Nonlinear Dyn
https://doi.org/10.1007/s11071-024-10497-y

RESEARCH

Parameter identification of shallow water waves using the generalized equal width equation and physics-informed neural networks: a conservative approximation scheme

Nima Mohammadi · Mostafa Abbaszadeh · Mehdi Dehghan · Clemens Heitzinger

Received: 12 July 2024 / Accepted: 12 October 2024

Abstract In this investigation, we implement a numerical approach employing Physics-Informed Neural Networks (PINN) based on a shallow water waves model described by the generalized equal width (GEW) equation, a highly nonlinear partial differential equation (PDE) as well as an extremely difficult PDE that is well-known for its stiffness. Utilizing a mesh-free technique, we achieve a continuous solution and derive a nonlinear function for the water waves solution using a reduced number of points within the problem domain. To insure the numerical procedure adheres to mass, momentum, and energy conservation, we introduce a new term in the loss function to insure the adherence to these properties and we demonstrate that it performs better compared to PINN. Furthermore, we closely monitor the conservation of mass, momentum, and energy throughout the simulation and on the other hand we estimated unknown parameters of GEW model using inverse PINN with high accuracy. To assess the effectiveness of our proposed methodology, we demonstrate its effectiveness on three classic test scenarios: the propagation of a single solitary wave, the interaction of two solitary waves, and the Maxwellian initial condition.

Keywords Shallow water waves · Generalized equal width equation · Physics-informed neural networks · Inverse physics-informed neural networks · Conservation law

N. Mohammadi · M. Abbaszadeh (✉) · M. Dehghan
Department of Applied Mathematics, Faculty of Mathematics and Computer Sciences, Amirkabir University of Technology (Tehran Polytechnic), No. 424, Hafez Ave., 15914 Tehran, Iran
e-mail: m.abbaszadeh@aut.ac.ir

N. Mohammadi
e-mail: nima_mohammadi@aut.ac.ir

M. Dehghan
e-mail: mdehghan@aut.ac.ir

C. Heitzinger
Institute for Analysis and Scientific Computing, Vienna University of Technology (TU Wien), Wiedner Hauptstraße 8-10, 1040 Vienna, Austria
e-mail: clemens.heitzinger@tuwien.ac.at

School of Mathematical and Statistical Sciences, Arizona State University, Tempe, AZ 85287, USA

1 Introduction

Nonlinear partial differential equations (PDEs) have wide applications across diverse fields, including applied mathematics, mechanics, physics, and chemistry [72]. The analytical solutions of PDEs offer valuable insights into the underlying physics of a problem. However, obtaining analytical solutions is not always feasible for certain PDEs, necessitating the utilization of numerical methods to approximate solutions [56,57].

Numerical methods have experienced significant growth, encompassing finite difference methods [58], finite element methods [15], finite volume methods [53], meshfree methods [51], and the emergence of machine learning. The development of deep learning, driven by advancements in computer technology, data science, and neural network theory [34], has led to the rapid integration of machine learning in solving PDEs. Initially explored by Lagaris et al. in 1998 [43], the application of neural networks to solve ordinary and partial differential equations gained momentum with the introduction of Physics-Informed Neural Networks (PINN) by Raissi et al. [59]. PINN embeds physical laws described by nonlinear PDEs into a loss function, enabling the solution of both forward and inverse problems. Authors of [38] reviewed some of the prevailing trends in embedding physics into machine learning, presented some of the current capabilities and limitations and discussed diverse applications of physics-informed learning both for forward and inverse problems, including discovering hidden physics and tackling high-dimensional problems. A composite deep operator network (DeepONet) is presented in [28] for learning use two datasets with different levels of fidelity to accurately learn complex operators when sufficient high-fidelity data is not available. The DeepONet approach is used in [27] to tain fast and accurate predictions of the nonlinear evolution of instability waves in high-speed boundary layers. However, the problem required specialized numerical algorithms, and augmenting limited observations in this extreme flow regime. The augmented physics-informed neural network (APINN), which adopts soft and trainable domain decomposition and flexible parameter sharing is proposed in [30] to further improve the extended PINN (XPINN) as well as the vanilla PINN methods. Furthermore, to handle PDEs with non-smooth solutions, a variational formulation of PINNs based on the Galerkin method (hp-VPINN) was suggested in [41] and domain decomposition was taken into consideration by the variational hp-VPINN, and comparable pointwise variants have been investigated in cPINN [32]. In [66], a general parallel implementation of PINNs for flow issues using domain decomposition is presented. Some other PNN studies concentrated on training and neural network design construction, for example, by employing multi-fidelity frameworks [50], adaptive activation functions [31], dynamic loss function weights [68] and network architectures based on CNN [23], which can enhance PINN performance across a range of problems. Shengze Cai and et al in a review article [12] listed some developments of PINN method. Additionally, [2] introduces a radial basis function (RBF)-finite difference method designed to solve the improved Boussinesq model, with a focus on error estimation and the characterization of solitary waves. The primary objective of [3] is to apply the compact local integrated RBFs technique to numerically solve the fourth-order time-fractional diffusion-wave system. In [4], the rational RBFs (RRBFs) collocation method, based on the partition of unity (PU) approach, is utilized to obtain the numerical solution of the multi-dimensional Ginzburg-Landau equation.

The regularized-long wave (RLW) equation, proposed by Benjamin et al. [8] and also they calculated its precise expression while keeping in mind the limitations of the initial and boundary conditions [10], this equation serves as a model for small-amplitude long waves on the channel water surface and finds application in phenomena like plasma waves [5] and shallow water waves [29,56]. The Equal-Width (EW) model, proposed by Morrison et al. [52], presents an alternative to RLW and the well-known Korteweg–de Vries (KdV) equations. The EW equation is a specific case within the generalized equal width (GEW) equation, for which various numerical methods have been developed, including the Petrov–Galerkin method [9,64], quadratic collocation method [22,35,63], δ-shaped basis functions [55], compact local integrated radial basis function method combined with adaptive residual subsampling technique [20], quintic B-spline collocation algorithm with two different linearization techniques [79], a septic B-spline collocation method [36], sextic B-spline Subdomain finite element method [37], cubic B-spline base functions as element shape functions and quadratic B-spline base functions as the weight functions are used in a Petrov–Galerkin finite element method [24], local meshless collocation method [1] and sextic B-spline collocation technique [54]. In [39] the primary problem is divided into linear and non-linear subequations, by using the cubic B-spline and Galerkin finite element methods on each sub-equation.

The finite volume framework was extended to model dispersive unidirectional water wave propagation in one spatial dimension by Denys Dutykh et al. Specifically, they considered it in the context of a KdV-BBM type equation [19]. Additionally, in their work [18], they demonstrated that geometrical methods are particularly well-suited for modeling complex nonlinear wave phenomena, offering accuracy and reliability comparable to Fourier-type pseudospectral approaches in capturing the long-term dynamics of KdV equations.

Direct numerical simulation techniques were employed to investigate the collective behavior of soliton ensembles. Previously, this problem was mainly addressed in the context of integrable models, such as the well-known KdV equation. However, in [17], the analysis was extended to include non-integrable KdV-BBM type models, using asymptotic methods alongside Monte Carlo simulations. Craig et al. addressed a fully nonlinear Hamiltonian system to investigate the interactions of individual water waves [13].

The mix-training physics-informed neural networks (MTPINNs) and prior information mix-training physics-informed neural networks (PMTPINNs) are developed in [67] to solve cmKdV equation. Authors of [46] presented a systematic study of the soliton interaction dynamics of the Maccari system. The main aim of [73] is to introduce two extended (3 + 1)- and (2 + 1)-dimensional Painlevé integrable Kadomtsev-Petviashvili (KP) equations. Authors of [70] established exact solutions for nonlinear wave equations. The tanh method and the extended tanh method are used in [71] for analytic treatment for solving the Kuramoto-Sivashinsky and the Kawahara equations.

Concurrently, the PINN method has gained prominence through continuous advancements and applications, particularly in the field of fluid mechanics [33,42,44,45,47,48,60,61,76,77]. It has been applied to various scenarios, including compressible flows [48], turbulent convection flows [45], biomedical flows [42, 77], and free boundary and Stefan problems [6]. For instance, Jin et al. [33] applied physics-informed neural networks (PINNs) to model incompressible flows, ranging from laminar to turbulent regimes. Their simulations utilized two distinct formulations of the Navier–Stokes equations: the vorticity-velocity (VV) and the velocity-pressure (VP) formulations. They also conducted a comprehensive analysis of the weights used in the data/physics components of the loss function and explored a novel method for dynamically adjusting these weights to enhance accuracy and speed up training.

Raissi et al. [61] introduced a physics-informed deep learning framework called Hidden Fluid Mechanics (HFM), which can encode the Navier–Stokes equations-a fundamental set of equations governing fluid flows. Eivazi et al. [21] presented several practical examples in fluid mechanics, such as Burgers' equation, two-dimensional vortex shedding behind a circular cylinder, and minimal turbulent channel flow, demonstrating the effectiveness of the PINN method even with limited and noisy data.

Yang et al. [76] discussed the benefits of a data-driven approach for wall modeling, emphasizing the importance of incorporating physical insights into model inputs. They demonstrated that inputs inspired by eddy population density scalings and vertically integrated thin-boundary-layer equations enhance a neural network's ability to extrapolate to flow conditions beyond the training data.

Additionally, researchers have explored the efficacy of PINNs in solving various KdV equations, including the KdV-Burgers equation and the KdV equation [25], coupled KdV equations [75], the fourth-order Boussinesq equation and the fifth-order KdV equation [14], and the nonlocal modified Korteweg–de Vries (mKdV) equation for numerical solutions and parameter discovery [81].

In this article, we will evaluate numerical solution of the GEW equation that has the following mathematical model:

$$\mathbf{u_t}+\xi\mathbf{u}^{\rho}\mathbf{u_x}-\eta\mathbf{u_{xxt}}=0,\quad (\mathbf{x},\mathbf{t})\in\mathbf{D}\times(0,T],\quad \rho\geq 1, \tag{1.1}$$

where $\mathbf{D} = [\mathbf{x}_{min}, \mathbf{x}_{max}]$, ξ,ρ and η are positive real numbers, with the periodic boundary conditions given by

$$\mathbf{u}(\mathbf{x}_{min}, \mathbf{t}) = \mathbf{u}(\mathbf{x}_{max}, \mathbf{t}), \tag{1.2}$$

$$\mathbf{u_{xx}}(\mathbf{x}_{min}, \mathbf{t}) = \mathbf{u_{xx}}(\mathbf{x}_{max}, \mathbf{t}), \tag{1.3}$$

and initial condition

$$\mathbf{u}(\mathbf{x}, 0) = \mathbf{u}_0(\mathbf{x}),\qquad \mathbf{x}\in\mathbf{D}. \tag{1.4}$$

In this context, a conservative scheme is proposed for the focal model, specifically the shallow water waves model characterized by the generalized equal width (GEW) equation, utilizing physics-informed neural networks (PINNs). The target model must satisfy three essential conservation laws: mass, momentum, and energy conservation. These principles are fundamental in describing the behavior of fluid flow.

The mass conservation law ensures that during the wave's propagation, the total quantity, such as the volume of water, remains constant, meaning the system's total mass is conserved over time. The momentum conservation law in the GEW model relates to the balance of forces acting on the wave. Additionally, energy conservation is maintained through the interaction of potential energy, related to wave elevation, and kinetic energy, associated with wave motion. These conservation laws impose crucial constraints on the wave's evolution over time, ensuring that the solutions are both physically accurate and realistic.

Conservation of invariants is a key quality for both numerical and theoretical research, as it allows for the validation of numerical schemes and the assessment of the accuracy of the results [19]. To meet this requirement, we introduce a new term in the loss function, specifically targeting these critical conservation aspects. This approach results in a robust numerical formulation that ensures the conservation of mass, momentum, and energy.

The ensuing sections of the paper unfold as follows: In Sect. 2, we elucidate the PINN approach. Section 3 delineates the numerical experiments conducted through the PINN framework, and the paper culminates in Section 4, where we articulate our findings and draw conclusions.

2 An abstract framework for PINNs

In this section, first we explain deep neural networks (DNN) and automatic differentiation, which are the basic components of PINNs and next we introduce original PINN framework.

2.1 DNNs and automatic differentiation

The training model in a Physics-Informed Neural Network (PINN) frameworks is built on a fully connected feedforward neural network. This architecture typically comprises an input layer, an output layer, and N hidden layers. The interconnection between these layers is defined as follows [16]:

$$\mathbf{y_n} = \sigma(W_\mathbf{n}.\mathbf{y_{n-1}} + \mathbf{b_n}), \qquad 1 \le \mathbf{n} \le \mathbf{N}, \tag{2.1}$$

$$\mathbf{O} = W_{\mathbf{N}+1}.\mathbf{y_N} + \mathbf{b}_{\mathbf{N}+1}, \tag{2.2}$$

where $\mathbf{n} = 1, 2, \ldots, \mathbf{N}$ refers to the hidden layers, and $\mathbf{N}+1$ is the output layer. The output of the $\mathbf{n}$-th layer is denoted as $\mathbf{y_n}$, and the neural network's overall output is represented by $\mathbf{O}$. The activation function, denoted as $\sigma(.)$, facilitates the neural network in capturing nonlinear relationships. The parameters associated with the $\mathbf{n}$-th layer, namely the weight and bias, are expressed as $W_\mathbf{n}$ and $\mathbf{b_n}$, respectively, and undergo adjustments during the training process.

A fully connected feedforward neural network integrates differentiable activation functions, such as tanh, relu, sin, etc., along with linear summations. The process of automatic differentiation (also known as autodiff, algorithmic differentiation or AD), employed in the backward chain, is utilized for computing derivatives. [7,78]. Numerical and symbolic differentiation are not the same as automatic differentiation. Symbolic differentiation might result in inefficient code since it is challenging to reduce a computer program to a mathematical equation. Numerical differentiation (the finite differences approach), has the potential to cause round-off errors during the discretization process. Higher-order derivatives are more complex to calculate and result in more errors when using any of these classical methods. Lastly, gradient-based optimization algorithms require the partial derivatives of a function to be computed with respect to numerous inputs, which is a computationally demanding task for both of these classical methods. All of these issues can be resolved using automatic distinction. See [7,11] for more details about calculating automatic differentiation.

