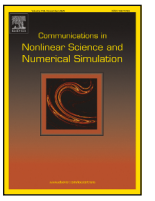
Commun Nonlinear Sci Numer Simulat 158 (2026) 109866

Contents lists available at ScienceDirect

Communications in Nonlinear Science and Numerical Simulation

journal homepage: www.elsevier.com/locate/cnsns



Research paper

Physics-informed neural networks with dynamical boundary constraints

Andrés Martínez-Esteban [a,b], Pablo Calvo-Barlés [b,c], Luis Martín-Moreno [b,c], Sergio G. Rodrigo [a,b,*]

[a] Departamento de Física Aplicada, Universidad de Zaragoza, Zaragoza, 50009, Spain
[b] Instituto de Nanociencia y Materiales de Aragón (INMA), CSIC-Universidad de Zaragoza, Zaragoza, 50009, Spain
[c] Departamento de Física de la Materia Condensada, Universidad de Zaragoza, Zaragoza, 50009, Spain

ARTICLE INFO

Keywords:
Neural networks
Numerical methods
Differential equations

ABSTRACT

Physics-informed neural networks (PINNs) are numerical solvers that embed all the physical information of a system into the loss function of a neural network. In this way, the learned solution accounts for data (if available), the governing differential equations, or any other constraint known of the physical problem. However, they face serious issues, notably their tendency to converge on trivial or misleading solutions. The latter occurs when, although the loss function reaches low values, the model makes incorrect predictions. These difficulties become especially significant in differential equations involving multiscale behavior, such as those containing rapidly varying terms, as well as in problems whose solutions exhibit strong oscillatory behavior. To address these challenges, we introduce the Dynamical Boundary Constraint (DBC) algorithm, which imposes restrictions on the loss function based on prior training of the PINN. To demonstrate its applicability, we tested this approach on examples of different areas of physics.

1. Introduction

Differential equations are the backbone of calculus, as they describe the relationships between variables and their derivatives. They are essential to our comprehension of the natural world in physics areas and are extensively employed in other domains of knowledge. The challenge is that most differential equations do not have manageable analytical solutions, so we often have to rely on numerical methods instead.

As deep learning becomes widely used across scientific fields, interest in applying it to differential equations has grown rapidly, positioning it as a promising alternative or complement to traditional numerical methods. Early attempts to solve differential equations with Neural Networks (NN) date back to the 1990s [1]. These methods relied on trial functions that satisfied boundary conditions, which had the drawback that each problem required a different trial function. More recently, new techniques have been developed to overcome the limitations of the early methods, most notably the Physics-Informed Neural Networks (PINNs) [2]. A PINN is a NN that embeds the governing physical laws, usually written as differential equations, directly into its training process. Instead of learning from data alone, a PINN is trained to minimize a loss function that includes the residuals of the governing differential equations and related quantities (initial and/or boundary conditions), the error in observed data (if available), and, in general, any other constraint

* Corresponding author.
E-mail address: sergut@unizar.es (S. G. Rodrigo).

https://doi.org/10.1016/j.cnsns.2026.109866
Received 30 July 2025; Received in revised form 14 February 2026; Accepted 17 February 2026
Available online 18 February 2026

known of the solution. NNs can naturally incorporate differential equations because they can compute derivatives through automatic differentiation.

PINNs are already applied across various fields, including fluid dynamics [3], quantum mechanics [4], photonics [5], or engineering studies [6,7]. This underscores their applicability to a wide range of scientific fields, motivating numerous studies focused on their advancement and optimization [8,9].

However, PINNs still face significant challenges [10]. One major issue is their tendency to converge to trivial (null) solutions when applied to differential equations over large domains. Additionally, they can also produce results that appear reliable - because the total loss is low - yet are actually incorrect. Such difficulties are especially evident in differential equations whose solutions or governing terms exhibit multiscale features and/or strongly oscillatory behavior.

Several approaches have been proposed to address the challenges faced by PINNs when dealing with differential equations that involve multiscale terms or solutions exhibiting strong short-scale variations. These approaches include adding regularization terms to keep the different components of the loss at comparable scales or using two NNs to separately handle the low-order and high-order oscillatory terms [11]. Unbalanced gradients during backpropagation have been identified as a potential cause of PINN malfunction [12], suggesting that adjusting the learning rate for individual loss terms may improve predictive accuracy.

Chaotic systems provide a typical example of differential equations with highly oscillatory solutions. PINNs have been shown to struggle with such systems, as reported in Ref. [13], where a PINN was applied to solve the double pendulum equation. Several strategies have been proposed to overcome this limitation. One notable approach incorporates Fourier layers into the PINN architecture, applying sinusoidal filters to the input. This modification improves the PINN capacity to represent and adapt to the rapid variations characteristic of highly oscillatory solutions [14]. Another technique uses transfer learning, in which the PINN is first trained on low-frequency problems and then fine-tuned on high-frequency ones [15]. This approach has been shown to yield improved predictive accuracy for oscillatory equations, such as the damped cosine.

Finally, PINNs tend to enforce initial conditions and reduce the differential equation loss independently, which can result in solutions that are not globally consistent across the domain [16]. One possible strategy is to divide the domain into smaller subintervals and train an independent PINN on each segment. However, this strategy significantly limits the model's ability to generalize across the full domain.

Here, we introduce the Dynamic Boundary Constraint (DBC) algorithm as a solution to the limitations described above for PINNs. The core principle of the algorithm lies in the incorporation of additional restrictions, interpreted as boundary conditions, which are dynamically integrated into the training process alongside the existing initial conditions, thus enabling the PINN to ensure accuracy throughout the domain.

While enriching the PINN loss function with additional physical, analytical, or structural constraints is a well-established strategy in the literature -such as gradient-enhanced [17], symmetry-enforced [18], and analytically constrained formulations [19]-our contribution focuses on a specific formulation tailored to the problem considered here.

The rest of this paper is organized as follows. Section 2 presents the DBC method within the general PINN framework for solving Ordinary Differential Equation (ODE) systems. In Section 3.1, we highlight how standard PINNs struggle to accurately solve even a basic ODE, such as the harmonic oscillator. In contrast, the proposed DBC method yields a solution that closely matches the correct result. To further investigate the capabilities of the DBC method, we next apply it to a series of more challenging problems. In Section 3.2, to illustrate a multiscale problem, we will model the trajectory of a light ray in curved spacetime near a massive gravitational object (e.g., a black hole) using the optical ray equation. In Section 3.3, highly oscillatory behavior is studied with the Lorenz equations, the most iconic dynamical system that laid the foundation for chaos theory. These examples evaluate the robustness of our method on ODEs with multiscale and highly oscillatory behavior, while also demonstrating its applicability across diverse areas of physics.

2. PINNs with dynamic boundary constraints

As described in the introduction, PINNs are standard NNs that embed the governing differential equation, along with its initial and/or boundary conditions, directly into the loss function. In this work, we focus on general ODE systems. Mathematically, such a system can be expressed as:

$$D\left(x, \mathbf{y}, \mathbf{y}^{(1)}, \mathbf{y}^{(2)}, \ldots, \mathbf{y}^{(k)}\right) = \mathbf{0}, \tag{1}$$

where $x \in \Omega \subset \mathbb{R}$ is the independent variable in the domain Ω, $\mathbf{y} = \mathbf{y}(x) \in \mathbb{R}^m$ is the unknown (possibly vector-valued) solution, the superscript denoting differentiation of order k with respect to x, and D denotes a possibly linear system of ODEs.

