Best of Both Worlds: Bridging Laplace and Fourier for Generalizable and Efficient Operator Learning

Jeongun Ha[*]
Department of Mathematics
Korea University
Seoul, Korea 02841
highsoldier@korea.ac.kr

Donghun Lee[†]
Department of Mathematics
Korea University
Seoul, Korea 02841
holy@korea.ac.kr

Abstract

We introduce Laplace–Fourier Neural Operator (LFNO), a novel operator learning model that bridges the strengths of Laplace Neural Operators (LNO) and Fourier Neural Operators (FNO). By combining the transient response of LNO with the steady-state response of FNO through the Fourier integral operator, our model enables capturing transient behavior more effectively than both LNO and FNO while remaining comparable on linear and nonlinear PDEs. We demonstrate LFNO's effectiveness in solving ODE (Duffing) and two PDEs (Euler-Bernoulli beam, Heat) in comparison to LNO. These results suggest LFNO as a versatile and generalizable neural operator framework for modeling dynamical systems in both linear and nonlinear regimes, offering a principled step toward ML-powered scalable methods for physical sciences.

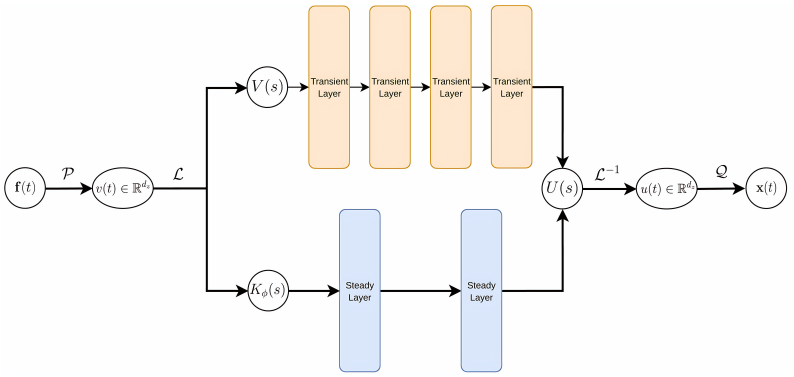
1 Introduction

Differential equations and their applications often form the mathematical backbone of the physical sciences, governing phenomena such as fluid dynamics, electromagnetism, and quantum mechanics. Recently, machine learning has emerged as a powerful paradigm for solving differential equations. Physics-Informed Neural Networks (PINNs) [Raissi et al., 2019] integrate physical laws into the learning process, enabling data-efficient and mesh-independent solutions for nonlinear PDEs. PINN's effectiveness has been demonstrated in applications such as modeling turbulent flows through the Reynolds-averaged Navier-Stokes equations [Eivazi et al., 2022] and solving inverse problems via Miura transformation–based formulations [Lin and Chen, 2023], for instance. Yet, the PINN-based approach is case-specific, as it solves a specific instance and parametrization of differential equations.

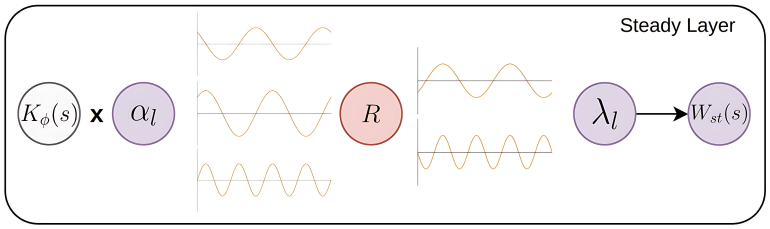
Seeking for a greater generalizability, the operator learning paradigm has emerged. One of the earliest cases, DeepONet [Lu et al., 2019], is built upon the universal approximation theorem of operators, enabling the approximation of highly nonlinear operators across diverse domains. Based on this concept, various neural networks for solving PDEs, such as Li et al. [2020b], emerged to generalize resolution-invariant methods. Fourier Neural Operator (FNO) [Li et al., 2020a] was later introduced as a spectral approach to operator learning. FNO has been extended in many directions, including refined Fourier-based kernels for enhanced expressiveness in high-frequency or inhomogeneous scenarios [Shin et al., 2022], hybrid Fourier–Chebyshev representations for non-periodic settings [Fanaskov and Oseledets, 2023], and generalizations to arbitrary geometries [Li et al., 2023]. More recently, Laplace Neural Operator (LNO) [Cao et al., 2024] was proposed to extend operator learning into the Laplace domain. By exploiting a pole–residue formulation, LNO provides a general framework to capture both transient and steady-state behaviors of dynamical systems. Despite LNO's theory, its

[*]first author: highsoldier@korea.ac.kr, prism4304@naver.com
[†]corresponding author

Machine Learning and the Physical Sciences Workshop, NeurIPS 2025.



