Advances in Science and Technology Research Journal

Advances in Science and Technology Research Journal, 2026, 20(2), 1–14
https://doi.org/10.12913/22998624/210480
ISSN 2299-8624, License CC-BY 4.0

Received: 2025.07.20
Accepted: 2025.11.15
Published: 2025.12.23

Duffing system parameter estimation by inverse physics-informed neural networks with sine activation function

Maciej Czarnacki[1*]

[1] Department of Technical Computer Science, Faculty of Mathematics and Information Technology, Lublin University of Technology, ul. Nadbystrzycka 36, 20-618 Lublin, Poland
* Corresponding author's e-mail: m.czarnacki@pollub.pl

ABSTRACT

This work presents the application of the inverse Physics-Informed Neural Networks (PINNs) algorithm for the estimation of parameters of the nonlinear dynamic Duffing system. To achieve fast convergence and accuracy during the neural network training with the smallest possible set of training data, a sine activation function was used. The research provides a comprehensive comparison of the capabilities of inverse PINNs in reproducing the parameters of the studied system using classical activation functions such as sigmoid, tanh, GELU, and the proposed sine function. The issue of training data resources (dense vs. sparse data) is also discussed. This research demonstrates the advantage of the model with the sine activation function when analysing very sparse data across a wide time domain of the Duffing system's solution. However, these studies also indicate the difficulty of estimating the frequency of the periodic driving force in the studied system using a sparse dataset for training. As an example, for a time domain of a chaotic solution of the Duffing equation t = 100 s, with a training set containing only 50 measurement points, PINNs with sine activation can easily reproduce the damping parameter and the oscillator potential, as well as the attractor in phase space, while models with the sigmoid, tanh, or GELU activation function are not even able to converge. Additionally, an attempt was made to reproduce the system's parameters, as well as the time series and the attractor, for noisy data.

Keywords: physics-informed neural networks, PINNs, chaotic dynamical systems, automatic differentiation, Duffing equation, parameter estimation, sine activation function.

INTRODUCTION

Recently, artificial neural networks have been increasingly applied to solve scientific problems. Among these, physics-informed neural networks (PINNs) have emerged as a powerful tool for solving ordinary and partial differential equations (ODEs and PDEs) that describe nonlinear dynamical systems. The foundations for this method were laid in two seminal publications [1, 2]. The main idea of PINNs is to construct a neural network that not only fits measured data but also satisfies the governing differential equations. To achieve this, the loss function is formulated to include terms that penalize deviations from the physical laws, guiding the model to learn solutions that are physically consistent. In their initial papers, the authors formulated the basic principles of PINNs and demonstrated their effectiveness in solving forward and inverse problems for various nonlinear dynamical systems. These works initiated a surge of research in this area, which has been summarized in recent reviews [3–5]. Despite their conceptual simplicity and numerous successes, PINNs also face several significant challenges. The primary challenge lies in the optimization process. The composite loss function is a weighted sum of different terms – typically one for data fidelity and one for physics compliance. A significant imbalance between the gradients of these components can stall the training process. For instance, the network might learn a trivial (e.g., zero) solution that satisfies the differential equation but fails to meet the initial and boundary conditions.

Another challenge is the spectral bias inherent in neural networks. Artificial neural networks tend to learn low-frequency functions more readily than high-frequency ones. This poses a critical problem when the solution to the differential equation contains high-frequency components, sharp gradients, or multi-scale features, often hindering convergence. The network struggles to capture these fine details, resulting in an overly smooth and inaccurate approximation of the true solution [6].

A further issue arises with stiff differential equations. In such systems, gradients from the fast-evolving components of the solution can be orders of magnitude larger than those from the slowly evolving parts. Consequently, the optimizer (e.g., Adam) may cause the network to focus on minimizing the error from the fast dynamics while neglecting the slower components. This can lead to a scenario where the loss function decreases, but only because the network has adapted to one aspect of the physics, failing to learn the system's complete and complex dynamics [7].

Several strategies have been proposed to address these limitations. One approach involves using adaptive algorithms that dynamically adjust the weights of the loss components during training. Another strategy is domain decomposition, which divides the problem's temporal or spatial domain into smaller subdomains. Multiple neural networks are then trained in parallel, each responsible for approximating the solution within one subdomain. After training, these individual solutions are assembled to form the complete solution for the entire domain. Such techniques are described in detail in the literature [8] and [9].

A promising approach to mitigate the issues is the use of Fourier feature mapping (FFM), a technique that transforms low-dimensional input data into a higher-dimensional feature space using a basis of sine and cosine functions of various frequencies [10]. Analogous to a Fourier transform, this mapping decomposes the input signal into its constituent frequency components. This allows the neural network to learn a function in this new space where high-frequency components are explicitly represented and therefore easier to approximate, thereby helping to overcome the network's inherent spectral bias.

In this work, is employed a simplified variation of this technique. Instead of a full feature mapping, the sine function is used as the activation function in the first layer of the neural network. This approach serves a similar purpose to FFM and is a core characteristic of Sinusoidal Representation Networks (SIRENs), as described in [11]. The method has demonstrated considerable success in the context of PINNs, particularly for modelling nonlinear dynamical systems [12–14] including the modeling of hydrodynamic systems described by the Navier-Stokes equations [15, 16].

This publication aims to develop an innovative tool for analyzing nonlinear dynamic systems with an extremely minimal amount of input data which is crucial for easier and more efficient analysis of complex dynamic systems in various fields of science and engineering. The use of a sine activation function achieves this goal by eliminating model convergence problems during the learning process related to the phenomena described above, particularly spectral bias. The final discussion presents a comparison of this method to PINNs based on classical activation functions (tanh, elu) [17] and to the popular Sparse Identification of Nonlinear Dynamics (SINDy) algorithm and its variants, used to identify the dynamics of nonlinear systems [18] and [19].

The following sections of the article present the principles of the PINNs algorithm, a description of the selected neural network model, and the dynamical system chosen for the study – the Duffing dynamical system. The results of the study are then presented, followed by a comprehensive discussion of the obtained results in the context of the literature on the subject.

METHODOLOGY

Consider a general nonlinear partial differential equation (PDE) with initial and boundary conditions defined as follows:

$$\begin{aligned} D(u(x,t);\lambda) = f(x,t),\ (x,t) \in \Omega \times [0,T], \\ u(x,0) = u_0(x), x \in \Omega, \\ Bu(x,t) = g(x,t),\ (x,t) \in \partial\Omega \times [0,T], \end{aligned} \tag{1}$$

where: $u(x,t)$ is the solution; $x \in R^d$ represents the spatial variables; $t \in [0, T]$ is the time domain; $\Omega \subset R^d$ is the spatial domain with boundary $\partial\Omega$; D is a differential operator; $f(x,t)$ is a source function; B is a boundary operator; and $g(x,t)$ defines the boundary values. The system may also depend on a set of parameters, denoted by the vector λ.