In this article, we use the components mentioned in Tables 1 and 2 for the neural networks and also piece wise constant decay function for learning rate scheduler in Keras pachage:

2.2 Physics informed neural network (PINN)

In this method we consider the initial-boundary value problem:

$$\frac{\partial \mathbf{u}}{\partial \mathbf{t}} + \mathcal{N}(\mathbf{u}(\mathbf{x}, \mathbf{t})) = 0, \quad \mathbf{x}, \mathbf{t} \in \mathbf{D} \times (0, T], \tag{2.3}$$

$$\mathcal{B}(\mathbf{u}(\mathbf{x}, \mathbf{t})) = g(\mathbf{x}, \mathbf{t}), \qquad \mathbf{x}, \mathbf{t} \in \partial\mathbf{D}, \times (0, T], \tag{2.4}$$

$$\mathbf{u}(\mathbf{x}, 0) = h(\mathbf{x}), \qquad \mathbf{x} \in \mathbf{D}, \tag{2.5}$$

where $\mathcal{N}$ and $\mathcal{B}$ are nonlinear differential operators acting on $\mathbf{u}$, for example in Eq. (1.1) and (1.2), $\mathcal{N}(\mathbf{u}(\mathbf{x}, \mathbf{t})) = \xi \mathbf{u}^\rho \mathbf{u_x} - \eta \mathbf{u_{xxt}}$ and $\mathcal{B}(\mathbf{u}(\mathbf{x}, \mathbf{t})) = \mathbf{u_{xx}}(\mathbf{x}, \mathbf{t})$, $\mathbf{D} \subset \mathbb{R}^d$ a bounded domain, T denotes the final time and $h : \mathbf{D} \to \mathbb{R}$ shows the prescribed initial data and also $\partial\mathbf{D}$ denotes the boundary of the domain $\mathbf{D}$ and $g : \partial\mathbf{D} \times (0, T] \to \mathbb{R}$ denotes the given boundary data. The approach involves constructing a neural network approximation

$$\mathbf{u}_\Theta(\mathbf{x}, \mathbf{t}) \approx \mathbf{u}(\mathbf{x}, \mathbf{t}),$$

Table 1 Artificial Neural Network parameters for forward problems

Hidden layers	Neurons in each layers	Optimizer	Activation	Number of epochs
9	50	Adam	Tanh	80 k

Table 2 Artificial Neural Network parameters for inverse problems

Hidden layers	Neurons in each layers	Optimizer	Activation	Number of epochs
3	30	Adam	Tanh	20 k

to represent the solution of a nonlinear partial differential equation (PDE), where $\mathbf{u}_\Theta : \mathbf{D} \times [0, T] \to \mathbb{R}$ denotes a function implemented by a neural network (NN) with parameters Θ. The continuous-time strategy for parabolic PDEs, as outlined in [59], revolves around the residual of a given NN approximation $\mathbf{u}_\Theta : \mathbf{D} \times [0, T] \to \mathbb{R}$ with respect to the solution $\mathbf{u}$.

$$\mathbf{r}_\Theta(\mathbf{x}, \mathbf{t}) := \partial_t \mathbf{u}_\Theta(\mathbf{x}, \mathbf{t}) + \mathcal{N}[\mathbf{u}_\Theta](\mathbf{x}, \mathbf{t}). \tag{2.6}$$

In order to integrate the PDE residual $\mathbf{r}_\Theta$ into a loss function for minimization, Physics-Informed Neural Networks (PINNs) necessitate additional differentiation to compute the differential operators $\partial_\mathbf{t} \mathbf{u}_\Theta$ and $\mathcal{N}[\mathbf{u}_\Theta]$. Consequently, the PINN term $\mathbf{r}_\Theta$ shares the same parameters as the original network $\mathbf{u}_\Theta(\mathbf{t}, \mathbf{x})$, but adheres to the inherent "physics" encoded in the nonlinear PDE. Both types of derivatives can be readily computed using automatic differentiation present in modern machine learning libraries, such as TensorFlow and PyTorch.

The PINN approach for the solution of the initial and boundary value problem now proceeds by minimizing the loss function:

$$\Psi_\Theta(\chi) := \Psi_\Theta^{\mathbf{r}}(\chi^{\mathbf{r}}) + \Psi_\Theta^0(\chi^0) + \Psi_\Theta^b(\chi^b), \tag{2.7}$$

where χ denotes a collection of training data. Additionally, $\mathbf{r}$, 0, and b represent residual points, initial points, and boundary points, respectively. Furthermore, the loss function Ψ_Θ includes

1. The mean squared residual (MSR) loss corresponding to the PDE:

$$\Psi_\Theta^{\mathbf{r}}(\chi^{\mathbf{r}}) := \frac{1}{\mathbf{N}_r} \sum_{i=1}^{\mathbf{N}_r} \left| \mathbf{r}_\Theta \left(\mathbf{x}_i^{\mathbf{r}}, \mathbf{t}_i^{\mathbf{r}} \right) \right|^2,$$

where $\chi^{\mathbf{r}} := \{(\mathbf{x}_i^{\mathbf{r}}, \mathbf{t}_i^{\mathbf{r}})\}_{i=1}^{\mathbf{N}_r} \subset \mathbf{D} \times (0, T]$ is a set of collocation points and $\mathbf{r}_\Theta$ is the residual caused by the PDE.

2. The MSR related to the initial and boundary conditions:

$$\Psi_\Theta^0(\chi^0) := \frac{1}{\mathbf{N}_0} \sum_{i=1}^{\mathbf{N}_0} \left| \mathbf{u}_\Theta \left(\mathbf{x}_i^0, 0 \right) - h \left(\mathbf{x}_i^0 \right) \right|^2,$$

$$\Psi_\Theta^b(\chi^b) := \frac{1}{\mathbf{N}_b} \sum_{i=1}^{\mathbf{N}_b} \left| \mathcal{B}(\mathbf{u}(\mathbf{x}_i^b, \mathbf{t}_i^b)) - g(\mathbf{x}_i^b, \mathbf{t}_i^b)) \right|^2,$$

in a number of points $\chi^0 := \{(\mathbf{x}_i^0, 0)\}_{i=1}^{\mathbf{N}_0} \subset \mathbf{D} \times \{0\}$ and $\chi^b := \{(\mathbf{x}_i^b, \mathbf{t}_i^b)\}_{i=1}^{\mathbf{N}_b} \subset \partial\mathbf{D} \times (0, T]$. Figure 1 Shows a schematic of PINN.

2.3 The physic-informed neural networks for inverse problems

One of the useful applications of PINN, which is proposed by Raissi et al. [59] is discovery parameters of the partial differential equations using the provided data. We formulate this problem as follows:

$$\frac{\partial \mathbf{u}}{\partial \mathbf{t}} + \mathcal{N}(\mathbf{u}(\mathbf{x}, \mathbf{t}), \lambda) = 0, \qquad \mathbf{x}, \mathbf{t} \in \mathbf{D} \times (0, T] \tag{2.8}$$

where λ is an unknown parameter. The inverse problem estimates λ by setting it as a learning weight in the DNN model and as well as find the optimal value with minimizing the loss function like forward PINN. Also it is considered that there are some additional information on points $\mathcal{D} \subset \mathbf{D}$ in problem domain and on boundary.

$$\Psi_\Theta(\chi^{\mathcal{D}}, \lambda) = \Psi_\Theta^r(\chi^{\mathcal{D}}, \lambda) + \Psi_\Theta^{\mathcal{D}}(\chi^{\mathcal{D}}), \tag{2.9}$$

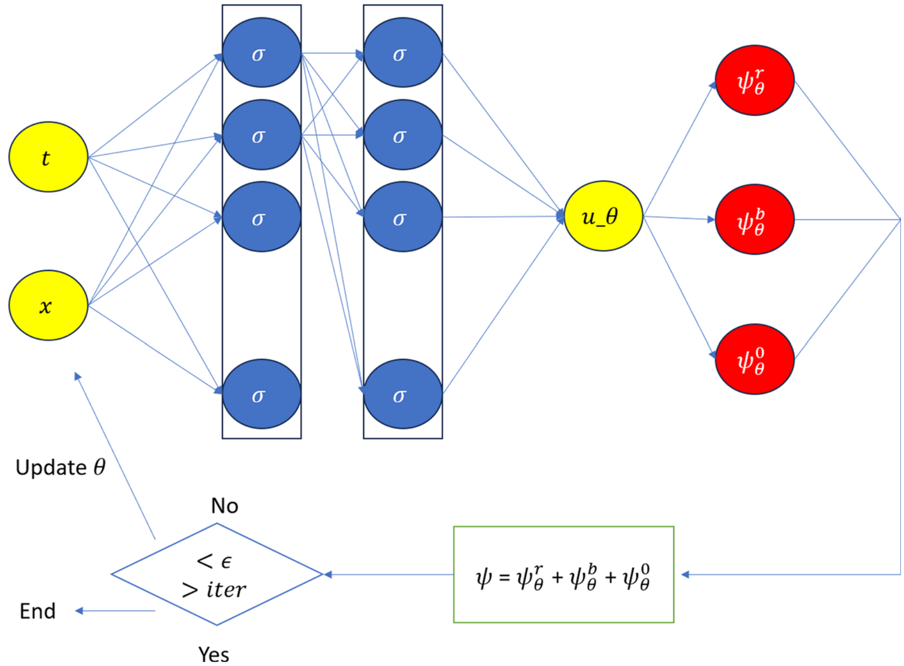


Fig. 1 Schematics of a PINN for solving the diffusion equation

where

$$\Psi_\Theta^{\mathcal{D}}(\chi^{\mathcal{D}}) = \frac{1}{\mathbf{N}_{\mathcal{D}}} \sum_{i=1}^{\mathbf{N}_{\mathcal{D}}} \left| \mathbf{u}_\Theta \left(\mathbf{x}_i^{\mathcal{D}}, \mathbf{t}_i^{\mathcal{D}} \right) - \mathbf{u}_{exact} \left(\mathbf{x}_i^{\mathcal{D}}, \mathbf{t}_i^{\mathcal{D}} \right) \right|^2.$$

Now, we find an approximate value for λ by training our model:

$$\lambda^* = \arg\min_{\Theta,\lambda} \Psi_\Theta(\chi, \lambda). \tag{2.10}$$

2.4 Weighting the loss function terms in PINN

While the baseline PINN approach discussed in the preceding section is extremely effective in solving a large number of linear and nonlinear PDEs, it may not converge at all or result in inaccurate approximations when solving some "stiff" PDEs. Gradient descent exhibits a greedy nature, prioritizing certain components over others, resulting in an uneven descent rate among various loss components. This imbalance hinders the convergence to the correct solution, which explains the occurrence of this phenomenon. In the literature on PINNs, adding weights to Eq. (2.7) is the traditional method for attempting to fix the imbalance [65,68,69]:

$$\Psi_\Theta(\chi) := \lambda_r \Psi_\Theta^{\mathbf{r}}(\chi^{\mathbf{r}}) + \lambda_0 \Psi_\Theta^0(\chi^0) + \lambda_b \Psi_\Theta^b(\chi^b). \tag{2.11}$$

There are several ways to determine these weights' values; a few are listed below:

1. Non-adaptive weighting: In [74], it was suggested to use the adaptive concept in both space and time variables, and a number of sampling techniques that might raise the PINN's accuracy and efficiency were shown. Consequently, the following loss function was suggested:
$$\Psi(\theta) = \Psi_r(\theta) + \Psi_b(\theta) + C\Psi_0(\theta) \tag{2.12}$$
in which $C >> 1$ is a hyper-parameter.
2. Learning rate annealing: In [68], they suggested utilizing weights that are adjusted during training based on statistics from the loss function's backpropagated gradients. Notably, backpropagation is not used to modify the weights directly. Instead, they operate as learning rate coefficients that are refreshed following every training epoch.
3. In the work of Wang et al. [69], the authors computed the NTK kernel matrix for Physics-Informed Neural Networks (PINNs) and, employed a heuristic rationale, dynamically adjusted the weights based on the evolving eigenvalues of the NTK matrix during the training process.
4. Self-adaptive weighting: In McClenny's study [49], the adaptation weights are trained concurrently with the network weights. Consequently, the approximation is compelled to enhance its performance

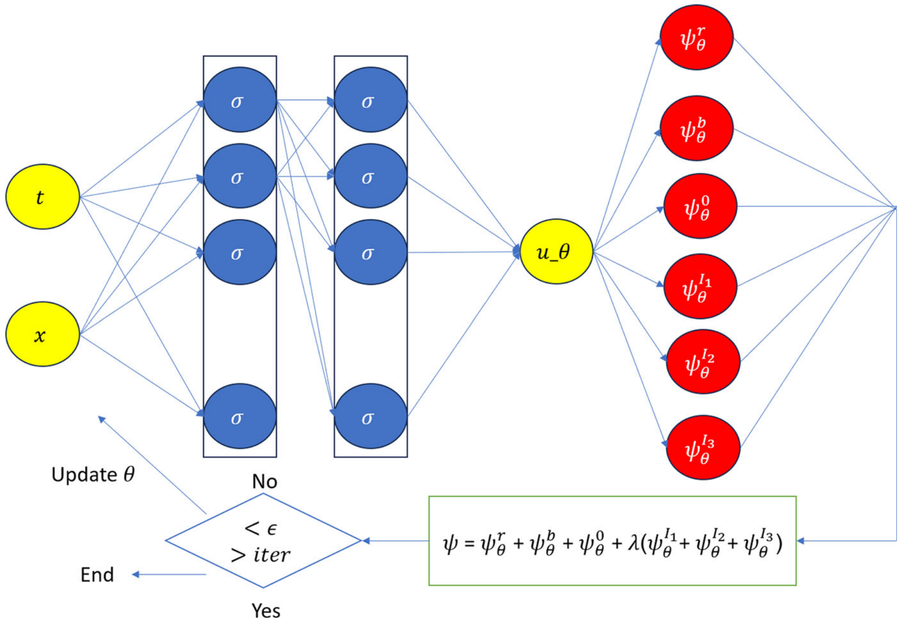


Fig. 2 Schematic of a conservative based PINN

at challenging locations of the solution, including initial, boundary, or residue points, which are accorded higher weights automatically in the loss function. The core concept of Self-Adaptive Physics-Informed Neural Networks (SA-PINNs) involves training the network to simultaneously minimize losses and maximize weights, aiming to identify a saddle point in the cost surface. This approach enables the weights to escalate in conjunction with their associated losses.

