation has the same form as (15) with unit coefficients. The initial conditions are

$$\begin{aligned} w_1(x, 0) &= \sin(x), & w_{1,t}(x, 0) &= 0 \\ w_2(x, 0) &= \frac{\pi}{2} \sin(x), & w_{2,t}(x, 0) &= 0. \end{aligned}$$

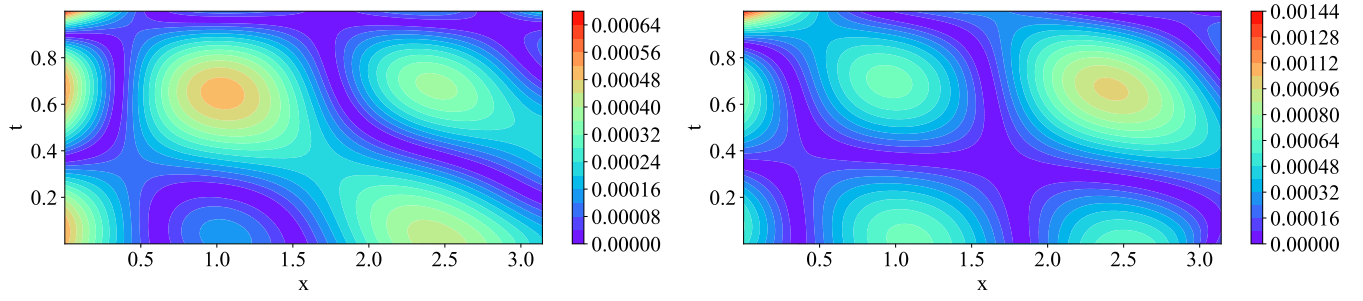


Fig. 8. Euler–Bernoulli double-beam: color bar represents absolute error in predictions. Left: $|w_1 - w_1^*|$. Right: $|w_2 - w_2^*|$.

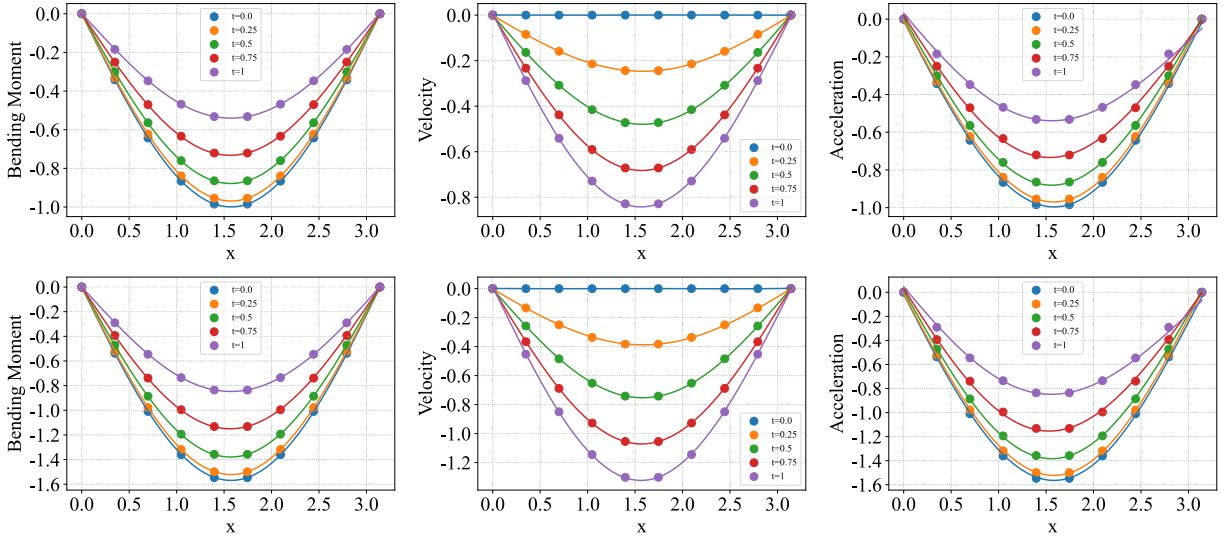


Fig. 9. Derived quantities for the Euler–Bernoulli double-beam. Scattered points represent the exact solution and the continuous line refers to the derived solution. Top: first beam, left: bending moment, mid: velocity, and right: acceleration. Bottom: second beam, left: bending moment, mid: velocity, and right: acceleration.