(a) Full Architecture of LFNO



(b) Description of Applying the Fourier Integral Operator in Steady Layer

Figure 1: Model architecture of LFNO. (a) Overall process: given input function $\mathbf{f}(t)$, 1. apply Laplace transform, 2. take the Transient and the Steady layers for each component, and 3. project the output, $u(t)$. (b) Steady layer structure: apply the pole-residue method, linear transform R on the lower Fourier modes, and filter out the higher modes.

steady-state response has implementation-level limitations due to information loss across frequency intervals.

Hence, we propose Laplace-Fourier Neural Operator (LFNO), a novel operator learning framework combining the best of both LNO and FNO, both theoretical elegance and practical efficiency. LFNO leverages LNO's framework to separate transient and steady-state dynamics and applies the Fourier integral operator to model the steady-state component in the frequency domain efficiently. We give theoretical justification for this merger, mathematical formulations of LFNO, and its empirical validations on benchmark ODE and PDE systems.

2 Methodology

We address the practical limitation of LNO [Cao et al., 2024] by adopting the essence of FNO framework [Li et al., 2020a]. In particular, operating in the frequency domain by decomposing input signals into frequency components, and then applying a learned operator to map them to outputs works, as LNO's steady-state response is a superposition of discrete frequency components, aligned with FNO's mechanism, and to enhance expressivity while preserving mathematical consistency. This integration enhances efficiency by approximating infinite-frequency expansions with finite computations in the frequency domain while maintaining accuracy. We hereby refer to this integrated approach as the Laplace–Fourier Neural Operator (LFNO).

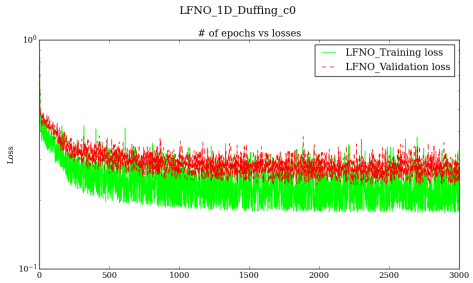
LFNO's architecture and layer composition are presented in Fig. 1, using notations from LNO [Cao et al., 2024] for readability, and its procedure is explained in plain language as follows. LFNO takes an input function $\mathbf{f}(t)$ and lifts it to a higher-dimensional representation using a neural network $\mathcal{P}$. Next, Laplace transform is applied to the resulting higher-dimensional function $v(t)$, and then the transformed output is decomposed into its transient and steady-state components for each dimension. In the steady layer, the linear transform R is applied to the lower Fourier modes, while the higher

modes are filtered out. Then the steady-state response residues λ_l are computed, based on input poles $i\omega_l$ and residues α_l. After all computations of the transient and stable layers, the local linear transformation W is applied. Finally, the output $u(t)$ is projected back to the target dimension using another neural network $\mathcal{Q}$.

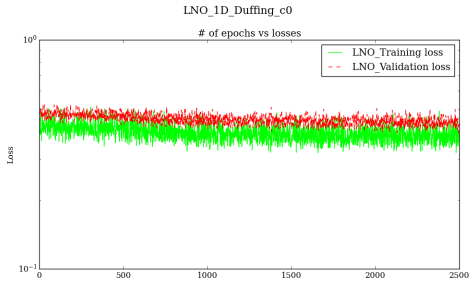
Our integration combines the generalizable theory from LNO's solution decomposition with the practical learning capacity of the frequency domain from FNO. Theoretical analysis and justification of this approach, which starts with expressing the LNO solution in its standard decomposition, are detailed in Appendix A.