The PINN model

In the PINN framework, the unknown function u (x,t) is approximated by a deep neural network, $u_{NN}(x, t; \theta)$. This network takes the independent variables x and t as input and outputs an approximation of the solution. The network is parameterized by a vector θ , which consists of all its weights and biases. For inverse problems, the system parameters λ can also be included alongside θ as trainable variables, allowing the model to identify them from data and physics.

A cornerstone of the PINN methodology is automatic differentiation (AD), implemented in modern deep learning frameworks such as PyTorch [20] and TensorFlow [21]. AD allows for the precise and efficient computation of the derivatives of the network's output with respect to its inputs (x and t). This mechanism is crucial for defining the PDE residual, which measures how well the network's approximation satisfies the governing equation:

$$R(x,t;\theta;\ \lambda) = D\ u_{NN}(x,t;\theta) - f(x,t,\lambda) \quad (2)$$

The loss function

The training of a PINN involves minimizing a composite loss function, which is a weighted sum of several components. These terms represent the PDE residual, the initial and boundary conditions, and any available measurement data. The total loss function has the form:

$$L(\theta,\lambda) = w_r L_r(\theta,\lambda)\ + w_{ic}L_{ic}(\theta) + \\ +w_{bc}L_{bc}(\theta) + w_d L_d(\theta) \quad (3)$$

where: L_r is the loss from the PDE residual, L_{ic} is the loss from the initial conditions, L_{bc} is the loss from the boundary conditions, L_d is the loss from the measurement data, if available, w_r, w_{ic}, w_{bc}, w_d are weights that control the relative importance of each component and can help balance terms of different magnitudes or physical units. The selection of these weights is crucial for effective training.

Each loss component is typically defined as the mean squared error (MSE) over a set of training points. The residual loss L_r enforces the PDE at a set of collocation points distributed throughout the domain Ω × [0, T]:

$$L_r = \frac{1}{N_r}\sum_{i=1}^{N_r}\left\| \begin{matrix} D\left(u_{NN}\left(x_r^{(i)},t_r^{(i)};\theta\right);\lambda\right) \\ -f\left(x_r^{(i)},t_r^{(i)}\right) \end{matrix} \right\|^2 \quad (4)$$

where: $\left(x_r^{(i)},t_r^{(i)}\right)$ are the N_r collocation points.

The initial condition loss L_{ic} is calculated at time t = 0:

$$L_{ic} = \frac{1}{N_{ic}}\sum_{i=1}^{N_{ic}}\left\| \begin{matrix} u_{NN}\left(x_{ic}^{(i)},0;\theta\right) \\ -u_0\left(x_{ic}^{(i)}\right) \end{matrix} \right\|^2 \quad (5)$$

where: $x_{ic}^{(i)} \in \Omega$ are the N_{ic} points sampled from the initial domain.

The boundary condition loss L_{bc} is calculated on the domain boundary:

$$L_{bc} = \frac{1}{N_{bc}}\sum_{i=1}^{N_{bc}}\left\| \begin{matrix} Bu_{NN}\left(x_{bc}^{(i)},t_{bc}^{(i)};\theta\right) \\ -g\left(x_{bc}^{(i)},t_{bc}^{(i)}\right) \end{matrix} \right\|^2 \quad (6)$$

where: $\left(x_{bc}^{(i)},t_{bc}^{(i)}\right)$ are the N_{bc} points sampled from the boundary $\partial\Omega$ × [0, T] . The data loss L_b is included when direct measurements of the solution are available:

$$L_d = \frac{1}{N_d}\sum_{i=1}^{N_d}\left\| \begin{matrix} u_{NN}\left(x_d^{(i)},t_d^{(i)};\theta\right) \\ -\ u_d^{(i)} \end{matrix} \right\|^2 \quad (7)$$

where: $\left(x_d^{(i)},t_d^{(i)},u_d^{(i)}\right)$ are the N_d measurement data points.

The total loss function L(θ, λ) is minimized using gradient-based optimization algorithms, such as Adam or L-BFGS, which leverage backpropagation. This process iteratively updates the trainable parameters (θ and, optionally, λ) to reduce the error across all components. Upon successful training, the network with the optimized parameters θ^*, denoted as $u_{NN}(x, T; \theta^*)$, provides an approximation of the true solution u(x,t). For inverse problems, the optimized vector λ^* represents the identified system parameters.

Duffing oscillator

The Duffing oscillator stands as a quintessential example of a nonlinear dynamic system, characterized by an exceptionally rich range of

solutions [22]. Its applications extend beyond the analysis of harmonic vibrations, encompassing fields such as the modelling of stiffening springs, nonlinear electronic circuits, plasma physics, and superconducting Josephson amplifiers [23]. Furthermore, this system finds utility in neurobiology, particularly in modelling the behaviour of neuronal cells during the analysis of epileptic seizures [24].

The dimensionless Duffing equation is given by the formula:

$$\ddot{x} + \delta\dot{x} + \alpha x + \beta x^3 = \gamma cos(\omega t) \qquad (8)$$

where: $x(t)$ represents the solution to the equation, while $\dot{x}$ and $\ddot{x}$ denote the first and second derivatives of x with respect to time, respectively. The symbols δ, a and β correspond to the damping coefficient, the linear stiffness parameter, and the nonlinear restoring force parameter, in that order. Meanwhile, γ and ω signify the amplitude and frequency of the external driving force acting on the system. The dynamics of the Duffing system are extensively documented in the relevant literature [23]. In this paper, the equation's parameters and the trajectory of the driving force are reconstructed by an inverse physics-informed neural networks. In the context of the neural network applications considered here, it is worth mentioning the publication presenting an analytical approach to the use of the Duffing system to detect and characterize periodic vibrations [25].

Methodology for inverse problem analysis

To investigate the applicability of the Physics-Informed Neural Networks (PINNs) algorithm to an inverse problem, two distinct sets of equation parameters and initial conditions were selected, as detailed in Table 1.

The training data were obtained through computational simulation by solving the Duffing equation using the fourth-order Runge-Kutta method (RK4). This simulation was performed with a time step of $dt = 0.001$ over $N = 100{,}000$ steps, which defined the time domain of the solution as $t \in [0, 100]$ Each training dataset comprised 50 equispaced points of the numerical solution. The chosen parameter sets corresponded to distinct dynamic behaviors of the Duffing system: one represented a nonlinear oscillator without damping, and the other a nonlinear oscillator with damping and a driving force exhibiting chaotic vibrations. Additionally, for the chaotic vibration scenario, noisy signal were generated from normal distribution with a standard deviation of $\sigma = 0.15$, equivalent to 10% of the system's vibration amplitude. Due to the high complexity involved in analyzing noisy signals, 1000 training data points were utilized across the entire time domain for this specific case.