All the four ends of the beams are assumed to be simply supported, expressed as

$$\begin{aligned} w_1(0, t) = w_1(\pi, t) = w_{1,xx}(0, t) = w_{1,xx}(\pi, t) = 0 \\ w_2(0, t) = w_2(\pi, t) = w_{2,xx}(0, t) = w_{2,xx}(\pi, t) = 0. \end{aligned}$$

The external acting force is

$$\begin{aligned} f_1(x, t) &= \left(1 - \frac{\pi}{2}\right) \sin(x) \cos(t) \\ f_2(x, t) &= \left(\frac{\pi}{2} - 1\right) \sin(x) \cos(t). \end{aligned}$$

For the considered problem, the analytical solution is given by

$$w_1(x, t) = \sin(x) \cos(t), \quad w_2(x, t) = \frac{\pi}{2} \sin(x) \cos(t).$$

In addition to computing the beam displacements, derived quantities such as velocity, acceleration, and bending moment are also computed for this problem. These derived quantities also help in the prognosis and diagnostics of the system. For instance, the bending moment estimates the bending effect when an external force is applied to a structural element. Estimating the bending moment can be used to quantify the bending upon the action of applied forces. The beam is the most common structural member vulnerable to bending

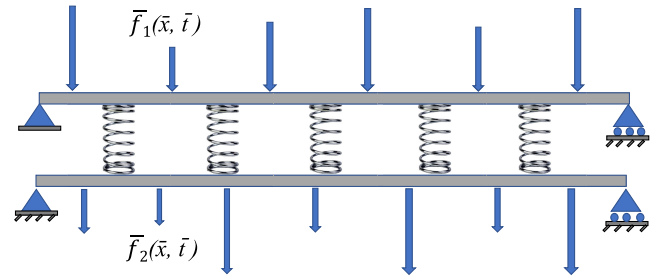


Fig. 10. Double-beam system connected by a Winkler foundation.

moments because it can bend at any point along its length when subjected to an external force.

For simulating Euler–Bernoulli double beams, the same neural network architecture as for the single Euler–Bernoulli beam is considered. The only change is in the residual parameter, which is 1 for this case. The results are illustrated in Figs. 8 and 9 and Table II. The absolute difference between the PINN-predicted solution and the exact solution for the first beam is approximately 10^{-4} , and for the second beam, it is approximately 10^{-3} , as shown in Fig. 8. The bending moment, velocity, and acceleration are computed using the neural network’s autodifferentiation and backpropagation features. Table II describes the efficiency in the computation of

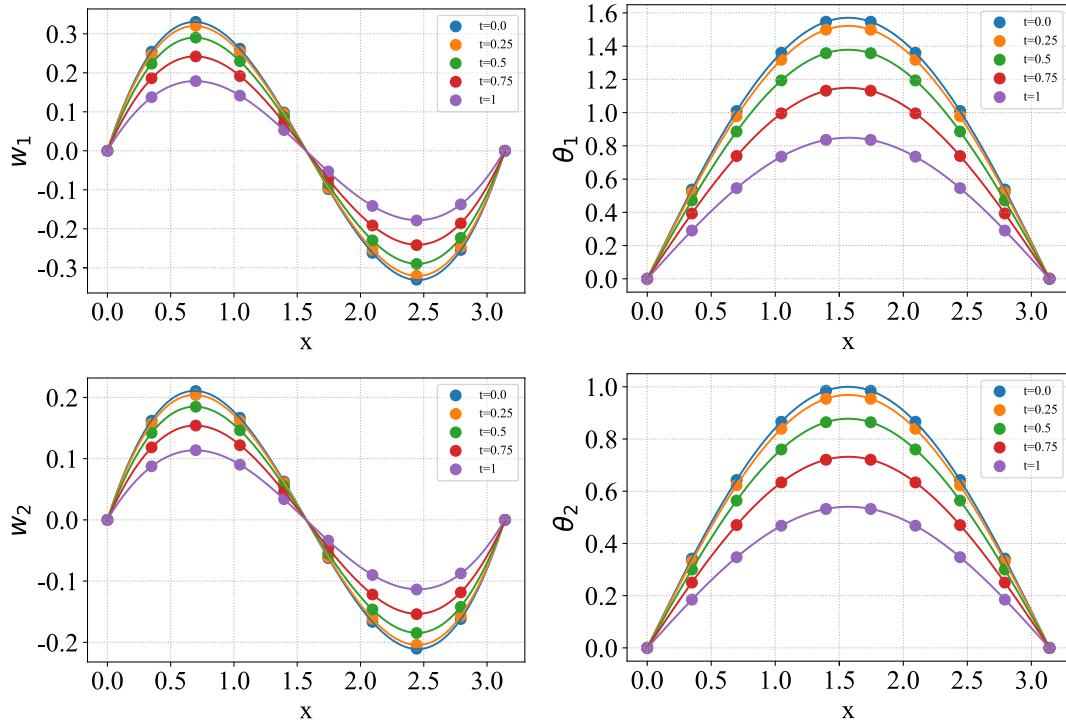


Fig. 11. Timoshenko double-beam. Scattered points represent the exact solution, and the continuous line refers to the predicted solution. Top: first beam. Left: displacement (w_1). Right: rotation (θ_1). Bottom: second beam. Left: displacement (w_2). Right: rotation (θ_2).