3 Numerical Experiments

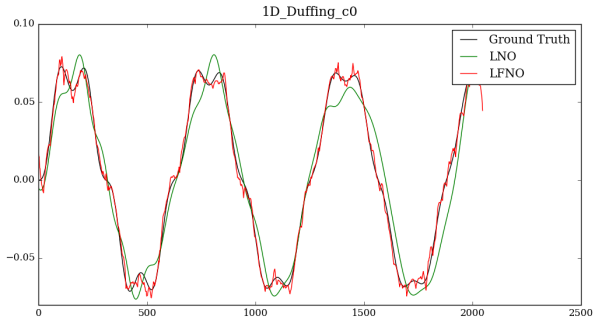
In this section, we compare the performance of LFNO with that of LNO. We adopt ODE (Duffing) and two PDEs (Beam, Heat) used in the original evaluation of LNO [Cao et al., 2024]. All parameters related to the Differential Equation are identical to those used in LNO. We calculate the relative error $\mathcal{L}_2$ for the prediction of the test samples and use it as a quantitative metric. We then compare the samples from the two models, using them as a qualitative and quantitative metric. We conduct experiments with a fixed number of epochs for each task. The experimental results for Duffing, Euler-Bernoulli beam, and Heat equation are presented in Fig. 2, Fig. 3, Fig. 4, Fig. 5, and Table 1, while details of the implementation of the proposed model are provided in Appendix C.



(a) LFNO learning curve



(b) LNO learning curve

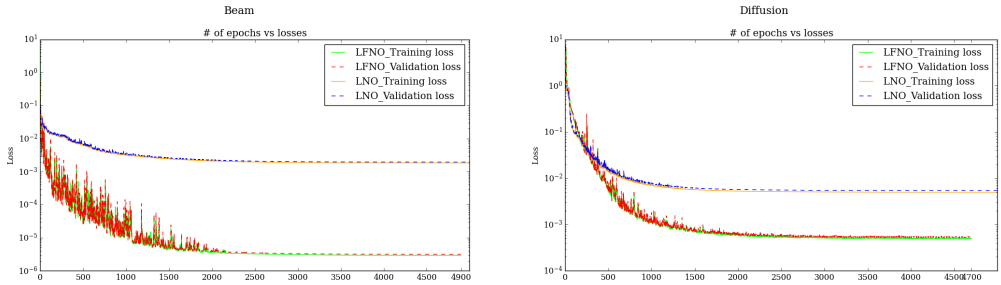


(c) Duffing $c = 0$

Figure 2: Duffing Learning curves on different models and the qualitative result. (a), (b): Learning curves of LFNO and LNO, (c): Comparison of LFNO and LNO with the target sample. Red: LFNO result; Green: LNO result.

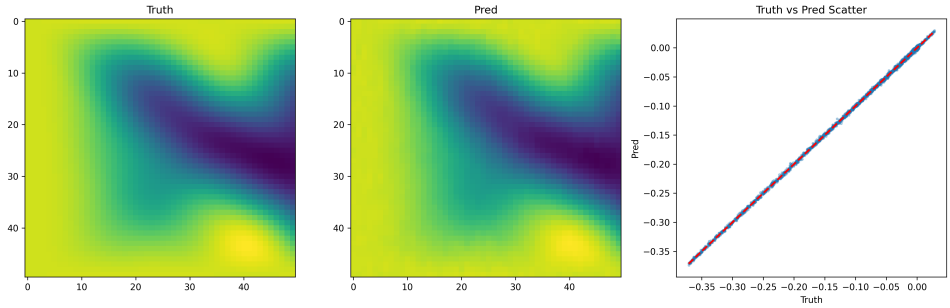
ODE Results Fig. 2 illustrates the learning curve and the qualitative results of two models. LNO produces a validation loss that is lower than the training loss. Although LFNO achieves a lower train loss value, LNO appears to train more stably. Fig. 2c indicates that LFNO better describes the inflection points in the data than LNO. Also, LFNO provides a more accurate representation of the initial point.

PDE Results Fig. 3 illustrates the learning curves and qualitative results of LFNO and LNO. Fig. 3a and Fig. 3b show that the training and validation losses of two models have notable differences.

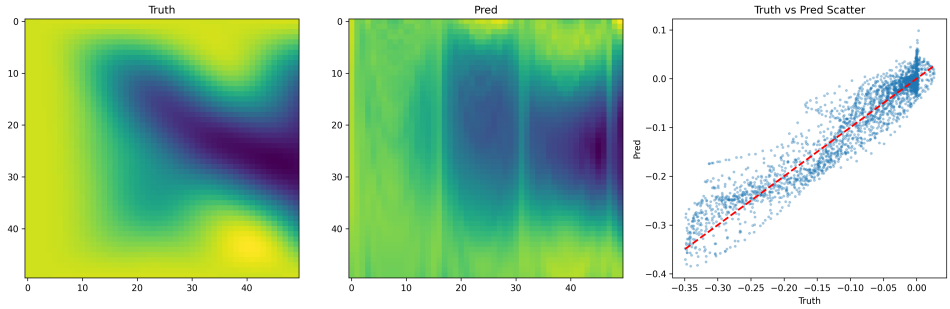


(a) Euler-Bernoulli beam (b) Heat

Figure 3: PDE Learning curves on different models. Training and validation sets are represented by lime and red for LFNO, orange and blue for LNO.



