Journal of Computational Physics 457 (2022) 111053

Contents lists available at ScienceDirect

Journal of Computational Physics

www.elsevier.com/locate/jcp

A two-stage physics-informed neural network method based on conserved quantities and applications in localized wave solutions ☆

Shuning Lin [a], Yong Chen [a,b,*]

[a] School of Mathematical Sciences, Shanghai Key Laboratory of Pure Mathematics and Mathematical Practice, East China Normal University, Shanghai, 200241, China
[b] College of Mathematics and Systems Science, Shandong University of Science and Technology, Qingdao, 266590, China

ARTICLE INFO

Article history:
Received 2 July 2021
Received in revised form 27 November 2021
Accepted 6 February 2022
Available online 14 February 2022

Keywords:
Two-stage PINN
Localized wave solutions
Soliton molecules
Conserved quantities

ABSTRACT

With the advantages of fast calculating speed and high precision, the physics-informed neural network method opens up a new approach for numerically solving nonlinear partial differential equations. Based on conserved quantities, we devise a two-stage PINN method which is tailored to the nature of equations by introducing features of physical systems into neural networks. Its remarkable advantage lies in that it can impose physical constraints from a global perspective. In stage one, the original PINN is applied. In stage two, we additionally introduce the measurement of conserved quantities into mean squared error loss to train neural networks. This two-stage PINN method is utilized to simulate abundant localized wave solutions of integrable equations. We mainly study the Sawada-Kotera equation as well as the coupled equations: the classical Boussinesq-Burgers equations and acquire the data-driven soliton molecule, M-shape double-peak soliton, plateau soliton, interaction solution, etc. Numerical results illustrate that abundant dynamic behaviors of these solutions can be well reproduced and the two-stage PINN method can remarkably improve prediction accuracy and enhance the ability of generalization compared to the original PINN method.

1. Introduction

Relying on the advantages of fast calculating speed and high precision, deep neural networks have developed rapidly and been applied widely in various fields, such as image recognition, speech recognition, natural language processing and so on. The neural network method also plays an important role in the area of scientific computing, especially in solving forward and inverse problems of nonlinear partial differential equations. As a major landmark, Raissi et al. proposed the physics-informed neural network (PINN) method [1], which is one of the most powerful and revolutionary data-driven approaches. It aims to train neural networks to solve supervised learning tasks while respecting laws of physics described by nonlinear partial differential equations. On this basis, abundant significant physics-informed neural network frameworks,

☆ This work is supported by National Natural Science Foundation of China (No. 12175069) and Science and Technology Commission of Shanghai Municipality (No. 21JC1402500 and No. 18dz2271000).

* Corresponding author at: School of Mathematical Sciences, Shanghai Key Laboratory of Pure Mathematics and Mathematical Practice, East China Normal University, Shanghai, 200241, China.
E-mail address: ychen@sei.ecnu.edu.cn (Y. Chen).

https://doi.org/10.1016/j.jcp.2022.111053
0021-9991/

e.g. NSFnets [2], VPINNs [3], fPINNs [4], B-PINNs [5] and hp-VPINNs [6], were devised and targeted at different application situations. This PINN methodology and its variants also have demonstrated extraordinary performance in approximating the unknown solutions [7–9], data-driven discovery of partial differential equations [10,11], the research of extracting physical information from flow visualizations [12] and beyond [13,14]. In order to improve the performance of physics-informed neural networks, Jagtag et al. also proposed different ways of locally adaptive activation functions with slope recovery term and these methods are capable of accelerating the training process [15]. Also noteworthy, many scholars have obtained a number of research results [16–21]. Our group mainly focused on integrable equations possessing remarkable properties, such as the KdV equation, mKdV equation, nonlinear Schrödinger equation, derivative nonlinear Schrödinger equation (DNLS) and Chen-Lee-Liu equation [17–21]. By means of the PINN method, we reproduced abundant dynamic behaviors of data-driven solutions with regard to mentioned equations, including the breathing solution [19], rogue wave solutions [19,20], rogue periodic wave [21] and so on.

Currently, we are devoting to the research of integrable-deep learning algorithms, which aim to study integrable systems via the deep learning algorithm and further improve the neural network method with the advantages of integrable systems. First of all, numerous exact solutions can be obtained since integrable systems have outstanding properties. Therefore, it provides abundant samples for the PINN algorithm in reproducing dynamic behaviors of solutions. Secondly, due to the good properties of integrable systems such as abundant symmetry, infinite conservation laws and the Lax pair, as well as the mature methods for studying integrable systems including the Darboux transformation [22–26], the Bäcklund transformation [27–31], the Hirota bilinear method [32,33] and the inverse scattering transformation [34–37], we can combine these properties and methods with the PINN method to obtain more accurate numerical solutions. Finally, considering that integrable systems can describe physical phenomena such as the localized wave and turbulence [38–40], we can observe more physical phenomena with the aid of PINN method, which can not be obtained by classical methods.

The two-stage physics-informed neural network method based on conserved quantities is proposed here. The specific way is that the original PINN is applied in stage one while in stage two we additionally introduce the measurement of conserved quantities into mean squared error loss to train neural networks. There are three motivations for this improved method. Above all, we intend to further improve integrable-deep learning algorithms and thus more features of physical systems are introduced into neural networks, such as conserved quantities considered in this paper. In the next place, our goal is to devise a more targeted PINN method in solving nonlinear systems, especially integrable systems, which is tailored to the nature of equations by digging out more underlying information of the given equations. Last but not least, we aim to impose constraints from a global perspective considering that the loss functions in the original PINN method reflect the local constrains at certain points solely.

In this paper, we mainly consider nonlinear integrable equations: the Boussinesq-Burgers equations [38,41], the classical Boussinesq-Burgers equations [42–46] as well as the Sawada-Kotera equation [57,58]. Here, our improved PINN method is utilized to reproduce the dynamic behaviors of localized wave solutions for the above equations, such as the interaction solution, soliton molecule, M-shape double-peak soliton, etc.

This paper is organized as follows. In Section 2, we review the physics-informed neural network method for completeness and put forward the two-stage PINN method based on conserved quantities. In Section 3, our two-stage PINN method based on conserved quantities is utilized to simulate abundant localized wave solutions including the one-soliton solution for the Boussinesq-Burgers equations and interaction solution for the classical Boussinesq-Burgers equations. Dynamic behaviors of soliton molecules for the Sawada-Kotera equation are also reproduced in Section 4. In above two sections, given that we just use the original PINN in stage one, the performance of the two models can be evaluated in terms of the accuracy by comparing the results of the two stages. Then we present the relative $\mathbb{L}_2$ errors of these two methods and calculate error reduction rates. Finally, the conclusion and expectation are given in the last section.

2. Methodology

2.1. The PINN method

The physics-informed neural network method is briefly reviewed in this section [1], which plays an important role in solving forward and inverse partial differential equations. We take the following (1+1)-dimensional nonlinear equation as an example to illustrate this method:

$$u_t + \mathcal{N}[u] = 0, x \in [x_0, x_1], t \in [t_0, t_1], \tag{2.1}$$

where $u = u(x,t)$ is the real-valued solution of this equation and $\mathcal{N}[\cdot]$ is a nonlinear differential operator in space. The governing equation $f(x,t)$ is defined by the left-hand-side of the Eq. (2.1) above:

$$f := u_t + \mathcal{N}[u]. \tag{2.2}$$

We aim to solve the initial-boundary value problem with the aid of physics-informed neural network technique. Meanwhile, the PINN method is introduced from three aspects as follows.

(1) Structure establishment of PINN:

Considering that the depth of neural network depends on the number of weighted layers, we construct a neural network of depth L consisting of one input layer, $L-1$ hidden layers and one output layer. The lth $(l=0,1,\cdots,L)$ layer has N_l neurons, which represents that it transmits N_l-dimensional output vector $\mathbf{x}^l$ to the $(l+1)$th layer as the input data. The connection between layers is achieved by the following affine transformation $\mathcal{A}$ and activation function $\sigma(\cdot)$:

$$\mathbf{x}^l = \sigma(\mathcal{A}_l(\mathbf{x}^{l-1})) = \sigma(\mathbf{w}^l\mathbf{x}^{l-1} + \mathbf{b}^l), \tag{2.3}$$

where $\mathbf{w}^l \in \mathbb{R}^{N_l \times N_{l-1}}$ and $\mathbf{b}^l \in \mathbb{R}^{N_l}$ denote the weight matrix and bias vector of the lth layer, respectively. Thus, the relation between input $\mathbf{x}^0$ and output $u(\mathbf{x}^0, \mathbf{\Theta})$ is given by

$$u(\mathbf{x}^0, \mathbf{\Theta}) = (\mathcal{A}_L \circ \sigma \circ \mathcal{A}_{L-1} \circ \cdots \circ \sigma \circ \mathcal{A}_1)(\mathbf{x}^0), \tag{2.4}$$

and here $\mathbf{\Theta} = \left\{\mathbf{w}^l, \mathbf{b}^l\right\}_{l=1}^{L}$ represents the trainable parameters of PINN.

Before training a NN model, we need to initialize the parameters. Usually, the bias term is initialized to zero. There are many effective methods to initialize weight matrixes, such as Xavier initialization [47], He initialization [48], etc. Given that the expression ability of the linear model is not enough, the activation function is used to add nonlinear factors to neural networks. The most frequently used nonlinear activation functions include *ReLU* function, *Sigmoid* function and *tanh* function. In this paper, we select *tanh* function as the activation function and initialize weights of the neural network with the Xavier initialization.

(2) Parameter optimization of PINN:

The essence of the training neural networks or deep learning models is to update the weights and biases. Based on the training data, our goal is to minimize the value of the loss function by optimizing the parameters of the neural network.

Assume we can obtain the initial-boundary dataset $\{x_u^i, t_u^i, u^i\}_{i=1}^{N_u}$ and the set of collocation points of $f(x,t)$, denoted by $\{x_f^i, t_f^i\}_{i=1}^{N_f}$. Then we construct the mean squared error function as the loss function to measure the difference between the predicted values and the true values of each iteration. The given information is investigated to merge into mean squared error, including the initial and boundary data as well as the governing equation:

$$MSE_1 = MSE_u + MSE_f, \tag{2.5}$$

where

$$MSE_u = \frac{1}{N_u}\sum_{i=1}^{N_u} |\widehat{u}(x_u^i, t_u^i) - u^i|^2, \tag{2.6}$$

$$MSE_f = \frac{1}{N_f}\sum_{i=1}^{N_f} |f(x_f^i, t_f^i)|^2. \tag{2.7}$$

Here, $\{\widehat{u}(x_u^i, t_u^i)\}_{i=1}^{N_u}$ denote the predicted results and the derivatives of the network u with respect to time t and space x are derived by automatic differentiation [49] to obtain $\{f(x_f^i, t_f^i)\}_{i=1}^{N_f}$. Based on MSE criteria, the parameters of neural networks are optimized to approach the initial and boundary training data and satisfy the structure imposed by (2.1). Several commonly used optimization methods of loss functions are: L-BFGS [50], SGD, Adam and we apply L-BFGS method here. Hence, numerical solutions of the given domain and period can be obtained according to the trained PINN.

(3) Capability Evaluation of PINN:

Actually, the PINN method only involves above two aspects. However, in this paper, we aim to evaluate the performance of the PINN method in the circumstances of known solutions of Eq. (2.1).

We divide spatial region $[x_0, x_1]$ and time region $[t_0, t_1]$ into N_x and N_t discrete equidistance points, respectively. Then the solution u is discretized into $N_x \times N_t$ data points in the given spatiotemporal domain. We randomly select N_u points of initial-boundary data on the above grids $(\mathcal{I} \cup \mathcal{B}, \mathcal{I} = [x_0 + j\frac{x_1-x_0}{N_x-1}, t_0], (j = 0,1,\cdots,N_x-1), \mathcal{B} = [x, t_0 + k\frac{t_1-t_0}{N_t-1}], (x = x_0 \text{ or } x_1, k = 0,1,\cdots,N_t-1))$ and obtain a random selection of N_f collocation points of $f(x,t)$ in $[x_0,x_1]\times[t_0,t_1]$, which is not required to appear on grids. Thus, the training data in this case is $\{x_u^i, t_u^i, u^i\}_{i=1}^{N_u}$ and $\{x_f^i, t_f^i\}_{i=1}^{N_f}$. Given that the size of training data is only a small percentage of total data on grids, we calculate the relative $\mathbb{L}_2$ error (RE) of $N_x \times N_t$ data points on grids to evaluate the generalization ability of the PINN model:

$$RE = \frac{\sqrt{\sum_{j=0}^{N_x-1}\sum_{k=0}^{N_t-1} |\widehat{u}(x_0 + j\frac{x_1-x_0}{N_x-1}, t_0 + k\frac{t_1-t_0}{N_t-1}) - u^{j,k}|^2}}{\sqrt{\sum_{j=0}^{N_x-1}\sum_{k=0}^{N_t-1} |u^{j,k}|^2}}, \tag{2.8}$$

where $\widehat{u}(x_0 + j\frac{x_1-x_0}{N_x-1}, t_0 + k\frac{t_1-t_0}{N_t-1})$ and $u^{j,k}$ represent the predictive value and true value, separately.

2.2. Introduction of conserved quantities

In this part, the introduction of conserved quantities is presented in brief [51].

For a finite-dimensional system, let $q_i, p_i (i = 1, 2, \cdots, n)$ be the generalized coordinates and momentums of the mechanical system. If Hamiltonian functions $H = H(q_i, p_i)$ exist, which satisfy

$$\frac{dq_i}{dt} = \frac{\partial H}{\partial p_i}, \quad \frac{dp_i}{dt} = -\frac{\partial I}{\partial q_i}, \quad (i = 1, 2, \cdots, n) \tag{2.9}$$

then (2.9) can be rewritten as

$$\begin{aligned} \dot{q}_i &= \{q_i, H\}, \quad \dot{p}_i = \{p_i, H\}, \\ \dot{q}_i &= \tfrac{dq_i}{dt}, \quad \dot{p}_i = \tfrac{dp_i}{dt}, \end{aligned} \tag{2.10}$$

after introducing Poisson brackets

$$\{F, G\} = \sum_{j=1}^{n} \left(\frac{\partial F}{\partial q_j} \frac{\partial G}{\partial p_j} - \frac{\partial F}{\partial p_j} \frac{\partial G}{\partial q_j} \right). \tag{2.11}$$

Besides, q_i and p_i satisfy the following relations

$$\{q_i, q_j\} = \{p_i, p_j\} = 0, \quad \{q_i, p_j\} = \delta_{ij}. \tag{2.12}$$

Therefore, Eq. (2.9) is called the Hamilton system. If there is $I = I(q_i, p_i)$, which holds

$$\frac{dI}{dt} = 0, \tag{2.13}$$

then I is called a conserved quantity of Eq. (2.9).

For infinite dimensional systems, we take the following (1+1)-dimensional nonlinear equation as an example

$$\Delta(x, t, u(x, t)) = 0, \tag{2.14}$$

and then a conserved quantity m_i can be defined similarly, which is time-independent and usually obtained by calculating the integral from $-\infty$ to ∞ with respect to a corresponding conserved density $\Gamma_i(x, t)$:

$$m_i = \int_{-\infty}^{\infty} \Gamma_i \mathrm{dx}, (i = 1, 2, \cdots). \tag{2.15}$$

Then Eq. (2.14) have the corresponding conservation law

$$D_t \Gamma_i + D_x J_i = 0, (i = 1, 2, \cdots) \tag{2.16}$$

which is satisfied for all solutions of (2.14). Here, $\Gamma_i(x, t)$ is the conserved density and $J_i(x, t)$ is the associated flux [52]. The above formulas reveal the relationship between conserved quantities and conservation laws.

Integrable systems have infinite conserved quantities, which is a pretty significant property. Generally speaking, it's not plain to derive conserved quantities. Sometimes, the first few conserved quantities in physical problems usually correspond to the conservation of mass, momentum, or energy. Others may facilitate the research of the quantitative and qualitative properties of solutions.

2.3. The two-stage PINN method based on conserved quantities

The main purpose of this article is to put forward a more targeted PINN algorithm of nonlinear mathematical physics. We try to introduce more features of integrable systems into neural networks to improve the precision and reliability. This part epitomizes the main idea of the two-stage PINN method based on conserved quantities.

(1) Stage One:

In the first stage, we use the original PINN method which is mentioned in Section 2.1. Under the principle of minimizing the mean squared error loss, we can acquire the numerical solution $\widehat{u}_1(x, t)$ of the given domain and period after parameter optimization.