The differential Eq. (1) is then solved for a given set of initial conditions:

$$\left\{\mathbf{y}(x_0), \mathbf{y}^{(1)}(x_0), \ldots, \mathbf{y}^{(k-1)}(x_0)\right\}, \tag{2}$$

where x_0 denotes the point at which the initial conditions are imposed.

A finite set of N collocation points in the interior of Ω, denoted by Γ_{ODE}, is introduced to discretize the solution. The PINN is evaluated at the points in Γ_{ODE}, during the training process.

The canonical loss function of a PINN consists of two contributions:

$$\mathbf{L}(\theta, \Gamma_{ODE}) = \omega_{ODE}\mathbf{L}_{ODE}(\theta, \Gamma_{ODE}) + \omega_{IC}\mathbf{L}_{IC}(\theta), \tag{3}$$

where θ denotes the NN trainable parameters. The term $\mathbf{L}_{ODE}(\theta, \Gamma_{ODE})$ measures the extent to which the NN approximation $\mathbf{y}_{NN}(x)$ satisfies the governing differential equation at the interior collocation points, while $\mathbf{L}_{IC}(\theta)$ measuring the accuracy with which the initial conditions are enforced. The pre-factors ω_{ODE} and ω_{IC} are real numbers being part of the hyperparameters of the problem. Explicitly, these losses are:

$$\mathbf{L}_{ODE}(\theta, \Gamma_{ODE}) = \frac{1}{N} \sum_{x \in \Gamma_{ODE}} \left\| D\left(x, \mathbf{y}_{NN}, \mathbf{y}_{NN}^{(1)}, \mathbf{y}_{NN}^{(2)}, \ldots, \mathbf{y}_{NN}^{(k)}\right) \right\|^2, \tag{4}$$

and

$$\mathbf{L}_{IC}(\theta) = \sum_{j=0}^{k-1} \left\| \mathbf{y}_{NN}^{(j)}(x_0) - \mathbf{y}^{(j)}(x_0) \right\|^2, \tag{5}$$

where the subscript NN denotes the quantities predicted by the PINN.

The central idea of the DBC method is to improve the generalization properties of a PINN by progressively enriching the loss function with additional constraints. To this end, the domain over which the ODE is solved is partitioned into a sequence of subintervals. If we introduce a number of DBCs equal to N_{DBC}, the number of subintervals becomes $N_S = N_{DBC} + 1$, which also corresponds to the number of required NN training runs, each involving a complete optimization over all epochs. Training is first performed on the subinterval containing the initial condition at x_0. Once the first training run convergence is achieved on the first subinterval, the network prediction at its endpoint is stored (the first DBC) and enforced as an additional constraint in the loss function for the second run, which includes the second subinterval. In other words, the training process is then continued on the subsequent subinterval, using the previously learned endpoint value to guide the solution. This procedure is repeated sequentially over all subintervals. In particular, the loss function for the final N_S training run (including all DBCs) is given by:

$$\mathbf{L}_{DBC}(\theta) = \frac{1}{N_{DBC}} \sum_{\alpha=1}^{N_{DBC}} \left\| \mathbf{y}_{NN}(\hat{X}_\alpha) - \hat{Y}_\alpha \right\|^2. \tag{6}$$

This loss function measures the accuracy with which the DBC conditions are enforced at endpoints $\hat{X}_\alpha$ of the α-th training-run/subinterval, where $\hat{Y}_\alpha$ is the saved PINN prediction on that training run. If the final training run is successful, $\mathbf{y}_{NN}$ should provide a good approximation to the solution of the differential equation for the given initial conditions within the domain of interest.

A final technical note. Two approaches may be employed in the algorithm: (i) the cumulative approach, which includes training points from both the current and all previous subintervals, or (ii) the individual subinterval-based approach, where the PINN is trained exclusively on points within the new subinterval. Regardless of the chosen approach, the final training run is performed over the entire domain.

3. Case studies and applications

3.1. Limitations of standard PINNs

We illustrate the challenges encountered by PINNs when applied to large domains and the corresponding DBC-based solution by considering the one-dimensional harmonic oscillator:

$$\frac{d^2y}{dx^2} = -y. \tag{7}$$

The PINN takes values of x as input and produces the corresponding predictions (approximations) for $y(x)$. The standard PINN consists of two loss terms, one for the differential equation and another for the initial conditions. That is,

$$\mathbf{L}_{ODE} = \frac{1}{N} \sum_{i=1}^{N} \left(\frac{dy_{NN}^2}{dx^2}(x_i) + y_{NN}(x_i) \right)^2, \tag{8}$$

for the differential equation and,

$$\mathbf{L}_{IC} = (y_{NN}(x_0) - y(x_0))^2 + (\frac{dy_{NN}}{dx}(x_0) - \frac{dy}{dx}(x_0))^2, \tag{9}$$

for the initial conditions.

The DBC loss for the final training run can be expressed as:

$$\mathbf{L}_{DBC} = \frac{1}{N_{DBC}} \sum_{\alpha=1}^{N_{DBC}} \left(y_{NN}(\hat{X}_\alpha) - \hat{Y}_\alpha \right)^2. \tag{10}$$

The subscript NN indicates the predictions made by the PINN. Eq. (8) evaluates these parameters at a set of N distinct points x_i, which corresponds to the training data. In Eq. (9), the output of the PINN is evaluated at the coordinate corresponding to the initial conditions, x_0. As described in the previous section, $\hat{Y}_\alpha$ is the saved PINN prediction at $\hat{X}_\alpha$, the endpoint of the α-th training-run/subinterval.

