Neural Differential Equation-Based Two-Stage Approach for Generalization of Beam Dynamics

Kapoor, Taniya; Wang, Hongrui; Stamou, Anastasios; Sayed, Kareem El; Nunez, Alfredo; Tartakovsky, Daniel M.; Dollevoet, Rolf

DOI
10.1109/TII.2024.3507213

Publication date
2024

Document Version
Final published version

Published in
IEEE Transactions on Industrial Informatics

Neural Differential Equation-Based Two-Stage Approach for Generalization of Beam Dynamics

Taniya Kapoor, *Member, IEEE*, Hongrui Wang, *Senior Member, IEEE*, Anastasios Stamou, Kareem El Sayed, Alfredo Núñez, *Senior Member, IEEE*, Daniel M. Tartakovsky, and Rolf Dollevoet

***Abstract*—Computer-aided simulations are routinely used to predict a prototype's performance. High-fidelity physics-based simulators might be computationally expensive for design and optimization, spurring the development of cheap deep-learning surrogates. The resulting surrogates often struggle to generalize and predict novel scenarios beyond their training domain. We propose a two-stage methodology addressing the challenge of generalization. It employs physics-based simulators, supplemented with ordinary differential equations integrated into the recurrent architecture, to learn the intrinsic dynamics. The proposed approach captures the inherent causality and generalizes the dynamics irrespective of a data source. The presented numerical experiments encompass five fundamental structural engineering scenarios, including beams on Winkler foundations based on Euler–Bernoulli and Timoshenko theories, beams under moving loads, and catenary-pantograph interactions in railways. The proposed methodology outperforms conventional recurrent methods and remains invariant to data sources, showcasing its efficacy. Numerical experiments highlight its prospects for design optimization, predictive maintenance, and enhancing safety measures.**

***Index Terms*—Catenary-pantograph, causality, finite element, generalization, moving load, neural differential equations, physics-informed neural networks (PINNs).**

I. Introduction

COMPUTER-AIDED dynamic simulations, which encompass the concept of digital twins, are pivotal across engineering industries [1], including structural, railway, automotive, manufacturing, and aerospace, among others. These simulations offer a cost-effective and efficient alternative to physical prototyping, saving time and resources. Simulating complex engineering systems and subsystems aids in comprehensive testing and validation and assists in analyzing and optimizing their performance and designs [2].

Several methods, including numerical solvers and deep learning, are utilized to simulate the dynamics of engineering structures. For instance, the finite element method (FEM) serves as a backbone for engineering software and is widely used for modeling, designing, and optimizing structural dynamics [3]. Deep learning-based methods are employed as surrogates to simulate the underlying dynamics [4]. One such deep learning-based approach is physics-informed machine learning (PIML), which integrates physical principles into neural network architecture [5]. These simulators collectively provide robust tools for realistic and efficient simulations of engineering systems.

Industrial scenarios such as design optimization necessitate numerous simulations with varying materials and conditions influencing the dynamics. Numerical solvers require repetitive iterations for each parameter change, increasing computational cost [6]. Each iteration is particularly challenging for long-time integration problems, where the long-term behavior of the engineering systems needs to be analyzed. In addition, as the number of degrees of freedom increases, the number of simulations escalates exponentially, rendering the process laborious and time-consuming. Similarly, in deep learning, predictions outside the training domain are challenging for engineering simulations [7], requiring multiple model training for varying parameter values. Thus, long-time predictions and incorporating parameter variations exacerbate the simulation challenge across numerical and deep learning methodologies.

Current engineering dynamic simulators are matured, yet predictions outside the training domain, termed extrapolation or generalization, remain challenging [8]. An ideal simulator should provide accurate predictions inside the training domain and reasonable accuracy for generalization scenarios. Mitigating the problem of generalization necessitates a method that can predict out-of-domain and complement the current dynamic solvers. A potential approach addressing this challenge involves leveraging the current simulation strategies and coupling them with a method to provide reliable generalization predictions, reducing the problem to a two-stage strategy.

Received 10 September 2024; accepted 19 November 2024. This work was supported in part by the ProRail and Europe's Rail Flagship Project IAM4RAIL —Holistic and Integrated Asset Management for Europe's RAIL System, in part by the European Union, and in part by the European Union's Horizon Europe research and innovation programme under Grant 101101966. Paper no. TII-24-4700. *(Anastasios Stamou and Kareem El Sayed contributed equally to this work.) (Corresponding author: Taniya Kapoor.)*

Taniya Kapoor, Hongrui Wang, Alfredo Núñez, and Rolf Dollevoet are with the Department of Engineering Structures, Delft University of Technology, 2628CN Delft, The Netherlands (e-mail: t.kapoor@tudelft.nl).

Anastasios Stamou and Kareem El Sayed are with the Department of Hydraulic Engineering, Delft University of Technology, 2628CN Delft, The Netherlands.

Daniel M. Tartakovsky is with the Department of Energy Science and Engineering, Stanford University, Stanford, CA 94305 USA.

Color versions of one or more figures in this article are available at https://doi.org/10.1109/TII.2024.3507213.

Digital Object Identifier 10.1109/TII.2024.3507213

The first stage involves the traditional solver simulating the problem precisely in the training domain. The second stage utilizes the predictions from the first stage to generalize beyond the training domain. For example, for a moving load problem like a train on the railway track, the quantity of interest is track deflection under different loadings. For certain loadings, deflections could be simulated using a traditional solver, such as FEM, and deflections for unseen loadings could be predicted in a second stage utilizing the FEM deflections.

For the second stage, prior knowledge available is the simulated data in the training domain, and utilizing it to predict in the generalized domain is nontrivial [9]. A potential way is to leverage the causality of the underlying dynamics from the trained domain and use them in an autoregressive approach to predict the quantities of interest in the untrained domain. Learning the inherent causality would allow reasonable generalization by capturing the underlying dynamics, structure, and symmetry of the problem.

Attention-based and recurrent neural architectures are viable for sequence modeling and prediction tasks. However, attention-based models require substantial data and disregard the underlying sequential causality inherent in the physical simulations. Recurrent neural networks (RNNs) are well-known to model time-series data. However, long sequences pose exploding and vanishing gradients problem (EVGP), which can even be observed with advanced gated architectures [10] like the long short-term memory (LSTM) [11] and gated recurrent unit (GRU) [12]. The challenge of EVGP could be mitigated by employing ordinary differential equations (ODEs) to update the hidden states of the recurrent architecture, facilitating efficient dynamic learning. Recent methods like coupled-oscillatory recurrent neural network (CoRNN) [13] and long expressive memory (LEM) [14] do not exhibit the EVGP and perform well for several sequential artificial intelligence (AI) tasks. This work proposes to merge dynamic simulations with recurrent architectures employing ODEs to update the hidden states, namely, CoRNN and LEM. The proposed two-stage methodology exploits the causality and learns temporal and parametric dependencies, enhancing the accuracy of generalization predictions of engineering dynamic simulations.

Five fundamental problems within structural engineering are examined to validate the proposed methodology. Simulation of beam dynamics under dynamic loading conditions is crucial for precise structural analysis and design [15]. Beams on Winkler foundations are crucial in civil engineering, providing stability by distributing loads, minimizing buckling [16], and resisting against vibrations [17]. Concrete beams are used extensively in the construction industry [18]. They are commonly used in applications like railway tracks, pile foundations, and composite elastomers, where understanding their behavior is essential for maintaining structural integrity and optimizing designs.

The first two cases are based on well-known Euler–Bernoulli and Timoshenko beam theories on the Winkler foundation. Both beam theories are fundamental in structural mechanics. Euler–Bernoulli beam theory models the behavior of slender, linear beams under various loading conditions, assuming the plane sections of the beam to remain plane and perpendicular to the cross-section where no longitudinal stresses or strains occur. The Euler–Bernoulli theory also assumes the shear deformations and rotational effects to be negligible [19]. This simplification allows the calculation of stresses and deflections in beams under static and dynamic loads by reducing the three-dimensional problem of beam bending to a one-dimensional model [20]. The governing equation is derived from equilibrium conditions (using Newton's laws or Lagrangian mechanics), material constitutive laws, and geometric properties, leading to a fourth-order differential equation.