TABLE II
EULER-BERNOULLI DOUBLE-BEAM: \mathcal{R} AT $t = 1$

	First beam	Second beam
Displacement (%)	1.9348×10^{-5}	4.3253×10^{-5}
Bending Moment (%)	9.6112×10^{-4}	6.5506×10^{-4}
Velocity (%)	1.9043×10^{-3}	2.0161×10^{-3}
Acceleration (%)	1.9011×10^{-2}	1.4442×10^{-2}

these quantities at $t = 1$ for both the beams. The relative percent error in computing the transverse displacement of the beams is on the order of 10^{-5} , and for acceleration, this error is on the order of 10^{-2} , which is very low and shows the potential of physics-informed learning. Fig. 9 illustrates the computed velocity, bending moment, and acceleration of both the beams.

D. Timoshenko Double-Beam Forward Problem

The double-beam system modeled by Euler-Bernoulli theory can also be modeled using Timoshenko theory under the same assumptions as described for the single Timoshenko equations [40]. In addition to providing the transverse displacement of the beams, the Timoshenko theory also provides the cross-sectional rotation of both the beams through the system of PDEs [40] given by

$$\begin{aligned}
 kA_1G(\bar{\theta}_{1\bar{x}} - \bar{w}_{1\bar{x}\bar{x}}) + \rho A_1\bar{w}_{1\bar{t}\bar{t}} + K(\bar{w}_1 - \bar{w}_2) &= \bar{f}_1(\bar{x}, \bar{t}) \\
 EI_2\bar{\theta}_{2\bar{x}\bar{x}} + GA_2k(\bar{w}_{2\bar{x}} - \bar{\theta}_2) - \rho I_2\bar{\theta}_{2\bar{t}\bar{t}} &= 0 \\
 kA_2G(\bar{\theta}_{2\bar{x}} - \bar{w}_{2\bar{x}\bar{x}}) + \rho A_2\bar{w}_{2\bar{t}\bar{t}} + K(\bar{w}_2 - \bar{w}_1) &= \bar{f}_2(\bar{x}, \bar{t}) \\
 EI_1\bar{\theta}_{1\bar{x}\bar{x}} + GA_1k(\bar{w}_{1\bar{x}} - \bar{\theta}_1) - \rho I_1\bar{\theta}_{1\bar{t}\bar{t}} &= 0 \quad (16)
 \end{aligned}$$

where $\bar{w}_i(\bar{x}, \bar{t})$ and $\bar{\theta}_i(\bar{x}, \bar{t})$, $i = 1, 2$ denote the transverse displacement and cross-sectional rotation of the beams

respectively. K is the stiffness modulus of the Winkler elastic layer. G is the shear modulus, and k is the Timoshenko shear coefficient. The rest of the parameters have the usual meanings as described earlier. For simplicity, we consider $A_1 = A_2$ and $I_1 = I_2$ and nondimensionalize (16). With some additional assumptions, the nondimensional equation has the same form as (16) with unit coefficients. For the numerical experiment, the initial state of the double-beam system is taken to be

$$\begin{aligned}
 \theta_1(x, 0) &= \left(\frac{\pi}{2} \cos(x) + \left(x - \frac{\pi}{2}\right)\right), \quad \theta_{1t}(x, 0) = 0 \\
 w_1(x, 0) &= \frac{\pi}{2} \sin(x), \quad w_{1t}(x, 0) = 0 \\
 \theta_2(x, 0) &= \frac{2}{\pi} \left(\frac{\pi}{2} \cos(x) + \left(x - \frac{\pi}{2}\right)\right), \quad \theta_{2t}(x, 0) = 0 \\
 w_2(x, 0) &= \sin(x), \quad w_{2t}(x, 0) = 0.
 \end{aligned}$$

Simply supported boundary conditions are provided to make the problem well-posed

$$\begin{aligned}
 \theta_1(0, t) &= \theta_1(\pi, t) = w_1(0, t) = w_1(\pi, t) = 0 \\
 \theta_2(0, t) &= \theta_2(\pi, t) = w_2(0, t) = w_2(\pi, t) = 0.
 \end{aligned}$$

Here, $f_1(x, t)$, $f_2(x, t)$, and the analytic solutions are as follows:

$$\begin{aligned}
 f_1(x, t) &= \cos(t)(1 - \sin(x)) \\
 f_2(x, t) &= \frac{2}{\pi} \cos(t) - \frac{\pi}{2} \sin(x) \cos(t) \\
 \theta_1(x, t) &= \left(\frac{\pi}{2} \cos(x) + \left(x - \frac{\pi}{2}\right)\right) \cos(t) \\
 \theta_2(x, t) &= \frac{2}{\pi} \left(\frac{\pi}{2} \cos(x) + \left(x - \frac{\pi}{2}\right)\right) \cos(t) \\
 w_1(x, t) &= \frac{\pi}{2} \sin(x) \cos(t), \quad w_2(x, t) = \sin(x) \cos(t).
 \end{aligned}$$