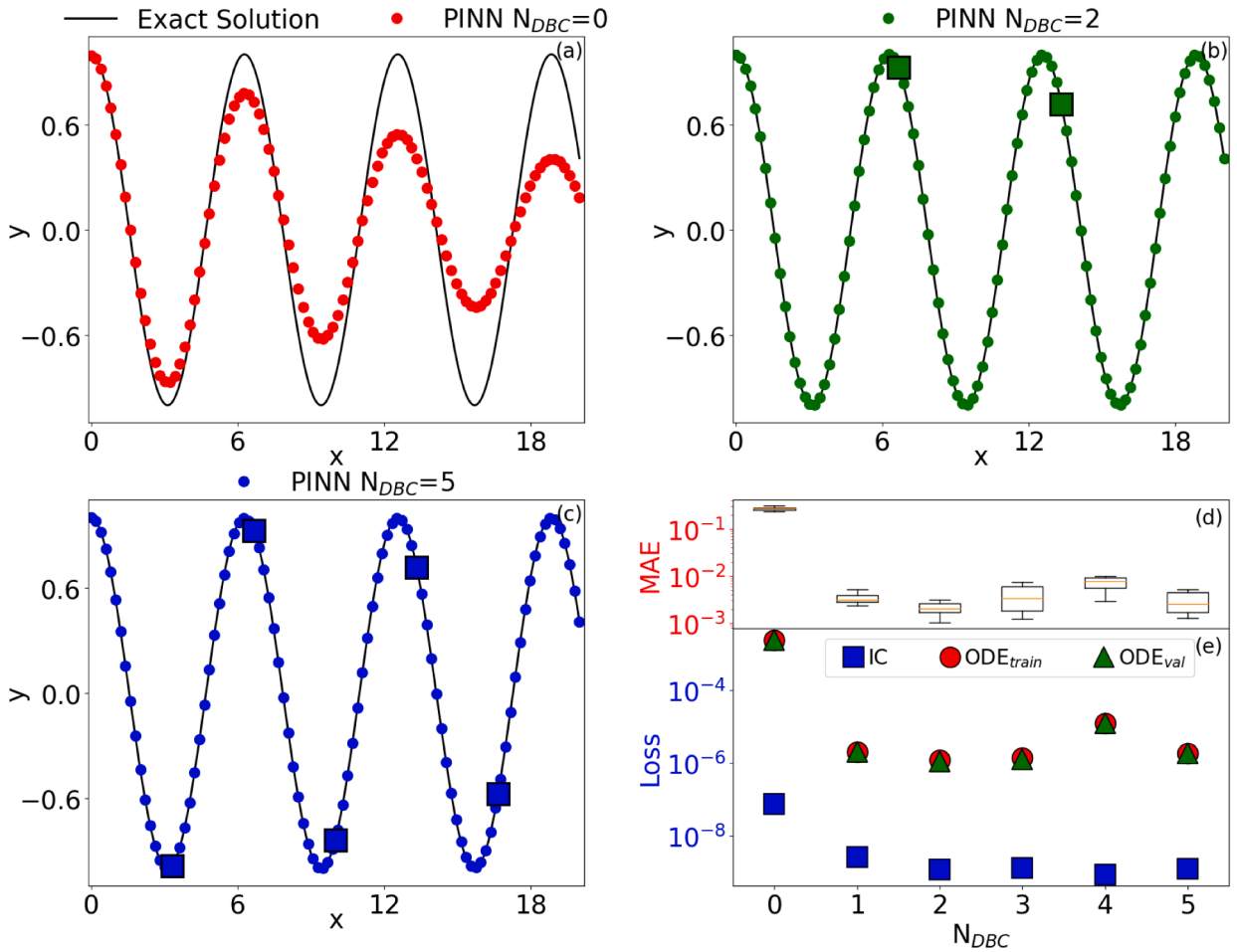


Fig. 1. Comparison of PINN predictions (dots) with the exact solution (solid line) for the harmonic oscillator equation using the cumulative approach: (a) standard PINN ($N_{DBC} = 0$); (b) PINN with two DBCs (square marks their location); and (c) PINN with five DBCs. (d) Box plot of the MAE between the PINN predictions and the exact solution across 10 independent training cycles, shown as a function of N_{DBC}. The orange line shows the median, while the whiskers represent the minimum and maximum errors. The PINN results shown in panels (a)-(c) correspond to the lowest observed error. (e) Training losses associated with the lowest-error model for each DBC configuration in (d). Squares indicate the initial condition loss, $\mathbf{L}_{IC}$. The ODE loss, $\mathbf{L}_{ODE}$, is evaluated at $N_{\text{train}} = 200$ training points (circles) and validation points (triangles), with the latter chosen so that $N_{\text{val}} = 10 \times N_{\text{train}}$.

The chosen PINN architecture has three hidden dense layers, each containing 500 neurons. The trainable parameters are initialized using the Glorot Uniform initializer, and the activation function employed is the hyperbolic tangent in all cases. We use the Adam optimizer with a learning rate of 5×10^{-5} for optimization. This setup will be used throughout this work unless otherwise stated. We emphasize, however, that the specific architecture is not critical: PINNs generally exhibit a high degree of robustness to choices such as the number of layers, neurons, or optimizers.

Fig. 1(a) shows that a standard PINN without DBCs (red dots) fails to predict the solution to the harmonic oscillator equation $y_{exact} = \cos(x)$ (black line) across a domain spanning multiple periods. The analytical solution is obtained for the initial conditions $y(x_0) = 1$ and $y'(x_0) = 0$ at $x_0 = 0$. For training, we use $N = 200$ points uniformly distributed across the interval $x \in [0, 20]$. Though trained on this interval, the PINN has difficulties capturing the harmonic behavior, leading to significant deviations from the actual solution, particularly as x increases. This illustrates the limitations of the standard PINN when applied to ODEs with strongly oscillatory solutions over extended domains.

An interesting tendency is observed in the prediction made by the standard PINN. Near the region where the initial conditions are imposed, the PINN prediction accurately approximates the analytical solution. However, as we move further from this point, the trend begins to deviate. In this situation, the solution asymptotically approaches the zero function, which, while valid, is not the solution to Eq. (7) for the specified initial conditions.

The impact of introducing DBCs is demonstrated in Fig. 1 (b) and (c); with two and five DBCs (represented by squares), the PINN predictions (shown as dots) perfectly match the exact solution. At this point, the deviation between the exact solution and the PINN prediction becomes evident in the standard PINN case, highlighting a critical limitation that is effectively addressed by incorporating DBCs.