(2) Stage Two:

On the basis of stage one, we make the following improvements. Based on conserved quantities, we aim to achieve further optimization of the numerical solution $\widehat{u}_1(x, t)$ in the first stage.

Firstly, we should gain a conserved quantity $m(t)$ of the corresponding equation, which evidently depends on the choice of the solution u and is actually time-independent, i.e. $\frac{dm(t)}{dt} = 0$. Based on the initial data of $u(x, t)$, the conserved quantity

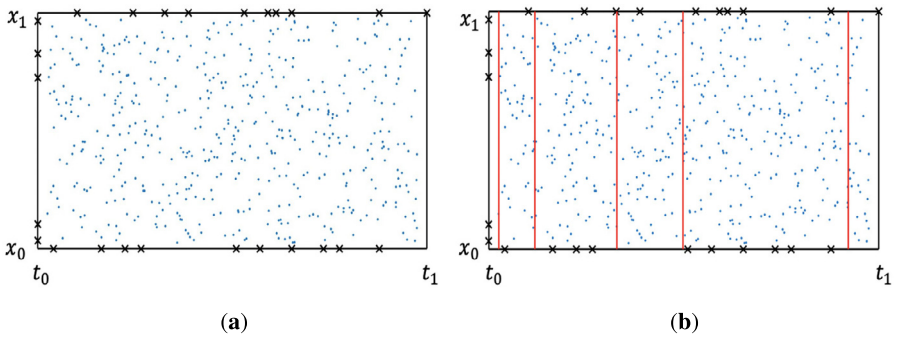

Fig. 1. (Color online.) Schematic diagrams of constraints: **(a)** Constraints of the original PINN method; **(b)** Constraints of the two-stage PINN method based on conserved quantities.

$m(t_0)$ can be calculated and taken as the criterion. We randomly select N_c different moments and measure the corresponding conserved quantities $\{m(t_m^i)\}_{i=1}^{N_c}$. Our goal is to make $\{m(t_m^i)\}_{i=1}^{N_c}$ approach the theoretical value $m(t_0)$ as close as possible.

According to the above analysis, the mean squared error loss of the original PINN is changed into:

$$MSE_2 = MSE_u + MSE_f + MSE_s + MSE_m, \tag{2.17}$$

where

$$MSE_s = \frac{1}{N_s}\sum_{i=1}^{N_s} |\widehat{u}(x_s^i, t_s^i) - \widehat{u}_1(x_s^i, t_s^i)|^2, \tag{2.18}$$

$$MSE_m = \frac{1}{N_c}\sum_{i=1}^{N_c} |m(t_m^i) - m(t_0)|^2. \tag{2.19}$$

Here, $\widehat{u}(x,t)$ denotes the numerical solution obtained in stage two and MSE_s measures the difference of numerical results between two stages at $\{x_s^i, t_s^i\}_{i=1}^{N_s}$, which implies that further optimization is based on stage one and N_s points $\{x_s^i, t_s^i, \widehat{u}_1(x_s^i, t_s^i), \widehat{u}(x_s^i, t_s^i)\}_{i=1}^{N_s}$ are sampled randomly on the grids. Meanwhile, MSE_m reflects the constraint of the conserved quantity.

With regard to the calculation of conserved quantities, we adopt the method of numerical integral by using summation instead of integrals. Suppose $\mathcal{M}[u]$ is a conserved density ($\mathcal{M}[\cdot]$ denotes a differential operator) and we divide spatial region $[x_0, x_1]$ into N_x discrete equidistance points with time region $[t_0, t_1]$ into N_t discrete equidistance points, then $\mathcal{M}[u]$ is discretized into $N_x \times N_t$ data points and the formulas of $m(t_0)$ and $m(t_m^i)$ are derived:

$$m(t_0) = \int_{x_0}^{x_1} \mathcal{M}[u](x, t_0)\mathrm{dx} \approx \sum_{j=2}^{N_x} \mathcal{M}[u](x^j, t_0)\frac{x_1 - x_0}{N_x - 1}, \tag{2.20}$$

$$m(t_m^i) = \int_{x_0}^{x_1} \mathcal{M}[\widehat{u}](x, t_m^i)\mathrm{dx} \approx \sum_{j=2}^{N_x} \mathcal{M}[\widehat{u}](x^j, t_m^i)\frac{x_1 - x_0}{N_x - 1}, \tag{2.21}$$

where $\mathcal{M}[u](x^j, t_0)$ and $\mathcal{M}[\widehat{u}](x^j, t_m^i)$ represent the true value and predictive value, respectively.

In the original PINN method, the loss MSE_u and MSE_f reflect the local constraints at certain points solely, which are selected stochastically. However, in stage two, the calculation of conserved quantities involves the integral operation. It is widely known that at any given time, conserved quantities mirror the global property of the solution u in $[x_0, x_1]$. Thus, our practice to introduce the measurement of this global property into the mean squared error loss is meaningful and is a kind of method to impose constraints from a global perspective.

Similarly, if we consider k conserved quantities $\boldsymbol{m} = (m_1, m_2, \cdots, m_k)$, MSE_m is transformed into:

$$MSE_{\boldsymbol{m}} = \frac{1}{N_c}\sum_{j=1}^{k}\sum_{i=1}^{N_c} |m_j(t_m^i) - m_j(t_0)|^2. \tag{2.22}$$

With regard to two methods above, we display the schematic diagrams of constraints in Fig. 1. Black crosses imply that these selected points need to meet initial-boundary conditions and blue dots represent the random selection of points

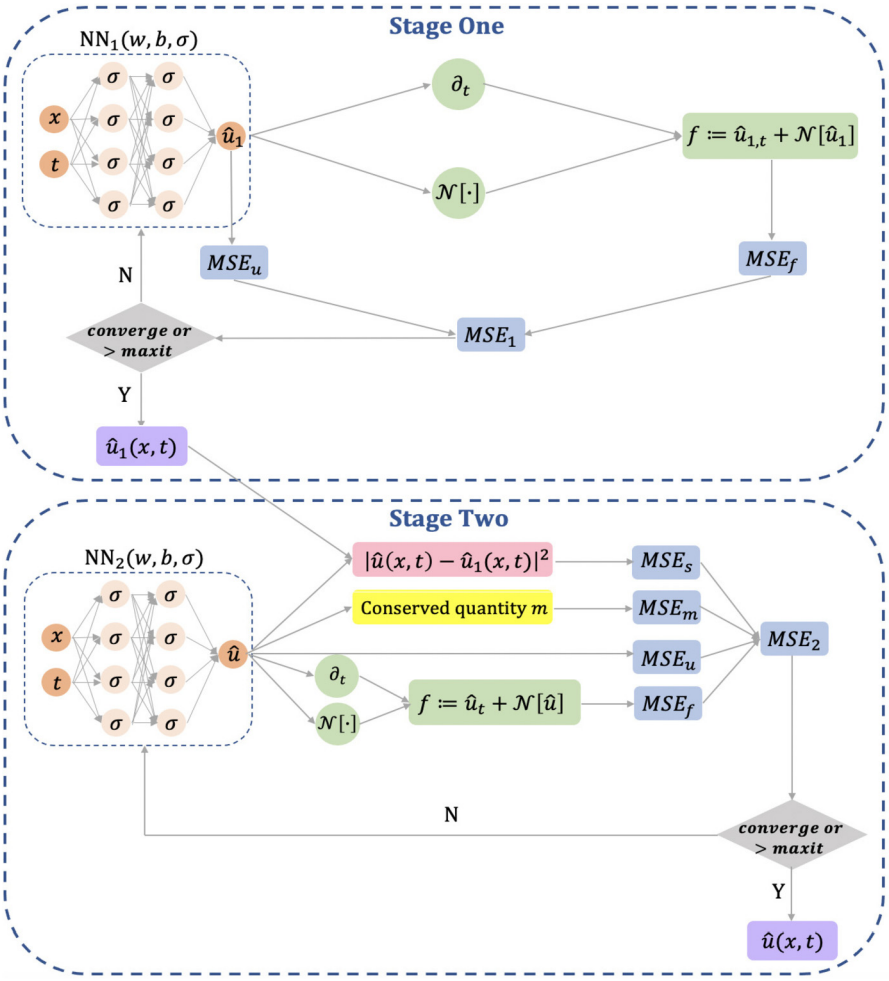

Fig. 2. (Color online.) Schematic diagram of the two-stage physics-informed neural network method based on conserved quantities.

should satisfy the structure imposed by the governing equation. They are both local constraints. The two-stage PINN method based on conserved quantities differs from the original PINN method in that it takes conserved quantities into consideration to impose constraints globally. We use the red lines to denote calculation of conserved quantity at certain moments, which involves the integral operation and the method of numerical integral is adopted by using summation instead of integrals.

Moreover, Fig. 2 shows a sketch of the two-stage PINN method based on conserved quantities, where *maxit* denotes the maximum number of iterations.

This method is established on the assumption that heights of background waves in the region $\mathcal{D}$ ($\mathcal{D} = [x,t], x \in (-\infty, x_0]\bigcup[x_1, +\infty), t \in [t_0, t_1]$) are almost consistent. Therefore, we can integral from x_0 to x_1 with respect to a conserved density to represent conserved quantities instead of from $-\infty$ to $+\infty$. It is a reasonable assumption in the sense that it can be easily satisfied by localized wave solutions, which are widely considered in the field of integrable systems to describe various physical phenomena.

Actually, a natural idea is to take the following formula

$$MSE = MSE_u + MSE_f + MSE_m, \tag{2.23}$$

as the optimization objective of PINN directly rather than carrying out this two-stage method. However, the result was a disappointment and even worse than the original PINN method. After the analysis, we are of the opinion that the optimization is dominated by MSE_m, and finally it converges to other local optimal point which causes unsatisfactory results. Consequently, we propose the two-stage PINN method to improve it. This method not only considers the global property measured by conserved quantities, but also further optimizes parameters of the original PINN. The numerical results also show that it can avoid converging to other non-ideal local optimums.

All codes are based upon Python 3.7 and Tensorflow 1.15, and the presented numerical experiments are run on a MacBook Pro computer with 2.3 GHz Intel Core i5 processor and 16-GB memory.

3. Data-driven one-soliton solution of the Boussinesq-Burgers equations and interaction solution of the classical Boussinesq-Burgers equations

In this section, we will apply the two-stage PINN method to numerically solve integrable equations and then contrast the simulation results of the two models: the PINN and two-stage PINN based on conserved quantities. Considering that we just use the original PINN in stage one, the performance of the two models can be evaluated in terms of the accuracy by comparing the results of the two stages.

The current research of coupled equations with the aid of neural networks is relatively less than that of the single equation and thus we mainly consider the coupled equations here: the Boussinesq-Burgers equations [38,41] and the classical Boussinesq-Burgers equations [42–46].

3.1. One-soliton solution of the Boussinesq-Burgers equations

Here, we investigate the Boussinesq-Burgers equations [38,41] with the first kind of boundary condition (Dirichlet boundary condition)

$$\begin{cases} u_t + 2uu_x - \frac{1}{2}v_x = 0, \\ v_t + 2(uv)_x - \frac{1}{2}u_{xxx} = 0, x \in [x_0, x_1], t \in [t_0, t_1], \\ u(x, t_0) = u_0(x), \\ v(x, t_0) = v_0(x), \\ u(x_0, t) = a_1(t), u(x_1, t) = a_2(t), \\ v(x_0, t) = a_3(t), v(x_1, t) = a_4(t). \end{cases} \tag{3.1}$$

Wang et al. [53] studied the Lax pair, Bäcklund transformation and multi-soliton solutions for the Boussinesq-Burgers equations. Many researchers also obtained a variety of soliton solutions and some exact interaction solutions of the Boussinesq-Burgers equations, which describe the propagation of shallow water waves [55,56]. In Ref. [62], Rady et al. have derived the multi-soliton solution of this equation. Firstly, they consider the following function transformation

$$v = \lambda u_x + \beta, \tag{3.2}$$

and set $\lambda = -1, \beta = 0$. In the light of the idea of homogeneous balance method [70] as well as the Bäcklund transformation, the multi-soliton solution can be obtained

$$\begin{aligned} u &= \frac{1}{2}\frac{\sum_{i=1}^{n} k_i \exp\left(k_i\left(x - 2\left(a + \frac{k_i}{4}\right)t\right)\right)}{1 + \sum_{i=1}^{n} \exp\left(k_i\left(x - 2\left(a + \frac{k_i}{4}\right)t\right)\right)} + a, \\ v &= -u_x. \end{aligned} \tag{3.3}$$

In this case, the governing equations $f_1(x, t)$ and $f_2(x, t)$ are as follows

$$\begin{aligned} f_1 &:= u_t + 2uu_x - \frac{1}{2}v_x, \\ f_2 &:= v_t + 2(uv)_x - \frac{1}{2}u_{xxx}. \end{aligned} \tag{3.4}$$

When $n = 1$, the corresponding initial-boundary conditions are given by

$$\begin{aligned} &u(-20, t) = -\frac{e^{20+\frac{7t}{2}}}{2(1 + e^{20+\frac{7t}{2}})} + 2, u(20, t) = -\frac{e^{-20+\frac{7t}{2}}}{2(1 + e^{-20+\frac{7t}{2}})} + 2, \\ &v(-20, t) = -\frac{\mathrm{e}^{20+\frac{7t}{2}}}{2\left(1 + \mathrm{e}^{20+\frac{7t}{2}}\right)} + \frac{\left(\mathrm{e}^{20+\frac{7t}{2}}\right)^2}{2\left(1 + \mathrm{e}^{20+\frac{7t}{2}}\right)^2}, \\ &v(20, t) = -\frac{\mathrm{e}^{-20+\frac{7t}{2}}}{2\left(1 + \mathrm{e}^{-20+\frac{7t}{2}}\right)} + \frac{\left(\mathrm{e}^{-20+\frac{7t}{2}}\right)^2}{2\left(1 + \mathrm{e}^{-20+\frac{7t}{2}}\right)^2}, \\ &u_0(x) = -\frac{e^{-x-7}}{2(1 + e^{-x-7})} + 2, v_0(x) = -\frac{e^{-x-7}}{2(1 + e^{-x-7})} + \frac{(e^{-x-7})^2}{2(1 + e^{-x-7})^2}, \end{aligned} \tag{3.5}$$

after choosing corresponding parameters as $a = 2, k_1 = -1, [x_0, x_1] = [-20, 20], [t_0, t_1] = [-2, 2]$. To obtain the training data, we divide the spatial region $[x_0, x_1] = [-20, 20]$ and time region $[t_0, t_1] = [-2, 2]$ into $N_x = 1025$ and $N_t = 201$ discrete

Table 1
One-soliton solution of the Boussinesq-Burgers equations: relative $\mathbb{L}_2$ errors of PINN and two-stage PINN based on conserved quantities as well as error reduction rates.

Solution \ Method	PINN	Two-stage PINN	Error reduction rate
u	8.965473e-04	7.343612e-04	18.09%
v	4.750580e-02	3.776971e-02	20.49%

equidistance points, separately. Thus, the solutions u and v are both discretized into 1025×201 data points in the given spatiotemporal domain. We randomly select $N_u = 100$ points from the initial-boundary dataset and proceed by sampling $N_f = 10000$ collocation points via the Latin hypercube sampling method [69]. A 8-layer feedforward neural network with 40 neurons per hidden layer is constructed to learn the one-soliton solution of the Boussinesq-Burgers equations. In addition, we use the hyperbolic tangent ($tanh$) activation function and initialize weights of the neural network with the Xavier initialization. The derivatives of the network u, v with respect to time t and space x are derived by automatic differentiation.

We utilize the L-BFGS algorithm to optimize loss functions. Obviously, v is a conserved density of the Boussinesq-Burgers equations and we select m defined by

$$m = \int_{x_0}^{x_1} v \mathrm{d}x \approx \sum_{j=2}^{N_x} v(x^j, t) \frac{x_1 - x_0}{N_x - 1}, \tag{3.6}$$

as the conserved quantity adopted in two-stage PINN. The loss function of stage one is (2.5) and that of stage two is (2.17) where we choose $N_s = 10000, N_c = 20$ and MSE_m is given by

$$\begin{aligned} MSE_m &= \frac{1}{N_c} \sum_{i=1}^{N_c} |m(t_m^i) - m(t_0)|^2 \\ &\approx \frac{1}{N_c} \sum_{i=1}^{N_c} \left| \sum_{j=2}^{N_x} \widehat{v}(x^j, t_m^i) \frac{x_1 - x_0}{N_x - 1} - \sum_{j=2}^{N_x} v(x^j, t_0) \frac{x_1 - x_0}{N_x - 1} \right|^2, \end{aligned} \tag{3.7}$$

where $v(x^j, t_0)$ and $\widehat{v}(x^j, t_m^i)$ represent the true value and predictive value, respectively.