(a) Euler-Bernoulli beam – LFNO



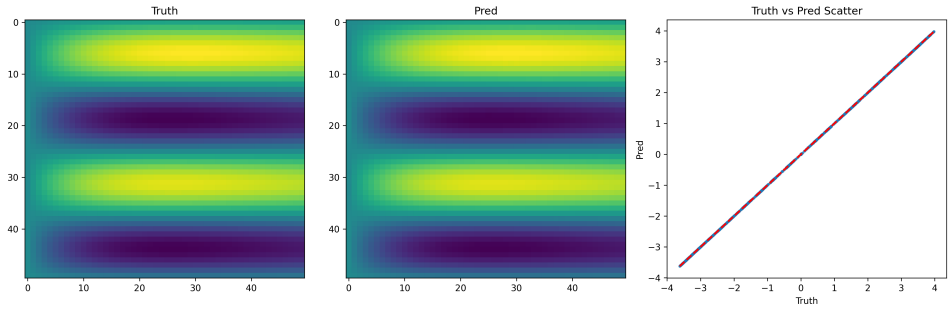
(b) Euler-Bernoulli beam – LNO

Figure 4: Euler-Bernoulli beam equation: Comparison of the qualitative results and scatter computed by LFNO and LNO: (a), (b) represent the ground truth shown in the left panel, and the predictions of LFNO and LNO in the center panel. The right panel of each subfigure shows the $(\text{truth}, \text{prediction})$ scatter plot, where points aligned closer to $y = x$ indicate a higher accuracy.

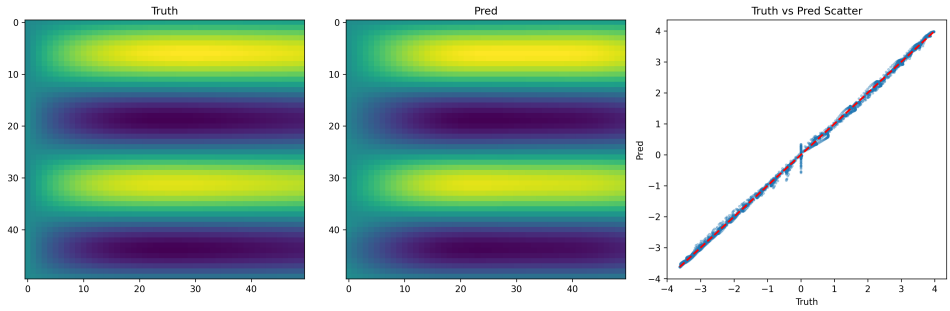
Comparison of the ground truth and prediction plots in Fig. 4, Fig. 5. We confirm that LNO breaks down completely in Fig. 4b, failing to produce any meaningful predictions, evident in the difference between the two plots on the left and the non-diagonal scatter plot on the right.

4 Discussion

For linear PDEs, the transient and steady-state responses can be clearly separated, and applying the Fourier integral operator to the steady-state term provides results that are consistent both theoretically and experimentally. For nonlinear PDEs, LFNO's novel elements over LNO enable a better approximation of complex mappings, allowing for a more accurate reproduction of their dynamic behavior.



(a) Heat – LFNO



(b) Heat – LNO

Figure 5: Heat equation: Comparison of the qualitative results and scatter computed by LFNO and LNO: (a), (b) represent the ground truth shown in the left panel, and the predictions of LFNO and LNO in the center panel. The right panel of each subfigure shows the $(\text{truth}, \text{prediction})$ scatter plot, where points aligned closer to $y = x$ indicate a higher accuracy.

	ODEs					**PDEs**				
Equation	Duffing $c = 0$	Duffing $c = 0.5$	Lorenz $\rho = 5$	Lorenz $\rho = 10$	Pendulum $c = 0.5$	Beam	Heat	Reaction Diffusion	Brusselator	Gray-scott
Epochs	6200	5200	8100	4000	6800	4900	4700	4700	5000	4800
FNO	0.9998	0.0565	0.1225	0.4340	0.0426	**0.0035**	**0.0017**	**0.0030**	**0.0024**	**0.0138**
LNO	0.2014	0.0762	0.0288	0.2571	0.0735	0.2509	0.0227	0.0779	0.0137	0.1709
LFNO	**0.1288**	**0.0314**	**0.0075**	**0.1731**	**0.0315**	0.0153	0.0051	0.0049	0.0041	0.0155

Table 1: $\mathcal{L}_2$ error table for ODEs and PDEs tasks.

