

MetaPINNs: Predicting soliton and rogue wave of nonlinear PDEs via the improved physics-informed neural networks based on meta-learned optimization

Yanan Guo(郭亚楠), Xiaoqun Cao(曹小群), Junqiang Song(宋君强), and Hongze Leng(冷洪泽)

Citation: Chin. Phys. B, 2024, 33 (**2**): 020203. DOI: 10.1088/1674-1056/ad0bf4
Journal homepage: http://cpb.iphy.ac.cn; http://iopscience.iop.org/cpb

What follows is a list of articles you may be interested in

Residual symmetry, CRE integrability and interaction solutions of two higher-dimensional shallow water wave equations

Xi-Zhong Liu(刘希忠), Jie-Tong Li(李界通), and Jun Yu(俞军)
Chin. Phys. B, 2023, 32 (**11**): 110206. DOI: 10.1088/1674-1056/acf11c

ESR-PINNs: Physics-informed neural networks with expansion-shrinkage resampling selection strategies

Jianan Liu(刘佳楠), Qingzhi Hou(侯庆志), Jianguo Wei(魏建国), and Zewei Sun(孙泽玮)
Chin. Phys. B, 2023, 32 (**7**): 070702. DOI: 10.1088/1674-1056/acc1d5

Meshfree-based physics-informed neural networks for the unsteady Oseen equations

Keyi Peng(彭珂依), Jing Yue(岳靖), Wen Zhang(张文), and Jian Li(李剑)
Chin. Phys. B, 2023, 32 (**4**): 040207. DOI: 10.1088/1674-1056/ac9cb9

Data-driven parity-time-symmetric vector rogue wave solutions of multi-component nonlinear Schrödinger equation

Li-Jun Chang(常莉君), Yi-Fan Mo(莫一凡), Li-Ming Ling(凌黎明), and De-Lu Zeng(曾德炉)
Chin. Phys. B, 2022, 31 (**6**): 060201. DOI: 10.1088/1674-1056/ac4cc5

A nonlocal Boussinesq equation: Multiple-soliton solutions and symmetry analysis

Xi-zhong Liu(刘希忠) and Jun Yu(俞军)
Chin. Phys. B, 2022, 31 (**5**): 050201. DOI: 10.1088/1674-1056/ac43a7

MetaPINNs: Predicting soliton and rogue wave of nonlinear PDEs via the improved physics-informed neural networks based on meta-learned optimization

Yanan Guo(郭亚楠)[1,2,3,†], Xiaoqun Cao(曹小群)[1,2,‡], Junqiang Song(宋君强)[1,2], and Hongze Leng(冷洪泽)[1]

[1] *College of Meteorology and Oceanography, National University of Defense Technology, Changsha* 410073, *China*
[2] *College of Computer, National University of Defense Technology, Changsha* 410073, *China*
[3] *Naval Aviation University, Huludao* 125001, *China*

(Received 25 September 2023; revised manuscript received 6 November 2023; accepted manuscript online 13 November 2023)

Efficiently solving partial differential equations (PDEs) is a long-standing challenge in mathematics and physics research. In recent years, the rapid development of artificial intelligence technology has brought deep learning-based methods to the forefront of research on numerical methods for partial differential equations. Among them, physics-informed neural networks (PINNs) are a new class of deep learning methods that show great potential in solving PDEs and predicting complex physical phenomena. In the field of nonlinear science, solitary waves and rogue waves have been important research topics. In this paper, we propose an improved PINN that enhances the physical constraints of the neural network model by adding gradient information constraints. In addition, we employ meta-learning optimization to speed up the training process. We apply the improved PINNs to the numerical simulation and prediction of solitary and rogue waves. We evaluate the accuracy of the prediction results by error analysis. The experimental results show that the improved PINNs can make more accurate predictions in less time than that of the original PINNs.

Keywords: physics-informed neural networks, gradient-enhanced loss function, meta-learned optimization, nonlinear science

PACS: 02.60.Cb, 07.05.Mh, 02.30.Jr, 05.45.Yv **DOI:** 10.1088/1674-1056/ad0bf4

1. Introduction

In recent years, with the rapid development of artificial intelligence technology, especially deep learning, many fields have undergone significant transformations. Currently, deep learning technology has achieved remarkable milestones across various domains, including object detection, autonomous vehicles, and time series analysis.[1–3] Moreover, deep learning techniques are progressively shifting from being solely data-driven to being driven by the fusion of data and knowledge. It is worth paying attention to the rapid development of this research direction, and the related algorithms are beginning to penetrate into various fields. In the realm of scientific research, deep learning methods are becoming increasingly indispensable as research tools, profoundly advancing scientific discoveries, theoretical derivations, and other related research. They are exerting a rapid and profound influence on the development of disciplines such as physics, mathematics, materials science, and geosciences.[4–7]

So far, scientific research driven by deep learning has resulted in a multitude of noteworthy accomplishments. For example, one notable application of deep learning methods has been in the field of weather and climate prediction, where it has substantially improved the accuracy of forecasts, particularly for highly impactful weather events like super typhoons and extreme precipitation.[8,9] Another exciting area of advancement is the utilization of deep neural networks for solving various types of partial differential equations (PDEs) and modeling complex natural physical phenomena.[10–13] In this context, Raissi *et al.* introduced the concept of physics-informed neural networks (PINNs), which represents an improvement over traditional neural network-based methods for solving forward and inverse problems of partial differential equations. Numerous studies have shown that PINNs can effectively integrate the intrinsic constraint information in the physical equations.[14] Physics-informed neural networks serve as a comprehensive framework that combines data-driven and knowledge-driven approaches for modeling physical phenomena. PINNs differ from traditional neural networks in their unique ability to seamlessly incorporate the physical constraints of PDEs. This innovative approach reduces the need for extensive training data and enhances the interpretability of the model. PINNs are characterized by a flexible loss function that allows for the integration of PDE constraints, as well as boundary and initial condition constraints, in addition to observational data constraints. In recent years, a multitude of researchers have dedicated significant efforts to the study and enhancement of PINNs, introducing various innovative variants and applying them across

[†]Corresponding author. E-mail: guoyn14@lzu.edu.cn
[‡]Corresponding author. E-mail: caoxiaoqun@nudt.edu.cn

© 2024 Chinese Physical Society and IOP Publishing Ltd

diverse domains, including mechanics, geology, oceanography, and meteorology.[15–18] For example, Chen *et al.* proposed a variable coefficient physics-informed neural network (VC-PINN),[19] which specializes in dealing with the forward and inverse problems of variable coefficient partial differential equations. Cao *et al.* adopted an adaptive learning method to update the weight coefficients of the loss function for gradient-enhanced PINNs and dynamically adjusted the weight proportion of each constraint term,[20] and numerically simulated localized waves and complex fluid dynamics phenomena, which achieved better prediction results than the traditional PINNs, and the improved method also performed well for the inverse problem under noise interference.

Solitary waves and rogue waves represent the distinctive phenomenon frequently observed in various natural environments,[21–24] including the atmosphere and oceans. This research area is a cross-discipline of atmospheric science, ocean science, applied mathematics, and engineering technology and is considered an essential frontier research with far-reaching theoretical significance and application prospects.[25] Consequently, solitary waves and rogue waves have emerged as a pivotal focus within the realm of geophysical fluid dynamics, spurring extensive theoretical exploration and high-precision simulation endeavors. However, traditional numerical computation methods encounter challenges such as low computational efficiency, substantial computational resource requirements, and error accumulation. Therefore, there is a growing imperative to explore novel numerical simulation techniques. For example, Chen *et al.* successfully solved the soliton solution of the nonlocal Hirota equation using a multi-layer physics-informed neural network algorithm and investigated the parameter discovery problem of the nonlocal Hirota equation from the perspective of the soliton solution.[26] In addition, Chen *et al.* designed a two-stage PINN method based on conserved quantities that can impose physical constraints from a global perspective,[27] and applied it to multiple types of solitary wave simulations. Numerical results show that the two-stage PINN method has higher prediction accuracy and generalization ability compared with the original PINN method. Li *et al.* combined mixed-training and a priori information to propose mixed-training physics-informed neural networks (PINNs) with enhanced approximation capabilities,[28] and used the model to simulate rogue waves for the nonlinear Schrödinger (NLS) equation, and the numerical results show that the model can significantly improve the prediction accuracy compared with the original PINNs. Even when the spatio-temporal domain of the solution is enlarged or the solution has local sharp regions, the improved model still achieves accurate prediction results. Furthermore, Li *et al.* simulated higher-order rogue waves of the cmKdV equation using mixed-training physics-informed neural networks (MTPINNs) and mixed-training physics-informed neural networks with a priori information (PMTPINNs),[29] which showed significantly better prediction accuracy compared to the original PINNs. In addition, these models have reliable robustness for the inverse problem under noisy data. In this study, we introduce a new structure and loss function specifically designed for modeling solitary and rogue wave phenomena using physics-informed neural networks. We conduct a comprehensive investigation into the simulation of solitary waves and rogue waves employing these enhanced PINNs and validate our approach through a series of numerical experiments.