Ultimately, the data-driven one-soliton solution of the Boussinesq-Burgers equations is obtained by two-stage PINN method based on conserved quantities.

Fig. 3 displays the density diagrams of the one-soliton solution, comparison between the predicted solutions and exact solutions as well as the error density diagrams. In the bottom panel of Fig. 3 (a) and Fig. 3 (c), we show the comparison between exact solutions and predicted solutions at different time points $t = -1.5, 0, 1.5$. Obviously, both u and v propagate along the positive direction of the x-axis as time goes by. Through contrastive analysis, one-soliton solution can be successfully simulated by two-stage PINN method with high accuracy. In Fig. 4, the three-dimensional plots of predicted one-soliton solutions $u(x, t)$ and $v(x, t)$ are showed respectively, where $v(x, t)$ is a dark soliton solution.

In stage one, the original PINN is applied. After 168 times iterations in about 41.7379 seconds, the relative $\mathbb{L}_2$ error of u is 8.965473e-04 and that of v is 4.750580e-02. In stage two, where the conserved quantity is considered, the relative $\mathbb{L}_2$ error of u is 7.343612e-04 and that of v is 3.776971e-02 after 1382 times iterations in about 407.5796 seconds. To compare the performance of two methods, error reduction rate (ERR) can be obtained according to the relative $\mathbb{L}_2$ error of PINN method (RE_1) and that of two-stage PINN method based on conserved quantities (RE_2):

$$ERR = \frac{RE_1 - RE_2}{RE_1}. \tag{3.8}$$

By calculation, the error reduction rate (ERR) of u is 18.09% and that of v is 20.49%, which are presented in Table 1. It turns out that our proposed two-stage PINN method based on conserved quantities can improve prediction accuracy and gain better generalization.

3.2. Interaction solution of the classical Boussinesq-Burgers equations

In this part, we consider the classical Boussinesq-Burgers (CBB) equations [42–46]

$$\begin{aligned} u_t &= \frac{1}{2}(\beta - 1)u_{xx} + 2uu_x + \frac{1}{2}v_x, \\ v_t &= \beta\left(1 - \frac{\beta}{2}\right)u_{xxx} + \frac{1}{2}(1 - \beta)v_{xx} + 2(uv)_x, \end{aligned} \tag{3.9}$$

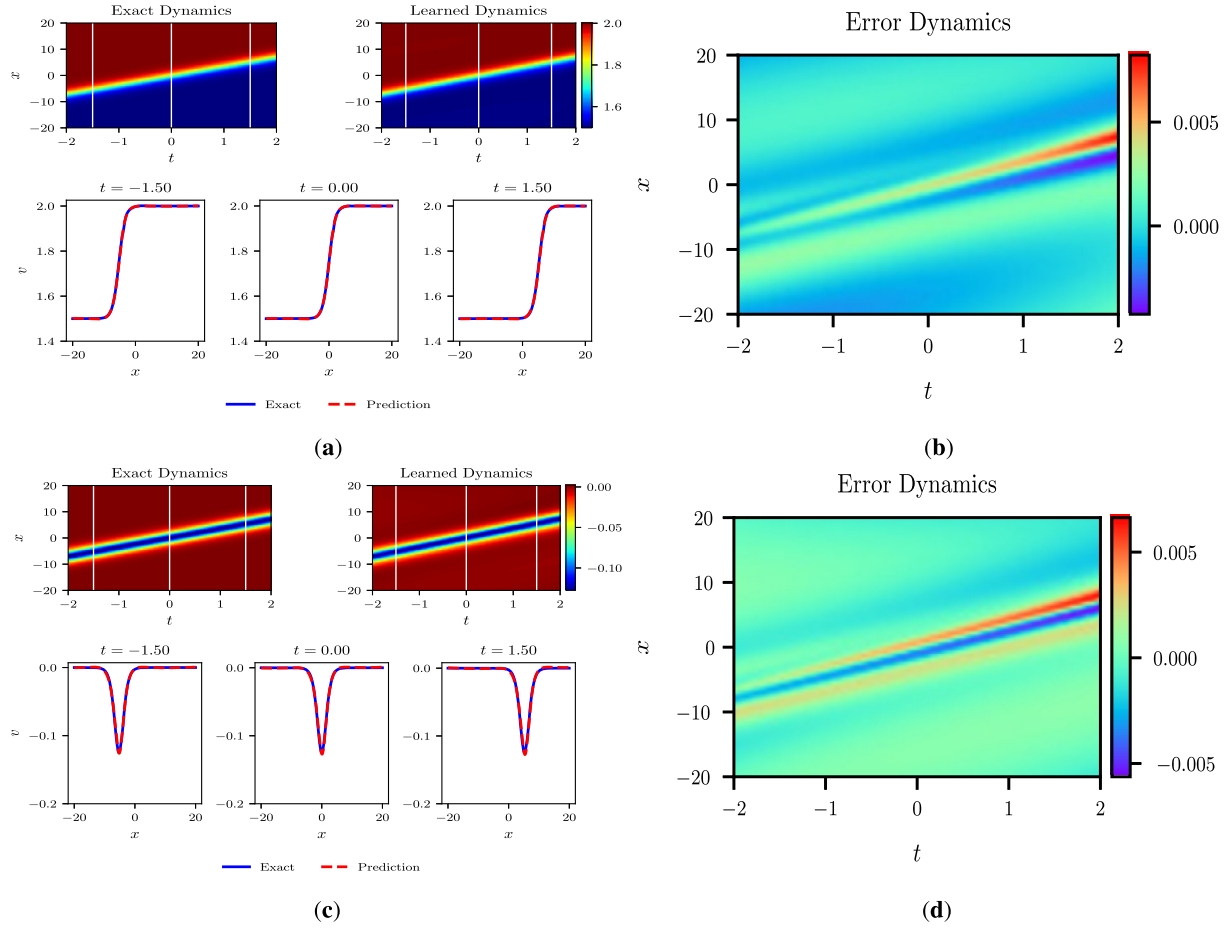

Fig. 3. (Color online.) One-soliton solution $u(x,t)$ and $v(x,t)$ of the Boussinesq-Burgers equations by two-stage PINN based on conserved quantities: **(a)** The density diagrams and comparison between the predicted solutions and exact solutions at the three temporal snapshots of $u(x,t)$; **(b)** The error density diagram of $u(x,t)$; **(c)** The density diagrams and comparison between the predicted solutions and exact solutions at the three temporal snapshots of $v(x,t)$; **(d)** The error density diagram of $v(x,t)$.

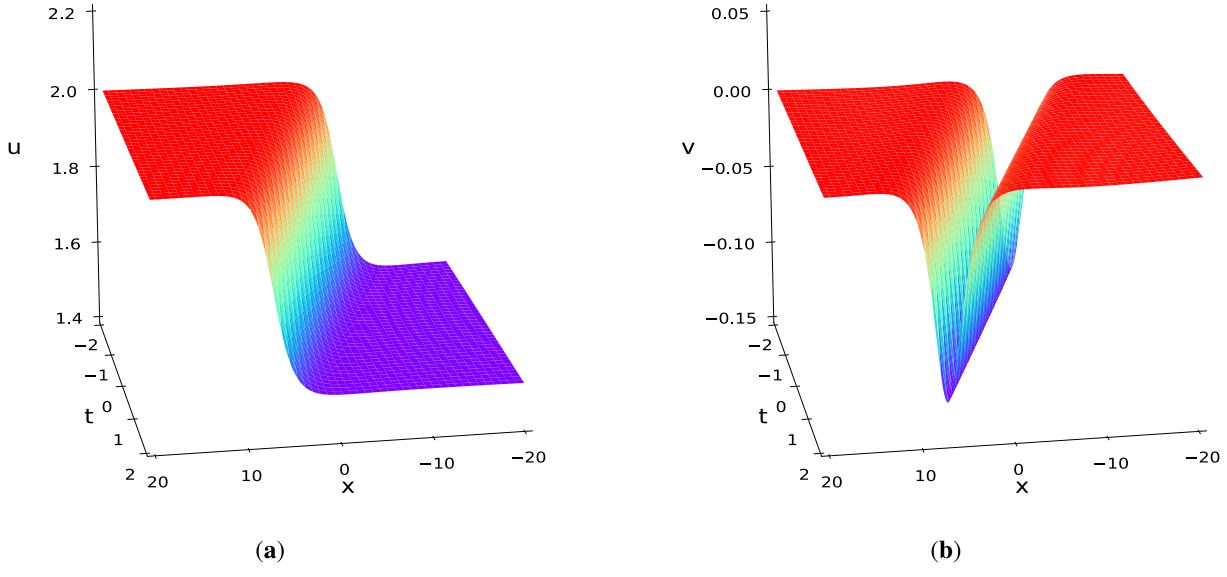

Fig. 4. (Color online.) One-soliton solution $u(x,t)$ and $v(x,t)$ of the Boussinesq-Burgers equations by two-stage PINN based on conserved quantities: **(a)** The three-dimensional plot of $u(x,t)$; **(b)** The three-dimensional plot of $v(x,t)$.

where $u = u(x,t)$ and $v = v(x,t)$ are real-valued solutions and β is an arbitrary constant. Obviously, the classical Boussinesq-Burgers equations are equivalent to the Boussinesq-Burgers equations under the condition $(u, v, x, t, \beta) \to (-u, -v, -x, -t, 1)$. Moreover, Darboux transformations and soliton solutions of the classical Boussinesq-Burgers equations have been given in Ref. [54]. Some scholars also have studied the finite-band solutions [46], rational solutions [71], conservation laws and dynamical behaviors [72]. Dong et al. [63] applied the consistent tanh expansion (CTE) to study the interaction solution for this equation given by

$$\begin{aligned} &u = u_0 + u_1 \tanh(w), \\ &v = v_0 + v_1 \tanh(w) + v_2 \tanh(w)^2, \end{aligned} \tag{3.10}$$

where

$$\begin{aligned} &u_1 = \frac{w_x}{2}, \quad u_0 = \frac{2w_t - w_{xx}}{4w_x}, \\ &v_2 = \frac{\beta w_x^2}{2} - w_x^2, \quad v_1 = w_{xx} - \frac{\beta w_{xx}}{2}, \\ &v_0 = -\frac{(\beta-2)(2w_x^4 - w_x w_{xxx} + 2w_x w_{xt} + w_{xx}^2 - 2w_{xx} w_t)}{4w_x^2}, \end{aligned} \tag{3.11}$$

and the interaction between soliton and resonance has the following form

$$\begin{aligned} &w = px + qt + \frac{1}{2}\ln\left(1 + \sum_{i=1}^{n} \exp(p_i x + q_i t)\right), i = 1, 2, \ldots \\ &q_i = \frac{p_i\left(2q + p_i p + 2p^2\right)}{2p}, \quad i = 1, 2, \ldots \end{aligned} \tag{3.12}$$

Here, we select the parameters as follows:

$$n = 1, \quad p = 1, \quad q = 1, \quad \beta = 1, \quad p_1 = 2. \tag{3.13}$$

We focus on the classical Boussinesq-Burgers (CBB) equations with the first kind of boundary condition (Dirichlet boundary condition)

$$\begin{cases} u_t = 2uu_x + \frac{1}{2}v_x, \\ v_t = \frac{1}{2}u_{xxx} + 2(uv)_x, x \in [x_0, x_1], t \in [t_0, t_1], \\ u(x, t_0) = u_0(x), \\ v(x, t_0) = v_0(x), \\ u(x_0, t) = a_1(t), u(x_1, t) = a_2(t), \\ v(x_0, t) = a_3(t), v(x_1, t) = a_4(t). \end{cases} \tag{3.14}$$

After setting $[x_0, x_1] = [-10, 15]$, $[t_0, t_1] = [-3, 2]$, initial conditions of the interaction solution above are obtained as follows

$$\begin{aligned} u_0(x) &= \frac{(4e^{-36+4x} + 4e^{-18+2x} + 1)(\tanh(x - 3 + \frac{\ln(1+e^{-18+2x})}{2}) + 1)}{4(e^{-18+2x})^2 + 6e^{-18+2x} + 2}, \\ v_0(x) &= \frac{2e^{-18+2x}\sinh(x - 3 + \frac{\ln(1+e^{-18+2x})}{2})\cosh(x - 3 + \frac{\ln(1+e^{-18+2x})}{2})}{2(1 + e^{-18+2x})^2\cosh^2(x - 3 + \frac{\ln(1+e^{-18+2x})}{2})} \\ &\quad + \frac{2e^{-18+2x}\cosh^2(x - 3 + \frac{\ln(1+e^{-18+2x})}{2}) + 4e^{-36+4x} + 4e^{-18+2x} + 1}{2(1 + e^{-18+2x})^2\cosh^2(x - 3 + \frac{\ln(1+e^{-18+2x})}{2})}, \end{aligned} \tag{3.15}$$

as well as corresponding boundary conditions, which are no longer presented here due to space limitation.

We construct a 9-layer feedforward neural network with 40 neurons per hidden layer to learn the interaction solution between a soliton and one resonant. With the help of MATLAB, spatial region $[x_0, x_1] = [-10, 15]$ and time region $[t_0, t_1] = [-3, 2]$ are divided into $N_x = 1025$ and $N_t = 201$ discrete equidistance points, respectively. After adopting the same generation and sampling method of training data in Section 3.1, we randomly select $N_u = 100$ points from the initial-boundary dataset and $N_f = 10000$ collocation points.

Here, v is a conserved density of the classical Boussinesq-Burgers equations and the conserved quantity adopted in two-stage PINN is defined just as (3.6). The loss function of stage one is (2.5) and that of stage two is (2.17) where we choose

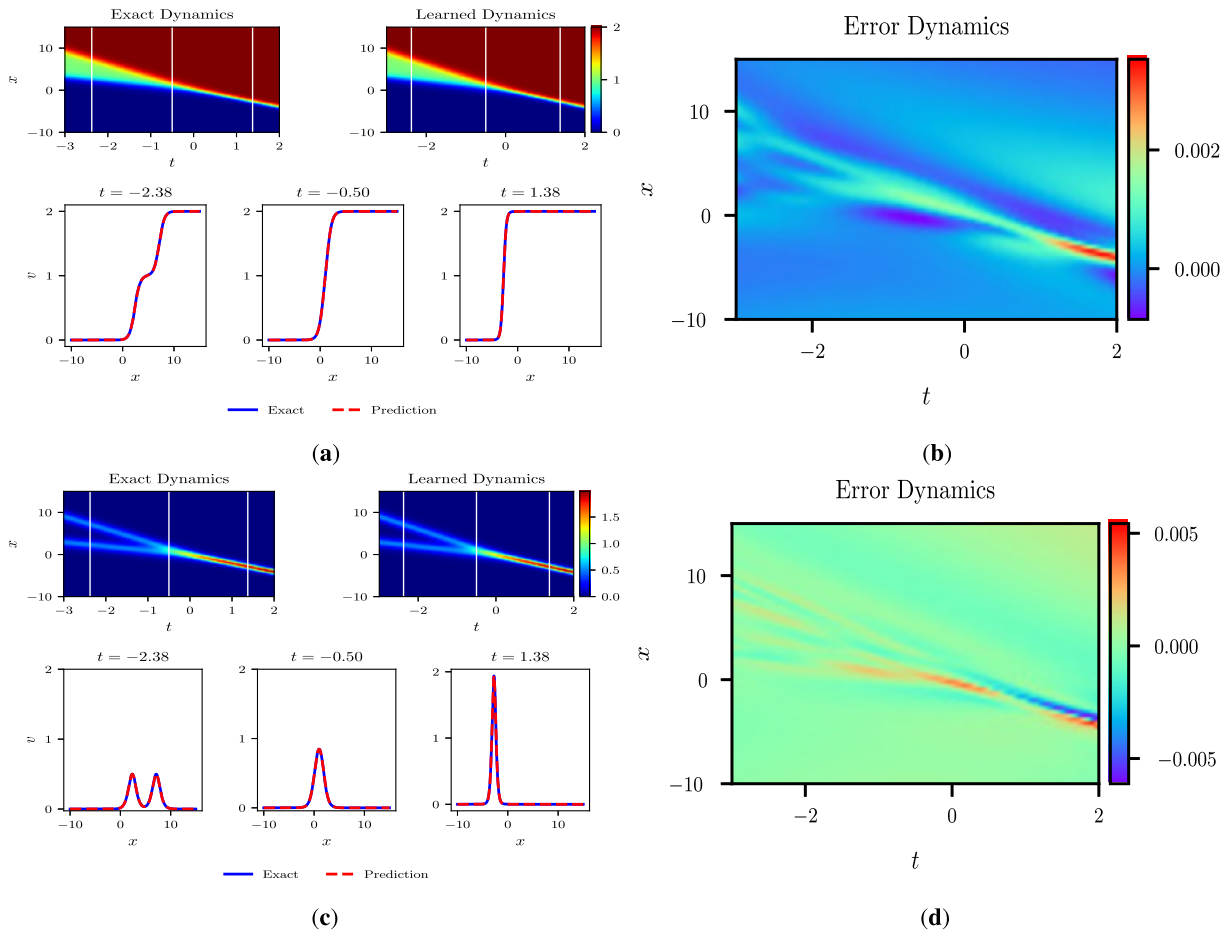

Fig. 5. (Color online.) Interaction solution $u(x,t)$ and $v(x,t)$ between soliton and resonance of the classical Boussinesq-Burgers equations by two-stage PINN based on conserved quantities: **(a)** The density diagrams and comparison between the predicted solutions and exact solutions at the three temporal snapshots of $u(x,t)$; **(b)** The error density diagram of $u(x,t)$; **(c)** The density diagrams and comparison between the predicted solutions and exact solutions at the three temporal snapshots of $v(x,t)$; **(d)** The error density diagram of $v(x,t)$.