Neural network model and training

A fully connected feedforward neural network was implemented using the PyTorch framework. The network had one input neuron representing time, two output neurons representing position and velocity of the system, and four hidden layers, each with 36 neurons. Different activation functions were tested: sigmoid, tanh, GELU, and a hybrid configuration in which a sine activation was applied to the first hidden layer and tanh to the remaining layers. The output layer employed a linear activation function to ensure that the range of values generated by the neural network remained unrestricted. In accordance with the PINNs approach, automatic differentiation was employed during training to compute the time derivatives of the output vectors. These derivatives were used to construct the physics-based component of the loss function, representing the residual of the Duffing differential equation.

The dynamic system parameters were introduced as trainable variables, initialized with values of 1. Automatic differentiation was performed on a time vector with 5.000 collocation points for parameter estimation and 10.000 points for reconstructing the driving force time series. Training was carried out for 50,000 epochs using the Adam optimizer with a learning rate of 0.001. When the loss value fell below 0.005, the learning rate was reduced to 0.0001 to improve convergence.

The selection of loss function weights was carried out through a series of trials, initially assigning a value of 1 to each weight. The primary criterion for evaluation was the convergence of the model and the stability of the estimated parameters during training. For weights $w_r = 0.05$, $w_{ic} = 0.05$, $w_d = 10$, the training process and parameter estimation were found to be optimal. Choosing weights of the correct order of magnitude is

Table 1. Duffing dynamical system parameters and initial conditions for the experiment.

Solution type	δ	α	β	γ	ω	x(0)	v(0)
Periodic	0.0	1.0	-1.0	0.0	0.0	0.0	1.0
Chaotic	-0.1	1.0	-1.0	0.38	1.4	0.0	0.0

crucial, as it ensures that the model learns not only from the experimental data but also from the underlying physics encoded in the loss function.

All computations were performed on a system equipped with an AMD Ryzen 5 9600X processor, 8 GB of RAM, and an Nvidia RTX 4060 GPU. On this hardware, training the neural network for 50,000 epochs required no more than 6 minutes. This short runtime highlights the efficiency of the proposed method, making it an attractive candidate for practical applications.

RESULTS

Initially, a comparative test of neural network training with different activation functions was conducted. The objective of this test was to identify the optimal network structure for subsequent experiments. The evolution of the loss function for the analysed activation functions is presented in Figure 1. It is observed that for such a small training data set (50 points), neural networks employing classical activation functions, such as sigmoid, tanh, and GELU, were unable to achieve satisfactory convergence, thereby disqualifying them for this type of application. For instance, in the case of the network utilizing hyperbolic tangent activation, the mean square error (MSE) was 0.22 for a test set comprising 1000 points. Only the application of a periodic activation function facilitated good convergence, yielding an MSE value of 1.46×10^{-5} for the test set. This indicates that the MSE for the periodic activation function is four orders of magnitude smaller than for the tanh activation. For comparison, Figures 2 and 3 illustrate the estimated parameters of the Duffing equation for the tanh activation function and the sine function, respectively. The lack of convergence observed with classical activation functions results in a poor-quality estimation of the investigated system's parameters.

Subsequently, the parameters of the tested dynamic system exhibiting chaotic behaviour were reconstructed. Such a system is characterized by five non-zero parameters: linear damping, linear and nonlinear elasticity, and the amplitude and frequency of the driving force. As

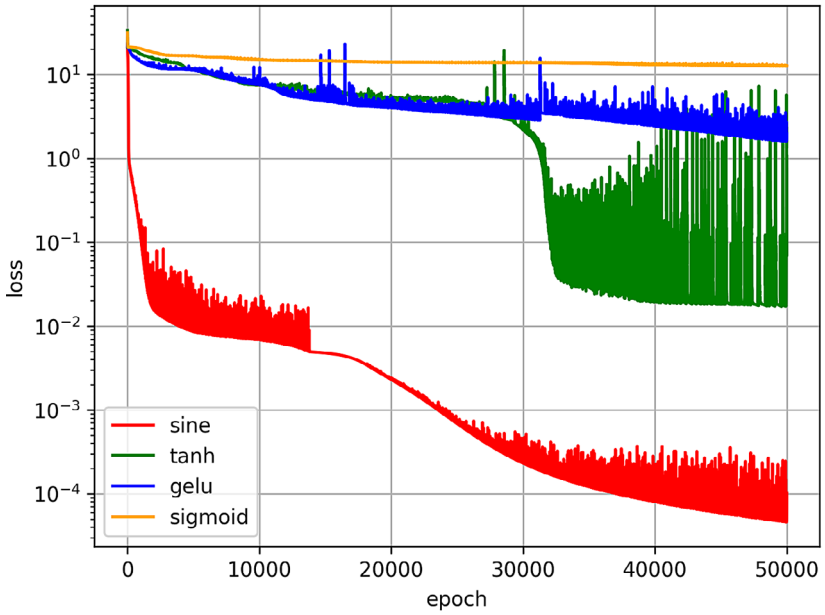

Figure 1. Convergence test of the learning process for sigmoid, tanh, GELU and sine activation functions showing losses as a function of epochs

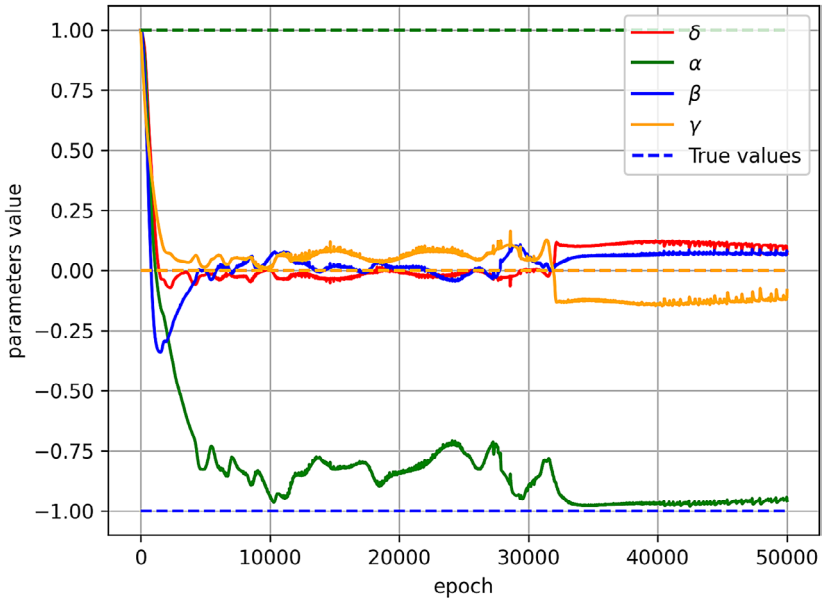

Figure 2. System parameter estimation curves for a neural network with tanh activation functions only