2.5 The conservative scheme based PINN

In this part, we add three conditions to the basic PINN so that $\mathbf{u}_\Theta$ has conservative properties. Next, we rewrite the loss function defined in Eq. (2.7) as follows:

$$\Psi_\Theta(\chi) := \Psi_\Theta^{\mathbf{r}}(\chi^{\mathbf{r}}) + \Psi_\Theta^0(\chi^0) + \Psi_\Theta^b(\chi^b) + \lambda(\Psi_\Theta^{\mathbf{I}_1} + \Psi_\Theta^{\mathbf{I}_2} + \Psi_\Theta^{\mathbf{I}_3}), \tag{2.13}$$

where $\lambda << 1$ is a non-adaptive weight, the loss terms $\Psi_\Theta^{\mathbf{I}_1}$, $\Psi_\Theta^{\mathbf{I}_2}$ and $\Psi_\Theta^{\mathbf{I}_3}$ are defined as:

$$\Psi_\Theta^{\mathbf{I}_1} = \frac{1}{\mathbf{N_t}} \sum_{\mathbf{t}_i} |\mathbf{I}_1(\mathbf{t}_i) - \mathbf{I}_1(\mathbf{t}_0)|^2, \tag{2.14}$$

$$\Psi_\Theta^{\mathbf{I}_2} = \frac{1}{\mathbf{N_t}} \sum_{\mathbf{t}_i} |\mathbf{I}_2(\mathbf{t}_i) - \mathbf{I}_2(\mathbf{t}_0)|^2, \tag{2.15}$$

$$\Psi_\Theta^{\mathbf{I}_3} = \frac{1}{\mathbf{N_t}} \sum_{\mathbf{t}_i} |\mathbf{I}_3(\mathbf{t}_i) - \mathbf{I}_3(\mathbf{t}_0)|^2, \tag{2.16}$$

where $\mathbf{N_t}$ is number of time discretization, $\mathbf{t}_i$ is i th time steps that $\mathbf{t}_i \in [0, T]$, $\mathbf{I}_1(\mathbf{t}), \mathbf{I}_2(\mathbf{t}), \mathbf{I}_3(\mathbf{t})$ are conservative conditions to approximate the solution that will define in next section. It is important to note that the absolute values of $\Psi_\Theta^{\mathbf{I}_1}$, $\Psi_\Theta^{\mathbf{I}_2}$, and $\Psi_\Theta^{\mathbf{I}_3}$ depend on the values of $\mathbf{I}_1, \mathbf{I}_2$, and $\mathbf{I}_3$. Since $\mathbf{I}_3$ is much smaller compared to $\mathbf{I}_1$, the PINN tends to focus more on minimizing $\Psi_\Theta^{\mathbf{I}_1}$ than $\Psi_\Theta^{\mathbf{I}_3}$. While this issue is not particularly severe, Eq. 2.13 can be reformulated as follows to address this concern:

$$\Psi_\Theta(\chi) := \Psi_\Theta^{\mathbf{r}}(\chi^{\mathbf{r}}) + \Psi_\Theta^0(\chi^0) + \Psi_\Theta^b(\chi^b) + \lambda_1\Psi_\Theta^{\mathbf{I}_1} + \lambda_2\Psi_\Theta^{\mathbf{I}_2} + \lambda_3\Psi_\Theta^{\mathbf{I}_3}, \tag{2.17}$$

where $\lambda_1 < \lambda_2 < \lambda_3 << 1$ are non-adaptive weights. However, in this paper, Eq. (2.13) is used as the loss function for the neural network. Figure 2 shows a schematic diagram of a conservative PINN.

PINNs methods have some advantage over traditional methods like Finite difference, Finite valume etc:

- It is possible to solve PDEs over all entire spatial-temporal domains at once.
- It is a meshfree method so solve PDE using irregularly mesh training points on the domain.

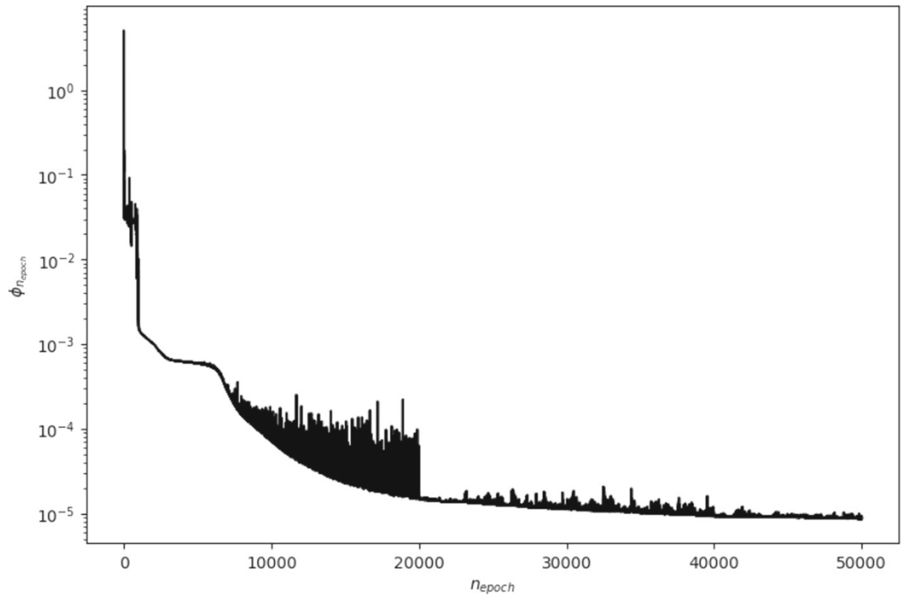


Fig. 3 the value of loss in each epoch

Table 3 Numerical results for Example 1 with $\eta = 1$, $\xi = 3$, $\rho = 4$, $\mathbf{x}_0 = 30$, $\nu = 1/32$, $\Delta\mathbf{t} = 0.001$ and $T = 1$

	Present method		Method in [55]	
N	L_2	L_∞	L_2	L_∞
40	0.00227	0.00035	0.01963	0.01135
80	0.00155	0.00021	0.01994	0.01273
160	0.00187	0.00032	0.00111	0.00144

Table 4 Comparison of results for Example 1 on [0, 80] with $\eta = 1$, $\xi = 3$, $\rho = 2$, $\mathbf{x}_0 = 32$, $\nu = 1/32$ and $T = 1$

Article	I_1	I_2	I_3	L_2	L_∞
Present	0.78521	0.16665	0.00520	1.7229×10^{-4}	1.5427×10^{-4}
[22]	0.78528	0.16658	0.00520	1.5695×10^{-4}	2.0214×10^{-4}
[62]	0.78466	0.16666	0.00519	1.9588×10^{-4}	1.7443×10^{-4}
[35]	0.78539	0.16666	0.00520	7.8337×10^{-5}	4.4485×10^{-5}
[64]	0.78539	0.16666	0.00520	2.5017×10^{-6}	2.7551×10^{-6}

Table 5 Comparison of results for Example 1 on [0, 80] with $\eta = 1$, $\xi = 3$, $\rho = 4$, $\mathbf{x}_0 = 32$, $\nu = 0.2$ and $T = 1$

Article	I_1	I_2	I_3	L_2	L_∞
Present	2.62277	2.35527	0.78547	2.74448×10^{-3}	5.31315×10^{-4}
[55]	2.62205	2.35582	0.78502	7.89400×10^{-3}	5.60170×10^{-3}
[64]	2.62206	2.35615	0.78534	2.30500×10^{-3}	1.88282×10^{-3}
[35]	2.63278	2.37300	0.80233	8.90620×10^{-3}	8.21994×10^{-3}
[9]	2.62209	2.35989	0.78547	1.96050×10^{-3}	1.33420×10^{-3}

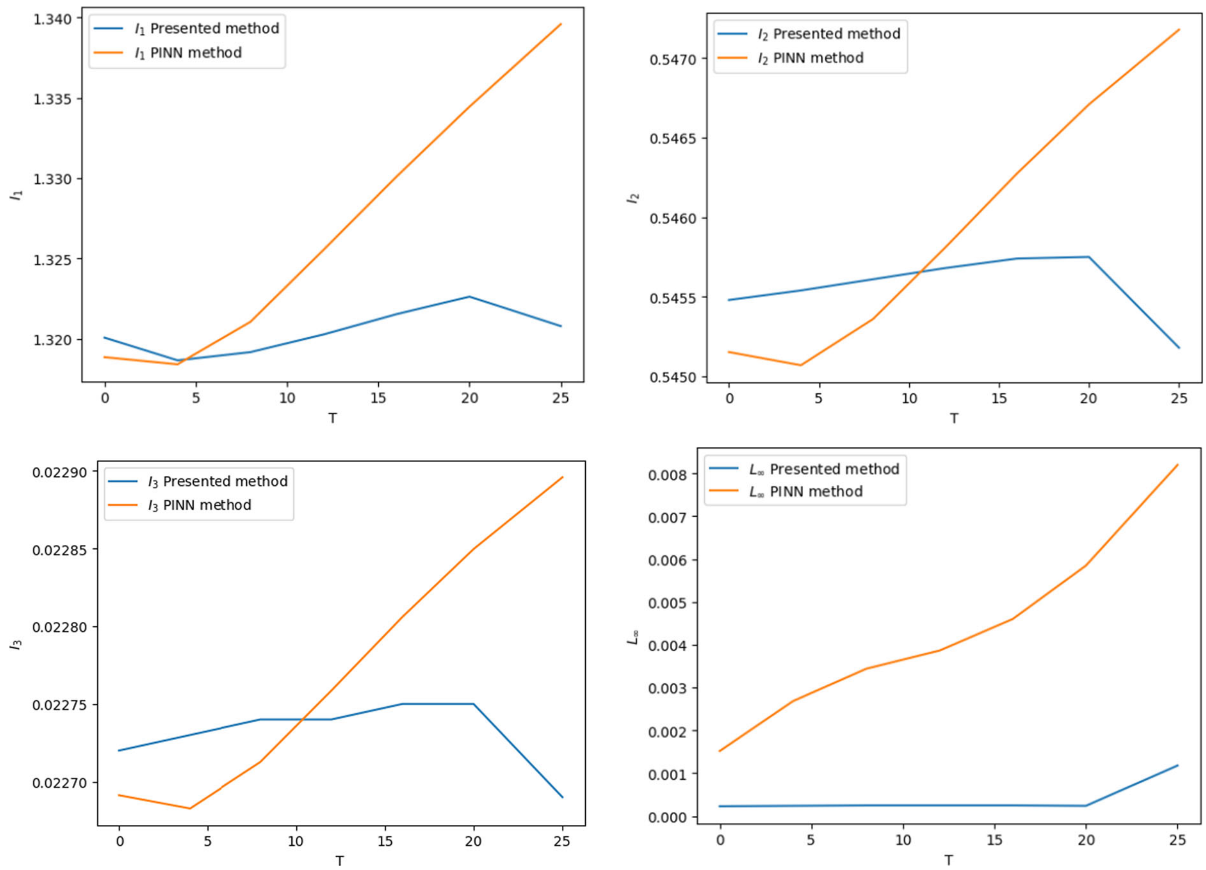


Fig. 4 Value of I_1, I_2, I_3 and L_∞ with $\nu = 1/32$, $\rho = 3$, $\mathbf{x}_0 = 32$, $\eta = 1$ and $\xi = 3$ for Example 1

- Also, solution of PINNs is continuous on domain.

On the other side we forced PINN to keep mass, momentum, and energy conservation in conservative based version of PINN by adding new terms in loss function.

3 Numerical analysis

In this section, we evaluate the efficacy of the proposed method in solving the primary mathematical model. The simulations and parameters recovery are conducted utilizing Anaconda(Jupyter Notebook) software on an Intel Core i9 machine equipped with 64 GB of memory and 24GB 3090 of GPU. We assess the efficacy of the recent numerical method through multiple examples, evaluating its performance using both L_∞ and L_2 error metrics

$$L_2 = ||\mathbf{u}^{exact} - \mathbf{u}^{approximated}||_2, \tag{3.1}$$

$$L_\infty = ||\mathbf{u}^{exact} - \mathbf{u}^{approximated}||_\infty, \tag{3.2}$$

Additionally, we verify the conservation of three attributes:

- **Conservation of Mass:**

$$\mathbf{I}_1 = \int_{\mathbf{x}_{min}}^{\mathbf{x}_{max}} \mathbf{u}(\mathbf{x}, t) d\mathbf{x}, \tag{3.3}$$

- **Conservation of Momentum:**

$$\mathbf{I}_2 = \int_{\mathbf{x}_{min}}^{\mathbf{x}_{max}} [\mathbf{u}^2(\mathbf{x}, t) + \eta \mathbf{u}_{\mathbf{x}}^2(\mathbf{x}, t)] d\mathbf{x}, \tag{3.4}$$

- **Conservation of Energy:**

$$\mathbf{I}_3 = \int_{\mathbf{x}_{min}}^{\mathbf{x}_{max}} \mathbf{u}(\mathbf{x}, t)^{\rho+2} d\mathbf{x}, \tag{3.5}$$

The integrals are approximated with the trapezoidal rule by using tfp.math.trapz in Tensorflow probability.

