Learning Solutions of Coupled Differential Equations with Physics-Informed Neural Networks

Aman Sharma[1], Ayushi Awasthi[1] and O.S.K.S. Sastri[1]

[1]Department of Physics and Astronomical Science,
Central University of Himachal Pradesh,
Dharamshala 176215, India.
rattanaman7@gmail.com

Submitted on 05-Feb-2026

Abstract

This study demonstrates the application of Physics-informed neural networks (PINNs) to solve the initial-value problem of a two-mass coupled nonlinear oscillator system. The system is governed by the coupled second-order ODEs.

A fully-connected feed-forward neural network is trained to directly approximate the displacement fields $x_1(t)$ and $x_2(t)$. The network is optimized by minimizing a composite loss function consisting of the physics residual and a weighted initial-condition loss. No labeled trajectory data are used to train the model. The PINNs solution is validated against a high-accuracy numerical method. The predicted displacements show excellent quantitative agreement with the numerical solution, achieving sub-millimeter root-mean-square errors for both masses throughout the simulated interval. These results illustrate that PINNs can accurately predict the solutions of nonlinear differential equations.

Keywords: Physics-Informed Neural Networks, Coupled Oscillators, Nonlinear Dynamics, Automatic Differentiation.

1 Introduction

Coupled oscillator systems are fundamental models in classical mechanics, appearing in a wide range of physical phenomena such as molecular vibrations, acoustic meta materials, electrical circuits, and mechanical structures with multiple degrees of freedom. When the restoring forces are linear, the system can be solved analytically using normal mode analysis. However in real world, most of systems exhibit nonlinearities such as cubic stiffness terms that lead to complex behaviors including amplitude-dependent frequencies, internal resonance, and energy transfer between modes. These nonlinear effects are difficult to treat exactly and usually require numerical methods that become computationally expensive for long-time simulations or when many parameters must be explored.

Physics-informed neural networks (PINNs) have emerged as a best alternative for solving both forward and inverse problems governed by differential equations[1]. By embedding the governing equations directly into the loss function of a neural network via automatic differentiation, PINNs can approximate solutions without requiring any labeled simulation data. It requires training data in the form of differential equations. Since their introduction by Raissi et al. [2], PINNs have been successfully applied to a variety of problems including quantum mechanics [3], fluid dynamics & structural dynamics. Their ability to handle strong nonlinearities with relatively small networks makes them particularly attractive for nonlinear dynamical systems.

In this work we apply PINNs to the classic two-mass, three-spring system with an additional cubic nonlinear stiffness term on each mass. We demonstrate that a compact feed-forward neural network is trained solely on the physics residual and initial conditions. It can accurately reproduce the nonlinear beating phenomenon observed in the system over a long time interval. The PINNs solution is validated against a high-accuracy numerical solution obtained with SciPy's `odeint` solver, showing excellent agreement.

This study serves as a clear, reproducible demonstration that PINNs can capture the essential nonlinear dynamics of coupled oscillators with minimal computational overhead and no need for mesh generation or extensive training data. The results highlight the potential of PINNs as a versatile tool for exploring nonlinear mechanical systems, parameter studies, and real-time digital twins.

2 Methodology

In this section, we will explain the working of feed-forward neural network combined with physics informed loss functions shown in figure 1

2.1 Governing Equations

The system under consideration is a two-mass coupled nonlinear oscillator with cubic stiffness nonlinearity. The governing differential equations are

$$m\ddot{x}_1 + k_1 x_1 - k_c(x_2 - x_1) + k_{nl} x_1^3 = 0 \quad (1)$$
$$m\ddot{x}_2 + k_2 x_2 - k_c(x_1 - x_2) + k_{nl} x_2^3 = 0 \quad (2)$$

with symmetric linear stiffness ($k_1 = k_2 = 1$), weak coupling ($k_c = 0.5$), and a small nonlinear coefficient ($k_{nl} = 0.01$). Initial conditions are chosen to excite primarily the first mass: $x_1(0) = 1$, $x_2(0) = 0$, $\dot{x}_1(0) = \dot{x}_2(0) = 0$ [4].