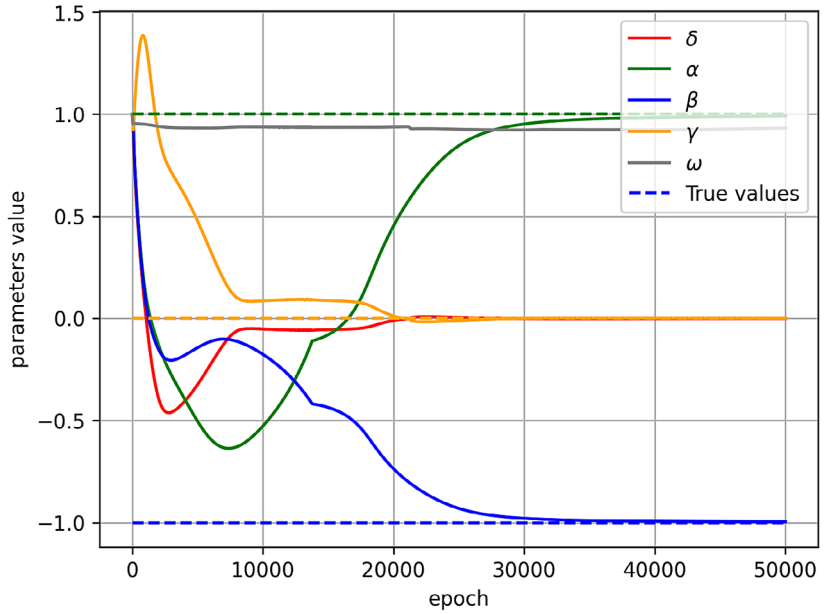

Figure 3. System parameter estimation curves for a neural network with sine activation function for periodic solution

depicted in Figure 4 showing the loss function, good convergence was not achieved, resulting in poor estimation of the system's parameters (Figure 5). This indicated that the problem's complexity is too high. This limitation arises because the effect of the forcing frequency on the system's response is strongly coupled with the linear stiffness and damping parameters, shifts in frequency can be compensated by adjustments in these parameters, leading to nearly indistinguishable trajectories. As a result, the inverse problem becomes ill-posed – multiple parameter combinations can explain the observed data equally well – and the network fails to uniquely identify the excitation frequency without additional constraints or prior information. A short proposal for a new broader algorithm for the future is included in the discussion section.

To address this, the problem's complexity was reduced by setting the frequency of the driving force as a known parameter. Figure 6 and 7 illustrates the loss function and estimated

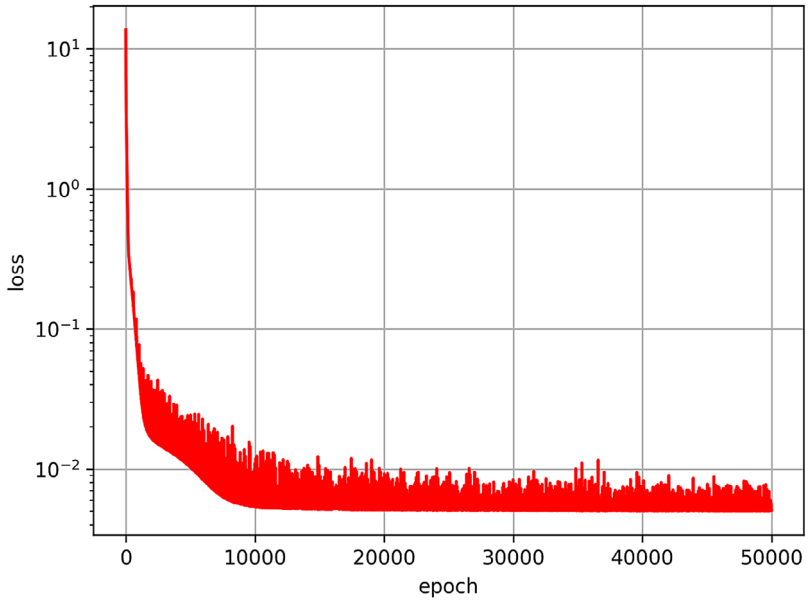

Figure 4. Loss function curve of the neural network model with a sine activation function, illustrating poor convergence caused by coupling between the driving force and the other system parameters

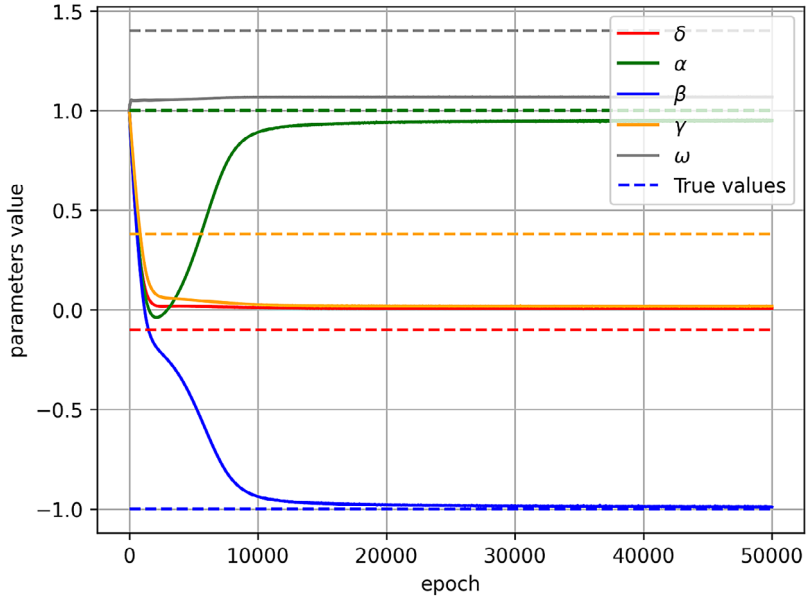

Figure 5. System parameter estimation curves for a neural network with a sine activation function under a chaotic solution. The absence of clear convergence is attributed to coupling between the driving force and the other system parameters

values of the system parameters during the neural network training process under this modified condition. To obtain good convergence, 25,000 epochs were sufficient. It is observed that eliminating the driving force frequency parameter led to good model convergence, with an MSE of 5.35×10^{-5} for 1000 testing points, and accurate estimation of the remaining system parameters. The estimated system parameter values are listed in Table 2.