2. Methodology

2.1. Physics-informed neural networks

Physics-informed neural networks are a class of neural networks used for addressing supervised learning tasks while simultaneously adhering to the physical laws described by partial differential equations. As a machine learning method applied in the scientific computing, PINNs are employed to tackle a wide range of issues related to partial differential equations, encompassing approximate solutions of these equations, parameter inversion, model discovery, control, optimization, and more. After PINNs were proposed, a large amount of follow-up research work was triggered, and they gradually became a research hotspot in the emerging cross-cutting field of scientific machine learning. From the perspective of mathematical function approximation theory, neural network can be regarded as a general nonlinear function approximator, and the modeling process of partial differential equations is also to find nonlinear functions that satisfy the constraints, which are quite consistent with each other. The neural network trained in this way not only approximates the observed data, but also automatically satisfies the physical properties of symmetry, invariance, and conservation followed by the partial differential equations. The following section describes the design method of PINNs using a partial differential equation in generalized form as an example. Let the function $u = u(x,t)$ satisfies a partial differential equation of the following form:

$$u_t + N(u,\lambda) = 0, \quad x \in \Omega, \quad t \in [0,T], \tag{1}$$

where $N(u,\lambda)$ is a generalized function with parameter λ that performs differential operation on u, x is a spatial variable, t is a temporal variable, Ω is a subset of the Euclidean space, and T is the termination moment. The traditional physical model is that given the initial conditions, boundary conditions and physical parameter λ of $u(x,t)$, the value of the $u(x,t)$ at any point in time and space can be predicted by solving the partial differential equation, and when an analytical solution is not

available, it can be solved by using numerical methods, such as finite difference method and finite element method, etc. The PINNs consider the establishment of a neural network $\widehat{u}(x,t)$ to approximate the solution of the partial differential equation. $f(t,x) = \widehat{u}_t + N(\widehat{u},\lambda)$ is the residual of the partial differential equation, and the loss function of the PINN is defined as $\text{Loss} = L_\text{o} + L_\text{IC} + L_\text{BC} + L_\text{PDE}$, where

$$L_\text{o}(\theta;N_\text{o}) = \frac{1}{2|N_\text{o}|}\sum_{j=1}^{N_\text{o}} \left\|\hat{u}\left(x_\text{o}^j,t_\text{o}^j;\theta\right) - u_\text{o}^j\right\|_2^2, \tag{2}$$

$$L_\text{IC}(\theta;N_\text{i}) = \frac{1}{2|N_\text{i}|}\sum_{j=1}^{N_\text{i}} \left\|\widehat{u}\left(x_\text{i}^j,0;\theta\right) - h_\text{i}^j\right\|_2^2, \tag{3}$$

$$L_\text{BC}(\theta;N_\text{b}) = \frac{1}{2|N_\text{b}|}\sum_{j=1}^{N_\text{b}} \left\|\widehat{u}\left(x_\text{b}^j,t_\text{b}^j;\theta\right) - g_\text{b}^j\right\|_2^2, \tag{4}$$

$$L_\text{PDE}(\theta;N_\text{f}) = \frac{1}{2|N_\text{f}|}\sum_{j=1}^{N_\text{f}} \left\|f\left(x_\text{f}^j,t_\text{f}^j;\theta\right)\right\|_2^2. \tag{5}$$

Equation (2) represents the data-driven part of the loss function, and u_o^j represents the training data obtained through observation, numerical simulation and other means. Equations (3) and (4) represent the initial and boundary condition constraints, respectively. Equation (5) represents the physical model-driven part of the loss function, $(x_\text{f}^j,t_\text{f}^j)$ denotes the training collocation point of the domain, which can be utilized by automatic differentiation to efficiently obtain the residual value $f(x_\text{f}^j,t_\text{f}^j;\theta)$. The trained neural network not only needs to approximate the function with the smallest possible error based on the known training data, but also needs to satisfy the constraints of the partial differential equation as much as possible. The network structure of the PINNs is shown in Fig. 1. After inputting the temporal and spatial data, the function is firstly approximated by the fully-connected neural network, and then automatic differentiation is utilized to derive the residuals of the partial differential equation, the initial conditions and boundary condition constraints, and put them into the loss function as regularization terms, and finally use optimization algorithms such as the gradient descent method to obtain the neural network weight parameters and the physical parameters of the partial differential equation.

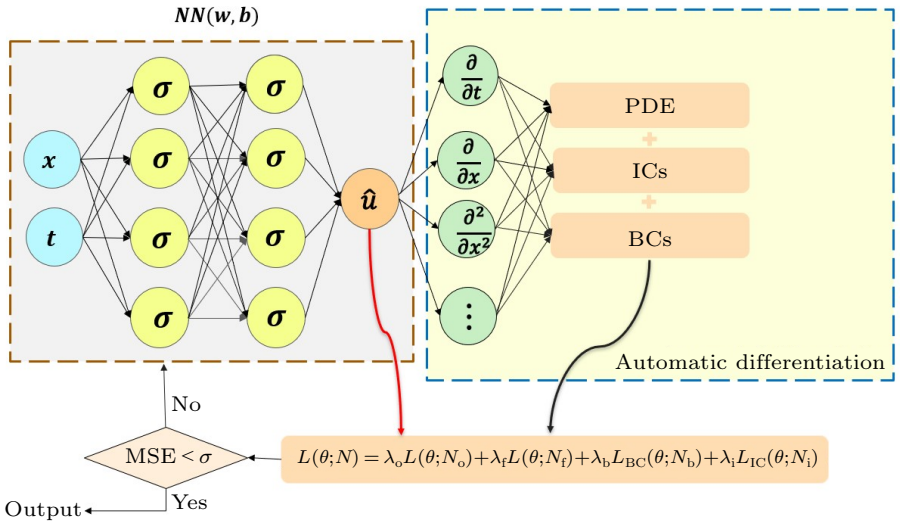


Fig. 1. Schematic of physics-informed neural networks for solving partial differential equation.

2.2. Meta learning

While traditional deep learning methods use fixed learning algorithms to solve problems from scratch, meta-learning aims to improve the learning algorithm itself, which avoids many of the challenges inherent in deep learning, such as data and computational bottlenecks, and generalization problems. Meta-learning is divided into three main categories, namely metrics-based approaches, model-based approaches, and optimization-based approaches. Metric-based approaches learn a metric or distance function within a task, while model-based approaches design an architecture or training process for fast generalization across tasks. Optimization-based methods achieve fast tuning by adjusting the optimization function using only a small number of samples. The framework of meta-learning is widely used in small-sample learning, and its core idea is to learn generalized prior knowledge on similar learning tasks so that only a few samples are needed, and the prior knowledge is used to quickly fine-tune on a new task to achieve the desired result. Therefore, the key of meta-learning is to discover the universal laws between different problems and utilize the universal laws to solve new tasks and problems. The universal laws need to achieve a balanced representation of the generalities and characteristics of the tasks. The search for universal laws mainly relies on the following points.

(i) Discovering and summarizing the links between already solved problems and new problems, and extracting the

universal laws of already solved problems for use in solving new problems.

(ii) Decomposing and simplifying the new problem, finding universal laws in the already solved problem that are closely linked to the various subtasks of the new problem, and analyzing the scope of application of these laws.

(iii) Learning logical reasoning in new tasks, using logical reasoning to make representations of new problems, finding laws in these representations, and finding solutions to complex new tasks through logical reasoning between the parts of the new problem.

Meta-learning is essentially a bilevel optimization problem, where one optimization problem is nested within another optimization problem. The external optimization problem and the internal optimization problem are often referred to as the upper level optimization problem and the lower level optimization problem, respectively. The two-layer optimization problem involves two participants: the upper layer participant is the meta-learner and the lower layer participant is the base-learner. The optimal decision of the meta-learner depends on the response of the base-learner, which itself optimizes its own internal decisions. These two levels have their own different objective functions, constraints and decision variables. The objects and functions of the base-learner and meta-learner are shown in Fig. 2. Base-learner is the model in the base layer. The data set on a single task is considered while training the base-learner each time. Meta-learner is the model in the meta-layer that summarizes the training experience on all tasks. After training the base-learner each time, the meta-learner synthesizes the new experience and updates the parameters in the meta-learner. The main purpose of meta-learning is to find the meta-learner F, under the guidance of F, the base-learner q can get the optimal state q^* adapted to the current new task after a few steps of fine-tuning under the support set D^{tr}. And the optimization of F requires the cumulative sum of the losses of all current tasks, i.e., $\nabla\sum_{n=1}^{N} l\left(q_n, D_n^{\mathrm{te}}\right)$. The meta-learning works as shown in Fig. 3.