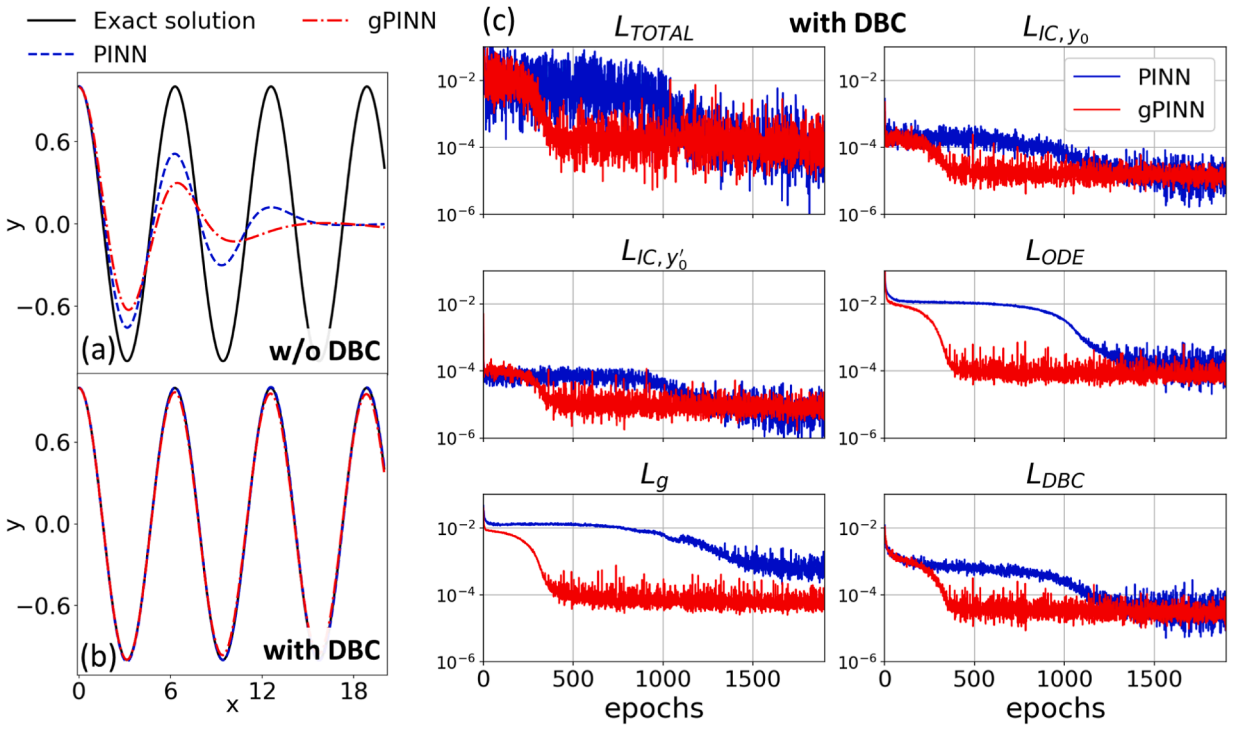


Fig. 2. Standard PINN vs. gradient-enhanced PINN (gPINN), with and without DBCs. Panels (a) and (b) show the exact solution (black solid line) together with the predictions obtained using the cumulative approach for the PINN (blue) and gPINN (red): (a) without DBCs ($N_{DBC} = 0$) and (b) with DBCs ($N_{DBC} = 2$). These results correspond to the lowest total loss reached during training. Panel (c) depicts the evolution of all loss terms during training for both the PINN and gPINN, corresponding to the results in panel (b). The gradient loss $\mathbf{L}_g$ is shown for comparison in the case of the standard PINN, although this term is not active in that case.

Note that the cumulative approach was used in Fig. 1, but the individual subinterval-based approach can also yield accurate solutions, as demonstrated in Fig. 1 in the Supplementary Material.

This improvement is statistically supported, as evidenced by the box plots in Fig. 1(d), which depict the Mean Absolute Error (MAE) distributions obtained from 10 independent training cycles for each value of N_{DBC}. The error bars (whiskers) indicate the maximum and minimum values of the error. The orange line indicates the median of the distribution.

At first glance, the error observed in standard PINNs might appear to stem from difficulties in minimizing the loss during training. However, this is not the case. As shown in Fig. 1(e), the losses corresponding to the initial conditions $\mathbf{L}_{IC}$ (squares) are very low in the standard PINN configuration ($\mathbf{L}_{IC} \sim 10^{-7}$). The losses associated with the differential equation, $\mathbf{L}_{ODE}$, remain low ($\sim 10^{-3}$) regardless of whether they are computed on the training or validation dataset (a denser dataset with ten points between each training pair). The absence of overfitting in this case is further confirmed by Fig. 2 (a) ($N_{DBC} = 0$) and (b) ($N_{DBC} = 5$) of the Supplementary Material, which plot $\mathbf{L}_{ODE}$ as a function of x for both datasets. This relatively small value would indicate a good performance of the PINN, considering that the oscillatory terms in Eq. (7) are generally of larger magnitude. This showcases that low-loss values do not necessarily guarantee accurate PINN predictions. This can be attributed to a common behavior in PINNs. Once the initial conditions are enforced, $\mathbf{L}_{ODE}$ decreases at all collocation points, even though the resulting local solutions may no longer be consistent with the prescribed initial conditions.

It is important to note that the number of DBCs, despite the intuitive expectation that increasing them should improve performance, cannot be used as a reliable indicator of solution accuracy. As depicted in Fig. 1(d), the associated error (minimum value in the box plot) increases when moving from $N_{DBC} = 2$ to $N_{DBC} = 3$, even though, as shown in panel (e), this change slightly reduces $\mathbf{L}_{ODE}$. This trend is even more obvious in later examples.

DBCs can be naturally integrated with other techniques designed to improve the performance and robustness of PINN architectures. In Fig. 2, we illustrate the combined application of DBCs with the gradient-enhanced PINN (gPINN) [17]. This technique is based on the properties of $\mathcal{D}$ (Eq. (1)). The method assumes that the exact solutions are smooth enough for the gradient of $\mathcal{D}$ to exist, thereby justifying the additional enforcement of its vanishing derivatives. For the harmonic oscillator example, this results in an additional loss term,

$$\mathbf{L}_g = \frac{d\mathbf{L}_{ODE}}{dx} = \frac{1}{N}\sum_{i=1}^{N}\left(\frac{dy_{NN}^3}{dx^3}(x_i) + \frac{dy_{NN}}{dx}(x_i)\right)^2. \tag{11}$$

As shown in Fig. 2(a), the gPINN alone exhibits the same difficulty as the standard PINN in recovering the solution, even for the simple case of a harmonic oscillator when DBCs are not employed. Moreover, Fig. 2(b) indicates that adding the gradient loss to the total loss function does not lead to a noticeable improvement in accuracy once DBCs are included. Using gPINNs, the minimum loss values are achieved in fewer training epochs compared to standard PINNs, as clearly demonstrated by the loss evolution shown in Fig. 2(c). These loss curves correspond to the training run presented in panel (b). However, the inclusion of the additional term, which requires computing third-order derivatives, increases the computational time per epoch by approximately a factor of 1.6.

For the remaining examples in the manuscript, we have not performed additional gPINN comparisons to isolate the discussion and focus specifically on the performance of the DBC approach, which is the central contribution of this work.

3.2. Multiscale problems

The Fermat principle remains fundamental in geometric optics despite being formulated centuries ago. It inspires innovative applications, such as the recent development of generalized laws of reflection and refraction in metasurfaces [20]. Another example of the method's relevance is the study of light trajectories in GRaded-INdex (GRIN) media using the optical ray equation derived from the principle of minimal action. GRIN lenses offer significant advantages over traditional lenses, including their ability to miniaturize optical systems through compact designs while maintaining high-resolution imaging and efficient light transmission. The optical ray equation is also instrumental in the design of mini-endoscopes for probing deep brain regions [21,22], in the development of lightweight and miniaturized lens systems [23], and in the inverse reconstruction of refractive-index distributions from prescribed light trajectories [24]. The optical ray equation admits closed-form analytical solutions only in a limited set of simple geometries. Heterogeneous refractive-index distributions and realistic material optical properties (e.g., dispersion) typically give rise to differential equations for which analytical solutions are either intractable or entirely unavailable. In such cases, numerical methods or approximations are required to obtain a solution [25–27].

We applied the standard PINN enhanced with the DBC method to the optical ray equation. Exemplarily, we calculated light paths around massive stellar objects (e.g., black holes), where the gravitational pull on light is modeled as an effective refractive index [28], based on the Schwarzschild metric [29],

$$n(r) = \frac{1}{1 - r_s/r}. \tag{12}$$

Here, $r_s = \frac{2GM}{c^2}$ is the Schwarzschild radius, which is composed of the gravitational constant G, the speed of light in vacuum c, the mass M, and the radial distance to the center of the astronomical object r.

To describe the light path $\vec{r}$ in such an effective medium, we use the optical ray equation,

$$\frac{d}{ds}\left(n\frac{d\vec{r}}{ds}\right) = \nabla n, \tag{13}$$

where the parameter s represents the arc-length along the ray path.

This is a clear example of multiscale behavior in ODEs, as evidenced by the dependence of the path light on the gradient of n, which strongly varies near the Schwarzschild radius.