It can be noticed that PINNs is not able to estimate the value of the frequency of the driving force, however, for the case of a periodic solution, the value of the estimated frequency is irrelevant due to the zero amplitude of this force.

The time series of position and velocity and reconstructed phase portrait are shown in Figures 8, 9 and 10. The figures below show a high agreement of the obtained results with the ground true numerical solution of the differential equation.

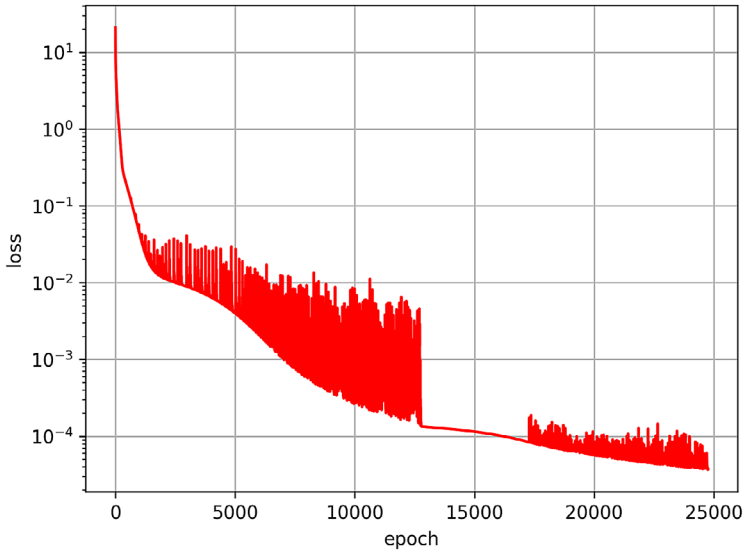

Figure 6. Loss function curve of the neural network model with a sine activation function for a fixed value of the driving force frequency

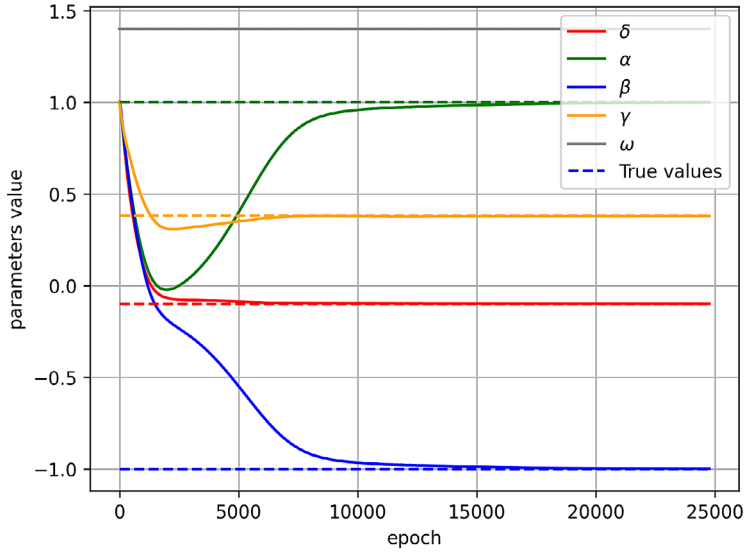

Figure 7. System parameter estimation curves for a neural network with a sine activation function under a chaotic solution for a fixed value of the driving force frequency

Table 2. Estimated parameters of the Duffing system

Solution type	δ	α	β	γ	ω
Periodic	0.0004	0.994	-0.994	0.0006	0.936
Chaotic	-0.101	0.992	-0.991	0.382	-

Next, an experiment is conducted to reconstruct the time series of the driving force. For this purpose, all system parameters, except driving force amplitude and frequency, were treated as known. The driving force is defined as a learnable parameter, represented as a vector whose number of elements matched the number of collocation points. This vector replaced the explicit form of the force within the loss function, which accounted for the system's dynamics. The initial values of the elements in this vector are randomly generated from a normal distribution with a standard deviation of 0.25. The result of the driving force reconstruction is presented in Figure 11. The MSE for the estimated

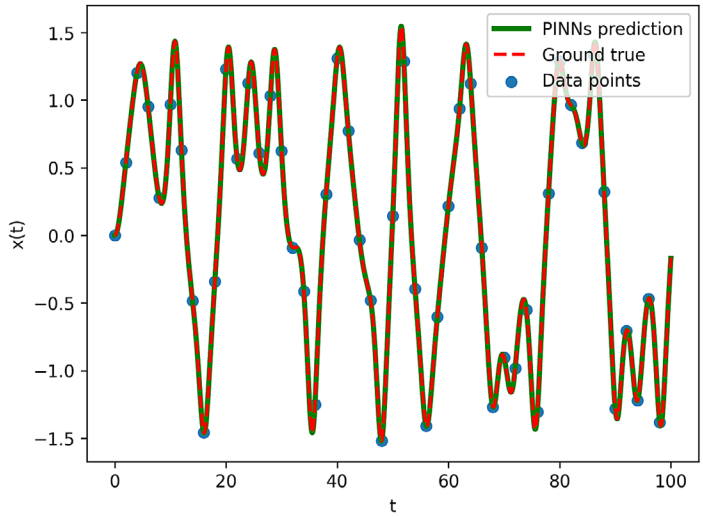

Figure 8. Comparison of the PINNs predicted position over time with the ground true solution and data points for chaotic solution

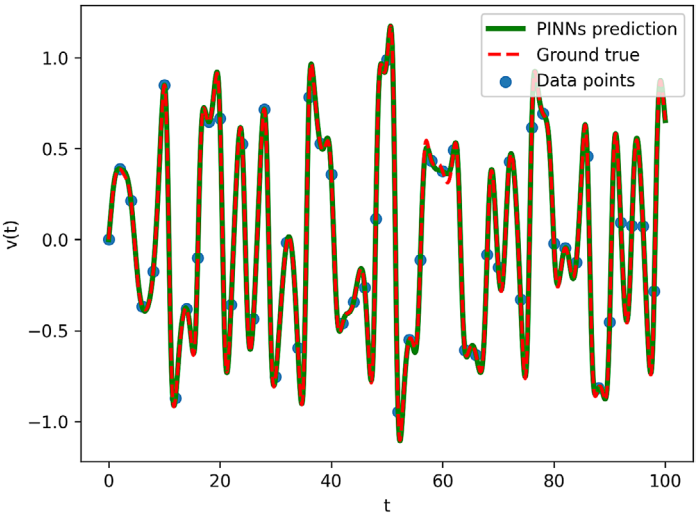

Figure 9. Comparison of the PINNs predicted velocity over time with the ground true solution and data points for chaotic solution