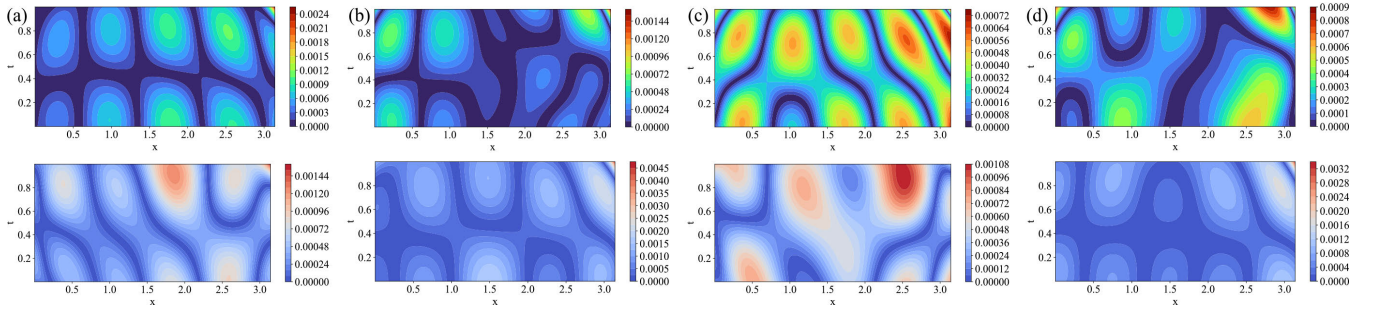


Fig. 12. Timoshenko double-beam absolute errors in prediction; θ_1 and w_1 are the rotation and displacement of the first beam, respectively, and θ_2 and w_2 are the rotation and displacement of the second beam, respectively. Top: 16 000 training points (a) $|\theta_1 - \theta_1^*|$, (b) $|w_1 - w_1^*|$, (c) $|\theta_2 - \theta_2^*|$, and (d) $|w_2 - w_2^*|$. Bottom: 1600 training points (a) $|\theta_1 - \theta_1^*|$, (b) $|w_1 - w_1^*|$, (c) $|\theta_2 - \theta_2^*|$, and (d) $|w_2 - w_2^*|$.

TABLE III

TIMOSHENKO DOUBLE-BEAM: HYPERPARAMETERS

No. of points	N_i	N_b	N_{int}	Layers	Neurons	Epochs
16000	2000	2000	10000	4	20	15K
1600	200	200	1000	4	20	15K

TABLE IV

TIMOSHENKO DOUBLE-BEAM: \mathcal{R} AT $t = 1$

	16000 points	1600 points
θ_1 (%)	1.6038×10^{-3}	2.6211×10^{-3}
w_1 (%)	3.9302×10^{-5}	2.503×10^{-4}
θ_2 (%)	1.0826×10^{-3}	4.9405×10^{-3}
w_2 (%)	7.8614×10^{-5}	3.4904×10^{-4}

Two experiments are performed, varying the number of training points, as shown in Table III. Table IV shows the relative percent error in approximating the transverse displacement and cross-sectional rotations for both the beams. For cross-sectional rotations θ_1 and θ_2 , the magnitude of the percent error remains the same even for fewer training points.

Using a large number of training points can increase the training time and may not be feasible for problems with many parameters. In these cases, using fewer training points can lead to less accurate solutions, but they can be obtained relatively faster. This approach allows engineers to make informed decisions about the parameters, and once optimal parameters have been identified, forward solutions can be recalculated with higher accuracy by using more training points. This is referred to as training with fewer points for the forward problem. The absolute difference between the predicted and exact solutions of θ_1 , w_1 , θ_2 , and w_2 , even for 1600 training points, is very small as shown in Figs. 11 and 12. Fig. 11 presents the PINNs; prediction for a double Timoshenko beam. The scattered points refer to the exact solution, and the continuous line represents the predicted solution. The force is applied uniformly in both the beams; however, the deflection and rotation of the first beam are greater than those of the second beam. The results in Fig. 12 indicate that for the second beam, a larger number of training points (16 000) result in a more accurate prediction of deflection and rotation than a smaller number of training points (1600). Conversely, for the first beam, a smaller number of training points (1600) result in a more accurate prediction of the quantity of interest than a larger number of training

points (16 000). In any case, the difference in absolute error is relatively small, demonstrating that even with fewer training points, PINNs can still produce accurate predictions.