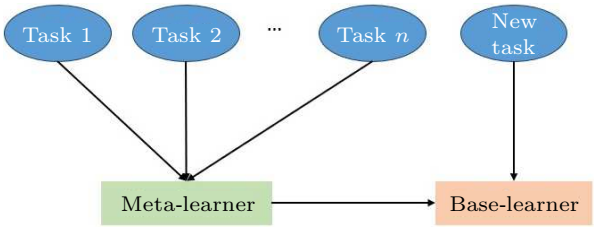


Fig. 2. Schematic diagram of the base-learner and the meta-learner.

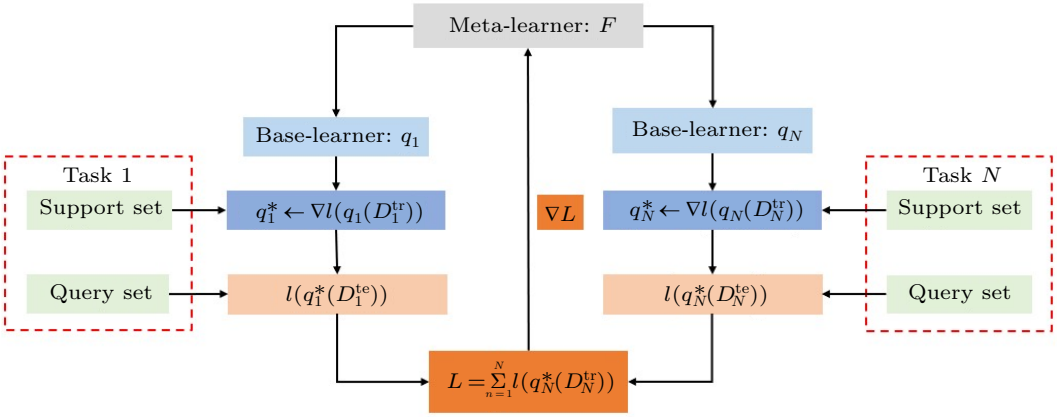


Fig. 3. The schematic of meta learning's working mechanism.

2.3. The improved physics-informed neural networks

In the original PINNs, only the PDE residuals f are constrained to be zero. However, upon further analysis, it becomes evident that if $f(\boldsymbol{x})$ is zero for any $\boldsymbol{x}$, then its derivative, $\nabla f(\boldsymbol{x})$, is also zero. Therefore, we make the assumption that the exact solution of the PDE possesses sufficient smoothness to warrant the existence of the gradient of PDE residuals, $\nabla f(\boldsymbol{x})$. Consequently, we introduce gradient-enhanced PINNs, where not only are the PDE residuals zero, but also their derivatives, as described mathematically by the following equation:[30]

$$\nabla f(\boldsymbol{x}) = \left(\frac{\partial f}{\partial x_1}, \frac{\partial f}{\partial x_2}, \ldots, \frac{\partial f}{\partial x_d}\right) = \boldsymbol{0}, \quad \boldsymbol{x} \in \Omega. \tag{6}$$

Based on the aforementioned definition, it is inherent to derive the loss function for gradient-enhanced PINNs

$$\begin{aligned} L(\boldsymbol{\theta}; N) = {} & w_{\mathrm{o}} L_{\mathrm{o}}\left(\boldsymbol{\theta}; N_{\mathrm{o}}\right) + w_{\mathrm{f}} L_{\mathrm{f}}\left(\boldsymbol{\theta}; N_{\mathrm{f}}\right) + w_{\mathrm{b}} L_{\mathrm{b}}\left(\boldsymbol{\theta}; N_{\mathrm{b}}\right) \\ & + w_{\mathrm{i}} L_{\mathrm{i}}\left(\boldsymbol{\theta}; N_{\mathrm{i}}\right) + \sum_{i=1}^{d} w_{\mathrm{g}_i} L_{\mathrm{g}_i}\left(\boldsymbol{\theta}; \mathcal{T}_{\mathrm{g}_i}\right), \end{aligned} \tag{7}$$

where the loss term of the derivative with respect to x_i is

$$L_{\mathrm{g}_i}\left(\boldsymbol{\theta}; \mathcal{T}_{\mathrm{g}_i}\right) = \frac{1}{\left|\mathcal{T}_{\mathrm{g}_i}\right|} \sum_{\boldsymbol{x} \in \mathcal{T}_{\mathrm{g}_i}} \left|\frac{\partial f}{\partial x_i}\right|^2. \tag{8}$$

Here, $\mathcal{T}_{\mathrm{g}_i}$ represents the set of residual points for the derivative $\partial f / \partial x_i$. It is important to emphasize that $\mathcal{T}_{\mathrm{g}_i}$ (for $i = 1, 2, \ldots, d$) may vary between different equations. To enhance

the generalization capabilities of PINNs, this study integrates meta-learning into their training process.

While the original optimizer uses manually designed optimization rules, such as the Adam optimizer for updating the weight vector $\boldsymbol{\theta}$ of a physics-informed neural network with loss function $L(\boldsymbol{\theta})$, meta-learning can improve upon these previously manually designed optimization rules. Specifically, the original Adam update rule is as follows:

$$\begin{aligned}
&\boldsymbol{m}_t = \beta_1 \boldsymbol{m}_{t-1} + (1-\beta_1)\nabla_\theta L(\boldsymbol{\theta}_{t-1}),\\
&\boldsymbol{v}_t = \beta_2 \boldsymbol{v}_{t-1} + (1-\beta_2)\left(\nabla_\theta L(\boldsymbol{\theta}_{t-1})\right)^2,\\
&\hat{\boldsymbol{m}}_t = \boldsymbol{m}_t / \left(1-\beta_1^t\right), \quad \hat{\boldsymbol{v}}_t = \boldsymbol{v}_t / \left(1-\beta_2^t\right),\\
&\boldsymbol{\theta}_t = \boldsymbol{\theta}_{t-1} - \eta \boldsymbol{w}_{\text{adam}} = \boldsymbol{\theta}_t - \eta \hat{\boldsymbol{m}}_t / \left(\sqrt{\hat{\boldsymbol{v}}_t} + \boldsymbol{\varepsilon}\right), \qquad (9)
\end{aligned}$$

where $\boldsymbol{m}_t$ and $\boldsymbol{v}_t$ are the first and second moments at time step t, respectively, with $\boldsymbol{\varepsilon}$ being a small regularization constant used for numerical stability, $\beta_1, \beta_2 \in [0,1)$ being the exponential decay rates for the moment estimates, and η being the learning rate. The meta-learned optimizer's parameters undergo updates through the following process:

$$\boldsymbol{\theta}_t = \boldsymbol{\theta}_{t-1} - \boldsymbol{f}(\boldsymbol{z}_t; \boldsymbol{\vartheta}), \qquad (10)$$

where $\boldsymbol{f}$ is the parameter update function of the learnable optimizer, $\boldsymbol{z}_t$ is the input feature, and $\boldsymbol{\vartheta}$ is the trainable meta-parameter. For the component form, each parameter θ_{i} of the updated parameter vector $\boldsymbol{\theta}$ is computed in the same way. The optimizer architecture used in this study utilizes a multilayer perceptron (MLP), which is simple, but better illustrates the potentiality of the approach in this paper. Specifically, the update formula for each parameter θ_{i} is as follows:

$$f = \lambda_1 \exp\left(\lambda_2 s_{\boldsymbol{\vartheta}}^{\text{adam}}\right) w_{\text{adam}} + \frac{\lambda_3}{\sqrt{v_t}+\boldsymbol{\varepsilon}} d_{\boldsymbol{\vartheta}}^{\text{bb}} \exp\left(\lambda_4 s_{\boldsymbol{\vartheta}}^{\text{bb}}\right). \qquad (11)$$

The above equation gives the positive constant λ_i, $i = 1,2,3,4$, and the w_{adam} corresponding to the Adam update step. The $s_{\boldsymbol{\vartheta}}^{\text{adam}}$, $s_{\boldsymbol{\vartheta}}^{\text{bb}}$ and $d_{\boldsymbol{\vartheta}}^{\text{bb}}$ correspond to the output head of the meta-learning optimizer, while the neural network weights are $\boldsymbol{\vartheta}$.