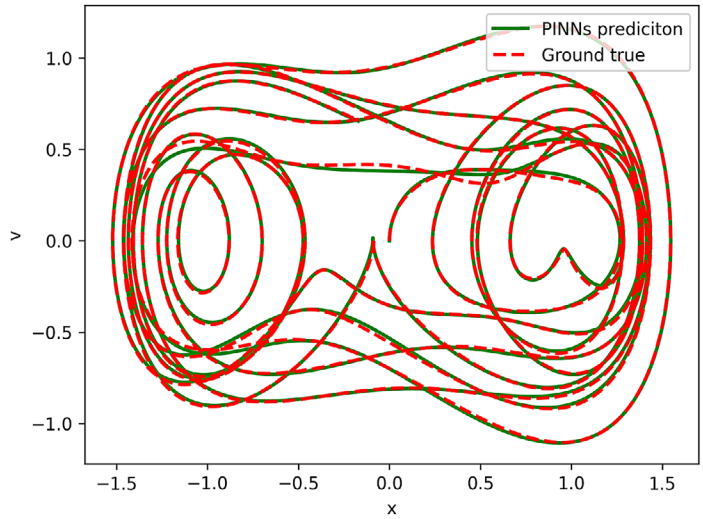

Figure 10. Reconstructed phase portrait of the Duffing system: PINNs prediction vs ground true solution

driving force is 0.0002 for 10000 collocation points. The final test involved estimating the Duffing system's parameters using training data corrupted with noise. The noise is generated from a normal distribution with a standard deviation of 0.15, which is approximately 10% of the system's vibration amplitude and velocity. The Figures 12 and 13 show the noisy training data for position and velocity, respectively. The system's reconstructed phase portrait and the values of the estimated system parameters are presented in Figure 14 and Table 3, respectively. The reconstructed phase portrait exhibited an MSE of 0.0019 for 1000 testing points, calculated with respect to the original (undistorted) portrait.

DISCUSSION

PINNs are a very efficient tool for analysing nonlinear dynamic systems described by differential equations [1–3]. In the literature, there are studies in which PINNs have been used to analyse the Duffing equation [13, 17]. The first of these works focuses exclusively on the forward PINNs problem, where the authors successfully found solutions to the Duffing equation but within a narrow time domain. In contrast, the second study [17] presents an effective approach to retrieving the forcing term in the inverse problem. To achieve this, the authors constructed a dense neural network composed of 10 layers with 1, 15, 30,

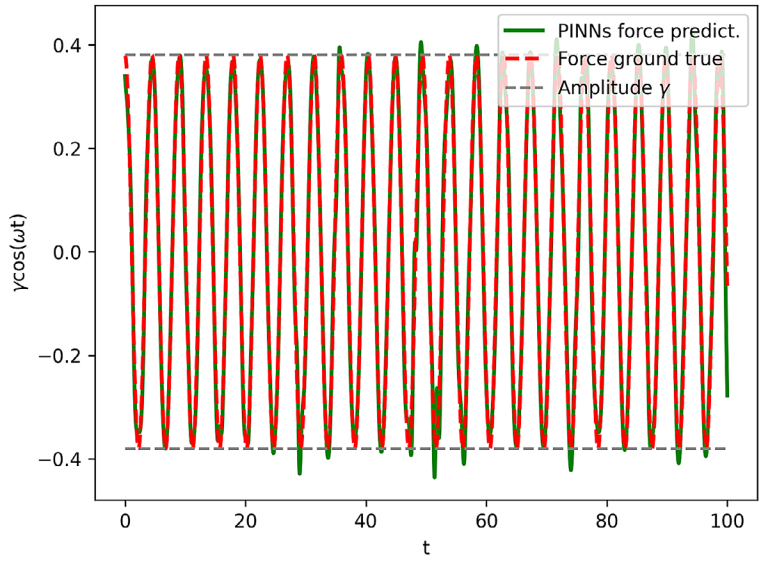

Figure 11. Reconstructed driving force of the Duffing system: PINNs prediction vs ground true

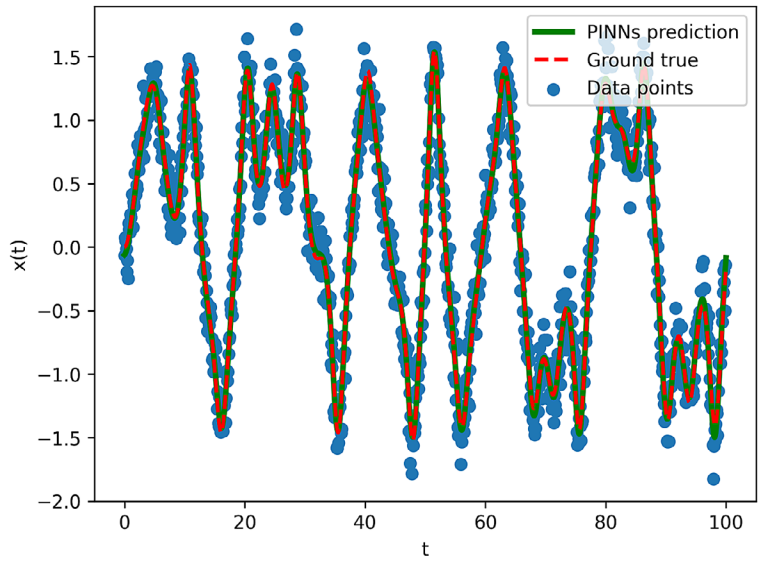

Figure 12. Noisy data points of oscillator position vs ground true simulation and PINNs prediction

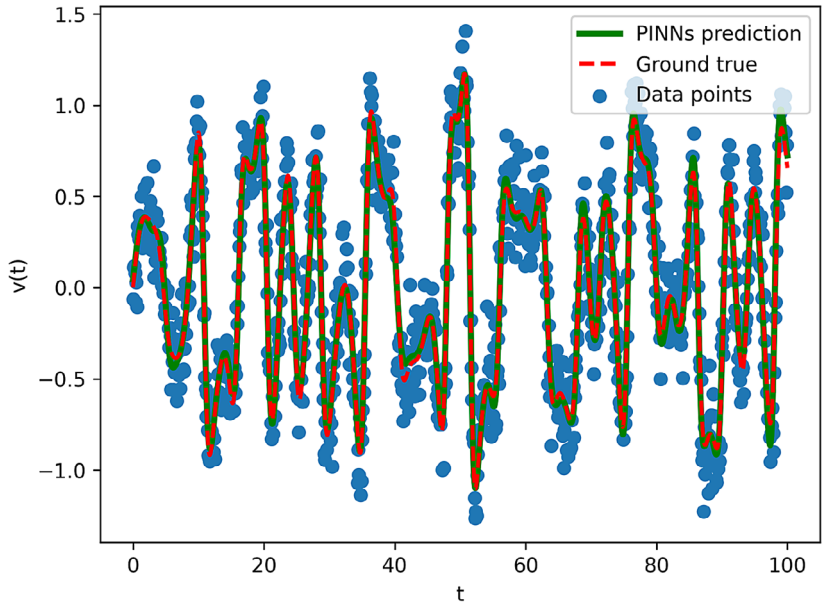

Figure 13. Noisy data points of oscillator velocity vs ground true simulation and PINNs prediction