$N_s = 10000, N_c = 20$ and the computing formula of MSE_m is consistent with (3.7). In addition, the L-BFGS algorithm to optimize loss functions is the same in Section 3.1, as well as the Xavier initialization and the hyperbolic tangent ($tanh$) activation function.

The two-stage PINN method based on conserved quantities eventually succeeds in numerical simulations of the interaction solution between a soliton and one resonant.

In Fig. 5, the density diagrams of interaction solution, comparison between the predicted solutions and exact solutions as well as the error density diagrams are plotted. From the bottom panel of Fig. 5 (a) and Fig. 5 (c), it implies there is little difference between exact solutions and predicted solutions. Meanwhile, it can be seen that two peaks converge into one of higher amplitude according to the wave propagation pattern of $v(x,t)$ and it propagates along the negative direction of the x-axis. Fig. 6 displays the predicted 3D plots of the interaction solution, which show interaction behaviors between soliton and resonance.

In stage one, we use the original PINN method. After 784 times iterations in about 267.7466 seconds, the relative $\mathbb{L}_2$ error of u is 3.536702e-04 and that of v is 3.304951e-03. In stage two, where the conserved quantity is considered, the relative $\mathbb{L}_2$ error of u is 2.756669e-04 and that of v is 2.576679e-03 after 6411 times iterations in about 2421.5709 seconds. To compare the performance of two methods, the results of calculation show that the error reduction rate (ERR) of u is 22.06% and that of v is 22.04%, which are shown in Table 2. Compared with the original PINN, we also confirm that the precision and the generalization ability of neural networks can be improved by our two-stage PINN method based on conserved quantities.

4. Data-driven soliton molecule and new types of solitons of the Sawada-Kotera equation

In recent years, soliton molecules, bound states of solitons, have been widely concerned. A pair of bright solitons, bound together by a dark soliton were discovered in optical fibers through numerical simulations and experimental verifications

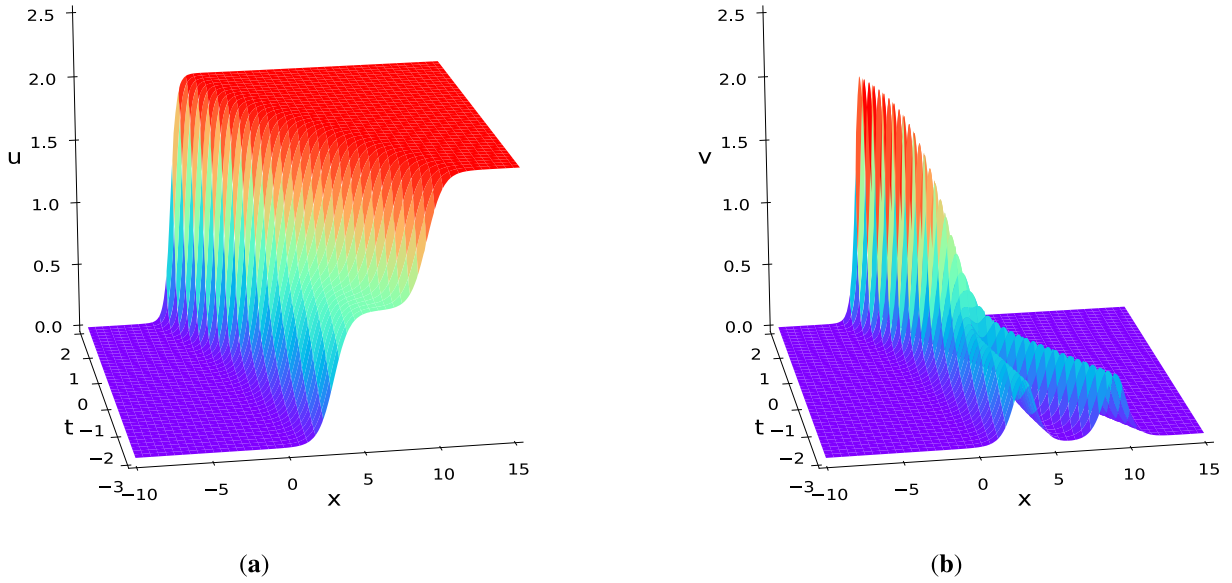

Fig. 6. (Color online.) Interaction solution $u(x,t)$ and $v(x,t)$ between soliton and resonance of the classical Boussinesq-Burgers equations by two-stage PINN based on conserved quantities: **(a)** The three-dimensional plot of $u(x,t)$; **(b)** The three-dimensional plot of $v(x,t)$.

Table 2
Interaction solution of the classical Boussinesq-Burgers equations: relative $\mathbb{L}_2$ errors of PINN and two-stage PINN based on conserved quantities as well as error reduction rates.

Method / Solution	PINN	Two-stage PINN	Error reduction rate
u	3.536702e-04	2.756669e-04	22.06%
v	3.304951e-03	2.576679e-03	22.04%

in 2005 [64]. Later, soliton molecules were obtained in dipolar Bose-Einstein condensates by the method of numerical prediction [65]. Lou [66] used a new mechanism, namely the velocity resonant, to find soliton molecules in three fifth order integrable systems (fifth order KdV, KK and SK equations). Ren et al. [67] studied soliton molecules of the Korteweg-de Vries equation with higher-order corrections via the velocity resonance mechanism and they found the collision between a soliton molecule and one soliton is elastic.

To our knowledge, there is poor study of the data-driven soliton molecules by physics-informed neural networks. Consequently, in this part, abundant travelling wave structures are numerically simulated through the two-stage PINN method based on conserved quantities, like the soliton molecule, kink-antikink molecule and so on.

For the following Sawada-Kotera (SK, also called Caudrey-Dodd-Gibbon-Sawada-Kotera (CDGSK)) equation

$$u_t + u_{xxxxx} + 15uu_{xxx} + 45u^2u_x + 15u_xu_{xx} = 0, \tag{4.1}$$

which was introduced in Ref. [57,58], Ye et al. [59] obtained many new periodic travelling wave solutions via Jacobi elliptic function linear superposition approach. Meanwhile, singular travelling wave solutions of this equation were researched [60]. Lou [61] derived the inverse recursion operator by using the pseudopotential of SK.

In this part, we aim to reproduce dynamic behaviors of the soliton molecule and new types of solitons for Eq. (4.1). Wang et al. [68] obtained the soliton molecule solutions via the travelling wave approach

$$u = -ak^2 + 6ac\frac{c + \cosh\left[\sqrt{3a}k\left(x - 9a^2k^4t - b\right)\right]}{\left[c\cosh\left[\sqrt{3a}k\left(x - 9a^2k^4t - b\right)\right] + 1\right]^2}, \tag{4.2}$$

where $a > 0$, c, k and b are arbitrary constants. Here, we consider the Sawada-Kotera (SK) equation with the first kind of boundary condition (Dirichlet boundary condition) as follows

$$\begin{cases} u_t + u_{xxxxx} + 15uu_{xxx} + 45u^2u_x + 15u_xu_{xx} = 0, x \in [x_0, x_1], t \in [t_0, t_1], \\ u(x, t_0) = u_0(x), \\ u(x_0, t) = a_1(t), \\ u(x_1, t) = a_2(t). \end{cases} \tag{4.3}$$

Obviously, the governing equation $f(x,t)$ is given by

$$f := u_t + u_{xxxxx} + 15uu_{xxx} + 45u^2u_x + 15u_xu_{xx}, \tag{4.4}$$

and u is a conserved density considered in this section.

We only present the following soliton molecule solution and new types of soliton solutions of the Sawada-Kotera equation here:

4.1. The soliton molecule (SM) for $0 < c \ll \frac{1}{2}$

After taking $c = \frac{1}{4000}, k = a = 1, b = 0$, $[x_0, x_1] = [-10, 10], [t_0, t_1] = [-0.01, 0.01]$, we have:

$$\begin{aligned}
u(-10,t) &= -1 + \frac{3\left(\frac{1}{4000} + \cosh(\sqrt{3}(-9t-10))\right)}{2000\left(\frac{\cosh(\sqrt{3}(-9t-10))}{4000}+1\right)^2},\\
u(10,t) &= -1 + \frac{3\left(\frac{1}{4000} + \cosh(\sqrt{3}(-9t+10))\right)}{2000\left(\frac{\cosh(\sqrt{3}(-9t+10))}{4000}+1\right)^2},\\
u_0(x) &= -1 + 3\frac{\frac{1}{4000} + \cosh(\sqrt{3}(0.09+x))}{2000(\frac{1}{4000}\cosh(\sqrt{3}(0.09+x))+1)^2}.
\end{aligned} \tag{4.5}$$

With the aid of MATLAB, we divide spatial region $[x_0, x_1] = [-10, 10]$ into $N_x = 513$ discrete equidistance points and time region $[t_0, t_1] = [-0.01, 0.01]$ into $N_t = 201$ discrete equidistance points. Thus, the solution u in the given spatiotemporal domain is discretized into 513×201 data points. We randomly select $N_u = 100$ points $\{x_u^i, t_u^i, u^i\}_{i=1}^{N_u}$ from the initial-boundary dataset and proceed by sampling $N_f = 2000$ collocation points $\{x_f^i, t_f^i\}_{i=1}^{N_f}$ via the Latin hypercube sampling method. A 8-layer feedforward neural network with 40 neurons per hidden layer is constructed to learn the soliton molecule (SM) of the Sawada-Kotera equation. Besides, we use the hyperbolic tangent ($tanh$) activation function and initialize weights of the neural network with the Xavier initialization. The derivatives of the network u with respect to time t and space x are derived by automatic differentiation.

The loss function of stage one is (2.5) and that of stage two is (2.17) where we choose $N_s = 2000, N_c = 20$ and MSE_m is given by

$$\begin{aligned}
MSE_m &= \frac{1}{N_c}\sum_{i=1}^{N_c} |m(t_m^i) - m(t_0)|^2\\
&\approx \frac{1}{N_c}\sum_{i=1}^{N_c}\left|\sum_{j=2}^{N_x}\widehat{u}(x^j, t_m^i)\frac{x_1-x_0}{N_x-1} - \sum_{j=2}^{N_x}u(x^j,t_0)\frac{x_1-x_0}{N_x-1}\right|^2,
\end{aligned} \tag{4.6}$$

where $u(x^j, t_0)$ and $\widehat{u}(x^j, t_m^i)$ represent the true value and predictive value, respectively. Then the L-BFGS algorithm is utilized to optimize loss functions above.

The numerical solution can be obtained through our two-stage PINN method. When the original PINN method is applied in stage one, it achieves the relative $\mathbb{L}_2$ error of 1.157014e-02 after 14564 times iterations in about 9636.1083 seconds. In stage two, the conserved quantity ($m = \int_{x_0}^{x_1} u\mathrm{d}x \approx \sum_{j=2}^{N_x} u(x^j,t)\frac{x_1-x_0}{N_x-1}$) is considered. After 8847 times iterations in about 6117.9302 seconds, the relative $\mathbb{L}_2$ error of u is 8.054410e-03.

Fig. 7 exhibits the density diagrams of soliton molecule $u(x,t)$ and comparison between the predicted solutions and exact solutions at different time points $t = -0.01, 0, 0.01$. It is obvious that dynamic behavior of this solution can be well simulated with high precision from contrast in the (a) of Fig. 7.

4.2. The M-shape double-peak soliton for $c < \frac{1}{2}$

Here we take $c = \frac{1}{4}, k = a = 1, b = 0$, $[x_0, x_1] = [-8, 8], [t_0, t_1] = [-0.01, 0.01]$, then the initial-boundary conditions are obtained

$$\begin{aligned}
u(-8,t) &= -1 + \frac{3\left(\frac{1}{4} + \cosh(\sqrt{3}(-9t-8))\right)}{2\left(\frac{\cosh(\sqrt{3}(-9t-8))}{4}+1\right)^2},\\
u(8,t) &= -1 + \frac{3\left(\frac{1}{4} + \cosh(\sqrt{3}(-9t+8))\right)}{2\left(\frac{\cosh(\sqrt{3}(-9t+8))}{4}+1\right)^2},
\end{aligned}$$

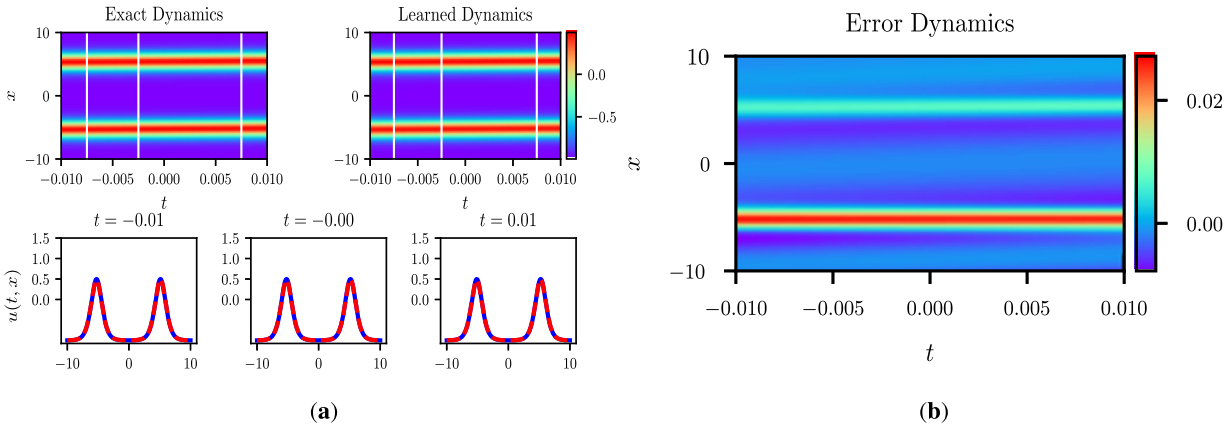

Fig. 7. (Color online.) Soliton molecule $u(x,t)$ of the Sawada-Kotera equation by two-stage PINN based on conserved quantities: **(a)** The density diagrams and comparison between the predicted solutions and exact solutions at the three temporal snapshots of $u(x,t)$; **(b)** The error density diagram of $u(x,t)$.

$$u_0(x) = -1 + 3\frac{\frac{1}{4} + \cosh(\sqrt{3}(0.09+x))}{2(\frac{1}{4}\cosh(\sqrt{3}(0.09+x)) + 1)^2}. \tag{4.7}$$

Then, we construct a 7-layer feedforward neural network with 40 neurons per hidden layer to simulate the M-shape double-peak soliton of the Sawada-Kotera equation. The initial-boundary data is obtained via the same generation and sampling method in Section 4.1 and here we choose $N_x = 513, N_t = 201$ as well. Moreover, the selected values of N_u, N_f, N_s, N_c and the neural network setting, such as loss functions, the optimization algorithm, the activation function and so on, are the same as the previous section.

By means of the two-stage PINN method based on conserved quantities, we finally acquire the data-driven M-shape double-peak soliton solution.