The learnable update formula (11) consists of two terms. The first term is derived from the Adam update formula and has a scalable learning rate $\lambda_1 \exp(\lambda_2 s_{\boldsymbol{\vartheta}}^{\text{adam}})$, which ensures that the update step is in the descent direction. The second term corresponds to the black box update term, which is the product of the direction term $d_{\boldsymbol{\vartheta}}^{\text{bb}}$ and the size term $\lambda_1 \exp(\lambda_4 s_{\boldsymbol{\vartheta}}^{\text{bb}})$. The denominator $\sqrt{v_t}+\boldsymbol{\varepsilon}$ acts as a preprocessor and guarantees that the overall update formula leads to a decrease in the loss surface.

The inputs $\boldsymbol{z}_t$ provided for the MLP optimizer with output head $d_{\boldsymbol{\vartheta}}^{\text{bb}}$, $s_{\boldsymbol{\vartheta}}^{\text{adam}}$ and $s_{\boldsymbol{\vartheta}}^{\text{bb}}$ at optimization step t are specified as follows:

(i) The neural network weights $\boldsymbol{\theta}_t$.

(ii) The gradients of loss function $\nabla_\theta L(\boldsymbol{\theta}_t)$.

(iii) The second momentum accumulators $\boldsymbol{v}_t$ with decay rates $\beta_2 \in \{0.5, 0.9, 0.99, 0.999\}$ and inverse of the square root of these second momentum accumulators.

(iv) The time step t.

The present study improved physics-informed neural networks by utilizing the aforementioned input parameters. All input features, except for the time step, were normalized such that one of their second-order moment was equal to 1. The time step was transformed into a total of eleven features by computing $\tanh(t/x)$, where $x \in \{1,3,10,30,100,300,1000,3000,10\text{k},30\text{k},100\text{k}\}$. Subsequently, all features were concatenated and inputted into the MLP to produce the aforementioned three output heads.

3. Numerical experiments and results

In this section, we conduct experiments to investigate the solitary wave and rogue wave phenomena utilizing gradient-enhanced PINNs with meta-learned optimization. Specifically, we seek to approximate the solitary wave solutions of the Kaup–Kuperschmidt (KK) equation and rogue waves of the nonlinear Schrödinger equation. At the end of model training, we quantitatively evaluate the performance of the enhanced PINN model for solitary wave and rogue wave simulation. This assessment encompasses an in-depth analysis of its convergence rate and the accuracy of the prediction results. The accuracy of the approximate solutions generated by the enhanced PINN model is quantified using the L2 norm relative error, defined as follows:

$$\text{L2 relative error} = \frac{\sqrt{\sum_{i=1}^{N} \left|\hat{u}(x_i,t_i) - u(x_i,t_i)\right|^2}}{\sqrt{\sum_{i=1}^{N} \left|u(x_i,t_i)\right|^2}}, \qquad (12)$$

where N represents the number of grid points, $\hat{u}(x_i,t_i)$ indicates the approximate solution generated by the neural network model, and $u(x_i,t_i)$ represents the exact solution.

3.1. Numerical simulation of solitary wave of the Kaup–Kuperschmidt equation

Next, we conduct experimental research on the Kaup–Kuperschmidt equation,[31,32] which is mathematically defined as follows:

$$u_t + 20u^2 u_x + 25u_x u_{xx} + 10uu_{xxx} + u_{xxxxx} = 0. \qquad (13)$$

The KK equation is commonly used to describe the evolution and nonlinear effects of water waves. It has a wide range of applications, including oceanography, fluid dynamics, and nonlinear optics, making it an important tool for studying various nonlinear wave phenomena. Meanwhile, researchers have found that the KK equation can support multiple solutions, including solitary waves and other nonlinear wave solutions,

which play a crucial role in the propagation and evolution of water waves. However, the study and analysis of the KK equation are typically very complex because it is a highly nonlinear partial differential equation.

For Eq. (13), we meticulously defined our numerical simulation parameters, limiting our spatial exploration to the interval $[-20,20]$ and our temporal investigation to the period $[-5,5]$. These parameter choices were guided by careful consideration of the following set of initial condition and boundary condition:

$$u(x,-5)=\frac{24\mathrm{e}^{x+5}\left(16+4\mathrm{e}^{x+5}+\mathrm{e}^{2x+10}\right)}{\left(16+16\mathrm{e}^{x+5}+\mathrm{e}^{2x+10}\right)^2}, \tag{14}$$

$$u(-20,t)=\frac{24k_1^2\mathrm{e}^{k_1(-20)-k_1^5t}\left(16+4\mathrm{e}^{k_1(-20)-k_1^5t}+\mathrm{e}^{2(k_1(-20)-k_1^5t)}\right)}{\left(16+16\mathrm{e}^{k_1(-20)-k_1^5t}+\mathrm{e}^{2(k_1(-20)-k_1^5t)}\right)^2}, \tag{15}$$

$$u(20,t)=\frac{24k_1^2\mathrm{e}^{k_1(20)-k_1^5t}\left(16+4\mathrm{e}^{k_1(20)-k_1^5t}+\mathrm{e}^{2(k_1(20)-k_1^5t)}\right)}{\left(16+16\mathrm{e}^{k_1(20)-k_1^5t}+\mathrm{e}^{2(k_1(20)-k_1^5t)}\right)^2}. \tag{16}$$

In accordance with the previously stated conditions, the solitary wave solution of the KK equation is expressed as follows:

$$u(x,t)=\frac{24k_1^2\mathrm{e}^{\theta_1}\left(16+4\mathrm{e}^{\theta_1}+\mathrm{e}^{2\theta_1}\right)}{\left(16+16\mathrm{e}^{\theta_1}+\mathrm{e}^{2\theta_1}\right)^2}, \tag{17}$$

where $\theta_1=k_1x-k_1^5t$ and $u(x,t)$ describes the amplitude of the wave at position x and time t.

To effectively approximate the precise solution of the KK equation, we employed a neural network architecture consisting of five hidden layers, each comprising 64 neurons. In terms of the initial and boundary conditions, we generated a training dataset through random sampling, incorporating 400 samples. We selected random spatiotemporal locations to calculate residuals, facilitating the formulation of physical constraints. A total of 15000 data points were sampled during the experiment. Furthermore, a limited set of observations, consisting of 400 data points, was employed. Utilizing these data samples, we devised a composite loss function. We utilized a learnable optimizer to optimize the loss function, with the optimizer's constant set as $\lambda_j=10^{-4}$. Following the training process, the neural network model was employed to approximate the solitary wave solution of the Kaup–Kuperschmidt equation. In Fig. 4, both the exact solution and the predictions generated by the trained model are presented. From Fig. 4, it is evident that the enhanced PINNs can accurately predict the precise solution of the KK equation with high precision. Subsequently, for a detailed analysis of the experimental outcomes, we selectively examine the prediction results at different moments, comparing them with the exact solution to assess the accuracy of the improved PINNs. Figure 5 provides a visual representation of the comparison between the exact solution and the approximated solution at different times. The prediction values generated by the enhanced PINNs closely match the exact solutions, signifying a noteworthy degree of accuracy. Furthermore, following multiple experiments, we quantify the disparities between the approximate solution and the exact solution. The L2 norm relative error between the predictions generated by the enhanced PINNs and the exact solution is approximately 8.20×10^{-4}. With the same training data and parameter configurations, the L2 norm relative error between the predictions made by the original PINNs and the true solution is approximately 5.22×10^{-3}. Figure 6 displays the error statistics for both the improved PINNs and the original PINNs. Furthermore, we conduct an analysis of the model's training time before and after the enhancements.

Table 1. For the KK equation, the errors of the predicted solutions are compared at different numbers of the neural network layers, different numbers of neurons, and different numbers of collocation points.

	$N_\mathrm{f}=10000$		$N_\mathrm{f}=15000$		$N_\mathrm{f}=20000$	
Hidden layers–neurons	PINNs (10^{-3})	IPINNs (10^{-4})	PINNs (10^{-3})	IPINNs (10^{-4})	PINNs (10^{-3})	IPINNs (10^{-4})
4–40	9.774	12.07	9.076	9.572	8.662	9.040
4–50	9.243	9.823	8.924	9.260	8.172	8.912
4–60	9.061	9.557	8.070	9.324	7.956	8.526
5–40	7.352	9.440	7.073	9.071	6.592	8.013
5–50	7.126	9.159	6.895	8.765	5.906	7.805
5–60	6.916	8.930	5.970	8.270	5.352	7.040
6–40	6.530	8.590	5.340	8.026	4.726	6.229
6–50	6.464	8.012	5.029	7.580	4.354	5.430
6–60	5.980	7.632	4.970	6.985	4.070	5.114

The results indicate that the enhanced PINNs complete the training process in approximately 47.6% of the time required by the original PINNs. To more thoroughly analyze the effects of the number of neural network layers, the number of neurons and the number of collocation points on the prediction solution error, we chose different numbers of the hidden layers, different numbers of neurons and different numbers of collocation points to conduct numerical experiments and compared the prediction solution errors of the original PINNs and the improved PINNs. The experimental results are shown in Table 1. Analyzing the experimental results, it can be found that as the number of neural network layers increases, the number of neurons increases and the number of collocation points increases, the prediction solution error decreases. In addition, it is evident that the improved PINNs exhibit superior training efficiency and yield more accurate prediction outcomes compared to the original PINNs when simulating a single solitary wave of the KK equation.