The Timoshenko beam theory builds on the Euler–Bernoulli theory by including the effects of shear force. Unlike the Euler–Bernoulli theory, which assumes that beam cross-sections remain flat and perpendicular to the neutral axis after deformation, Timoshenko theory accounts for additional angular rotation due to shear strain [21], affecting the displacement by shear and bending deformations. The theory introduces an extra degree of freedom, represented by the angular rotation. Therefore, the key quantities of interest are the displacement and the angular rotation. These quantities of interest are computed by solving the governing equations, consisting of two coupled second-order partial differential equations.

The third case is the moving load problem, studying the deflection of the beam under different loadings. Understanding beam behavior under moving loads is essential for structural health monitoring and maintaining infrastructure integrity. For the fourth experiment, real-world catenary-pantograph interactions in railway systems are studied. Comprehending the vertical displacement of the catenary contact wire under various train speeds is critical for railway infrastructure safety and design. Varying speeds induce dynamic loads affecting the stability of the catenary. Accurate contact wire uplift predictions aid engineers in estimating the undesired condition of the catenary, which directly influences the power supply safety and stability of the traction power system, averting potential disruptions and accidents. The final experiment deals with estimating the unknown force applied on a system of Timoshenko beams by solving an inverse problem.

The main contributions of this work can be summarized as follows.

1) This work introduces a two-stage approach for generalizing engineering dynamic simulations. First, dynamics are simulated using a state-of-the-art simulator preferred for the application, followed by classical mathematical models (ODEs) infused in neural architecture to generalize the dynamics.
2) AI for engineering and industry trained on specific resolutions often struggles to generalize well to unseen resolutions. The proposed work tackles this issue through a resolution invariant pipeline where both stages can process data at different resolutions.
3) The proposed workflow efficiently generalizes the dynamics in the time domain for spatio-temporal engineering systems without using any data from the untrained time domain.
4) Furthermore, for spatio-temporal parametric systems, the workflow eliminates tedious remeshing and resimulation in computer-aided simulation software for novel parameters belonging to the parameter space.

5) The performance of the proposed approach is evaluated on four different dynamic simulation problems in structural engineering, including real-world catenary-pantograph systems, demonstrating superior performance compared to traditional recurrent architectures.

The rest of this article is organized as follows. Section II details the related works to this article. Section III provides an overview of the problem statement. Section IV presents the proposed two-stage methodology in detail to enhance the generalization. Section V presents the performed numerical experiments to validate the proposed methodology. Finally, Section VI concludes this article.

II. Related Work

This work focuses on generalizing spatio-temporal parametric engineering dynamics in temporal and parametric space. Industrial applications often necessitate simulating spatio-temporal parametric systems as discussed in the works of [22], [23], [24]. In particular, the authors in [23] and [24] discussed the challenges and importance of spatio-temporal simulations in industry. In addition to sharing the core idea of simulating spatio-temporal parametric dynamics, this work proposes a method to generalize the dynamics in untrained domains, making it more applicable to real-world situations where systems may need to be deployed in unseen situations for which data collection may not even be possible at priori [9], [25].

In other works, generalization of deep learning-based methods for industrial tasks has been explored in [25], [26], and [27], among others, aligning with the motivation of this work. However, this work proposes a generic two-step approach enabling the investigation through the state-of-the-art simulator and coupling it with ODE-based recurrent architecture to capture the intrinsic dynamics. This approach democratizes the challenge of generalization and hence could be used by a larger industrial and engineering community.

The second stage in the proposed methodology employs ODEs in the recurrent neural architecture to enhance generalization. Pioneered by the seminal works of [28], neural ODEs have been explored for cooling-system prediction [29], process quality evaluation [30], and remaining useful life estimation [31]. However, this work posits that neural differential equations-based architectures capture the causality better than the gated architectures, improving the generalization ability of the learned model and making it distinct from the aforementioned works.

III. Problem Statement

This section presents an abstract formulation of the generalization problem. In general, spatio-temporal parametric systems are governed by an operator $\mathcal{D}[u(x,t;\mu)] = 0$, where, $(x,t) \in D \times T$. Here, $D \subset \mathbb{R}$ and $T \subset \mathbb{R}$ represent the spatial and temporal domain, respectively. In addition, μ represents the parameters in the parametric space $M \subset \mathbb{R}$, and $u \in \mathcal{U} \subset \mathbb{R}^d$ is the quantity of interest in a d-dimensional space. The operator $\mathcal{D}$ could be explicitly known, for instance, the Euler–Bernoulli and Timoshenko partial differential equations (PDEs) governing the beam dynamics, or could be unknown as in the case of real-world catenary-pantograph system.

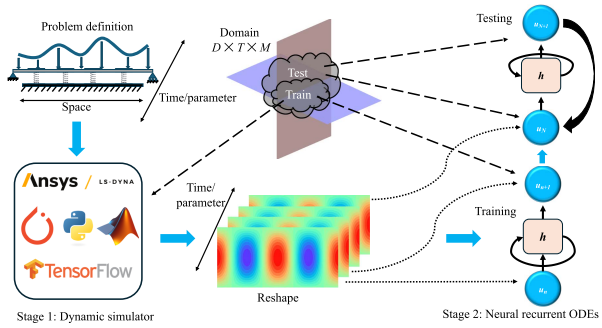

Fig. 1. Schematic of the proposed methodology: Two-stage methodology starts with defining the problem and training and testing domain in which generalization is sought. A dynamic simulator is then utilized in the first stage to simulate the problem in the training domain, whose solutions are reshaped to feed into the second stage comprising recurrent neural ODEs. Neural ODEs are trained on the simulator predictions. The trained neural ODEs are used to predict in the testing domain.

The proposed approach transcends traditional limitations and seamlessly applies across temporal and parametric domains. Hence, to formalize the generalization problem, the domain Ω hereafter invariantly represents the domain T or M. The entire spatio-temporal parametric space is divided into two disjoint sets $\mathcal{X}_1$ and $\mathcal{X}_2$, where $\mathcal{X}_1 := D \times \Omega$ and $\mathcal{X}_2 := D \times \Omega'$. Here, Ω' is the generalized temporal or parametric domain with $\inf(\Omega') \geq \sup(\Omega)$, which implies that testing is performed for $\omega' \in \Omega' \geq \omega \in \Omega$. Concretely, the temporal or parametric space is divided into two segments: $\Omega := [0, \Omega_{\text{train}}]$ and $\Omega' := (\Omega_{\text{train}}, \Omega_{\text{test}}]$, where $\Omega_{\text{test}} > \Omega_{\text{train}} > 0$. The numerical or deep learning-based simulator is used in the first stage to simulate the dynamics in $\mathcal{X}_1$, and the problem reduces to predicting the dynamics in the testing domain $\mathcal{X}_2$ in the second stage. Concretely, the objective is to make predictions beyond $\mathcal{X}_1$, i.e., on $\mathcal{X}_2$, and to assess how well the trained models could be generalized through the two-stage training strategy.

IV. Methodology

The proposed two-stage methodology merges dynamic simulators and neural ODE-based methods to generalize the dynamics. The key steps of the proposed two-stage methodology are presented in Algorithm 1 and Fig. 1. The first stage entails simulating the engineering dynamics through a preferable simulator tailored for the application, providing flexibility to the workflow. Simulators are computational tools in form of software or modules that act as surrogates for the real-world systems through mathematical models and algorithms, allowing engineers and scientists to analyze and predict the performance of complex systems under various conditions without the need for physical prototypes.

However, numerical methods like FEM or a deep learning approach like physics-informed neural networks (PINNs) [32] simulate the problem in a confined domain. To mitigate the challenge of predictions in a larger domain, the second stage in the proposed methodology employs neural ODEs to capture

the intrinsic dynamics from the data generated in the first stage and generalize the dynamics to larger domains, making them further applicable and advantageous for real-world systems. The following sections describe both the stages of the proposed methodology in detail.