By limiting the problem to a plane that includes the massive object situated at the origin of the coordinate system and by expanding Eq. (13), we can express the ray equation for the trajectory $\vec{r} = (x, y)$ in Cartesian coordinates as follows:

$$\begin{aligned}
\frac{dp_x}{ds} &= \frac{\partial n(x,y)}{\partial x} = -n^2\frac{r_s x}{r^3} \\
\frac{dp_y}{ds} &= \frac{\partial n(x,y)}{\partial y} = -n^2\frac{r_s y}{r^3},
\end{aligned} \tag{14}$$

where $r = +\sqrt{x^2 + y^2}$ and the momentum $\vec{p} = (p_x, p_y) = n\frac{d\vec{r}}{ds}$. Therefore:

$$\begin{aligned}
\frac{dx}{ds} &= \frac{p_x}{n(x,y)} \\
\frac{dy}{ds} &= \frac{p_y}{n(x,y)}.
\end{aligned} \tag{15}$$

To solve Eqs. (14) and (15), we define a PINN whose input layer takes the arc-length parameter s, and whose output layer represents the four-dimensional vector (x, y, p_x, p_y). The total loss function of the PINN is defined as $\mathbf{L} = \mathbf{L}_{ODE} + \mathbf{L}_C$, where $\mathbf{L}_{ODE}$ accounts for the contributions from Eqs. (14) and (15) (i.e., the differential equation residuals), while $\mathbf{L}_C = \mathbf{L}_{IC} + \mathbf{L}_{DBC}$, the contributions from the initial conditions and DBCs respectively. Explicitly:

$$\begin{aligned}
\mathbf{L}_{ODE} = \frac{1}{N}\sum_{i=1}^{N}\Bigg(& a_1 \cdot \left[r^3\frac{dp_x^{NN}}{ds}(s_i) + n^2 r_s x^{NN}(s_i)\right]^2 + \\
& a_2 \cdot \left[r^3\frac{dp_y^{NN}}{ds}(s_i) + n^2 r_s y^{NN}(s_i)\right]^2 + \\
& a_3 \cdot \left[\frac{dx^{NN}}{ds}(s_i) - \frac{p_x^{NN}(s_i)}{n}\right]^2 +
\end{aligned}$$

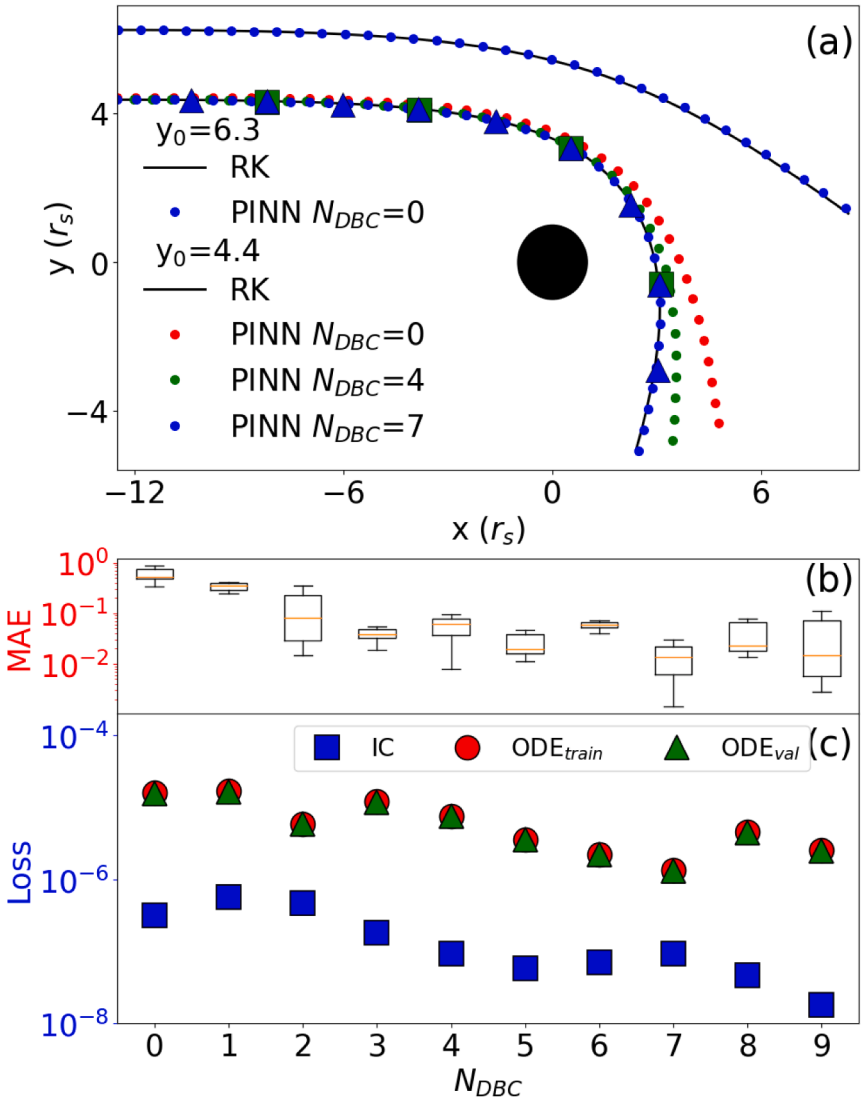


Fig. 3. (a) Comparison of light trajectories between the PINN predictions (dots) and the RK solutions (solid lines) for two initial conditions. Triangles and squares indicate the locations of the DBCs for the two example configurations, while the central circle denotes the radius of the massive object's event horizon. (b) Box plot of the MAE between the PINN predictions and the RK method across 10 independent training cycles, shown as a function of N_{DBC}. The error bars (whiskers) represent the minimum and maximum error values. The orange line is the median of the distribution. (c) Training losses corresponding to the lowest-error model for each DBC configuration in (b). The initial condition loss, $\mathbf{L}_{IC}$, is represented by squares. The ODE loss, $\mathbf{L}_{ODE}$, is evaluated at $N_{\text{train}} = 200$ training points (circles) and at validation points (triangles), with the latter chosen such that $N_{\text{val}} = 10 \times N_{\text{train}}$. The cumulative approach was used.

$$a_4 \cdot \left[\frac{dy^{NN}}{ds}(s_i) - \frac{p_y^{NN}(s_i)}{n} \right]^2 \Big), \tag{16}$$

and,

$$\begin{aligned} &\mathbf{L}_{IC} = \\ &b_1 \cdot [x^{NN}(s_0) - x_0]^2 + b_2 \cdot [y^{NN}(s_0) - y_0]^2 + \\ &b_3 \cdot [p_x^{NN}(s_0) - p_{x_0}]^2 + b_4 \cdot [p_y^{NN}(s_0) - p_{y_0}]^2. \end{aligned} \tag{17}$$

The same mathematical expression as in Eq. (17) is used for $\mathbf{L}_{DBC}$ with the phase-space vectors $\vec{\xi} = (\vec{r}, \vec{p})$ taken from the set of DBCs.

Here, the superscript NN denotes the quantities predicted by the PINN, so that $n = n(x^{NN}, y^{NN})$. Eqs. (14) have been rearranged in Eq. (16) to avoid singularities for $r \to 0$. Eq. (17) is evaluated at the initial value of the parameter s (s_0), whereas Eq. (16) is evaluated at s_i ($i = 1, \ldots, N$), corresponding to the training points. The hyperparameters a_i and b_i (i=1,...4) can be tuned to improve

the performance of the PINN. In our case, $a_1 = 5$ and $a_2 = 10$, with all other values equal to one. This adjustment was based on initial tests, indicating that these components were more challenging to predict than other terms in the loss function.