E. Timoshenko Double-Beam Inverse Problem

The applied force on structural systems is critical for structural design and condition assessment. In design, control, and diagnosis, accurate estimation of dynamic forces acting on a structure is essential. These details can be used to evaluate the structural condition. For example, understanding the impact of heavy vehicles on bridge structures can aid in detecting early damage to them. Indirect force determination is of special interest when the applied forces cannot be measured directly, while the responses can be measured easily.

For the inverse problem, three distinct experiments are performed on (16). First, the unknown parameter is learned from the Timoshenko double-beam system. We consider the unknown parameter to be ρA_1 from (16). For the value of $\rho A_1 = 1$, the data for transverse displacement and cross-sectional rotation are provided at some points in the computational domain. Second, the unknown applied function on the first beam is learned by providing noise-free simulated displacement and cross-sectional rotation data. For this case, all other parameters, and initial and boundary conditions are considered to be known, and only the function $f_1(x, t)$ is unknown. Third, the same force function is predicted by providing noisy displacement and cross-sectional rotation data. The data generated for learning the function in the second case are corrupted with noise to be used in the third case. The exact solution for the function to be learned in the second and third cases is $\cos(t)(1 - \sin(x))$.

The inverse problem in engineering refers to the process of estimating unknown parameters or functions from a set of measured data. In PINNs, the inverse problem is usually solved by training a neural network to fit the measured data and the known physical laws. However, the measured data can be affected by various sources of noise, which can make estimation of the quantity of interest more challenging. The noise can make the measured data unreliable, and the neural network may not be able to accurately estimate the unknown parameters or functions. In such a scenario, the optimizer of the neural network does not necessarily converge to local minima.

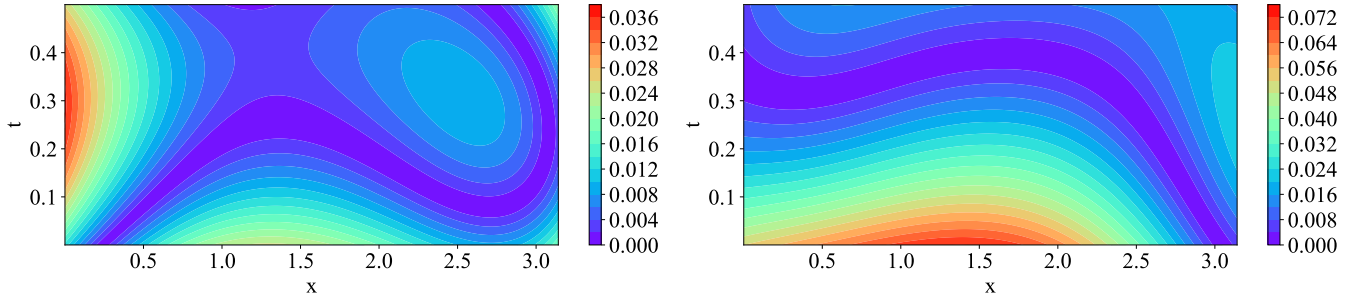


Fig. 13. Timoshenko double-beam inverse problem: absolute error in the prediction of force when the additional data of rotation and deflections provided at five locations has no noise (left) and 20% Gaussian noise (right).

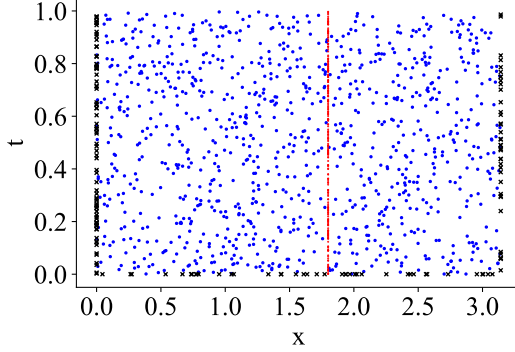


Fig. 14. Data to learn material properties for the Timoshenko double-beam: Blue dots: collocation points. Red dots: additional data points of displacement and rotation for the double-beam at one location. Black dots: initial and boundary points.

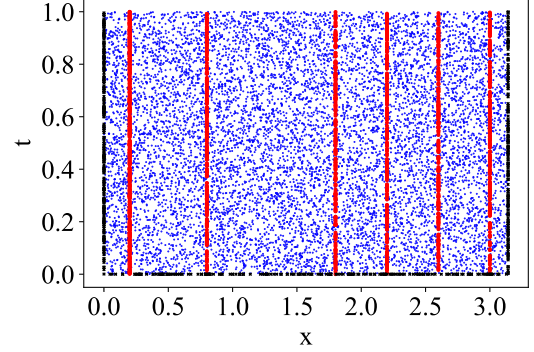


Fig. 15. Data to learn force for the Timoshenko double-beam: Blue dots: collocation points. Red dots: additional data points of displacement and rotation for the double-beam at six different locations. Black dots: initial and boundary points.