A. First Stage—Numerical/PIML Simulator

This article employs two dynamic simulators to exemplify that the proposed methodology remains invariant to the simulator employed in the first stage. The first simulator employed is an advanced version of a PINN that enforces causality in the learning algorithm termed causal PINN [6], [33]. Causal PINN predictions of beam dynamics are utilized to test temporal generalizations. The simulator causal PINN could be described by considering an abstract PDE with implicit initial and boundary conditions defined by

$$\mathcal{K}(x,t) := \mathcal{D}[u](x,t) - f(x,t) \quad \forall (x,t) \in D \times T \subset \mathbb{R}^{\mathrm{d}} \times \mathbb{R} \tag{1}$$

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

Causal PINN is based on a feedforward neural network, where the inputs (x,t) map to output (u) through an iterative composition of hidden layers. To train causal PINN, the loss function containing the physical model of a PDE is minimized along with initial and boundary conditions. The loss function is formulated such that the network first minimizes the loss corresponding to lower times before resolving the solutions at higher times. Mathematically, the loss function is defined as

$$J = \underset{\theta}{\mathrm{Min}} \frac{1}{N} \left[L(t_1) + \sum_{i=2}^{N} e^{-\eta \sum_{k=1}^{i-1} L(t_k)} L(t_i) \right] \tag{2}$$

where the loss components are defined as

$$L(t_n) = \frac{1}{N} \sum_{n=1}^{N} ||\mathcal{K}(x,t_n)||^2 \tag{3}$$

where (x,t_n) represents the training tuple for each time step n. The total number of training points inside the computational domain is denoted by N, and η is the causality hyperparameter, which depends on the complexity of the problem. Minimizing the loss function (2) using a suitable optimization algorithm provides optimal parameters θ. Following the causal PINN training, its testing is conducted on k_t uniform time steps in T and k_x uniform locations in D, making a total of $k_t \cdot k_x$ testing points within the training domain.

The second simulator employed is a FEM. The physical models, which are the PDEs governing the beam dynamics, are available for causal PINN. Several systems in engineering and industry do not possess a known operator and are rather unknown. The efficacy of the proposed methodology is demonstrated for such scenarios by taking the solutions from FEM as the output of the first stage, considering the governing model and solution strategy as a black box. FEM predictions are utilized to test parametric generalizations. Specifically, in one of the numerical experiments, deflection profiles of beams with different loading are taken as the output of the first stage. Here, the parameter is the load, and the quantity of interest is the deflection profile. Another numerical experiment studies the real-world uplift of catenary contact wire in railway systems, depending on the different speeds of the train. The solutions obtained from the causal PINN and FEM are reshaped for further use in the second stage.

Algorithm 1: Proposed Two-Stage Methodology for Generalizing Dynamic Simulators.

Input: Problem and domain, $(x,t;\mu) \in D \times T \times M$.
Output: Generalized dynamic predictions, $u(x,t;\mu) \in \mathcal{X}_2$.
Stage 1: simulation with causal PINN or FEM:
if Causal PINN **then**
  Train causal PINN.
  Test causal PINN for $u(x,t;\mu) \in \mathcal{X}_1$ at k_t time steps.
else
  Collect spatio-temporal data for FEM in $\mathcal{X}_1$.
end if
Stage 2: Training neural differential equations:
Integrate ODEs into the recurrent architecture for updating hidden states.
Utilize predictions from Stage 1 to train LEM or CoRNN in $\mathcal{X}_1$.
Loop for generalization:
for $i = (k_t + 1)$ to $(k_t + m)$ **do**
  Input condition at time step $i - 1$.
  Use trained model from Stage 2 to predict dynamics at parametric or time step i in $\mathcal{X}_2$.
end for
return $u(x,t;\mu) \in \mathcal{X}_2$

B. Second Stage—Neural ODE

Generalization is an open challenge for both methods, causal PINN and FEM. The second stage aims to mitigate this challenge by employing neural ODE-based architecture to capture temporal and parametric dependency. Neural ODEs model the evolution of hidden states over time using differential equations, leveraging the longstanding potential of ODEs in handling complex temporal dynamics. In particular, this work employs two neural ODE methods, CoRNN [13] and LEM [14], processing the outputs from the first stage as sequential data and predicting the dynamics outside the training domain. CoRNN employs second-order ODE to model the dynamics of hidden states preserving long-term dependencies and mitigating EVGP. By incorporating damping factors and oscillatory components in the ODE, CoRNN enforces the computed hidden states to remain within bounds, enhancing the training stability. LEM employs a system of first-order coupled differential equations to update the hidden states. The coupled equations allow LEM to maintain a robust representation of sequential data, addressing EVGP typical with RNN-based methods. In the following, the ODEs used for CoRNN and LEM are presented in detail.

1) Coupled-Oscillatory Recurrent Neural Network: CoRNN updates the hidden states, $\boldsymbol{y} = \boldsymbol{y}(\omega) \in \mathbb{R}^m$ by solving the second-order ODE

$$\boldsymbol{y}'' = \sigma\left(\boldsymbol{W}\boldsymbol{y} + \mathcal{W}\boldsymbol{y}' + \boldsymbol{V}\boldsymbol{u} + \boldsymbol{b}\right) - \gamma\boldsymbol{y} - \epsilon\boldsymbol{y}'. \tag{4}$$

Here, $\boldsymbol{y}$ represents the hidden state, $\boldsymbol{y}'$ and $\boldsymbol{y}''$ denote the first and second derivative of the hidden state. The activation function is $\sigma(u) = \tanh(u)$. The weight tensors for the hidden state and its first derivative are represented by $\boldsymbol{W}, \mathcal{W} \in \mathbb{R}^{m\times m}$ and the bias term is $\boldsymbol{b} \in \mathbb{R}^m$. The weight tensor for the input ($\boldsymbol{u} \in \mathbb{R}^{k_x}$) is $\boldsymbol{V} \in \mathbb{R}^{m\times k_x}$. The terms $\gamma, \epsilon > 0$ are the hyperparameters representing oscillation frequency and damping.

The motivation to employ ODE (4) to update the hidden states is attributed to its underlying capabilities in modeling complicated nonlinear oscillations [13]. The dynamics of the ODE could be analyzed for a simplified case by setting $k_x = m = 1$ in (4) with an identity activation function $\sigma(u) = u$. Considering the terms $\boldsymbol{W} = \mathcal{W} = \boldsymbol{V} = \boldsymbol{b} = \epsilon = 0$, the ODE reduces to, $\boldsymbol{y}'' + \gamma\boldsymbol{y} = 0$, modeling the well-known spring-mass simple harmonic motion with frequency γ. Including further terms in this simplified ODE, like $\epsilon > 0$, induces damping in the system. For a nonzero vector $\boldsymbol{V}$, the system experiences a driving force proportional to the input signal $\boldsymbol{u}$, where $\boldsymbol{V}$ and $\boldsymbol{b}$ modulate the influence of this force. The tensor $\boldsymbol{W}$ affects the oscillation frequency, while $\mathcal{W}$ influences the damping effect within the system. In addition, introducing the $\tanh$ activation function introduces a nonlinear dynamic response in the oscillator.

Substituting $\boldsymbol{z} = \boldsymbol{y}'(\omega) \in \mathbb{R}^m$, (4) could be transformed to the first-order system

$$\boldsymbol{y}' = \boldsymbol{z}, \quad \boldsymbol{z}' = \sigma\left(\boldsymbol{W}\boldsymbol{y} + \mathcal{W}\boldsymbol{z} + \boldsymbol{V}\boldsymbol{u} + \boldsymbol{b}\right) - \gamma\boldsymbol{y} - \epsilon\boldsymbol{z}. \tag{5}$$

Discretizing (5) using an explicit scheme with a time step $0 < \Delta t < 1$

$$\begin{aligned}\boldsymbol{y}_n &= \boldsymbol{y}_{n-1} + \Delta t\boldsymbol{z}_n\\ \boldsymbol{z}_n &= \boldsymbol{z}_{n-1} + \Delta t\sigma\left(\boldsymbol{W}\boldsymbol{y}_{n-1} + \mathcal{W}\boldsymbol{z}_{n-1} + \boldsymbol{V}\boldsymbol{u}_n + \boldsymbol{b}\right)\\ &\quad - \Delta t\gamma\boldsymbol{y}_{n-1} - \Delta t\epsilon\boldsymbol{z}_n.\end{aligned} \tag{6}$$

In the coupled-ODE system with $m > 1$, each neuron updates its hidden state by incorporating external input signals and other neurons. The diagonal components of $\boldsymbol{W}$, along with the scalar hyperparameter γ, regulate the intrinsic oscillatory frequency of individual neurons, whereas the diagonal elements of $\mathcal{W}$, together with the hyperparameter ϵ, govern the damping effects. The off-diagonal entries of these matrices serve to modulate the interaction dynamics between neurons. Further deep networks yield rich global dynamics, suggesting that such oscillator networks can achieve high expressivity, making them capable of approximating complex outputs from sequential inputs [13]. Finally, the output is computed through a learnable linear transformation, $\nu_n \in \mathbb{R}^{k_x}$ with $\nu_n = \mathcal{Q}\mathbf{y}_n$ and $\mathcal{Q} \in \mathbb{R}^{k_x\times m}$.