Regarding the NN, we used the same hyperparameters as those for the harmonic oscillator, except that here there are five dense hidden layers, each with 500 neurons. Similarly to the harmonic oscillator, the cumulative approach is employed.

Fig. 3(a) shows the light paths computed using the PINN (dots) and compares them to calculations with the fourth-order Runge-Kutta (RK) method (solid lines). We choose $r_s = 1$ so that the light trajectory is neither perfectly straight nor does it collapse into the massive object within the region of space under consideration. Two distinct impact parameters have been chosen (in units of r_s): i) $(x_0, y_0) = (-12.5, 6.25)$, and ii) $(x_0, y_0) = (-12.5, 4.375)$. In both cases, the initial momentum is the same: $(p_{x_0}, p_{y_0}) = (12.5, 0)$. In this case, $N = 200$ training points are used, evenly spaced within the range $s \in [0, 2]$. This range is large enough to capture the curvature of light induced by the massive object. The central circle denotes the radius of the massive object's event horizon.

The PINN performs well with $N_{DBC} = 0$ at $y_0 = 6.3$ because this trajectory of light is only minimally affected by the presence of the massive object. However, the agreement progressively deteriorates as the impact parameter decreases, as observed for $y_0 = 4.4$. For this scenario, the RK method predicts more pronounced light bending near the massive object than the PINN prediction without DBCs. As the light trajectory approaches the horizon of the massive object, the gravitational contribution from the source terms in Eqs. (14) and (15) increases sharply, creating a pronounced imbalance among the ODE terms and ultimately degrading the accuracy of the standard PINN predictions.

Returning to Fig. 3(a), we see the impact of applying the DBC method during training for the trajectory with $y_0 = 4.4$. The inclusion of DBCs, represented by triangles and squares, mitigates the discrepancies observed when the trajectory approaches the singularity.

To statistically validate the results, we trained 10 different PINNs for each N_{DBC} configuration corresponding to the ray path with the strongest bending ($y_0 = 4.4$). The results are presented in Fig. 3(b), where a box plot summarizes the statistical distribution of the error (MAE) between the PINN predictions and the RK method for each N_{DBC}. Overall, agreement with the RK results generally improves as the number of DBCs increases, albeit not monotonically, a trend already found in the harmonic oscillator study. This is evident upon inspection of all the trajectories obtained with different DBCs (see Fig. 3 (a) and (b) of the Supplementary Material). As before, no clear correlation is observed between the error and the loss values. Fig. 4 in the Supplementary Material shows the validation-set error as a function of $\mathbf{L}_{ODE}$, indicating that the lowest error, and thus the closest agreement between the PINN prediction and the RK solution, occurs at $N_{DBC} = 7$.

In Fig. 3(c), the losses $\mathbf{L}_{ODE}$ and $\mathbf{L}_{IC}$ corresponding to the lowest errors achieved during all training cycles are indicated by triangles (validation data), circles (training data), and squares (initial conditions). The ODE loss remains consistently low ($< 10^{-4}$) across all cases, including the standard PINN, reinforcing the earlier observation from the harmonic oscillator study that a low loss value does not necessarily guarantee an accurate solution to the differential equation.

As in the previous example, the individual subinterval-based approach produces accurate predictions, as shown in Fig. 5 of the Supplementary Material.

3.3. Highly oscillatory dynamics

The Lorenz equations, originally conceived as a model to study atmospheric convection, revealed a system of three-dimensional differential equations exhibiting chaotic dynamics [30]. One of the most characteristic behaviors in chaotic systems is the appearance of regions in phase space toward which the system's trajectories converge, known as attractors, which present a strongly oscillatory dynamics. These equations are:

$$\begin{aligned}
\frac{dy_1}{dx} &= a \cdot (y_2 - y_1) \\
\frac{dy_2}{dx} &= y_1 \cdot (b - y_3) - y_2 \\
\frac{dy_3}{dx} &= y_1 y_2 - c y_2,
\end{aligned} \tag{18}$$

where a is the Prandtl number, and b is the Rayleigh number.

The corresponding PINN consists of a single input neuron for x and three output neurons representing the predictions of y_1, y_2, and y_3. The PINN's loss function is defined as before, $\mathbf{L} = \mathbf{L}_{ODE} + \mathbf{L}_C$, where $\mathbf{L}_{ODE}$ includes the contributions from Eq. (18), and $\mathbf{L}_C = \mathbf{L}_{IC} + \mathbf{L}_{DBC}$ accounts for the contributions from both the initial conditions and DBCs. As before, $\mathbf{L}_{DBC}$ has the same form as $\mathbf{L}_{IC}$ but is evaluated at the DBCs. Mathematically:

$$\begin{aligned}
\mathbf{L}_{ODE} = \frac{1}{N} \sum_{i=1}^{N} \Bigg(& \alpha_1 \left[\frac{dy_1^{NN}}{dx}(x_i) - a \cdot (y_2^{NN}(x_i) - y_1^{NN}(x_i)) \right]^2 + \\
& \alpha_2 \left[\frac{dy_2^{NN}}{dx}(x_i) - y_1^{NN}(x_i) \cdot (b - y_3^{NN}(x_i)) + y_2^{NN}(x_i) \right]^2 + \\
& \alpha_3 \left[\frac{dy_3^{NN}}{dx}(x_i) - y_1^{NN}(x_i) y_2^{NN}(x_i) + c y_2^{NN}(x_i) \right]^2 \Bigg),
\end{aligned} \tag{19}$$