2.2 Physics-Informed Neural Network

A fully connected feedforward neural network [5] in combination with physics informed loss functions was used to approximate the solution pair $(x_1(t), x_2(t))$. The network takes time t as the sole input and outputs a two-dimensional vector. It consists of an input layer with one neuron, three hidden layers each containing 64 neurons with hyperbolic tangent (tanh) activation functions, and an output layer with two neurons.

The neural network is trained iteratively using the following six-step process:

1. **Random Initialization:** At the very beginning, all weights w_{ij} and biases b_j are assigned small random values using normal or uniform distribution.

2. **Forward Pass:** We feed the input through the network, then each neuron computes its weighted sum and activation. Finally, the network gives some prediction $\hat{y}$.

3. **Computing Loss:** Compare prediction with the true output value y and a loss function is used to measure the error.

4. **Back Propagation:** The network calculates gradients of loss w.r.t. each weight and bias using chain rule. Gradient indicates that how much a small change in weight would increase or decrease the loss.

5. **Gradient Descent:** Update each weight in the direction that reduces the loss:
$$\begin{aligned} w_{ij}^{new} &= w_{ij}^{old} - \eta \frac{\partial \mathcal{L}}{\partial w_{ij}}, \\ b_j^{new} &= b_j^{old} - \eta \frac{\partial \mathcal{L}}{\partial b_j}, \end{aligned} \tag{3}$$
where η is a small positive number and denotes learning rate.

6. **Iterations:** Steps 2 to 5 are repeated many number of times called as iterations or epochs. With the time, the weights and biases are adjusted so that the network prediction $\hat{y}$ gets closer to output y and hence loss gets minimized.

The architecture of neural network is shown in Figure 1.

2.3 Loss Function Formulation

The total loss was defined as a weighted sum of the physics residual loss and the initial condition loss:

$$\mathcal{L}(\theta) = \mathcal{L}_{\text{phy}} + \lambda_{\text{IC}} \mathcal{L}_{\text{IC}}, \tag{4}$$

with weighting factor $\lambda_{\text{IC}} = 10.0$.

The physics loss was computed from the mean squared residual of the governing equations evaluated at a fixed set of $N = 200$ uniformly spaced collocation points $t_i \in [0, 16]$:

$$\mathcal{L}_{\text{phy}} = \frac{1}{N} \sum_{i=1}^{N} \Big(r_1(t_i)^2 + r_2(t_i)^2 \Big), \tag{5}$$

where the residuals are:

$$\begin{aligned} r_1(t) &= \ddot{x}_1(t) + x_1(t) - 0.5\big(x_2(t) - x_1(t)\big) \\ &+ 0.01\, x_1(t)^3, \end{aligned}$$