Example 3.1 **Single solitary wave motion** We consider Eq. (1.1), the following initial condition [26]:

$$\mathbf{u}(\mathbf{x}, 0) = \sqrt[\rho]{\frac{\nu(\rho+1)(\rho+2)}{2\xi}} sech^2\left(\frac{\rho}{2\sqrt{\eta}}(\mathbf{x} - \mathbf{x}_0)\right),$$

Table 6 Numerical results for Example 1 with $\eta = 1$, $\xi = 3$, $\mathbf{x}_0 = 32$ and $\nu = 1/32$

	T	$\mathbf{I}_1$	$\mathbf{I}_2$	$\mathbf{I}_3$	L_∞
$\rho = 2$	0	0.78521	0.16665	0.00520	0.00015
	4	0.78236	0.16685	0.00521	0.00047
	8	0.77755	0.16690	0.00522	0.00060
	12	0.77644	0.16693	0.00522	0.00064
	16	0.77636	0.16692	0.00522	0.00080
	20	0.77513	0.16680	0.00521	0.00105
	25	0.77422	0.16645	0.00519	0.00147
$\rho = 3$	0	1.32008	0.54548	0.02272	0.00023
	4	1.31867	0.54554	0.02273	0.00024
	8	1.31918	0.54561	0.02274	0.00025
	12	1.32028	0.54568	0.02274	0.00025
	16	1.32154	0.54574	0.02275	0.00025
	20	1.32263	0.54575	0.02275	0.00024
	25	1.32080	0.54518	0.02269	0.00118
$\rho = 10$	0	2.18214	1.99409	0.22833	0.00040
	4	2.17916	1.99059	0.22595	0.00127
	8	2.18019	1.98963	0.22556	0.00249
	12	2.18824	1.99005	0.22590	0.00363
	16	2.19127	1.98981	0.22548	0.00466
	20	2.18734	1.98858	0.22453	0.00598
	25	2.18926	1.97404	0.21408	0.02957

$$(3.6)$$

The exact solution is

$$\mathbf{u}(\mathbf{x},\mathbf{t}) = \sqrt[\rho]{\frac{\nu(\rho+1)(\rho+2)}{2\xi} sech^2\left(\frac{\rho}{2\sqrt{\eta}}(\mathbf{x}-\nu\mathbf{t}-\mathbf{x}_0)\right)}, \tag{3.7}$$

Initially, we share the L_2 and L_∞ results obtained by the PINN method and the method introduced by [55] in Table 3. In Tables 4 and 5, we have compared the results of the proposed method with those of other approaches presented in the existing literature, detailing the values of I_1, I_2, I_3, as well as the L_2 norm and L_∞ norm errors. Additionally, Fig. 4 provides a comparison of the conservation laws of momentum, mass, and energy, along with the error of approximate solutions between the two methods, namely the simple PINN and the proposed method. The parameters utilized for these computations, as outlined in Table 3, include $\eta = 1$, $\xi = 3$, $\rho = 4$, $\mathbf{x}_0 = 30$, $\nu = 1/32$, $\Delta\mathbf{t} = 0.001$, and final time $T = 1$ within the computational domain $\mathbf{D} = [0, 80]$. In Table 6, we further explore the impact of different parameters on mass, momentum, and energy, with $\eta = 1$, $\xi = 3$, $\mathbf{x}_0 = 32$, and $\nu = 1/32$. Various values of ρ are considered at different times. Meanwhile, Fig. 5 provides a comparison between our approximation and exact solution graphs for Example 1, considering $\nu = 0.5$, $\rho = 2$, $\mathbf{x}_0 = 32$, $\eta = 1$, and $\xi = 2$. In Fig. 6, the approximation and exact solution graphs are presented for Example 1 with $\nu = 1/32$, $\rho = 5$, $\mathbf{x}_0 = 32$, $\eta = 1$, and $\xi = 3$. Lastly, Fig. 7 showcases the plot of approximate and exact solutions for Example 1, featuring $\nu = 1/32$, $\rho = 10$, $\mathbf{x}_0 = 32$, $\eta = 1$, and $\xi = 3$. In this example, we face a stiff problem, and as ρ increases, so does its stiffness. As we can see from the graphs that this method has been able to accurately approximate this problem with a different value of ρ. According to the results in the tables, we can see that although we have chosen small time and space steps in the discretization, but it has been able to maintain the constancy to a good extent.

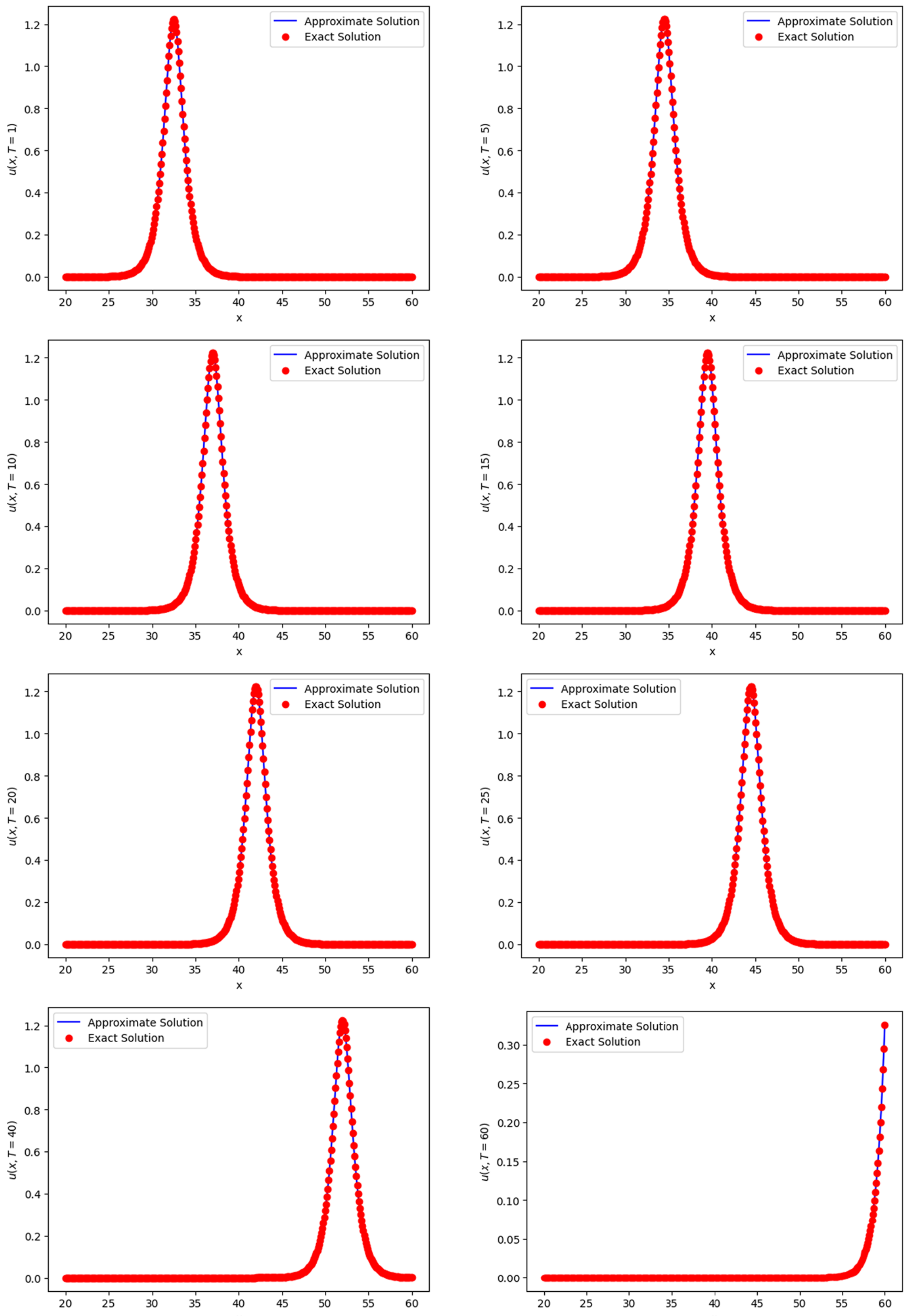


Fig. 5 Approximate and exact solutions with $\nu = 0.5$, $\rho = 2$, $\mathbf{x}_0 = 32$, $\eta = 1$ and $\xi = 2$ for Example 1

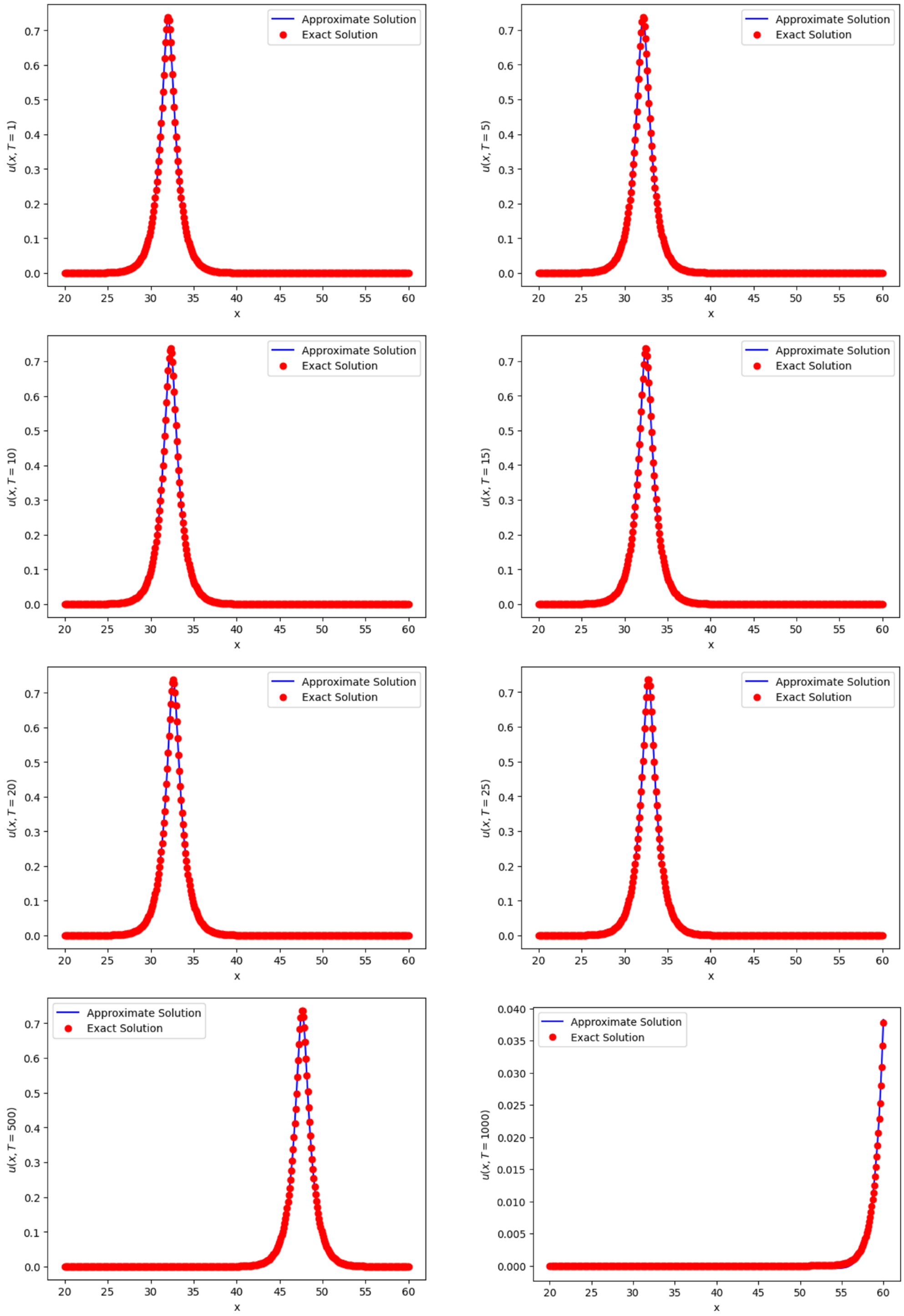


Fig. 6 Approximate and exact solutions with $\nu = 1/32$, $\rho = 5$, $\mathbf{x}_0 = 32$, $\eta = 1$ and $\xi = 3$ for Example 1

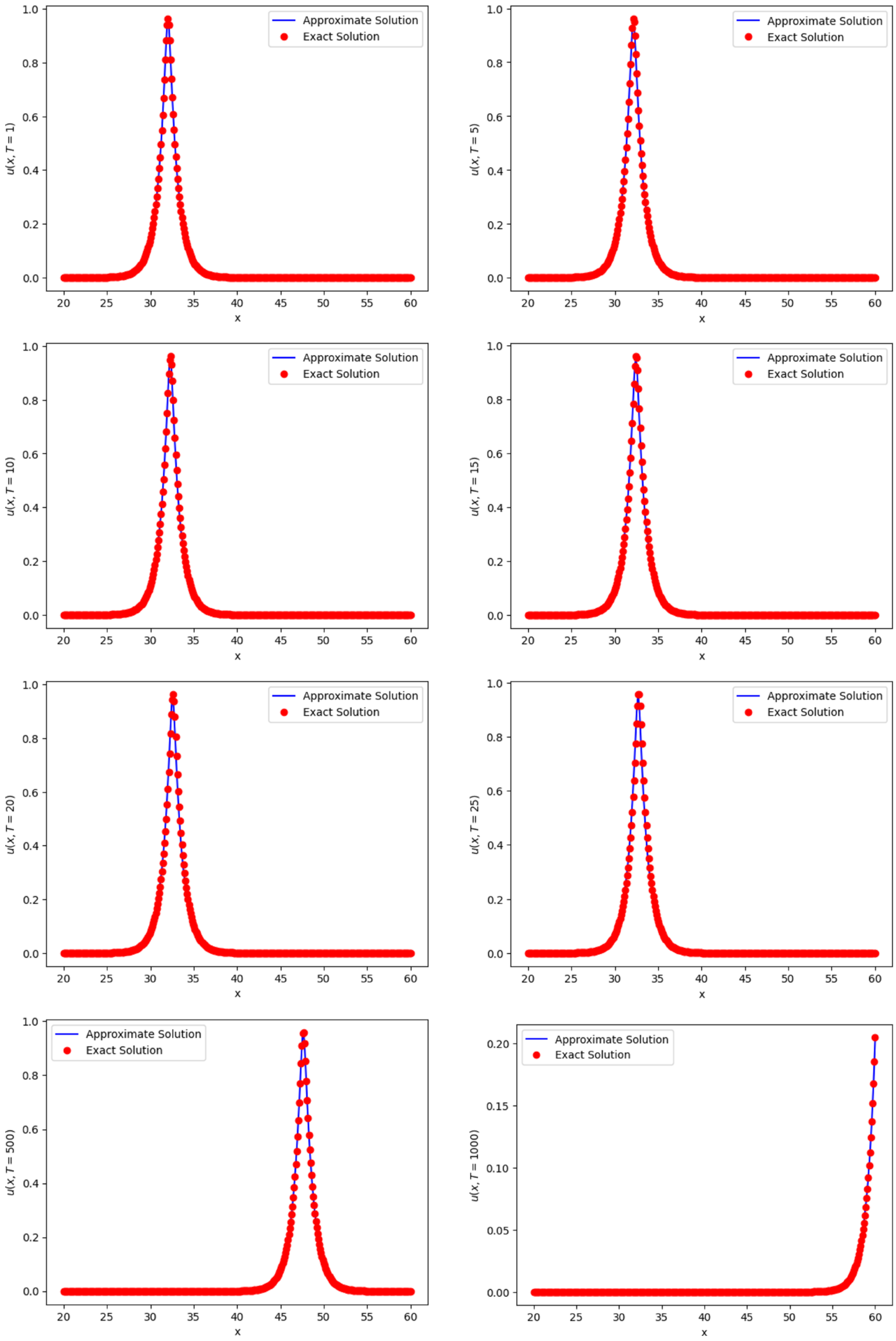


Fig. 7 Approximate and exact solutions with $\nu = 1/32$, $\rho = 10$, $\mathbf{x}_0 = 32$, $\eta = 1$ and $\xi = 3$ for Example 1