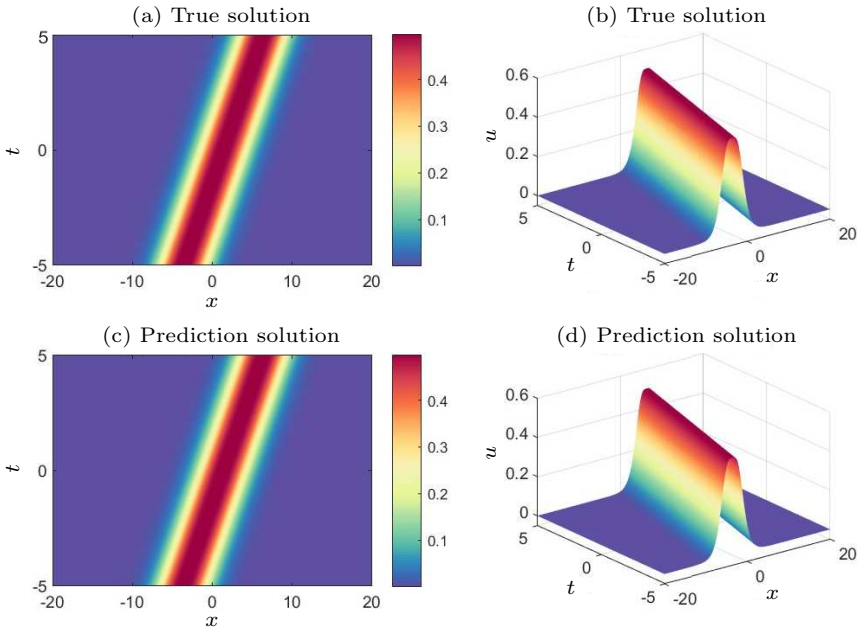


Fig. 4. Exact soliton solutions and predicted solutions of the KK equation: (a) two-dimensional visualization of the true solution, (b) three-dimensional visualization of the true solution, (c) two-dimensional visualization of the predicted solution of the improved PINNs, (d) three-dimensional visualization of the predicted solution of the improved PINNs.

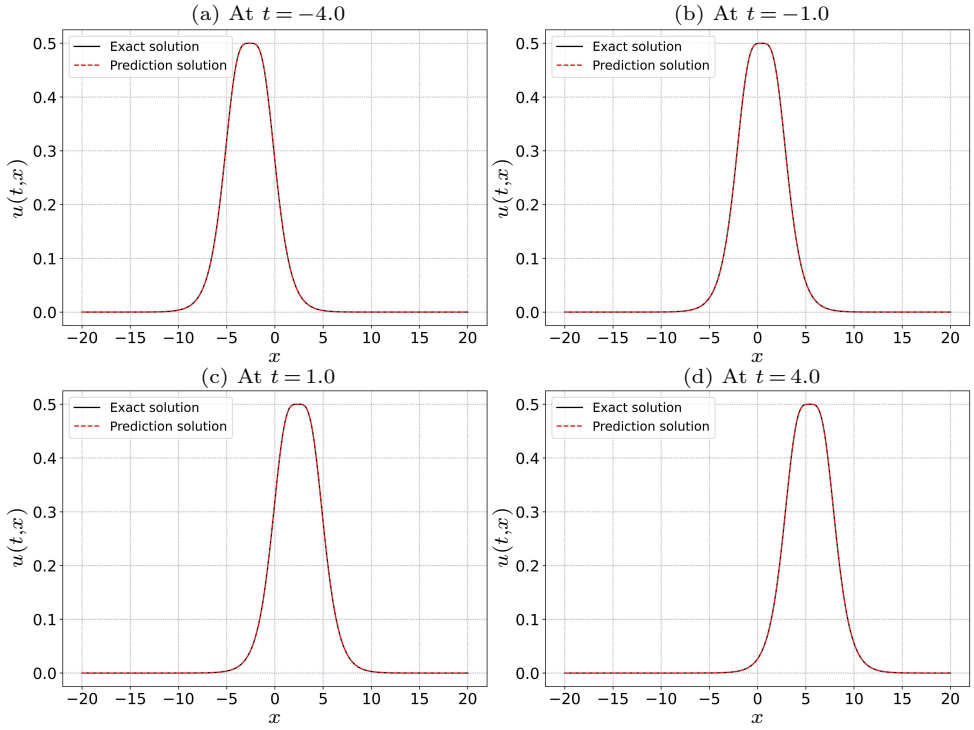


Fig. 5. Comparison of the exact solitary wave solution to the predicted solitary wave solution of the KK equation at four different time instances: (a) $t=-4.0$, (b) $t=-1.0$, (c) $t=1.0$, and (d) $t=4.0$.

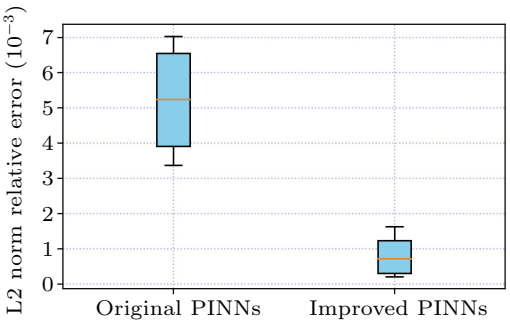


Fig. 6. Visual analysis of errors in the prediction results of the solitary wave solution of the KK equation: comparison of the errors of the improved PINNs and the original PINNs.

3.2. Numerical simulation of first-order rogue wave of the nonlinear Schrödinger equation

In order to illustrate the applicability of the improved PINN to rogue wave prediction, we conduct experimental research on the nonlinear Schrödinger equation,[33–35] which is mathematically defined as follows:

$$\mathrm{i}u_t + u_{xx} + |u|^2 u = 0. \tag{18}$$

The following solution is obtained by the Hirota transformation:

$$u = \exp(\mathrm{i}\theta)\frac{g_N}{f_N}, \quad \theta = kx - \left(k^2 - 1\right)t, \tag{19}$$

where

$$f_N = \sum_{\mu=0,1} \exp\left(\sum_{j=1}^{2N} \mu_j \Gamma_j + \sum_{1\le j<l}^{2N} \mu_j \mu_l A_{jl}\right), \tag{20}$$

$$g_N = \sum_{\mu=0,1} \exp\left(\sum_{j=1}^{2N} \mu_j \Omega_j + \sum_{1\le j<l}^{2N} \mu_j \mu_l A_{jl}\right), \tag{21}$$

$$\begin{aligned}
&\Omega_j = \mathrm{i}\sqrt{2}x\sin\phi_j - \left(2\sqrt{2}\mathrm{i}k\sin\phi_j + 2\sin^2\phi_j\cot\phi_j\right)t \\
&\quad + 2\mathrm{i}\phi_j + \Omega_j^{(0)}, \quad j = 1,2,\ldots,2N, \\
&\Gamma_j = \Omega_j - 2i\phi_j, \ \ \phi_n = \phi_{n+N}^* + \pi, \ \ \Omega_n^{(0)} = \Omega_{n+N}^{(0)*}, \\
&\exp\left(A_{jl}\right) = \left[\frac{\sin\frac{1}{2}\left(\phi_j - \phi_l\right)}{\sin\frac{1}{2}\left(\phi_j + \phi_l\right)}\right]^2, \ \ n = 1,2,\ldots,N.
\end{aligned} \tag{22}$$