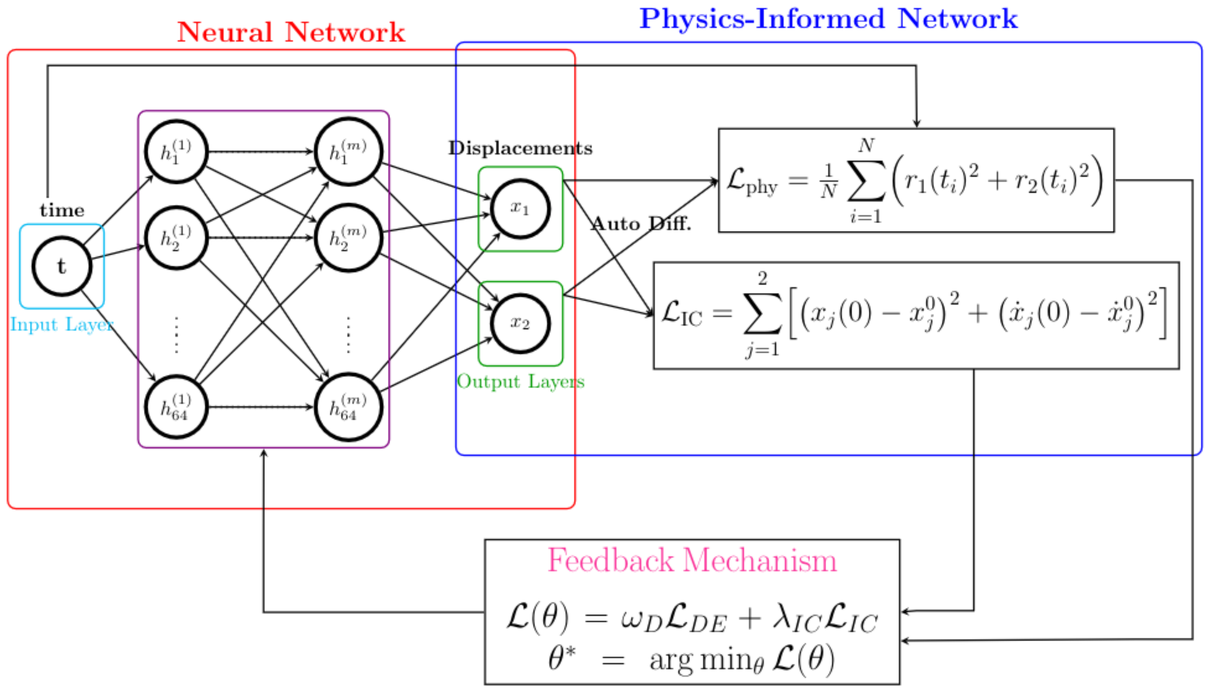

Figure 1: Physics Informed Neural Network Building Blocks

$$\begin{aligned} r_2(t) &= \ddot{x}_2(t) + x_2(t) - 0.5\big(x_1(t) - x_2(t)\big) \\ &+ 0.01\, x_2(t)^3. \end{aligned}$$

First and second-order time derivatives were obtained via automatic differentiation. The initial condition loss enforced the four prescribed values at $t = 0$:

$$\mathcal{L}_{\text{IC}} = \sum_{j=1}^{2}\left[\left(x_j(0) - x_j^0\right)^2 + \left(\dot{x}_j(0) - \dot{x}_j^0\right)^2\right] \quad (6)$$

where $x_1^0 = 1$, $x_2^0 = 0$, $\dot{x}_1^0 = \dot{x}_2^0 = 0$.

Now equation (6) can be expanded as:

$$\begin{aligned} \mathcal{L}_{\text{IC}} &= \left(x_1(0) - 1\right)^2 + \left(x_2(0) - 0\right)^2 \\ &+ \left(\dot{x}_1(0) - 0\right)^2 + \left(\dot{x}_2(0) - 0\right)^2 \quad (7) \end{aligned}$$

2.4 Training Procedure

1. **Optimizer:** Adam optimizer is used with learning rate 10^{-3}.

2. **Collocation points:** 200 uniform points are used within the interval $[0, 16]$

3. **Training steps:** Model is trained over 50,000 iterations.

4. Training history is saved after every 5000 iterations.

5. Intermediate predictions are plotted every 25,000 iterations.

3 Results and Discussion

In this section, we have validated the PINNs solution for coupled spring-mass system using PyTorch. All computations were performed using 64-bit floating-point precision (`torch.float64`) on 11th Gen Intel(R) Core(TM) i5-11300H CPU).

The composite loss function converges steadily over the 50,000 training iterations. Table 1 shows the evolution of the total loss, physics loss, and initial-condition loss on a logarithmic scale. The total loss drops from an initial value of approximately 8–9 to a final value of $\mathbf{1.3771 \times 10^{-5}}$. After 5000 iterations the physics residual becomes the dominant term, while the initial-condition loss remains fluctuating between 10^{-7} and 10^{-5} throughout the later stages.

Epoch	Total Loss	Physics Loss	IC Loss
0	8.5194	0.0813	0.8438
5000	0.0248	0.0247	7.29856×10^{-6}
10000	0.0211	0.0210	5.55193×10^{-6}
15000	0.0169	0.0168	3.41986×10^{-6}
20000	0.0115	0.0113	2.38601×10^{-5}
25000	0.0057	0.0057	9.11852×10^{-7}
30000	0.0002	0.0002	2.14487×10^{-6}
35000	0.0001	0.0001	7.78829×10^{-7}
40000	0.0004	0.0002	2.18778×10^{-5}
45000	0.0007	0.0006	2.00658×10^{-6}
50000	1.3771×10^{-5}	1.23877×10^{-5}	1.3833×10^{-7}

Table 1: Training history of total loss, physics loss, and IC loss.