Table 7 Numerical results for Example 2 with $N = 200$, $\Delta\mathbf{t} = 0.2$, $\eta = 1$, $\xi = 3$, $\mathbf{x}_1 = 15$, $\mathbf{x}_1 = 30$, $\nu_1 = 0.5$ and $\nu_2 = 0.125$

	T	$\mathbf{I}_1$	$\mathbf{I}_2$	$\mathbf{I}_3$	L_∞	L_2
$\rho = 2$	0	4.7127	3.3332	1.4166	0.0001134	0.00049
	4	4.7127	3.3300	1.4140	0.0010888	0.00311
	8	4.7158	3.3295	1.4130	0.0024816	0.00591
	12	4.7205	3.3301	1.4133	0.0028067	0.00784
	16	4.7169	3.3288	1.4127	0.0034397	0.00948
	20	4.7153	3.3249	1.4086	0.0059529	0.01443
$\rho = 3$	0	5.4178	4.8357	2.5341	0.000801	0.00129
	4	5.4170	4.8275	2.5233	0.001667	0.00643
	8	5.4031	4.8241	2.5220	0.003461	0.00862
	12	5.3932	4.8215	2.5193	0.006804	0.00994
	16	5.3911	4.8189	2.5141	0.014854	0.03031
	20	5.3808	4.8101	2.5101	0.075431	0.09914

Table 8 Numerical results for Example 2 with $N = 200$, $\Delta\mathbf{t} = 0.2$, $\eta = 1$,$\xi = 3$, $\mathbf{x}_1 = 15$, $\mathbf{x}_1 = 30$, $\nu_1 = 0.3$ and $\nu_2 = 0.0375$

	T	$\mathbf{I}_1$	$\mathbf{I}_2$	$\mathbf{I}_3$	L_∞	y L_2
$\rho = 2$	0	3.2957	1.7970	0.4892	0.0000763	0.00032
	4	3.2856	1.7954	0.4801	0.0009853	0.00418
	8	3.2804	1.7902	0.4784	0.0027298	0.00760
	12	3.2809	1.7896	0.4730	0.0047965	0.00960
	16	3.2794	1.7899	0.4714	0.0057248	0.01197
	20	3.2725	1.7864	0.4605	0.0055085	0.01346
$\rho = 3$	0	4.2066	3.0800	1.0164	0.0000763	0.00032
	4	4.2085	3.0812	1.0185	0.0009853	0.00424
	8	4.2150	3.0834	1.0204	0.0027298	0.00694
	12	4.2210	3.0849	1.0210	0.0047965	0.00856
	16	4.2251	3.0844	1.0194	0.0057248	0.01469
	20	4.2262	3.0793	1.0146	0.0055085	0.01537

Example 3.2 **The interaction of two solitary waves:** We consider Eq (1.1) with the following initial condition [55]:

$$u(\mathbf{x}, 0) = \sqrt[\rho]{\frac{\nu_1(\rho+1)(\rho+2)}{2\xi} sech^2\left(\frac{\rho}{2\sqrt{\eta}}(\mathbf{x}-\mathbf{x}_1)\right)} + \sqrt[\rho]{\frac{\nu_2(\rho+1)(\rho+2)}{2\xi} sech^2\left(\frac{\rho}{2\sqrt{\eta}}(\mathbf{x}-\mathbf{x}_2)\right)}. \tag{3.8}$$

The exact solution for this problem is:

$$u(\mathbf{x}, t) = \sqrt[\rho]{\frac{\nu_1(\rho+1)(\rho+2)}{2\xi} sech^2\left(\frac{\rho}{2\sqrt{\eta}}(\mathbf{x}-\nu_1 t-\mathbf{x}_1)\right)} + \sqrt[\rho]{\frac{\nu_2(\rho+1)(\rho+2)}{2\xi} sech^2\left(\frac{\rho}{2\sqrt{\eta}}(\mathbf{x}-\nu_2 t-\mathbf{x}_2)\right)}, \tag{3.9}$$

L_2, L_∞ errors, and the values of $\mathbf{I}_1$, $\mathbf{I}_2$, and $\mathbf{I}_3$ are reported in Table 7 for $N = 200$, $\Delta\mathbf{t} = 0.2$, $\eta = 1$, $\xi = 3$, $\mathbf{x}_1 = 15$, $\mathbf{x}_2 = 30$, $\nu_1 = 0.5$, and $\nu_2 = 0.125$. Subsequently, results for L_2, L_∞ errors, $\mathbf{I}_1$, $\mathbf{I}_2$, and $\mathbf{I}_3$ are reported in Table 8 for $N = 200$, $\Delta\mathbf{t} = 0.2$, $\eta = 1$, $\xi = 3$, $\mathbf{x}_1 = 15$, $\mathbf{x}_2 = 30$, $\nu_1 = 0.3$, and $\nu_2 = 0.0375$.

In Fig. 8, we show approximate and exact solutions with $\nu_1 = 0.5$, $\nu_2 = 0.125$, $\rho = 2$, $\mathbf{x}_1 = 32$, $\mathbf{x}_2 = 25$, $\eta = 1$, and $\xi = 3$ as well as for $\nu_1 = 0.5, \nu_2 =$

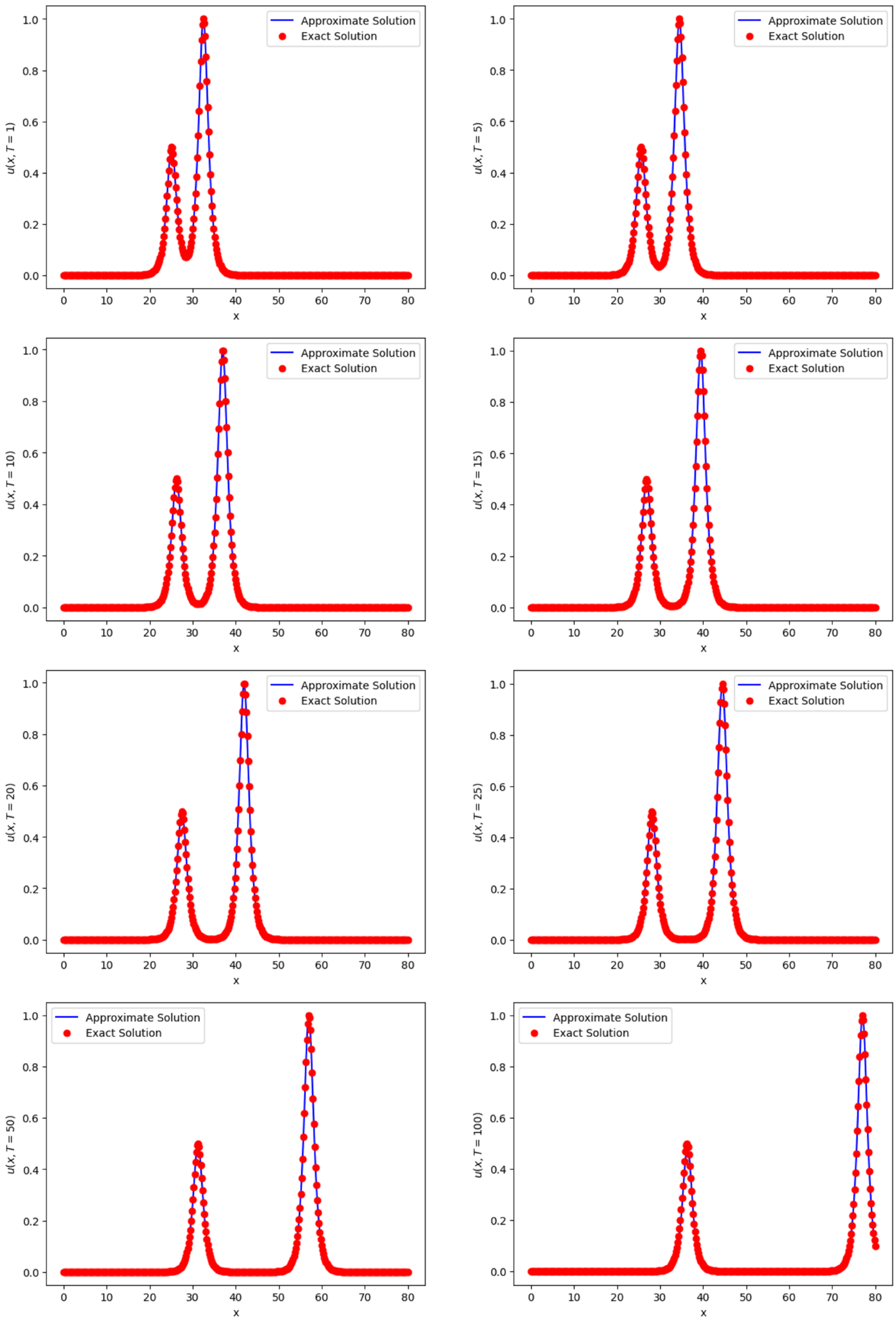


Fig. 8 Approximate and exact solutions with $\nu_1 = 0.5$, $\nu_2 = 0.125$, $\rho = 2$, $\mathbf{x}_1 = 32$, $\mathbf{x}_2 = 25$, $\eta = 1$ and $\xi = 3$ for Example 2

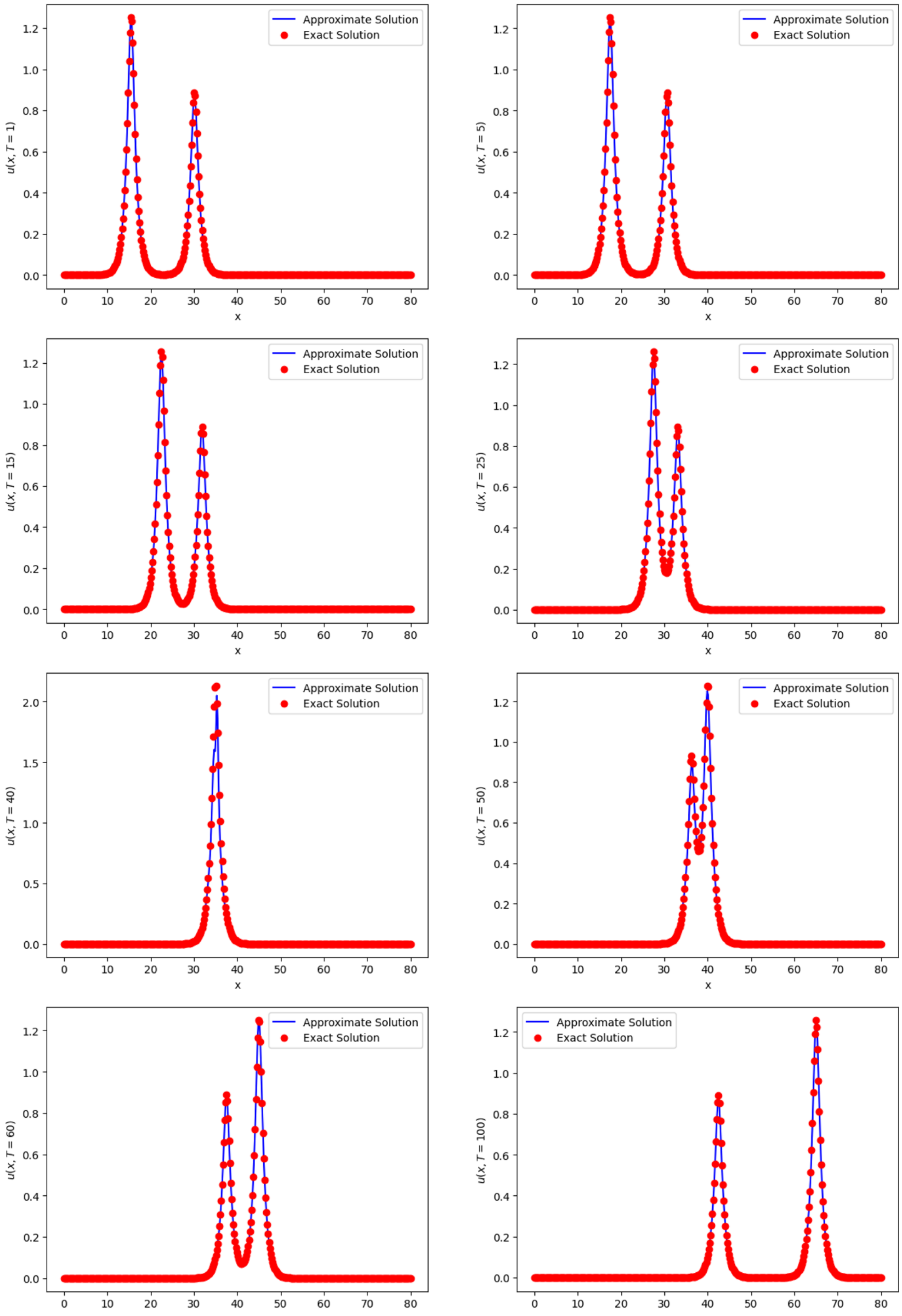


Fig. 9 Approximate and exact solutions with $\nu_1 = 0.5$, $\nu_2 = 0.125$, $\rho = 4$, $\mathbf{x}_1 = 15$, $\mathbf{x}_2 = 30$, $\eta = 1$ and $\xi = 3$ for Example 2