In stage one, the PINN model achieves the relative $\mathbb{L}_2$ error of 1.557140e-03 after 5294 times iterations in about 1927.8552 seconds. In stage two, where we introduce the conserved quantity into the neural network, after 4565 times iterations in about 1796.5973 seconds, the relative $\mathbb{L}_2$ error of u is 8.148663e-04.

In Fig. 8, we present the density diagrams of M-shape double-peak soliton $u(x,t)$ and comparison between the predicted solutions and exact solutions. Besides, in the (b) and (c) of Fig. 8, error density diagrams generated by the original PINN and two-stage PINN based on conserved quantities are plotted separately, which fully verified the advantage of high precision of our improvement.

4.3. The kink-antikink molecule (KAKM) or plateau soliton for $c = \frac{1}{2}$

The initial-boundary conditions are given by

$$\begin{aligned}
u(-6,t) &= -1 + \frac{3\left(\frac{1}{2} + \cosh(\sqrt{3}(-9t-6))\right)}{\left(\frac{\cosh(\sqrt{3}(-9t-6))}{2} + 1\right)^2}, \\
u(6,t) &= -1 + \frac{3\left(\frac{1}{2} + \cosh(\sqrt{3}(-9t+6))\right)}{\left(\frac{\cosh(\sqrt{3}(-9t+6))}{2} + 1\right)^2}, \\
u_0(x) &= -1 + 3\frac{\frac{1}{2} + \cosh(\sqrt{3}(0.09+x))}{(\frac{\cosh(\sqrt{3}(0.09+x))}{2} + 1)^2},
\end{aligned} \tag{4.8}$$

after choosing corresponding parameters as $c = \frac{1}{2}, k = a = 1, b = 0$, $[x_0, x_1] = [-6, 6]$, $[t_0, t_1] = [-0.01, 0.01]$.

Similarly, we divide spatial region $[x_0, x_1] = [-6, 6]$ into $N_x = 513$ discrete equidistance points and time region $[t_0, t_1] = [-0.01, 0.01]$ into $N_t = 201$ discrete equidistance points. Thus, the solution u is discretized into 513×201 data points in the given spatiotemporal domain. The initial-boundary dataset sampled randomly is served as input to the neural network for training. In the process of establishing the two-stage physics-informed neural network, we also adopt the fully-connected structure with the Xavier initialization and hyperbolic tangent activation function. The loss functions of two stages are optimized in the same way as described above. The only difference is the number of hidden layers. We construct a 8-layer feedforward neural network with 40 neurons per hidden layer here.

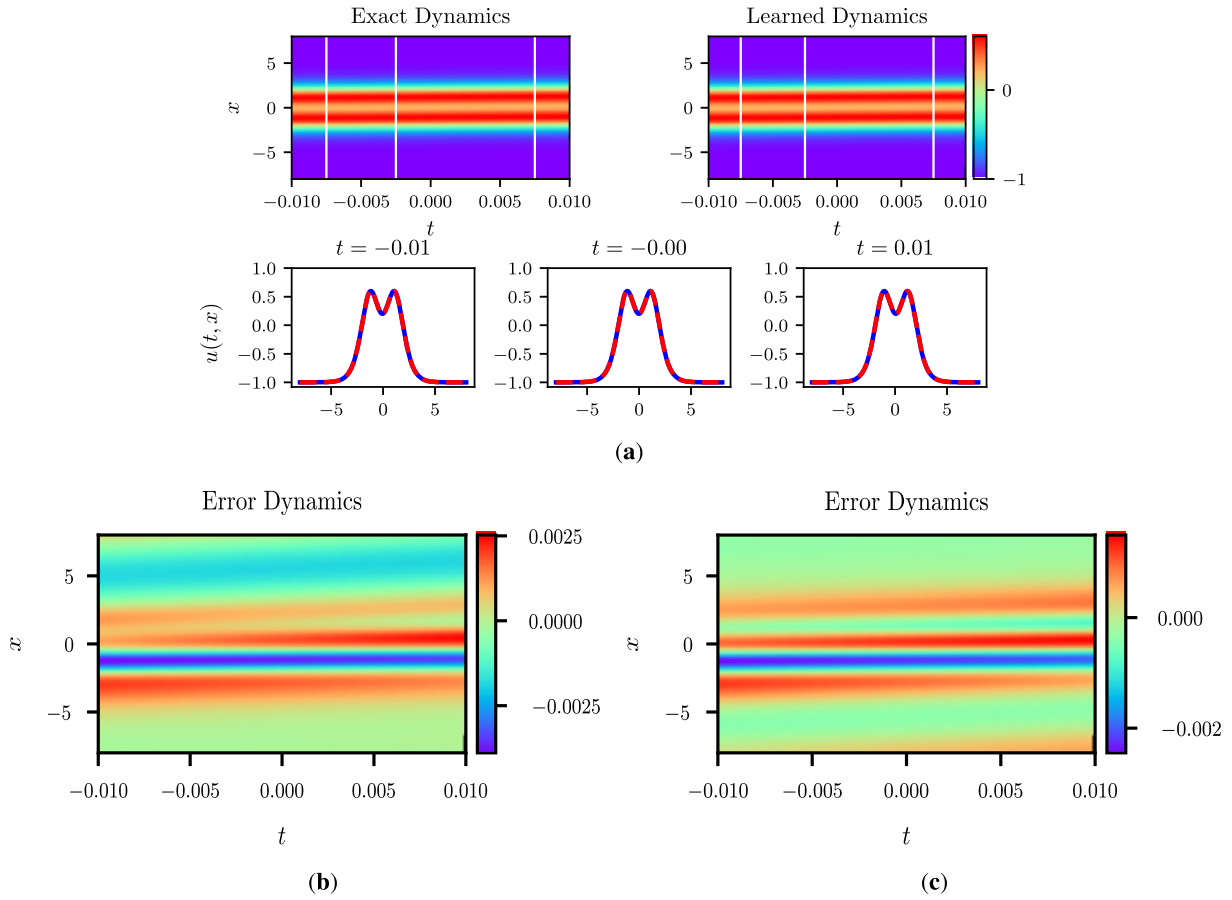

Fig. 8. (Color online.) M-shape double-peak soliton $u(x,t)$ of the Sawada-Kotera equation: **(a)** The density diagrams and comparison between the predicted solutions and exact solutions at the three temporal snapshots of $u(x,t)$ by two-stage PINN based on conserved quantities; **(b)** The error density diagram of $u(x,t)$ by original PINN; **(c)** The error density diagram of $u(x,t)$ by two-stage PINN based on conserved quantities.

After 5381 times iterations in about 2310.8078 seconds, the relative $\mathbb{L}_2$ error of u is 1.766738e-03 in stage one. The numerical results show that after 2972 times iterations, our two-stage PINN model based on conserved quantities achieves the relative $\mathbb{L}_2$ error of 3.678135e-04 in about 1315.2801 seconds in stage two.

Fig. 9 exhibits the density diagrams of the plateau soliton $u(x,t)$ and comparison between the predicted solutions and exact solutions at different time points $t = -0.01, 0, 0.01$. Similarly, error density diagrams generated by the original PINN and two-stage PINN based on conserved quantities are plotted respectively in the (b) and (c) of Fig. 9. Most notably, the relative $\mathbb{L}_2$ of $u(x,t)$ by two-stage PINN based on conserved quantities nearly reach to 5e-4, about one order of magnitude lower than that by the original PINN. This result accentuates that the improved method can enhance the performance in terms of accuracy.

4.4. The single-peak soliton for $c > \frac{1}{2}$

We choose $c = 1, k = a = 1, b = 0$, $[x_0, x_1] = [-6, 6]$, $[t_0, t_1] = [-0.01, 0.01]$, which yields

$$u(-6,t) = -1 + \frac{6 + 6\cosh(\sqrt{3}(-9t-6))}{(\cosh(\sqrt{3}(-9t-6))+1)^2},$$

$$u(6,t) = -1 + \frac{6 + 6\cosh(\sqrt{3}(-9t+6))}{(\cosh(\sqrt{3}(-9t+6))+1)^2},$$

$$u_0(x) = -1 + 6\frac{1 + \cosh(\sqrt{3}(0.09+x))}{(\cosh(\sqrt{3}(0.09+x))+1)^2}. \tag{4.9}$$

After exploiting the same data discretization and sampling method, spatial region $[x_0, x_1] = [-6, 6]$ and time region $[t_0, t_1] = [-0.01, 0.01]$ are divided into $N_x = 513$ and $N_t = 201$ discrete equidistance points, respectively. Based on the

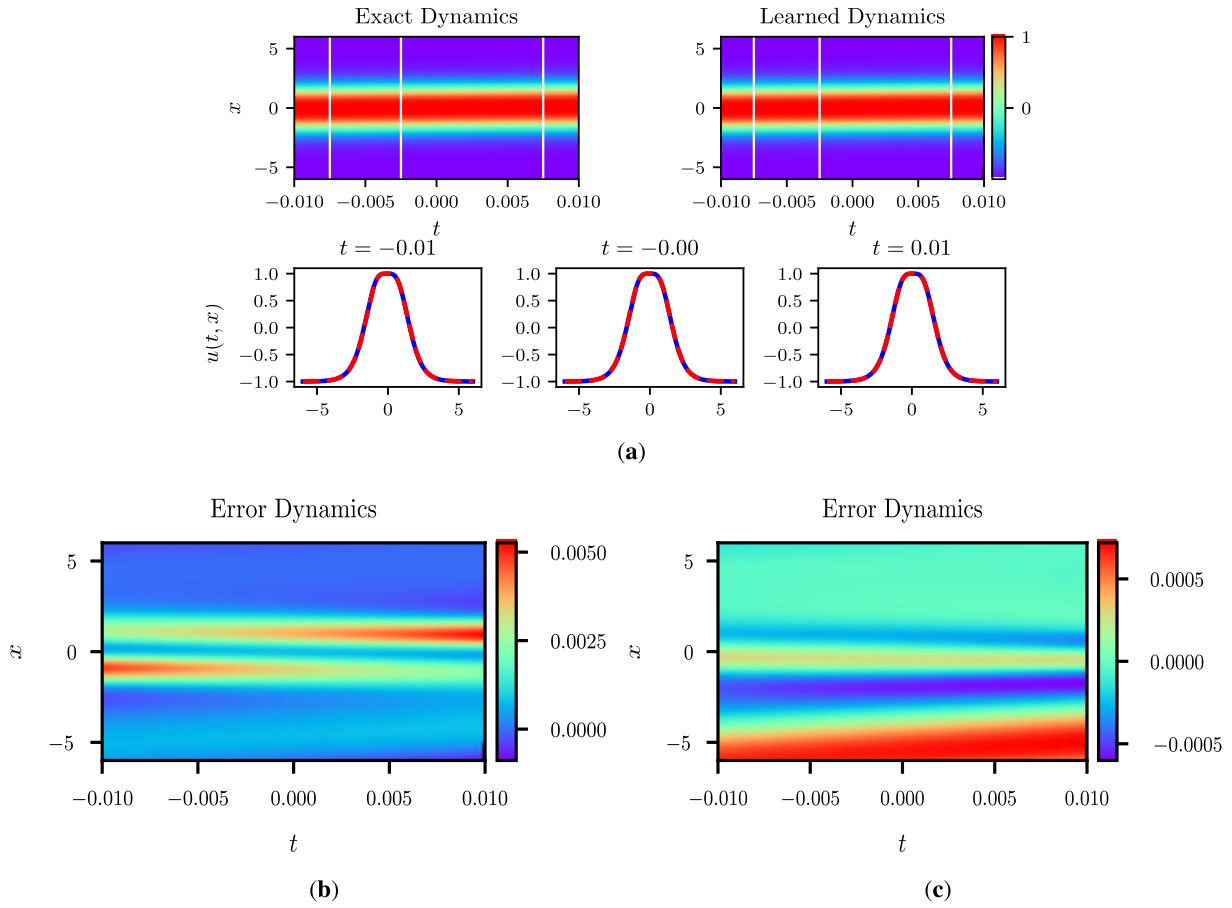

Fig. 9. (Color online.) Plateau soliton $u(x,t)$ of the Sawada-Kotera equation: **(a)** The density diagrams and comparison between the predicted solutions and exact solutions at the three temporal snapshots of $u(x,t)$ by two-stage PINN based on conserved quantities; **(b)** The error density diagram of $u(x,t)$ by original PINN; **(c)** The error density diagram of $u(x,t)$ by two-stage PINN based on conserved quantities.

training data sub-sampled by the Latin hypercube sampling method, a 8-layer feedforward neural network with 40 neurons per hidden layer is established to derive numerical solutions in the form of single-peak soliton after setting up parameters as $N_u = 100, N_f = 2000$. Likewise, we employ the Xavier initialization and hyperbolic tangent ($tanh$) activation function as well as the L-BFGS algorithm to optimize loss functions where $N_s = 2000, N_c = 20$.

With the advantage of the proposed two-stage PINN method, we finally reproduce the single-peak soliton solution.

In stage one, we construct the original PINN model. After 1583 times iterations in about 813.6134 seconds, the relative $\mathbb{L}_2$ error of u is 1.772205e-02. In stage two, where the conserved quantity is considered, the relative $\mathbb{L}_2$ error of u is 9.931406e-03 after 1034 times iterations in about 611.5201 seconds.

In Fig. 10, the density diagrams of single-peak soliton $u(x,t)$ and comparison between the predicted solutions and exact solutions are displayed. Through comparing error density diagrams of two methods showed in the (b) and (c) of Fig. 10, it demonstrates that our two-stage PINN method based on conserved quantities is also more accurate for simulating single-peak soliton.

In Fig. 11, the three-dimensional plots of four structures are plotted respectively: soliton molecule, M-shape double-peak soliton, plateau soliton and single-peak soliton. It illustrates that the two-stage PINN method can effectively reproduce different dynamic behaviors.

Similarly, according to the relative $\mathbb{L}_2$ error of the PINN method (RE_1) and that of the two-stage PINN method based on conserved quantities (RE_2), the error reduction rate (ERR) can be obtained. In addition, Table 3 shows the relative $\mathbb{L}_2$ errors of above solutions as well as the contrast of two methods in terms of error reduction rates.

From Table 3, it can be seen that the proposed two-stage PINN method based on conserved quantities remarkably improves the original PINN method according to error reduction rates. Especially for the plateau soliton, its error reduction rate (79.18%) is extraordinarily significant in the numerical experiments. What's more, our method can improve the accuracy by an order of magnitude in all experiments above. Consequently, our improvement is shown to effectively enhance the prediction accuracy.

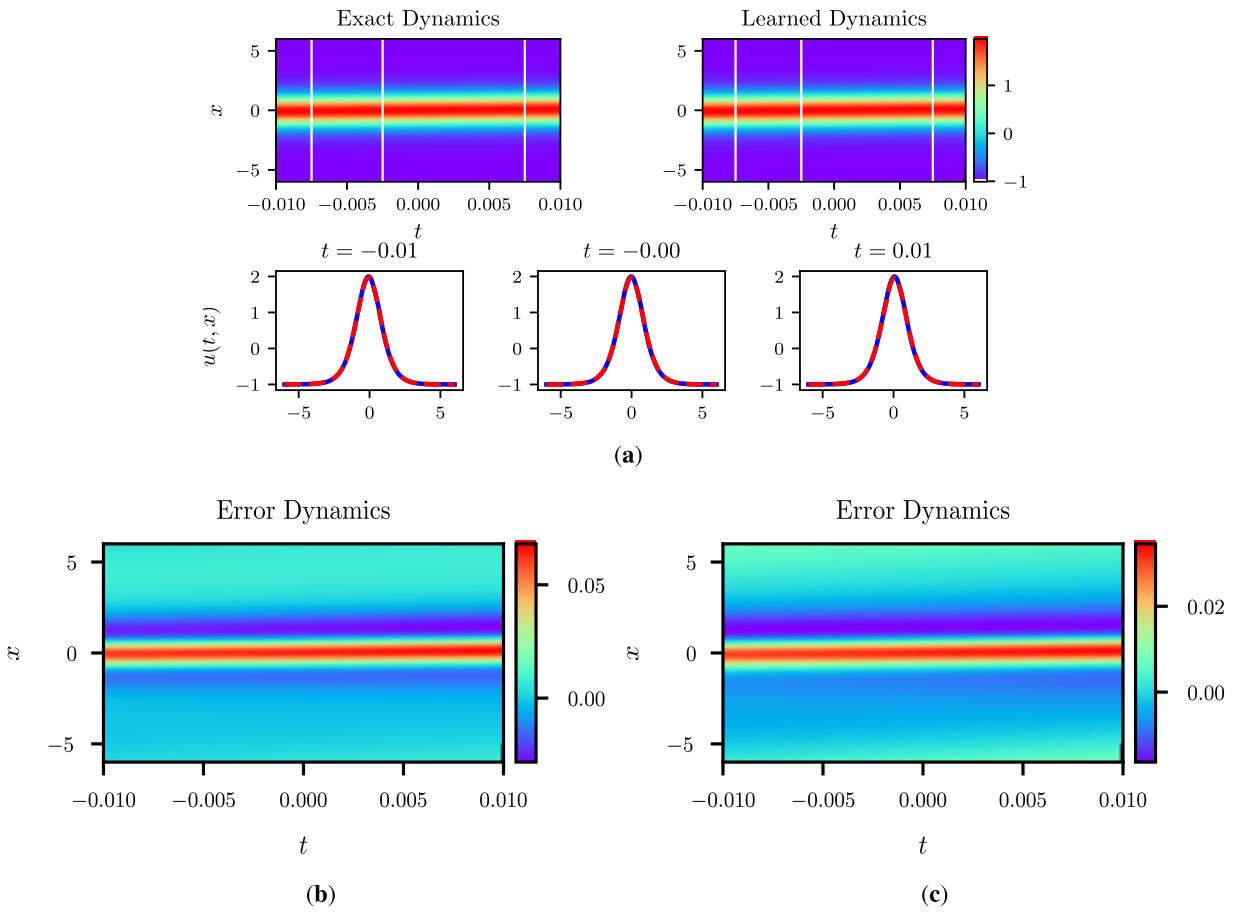

Fig. 10. (Color online.) Single-peak soliton $u(x,t)$ of the Sawada-Kotera equation: **(a)** The density diagrams and comparison between the predicted solutions and exact solutions at the three temporal snapshots of $u(x,t)$ by two-stage PINN based on conserved quantities; **(b)** The error density diagram of $u(x,t)$ by original PINN; **(c)** The error density diagram of $u(x,t)$ by two-stage PINN based on conserved quantities.