The validation loss was observed to be lower than the training loss for LNO. This unexpected outcome is conjectured to arise from particularities of how training and validation datasets are created in Cao et al. [2024]. Adjusting the ODE training data helped alleviate this discrepancy. While LFNO exhibited stable training on linear PDEs, its performance on nonlinear PDEs was comparatively less stable. This suggests a natural line of future work: (i) scaling PDE datasets to facilitate further experimentation and analysis, and (ii) increasing model depth to enhance training stability.

5 Conclusion

We propose Laplace–Fourier Neural Operator (LFNO), a unified framework that integrates the LNO and FNO, inheriting the merits of both prior works. LFNO captures the transient response based on LNO's theoretical generalizability, while the Fourier integral operator enhances the steady-state response, and the nonlinearity introduced into the steady-state component enables a broader representation of signals. These results demonstrate that LFNO offers clear advantages as an operator learning framework for linear and nonlinear DEs with transient states, while showing performance comparable to that of existing models.

Acknowledgement

This work was supported by the National Research Foundation of Korea (NRF) grant funded by the Korea government (MSIT) (RS-2025-24873052).

References

Q. Cao, S. Goswami, and G. E. Karniadakis. Laplace neural operator for solving differential equations. *Nature Machine Intelligence*, 6(6):631–640, 2024.

H. Eivazi, M. Tahani, P. Schlatter, and R. Vinuesa. Physics-informed neural networks for solving reynolds-averaged navier–stokes equations. *Physics of Fluids*, 34(7), 2022.

V. S. Fanaskov and I. V. Oseledets. Spectral neural operators. In *Doklady Mathematics*, volume 108, pages S226–S232. Springer, 2023.

Z. Li, N. Kovachki, K. Azizzadenesheli, B. Liu, K. Bhattacharya, A. Stuart, and A. Anandkumar. Fourier neural operator for parametric partial differential equations. *arXiv preprint arXiv:2010.08895*, 2020a.

Z. Li, N. Kovachki, K. Azizzadenesheli, B. Liu, K. Bhattacharya, A. Stuart, and A. Anandkumar. Neural operator: Graph kernel network for partial differential equations. *arXiv preprint arXiv:2003.03485*, 2020b.

Z. Li, D. Z. Huang, B. Liu, and A. Anandkumar. Fourier neural operator with learned deformations for pdes on general geometries. *Journal of Machine Learning Research*, 24(388):1–26, 2023.

S. Lin and Y. Chen. Physics-informed neural network methods based on miura transformations and discovery of new localized wave solutions. *Physica D: Nonlinear Phenomena*, 445:133629, 2023.

L. Lu, P. Jin, and G. E. Karniadakis. Deeponet: Learning nonlinear operators for identifying differential equations based on the universal approximation theorem of operators. *arXiv preprint arXiv:1910.03193*, 2019.

M. Raissi, P. Perdikaris, and G. E. Karniadakis. Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations. *Journal of Computational physics*, 378:686–707, 2019.

J. Y. Shin, J. Y. Lee, and H. J. Hwang. Pseudo-differential neural operator: Generalized fourier neural operator for learning solution operators of partial differential equations. *arXiv preprint arXiv:2201.11967*, 2022.

A Mathematical Formulation

All notation is adopted in [Cao et al., 2024].

$$u(t) = \sum_{n=1}^{N} \gamma_n \exp(\mu_n t) + \sum_{l=-L}^{L} \lambda_l \exp(i\omega_l t), \tag{1}$$

where the first summation term is the transient response and the second summation term is the familiar steady-state response. Simplify the above equation: $u(t) = u_{transient}(t) + u_{steady}(t)$. Since the steady-state response is represented in the frequency domain, we can apply the Fast Fourier Transform to the steady-state response. Applying the Fourier transform to $u_{steady}(t)$ yields the following result.

$$\mathcal{F}(u_{steady})(\omega) = \int_{-\infty}^{\infty} \left(\sum_{l=-\infty}^{\infty} \lambda_l e^{i\omega_l t} \right) e^{-i\omega t} dt. \tag{2}$$