Table 9 Numerical results for Example 3 with $N = 200$, $\Delta\mathbf{t} = 0.2$, $\eta = 0.1$ and $\xi = 3$

	T	Present method $\mathbf{I}_1$	$\mathbf{I}_2$	$\mathbf{I}_3$	Method in [1] $\mathbf{I}_1$	$\mathbf{I}_2$	$\mathbf{I}_3$
$\rho = 2$	0	1.77081	1.37540	0.88379	1.77245	1.37864	0.88622
	4	1.72156	1.31660	0.87048	1.77245	1.37865	0.88623
	8	1.72754	1.28984	0.86384	1.77245	1.37865	0.88623
	12	1.72369	1.28300	0.85900	1.77245	1.37865	0.88623
$\rho = 3$	0	1.76917	1.37391	0.78855	1.77245	1.37864	0.79266
	4	1.68743	1.27782	0.68135	1.77245	1.37865	0.79267
	8	1.63640	1.22320	0.64510	1.77245	1.37865	0.79267
	12	1.65520	1.20922	0.63437	1.77245	1.37865	0.79267

Table 10 Numerical results for Example 3 with $N = 200$, $\Delta\mathbf{t} = 0.2$, $\eta = 0.05$ and $\xi = 3$

	T	$\mathbf{I}_1$	$\mathbf{I}_2$	$\mathbf{I}_3$
$\rho = 2$	0	1.77242	1.31437	0.88493
	4	1.76411	1.29539	0.86655
	8	1.75039	1.26075	0.85927
	12	1.74939	1.25998	0.83574
$\rho = 3$	0	1.77245	1.31593	0.79264
	4	1.17632	1.28965	0.78172
	8	1.75432	1.26354	0.77672
	12	1.74455	1.26021	0.75032

0.125, $\rho = 4$, $\mathbf{x}_1 = 15$, $\mathbf{x}_2 = 30$, $\eta = 1$ and $\xi = 3$ in Fig. 9.

In this example, according to the plots we conclude that this method is useful for simulation of the interaction of two solitary waves which is a stiff problem. Also we can see it maintains the constancy well according to the tables.

Example 3.3 **Maxwellian initial condition** We consider Eq. (1.1), the following initial condition [40,80]

$$u(x, 0) = e^{-(\mathbf{x}-20)^2}, \qquad \mathbf{x} \in [0, 40], \tag{3.10}$$

The L_2, L_∞ errors, and the values of $\mathbf{I}_1$, $\mathbf{I}_2$, and $\mathbf{I}_3$ are presented in Table 9 with parameters set to $N = 200$, $\Delta\mathbf{t} = 0.2$, $\eta = 0.1$, and $\xi = 3$ and also for $N = 200$, $\Delta\mathbf{t} = 0.2$, $\eta = 0.05$ and $\xi = 3$ in Table 10. Additionally, Fig. 10 illustrates our approximate solution for the Maxwellian example. The corresponding parameters for this figure are $N = 100$, $\rho = 2$, $\eta = 0.05$, and $\xi = 3$. Furthermore, Fig. 11 depicts the same solution with parameters set to $N = 100$, $\rho = 3$, $\eta = 0.01$, and $\xi = 3$.

Example 3.4 **Parameter estimation for Single solitary wave motion** We assume Example 1 again, but this time we consider our PDE has an unknown parameter. At first we assume ξ is unknown with its exact value $\xi = 1.651710$ and with the other parameters given by $\nu = 0.5$, $\rho = 2$, $\eta = 1$ and $\mathbf{x}_0 = 30$ then approximated solution from the inverse PINN with 20000 epochs shown Fig. 12 is 1.650976. Now, consider $\xi \in [1, 4]$ for 15 times randomly then generated 20000 data in the problem domain $\mathbf{D} = [0, 80]$ uniformly for each one so that its parameters are $\nu = 0.5$, $\rho = 2$, $\eta = 1$ and $\mathbf{x}_0 = 30$ and then recovery ξ in Fig. 13 also MSE and R^2score are 1.793208×10^{-7} and 0.9999997 respectively as well as for parameters $\nu = 0.5$, $\rho = 4$, $\eta = 1$ and $\mathbf{x}_0 = 30$ in Fig. 14 with MSE: 4.366964×10^{-6} and R^2score: 0.9999932. Figure 15 estimated η which is in $[1, 4]$ with $\nu = 0.5$, $\rho = 2$, $\xi = 3$ and $\mathbf{x}_0 = 30$ as well as MSE and R^2score are 2.751684×10^{-6} and 0.9999964 respectively. also in Fig. 16 estimated η with $\nu = 0.5$, $\rho = 4$, $\xi = 3$ and $\mathbf{x}_0 = 30$ as

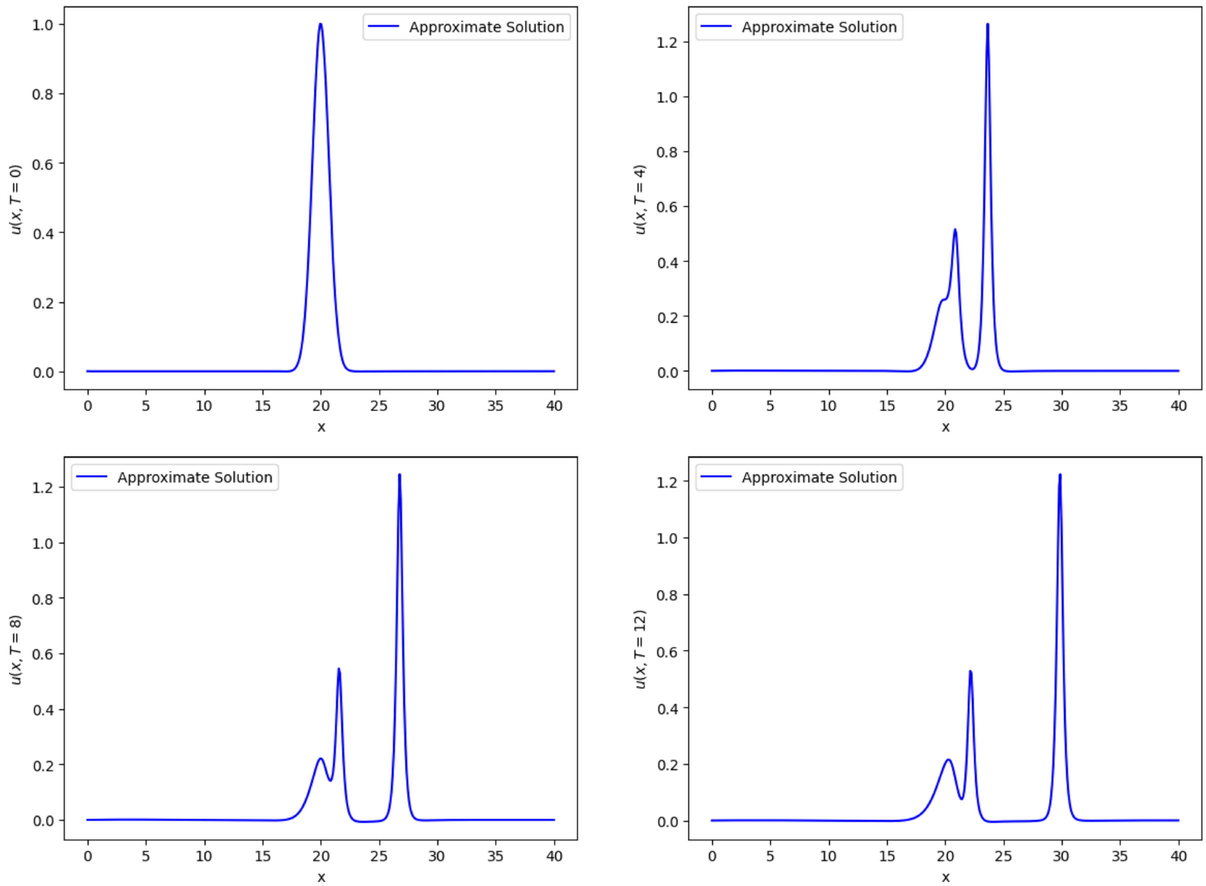


Fig. 10 Approximate solutions with $N = 100$, $\rho = 2$, $\eta = 0.05$ and $\xi = 3$ for Example 3

well as MSE and R^2score are 2.989937×10^{-6} and 0.9999937 respectively.

Example 3.5 **Parameter estimation for the interaction of two solitary waves:** We assume Example 2 again, but this time consider our PDE has an unknown parameter. At first assume ξ is unknown with the exact value drawn randomly $\xi \in [1, 4]$ for 15 times then we generate 20000 data in the problem domain $\mathbf{D} = [0, 80]$ for each ξ so that its parameters are $\nu_1 = 0.2, \nu_2 = 1/80, \rho = 2, \eta = 1, \mathbf{x}_1 = 15$ and $\mathbf{x}_2 = 30$ and then recovery ξ in Fig. 17 also MSE and R^2score are 1.437894×10^{-5} and 0.9999757 respectively. also for parameters $\nu_1 = 0.2, \nu_2 = 1/80, \rho = 4, \eta = 1, \mathbf{x}_1 = 15$ and $\mathbf{x}_2 = 30$ in Fig. 18 with MSE: 8.968579×10^{-6} and R^2score: 0.999982. Figure 19 estimated η which is in $[1, 4]$ with $\nu_1 = 0.2, \nu_2 = 1/80, \rho = 2, \xi = 3, \mathbf{x}_1 = 15$ and $\mathbf{x}_2 = 30$ as well as MSE and R^2score are 3.4988931×10^{-4} and 0.999465 respectively. Also for parameters $\nu_1 = 0.2, \nu_2 = 1/80, \rho = 4, \xi = 3, \mathbf{x}_1 = 15$ and $\mathbf{x}_2 = 30$ in Fig. 20 with MSE: 1.5514632×10^{-4} and R^2score: 0.999806.

4 Conclusion

We introduced, a novel approach for utilizing Physics-Informed Neural Networks (PINN), specifically tailored for the numerical simulation of the Generalized Equal Width Equation (GEW), a challenging partial differential equation known for its stiffness. The method incorporates three conservation conditions into the loss function of a conventional PINN. The outcome is a continuous nonlinear function, representing an approximate solution to our Partial Differential Equation (PDE), within a neural network framework. This function is derived from the trained weights of a feedforward neural network in our model. Upon

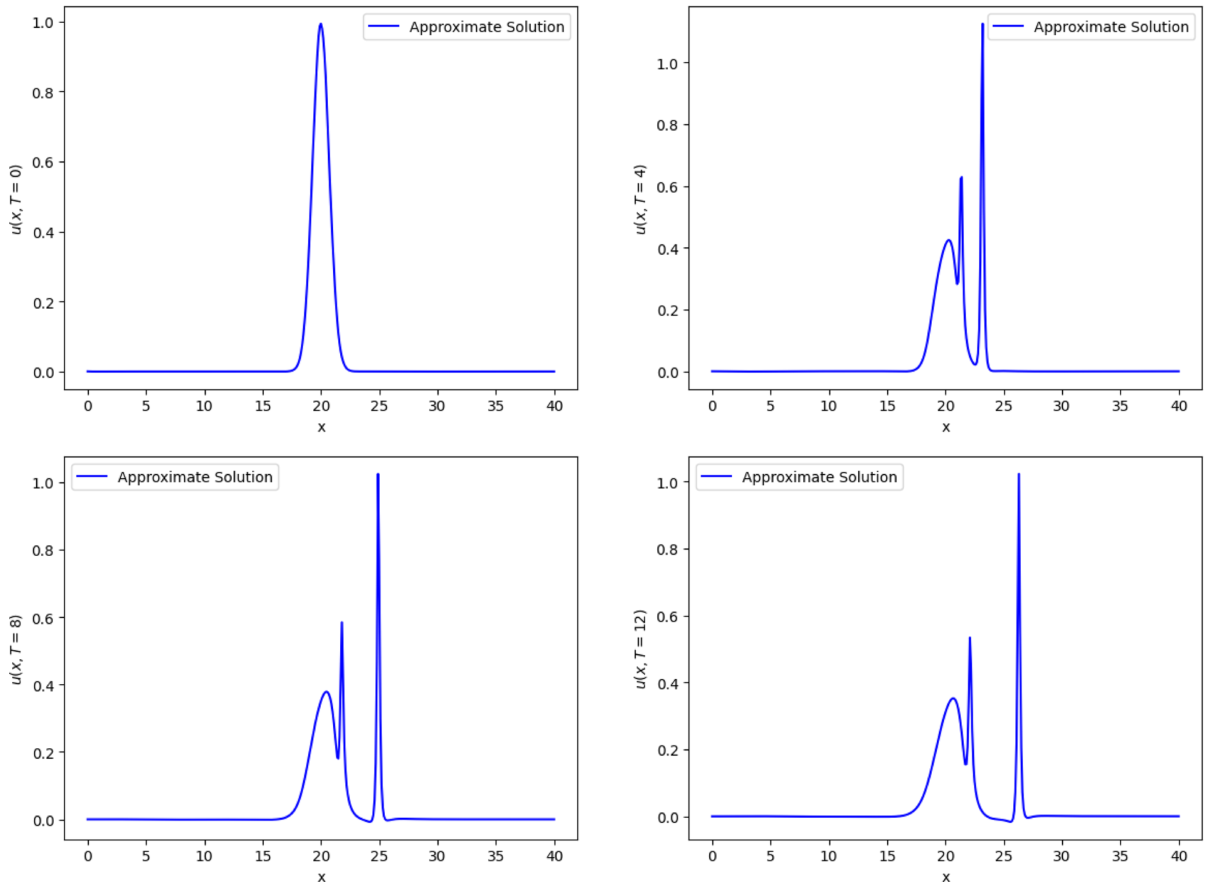


Fig. 11 Approximate solutions with $N = 100$, $\rho = 3$, $\eta = 0.01$ and $\xi = 3$ for Example 3