Table 3
Soliton molecule and new types of solitons of the Sawada-Kotera equation: relative $\mathbb{L}_2$ errors of PINN and two-stage PINN based on conserved quantities as well as error reduction rates.

Solution \ Method	PINN	Two-stage PINN	Error reduction rate
Soliton molecule	1.157014e-02	8.054410e-03	30.39%
M-shape soliton	1.557140e-03	8.148663e-04	47.67%
Plateau soliton	1.766738e-03	3.678135e-04	79.18%
Single-peak soliton	1.772205e-02	9.931406e-03	43.96%

5. Remark

5.1. Effect of random initialization

Considering that weights of the neural network are initialized with the Xavier initialization, the setting of the parameter *seed* in the codes will affect the numerical results. For the sake of verifying the stableness of this two-stage PINN method, numerical experiments with different randomly generated seed values are performed to explore the impact of random initialization on the results. Ten sets of numerical experiments under the condition of different initial seeds are carried out for every example, which are displayed in Table 6-Table 13 in Appendix A. In Table 4, we present the average iteration times and error reduction rates of two methods (PINN and two-stage PINN based on conserved quantities), where the iteration times of the two-stage PINN are the total iteration times of stage one and stage two.

As can be seen from Table 6-Table 13 and Table 4, when different seed values are taken, our method still shows satisfactory performance. After several random experiments, the error reduction rates all maintain at a considerably good level,

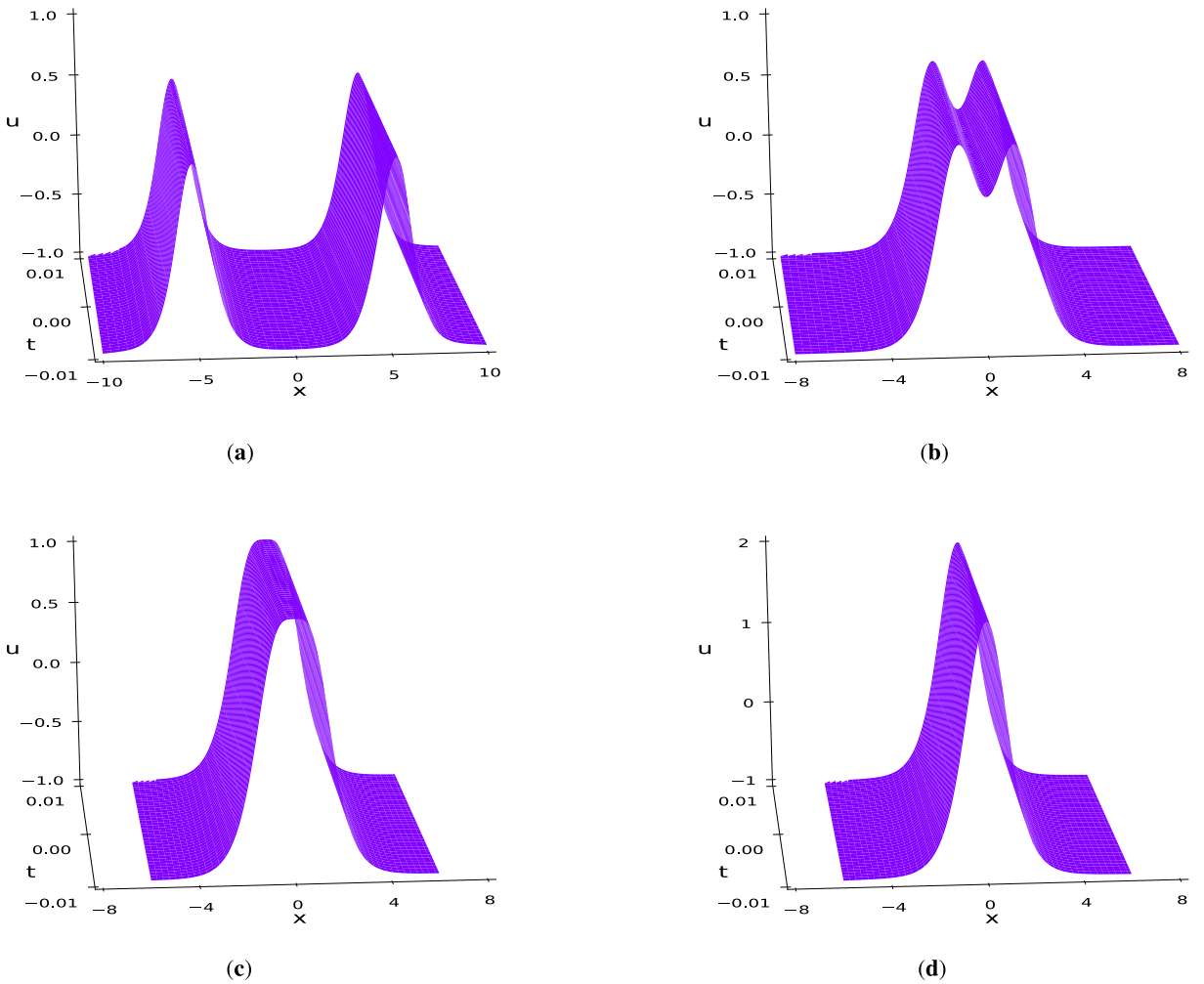

Fig. 11. (Color online.) The three-dimensional plots of background induced soliton and soliton molecules $u(x,t)$ by two-stage PINN based on conserved quantities: **(a)** Soliton molecule; **(b)** M-shape double-peak soliton; **(c)** Plateau soliton; **(d)** Single-peak soliton.

Table 4
Average iteration times and error reduction rates of PINN and two-stage PINN based on conserved quantities.

Solution	Average iteration times		Average error reduction rate
	PINN	Two-stage PINN	
One soliton (u)	176.3	1977.1	23.49%
One soliton (v)	176.3	1977.1	28.71%
Interaction solution (u)	1033	7613.5	20.48%
Interaction solution (v)	1033	7613.5	20.50%
Soliton molecule	6755.7	12145.9	34.67%
M-shape soliton	5200.7	9723.4	40.52%
Plateau soliton	6231.4	10662.7	45.62%
Single-peak soliton	2036.3	4091.1	49.49%

which implies that our two-stage method has good stability. Especially for the data-driven M-shape, plateau and single-peak soliton solution of the Sawada-Kotera equation, the two-stage PINN method based on conserved quantities outperforms the original PINN method in accuracy. Also remarkably, many of the above experiments can improve the accuracy of the solution by an order of magnitude, such as No. 1, No. 5, No. 8 in Table 11 and so on. What's more, the iteration times of the last four solutions in stage two are almost the same as those in stage one, but the prediction accuracy can be effectively improved. However, for the one-soliton solution of the Boussinesq-Burgers equations and interaction solution of the classical Boussinesq-Burgers equations, the training cost of the second stage is several times than that of the first stage. According to

Table 5
Average iteration times and error reduction rates of PINN and two-stage PINN (fine-tuning) based on conserved quantities.

Solution	Average iteration times		Average error reduction rate
	PINN	Two-stage PINN (fine-tuning)	
One soliton (u)	176.3	254.9	-0.45%
One soliton (v)	176.3	254.9	0.28%
Interaction solution (u)	1033	1457.9	1.08%
Interaction solution (v)	1033	1457.9	2.78%
Soliton molecule	6755.7	6815.8	1.99%
M-shape soliton	5200.7	5264.9	2.53%
Plateau soliton	6231.4	6291.5	-2.50%
Single-peak soliton	2036.3	2129.6	-5.59%

the number of iterations in stage one, we can find out that the first stage of these two solutions only needs about several hundred iterations to obtain the data-driven solution with satisfied accuracy, which means that the original PINN method has enough ability to train these two solutions well. Nevertheless, the number of iterations in the first stage of following four solutions generally requires several thousand times, which is much higher than the previous two solutions. As a common sense, compared to an unsatisfactory numerical experiment, it will be much more difficult if we want to notably improve a great one. Since in the first two solutions, the first stage has pretty good performance in accuracy and efficiency, we have to sacrifice considerable efficiency in order to significantly improve the accuracy.

The experimental results provide a basis for the stability of our two-stage PINN method with respect to initialization, which make our explanation more intuitive and reliable.

5.2. Comparison between training a separate PINN and fine-tuning the same network in stage two

In this part, numerical experiments are carried out to show the comparison between training a separate PINN and fine-tuning the same network in stage two.

With the advantage of transfer learning, we fine-tune the same network after the first stage in the following ways. Firstly, we save the weight matrixes and bias vectors of the first-stage network at the end of the iteration process and then those of the second-stage neural network are initialized with the saved data. Thus, the difference between the training of two stages also includes the weight initialization besides the mean squared error loss function. The role of the first stage is not only to generate the rough numerical solution $\widehat{u}_1(x,t)$, which facilitates further optimization in stage two, but also to transmit the data of weight matrixes and bias vectors corresponding to the approximate solution to the second-stage training. In this sense, the second-stage training is based on stage one instead of training from scratch, which helps to accelerate convergence to the approximate optimal solution. Under the condition of different initial seeds, the results of numerical experiments of every example are obtained by fine-tuning the same network after the first stage, which are presented in Table 14-Table 21 in Appendix A. Here, average iteration times and error reduction rates of PINN and two-stage PINN (fine-tuning) based on conserved quantities are shown in Table 5.

By contrasting the above results in Table 4 and Table 5, it demonstrates that with regard to the two-stage PINN (fine-tuning), the accuracy improvement on the basis of stage one (PINN) is far from satisfactory despite the reduction in training costs compared with the two-stage PINN. Based on above-mentioned numerical results, we speculate that the way by using pre-trained weight matrixes and bias vectors as the initial values will merely lead to the numerical solution near that obtained in the first stage rather than the more optimal solution.

Next, let's analyze the reason why we need to train a separate neural network in the second training stage instead of fine-tuning the same network after the first stage. The purpose of the first stage of training is to acquire the rough numerical solution $\widehat{u}_1(x,t)$, and then based on conserved quantities, we aim to achieve further optimization of the numerical solution $\widehat{u}_1(x,t)$ by adding two items (MSE_s and MSE_m) into the mean squared error loss function. However, if we use the fine-tuning method to inherit the weights and biases obtained in the first stage as the initialization parameters in the second stage, it is not theoretically reasonable enough. For two relatively close solutions $\widehat{u}_1(x,t)$ and $\widehat{u}_2(x,t)$ (the proximity here means that both of them can be used as good approximations of the real solution $u(x,t)$), every weight and bias at the corresponding position between them can differ greatly. Take a fully-connected neural network of depth L as an example to explain the previous sentence. The connection between layers and the relation between input $\mathbf{x}^0$ and output $u(\mathbf{x}^0, \mathbf{\Theta})$ are mentioned in (2.3)-(2.4). Here we use $\Theta_1 = \left\{\mathbf{w}_1^l, \mathbf{b}_1^l\right\}_{l=1}^L$ and $\Theta_2 = \left\{\mathbf{w}_2^l, \mathbf{b}_2^l\right\}_{l=1}^L$ to denote parameters in the networks of $\hat{u}_1(x,t)$ and $\hat{u}_2(x,t)$ separately. Due to the complication of the networks, it is not required to achieve

$$\mathbf{w}_{1,ij}^l \approx \mathbf{w}_{2,ij}^l, \mathbf{b}_{1,i}^l \approx \mathbf{b}_{2,i}^l, \forall l = 1, 2, \cdots, L, i = 1, 2, \cdots, N_l, j = 1, 2, \cdots, N_{l-1}, \tag{5.1}$$

(here the symbol '≈' in Eq. (5.1) means close enough) but the composition of the affine transformation and activation function can realize that both numerical solutions $\hat{u}_1(x,t)$ and $\hat{u}_2(x,t)$ are within required accuracy in the given domain and period.

I train a separate neural network in the second stage in order to find the more optimal solution with better constraints. But if we use the idea of weight transfer, it will maybe not run out of the neighborhood of the local optimum obtained in the first stage and can only be slightly optimized nearby.

Of course, pre-training and fine-tuning are very popular tricks in deep learning. The goal of transfer learning is to apply the knowledge or patterns learned in previous fields or tasks to novel but related fields or problems and it does exert advantage in many applications. However, in this paper, the problem itself has not changed and we aim to remarkably improve prediction accuracy compared to the original PINN method instead of obtaining the somewhat satisfactory solution solely. This may be the reason why the transfer learning algorithm is not suitable here.

5.3. Summary of method

In particular, we should utilize the control variable method by running the first-stage training for longer time (using the same number of iterations as total iterations of two-stage training) to confirm the improvement of the generalization ability is contributed from more training iterations or the second stage PINN. As is shown in Fig. 2, there are two conditions for the iteration termination: one is that the number of iterations exceeds the maximum number of iterations, and the other is that the difference of the mean squared error losses between two adjacent iterations is less than a predetermined constant ε (equivalent to convergence in the sense of the difference ε). In all codes, it can be seen from the operation results that the training procedures stop because the second terminating condition of iterations is met. However, our subsequent practice shows that the first-stage training cannot go on effectively. Thus, it is not feasible to make the total number of iterations the same. Of course, in the accuracy comparison of this paper, our scheme is also reasonable in the sense of controlling the same convergence conditions. Based on the rough numerical solution $\widehat{u}_1(x,t)$ in stage one and conserved quantities, the second stage can converge to a local optimum closer to the global optimum by modifying the loss function. In a word, our two-stage PINN method can be regarded as breaking the convergence of the original PINN method and contribute to the improvement of prediction accuracy and generalization. All the experiments in this paper have such results.

Although it must be admitted that, compared with the original PINN method, our method needs a relatively large loss of efficiency in simple calculation examples, the increase in the number of iterations for complex calculation examples is within an acceptable range. In general, our two-stage method is mainly to deal with some solutions that the original PINN method do not perform well, such as the last four solutions. By adding an overall constraint, we break the original convergence and reach a more optimal extreme point, thereby greatly improving the accuracy of the solution.

Superiority and inferiority of an algorithm is primarily determined by its efficiency and accuracy. As one of the most powerful and revolutionary data-driven approaches, compared with the traditional numerical methods, such as the finite element method (FEM), the main disadvantage of PINN is that it can only reach an accuracy of 10^{-4} scale, which still needs to be raised. Consequently, it is reasonable for our proposed two-stage method to pay more attention to improving accuracy. At the same time, our method can improve the accuracy by an order of magnitude in many numerical experiments, which shows that this improved method is effective and meaningful.