We can change the order of summation and integration by the Fubini theorem. Therefore, we formulate $u_{steady}(t)$ as follows:

$$\mathcal{F}(u_{steady})(\omega) = \int_{-\infty}^{\infty} \Big(\sum_{l=-\infty}^{\infty} \lambda_l e^{i\omega_l t} \Big) e^{-i\omega t} dt = \sum_{l=-\infty}^{\infty} \lambda_l \int_{-\infty}^{\infty} e^{i\omega_l t} e^{-i\omega t} dt \tag{3}$$

The integral in Eq. (3) is formally divergent as a classical integral. Therefore, it is understood in the sense of distributions, yielding the Dirac delta function:

$$\int_{-\infty}^{\infty} e^{i\omega_l t} e^{-i\omega t} dt = \int_{-\infty}^{\infty} e^{i(\omega_l - \omega)t} dt = 2\pi\, \delta(\omega_l - \omega), \tag{4}$$

where the factor 2π appears according to the Fourier transform convention used. Using the sifting property of the Dirac delta, we can then write

$$\mathcal{F}(u_{steady})(\omega) = \sum_{l=-\infty}^{\infty} \lambda_l \cdot \delta(\omega_l - \omega), \tag{5}$$

which explicitly represents the discrete frequency components of the steady-state response. We now apply the Fourier integral operator [Li et al., 2020a] to these frequency components. FNO learns a non-linear operator $R(\omega; \theta)$ in the frequency domain, where θ denotes learnable parameters, mapping each input frequency component to the corresponding output. The output frequency components are

$$\begin{aligned} \hat{u}_{out}(\omega) &= R(\omega; \theta) \cdot \mathcal{F}(u_{steady})(\omega) \\ &= R(\omega; \theta) \cdot \sum_{l=-\infty}^{\infty} \lambda_l \cdot \delta(\omega_l - \omega) \\ &= \sum_{l=-\infty}^{\infty} \lambda_l \cdot R(\omega_l; \theta) \cdot \delta(\omega_l - \omega), \end{aligned} \tag{6}$$

where we have applied the sifting property of the Dirac delta to evaluate $R(\omega; \theta)$ at $\omega = \omega_l$. Finally, applying the inverse Fourier Transform yields the time-domain output:

$$u_{out}(t) = \mathcal{F}^{-1}(\hat{u}_{out})(\omega) = \sum_{l=-\infty}^{\infty} \lambda_l R(\omega_l; \theta) e^{i\omega_l t}. \tag{7}$$

Thus, the full solution combining the transient and steady-state responses can be written as

$$u(t) = \sum_{n=1}^{N} \gamma_n \exp(\mu_n t) + \sum_{l=-\infty}^{\infty} \lambda_l R(\omega_l; \theta) e^{i\omega_l t}. \tag{8}$$

B Example equations for Experiment

Duffing Equation The Duffing equation is a nonlinear second-order differential equation used to model the driven oscillator. The Duffing equation is written as follows.

$$\ddot{x} + \delta\dot{x} + \alpha x + \beta x^3 = f(t), \tag{9}$$

where δ is the damping coefficient, α and β are the stiffness of the system, and $x, \dot{x}$, and $\ddot{x}$ are the displacement, velocity, and acceleration of the dynamic response. We only experiment with a Duffing oscillator when it does not include damping.

Euler-Bernoulli beam The Euler-Bernoulli beam is a simplification of the linear theory of elasticity, providing a means to calculate the load-carrying and deflection characteristics of beams. The equation is written as:

$$EI \frac{\partial^4 y(x,t)}{\partial x^4} + \rho A \frac{\partial^2 y(x,t)}{\partial t^2} = f(x,t), \tag{10}$$

where $y(x,t)$ is the deflection of the beam at the location x and time t, $f(x,t)$ is the source term, E, I the elastic modulus and the second moment of area of the cross-section of the beam, and ρ and A are the material density and the area of the cross-section of the beam.

C Implementation details

For all models, we train them using the Adam optimizer and step learning rate scheduler to improve stability and convergence. We train the model on an NVIDIA RTX 3090 GPU. The model hyperparameters are listed in Table 2.

Application	Duffing LNO	Duffing LFNO	Beam LNO	Beam LFNO
Layer	1	6	1	6
Width	4	4	16	16
Mode	16	16	8	8
Train batch size	20	64	200	200
Learning rate	0.002	0.002	0.002	0.002
weight decay	0.0001	0.01	0.0001	0.01
scheduler step size	100	100	100	100
gamma	0.5	0.8	0.5	0.8
Activation function	sin	relu	sin	relu

Table 2: Hyperparameters used in the LNO and LFNO for training