The same neural network architecture is used as in the forward double-beam Timoshenko problem, with residual parameter 1 to regularize the physical equation in the loss function. Here, 2500 epochs are performed using the L-BFGS optimizer to train the neural network. For learning the parameter, 5000 data points are provided at $x = 1.8$, as shown in Fig. 14. The exact value of the unknown parameter is $\rho A_1 = 1$ in (16), and the predicted value of the parameter using the PINN framework is 1.0208, which is close to the desired value. Even for a system of four PDEs, by only providing data at one particular beam location, the unknown parameter is learned successfully using PINNs. This shows that PINNs can handle large complex systems of PDEs efficiently.

The function $f_1(x, t)$, the applied force on the first beam, is predicted in the second experiment. As illustrated in Fig. 15, the data for transverse displacement and cross-sectional rotation are provided at six different locations with 5000 data points at each location.

For the third experiment, the data provided for learning the unknown function $f_1(x, t)$ are provided with 10% and 20% Gaussian noise, and the corresponding performance in learning the function is shown in Table V. Even with 10% and 20% noise, the relative error percent between analytic and predicted force is lower, as seen in Table V. Fig. 13 shows the force prediction along the beam when rotation and deflection observations are available at five points. The results demonstrate that the PINN is more precise in its predictions when the data are free from noise compared with when they are noisy. Despite the presence of noise in the data, the absolute error remains within the magnitude of 10^{-2} , which

TABLE V
TIMOSHENKO DOUBLE-BEAM INVERSE PROBLEM: NOISE VERSUS \mathcal{R}

Noise percent (%)	Relative error percent (%)
0	4.3271×10^{-2}
10	4.8688×10^{-2}
20	1.1123×10^{-1}

is comparable to the error observed when data are not noisy. To be more precise, Fig. 13 shows the absolute difference error of the PINN-predicted force and exact force at $t = 0.5$ with 0% and 20% noise. Even with 20% noise, the unknown force is learned with less than 1% error over the entire space–time domain, demonstrating that PINN is a very accurate and robust approach.

The minimum number of data points required to estimate the model parameters depends on several factors, such as the complexity of the physics, the number of physical parameters in the model, and the quality of the data. More data points and more complex physics require more neural network capacity, resulting in a larger neural network with more hyperparameters. In practice, more data points lead to overfitting. The minimum training data points required for a PINN framework are determined empirically by gradually increasing the number of training points until the model's performance is satisfactory.

Finally, a sensitivity analysis is carried out to examine the influence of input variables, specifically the displacement and rotation, on the output variable, which is the force. The analysis involves adding 20% Gaussian noise to the displacement data while no noise is added to the rotation data. The resulting mean accuracy of the force is 0.14313413.

In contrast, when 20% noise is introduced to the rotation data with the displacement data remaining unaltered, the mean accuracy of the force is 0.204627. The results of this analysis show that the force is more sensitive to rotations than the displacement data.

V. CONCLUSION

The design and maintenance of complex structural systems are challenging due to the multiscale interaction of their components. It is desirable to predict the behavior of these complex systems by solving the governing model of interest. Recently, PINNs have emerged as a viable method for simulating PDEs. In this work, we propose using the PINN algorithm with the nondimensionalization step aiding in the learning procedure for complex beam systems. The PINN framework successfully solves the forward and inverse problems for nondimensional single- and double-beam systems. Based on the numerical experiments, the following conclusions are drawn.

First, the relative percent error in computing the beam displacement does not increase with increasing model complexity when solving the forward problem. In fact, for both Euler–Bernoulli and Timoshenko theories, the error decreases by an order of magnitude for double-beam systems compared with single-beam systems. In addition, the error in computing the bending rotation is comparable for single and double Timoshenko beam systems. This nonincrease in error as the model complexity increases suggests that the PINN framework is appropriate for simulating large-scale systems with multiple connected components.

Second, it is demonstrated that PINNs precisely discover the unknown force function and model parameters through their inverse problem-solving capability. The proposed algorithm successfully learns the model parameter with less than 3% error for the single Timoshenko beam. In addition, for the double-beam Timoshenko system, the unknown function is approximated on the whole space–time domain with less than 0.05% error, demonstrating the algorithm’s effectiveness for solving inverse problems.

Third, physical quantities such as velocity, acceleration, and bending moment characterize the system’s behavior. Even though the derived quantities are not directly trained in the neural network, they are approximated with less than $2e - 2\%$ error for the Euler–Bernoulli double-beam system.