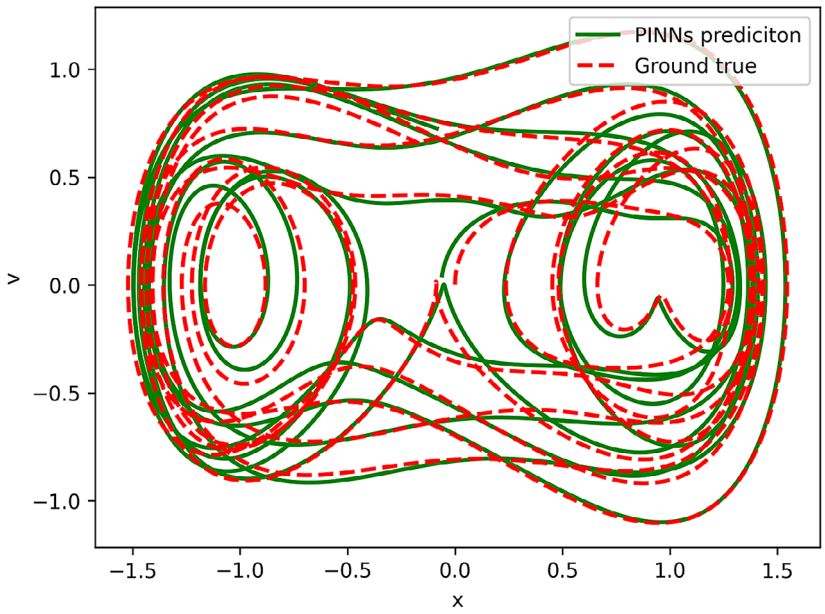

Figure 14. Reconstructed phase portrait of the Duffing system from noisy data: PINNs prediction vs ground true solution

Tabel 3. Estimated Duffing system parameters from noisy data

Solution type	δ	α	β	γ	ω
Chaotic	-0.098	1.012	-1.013	0.374	-

60, 120, 240, 120, 60, 30, 15, and 3 neurons, employing the eLU activation function in each layer. The analysis was performed over a time domain of 50 seconds, and the training dataset was significantly larger than in this study (the domain is twice as small and the amount of required training data is ten times greater than that presented in this work). The use of periodic activation functions ensures faster convergence with less training data and a significantly smaller neural network size.

Additionally, analyzing publications presenting the application of the Sparse Identification

of Nonlinear Dynamics (SINDy) algorithm in the classic publication on the topic [18, 19] and applying this method to the analysis of the Duffing system, it is noted that this method requires as many as 700 measurement points to identify the system parameters in a very narrow time domain of 15s. This fact shows the inverse PINNs algorithm with a periodic activation function as a highly effective and competitive analytical tool in relation to widely used methods (SINDy). The authors of work [19] omitted the discussion of the impossibility of determining the frequency of the driving force and in their calculations assumed this parameter of the system to be known without any description and analysis.

The results presented in this paper show that employing the sine function as the activation function is a key solution to addressing the challenge posed by very sparse training data sets. To achieve good convergence and small mean squared error in the reconstructed phase portrait and the parameters of the described dynamic system, only 50 measurement points over a wide time domain $t \in [0, 100]$ are sufficient. Furthermore, the use of a periodic activation function helps to mitigate the spectral bias problem, which can be observed by comparing solutions for models with and without the sinusoidal activation. One might expect that employing a periodic activation function could introduce multiple minima during the optimization process of neural network parameters. However, the results show that this problem, if it occurs at all, is practically imperceptible. The nonlinear Duffing oscillator is a very complex system, for which it is not possible to solve the inverse problem for all its parameters at once. The presence of a periodic forcing term prevents accurate estimation of the remaining parameters. This makes the inverse problem too challenging for PINNs. However, when driving force frequency is known, it is possible to accurately reconstruct the other parameters of the system.

Additionally, the algorithm allows for the reconstruction of the driving force time series when the rest of the system's parameters are known. As a direction for future work, the identifiability of the forcing frequency could be improved by combining PINNs with a classical solver. In such a hybrid scheme, candidate frequencies would be sampled, the PINN would infer the remaining parameters, and the trajectories would be validated via RK4 integration. This approach may reduce parameter coupling and provide a more robust estimation framework. The results demonstrate that PINNs can be successfully used to reconstruct time series for nonlinear systems from very sparse and noisy data.

In the proposed neural network architecture choice can be regarded as an implicit form of Fourier feature mapping: by introducing a sinusoidal nonlinearity to the linearly transformed inputs, the representation space is enriched with periodic basis functions, analogous to explicit Fourier embeddings commonly employed in PINNs and coordinate networks. Since the argument of the sine includes both a learned linear transformation of the inputs and an additive bias term, the activation is effectively equivalent to using a cosine. The bias can be interpreted as a phase shift, and sine and cosine differ only by such a phase. In this way, the network gains access to a richer set of harmonic features, which facilitates the representation of oscillatory dynamics without requiring hand-crafted embeddings. Furthermore, the number of neurons in the first hidden layer plays a role directly comparable to the feature number in Fourier feature mapping, as it determines how many distinct periodic components can be represented by the model. Note, however, that there is an important distinction between using a learned sinusoidal layer (SIREN) and applying a fixed/random Fourier feature mapping prior to a standard multilayer perceptron: Fourier feature methods typically use a fixed projection to enforce a chosen spectral bandwidth and improve generalization to high-frequency components, whereas SIREN learns the projection weights and therefore trades explicit spectral control for flexibility and smoother derivatives [10, 11].

Future work will extend this investigation to other dynamical systems to gain a broader understanding of spectral bias and methods to mitigate it.

CONCLUSIONS

As shown in this paper Physics-Informed Neural Network algorithms are increasingly used to analyse complex problems in the nonlinear dynamics of physical systems. The applicability of this method, combined with the incorporation of a periodic activation function into the network architecture, results in improved efficiency when analysing systems with

very sparse measurement data. This approach avoids convergence issues related to spectral bias and significantly improves convergence speed. Based on the analysed Duffing system, the method's usefulness for investigating chaotic behaviour in dynamic systems – including those affected by noise. The simultaneous estimation of all system parameters presented a significant challenge in this work, highlighting an area for future research. A strong coupling exists between the driving and damping forces, in addition to the linear spring force components, which hinders their unambiguous identification. Consequently, this necessitates the development of a more advanced and comprehensive Physics-Informed Neural Network algorithm.