2) Long Expressive Memory: Akin to CoRNN, LEM uses a system of differential equations to update the hidden states. However, the system of equations to be solved in LEM is

$$\begin{aligned}\boldsymbol{y}' &= \hat{\sigma}(\boldsymbol{W}_2\boldsymbol{y} + \boldsymbol{V}_2\boldsymbol{u} + \boldsymbol{b}_2) \odot [\sigma(\boldsymbol{W}_y\boldsymbol{z} + \boldsymbol{V}_y\boldsymbol{u} + \boldsymbol{b}_y) - \boldsymbol{y}]\\ \boldsymbol{z}' &= \hat{\sigma}(\boldsymbol{W}_1\boldsymbol{y} + \boldsymbol{V}_1\boldsymbol{u} + \boldsymbol{b}_1) \odot [\sigma(\boldsymbol{W}_z\boldsymbol{y} + \boldsymbol{V}_z\boldsymbol{u} + \boldsymbol{b}_z) - \boldsymbol{z}].\end{aligned} \tag{7}$$

In addition to the previously stated learnable quantities, LEM additionally learns weight tensors $W_{1,2}, W_{y,z} \in \mathbb{R}^{m\times m}$, weight tensors for input $V_{1,2}, V_{y,z} \in \mathbb{R}^{m\times k_x}$, bias vectors $b_{1,2}$, and $b_{y,z} \in \mathbb{R}^m$. The function $\hat{\sigma}$ and $\odot$ represent the sigmoid activation function and componentwise product of vectors, respectively. $\boldsymbol{y}$ and $\boldsymbol{z}$ are hidden state vectors. The output of LEM is linearly transformed in the same way as in the case of CoRNN. A discretization of (7) using explicit Euler scheme results in

$$\begin{aligned}\boldsymbol{\Delta t}_n &= \Delta t\hat{\sigma}(\boldsymbol{W}_1\boldsymbol{y}_{n-1} + \boldsymbol{V}_1\boldsymbol{u}_n + \boldsymbol{b}_1)\\ \overline{\boldsymbol{\Delta t}}_n &= \Delta t\hat{\sigma}(\boldsymbol{W}_2\boldsymbol{y}_{n-1} + \boldsymbol{V}_2\boldsymbol{u}_n + \boldsymbol{b}_2)\\ \boldsymbol{z}_n &= (1 - \boldsymbol{\Delta t}_n) \odot \boldsymbol{z}_{n-1}\\ &\quad + \boldsymbol{\Delta t}_n \odot \sigma(\boldsymbol{W}_z\boldsymbol{y}_{n-1} + \boldsymbol{V}_z\boldsymbol{u}_n + \boldsymbol{b}_z)\\ \boldsymbol{y}_n &= (1 - \overline{\boldsymbol{\Delta t}}_n) \odot \boldsymbol{y}_{n-1}\\ &\quad + \overline{\boldsymbol{\Delta t}}_n \odot \sigma(\boldsymbol{W}_y\boldsymbol{z}_n + \boldsymbol{V}_y\boldsymbol{u}_n + \boldsymbol{b}_y).\end{aligned} \tag{8}$$

The proposed methodology utilizes the simulator solutions to extrapolate through the neural ODE methods, CoRNN, and LEM. However, a key challenge for recurrent neural architectures is EVGP. The theoretical analysis of CoRNN and LEM presented in [13] and [14], and collated in the form of propositions below, motivate employing them for extrapolating the beam dynamics. The detailed proofs of bounds can be found in [13] and [14].

Proposition 1: The hidden states of CoRNN ($\boldsymbol{y}_n$) and LEM ($\boldsymbol{y}_n$, $\boldsymbol{z}_n$) are bounded, eliminating the possibility of chaotic hidden state behavior [13], [14].

Proposition 2: Let $\mathcal{L}(\theta)$ denote the $L2$ loss function to be minimized for training CoRNN and LEM with trainable parameters $\theta \in \Theta$, where $\Theta = [\boldsymbol{W}, \mathcal{W}, \boldsymbol{V}, \boldsymbol{b}]$ for CoRNN, and $\Theta = [\boldsymbol{W}_{\boldsymbol{1,2,y,z}}, \boldsymbol{V}_{\boldsymbol{1,2,y,z}}, \boldsymbol{b}_{\boldsymbol{1,2,y,z}}]$ for LEM. Then, the gradients of the loss function with respect to the trainable parameters is bounded, i.e., $|\frac{\partial\mathcal{L}}{\partial\theta}| \leq \kappa$, where κ depends on the parameters of the considered model [13], [14].

The following section presents the numerical experiments to validate the proposed methodology.

V. Numerical Experiments

Five distinct numerical experiments concerning generalizing dynamic simulations for forward and inverse problems are presented. The complexity of the experiments ranges from fundamental beam theories for beams to real-world catenary-pantograph interactions in railway systems. The first two experiments involve simulating PDEs modeled by the Euler–Bernoulli and Timoshenko theories on the Winkler foundation, where generalization is sought in the temporal domain. The third and fourth experiments generalize in the parametric space. The third experiment is the moving load problem, which predicts the mid-point beam deflection under various loading conditions. The fourth experiment involves predicting catenary contact wire

uplift for novel train speeds, considering it as a parameter. The final experiment aims to solve an inverse problem by estimating the applied force on a system of Timoshenko beams.

A. Test Cases

1) Euler–Bernoulli and Timoshenko Beam on the Winkler Foundation: The first two experiments are the PDEs governing the Euler–Bernoulli and Timoshenko beams on the Winkler foundation, described in detail in [6]. For the dynamic simulator in the first stage, causal PINN is employed. Four hidden layers, 200 neurons, and the tanh activation function are utilized for training causal PINN. Limited memory Broyden–Fletcher–Goldfarb–Shanno (L-BFGS) optimizer is utilized with a learning rate of 0.1, with 10 000 epochs. The causality hyperparameter is 5, and $\mathcal{X}_1$ is divided into 100-time steps. The training utilized 500 initial points, 1000 boundary points, and 10 000 interior points. For both cases, the training and testing dataset is divided as $T_{\text{train}} = 0.8T_{\text{test}}$.

The objective is to make predictions beyond $\mathcal{X}_1$, i.e., on $\mathcal{X}_2$, and to evaluate the potential of the proposed method for temporal generalization. The training dataset for the second stage, LEM or CoRNN, is generated by testing causal PINN at 256 spatial locations across 160-time steps, implying $k_x = 256$ and $k_t = 160$. Finally, LEM and CoRNN testing is performed on the untrained temporal domain for 40 time steps.

2) Moving Load: This experiment validates the potential of the proposed methodology in generalizing in the parametric domain. Despite being a fundamental problem across structural engineering, simulating moving load problems within commercial finite element packages is computationally expensive [34], and generalizing it would aid the engineers in downstream predictions of deflection profiles at reduced computational cost. The moving load problem represents a train-track or catenary-pantograph interaction in railway systems. In particular, a point force moving across a simply supported beam is considered.

A finite element-based method is used to compute 100 mid-point deflection profiles of beams for varying loading ranging from 1 N to 6 N. Training and testing dataset is divided as $M_{\text{train}} = 0.8M_{\text{test}}$, i.e., the first 80 equispaced deflection profiles between the loads 1 N to 5 N are used to train the LEM and CoRNN. Specifically, the training dataset size for the proposed method represents mid-point deflection at 344 temporal locations across 80 different loadings. The beam deformations are predicted for 19 unseen equispaced loadings between 5 N to 6 N through LEM and CoRNN.