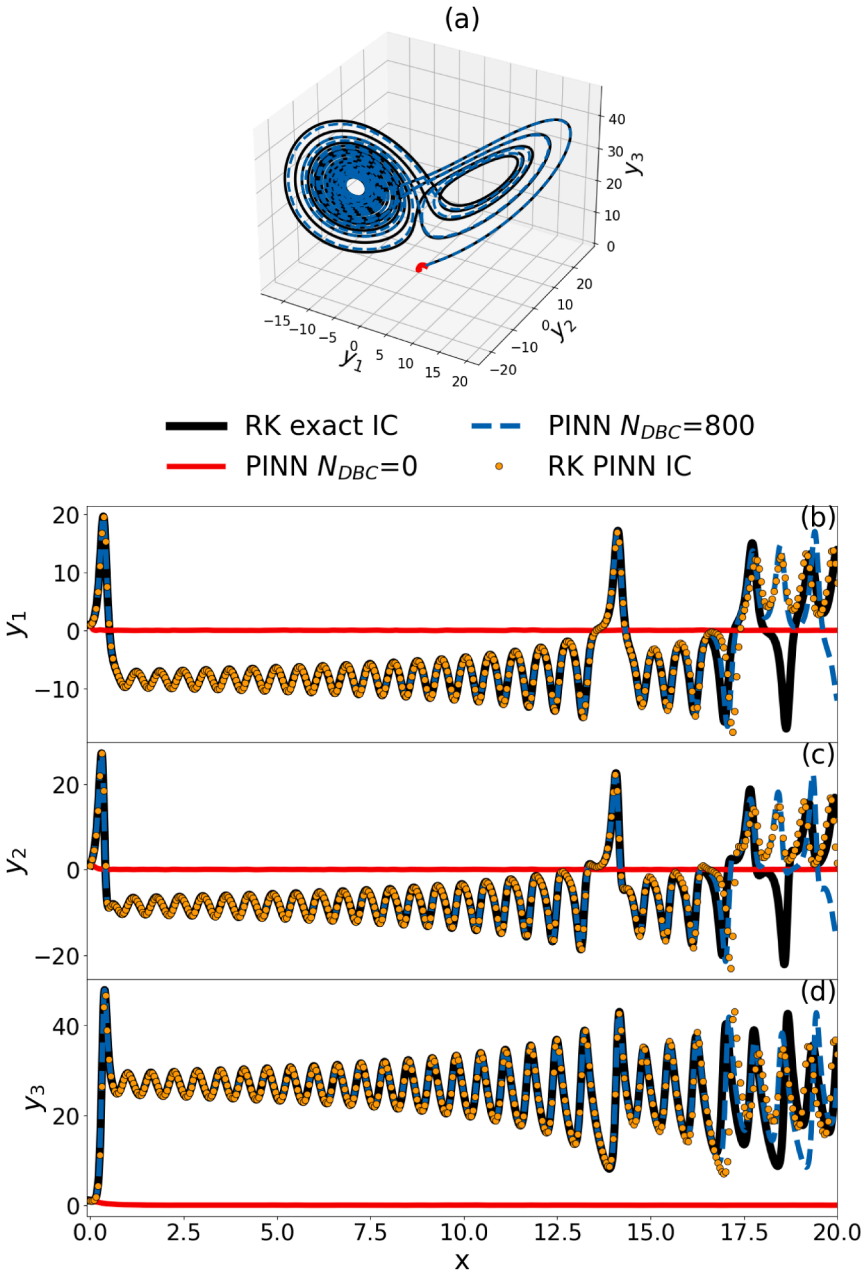


Fig. 4. (a) Three-dimensional trajectory of the Lorenz attractor showing the RK solution (black line), the PINN prediction with DBCs (blue line), and the standard PINN prediction without DBCs (red line). (b)-(d) Individual components of the trajectory. The solid black line represents the RK solution with the exact initial conditions, the solid red line shows the standard PINN prediction, and the dashed blue line corresponds to the PINN with $N_{DBC} = 800$. The dotted orange curve depicts the RK solution obtained using the initial conditions predicted by the PINN.

and,

$$\mathbf{L}_{IC}(x_0) = \beta_1[y_1^{NN}(x_0) - y_1(x_0)]^2 + \beta_2[y_2^{NN}(x_0) - y_2(x_0)]^2 + \beta_3[y_3^{NN}(x_0) - y_3(x_0)]^2. \tag{20}$$

Consistent with previous notation, x_i refers to the training coordinates, while x_0 denotes the initial point. The loss terms are weighted by $\alpha_i = 1$ and $\beta_i = 6000$ to prioritize the accurate enforcement of the initial conditions, which is critical for chaotic systems where small errors in the initial state can grow exponentially.

We used the same hyperparameters and network architecture as in the harmonic oscillator case, with the only modification being the inclusion of a Fourier layer preceding the dense layers, which enhances convergence in systems exhibiting chaotic behavior [14].

Given the complexity of the Lorenz dynamical system, we set $N_{DBC} = 800$, as using a comparable number to that in previous examples leads to prediction breakdown before the onset of chaotic behavior. It should be noted, however, that no attempt was made to determine the optimal number of DBCs.

To reduce the computational cost of solving the equations with a very large number of DBCs, we employ the individual subinterval-based approach described earlier, as it is more efficient than the cumulative approach in this case.

Fig. 4(a) shows the solution obtained for the standard PINN (red line), the PINN with DBCs (blue dashed line), and the RK method with exact initial conditions (solid black line). In this example, we consider the interval $x \in [0, 20]$ with initial conditions $y_1(0) = y_2(0) = y_3(0) = 1$. The parameters a, b, and c are strictly positive and set to $a = 10$, $b = 28$, and $c = \frac{8}{3}$, a classic configuration known to produce chaotic behavior in the Lorenz system. Since this solution demands greater precision, the number of training points is significantly increased to $N = 6000$, compared to the previous examples.

Notably, the standard PINN fails to produce a meaningful solution in the absence of DBCs, with its prediction remaining trapped at the initial condition for all x. This highlights the significant improvement in accuracy achieved by incorporating DBCs into the PINN framework.

To facilitate a clearer comparison, Fig. 4(b)–(d) shows the Lorenz trajectory decomposed into its components. These plots demonstrate the good agreement between the PINN with DBCs and the RK solution, with deviations appearing only near the end of the domain. This discrepancy can be partly attributed to the chaotic nature of the Lorenz system, where small inaccuracies in the PINN predictions, such as at the initial point $x_0 = 0$, can grow exponentially over time, eventually leading to significant divergence from the reference trajectory. This becomes evident when the RK method is initialized with the PINN-predicted value at $x_0 = 0$ (orange dots) rather than the prescribed initial conditions. In this case, the agreement between the two solutions improves markedly. This indicates that the PINN accurately captures the structure of the Lorenz solution but corresponds to a slightly shifted initial condition.

4. Conclusions

We presented the DBC method as a way to improve the accuracy of PINN-based differential-equation solvers. The method works by dynamically reintroducing information obtained in earlier training phases to guide subsequent optimization. This approach addresses two key issues in PINNs: their tendency to converge to trivial solutions and to provide misleading results (incorrect predictions despite low losses), especially at points far from the initial conditions. These problems are particularly pronounced in differential equations with rapidly changing terms or strongly oscillatory solutions. Our tests on the optical ray equation and the chaotic Lorenz attractor showed that incorporating DBCs significantly improves PINN performance, providing a practical and effective strategy for implementing PINNs in complex scenarios. Although the present study focuses on ODEs, this framework already illustrates the benefits of DBC-enhanced PINNs. Extending the approach to PDEs remains a challenging problem and is beyond the scope of this work. Nevertheless, ongoing efforts suggest that DBCs may provide a valuable tool for mitigating some of the numerical difficulties encountered in PINN-based PDE solvers.

CRediT authorship contribution statement

Andrés Martínez-Esteban: Methodology, Software, Investigation, Writing - original draft, Writing - review & editing; **Pablo Calvo-Barlés:** Methodology, Investigation, Writing - review & editing; **Luis Martín-Moreno:** Investigation, Writing - review & editing; **Sergio G. Rodrigo:** Conceptualization, Supervision, Methodology, Software, Investigation, Writing - original draft, Writing - review & editing.

Data availability

All PINNs developed for this study were implemented using the Keras and TensorFlow libraries. An example implementation of the proposed method is available at GitHub repository. The repository contains a simplified version of the code that is sufficient to reproduce the core methodology.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgements

The Spanish “Ministerio de Ciencia, Innovación y Universidades” supports and funds this work, MICIU/AEI/10.13039/501100011033 under Grants No. PID2023-148359NB-C21 (also funded by FEDER, UE), No. CEX2023-001286-S (Severo Ochoa Centers of Excellence) and CEX2021-001144-S-20-10 (predoctoral research fellowship). We also acknowledge the Aragón Regional Government through project QMAD (E09_23R).