In the above equations, $\Omega_n^{(0)}$ and ϕ_n are free complex parameters and k is a real parameter. In this subsection, the value of k is zero. For the Eq. (18), the study domain for the numerical simulations is set to the spatial range $[-5,5]$ and the time range $[-5,5]$. When choosing these parameters, we carefully consider the following set of initial and boundary conditions:

$$\begin{aligned}
u(x,-5) = &\left(1 - \frac{-40\mathrm{i}+4}{200k^2+40kx+100+2x^2+1}\right) \\
&\times \mathrm{e}^{-\mathrm{i}\left(-5k^2-kx+5\right)},
\end{aligned} \tag{23}$$

$$\begin{aligned}
u(-5,t) = &\left(1 - \frac{8\mathrm{i}t+4}{8k^2t^2+40kt+4t^2+51}\right) \\
&\times \mathrm{e}^{-\mathrm{i}\left(k^2t+5k-t\right)},
\end{aligned} \tag{24}$$

$$\begin{aligned}
u(5,t) = &\left(1 - \frac{8\mathrm{i}t+4}{8k^2t^2-40kt+4t^2+51}\right) \\
&\times \mathrm{e}^{-\mathrm{i}\left(k^2t-5k-t\right)}.
\end{aligned} \tag{25}$$

In the experiment, simulations are performed for the first-order rogue wave of the nonlinear Schrödinger equation. First, we construct a deep neural network with six hidden layers, each containing 64 neurons. During training, the loss function is optimized using a learnable optimizer with a constant set to $\lambda_j = 10^{-4}$. All weights and biases in the model are initialized using the Xavier method. For consistency and ease of performance comparison of subsequent methods, we still use the initial conditions as special Dirichlet boundary conditions on the spatio-temporal domain. It is worth noting that the number of observations is only 400, whereas 10000 collocation points were selected for the construction of the equation constraints and 400 sample points were selected for the construction of the initial condition constraints and boundary condition constraints. Figure 7 compares the difference between the prediction results of the improved PINNs and the true solution. The relatively good prediction results convince us that the improved PINNs can effectively predict the first-order rogue wave of the nonlinear Schrödinger equation. Figure 8 shows the comparison of the prediction results between the exact solutions and the improved PINNs at

Table 2. For the first-order rogue wave of the nonlinear Schrödinger equation, the errors of the predicted solutions are compared at different numbers of the neural network layers, different numbers of neurons, and different numbers of collocation points.

	$N_f = 10000$		$N_f = 15000$		$N_f = 20000$	
Hidden layers–neurons	PINNs (10^{-3})	IPINNs (10^{-4})	PINNs (10^{-3})	IPINNs (10^{-4})	PINNs (10^{-3})	IPINNs (10^{-4})
5–40	8.657	21.06	6.965	16.13	6.276	10.13
5–50	8.353	14.27	6.243	10.05	5.882	8.920
5–60	8.046	9.513	6.032	7.624	5.160	6.542
6–40	7.385	9.054	6.100	6.291	5.026	5.925
6–50	6.934	8.905	5.822	5.540	4.610	5.416
6–60	6.572	8.516	5.259	5.162	4.045	5.250
7–40	6.125	8.329	5.247	4.926	3.869	3.535
7–50	5.307	7.226	4.639	4.726	3.056	2.440
7–60	5.252	7.122	4.255	4.170	2.830	2.025

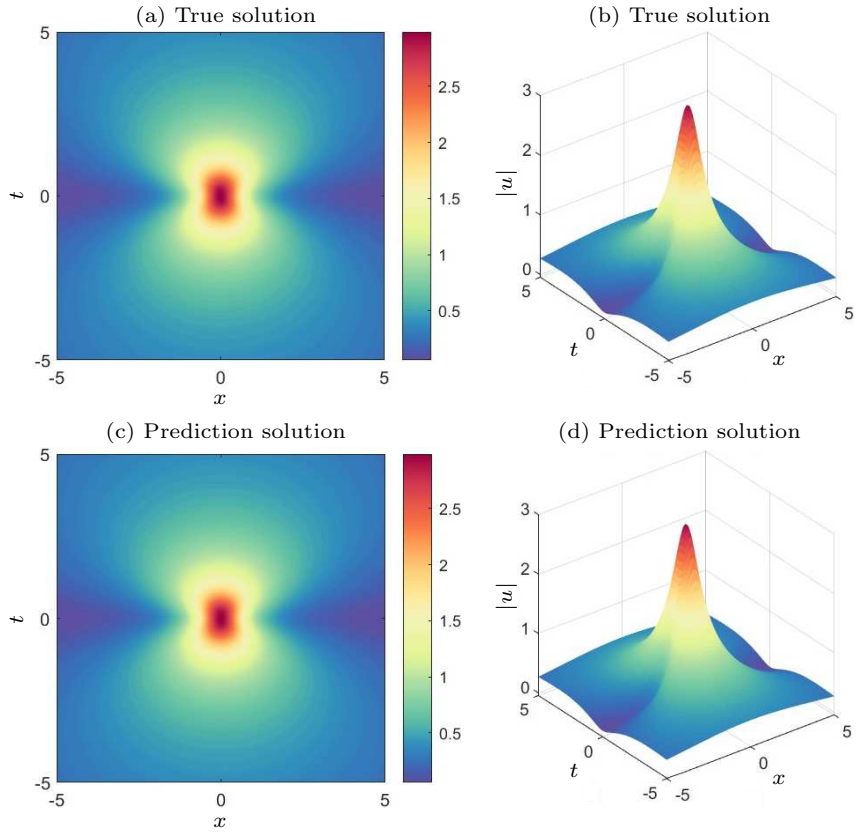


Fig. 7. Exact first-order rogue wave solutions and predicted solutions of the nonlinear Schrödinger equation: (a) two-dimensional visualization of the true solution, (b) three-dimensional visualization of the true solution, (c) two-dimensional visualization of the predicted solution of the improved PINNs, (d) three-dimensional visualization of the predicted solution of the improved PINNs.

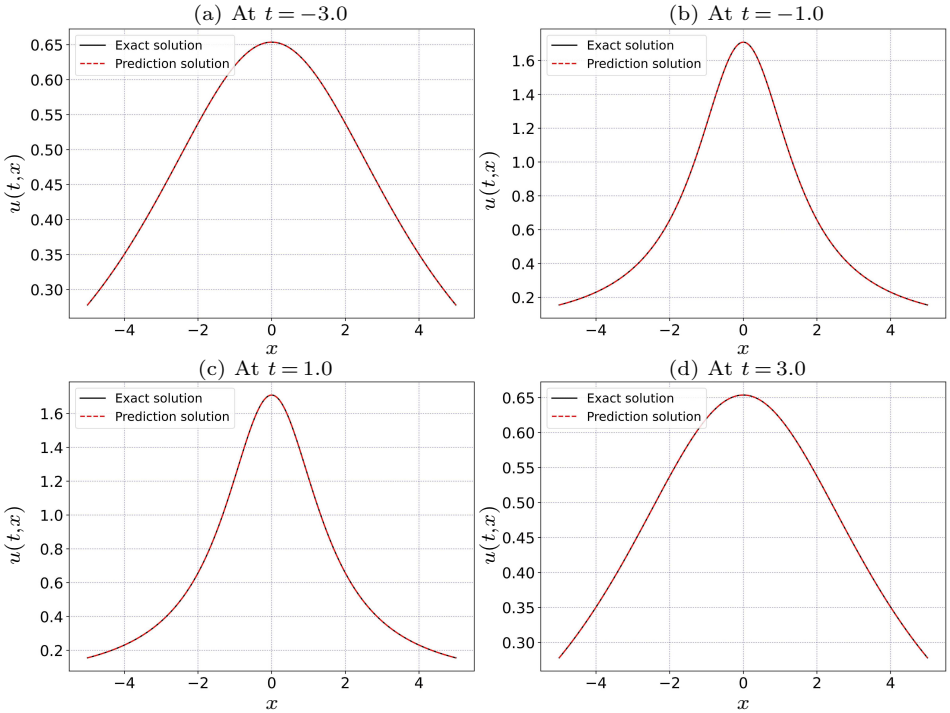


Fig. 8. Comparison of the exact first-order rogue wave solution to the predicted first-order rogue wave solution of the nonlinear Schrödinger equation at four different time instances: (a) $t=-3.0$, (b) $t=-1.0$, (c) $t=1.0$, and (d) $t=3.0$.

different moments, and we can find that the two solutions are in high consistency. Similarly, we calculate the L2 norm relative error of the prediction results of the improved PINNs with respect to the true solution, which has a value of 8.33×10^{-4}, while the L2 norm relative error of the original PINNs with the same experimental setup is 6.41×10^{-3}, and the comparison results are given in Fig. 9. As for the computational efficiency, the time required for the improved PINNs to complete

the training is only 51.6% of the training time of the original PINNs. In this study, we also quantitatively analyze the effects of the number of neural network layers, the number of neurons, and the number of collocation points on the prediction solution of the first-order rogue waves of the nonlinear Schrödinger equation, and select different numbers of hidden layers, different numbers of neurons, and different numbers of collocation points for numerical experiments, and compare the prediction solution errors of the original PINNs and the improved PINNs. The specific experimental results are shown in Table 2. Analyzing the experimental results, it can be found that as the number of neural network layers increases, the number of neurons increases, and the number of collocation points increases, the prediction solution error is also reduced. The experiments can show that the improved PINNs outperform the original PINNs in terms of both computational efficiency and prediction accuracy.