3) Catenary Contact Wire Uplift: This experiment aims to validate the method of generalizing a real-world catenary-pantograph interaction in the parametric domain. In the first stage, the interaction between the pantograph head and contact wire is modeled, and the calculation of dynamic contact wire uplift due to the contact forces is performed in a unknown sense. The method employed a validated finite element model using the absolute nodal coordinate formulation (ANCF) characterizing catenary nonlinearity. In addition, a simplified lumped mass model simulated the three critical modes of the pantograph [35], as shown in Fig. 2. For this work, only the data of catenary contact wire uplift for different speeds of train [36] serves as the output from the first stage, and the model and method details are treated as unknown.

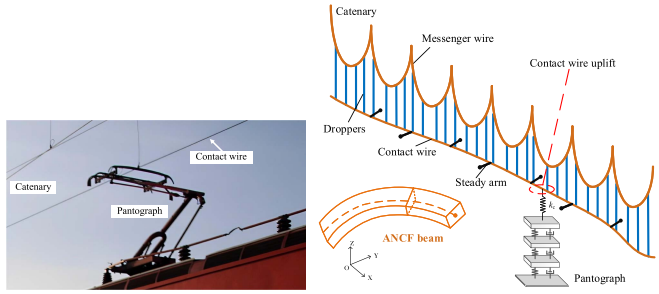

Fig. 2. Catenary-pantograph experiment. *Left:* Real-world catenary-pantograph setup in railways. Catenary-pantograph interaction simulation model. *Right:* The Catenary is modeled by ANCF beam elements (contact and messenger wires) and cable elements (droppers). The pantograph is a lumped mass model with three degrees of freedom.

The train speed spans from $50\,\text{kmh}^{-1}$ to $90\,\text{kmh}^{-1}$, incrementing in intervals of $10\,\text{kmh}^{-1}$. The dataset contains the catenary contact wire deflection at 353 spatial locations across five speeds. The recurrent networks in the second stage are trained on two specific speeds, $50\,\text{kmh}^{-1}$ and $60\,\text{kmh}^{-1}$. The trained recurrent models are used for predicting the contact wire deflection for three novel train speeds, which are $70\,\text{kmh}^{-1}$, $80\,\text{kmh}^{-1}$, and $90\,\text{kmh}^{-1}$.

4) Inverse Problem for Timoshenko Double Beam System: In addition, an experiment regarding inverse problem is carried out to showcase the potential of the proposed method in handling ill-posed problems. Inverse problems involve determining unknown parameters or functions based on known observables within a system. Such problems are ill-posed, requiring additional data at specific locations for the observables. These unknowns, often referred to as quantities of interest, include force functions, initial conditions, boundary conditions, or parameters. The two-stage method predicts quantities of interest in unseen domains. An inverse problem is solved for a Timoshenko double beam system connected by a Winkler foundation [15] to estimate the unknown force function acting on the system, given displacement profiles of the beam system.

B. Baselines, Hyperparameters, and Error Metrics

The second stage in the proposed methodology is based on recurrent neural architectures. Hence, the comparisons for all the experiments are carried out with traditional sequential architectures, RNN, LSTM, and GRU, replacing LEM and CoRNN in the second stage. In addition, comparisons with further advanced methods for modeling sequential data, such as neural ordinary differential equation (NODE) [28] and transformers [37] have been carried out. All methods employed the Adam optimizer to train the recurrent architecture. Consistent hyperparameters are chosen across different methods, and different hyperparameters are mentioned as follows.

For Euler–Bernoulli and Timoshenko beam experiments, the learning rate and hidden size are 0.0001 and 32, respectively. CoRNN and LEM-specific parameters, Δt, γ, and ϵ, are taken to be 0.05, 1, and 0.01, respectively. Specific to the transformer, the number of attention heads is 8. The model consists of

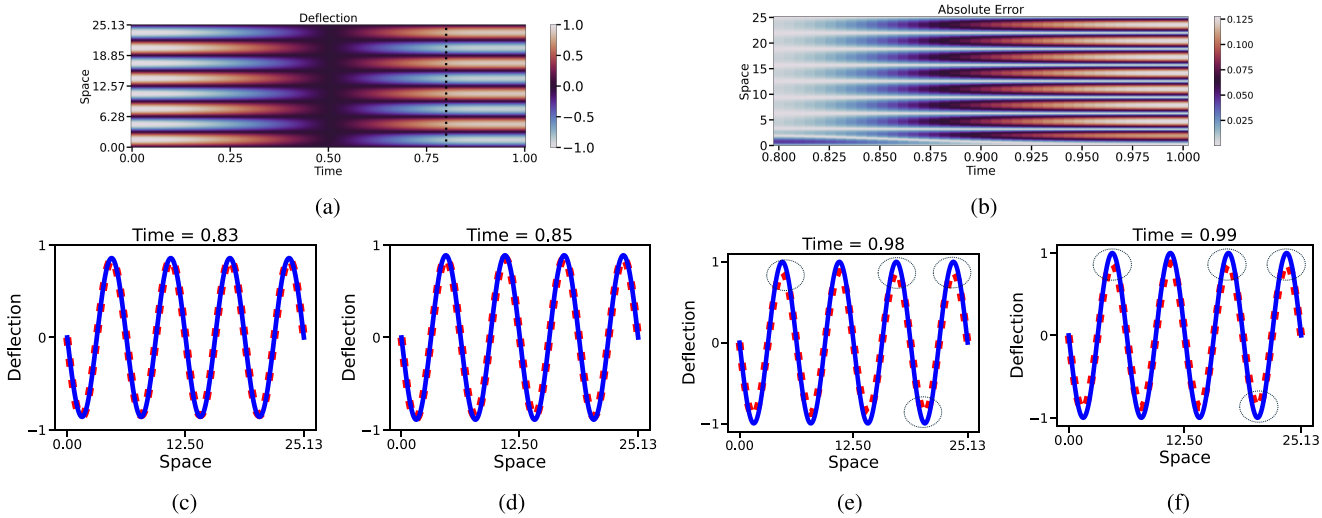

Fig. 3. Euler–Bernoulli beam on the Winkler foundation. The black dashed line separates the training and testing regions. The red dashed line indicates predictions, and the solid blue line represents the actual deflection. These representations are kept consistent with the following figures. The black dotted circles highlight the discrepancy region. (a) Deflection causal PINN-LEM. (b) Absolute error for deflection in generalized temporal domain. (c) Snapshot at $t = 0.83$. (d) Snapshot at $t = 0.85$. (e) Snapshot at $t = 0.98$. (f) Snapshot at $t = 0.99$.

6 encoder–decoder layers. The dimension of the feedforward network within the transformer is 512. For all the methods, 400 000 epochs are carried out for the Euler–Bernoulli case and 200 000 for the Timoshenko case.

For the moving load problem, the chosen hyperparameters, including the learning rate, hidden size, number of epochs, and Δt, maintain consistent values across all models: 0.00001, 160, 20 000, and 0.05, respectively. For training the transformer, the number of attention heads is 8, with three encoder–decoder layers. The dimension of the feedforward network within the transformer is 128. Similarly, for CoRNN, the hyperparameters γ and ϵ remain fixed at 1 and 0.01, respectively.

Subsequently, for the catenary-pantograph experiment, the hyperparameters for all sequential models, comprising the learning rate, hidden size, number of epochs, and Δt, are configured to 0.001, 128, 20 000, and 0.1, respectively. Analogously, in line with prior instances, the hyperparameters γ and ϵ for CoRNN are set at 1.0 and 0.01, respectively. For the transformer, the number of attention heads is 8 with six transformer layers and 512 as the dimension of the feedforward network.

Four different error metrics are used to evaluate the results. First, relative $L2$-norm of the quantity of interest $\hat{u}$, which is defined with respect to the ground truth u as $\frac{\|\hat{u}-u\|_2}{\|u\|_2}$. The second error metric is the maximum absolute error (max error) computed as $\max_{i=1}^{n} |u_i - \hat{u}_i|$. Here, $|.|$ represents the absolute value function. Here, u_i represents the ground truth at the ith data point and $\hat{u}_i$ represents the corresponding predicted value.