6. Conclusions

In this paper, we aim to devise a more targeted PINN algorithm tailored to the nature of equations by introducing conserved quantities of nonlinear systems into neural networks, which implies that the underlying information of the given equations is dug out to improve the precision and reliability. Moreover, the original PINN method considers the local constraints at certain points solely, which evokes the question of whether we can impose constraints from a global perspective. For these purposes, we propose the two-stage PINN method based on conserved quantities. In stage one, the original PINN is applied. In stage two, we additionally introduce the measurement of conserved quantities into mean squared error loss to train neural networks to achieve further optimization of the numerical solution in the first stage. This methodology provides a promising new direction to devise deep learning algorithms with the advantages of integrable systems.

At the same time, we richly exemplify the use of this improved PINN method by simulating the one-soliton solution of the Boussinesq-Burgers equations as well as the interaction solution between a soliton and one resonant of the classical Boussinesq-Burgers equations. Besides, considering that there is poor study of the data-driven soliton molecules by physics-informed neural networks, we reproduce the dynamical behaviors of the soliton molecule, M-shape double-peak soliton, plateau soliton and single-peak soliton for the Sawada-Kotera (SK) equation.

For the sake of comparing the performances of two methods: the original PINN and two-stage PINN based on the conserved quantities, we calculate error reduction rates according to their own relative $\mathbb{L}_2$ errors. Remarkably, results indicate that two-stage PINN method based on conserved quantities can obviously improve prediction accuracy and enhance the ability of generalization, which implies that our improvement is meaningful in simulating solutions of nonlinear partial differential equations. Meanwhile, this is the first time that features of integrable systems are introduced to PINN method. Thus, our practice can solve partial differential equations much more pertinently and promote the development of this field.

However, our proposed method increases the training cost for improving the accuracy. In the future, we will devote to devise a new physics-informed neural network algorithm which can improve prediction accuracy and generalization ability without sacrificing efficiency.

CRediT authorship contribution statement

Shuning Lin: Methodology, Software, Writing – original draft, Writing – review & editing. **Yong Chen:** Methodology, Supervision, Writing – original draft, Writing – review & editing.

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 authors would like to thank Zhengwu Miao sincerely for providing with support and help. This work is supported by National Natural Science Foundation of China (No. 12175069) and Science and Technology Commission of Shanghai Municipality (No. 21JC1402500 and No. 18dz2271000).

Appendix A. Model comparison among PINN, two-stage PINN and two-stage PINN (fine-tuning) under the condition of different initial seeds

Under the condition of different initial seeds, which are selected randomly, ten sets of numerical experiments for every solution are carried out and displayed in Tables 6–13 to verify the stableness of the two-stage PINN method. Meanwhile, the performance of the two-stage PINN (fine-tuning) is displayed in Tables 14–21.

Table 6
One-soliton solution u of the Boussinesq-Burgers equations under the condition of different initial seeds: relative $\mathbb{L}_2$ errors, iteration times and error reduction rates of PINN and two-stage PINN based on conserved quantities.

Order number	PINN		Two-stage PINN		Error reduction rate
	u	Iteration times	u	Iteration times	
1	1.022047E-03	143	8.037295E-04	1968	21.36%
2	1.513022E-03	183	1.310325E-03	1942	13.40%
3	1.000569E-03	156	7.850035E-04	3062	21.54%
4	6.198014E-04	165	4.354769E-04	1848	29.74%
5	7.494604E-04	153	4.464852E-04	1918	40.43%
6	1.010213E-03	160	7.375601E-04	2001	26.99%
7	3.881981E-04	217	2.996518E-04	1895	22.81%
8	6.711765E-04	184	5.544982E-04	1732	17.38%
9	1.166211E-03	239	9.358580E-04	1903	19.75%
10	8.332990E-04	163	6.541544E-04	1502	21.50%

Table 7
One-soliton solution v of the Boussinesq-Burgers equations under the condition of different initial seeds: relative $\mathbb{L}_2$ errors, iteration times and error reduction rates of PINN and two-stage PINN based on conserved quantities.

Order number	PINN		Two-stage PINN		Error reduction rate
	v	Iteration times	v	Iteration times	
1	4.988110E-02	143	3.052248E-02	1968	38.81%
2	6.346011E-02	183	4.739228E-02	1942	25.32%
3	5.850316E-02	156	4.453980E-02	3062	23.87%
4	5.006774E-02	165	2.932091E-02	1848	41.44%
5	4.581136E-02	153	3.010600E-02	1918	34.28%
6	5.326944E-02	160	4.034909E-02	2001	24.25%
7	1.851491E-02	217	1.484928E-02	1895	19.80%
8	4.567286E-02	184	2.869855E-02	1732	37.16%
9	6.160414E-02	239	4.869631E-02	1903	20.95%
10	4.866762E-02	163	3.835431E-02	1502	21.19%

Table 8
Interaction solution u of the classical Boussinesq-Burgers equations under the condition of different initial seeds: relative $\mathbb{L}_2$ errors, iteration times and error reduction rates of PINN and two-stage PINN based on conserved quantities.

Order number	PINN		Two-stage PINN		Error reduction rate
	u	Iteration times	u	Iteration times	
1	3.336242E-04	977	2.418497E-04	8057	27.51%
2	4.757502E-04	1061	3.713813E-04	7861	21.94%
3	3.087738E-04	1099	2.554277E-04	7438	17.28%
4	2.742805E-04	1035	2.083445E-04	8820	24.04%
5	2.923117E-04	1080	2.631685E-04	6522	9.97%
6	3.879219E-04	1144	3.019368E-04	7880	22.17%
7	4.586962E-04	888	3.864513E-04	8115	15.75%
8	3.163651E-04	909	2.482309E-04	6183	21.54%
9	3.849681E-04	1104	2.955302E-04	7019	23.23%
10	3.401000E-04	1033	2.673134E-04	8240	21.40%

Table 9
Interaction solution v of the classical Boussinesq-Burgers equations under the condition of different initial seeds: relative $\mathbb{L}_2$ errors, iteration times and error reduction rates of PINN and two-stage PINN based on conserved quantities.

Order number	PINN		Two-stage PINN		Error reduction rate
	v	Iteration times	v	Iteration times	
1	2.054654E-03	977	1.593131E-03	8057	22.46%
2	2.795753E-03	1061	2.171737E-03	7861	22.32%
3	2.961905E-03	1099	1.964070E-03	7438	33.69%
4	2.563365E-03	1035	1.955249E-03	8820	23.72%
5	3.115359E-03	1080	2.322891E-03	6522	25.44%
6	2.513343E-03	1144	2.224517E-03	7880	11.49%
7	3.846048E-03	888	3.175677E-03	8115	17.43%
8	3.055887E-03	909	2.441593E-03	6183	20.10%
9	1.965187E-03	1104	1.750304E-03	7019	10.93%
10	2.770811E-03	1033	2.287292E-03	8240	17.45%

Table 10
Soliton molecule of the Sawada-Kotera equation under the condition of different initial seeds: relative $\mathbb{L}_2$ errors, iteration times and error reduction rates of PINN and two-stage PINN based on conserved quantities.

Order number	PINN		Two-stage PINN		Error reduction rate
	u	Iteration times	u	Iteration times	
1	3.311141E-03	4569	2.360810E-03	9090	28.70%
2	1.046591E-03	7784	8.193358E-04	13860	21.71%
3	6.646270E-03	8437	4.429318E-03	15471	33.36%
4	1.834566E-03	4798	8.186580E-04	9908	55.38%
5	1.857327E-03	7942	9.532945E-04	11148	48.67%
6	9.397016E-03	7985	5.271889E-03	13532	43.90%
7	2.178783E-03	5965	1.782485E-03	11920	18.19%
8	2.440131E-03	6890	1.886930E-03	13327	22.67%
9	8.043736E-03	7525	5.785798E-03	13483	28.07%
10	1.689550E-03	5662	9.109926E-04	9720	46.08%

Table 11
M-shape double-peak soliton of the Sawada-Kotera equation under the condition of different initial seeds: relative $\mathbb{L}_2$ errors, iteration times and error reduction rates of PINN and two-stage PINN based on conserved quantities.

Order number	PINN		Two-stage PINN		Error reduction rate
	u	Iteration times	u	Iteration times	
1	1.931914E-03	4433	8.757620E-04	9711	54.67%
2	2.831166E-03	6521	1.596378E-03	11453	43.61%
3	2.332973E-01	4627	1.673380E-01	8548	28.27%
4	3.268729E-03	2750	1.933634E-03	6317	40.84%
5	1.437975E-03	4299	7.104947E-04	8673	50.59%
6	3.341850E-03	8592	2.286872E-03	11872	31.57%
7	2.065562E-03	6318	1.085886E-03	12022	47.43%
8	1.281252E-03	5576	8.728303E-04	10618	31.88%
9	2.276220E-03	4354	1.470364E-03	9364	35.40%
10	2.107036E-01	4537	1.244700E-01	8656	40.93%

Table 12
Plateau soliton of the Sawada-Kotera equation under the condition of different initial seeds: relative $\mathbb{L}_2$ errors, iteration times and error reduction rates of PINN and two-stage PINN based on conserved quantities.

Order number	PINN		Two-stage PINN		Error reduction rate
	u	Iteration times	u	Iteration times	
1	2.473586E-03	6573	8.324058E-04	10506	66.35%
2	1.107452E-01	5203	9.019751E-02	10495	18.55%
3	2.488100E-03	4265	1.160830E-03	7366	53.34%
4	2.208977E-03	8569	9.398346E-04	11776	57.45%
5	1.975001E-03	3688	1.172173E-03	8916	40.65%
6	1.992681E-03	6067	1.402009E-03	13419	29.64%
7	3.450265E-03	9691	1.457149E-03	13925	57.77%
8	2.220231E-03	6090	1.426968E-03	8561	35.73%
9	1.479445E-03	5495	7.942674E-04	9600	46.31%
10	2.880094E-03	6673	1.429492E-03	12063	50.37%

Table 13
Single-peak soliton of the Sawada-Kotera equation under the condition of different initial seeds: relative $\mathbb{L}_2$ errors, iteration times and error reduction rates of PINN and two-stage PINN based on conserved quantities.

Order number	PINN		Two-stage PINN		Error reduction rate
	u	Iteration times	u	Iteration times	
1	5.163666E-03	1351	3.036510E-03	3383	41.19%
2	1.040870E-02	2320	3.798210E-03	6380	63.51%
3	2.276302E-02	2062	5.526031E-03	4043	75.72%
4	8.646699E-03	1332	5.828349E-03	3890	32.59%
5	2.357372E-02	3227	7.597484E-03	4864	67.77%
6	9.075930E-03	2746	5.439070E-03	4262	40.07%
7	5.091968E-03	1724	3.012729E-03	3931	40.83%
8	3.080010E-03	1771	2.310492E-03	3180	24.98%
9	3.939520E-03	2110	2.223910E-03	3646	43.55%
10	4.703115E-03	1720	1.663019E-03	3332	64.64%

Table 14
One-soliton solution u of the Boussinesq-Burgers equations under the condition of different initial seeds: relative $\mathbb{L}_2$ errors, iteration times and error reduction rates of PINN and two-stage PINN (fine-tuning) based on conserved quantities.

Order number	PINN		Two-stage PINN (fine-tuning)		Error reduction rate
	u	Iteration times	u	Iteration times	
1	1.022047E-03	143	1.061781E-03	220	-3.89%
2	1.513022E-03	183	1.514958E-03	227	-0.13%
3	1.000569E-03	156	1.013353E-03	248	-1.28%
4	6.198014E-04	165	6.091569E-04	273	1.72%
5	7.494604E-04	153	7.994637E-04	244	-6.67%
6	1.010213E-03	160	9.958131E-04	253	1.43%
7	3.881981E-04	217	3.770619E-04	283	2.87%
8	6.711765E-04	184	6.506031E-04	269	3.07%
9	1.166211E-03	239	1.174681E-03	308	-0.73%
10	8.332990E-04	163	8.407394E-04	224	-0.89%

Table 15
One-soliton solution v of the Boussinesq-Burgers equations under the condition of different initial seeds: relative $\mathbb{L}_2$ errors, iteration times and error reduction rates of PINN and two-stage PINN (fine-tuning) based on conserved quantities.

Order number	PINN		Two-stage PINN (fine-tuning)		Error reduction rate
	v	Iteration times	v	Iteration times	
1	4.988110E-02	143	5.033882E-02	220	-0.92%
2	6.346011E-02	183	6.267900E-02	227	1.23%
3	5.850316E-02	156	5.641989E-02	248	3.56%
4	5.006774E-02	165	5.031941E-02	273	-0.50%
5	4.581136E-02	153	5.078001E-02	244	-10.85%
6	5.326944E-02	160	5.175873E-02	253	2.84%
7	1.851491E-02	217	1.893970E-02	283	-2.29%
8	4.567286E-02	184	4.325795E-02	269	5.29%
9	6.160414E-02	239	6.062242E-02	308	1.59%
10	4.866762E-02	163	4.727585E-02	224	2.86%

Table 16
Interaction solution u of the classical Boussinesq-Burgers equations under the condition of different initial seeds: relative $\mathbb{L}_2$ errors, iteration times and error reduction rates of PINN and two-stage PINN (fine-tuning) based on conserved quantities.

Order number	PINN		Two-stage PINN (fine-tuning)		Error reduction rate
	u	Iteration times	u	Iteration times	
1	3.336242E-04	977	3.133056E-04	2047	6.09%
2	4.757502E-04	1061	4.693227E-04	1329	1.35%
3	3.087738E-04	1099	3.032740E-04	1400	1.78%
4	2.742805E-04	1035	2.727682E-04	1339	0.55%
5	2.923117E-04	1080	3.053162E-04	1533	-4.45%
6	3.879219E-04	1144	3.882256E-04	1208	-0.08%
7	4.586962E-04	888	4.242304E-04	2014	7.51%
8	3.163651E-04	909	3.237128E-04	1015	-2.32%
9	3.849681E-04	1104	3.835493E-04	1240	0.37%
10	3.401000E-04	1033	3.399872E-04	1454	0.03%

Table 17
Interaction solution v of the classical Boussinesq-Burgers equations under the condition of different initial seeds: relative $\mathbb{L}_2$ errors, iteration times and error reduction rates of PINN and two-stage PINN (fine-tuning) based on conserved quantities.

Order number	PINN		Two-stage PINN (fine-tuning)		Error reduction rate
	v	Iteration times	v	Iteration times	
1	2.054654E-03	977	1.795106E-03	2047	12.63%
2	2.795753E-03	1061	2.820340E-03	1329	-0.88%
3	2.961905E-03	1099	2.633005E-03	1400	11.10%
4	2.563365E-03	1035	2.552349E-03	1339	0.43%
5	3.115359E-03	1080	3.224893E-03	1533	-3.52%
6	2.513343E-03	1144	2.467861E-03	1208	1.81%
7	3.846048E-03	888	3.567078E-03	2014	7.25%
8	3.055887E-03	909	3.037032E-03	1015	0.62%
9	1.965187E-03	1104	1.892184E-03	1240	3.71%
10	2.770811E-03	1033	2.919759E-03	1454	-5.38%

Table 18
Soliton molecule of the Sawada-Kotera equation under the condition of different initial seeds: relative $\mathbb{L}_2$ errors, iteration times and error reduction rates of PINN and two-stage PINN (fine-tuning) based on conserved quantities.

Order number	PINN		Two-stage PINN (fine-tuning)		Error reduction rate
	u	Iteration times	u	Iteration times	
1	3.311141E-03	4569	3.234553E-03	4642	2.31%
2	1.046591E-03	7784	1.009713E-03	7812	3.52%
3	6.646270E-03	8437	6.581568E-03	8466	0.97%
4	1.834566E-03	4798	1.761212E-03	4961	4.00%
5	1.857327E-03	7942	1.803327E-03	7961	2.91%
6	9.397016E-03	7985	9.323797E-03	8125	0.78%
7	2.178783E-03	5965	2.174527E-03	5985	0.20%
8	2.440131E-03	6890	2.413878E-03	6911	1.08%
9	8.043736E-03	7525	7.715137E-03	7625	4.09%
10	1.689550E-03	5662	1.688073E-03	5670	0.09%

Table 19
M-shape double-peak soliton of the Sawada-Kotera equation under the condition of different initial seeds: relative $\mathbb{L}_2$ errors, iteration times and error reduction rates of PINN and two-stage PINN (fine-tuning) based on conserved quantities.