Figure 2 compares the solution pair $x_1(t)$ and $x_2(t)$ obtained through PINNs predictions represented by solid lines, against the traditional solution obtained through SciPy's `odeint` represented by dashed lines. The two solutions are visually indistinguishable over $t \in [0, 16]$. The nonlinear beating phenomenon is accurately reproduced.

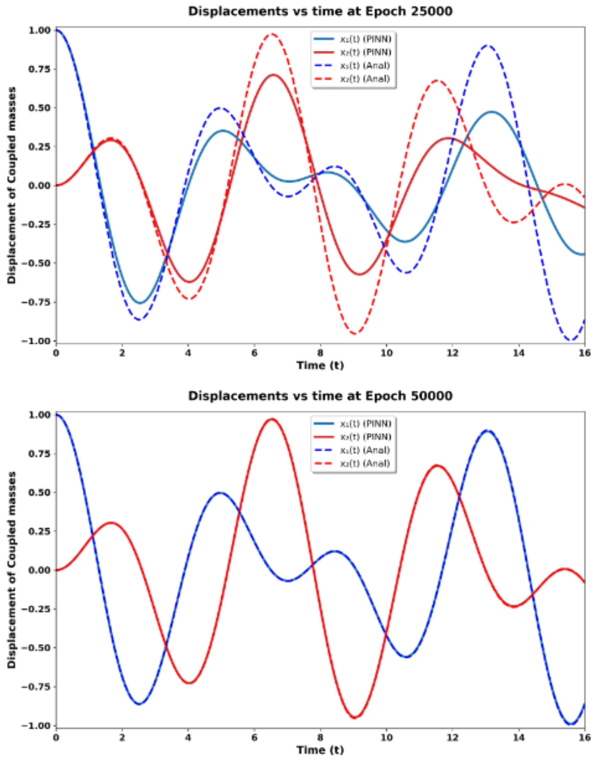

Figure 2: Comparison of PINN-predicted displacements (solid) and numerical reference (dashed) at the final epoch.

The results confirm that a compact PINN can solve the coupled nonlinear oscillator problem to sub-millimeter accuracy using only the governing equations and initial conditions. The network captures both the fast oscillation and the slow energy-transfer timescales without any numerical stiffness.

Advantages of using PINNs are mesh-free formulation, instant evaluation at arbitrary times, and fully differentiable solutions useful for further analysis. For stronger nonlinearities or longer times, adaptive collocation points or larger networks may be required. This study serves as a reproducible benchmark for PINNs performance on nonlinear multiple degree of freedom mechanical systems.

4 Conclusion

This work has demonstrated that PINNs can accurately solve the forward problem of a two-mass coupled nonlinear oscillator governed by second-order ODEs with cubic stiffness nonlinearity. Using a compact fully-connected network and a composite loss function that enforces both the physics residuals and the initial conditions, we obtained solutions that match a high-accuracy numerical solution. The approach is fully mesh-free, differentiable, and instantly evaluable at arbitrary times, making it particularly attractive for nonlinear mechanical systems where traditional numerical integrators may suffer from stiffness or require careful step-size control. The results provide a reproducible benchmark for PINNs performance on multiple degree of freedom nonlinear vibration problems where traditional numerical methods fail to find the solution.

References

[1] J. Xu, J. Ran, and J. She, "The improved pinns for solving partial differential equations with nonlinear couple systems," *Electronic Research Archive*, vol. 34, no. 1, pp. 113–159, 2026.

[2] Raissi *et al. Journal of Computational physics*, vol. 378, 2019.

[3] A. Sharma, A. Awasthi, I. Kant, and O. Sastri, "Solving phase equation using neural network," in *Proceedings of the DAE Symp. on Nucl. Phys*, vol. 69, p. 1219, 2025.

[4] R. Bivins, N. Metropolis, and J. R. Pasta, "Nonlinear coupled oscillators: Modal equation approach," *Journal of Computational Physics*, vol. 12, no. 1, pp. 65–87, 1973.

[5] G. Bebis and M. Georgiopoulos, "Feedforward neural networks," *Ieee Potentials*, vol. 13, no. 4, pp. 27–31, 2002.