The third error metric is the explained variance score

$$1 - \frac{\sum_{i=1}^{n}(u_i - \hat{u}_i)^2}{\sum_{i=1}^{n}(u_i - \bar{u})^2}$$

where n is the number of data points, and $\bar{u}$ represents the mean of the ground truth. Finally, the last metric is the mean absolute error calculated by $\frac{1}{n}\sum_{i=1}^{n} |u_i - \hat{u}_i|$, where the symbols have their same meaning.

C. Results

1) Euler–Bernoulli and Timoshenko Beam on the Winkler Foundation: Figs. 3 and 4 represent the predictions obtained by LEM for the Euler–Bernoulli and Timoshenko experiments, respectively. Fig. 3 top row left depicts the deflection of the beam over unseen time, and the right of the top row shows the absolute error obtained in the deflection prediction. The second row presents the snapshots of deflection predictions at four instances of unseen time. The red dots indicate predictions, and the solid blue line represents the deflection. Table I compares the neural differential equation-based methods with the traditional sequential methods for generalization. While RNN and other sequential methods exhibit some generalization ability, LEM outperforms them across all metrics, as evidenced by Table I. In addition, the other sequential methods demonstrate less accuracy in predicting unseen domains. Thus, both Fig. 3. Table I collectively demonstrates that LEM achieves higher accuracy in predicting beam deflection on the Winkler foundation than alternative methods.

Fig. 4 top two rows depict the Timoshenko beam deflection and rotation on the Winkler foundation over time. The second row in Fig. 4 displays the absolute errors of deflection and rotation observed in the generalized time, showing a minor increase in error as time progresses. The last row presents the snapshots of deflection and rotation predictions at two different instants of time. The red dots indicate predictions, and a solid blue line represents the actual deflection simulated using causal PINN. Table I compares the proposed and other sequential methods for Timoshenko beam deflection and rotation for the unseen temporal domain. LEM outperforms them across all metrics, as seen by the results provided in Table I.

2) Moving Load: Fig. 5 illustrates the beam deflections for a moving load across a simply supported beam. The top two rows in Fig. 5 show the predicted deflections and the obtained

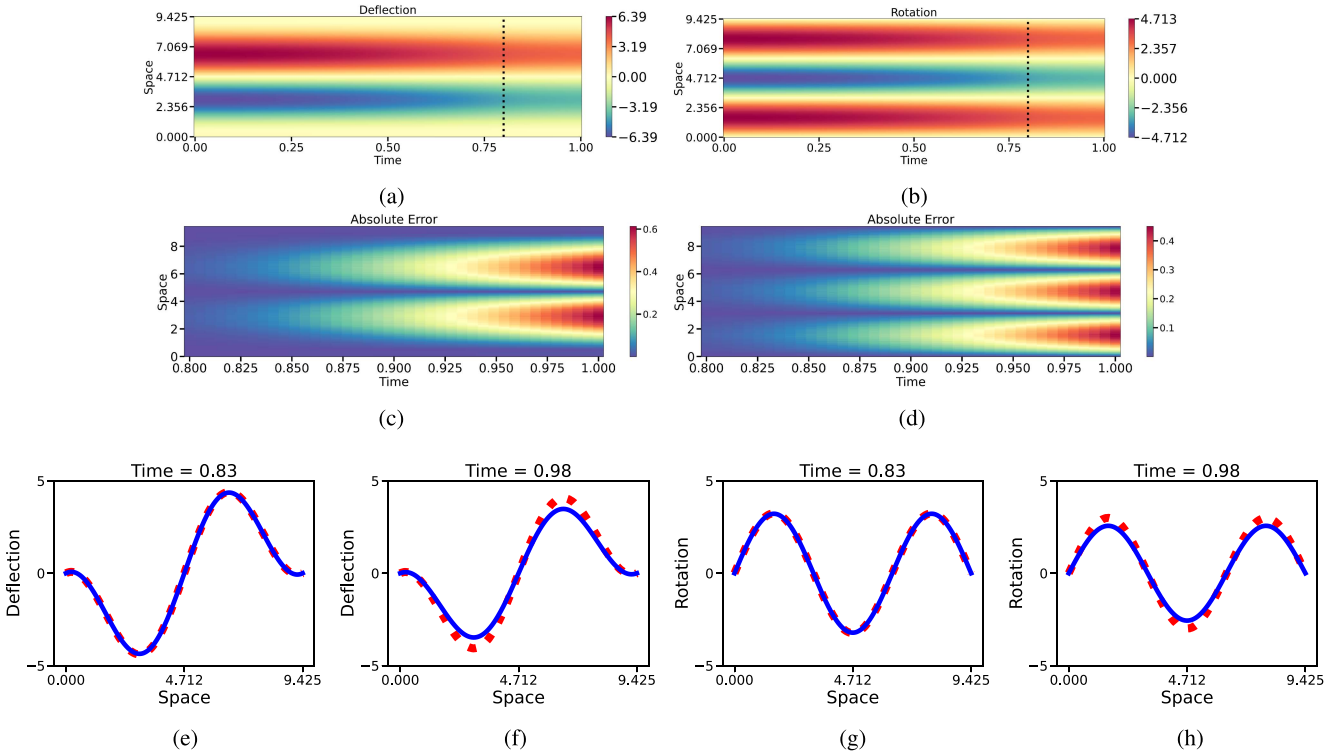

Fig. 4. Timoshenko beam on the Winkler foundation. (a) Deflection causal PINN-LEM. (b) Rotation causal PINN-LEM. (c) Absolute error for deflection in generalized temporal domain. (d) Absolute error for rotation in generalized temporal domain. (e) Deflection snapshot at $t = 0.83$. (f) Deflection snapshot at $t = 0.98$. (g) Rotation snapshot at $t = 0.83$. (h) Rotation snapshot at $t = 0.98$.

TABLE I
BEAM DEFLECTION, ROTATION, MOVING LOAD, AND CATENARY UPLIFT

	RNN	LSTM	GRU	CoRNN	LEM	Transformer	NODE
Euler-Bernoulli Beam Deflection							
L2-norm(↓)	0.012	0.4925	1.9471	0.0194	**0.0080**	0.0084	0.0081
Max error(↓)	0.2527	0.7814	1.8173	0.1842	**0.1276**	0.1345	0.2305
Explained Variance score(↑)	0.9874	0.5074	-0.9470	0.9805	**0.9919**	0.9915	0.9918
Mean absolute error(↓)	0.0585	0.4163	0.8015	0.0749	**0.0466**	0.0470	0.0410
Timoshenko Beam Deflection							
L2-norm(↓)	0.3813	0.3167	0.3798	0.2060	**0.0058**	0.0228	0.0095
Max error(↓)	2.9385	2.7387	2.9378	2.2792	**0.6126**	1.0249	0.6649
Explained Variance score(↑)	0.6186	0.6832	0.62017	0.7939	**0.9941**	0.9771	0.9904
Mean absolute error(↓)	1.2769	1.1622	1.2734	0.9330	**0.1270**	0.2758	0.1789
Timoshenko Beam Rotation							
L2-norm(↓)	0.3813	0.3146	0.3797	0.2059	**0.0055**	0.0228	0.0095
Max error(↓)	2.1680	2.0224	2.1683	1.6808	**0.4496**	0.7523	0.4907
Explained Variance score(↑)	0.6183	0.6847	0.6198	0.7935	**0.9942**	0.9766	0.9902
Mean absolute error(↓)	1.1368	1.0310	1.1334	0.8306	**0.1105**	0.2452	0.1593
Moving Load Mid-Point Beam Deflection							
L2-norm(↓)	0.5579	0.3803	0.6026	0.0039	**0.0002**	0.1307	0.0084
Max error(↓)	0.0190	0.0194	0.0188	0.0024	**0.0006**	0.0156	0.0042
Explained Variance score(↑)	0.2974	0.2714	0.3444	0.9927	**0.9995**	0.7056	0.9802
Mean absolute error(↓)	0.0084	0.0068	0.0088	0.0006	**0.0001**	0.0039	0.0009
Catenary Uplift with Different Train Speeds							
L2-norm(↓)	1.49e-6	1.54e-7	3.0e-7	6.38e-8	**6.43e-8**	1.01e-6	1.32e-6
Max error(↓)	0.0845	0.0085	0.0135	0.0063	**0.0063**	0.0194	0.0259
Explained Variance score(↑)	0.7728	0.9722	0.9562	0.9934	**0.9932**	0.8451	0.7961
Mean absolute error(↓)	0.0022	0.0015	0.0019	0.0009	**0.0009**	0.0043	0.0041

Higher (or lower) values are preferred by ↑ (or ↓).

absolute errors for different loading for LEM. The absolute error increases minorly with an increase in the loading. The last row of Fig. 5 present the snapshots of the predicted deflections. The red dots depict predictions, while the blue solid line represents actual deflection simulated using the FEM. Table I compares LEM and CoRNN with other sequential methods, showcasing the better generalization abilities of the proposed method.