Supplementary material

Supplementary material associated with this article can be found in the online version at 10.1016/j.cnsns.2026.109866.

References

[1] Lagaris IE, Likas A, Fotiadis DI. Artificial neural networks for solving ordinary and partial differential equations. IEEE Trans Neural Netw 1998;9(5):987–1000. https://doi.org/10.1109/72.712178

[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;378:686–707. https://doi.org/10.1016/j.jcp.2018.10.045

[3] Cai S, Mao Z, Wang Z, Yin M, Karniadakis GE. Physics-informed neural networks (PINNs) for fluid mechanics: a review. Acta Mech Sin 2021;37(12):1727–38. https://doi.org/10.1007/s10409-021-01148-1

[4] Shah K, Stiller P, Hoffmann N, Cangi A. Physics-informed neural networks as solvers for the time-dependent schrödinger equation 2022; https://doi.org/10.48550/ARXIV.2210.12522

[5] Chen Y, Lu L, Karniadakis GE, Dal Negro L. Physics-informed neural networks for inverse problems in nano-optics and metamaterials. Opt Express 2020;28(8):11618. https://doi.org/10.1364/oe.384875

[6] Zhang S, Deng J, Li X, Zhao Z, Wu J, Li W, et al. Solving the one dimensional vertical suspended sediment mixing equation with arbitrary eddy diffusivity profiles using temporal normalized physics-informed neural networks. Phys Fluids 2024;36(1):017132. https://doi.org/10.1063/5.0179223

[7] Varey J, Ruprecht JD, Tierney M, Sullenberger R. Physics-informed neural networks for satellite state estimation. In: 2024 IEEE aerospace conference. IEEE; 2024, https://doi.org/10.1109/aero58975.2024.10521414

[8] Cuomo S, Di Cola VS, Giampaolo F, et al. Scientific machine learning through physics-informed neural networks: where we are and what's next. J Sci Comput 2022;92, 88. https://doi.org/10.1007/s10915-022-01939-z

[9] Farea A, Yli-Harja O, Emmert-Streib F. Understanding physics-informed neural networks: techniques, applications, trends, and challenges. AI 2024;5(3):1534–57. https://doi.org/10.3390/ai5030074

[10] Rathore P, Lei W, Frangella Z, Lu L, Udell M. Challenges in training PINNs: a loss landscape perspective. In: Proceedings of the 41st international conference on machine learning. ICML'24; JMLR.org; 2024,.

[11] Wang Y, Yao Y, Guo J, Gao Z. A practical PINN framework for multi-scale problems with multi-magnitude loss terms. J Comput Phys 2024;510:113112. https://doi.org/10.1016/j.jcp.2024.113112

[12] Wang S, Teng Y, Perdikaris P. Understanding and mitigating gradient flow pathologies in physics-informed neural networks. SIAM J Scient Comput 2021;43(5):A3055–A3081. https://doi.org/10.1137/20m1318043

[13] Steger S, Rohrhofer F, Geiger B. How pinns cheat: predicting chaotic motion of a double pendulum. Symbiosis of Deep Learning and Differential Equations II @ the 36th Neural Information Processing Systems (NeurIPS) Conference Conference date: 09-12-2022 Through 09-12-2022.

[14] Wang S, Wang H, Perdikaris P. On the eigenvector bias of fourier feature networks: from regression to solving multi-scale PDEs with physics-informed neural networks. Comput Methods Appl Mech Eng 2021;384:113938. https://doi.org/10.1016/j.cma.2021.113938

[15] Mustajabw AH, Lyu H, Rizvi Z, Wuttke F. Physics-informed neural networks for high-frequency and multi-scale problems using transfer learning. Appl Sci 2024;14(8):3204. https://doi.org/10.3390/app14083204

[16] Krishnapriyan A, Gholami A, Zhe S, Kirby R, Mahoney MW. Characterizing possible failure modes in physics-informed neural networks. In: Ranzato M, Beygelzimer A, Dauphin Y, Liang PS, Vaughan JW, editors. Advances in neural information processing systems; vol. 34. Curran Associates, Inc.; 2021, p. 26548–60. https://proceedings.neurips.cc/paper_files/paper/2021/file/df438e5206f31600e6ae4af72f2725f1-Paper.pdf.

[17] Yu J, Lu L, Meng X, Karniadakis GE. Gradient-enhanced physics-informed neural networks for forward and inverse PDE problems. Comput Methods Appl Mech Eng 2022;393:114823. https://doi.org/10.1016/j.cma.2022.114823

[18] Zhang Z-Y, Zhang H, Zhang L-S, Guo L-L. Enforcing continuous symmetries in physics-informed neural network for solving forward and inverse problems of partial differential equations. J Comput Phys 2023;492:112415. https://doi.org/10.1016/j.jcp.2023.112415

[19] van der Heijden B, Li X, Lubineau G, Florentin E. Enforcing physics onto PINNs for more accurate inhomogeneous material identification. Comput Methods Appl Mech Eng 2025;441:117993. https://doi.org/10.1016/j.cma.2025.117993

[20] Yu N, Genevet P, Kats MA, Aieta F, Tetienne J-P, Capasso F, et al. Light propagation with phase discontinuities: generalized laws of reflection and refraction. Science 2011;334(6054):333–7. https://doi.org/10.1126/science.1210713

[21] Bagramyan A, Galstian T. Dynamic control of polarization mismatch and coma aberrations in rod-GRIN assemblies. Opt Express 2019;27:14144. https://doi.org/10.1364/oe.27.014144

[22] Tabourin L, Bretzner F, Galstian T. Towards a mini-endoscope design with spatially selective excitation and imaging. Biomed Opt Express 2024;15:1750. https://doi.org/10.1364/boe.512124

[23] Xiao D, Lin Z, Shechtman Y. All-in-focus large-FOV GRIN lens imaging by multi-focus image fusion. Optics Continuum 2023;2:2290. https://doi.org/10.1364/optcon.500503

[24] Zhang Z, Wang R, Zhang X, Ge R, Zheng W, Chen M, et al. Refractive index measurement deflectometry for measuring gradient refractive index lens. Opt Express 2024;32(7):12620. https://doi.org/10.1364/oe.518670

[25] Moore DT. Ray tracing in gradient-index media. J Opt Soc Am 1975;65(4):451. https://doi.org/10.1364/josa.65.000451

[26] Sharma A, Kumar DV, Ghatak AK. Tracing rays through graded-index media: a new method. Appl Opt 1982;21(6):984. https://doi.org/10.1364/ao.21.000984

[27] Ohno H. Symplectic ray tracing based on hamiltonian optics in gradient-index media. J Opt Soc Am A 2020;37(3):411. https://doi.org/10.1364/josaa.378829

[28] Ghatak AK, Thyagarajan K. Introduction to Fiber Optics. Cambridge: Cambridge Univ. Press; 2004. ISBN 0521577853.

[29] Nolte DD. Introduction to Modern Dynamics Chaos, Networks, Space, and Time. Oxford University Press; 2019. ISBN 9780198844624.

[30] Lorenz EN. Deterministic nonperiodic flow. J Atmosph Sci 1963;20(2):130–41. https://doi.org/10.1175/1520-0469(1963)020<0130:DNF>2.0.CO;2