REFERENCES

1. Raissi M., Karniadakis GE. Hidden physics models: Machine learning of nonlinear partial differential equations. J Comput Phys. 2018 Mar; 357: 125–141. https://doi.org/10.1016/j.jcp.2017.11.039
2. Raissi M., Perdikaris P., Karniadakis GE. Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations. J Comput Phys. 2019 Feb; 378: 686–707. https://doi.org/10.1016/j.jcp.2018.10.045
3. Cuomo S., Di Cola VS., Giampaolo F., Rozza G. Scientific machine learning through physics–informed neural networks: where we are and what's next. J Sci Comput. 2022 Sep; 92(3): 88. https://doi.org/10.1007/s10915-022-01939-z
4. Karniadakis GE, Kevrekidis IG, Lu L, Perdikaris P, Wang S, Yang L. Physics-informed machine learning. Nat Rev Phys. 2021 Jun; 3(6): 422–440. https://doi.org/10.1038/s42254-021-00314-5
5. Lawal ZK, Yassin H, Lai DTC, Che Idris A. Physics-informed neural network (PINN) evolution and beyond: a systematic literature review and bibliometric analysis. Big Data Cogn Comput. 2022 Dec; 6(4): 140. https://doi.org/10.3390/bdcc6040140
6. Rahaman N, Baratin A, Arpit D, Draxler F, Lin M, Hamprecht F, et al. On the spectral bias of neural networks. In: Chaudhuri K, Salakhutdinov R, editors. Proceedings of the 36th International Conference on Machine Learning; 2019; 5301–10. https://proceedings.mlr.press/v97/rahaman19a.html
7. Wang S, Teng Y, Perdikaris P. Understanding and Mitigating Gradient Flow Pathologies in Physics-Informed Neural Networks. SIAM J Sci Comput. 2021; 43(5): A3055–A3081. https://doi.org/10.1137/20M1318043
8. Jagtap AD, Karniadakis GE. Extended physics-informed neural networks (XPINNs): a generalized space-time domain decomposition based deep learning framework for nonlinear partial differential equations. Commun Comput Phys. 2020 Nov; 28(5): 2002–2041. https://doi.org/10.4208/cicp.OA-2020-0164
9. Moseley B, Markham A, Nissen-Meyer T. Finite basis physics-informed neural networks (FBPINNs): a scalable domain decomposition approach for solving differential equations. Adv Comput Math. 2023 Dec; 49(6): 62. https://doi.org/10.1007/s10444-023-10065-9
10. Tancik M, Srinivasan PP, Mildenhall B, Fridovich-Keil S, Raghavan N, Singhal U, et al. Fourier features let networks learn high frequency functions in low dimensional domains. NeurIPS 2020. https://doi.org/10.48550/arxiv.2006.10739
11. Sitzmann V, Martel JN, Bergman AW, Lindell DB, Wetzstein G. Implicit neural representations with periodic activation functions. NeurIPS 2020. https://doi.org/10.48550/arXiv.2006.09661
12. Faroughi SA, Soltanmohammad R, Datta P, Mahjour SK, Faroughi S. Physics-informed Neural Networks with Periodic Activation Functions for Solute Transport in Heterogeneous Porous Media. 2023. https://doi.org/10.48550/arXiv.2212.0896
13. Raj N. Physics Informed Neural Network for Solution of Duffing Oscillators. In: Saha A, Banerjee S, editors. Proceedings of the 2nd International Conference on Nonlinear Dynamics and Applications (ICNDA 2024), Volume 3. Cham: Springer; 2024. https://doi.org/10.1007/978-3-031-69146-1_14
14. Wong JC, Ooi CC, Gupta A, Ong YS. Learning in sinusoidal spaces with physics-informed neural networks. IEEE Trans Artif Intell. 2024 Mar; 5(3): 985–1000. https://doi.org/10.1109/TAI.2022.3192362
15. Miotto RF, Wolf WR, Zigunov F. Pressure field reconstruction with SIREN. Exp Fluids 2025; 66: 151. https://doi.org/10.1007/s00348-025-04074-1
16. Khademi A, Dufour S. Physics-informed neural networks with trainable sinusoidal activation functions for approximating the solutions of the Navier-Stokes equations. Comput Phys Commun 2025; 314: 109672. https://doi.org/10.1016/j.cpc.2025.109672
17. Shaikh SA, Cherukuri H, Khan T. Recovering the forcing function in systems with one degree of freedom using ANN and physics information. Algorithms. 2023 May; 16(5): 250. https://doi.org/10.3390/a16050250
18. Brunton SL, Proctor JL, Kutz JN. Discovering governing equations from data by sparse identification of nonlinear dynamical systems. Proc Natl Acad Sci U S A. 2016; 113(15): 3932–3937. https://doi.org/10.1073/pnas.1517384113

19. Hasan A. Discovering state-space representation of dynamical systems from noisy data. IEEE Access. 2024; 12: 108744–108754. https://doi.org/10.1109/ACCESS.2024.3438932
20. Paszke A, Gross S, Chintala S, Chanan G, Yang E, DeVito Z, et al. Automatic differentiation in PyTorch [Internet]. Paper presented at: NIPS 2017 Workshop on Autodiff; 2017. Available from: https://openreview.net/pdf?id=BJJsrmfCZ
21. Abadi M, Barham P, Chen J, Chen Z, Davis A, Dean J, et al. TensorFlow: a system for large-scale machine learning. In: Proceedings of the 12th USENIX Conference on Operating Systems Design and Implementation (OSDI'16). USA: USENIX Association; 2016; 265–283. https://www.usenix.org/conference/osdi16/technical-sessions/presentation/abadi
22. Duffing G. Erzwungene Schwingung bei veränderlicher Eigenfrequenz und ihre technische Bedeutung. Braunschweig: Vieweg; 1918.
23. Korsch HJ, Jodl H-J, Hartmann T. Chaos: A Program Collection for the PC. Third revised and enlarged edition. Berlin: Springer Science & Business Media; 2007. https://link.springer.com/content/pdf/10.1007/978-3-540-74867-0.pdf
24. afarian A, Freestone DR, Nesic D, Grayden DB. Slow-Fast Duffing Neural Mass Model. Annu Int Conf IEEE Eng Med Biol Soc. 2019:142–145. https://doi.org/10.1109/EMBC.2019.8857316
25. Martynyuk V, Fedula M, Balov O. Periodic signal detection with using Duffing system Poincare map analysis. Advances in Science and Technology Research Journal. 2014; 8(22): 26–30. https://doi.org/10.12913/22998624.1105158