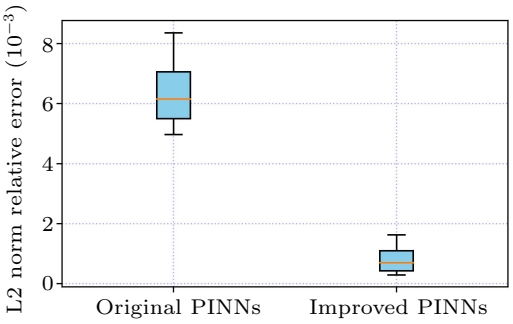


Fig. 9. Visual analysis of errors in the prediction results of the first-order rogue wave solution of the nonlinear Schrödinger equation: comparison of the errors of the improved PINNs and the original PINNs.

3.3. Numerical simulation of second-order rogue wave of the nonlinear Schrödinger equation

In this subsection, we continue our study of the second-order rogue wave of the nonlinear Schrödinger equation. For the second-order rogue wave solution of Eq. (18), the study region for the numerical simulations is set to the spatial interval $[-10,10]$ and the time interval $[-10,10]$. In addition, the value of k is likewise set to zero. With these parameters, we carefully consider the following set of initial and boundary conditions:

$$u(x,-10)=\exp(\mathrm{i}(kx+10(k^2-1)))\frac{g_2(x,-10)}{f_2(x,-10)}, \tag{26}$$

$$u(-10,t)=\exp(\mathrm{i}(-10k-(k^2-1)t))\frac{g_2(-10,t)}{f_2(-10,t)}, \tag{27}$$

$$u(10,t)=\exp(\mathrm{i}(10k-(k^2-1)t))\frac{g_2(10,t)}{f_2(10,t)}, \tag{28}$$

where

$$\begin{aligned} f_2(x,t) &= 8\left(2t^2+x^2\right)^3+12\left(6t^2-x^2\right)^2-32Ct\left(2t^2-3x^2\right)\\ &\quad +54x^2+396t^2-144Ct+16C^2+9,\\ g_2(x,t) &= 8\left(2t^2+x^2\right)^3-96\mathrm{i}t\left(2t^2+x^2\right)^2-36x^2\left(20t^2+x^2\right)\\ &\quad -528t^4-32\left(2t^2-3x^2\right)t\left(C+3\mathrm{i}\right)\\ &\quad +6\mathrm{i}\left(15\mathrm{i}-16C\right)x^2+12\mathrm{i}\left(39\mathrm{i}+16C\right)t^2\\ &\quad +48Ct+360\mathrm{i}t-48\mathrm{i}C+16C^2+45. \end{aligned} \tag{29}$$

In the above equation, the value of C is 100. First, we construct a deep neural network that is with seven hidden layers and 64 neurons in each hidden layer. During the training process, the learnable optimizer is still utilized to optimize the loss function, and the constants of the optimizer are set to $\lambda_j=10^{-4}$. In addition, we still use the initial conditions as special Dirichlet boundary conditions on the spatio-temporal domain. The number of observations in this experiment is 400, and 20000 collocation points were selected for the construction of the differential equation constraints, and 400 sample points were selected for the initial condition constraints and boundary condition constraints. Figure 10 compares the difference between the prediction results of the improved PINNs and the true solution, from which we find that the improved PINNs obtain prediction results close to the true solution. Figure 11 shows the comparison between the prediction results of the improved PINNs and the true solution at different moments, and again we can find a high consistency between the two. Similarly, we calculate the L2 norm relative error of the prediction results of the improved PINNs with respect to the true solution, which has a value of 7.10×10^{-4}, while

Table 3. For the second-order rogue wave of the nonlinear Schrödinger equation, the errors of the predicted solutions are compared at different numbers of the neural network layers, different numbers of neurons, and different numbers of collocation points.

	$N_\mathrm{f}=10000$		$N_\mathrm{f}=15000$		$N_\mathrm{f}=20000$	
Hidden layers–neurons	PINNs (10^{-3})	IPINNs (10^{-4})	PINNs (10^{-3})	IPINNs (10^{-4})	PINNs (10^{-3})	IPINNs (10^{-4})
6–40	9.627	35.81	9.043	11.24	8.693	9.012
6–50	9.230	30.50	8.954	10.56	8.355	8.520
6–60	9.019	19.85	8.787	9.557	8.240	8.226
7–40	8.875	15.40	8.560	9.220	8.126	8.070
7–50	8.658	9.693	8.376	9.152	7.830	7.692
7–60	8.190	9.342	7.990	8.740	7.479	7.110
8–40	7.795	9.178	7.354	8.615	7.260	7.020
8–50	7.594	9.105	7.242	8.520	6.965	6.528
8–60	7.247	8.896	6.865	7.317	6.042	5.260

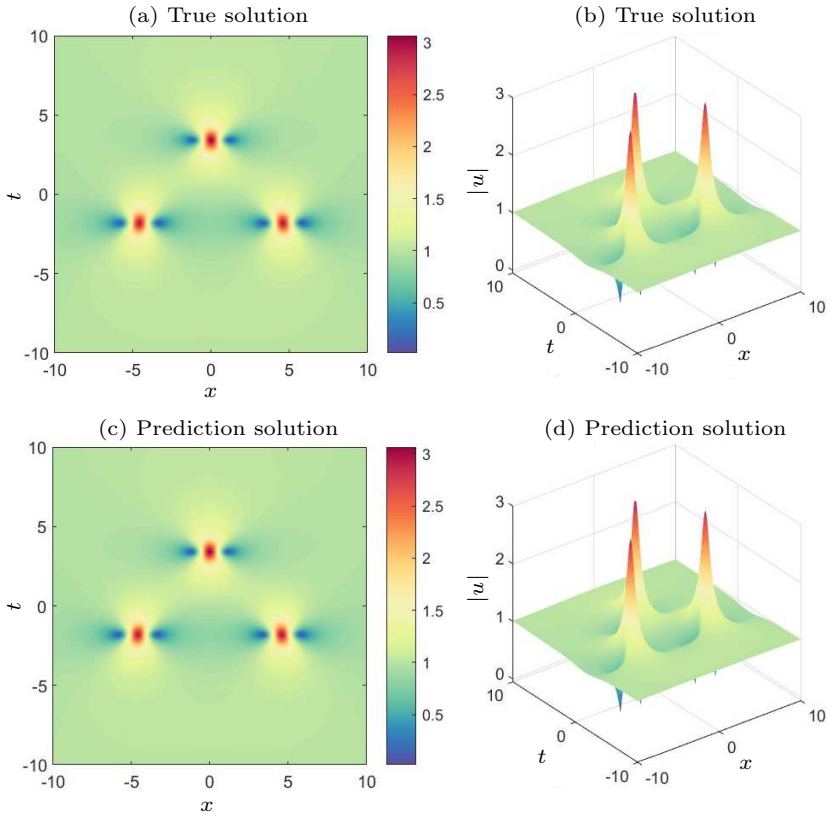


Fig. 10. Exact second-order rogue wave solutions and predicted solutions of the nonlinear Schrödinger equation: (a) two-dimensional visualization of the true solution, (b) three-dimensional visualization of the true solution, (c) two-dimensional visualization of the predicted solution of the improved PINNs, (d) three-dimensional visualization of the predicted solution of the improved PINNs.