Fourth, the algorithm’s ability to use fewer training points in forward problems and to accommodate noisy data in inverse problems is exploited. The obtained results show that even with 1600 training points, the double Timoshenko beam displacement is predicted on the entire space–time domain with less than $5e - 3\%$ error. In the case of the inverse problem, the force function is discovered with less than 0.2% error even when the data used in the learning procedure contain 20% Gaussian noise. These findings imply that the algorithm is accurate and robust under the tested noise levels.

To summarize, PINNs enable the simulation of complex structural systems with multiple interacting components efficiently, accurately, and robustly. In the future, this approach could be extended to estimate displacements for various input forces and mechanical vibration modes and incorporate robust

methods to account for stochasticities. In addition, future works on PINNs could be focused on reducing the computational cost and developing methodologies to augment their generalizability, thereby expanding the applicability of PINNs beyond the training domain.

ACKNOWLEDGMENT

The authors would like to express their appreciation to the anonymous reviewers and editors for their valuable comments and feedback, which have significantly improved the quality of this work. They extend their appreciation to Prof. Siddhartha Mishra (ETH Zürich) for his insightful suggestions in comparisons to numerical methods.

REFERENCES

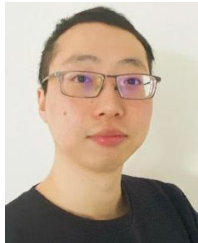
- [1] K. A. Kobbacy and D. P. Murthy, *Complex System Maintenance Handbook*. Cham, Switzerland: Springer, 2008.
- [2] M. N. Chatzis and G. Deodatis, “Modeling of very large interacting multiple-beam systems with application to suspension bridge cables,” *J. Struct. Eng.*, vol. 139, no. 9, pp. 1541–1554, Sep. 2013.
- [3] Z. Liu, Y. Song, Y. Han, H. Wang, J. Zhang, and Z. Han, “Advances of research on high-speed railway catenary,” *J. Mod. Transp.*, vol. 26, no. 1, pp. 1–23, Mar. 2018.
- [4] C. Foias, O. Manley, R. Rosa, and R. Temam, *Navier-Stokes Equations and Turbulence*, vol. 83. Cambridge, U.K.: Cambridge Univ. Press, 2001.
- [5] D. Lucor, A. Agrawal, and A. Sargent, “Simple computational strategies for more effective physics-informed neural networks modeling of turbulent natural convection,” *J. Comput. Phys.*, vol. 456, May 2022, Art. no. 111022.
- [6] V. Stojanović and P. Kozić, *Vibrations and Stability of Complex Beam Systems*. Cham, Switzerland: Springer, 2015.
- [7] J. Mercieca and V. Kadiramanathan, “Estimation and identification of spatio-temporal models with applications in engineering, healthcare and social science,” *Annu. Rev. Control*, vol. 42, pp. 285–298, Jan. 2016.
- [8] H.-X. Li and C. Qi, “Modeling of distributed parameter systems for applications—A synthesized review from time–space separation,” *J. Process Control*, vol. 20, no. 8, pp. 891–901, Sep. 2010.
- [9] T. Kapoor, H. Wang, A. Núñez, and R. Dollevoet, “Predicting traction return current in electric railway systems through physics-informed neural networks,” in *Proc. IEEE Symp. Ser. Comput. Intell. (SSCI)*, Dec. 2022, pp. 1460–1468.
- [10] A. Chandra, M. Curti, K. Tiels, E. A. Lomonova, and D. M. Tartakovsky, “Physics-informed neural networks for modelling anisotropic and bi-anisotropic electromagnetic constitutive laws through indirect data,” in *Proc. IEEE Symp. Ser. Comput. Intell. (SSCI)*, Dec. 2022, pp. 1451–1459.
- [11] L. Yuan, Y.-Q. Ni, X.-Y. Deng, and S. Hao, “A-PINN: Auxiliary physics informed neural networks for forward and inverse problems of nonlinear integro-differential equations,” *J. Comput. Phys.*, vol. 462, Aug. 2022, Art. no. 111260.
- [12] A. Fallah and M. M. Aghdam, “Physics-informed neural network for bending and free vibration analysis of three-dimensional functionally graded porous beam resting on elastic foundation,” *Eng. Comput.*, vol. 2023, pp. 1–18, Mar. 2023.
- [13] T. Kapoor, H. Wang, A. Núñez, and R. Dollevoet, “Physics-informed machine learning for moving load problems,” 2023, *arXiv:2304.00369*.
- [14] M. G. Kapteyn, J. V. R. Pretorius, and K. E. Willcox, “A probabilistic graphical model foundation for enabling predictive digital twins at scale,” *Nature Comput. Sci.*, vol. 1, no. 5, pp. 337–347, May 2021.
- [15] H. Wang, A. Núñez, Z. Liu, D. Zhang, and R. Dollevoet, “A Bayesian network approach for condition monitoring of high-speed railway catenaries,” *IEEE Trans. Intell. Transp. Syst.*, vol. 21, no. 10, pp. 4037–4051, Oct. 2020.
- [16] F.-G. Yuan, S. A. Zargar, Q. Chen, and S. Wang, “Machine learning for structural health monitoring: Challenges and opportunities,” *Proc. SPIE*, vol. 11379, Jan. 2020, Art. no. 1137903.
- [17] J. Borggaard and J. Burns, “A PDE sensitivity equation method for optimal aerodynamic design,” *J. Comput. Phys.*, vol. 136, no. 2, pp. 366–384, Sep. 1997.