Fig. 12 Loss function of parameter estimation for ξ with $\nu = 0.5$, $\rho = 2$, $\eta = 1$ and $\mathbf{x}_0 = 30$ in each epoch

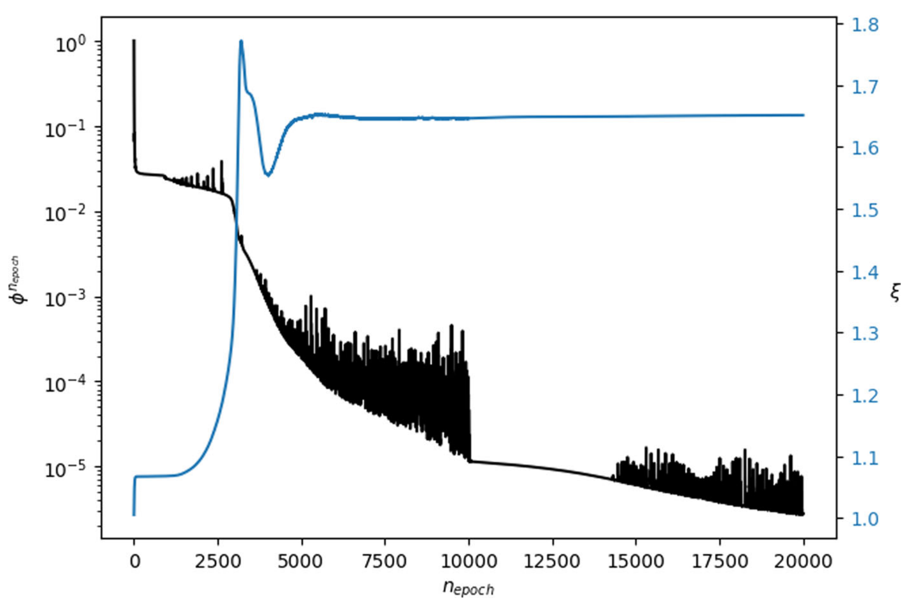


Fig. 13 Parameter estimation for ξ with $\nu = 0.5$, $\rho = 2$, $\eta = 1$ and $\mathbf{x}_0 = 30$

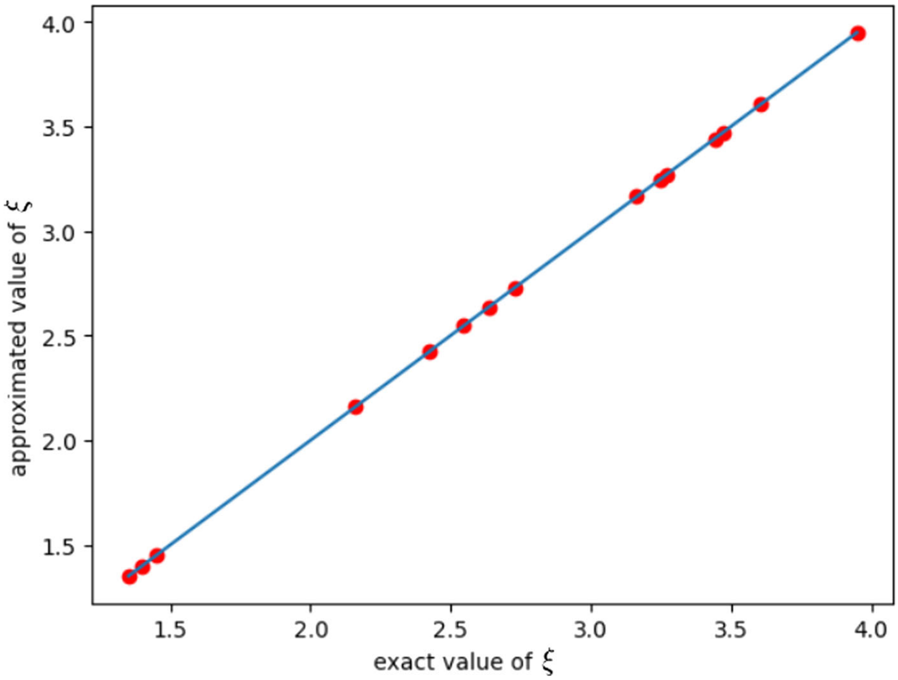
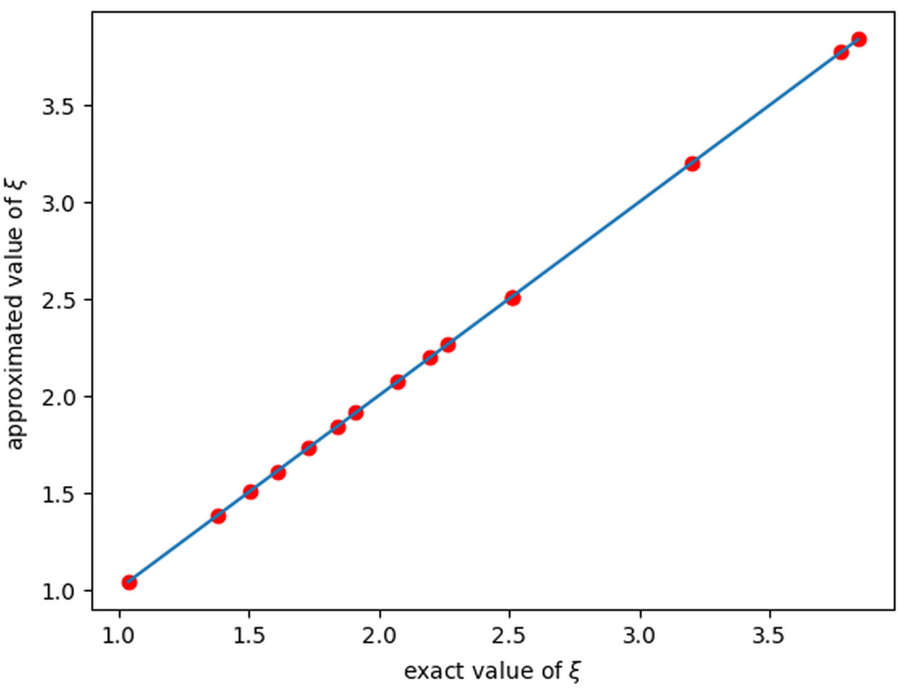


Fig. 14 Parameter estimation for ξ with $\nu = 0.5$, $\rho = 4$, $\eta = 1$ and $\mathbf{x}_0 = 30$



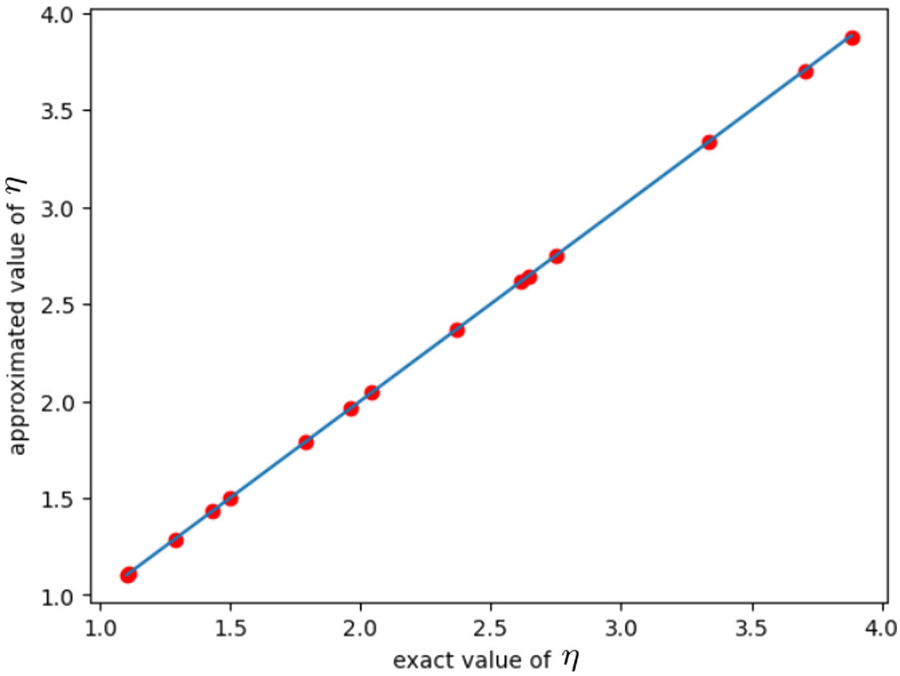


Fig. 15 Parameter estimation for η with $\nu = 0.5$, $\rho = 2$, $\xi = 3$ and $\mathbf{x}_0 = 30$

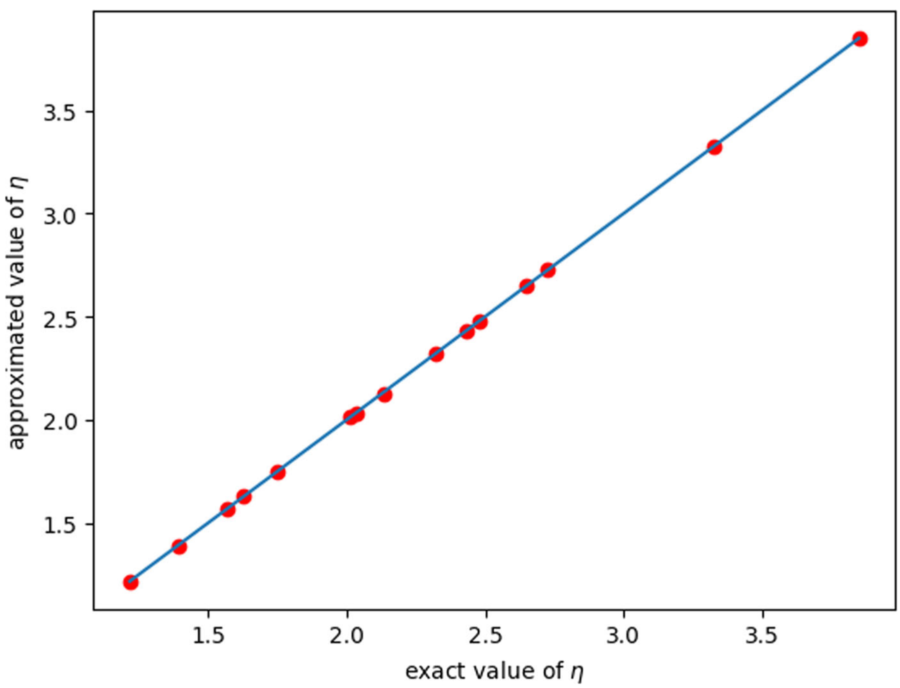


Fig. 16 Parameter estimation for η with $\nu = 0.5$, $\rho = 4$, $\xi = 3$ and $\mathbf{x}_0 = 30$

completion of the training process, we obtain reasonably accurate solutions and the proposed method outperforms the PINN approach. It surpasses alternative methods by requiring fewer data points in the domain. Furthermore, the assessment of mass, momentum, and energy conservation throughout the simulation underscores the superiority of this approach. On the other side, with the assumption that there was an unknown

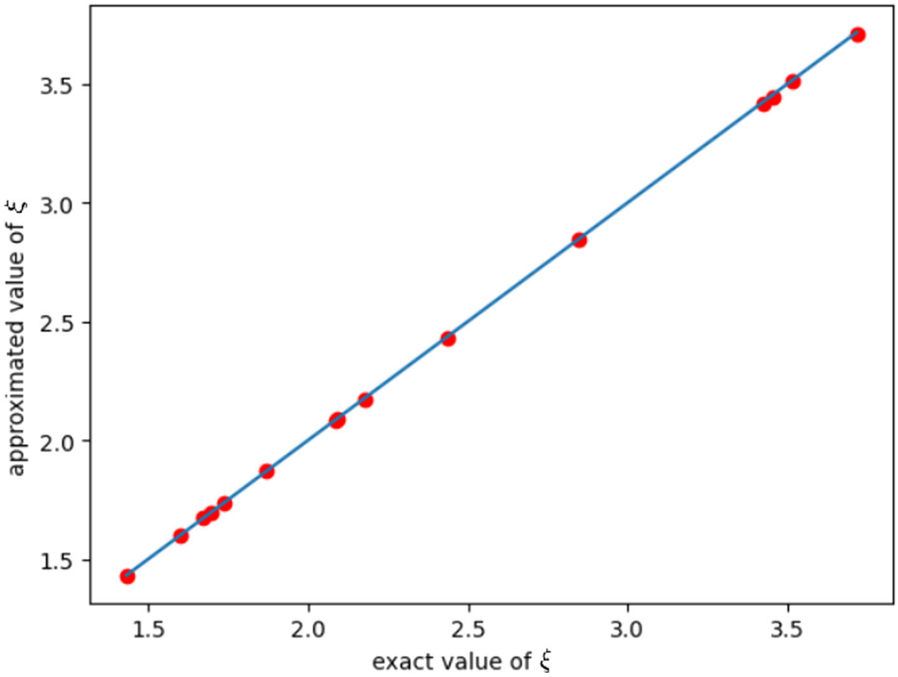


Fig. 17 Parameter estimation for ξ with $\nu_1 = 0.2$, $\nu_2 = 1/80$, $\rho = 2$, $\eta = 1$, $\mathbf{x}_1 = 15$ and $\mathbf{x}_2 = 30$

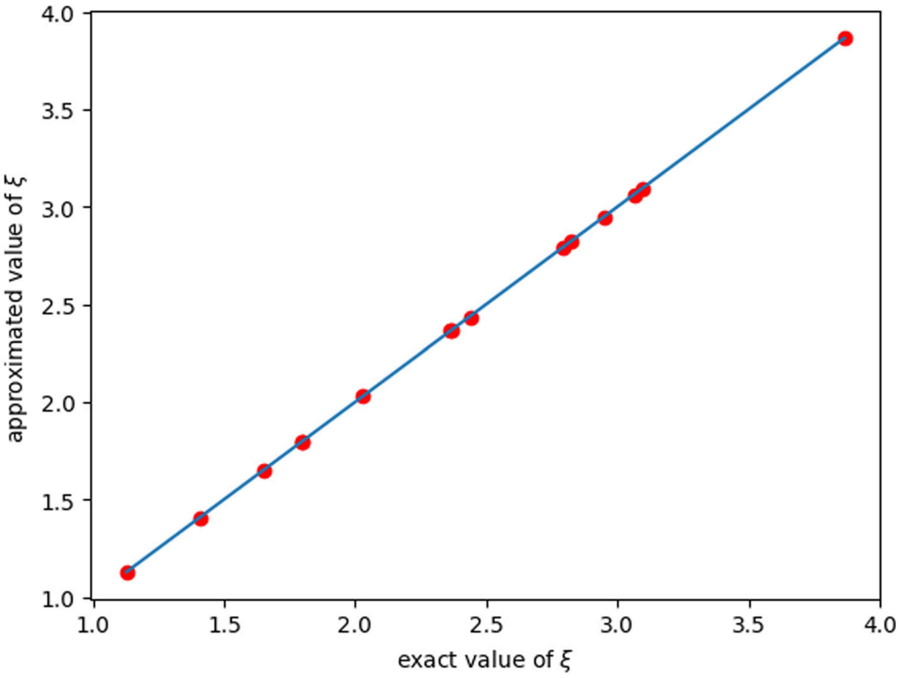


Fig. 18 Parameter estimation for ξ with $\nu_1 = 0.2$, $\nu_2 = 1/80$, $\rho = 4$, $\eta = 1$, $\mathbf{x}_1 = 15$ and $\mathbf{x}_2 = 30$ Example 5

parameter in the GEW model, we used the data at a few points in the domain to estimate parameters with very high accuracy using the inverse PINN approach. Parameter estimate and an approximate solution on the domain will be obtained simultaneously in the data-driven stage.