Order number	PINN		Two-stage PINN (fine-tuning)		Error reduction rate
	u	Iteration times	u	Iteration times	
1	1.931914E-03	4433	1.958975E-03	4481	-1.40%
2	2.831166E-03	6521	2.814291E-03	6579	0.60%
3	2.332973E-01	4627	2.308335E-01	4724	1.06%
4	3.268729E-03	2750	3.087881E-03	2809	5.53%
5	1.437975E-03	4299	1.395023E-03	4347	2.99%
6	3.341850E-03	8592	3.201241E-03	8652	4.21%
7	2.065562E-03	6318	2.040382E-03	6381	1.22%
8	1.281252E-03	5576	1.194092E-03	5610	6.80%
9	2.276220E-03	4354	2.177705E-03	4487	4.33%
10	2.107036E-01	4537	2.108079E-01	4579	-0.05%

Table 20
Plateau soliton of the Sawada-Kotera equation under the condition of different initial seeds: relative $\mathbb{L}_2$ errors, iteration times and error reduction rates of PINN and two-stage PINN (fine-tuning) based on conserved quantities.

Order number	PINN		Two-stage PINN (fine-tuning)		Error reduction rate
	u	Iteration times	u	Iteration times	
1	2.473586E-03	6573	2.581013E-03	6604	-4.34%
2	1.107452E-01	5203	1.102495E-01	5328	0.45%
3	2.488100E-03	4265	2.444347E-03	4323	1.76%
4	2.208977E-03	8569	2.252917E-03	8632	-1.99%
5	1.975001E-03	3688	2.229998E-03	3717	-12.91%
6	1.992681E-03	6067	1.912539E-03	6117	4.02%
7	3.450265E-03	9691	3.614261E-03	9760	-4.75%
8	2.220231E-03	6090	2.196172E-03	6158	1.08%
9	1.479445E-03	5495	1.483570E-03	5542	-0.28%
10	2.880094E-03	6673	3.112008E-03	6734	-8.05%

Table 21
Single-peak soliton of the Sawada-Kotera equation under the condition of different initial seeds: relative $\mathbb{L}_2$ errors, iteration times and error reduction rates of PINN and two-stage PINN (fine-tuning) based on conserved quantities.

Order number	PINN		Two-stage PINN (fine-tuning)		Error reduction rate
	u	Iteration times	u	Iteration times	
1	5.163666E-03	1351	4.851073E-03	1659	6.05%
2	1.040870E-02	2320	1.124872E-02	2418	-8.07%
3	2.276302E-02	2062	2.356507E-02	2135	-3.52%
4	8.646699E-03	1332	9.117324E-03	1399	-5.44%
5	2.357372E-02	3227	2.474661E-02	3354	-4.98%
6	9.075930E-03	2746	1.059646E-02	2798	-16.75%
7	5.091968E-03	1724	5.115509E-03	1759	-0.46%
8	3.080010E-03	1771	3.588833E-03	1819	-16.52%
9	3.939520E-03	2110	4.252574E-03	2162	-7.95%
10	4.703115E-03	1720	4.622005E-03	1793	1.72%

References

[1] M. Raissi, P. Perdikaris, G.E. Karniadakis, Physics-informed neural networks: a deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations, J. Comput. Phys. 378 (2019) 686–707.
[2] X.W. Jin, S.Z. Cai, H. Li, G.E. Karniadakis, NSFnets (Navier-Stokes flow nets): physics-informed neural networks for the incompressible Navier-Stokes equations, J. Comput. Phys. 426 (2021) 109951.
[3] E. Kharazmi, Z.Q. Zhang, G.E. Karniadakis, VPINNs: variational physics-informed neural networks for solving partial differential equations, arXiv preprint, arXiv:1912.00873, 2019.
[4] G.F. Pang, L. Lu, G.E. Karniadakis, fPINNs: fractional physics-informed neural networks, SIAM J. Sci. Comput. 41 (4) (2019) A2603–A2626.
[5] L. Yang, X.H. Meng, G.E. Karniadakis, B-PINNs: Bayesian physics-informed neural networks for forward and inverse PDE problems with noisy data, J. Comput. Phys. 425 (2021) 109913.
[6] E. Kharazmi, Z.Q. Zhang, G.E. Karniadakis, hp-VPINNs: Variational physics-informed neural networks with domain decomposition, Comput. Methods Appl. Mech. Eng. 374 (2021) 113547.
[7] M. Raissi, Deep hidden physics models: deep learning of nonlinear partial differential equations, J. Mach. Learn. Res. 19 (1) (2018) 932–955.
[8] Y.B. Yang, P. Perdikaris, Adversarial uncertainty quantification in physics-informed neural networks, J. Comput. Phys. 394 (2019) 136–152.
[9] Z.P. Mao, A.D. Jagtag, G.E. Karniadakis, Physics-informed neural networks for high-speed flows, Comput. Methods Appl. Mech. Eng. 360 (2020) 112789.
[10] M. Raissi, G.E. Karniadakis, Hidden physics models: machine learning of nonlinear partial differential equations, J. Comput. Phys. 357 (2018) 125–141.
[11] A.D. Jagtag, K. Kawaguchi, G.E. Karniadakis, Adaptive activation functions accelerate convergence in deep and physics-informed neural networks, J. Comput. Phys. 404 (2020) 109136.
[12] M. Raissi, A. Yazdani, G.E. Karniadakis, Hidden fluid mechanics: learning velocity and pressure fields from flow visualizations, Science 367 (6481) (2020) 1026–1030.
[13] Y. Shin, J. Darbon, G.E. Karniadakis, On the convergence of physics informed neural networks for linear second-order elliptic and parabolic type PDEs, arXiv preprint, arXiv:2004.01806, 2020.
[14] Q. Zheng, L. Zeng, G.E. Karniadakis, Physics-informed semantic inpainting: application to geostatistical modeling, J. Comput. Phys. 419 (2020) 109676.
[15] A.D. Jagtap, E. Kharazmi, G.E. Karniadakis, Locally adaptive activation functions with slope recovery for deep and physics-informed neural networks, Proc. R. Soc. A 476 (2239) (2020) 20200334.
[16] L. Wang, Z.Y. Yan, Data-driven rogue waves and parameter discovery in the defocusing nonlinear Schrödinger equation with a potential using the PINN deep learning, Phys. Lett. A 404 (2021) 127408.
[17] J. Li, Y. Chen, Solving second-order nonlinear evolution partial differential equations using deep learning, Commun. Theor. Phys. 72 (10) (2020) 105005.
[18] J. Li, Y. Chen, A deep learning method for solving third-order nonlinear evolution equations, Commun. Theor. Phys. 72 (11) (2020) 115003.
[19] J.C. Pu, J. Li, Y. Chen, Soliton, breather and rogue wave solutions for solving the nonlinear Schrödinger equation using a deep learning method with physical constraints, Chin. Phys. B 30 (2021) 060202.
[20] J.C. Pu, J. Li, Y. Chen, Solving localized wave solutions of the derivative nonlinear Schrödinger equation using an improved PINN method, arXiv preprint, arXiv:2101.08593, 2021.
[21] W.Q. Peng, J.C. Pu, Y. Chen, PINN deep learning for the Chen-Lee-Liu equation: rogue wave on the periodic background, arXiv preprint, arXiv:2105.13027, 2021.

[22] X. Wang, J.L. Cao, Y. Chen, Higher-order rogue wave solutions of the three-wave resonant interaction equation via the generalized Darboux transformation, Phys. Scr. 90 (10) (2015) 105201.
[23] X. Wang, Y.Q. Li, H. Fei, Y. Chen, Rogue wave solutions of AB system, Commun. Nonlinear Sci. Numer. Simul. 20 (2) (2015) 434–442.
[24] T. Xu, Y. Chen, Darboux transformation of the coupled nonisospectral Gross-Pitaevskii system and its multi-component generalization, Commun. Nonlinear Sci. Numer. Simul. 57 (2017) 276–289.
[25] T. Xu, Y. Chen, J. Lin, Localized waves of the coupled cubic-quintic nonlinear Schrödinger equations in nonlinear optics, Chin. Phys. B 26 (12) (2017) 120201.
[26] T. Xu, Y. Chen, Localized waves in three-component coupled nonlinear Schrödinger equation, Chin. Phys. B 25 (9) (2016) 090201.
[27] H.D. Wahlquist, F.B. Estabrook, Bäcklund transformation for solutions of the Korteweg-de Vries equation, Phys. Rev. Lett. 31 (23) (1973) 1386.
[28] R. Hirota, J. Satsuma, Nonlinear evolution equations generated from the Bäcklund transformation for the Boussinesq equation, Prog. Theor. Phys. 57 (3) (1977) 797–807.
[29] A. Nakamura, Bäcklund transform and conservation laws of the Benjamin-Ono equation, J. Phys. Soc. Jpn. 47 (4) (1979) 1335–1340.
[30] B. Lu, Bäcklund transformation of fractional Riccati equation and its applications to nonlinear fractional partial differential equations, Phys. Lett. A 376 (28–29) (2012) 2045–2048.
[31] S.J. Chen, W.X. Ma, X. Lü, Bäcklund transformation, exact solutions and interaction behaviour of the (3+1)-dimensional Hirota-Satsuma-Ito-like equation, Commun. Nonlinear Sci. Numer. Simul. 83 (2020) 105135.
[32] X.E. Zhang, Y. Chen, Deformation rogue wave to the (2+1)-dimensional KdV equation, Nonlinear Dyn. 90 (2) (2017) 755–763.
[33] X.E. Zhang, Y. Chen, X.Y. Tang, Rogue wave and a pair of resonance stripe solitons to KP equation, Comput. Math. Appl. 76 (8) (2018) 1938–1949.
[34] X.E. Zhang, Y. Chen, Inverse scattering transformation for generalized nonlinear Schrödinger equation, Appl. Math. Lett. 98 (2019) 306–313.
[35] M.J. Ablowitz, Z.H. Musslimani, Inverse scattering transform for the integrable nonlocal nonlinear Schrödinger equation, Nonlinearity 29 (3) (2016) 915.
[36] F.J. Yu, L. Li, Inverse scattering transformation and soliton stability for a nonlinear Gross-Pitaevskii equation with external potentials, Appl. Math. Lett. 91 (2019) 41–47.
[37] Y. Zhao, E.G. Fan, Inverse scattering transformation for the Fokas-Lenells equation with nonzero boundary conditions, J. Nonlinear Math. Phys. 28 (1) (2021) 38–52.
[38] B.A. Kupershmidt, Mathematics of dispersive water waves, Commun. Math. Phys. 99 (1) (1985) 51–73.
[39] D.S. Agafontsev, V.E. Zakharov, Integrable turbulence generated from modulational instability of cnoidal waves, Nonlinearity 29 (11) (2016) 3551.
[40] D.S. Agafontsev, V.E. Zakharov, Growing of integrable turbulence, Low Temp. Phys. 46 (8) (2020) 786–791.
[41] D.J. Kaup, A higher-order water-wave equation and the method for solving it, Prog. Theor. Phys. 54 (2) (1975) 396–408.
[42] E. Date, On quasi-periodic solutions of the field equation of the classical massive Thirring model, Prog. Theor. Phys. 59 (1) (1978) 265–273.
[43] M. Ito, Symmetries and conservation laws of the classical Boussinesq equation, Phys. Lett. A 104 (5) (1984) 248–250.
[44] S. Kawamoto, Exact explode-decay mode solitary wave solution of the classical Boussinesq equation, J. Phys. Soc. Jpn. 53 (2) (1984) 469–471.
[45] Z.Q. Gu, Classical Liouville completely integrable systems associated with the solutions of Boussinesq-Burgers' hierarchy, J. Math. Phys. 31 (6) (1990) 1374–1380.
[46] X. Geng, Y. Wu, Finite-band solutions of the classical Boussinesq-Burgers equations, J. Math. Phys. 40 (6) (1999) 2971–2982.
[47] X. Glorot, Y. Bengio, Understanding the difficulty of training deep feedforward neural networks, J. Mach. Learn. Res. 9 (2010) 249–256.
[48] K.M. He, X.Y. Zhang, S.Q. Ren, J. Sun, Delving deep into rectifiers: surpassing human-level performance on ImageNet classification, in: Proceedings of the IEEE International Conference on Computer Vision (ICCV), 2015, pp. 1026–1034.
[49] A.G. Baydin, B.A. Pearlmutter, A.A. Radul, J.M. Siskind, Automatic differentiation in machine learning: a survey, J. Mach. Learn. Res. 18 (2018) 1–43.
[50] D.C. Liu, J. Nocedal, On the limited memory BFGS method for large scale optimization, Math. Program. 45 (1) (1989) 503–528.
[51] Y.S. Li, Soliton and Integrable System, Shanghai Scientific and Technological Education Publishing House, Shanghai, 1999 (in Chinese).
[52] Ü. Göktaş, W. Hereman, Symbolic computation of conserved densities for systems of nonlinear evolution equations, J. Symb. Comput. 24 (5) (1997) 591–622.
[53] P. Wang, B. Tian, W.J. Liu, Y. Jiang, Lax pair, Bäcklund transformation and multi-soliton solutions for the Boussinesq-Burgers equations from shallow water waves, Appl. Math. Comput. 218 (5) (2011) 1726–1734.
[54] R. Xu, Darboux transformations and soliton solutions for classical Boussinesq-Burgers equation, Commun. Theor. Phys. 50 (3) (2008) 579–582.
[55] A.M. Wazwaz, A variety of soliton solutions for the Boussinesq-Burgers equation and the higher-order Boussinesq-Burgers equation, Filomat 31 (3) (2017) 831–840.
[56] Y.H. Wang, CTE method to the interaction solutions of Boussinesq-Burgers equations, Appl. Math. Lett. 38 (2014) 100–105.
[57] K. Sawada, T. Kotera, A method for finding N-soliton solutions of the K.d.V. equation and K.d.V.-like equation, Prog. Theor. Phys. 51 (5) (1974) 1355–1367.
[58] P.J. Caudrey, R.K. Dodd, J.D. Gibbon, A new hierarchy of Korteweg-de Vries equations, Proc. R. Soc. A 351 (1666) (1976) 407–422.
[59] L.J. Ye, J. Lin, Different-periodic travelling wave solutions for nonlinear equations, Commun. Theor. Phys. 41 (4) (2004) 481.
[60] A.H. Bilge, Singular travelling wave solutions of the fifth-order KdV, Sawada-Kotera and Kaup equations, J. Phys. A, Math. Gen. 29 (16) (1996) 4967.
[61] S.Y. Lou, Twelve sets of symmetrics of Caudrey-Dodd-Gibbon-Sawada-Kotera equation, Phys. Lett. A 175 (1) (1993) 23–26.
[62] A.S.A. Rady, E.S. Osman, M. Khalfallah, Multi-soliton solution, rational solution of the Boussinesq-Burgers equations, Commun. Nonlinear Sci. Numer. Simul. 15 (5) (2010) 1172–1176.
[63] M.J. Dong, S.F. Tian, X.W. Yan, Nonlocal symmetries, conservation laws and interaction solutions for the classical Boussinesq-Burgers equation, Nonlinear Dyn. 95 (1) (2019) 273–291.
[64] M. Stratmann, T. Pagel, F. Mitschke, Experimental observation of temporal soliton molecules, Phys. Rev. Lett. 95 (14) (2005) 143902.
[65] K. Lakomy, R. Nath, L. Santos, Soliton molecules in dipolar Bose-Einstein condensates, Phys. Rev. A 86 (1) (2012) 013610.
[66] S.Y. Lou, Soliton molecules and asymmetric solitons in three fifth order systems via velocity resonance, J. Phys. Commun. 4 (4) (2020) 041002.
[67] B. Ren, J. Lin, Soliton molecules, nonlocal symmetry and CRE method of the KdV equation with higher-order corrections, Phys. Scr. 95 (7) (2020) 075202.
[68] W. Wang, R.X. Yao, S.Y. Lou, Abundant traveling wave structures of (1+1)-dimensional Sawada-Kotera equation: few cycle solitons and soliton molecules, Chin. Phys. Lett. 37 (10) (2020) 100501.
[69] M.L. Stein, Large sample properties of simulations using Latin hypercube sampling, Technometrics 29 (2) (1987) 143–151.
[70] M.L. Wang, Y.B. Zhou, Z.B. Li, Application of homogeneous balance method to exact solutions of nonlinear equation in mathematical physics, Phys. Lett. A 216 (1–5) (1996) 67–75.
[71] M. Li, W.K. Hu, C.F. Wu, Rational solutions of the classical Boussinesq-Burgers system, Nonlinear Dyn. 94 (2) (2018) 1291–1302.
[72] Y.L. Jiang, C. Chen, Lie group analysis and dynamical behavior for classical Boussinesq-Burgers system, Nonlinear Anal., Real World Appl. 47 (2019) 385–397.