3) Catenary Contact Wire Uplift: Fig. 6. illustrates catenary contact wire uplift deflections at varying speeds ($70\,\mathrm{kmh}^{-1}$, $80\,\mathrm{kmh}^{-1}$, and $90\,\mathrm{kmh}^{-1}$). The red dots depict predictions,

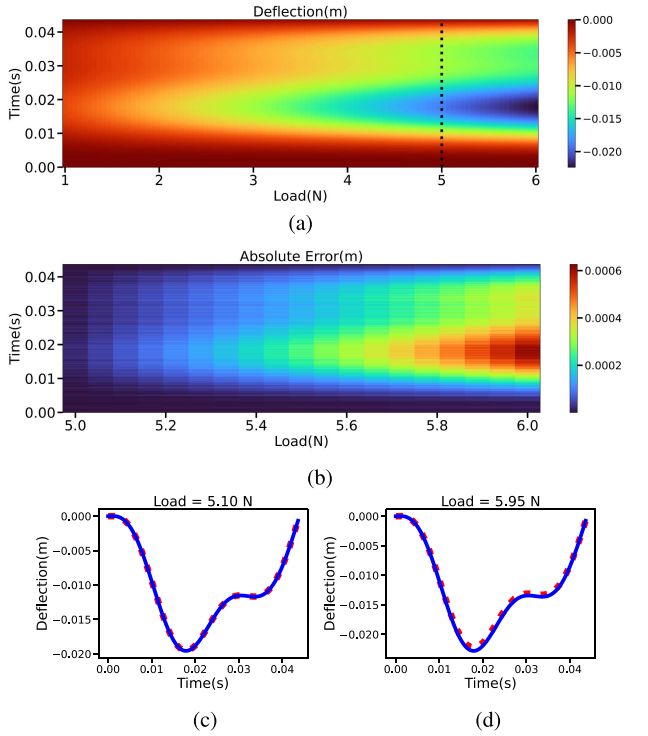

Fig. 5. Mid-point deflection for moving load. (a) Mid-point deflection FEM-LEM. (b) Absolute error in mid-point deflection for unseen loadings. (c) Snapshot at Load $5.10\,\mathrm{N}$. (d) Snapshot at Load $5.95\,\mathrm{N}$.

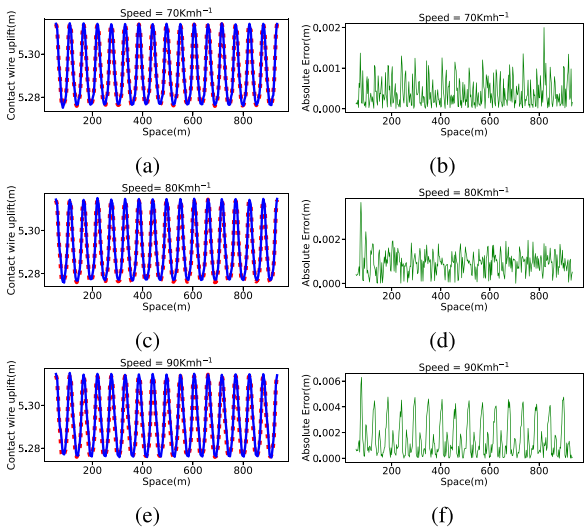

Fig. 6. Contact wire uplift with different train speeds. (a, c, e): Contact wire uplift prediction for unseen speeds. (b, d, f): Absolute errors in contact wire uplift predictions for different train speeds.

while the blue solid line represents actual deflection simulated using the FEM. Fig. 6(a), (c), and (e) represents the contact wire uplift predictions at $70\,\mathrm{kmh}^{-1}$, $80\,\mathrm{kmh}^{-1}$, and $90\,\mathrm{kmh}^{-1}$, respectively. Fig. 6(b), (d), and (f) shows absolute errors in those uplift predictions. The errors for different speeds fall within a similar range, yet it is evident that the error increases minorly with the increase in train speed. Table I compares LEM and CoRNN with other methods, showcasing that LEM performs better in all metrics.

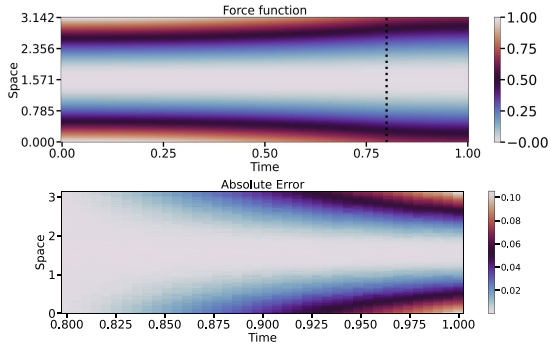

Fig. 7. Approximation of the force function (*top*) for the inverse problem and corresponding absolute error in extrapolation (*bottom*).

4) Inverse Problem for Timoshenko Double Beam System: To demonstrate the capability of the two-stage approach for out-of-domain prediction in inverse problems, a Timoshenko double-beam system is considered [15]. The goal is to predict the unknown force acting on the beam system outside the training domain. PINN simulates the force function in the first stage. In the second stage, these data are utilized to train LEM, with testing performed in the out-of-domain region. Fig. 7 presents the force function acting on the beam and the absolute error in predicting the force in an extrapolation scenario. The error metric results for this experiment include the L2 error, maximum absolute error, mean absolute error, and explained variance score, obtained as 0.0224, 0.1349, 0.0309, and 0.9720, respectively. The results illustrate the proposed method's efficacy in extrapolating an inverse problem.

VI. Conclusion

This work addressed the longstanding challenge of generalizing simulations for engineering and industry. A resolution-invariant pipeline is proposed, processing data at various resolutions by introducing a novel two-stage approach combining state-of-the-art simulators with classical mathematical models infused in neural recurrent architectures. The presented approach efficiently generalizes dynamics in the time and parametric domain without relying on data from the untrained domain, eliminating the need for tedious remeshing and resimulation in computer-aided simulation software. Moreover, the proposed workflow demonstrated superior performance over traditional recurrent architectures for various dynamic simulation problems in structural engineering. Experiments on real-world scenarios like catenary-pantograph interactions in railway systems motivate application to various industrial applications for enhancing simulation efficiency and accuracy in out-of-domain predictions.

Potential future works include extending the proposed methodology to high-dimensional problems and complex geometries such as shells and bridges. Another direction could focus on accelerating the simulations through optimization techniques and parallel computing, enabling real-time predictions. In addition, industrial applications often contend with

uncertainties and stochastic variations. Enhancing the method to account for uncertainties in factors, such as material properties, loading conditions, and external influences would allow for probabilistic predictions, thereby increasing the robustness and applicability of the method in diverse industrial scenarios.

Acknowledgment

Views and opinion expressed are however those of the authors(s) only and do not necessarily reflect those of the European Union. Neither the European Union nor the granting authority can be held responsible for them.

References

[1] M. Schluse, M. Priggemeyer, L. Atorf, and J. Rossmann, "Experimentable digital twins—Streamlining simulation-based systems engineering for industry 4.0," *IEEE Trans. Ind. Informat.*, vol. 14, no. 4, pp. 1722–1731, Apr. 2018.

[2] F. Tao, H. Zhang, A. Liu, and A. Y. Nee, "Digital twin in industry: State-of-the-Art," *IEEE Trans. Ind. Informat.*, vol. 15, no. 4, pp. 2405–2415, Apr. 2019.

[3] R. J. Smit, W. M. Brekelmans, and H. E. Meijer, "Prediction of the mechanical behavior of nonlinear heterogeneous systems by multi-level finite element modeling," *Comput. Methods Appl. Mech. Eng.*, vol. 155, no. 1/2, pp. 181–192, 1998.