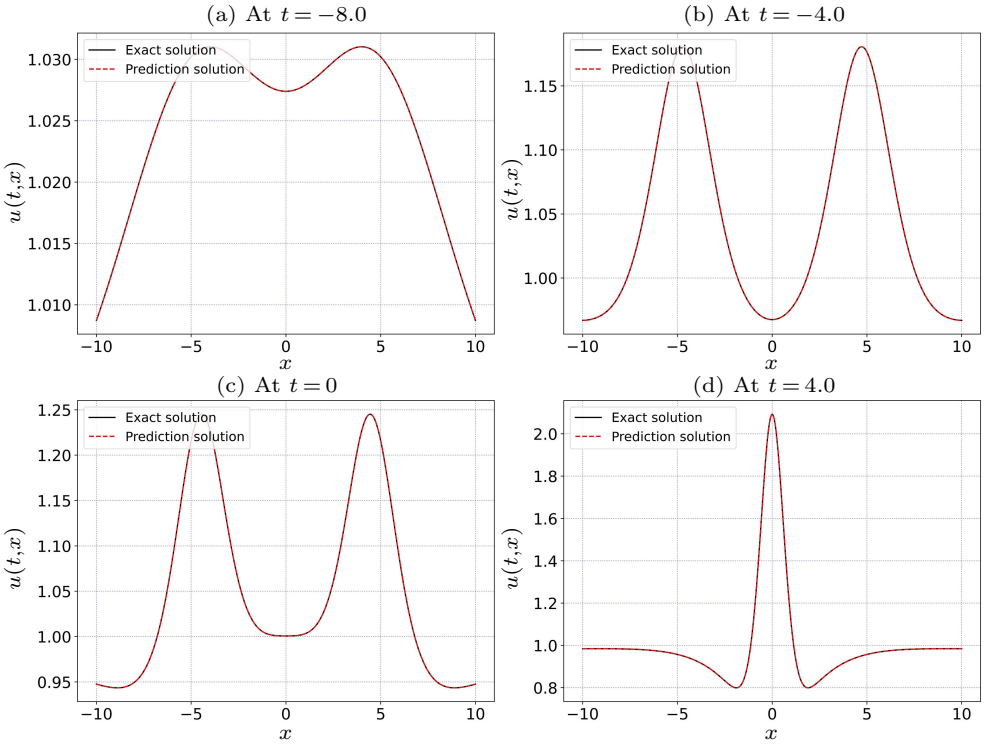


Fig. 11. Comparison of the exact second-order rogue wave solution to the predicted second-order rogue wave solution of the nonlinear Schrödinger equation at four different time instances: (a) $t = -8.0$, (b) $t = -4.0$, (c) $t = 0.0$, and (d) $t = 4.0$.

the L2 norm relative error of the original PINNs with the same experimental setup is 7.49×10^{-3}, and the results of the comparison are given in Fig. 12. Also analyzing the computational efficiency, the time required for the improved PINNs to complete the training is only 59.3% of the training time of the original PINNs. In this study, we have also taken different

numbers of neural network layers, neurons, and collocation points as variables, and analyze the effects of these variables on the prediction solution of second-order rogue waves for the nonlinear Schrödinger equation. In the specific experiments, we select different numbers of neural network layers, neurons, and collocation points for numerical experiments and compare the prediction solution errors of the original PINNs and the improved PINNs. The specific experimental results are shown in Table 3. Analyzing the experimental results, we can get the conclusion consistent with the previous experiments that with the increase of the number of neural network layers, the number of neurons, and the number of collocation points, the prediction solution error is also reduced. The experiment again demonstrates that for the second-order rogue wave prediction problem of the nonlinear Schrödinger equation, the improved PINNs outperform the original PINNs in terms of both computational efficiency and prediction accuracy.

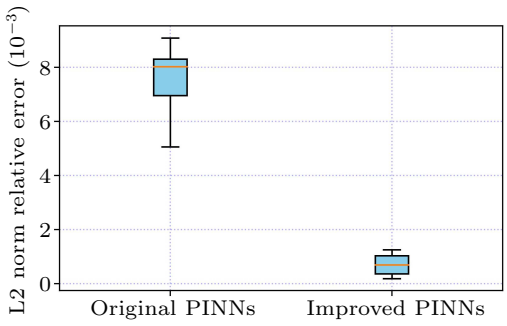


Fig. 12. Visual analysis of errors in the prediction results of the second-order rogue wave solution of the nonlinear Schrödinger equation: comparison of the errors of the improved PINNs and the original PINNs.

4. Conclusion

We have employed gradient-enhanced physics-informed neural networks (PINNs) with meta-learned optimization techniques for conducting numerical simulations of solitary waves and rogue waves. The enhanced PINNs incorporate gradient information, while the meta-learned optimization techniques significantly accelerate the training process of PINNs. As a result, the improved PINN model effectively harnesses more comprehensive physical insights and showcases superior generalization capabilities. Our experimental endeavors encompass the prediction of soliton solutions for the KK equation and rogue wave solution of the nonlinear Schrödinger equation using these enhanced PINNs, followed by a comprehensive evaluation of the predictive outcomes. Notably, our experimental results clearly show that the training speed of these improved PINNs is greatly improved, while the prediction error is significantly reduced. This overall enhancement substantially elevates the accuracy and efficiency of solitary wave and rogue wave simulations. Given the enhanced capacity of PINNs to simulate solitary wave and rogue wave phenomena, our future research endeavors are aimed at extending their applicability to other intricate system simulations. These include ocean mesoscale eddy simulations, typhoon simulations, and atmospheric turbulence simulations, among others. Furthermore, we will explore the integration of PINNs with pre-trained language models to further investigate the potential of its application in the field of complex physics simulation and to extend its application to physics problems at various scales.

Acknowledgment

Project supported by the National Natural Science Foundation of China (Grant Nos. 42005003 and 41475094).

References

[1] Guo Y, Cao X, Liu B and Peng K 2020 *Symmetry* **12** 893
[2] Li S, Song W, Fang L, Chen Y, Ghamisi P and Benediktsson J A 2019 *IEEE Trans. Geosci. Remote Sens.* **57** 6690
[3] Guo Y, Cao X, Liu B and Gao M 2020 *Symmetry* **12** 1056
[4] Bourilkov D 2019 *Int. J. Mod. Phys. A* **34** 1930019
[5] Choudhary K, DeCost B, Chen C, *et al.* 2022 *NPJ Comput. Mater.* **8** 59
[6] Davenport F V and Diffenbaugh N S 2021 *Geophys. Res. Lett.* **48** e2021GL093787
[7] Bryant P, Pozzati G and Elofsson A 2022 *Nat. Commun.* **13** 1
[8] Boussioux L, Zeng C, Guénais T and Bertsimas D 2022 *Weather Forecast.* **37** 817
[9] Dawood M, Asif A and Minhas F U A A 2020 *Neural Comput. Appl.* **32** 9009
[10] Guo Y, Cao X, Liu B and Gao M 2020 *Appl. Sci.* **10** 5917
[11] Sirignano J and Spiliopoulos K 2018 *J. Comput. Phys.* **375** 1339
[12] Guo Y, Cao X and Peng K 2023 *Commun. Theor. Phys.* **75** 095003
[13] van der Meer R, Oosterlee C W and Borovykh A 2022 *J. Comput. Appl. Math.* **405** 113887
[14] Raissi M, Perdikaris P and Karniadakis G E 2019 *J. Comput. Phys.* **378** 686
[15] Haruzi P and Moreno Z 2023 *Water Resour. Res.* **59** e2023WR034538
[16] Guo Y, Cao X and Peng K 2023 *International Joint Conference on Neural Networks (IJCNN)* pp. 1–10
[17] Guo Y, Cao X, Peng K, Tian W and Zhou M 2023 *International Conference on Intelligent Computing* pp. 235–246
[18] Manavi S, Becker T and Fattahi E 2023 *Int. Commun. Heat Mass Transfer* **142** 106662
[19] Miao Z and Chen Y 2023 *Physica D* **456** 133945
[20] Guo Y, Cao X, Song J, Leng H and Peng K 2023 *Phys. Fluids* **35** 106603
[21] Klymak J M and Moum J N 2003 *Geophys. Res. Lett.* **30**
[22] Wazwaz A M 2010 *Partial differential equations and solitary waves theory* (Berlin: Springer)
[23] Dysthe K, Krogstad H E and Müller P 2008 *Annu. Rev. Fluid Mech.* **40** 287
[24] Müller P, Garrett C and Osborne A 2005 *Oceanography* **18** 66
[25] Petviashvili V I and Pohkotelov O A 2016 *Solitary Waves in Plasmas and in the Atmosphere* (London: Routledge)
[26] Peng W Q and Chen Y 2022 *Physica D* **435** 133274
[27] Lin S and Chen Y 2022 *J. Comput. Phys.* **457** 111053
[28] Li J and Li B 2022 *Chaos Solitons Fractals* **164** 112712
[29] Tian S, Niu Z and Li B 2023 *Nonlinear Dyn.* **111** 16467
[30] Yu J, Lu L, Meng X and Karniadakis G E 2022 *Comput. Methods Appl. Mech. Eng.* **393** 114823
[31] Tauseef Mohyud-Din S, Yıldırım A and Sarıaydın S 2011 *Int. J. Numer. Methods Heat Fluid Flow* **21** 272
[32] Parker A 2000 *Physica D* **137** 25
[33] Fibich G 2015 *The nonlinear Schrödinger equation* Vol. 192 (Berlin: Springer)
[34] Zhang Z, Yang X, Li B, Wazwaz A M and Guo Q 2022 *Phys. Lett. A* **450** 128395
[35] Malomed B 2005 *Nonlinear Schrödinger equations*