- [62] Y. Yu, H. Yao, and Y. Liu, "Structural dynamics simulation using a novel physics-guided machine learning method," *Eng. Appl. Artif. Intell.*, vol. 96, Nov. 2020, Art. no. 103947.
- [63] Z. Zhang and C. Sun, "Structural damage identification via physics-guided machine learning: A methodology integrating pattern recognition with finite element model updating," *Struct. Health Monitor.*, vol. 20, no. 4, pp. 1675–1688, Jul. 2021.
- [64] A. Daw, A. Karpatne, W. Watkins, J. Read, and V. Kumar, "Physics-guided neural networks (PGNN): An application in lake temperature modeling," 2017, *arXiv:1710.11431*.
- [65] A. Khandelwal et al., "Physics guided machine learning methods for hydrology," 2020, *arXiv:2012.02854*.
- [66] J. Yu, L. Lu, X. Meng, and G. E. Karniadakis, "Gradient-enhanced physics-informed neural networks for forward and inverse PDE problems," *Comput. Methods Appl. Mech. Eng.*, vol. 393, Apr. 2022, Art. no. 114823.
- [67] Y. Miao and H. Li, "GPINN: Physics-informed neural network with graph embedding," 2023, *arXiv:2306.09792*.
- [68] J. Bettencourt, M. J. Johnson, and D. Duvenaud, "Taylor-mode automatic differentiation for higher-order derivatives in JAX," in *Proc. Program Transformation ML Workshop NeurIPS*, 2019, pp. 1–14.



Taniya Kapoor received the M.Sc. degree in applied mathematics from South Asian University, New Delhi, India, in 2019, and the M.Sc. degree in scientific computing from Université de Lille, Lille, France, in 2021. She is currently pursuing the Ph.D. degree with the Department of Engineering Structures, Delft University of Technology, Delft, The Netherlands.

Before this, she was an Intern in the seminar of applied mathematics with ETH Zürich, Zürich, Switzerland. She currently works on scientific machine learning with applications in engineering structures.



Hongrui Wang (Member, IEEE) received the Ph.D. degree from the Section of Railway Engineering, Delft University of Technology, Delft, The Netherlands, in 2019.

He was a Post-Doctoral Researcher with the Delft University of Technology, until November 2020, where he is currently an Assistant Professor with the Department of Engineering Structures. His research interests include signal processing, artificial intelligence, and their applications in structural health monitoring and digital modeling and the design of railway infrastructures.

Dr. Wang is an Associate Editor of the IEEE TRANSACTIONS ON INSTRUMENTATION AND MEASUREMENT.



Alfredo Núñez (Senior Member, IEEE) received the Ph.D. degree in electrical engineering from the University of Chile, Santiago, Chile, in 2010.

He was a Post-Doctoral Researcher with the Delft Center for Systems and Control, Delft University of Technology, Delft, The Netherlands, where he is currently an Associate Professor in the field of data-based maintenance for railway infrastructure with the Section of Railway Engineering, Department of Engineering Structures. He has authored or coauthored more than a hundred international journals and international conference papers. His current research interests include railway infrastructures, intelligent conditioning monitoring and maintenance of engineering structures, computational intelligence, big data, risk analysis, and optimization.

Dr. Núñez is on the editorial board of IEEE TRANSACTIONS ON INTELLIGENT TRANSPORTATION SYSTEMS and *Applied Soft Computing*.



Rolf Dollevoet received the M.Sc. degree in mechanical engineering from the Eindhoven University of Technology, Eindhoven, The Netherlands, in 2003, and the Ph.D. degree in rail research on rolling contact fatigue from the University of Twente, Enschede, The Netherlands, in 2010.

Since 2003, he has been with the Railway Sector, ProRail, Utrecht, The Netherlands. Since 2012, he has been appointed as a part-time Professor with the Section of Railway Engineering, Delft University of Technology, Delft, The Netherlands. He was also

a Railway System Expert with ProRail, where he was responsible for all the scientific research and innovation with the Civil Engineering Division, ProRail Asset Management.

Dr. Dollevoet was a recipient of the Jan van Stappen Spoorprijs 2010 Award (a yearly prize for contributions to the travel quality and service for passengers in The Netherlands) from the railway sector for his Ph.D. research and its huge potential to reduce track maintenance costs.