[4] O. Avci, O. Abdeljaber, S. Kiranyaz, M. Hussein, M. Gabbouj, and D. J. Inman, "A review of vibration-based damage detection in civil structures: From traditional methods to machine learning and deep learning applications," *Mech. Syst. Signal Process.*, vol. 147, 2021, Art. no. 107077.

[5] G. E. Karniadakis, I. G. Kevrekidis, L. Lu, P. Perdikaris, S. Wang, and L. Yang, "Physics-informed machine learning," *Nature Rev. Phys.*, vol. 3, no. 6, pp. 422–440, 2021.

[6] T. Kapoor, H. Wang, A. Núñez, and R. Dollevoet, "Transfer learning for improved generalizability in causal physics-informed neural networks for beam simulations," *Eng. Appl. Artif. Intell.*, vol. 133, 2024, Art. no. 108085.

[7] Y. Wang, Y. He, B. Kang, J. Liu, and C. Sun, "ProbSparse attention-based fault diagnosis for industrial robots under different working conditions," *IEEE Trans. Instrum. Meas.*, vol. 73, Dec. 4, 2024, Art. no. 3514512.

[8] X. Fu, X. Yang, P. Zanchetta, M. Tang, Y. Liu, and Z. Chen, "An adaptive data-driven iterative feedforward tuning approach based on fast recursive algorithm: With application to a linear motor," *IEEE Trans. Ind. Informat.*, vol. 19, no. 4, pp. 6160–6169, Apr. 2023.

[9] Z. Y. Ding, J. Y. Loo, S. G. Nurzaman, C. P. Tan, and V. M. Baskaran, "A zero-shot soft sensor modeling approach using adversarial learning for robustness against sensor fault," *IEEE Trans. Ind. Informat.*, vol. 19, no. 4, pp. 5891–5901, Apr. 2023.

[10] S. Li, W. Li, C. Cook, C. Zhu, and Y. Gao, "Independently recurrent neural network (IndRNN): Building a longer and deeper RNN," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2018, pp. 5457–5466.

[11] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[12] K. Cho et al., "Learning phrase representations using RNN encoder-decoder for statistical machine translation," in *Proc 2014 Conf. Empirical Methods Natural Lang. Process. (EMNLP)*, 2014, pp. 1724–1734.

[13] T. K. Rusch and S. Mishra, "Coupled oscillatory recurrent neural network (coRNN): An accurate and (gradient) stable architecture for learning long time dependencies," in *Proc. Int. Conf. Learn. Representations*, 2021.

[14] T. K. Rusch, S. Mishra, N. B. Erichson, and M. W. Mahoney, "Long expressive memory for sequence modeling," in *Proc. Int. Conf. Learn. Representations*, 2022.

[15] T. Kapoor, H. Wang, A. Núñez, and R. Dollevoet, "Physics-informed neural networks for solving forward and inverse problems in complex beam systems," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 35, no. 5, pp. 5981–5995, May 2024.

[16] M. Eisenberger and J. Clastornik, "Vibrations and buckling of a beam on a variable winkler elastic foundation," *J. Sound Vib.*, vol. 115, no. 2, pp. 233–241, 1987.

[17] F. Zonzini, M. Zauli, M. Mangia, N. Testoni, and L. De Marchi, "Model-assisted compressed sensing for vibration-based structural health monitoring," *IEEE Trans. Ind. Informat.*, vol. 17, no. 11, pp. 7338–7347, Nov. 2021.

[18] M. David, W. Derigent, G. Loubet, A. Takacs, and D. Dragomirescu, "Communicating materials: Communicating concrete development for construction industry," *IEEE Trans. Ind. Informat.*, vol. 20, no. 4, pp. 6983–6994, Apr. 2024.

[19] F. P. Beer, E. R. Johnston, J. T. DeWolf, D. F. Mazurek, and S. Sanghi, *Mechanics of Materials*, vol. 1. New York, NY, USA: McGraw-Hill, 1992.

[20] E. Carrera, G. Giunta, and M. Petrolo, *Beam Structures: Classical and Advanced Theories*. HoboKen, NJ, USA: Wiley, 2011.

[21] A. Öchsner and A. Öchsner, "Timoshenko beam theory," in *Classical Beam Theories of Structural Mechanics*. Cham, Switzerland: Springer, 2021, pp. 67–104.

[22] H.-P. Deng, Y.-B. He, B.-C. Wang, and H.-X. Li, "Physics-dominated neural network for spatiotemporal modeling of battery thermal process," *IEEE Trans. Ind. Informat.*, vol. 20, no. 1, pp. 452–460, Jan. 2024.

[23] X. Lu, F. Yin, C. Liu, and M. Huang, "Online spatiotemporal extreme learning machine for complex time-varying distributed parameter systems," *IEEE Trans. Ind. Informat.*, vol. 13, no. 4, pp. 1753–1762, Aug. 2017.

[24] K. Xu, H. Yang, C. Zhu, X. Jin, B. Fan, and L. Hu, "Deep extreme learning machines based two-phase spatiotemporal modeling for distributed parameter systems," *IEEE Trans. Ind. Informat.*, vol. 19, no. 3, pp. 2919–2929, Mar. 2023.

[25] J. Li, Y. Wang, Y. Zi, H. Zhang, and C. Li, "Causal consistency network: A collaborative multimachine generalization method for bearing fault diagnosis," *IEEE Trans. Ind. Informat.*, vol. 19, no. 4, pp. 5915–5924, Apr. 2023.

[26] C. Xie and S. Chen, "A physics-guided reversible residual neural network model: Applied to build forward and inverse models for turntable servo system," *IEEE Trans. Ind. Informat.*, vol. 19, no. 4, pp. 5882–5890, Apr. 2023.

[27] C. Liu and K. Gryllias, "Simulation-driven domain adaptation for rolling element bearing fault diagnosis," *IEEE Trans. Ind. Informat.*, vol. 18, no. 9, pp. 5760–5770, Sep. 2022.

[28] R. T. Chen, Y. Rubanova, J. Bettencourt, and D. K. Duvenaud, "Neural ordinary differential equations," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 31, 2018, pp. 6572–6583.

[29] Z. Yuan, Y. Wang, X. Ban, C. Ning, H.-N. Dai, and H. Wang, "Autonomous-jump-ODENet: Identifying continuous-time jump systems for cooling-system prediction," *IEEE Trans. Ind. Informat.*, vol. 19, no. 7, pp. 7894–7904, Jul. 2023.

[30] S. Wang, Y. Wang, B. Yang, F. Mo, and Z. Zhang, "Variational Bayesian learning with reliable likelihood approximation for accurate process quality evaluation," *IEEE Trans. Ind. Informat.*, vol. 20, no. 1, pp. 815–823, Jan. 2024.

[31] L. Qin, S. Zhang, T. Sun, and X. Zhao, "An interpretable neuro-dynamic scheme with feature-temporal attention for remaining useful life estimation," *IEEE Trans. Ind. Informat.*, vol. 20, no. 4, pp. 5505–5516, Apr. 2024.

[32] M. Raissi, P. Perdikaris, and G. E. Karniadakis, "Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations," *J. Comput. Phys.*, vol. 378, pp. 686–707, 2019.

[33] S. Wang, S. Sankaran, and P. Perdikaris, "Respecting causality for training physics-informed neural networks," *Comput. Methods Appl. Mech. Eng.*, vol. 421, 2024, Art. no. 116813.

[34] L. Fryba, *Vibration of Solids and Structures Under Moving Loads*, vol. 1. Berlin, Germany: Springer, 2013.

[35] Y. Song, H. Wang, and Z. Liu, "An investigation on the current collection quality of railway pantograph-catenary systems with contact wire wear degradations," *IEEE Trans. Instrum. Meas.*, vol. 70, Dec. 4, 2021, Art. no. 9003311.

[36] Y. Song, H. Wang, G. Frøseth, P. Nåvik, Z. Liu, and A. Rønnquist, "Surrogate modelling of railway pantograph-catenary interaction using deep long-short-term-memory neural networks," *Mechanism Mach. Theory*, vol. 187, 2023, Art. no. 105386.

[37] A. Vaswani, "Attention is all you need," in *Proc. Adv. Neural Inf. Process. Syst.*, 2017, pp. 6000–6010.