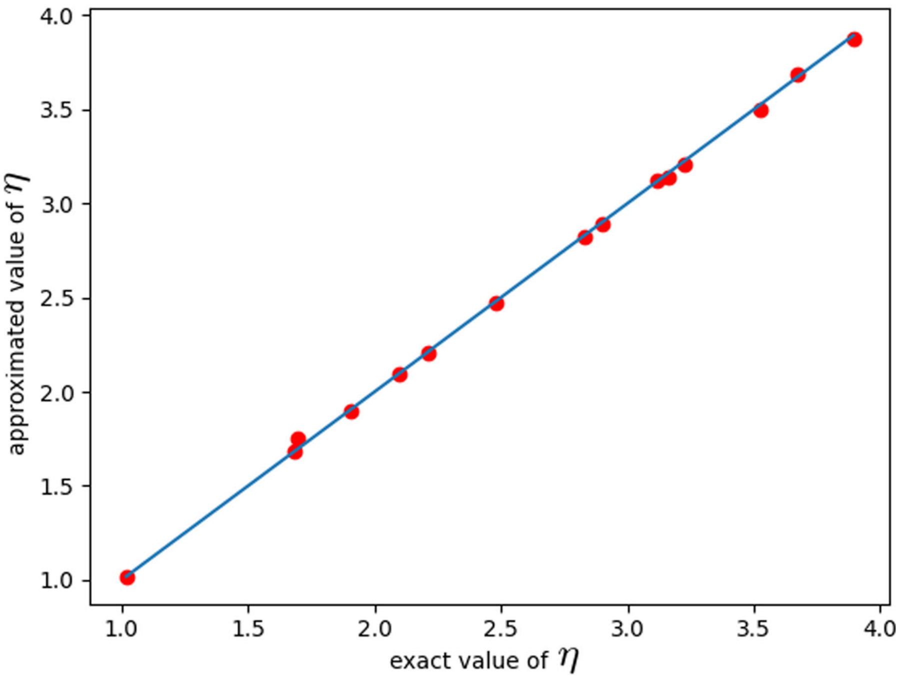


Fig. 19 Parameter estimation for η with $\nu_1 = 0.2$, $\nu_2 = 1/80$, $\rho = 2$, $\xi = 3$, $\mathbf{x}_1 = 15$ and $\mathbf{x}_2 = 30$

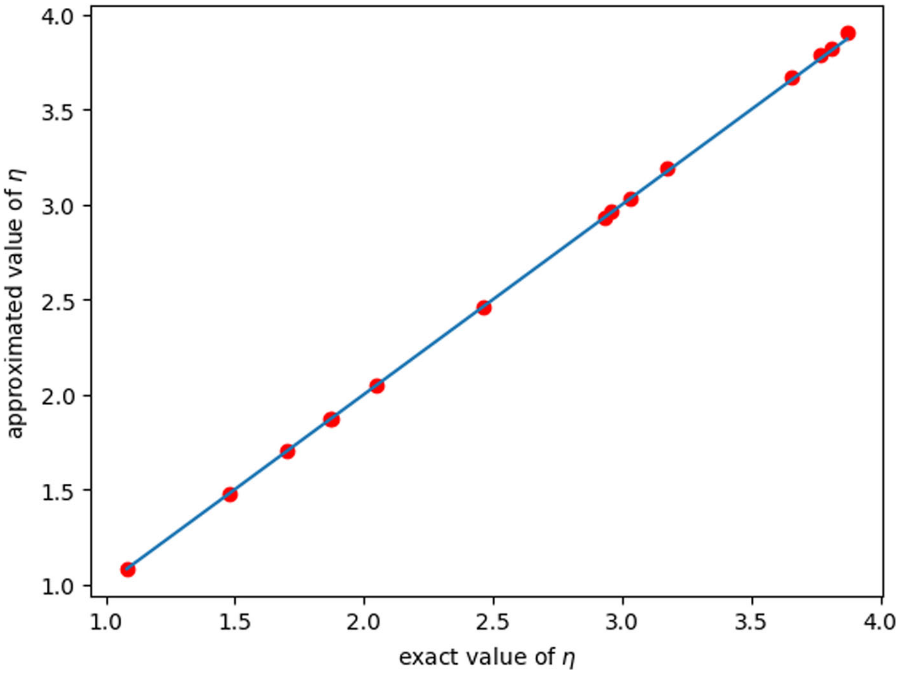


Fig. 20 Parameter estimation for η with $\nu_1 = 0.2$, $\nu_2 = 1/80$, $\rho = 4$, $\xi = 3$, $\mathbf{x}_1 = 15$ and $\mathbf{x}_2 = 30$

Acknowledgements We would like to express our sincere appreciation to the reviewers for their valuable feedback, which greatly contributed to the improvement of our paper.

Funding No funds were used for the research of this article.

Data availibility No Data associated with the manuscript.

Declarations

Conflict of interest The authors declare that they have no conflict of interest to disclose.

Appendix: Spatial-temporal derivative by automatic differentiation

One of the key advantages of the Physics-Informed Neural Networks (PINN) approach is its ability to leverage automatic differentiation. This feature streamlines both the implementation process and the calculation of spatial, temporal, and spatiotemporal derivatives. Since the process is uniform across different cases, for instance, to compute u_{xxt}, the following code would be used:

```
with tf.GradientTape(persistent=True) as tape:
    tape.watch(t)
    tape.watch(x)
    u_x = tape.gradient(u, x)
    u_xx = tape.gradient(u_x, x)
    u_xxt = tape.gradient(u_xx, t)
```

References

1. Abbaszadeh, M., Bayat, M., Dehghan, M.: The local meshless collocation method for numerical simulation of shallow water waves based on generalized equal width (GEW) equation. Wave Motion **107**, 102805 (2021)
2. Abbaszadeh, M., Bagheri Salec, A., Hatim Aal-Ezirej, T.A.-K.: A radial basis function (RBF)-finite difference method for solving improved Boussinesq model with error estimation and description of solitary waves. Numer. Methods Part. Differ. Equ. **40**(3), e23077 (2024)
3. Abbaszadeh, M., Bagheri Salec, A., Jebur, A.S.: Application of compact local integrated RBFs technique to solve fourth-order time-fractional diffusion-wave system. J. Math. Model. **12**(3), 431–449 (2024)
4. Abbaszadeh, M., Salec, A.B., Aal-Ezirej, T.A.-K.H.: Simulation of Ginzburg-Landau equation via rational RBF partition of unity approach. Opt. Quant. Electron. **56**(1), 96 (2024)
5. Abdulloev, K.O., Bogolubsky, I., Makhankov, V.: One more example of inelastic soliton interaction. Phys. Lett. A **56**(6), 427–428 (1976)
6. Arzani, A., Wang, J.-X., D'Souza, R.M.: Uncovering near-wall blood flow from sparse data with physics-informed neural networks. Phys. Fluids **33**(7), 071905 (2021)
7. Baydin, A.G., Pearlmutter, B.A., Radul, A.A., Siskind, J.M.: Automatic differentiation in machine learning: a survey. J. Mach. Learn. Res. **18**, 1–43 (2018)
8. Benjamin, T.B., Bona, J.L., Mahony, J.J.: Model equations for long waves in nonlinear dispersive systems. Philos. Trans. R. Soc. Lond. Ser. A Math. Phys. Sci. **272**(1220), 47–78 (1972)
9. Bhowmik, S.K., Karakoc, S.B.G.: Numerical solutions of the generalized equal width wave equation using the Petrov–Galerkin method. Appl. Anal. **100**(4), 714–734 (2021)
10. Bona, J., Bryant, P.: A mathematical model for long waves generated by wavemakers in non-linear dispersive systems. Math. Proc. Camb. Philos. Soc. **73**(2), 391–405 (1973)
11. Bücker, M.: Automatic Differentiation: Applications, Theory, and Implementations. Springer, Berlin (2006)
12. Cai, S., Mao, Z., Wang, Z., Yin, M., Karniadakis, G.E.: Physics-informed neural networks (PINNs) for fluid mechanics: a review. Acta. Mech. Sin. **37**(12), 1727–1738 (2021)
13. Craig, W., Guyenne, P., Hammack, J., Henderson, D., Sulem, C.: Solitary water wave interactions. Phys. Fluids **18**(5), 057106 (2006)
14. Cui, S., Wang, Z., Han, J., Cui, X., Meng, Q.: A deep learning method for solving high-order nonlinear soliton equations. Commun. Theor. Phys. **74**(7), 075007 (2022)
15. Dhatt, G., Lefrançois, E., Touzot, G.: Finite Element Method. Wiley, Hoboken (2012)
16. Dreyfus, G.: Neural Networks: Methodology and Applications. Springer, Berlin (2005)
17. Dutykh, D., Pelinovsky, E.: Numerical simulation of a solitonic gas in KdV and KdV-BBM equations. Phys. Lett. A **378**(42), 3102–3110 (2014)
18. Dutykh, D., Chhay, M., Fedele, F.: Geometric numerical schemes for the KdV equation. Comput. Math. Math. Phys. **53**, 221–236 (2013)
19. Dutykh, D., Katsaounis, T., Mitsotakis, D.: Finite volume methods for unidirectional dispersive wave models. Int. J. Numer. Meth. Fluids **71**(6), 717–736 (2013)
20. Ebrahimijahan, A., Dehghan, M., Abbaszadeh, M.: Numerical simulation of shallow water waves based on generalized equal width (GEW) equation by compact local integrated radial basis function method combined with adaptive residual subsampling technique. Nonlinear Dyn. **105**(4), 3359–3391 (2021)
21. Eivazi, H., Wang, Y., Vinuesa, R.: Physics-informed deep-learning applications to experimental fluid mechanics. Meas. Sci. Technol. **35**(7), 075303 (2024)
22. Evans, D.J., Raslan, K.: Solitary waves for the generalized equal width (GEW) equation. Int. J. Comput. Math. **82**(4), 445–455 (2005)
23. Gao, H., Sun, L., Wang, J.-X.: Phygeonet: physics-informed geometry-adaptive convolutional neural networks for solving parameterized steady-state PDEs on irregular domain. J. Comput. Phys. **428**, 110079 (2021)
24. GaziKarakoc, S.B., Ali, K.K.: Analytical and computational approaches on solitary wave solutions of the generalized equal width equation. Appl. Math. Comput. **371**, 124933 (2020)
25. Guo, Y., Cao, X., Liu, B., Gao, M.: Solving partial differential equations using deep learning and physical constraints. Appl. Sci. **10**(17), 5917 (2020)
26. Hamdi, S., Enright, W.H., Schiesser, W.E., Gottlieb, J.: Exact solutions of the generalized equal width wave equation. In: International Conference on Computational Science and Its Applications, pp. 725–734. Springer (2003)
27. Hao, Y., Di Leoni, P.C., Marxen, O., Meneveau, C., Karniadakis, G.E., Zaki, T.A.: Instability-wave prediction in hypersonic boundary layers with physics-informed neural operators. J. Comput. Sci. **73**, 102120 (2023)
28. Howard, A.A., Perego, M., Karniadakis, G.E., Stinis, P.: Multifidelity deep operator networks for data-driven and

physics-informed problems. J. Comput. Phys. **493**, 112462 (2023)
29. Hsieh, D.Y.: Water waves in an elastic vessel. Acta. Mech. Sin. **13**, 289–303 (1997)
30. Hu, Z., Jagtap, A.D., Karniadakis, G.E., Kawaguchi, K.: Augmented physics-informed neural networks (APINNs): a gating network-based soft domain decomposition methodology. Eng. Appl. Artif. Intell. **126**, 107183 (2023)
31. Jagtap, A.D., Kawaguchi, K., Karniadakis, G.E.: Adaptive activation functions accelerate convergence in deep and physics-informed neural networks. J. Comput. Phys. **404**, 109136 (2020)
32. Jagtap, A.D., Kharazmi, E., Karniadakis, G.E.: Conservative physics-informed neural networks on discrete domains for conservation laws: Applications to forward and inverse problems. Comput. Methods Appl. Mech. Eng. **365**, 113028 (2020)
33. Jin, X., Cai, S., Li, H., Karniadakis, G.E.: Nsfnets (Navier–Stokes flow nets): Physics-informed neural networks for the incompressible Navier–Stokes equations. J. Comput. Phys. **426**, 109951 (2021)
34. Jordan, M.I., Mitchell, T.M.: Machine learning: trends, perspectives, and prospects. Science **349**(6245), 255–260 (2015)
35. Karakoç, H.Z.S.B.G.: A cubic B-spline Galerkin approach for the numerical simulation of the GEW equation. Stat. Opt. Inf. Comput. **4**(1), 30–41 (2016)
36. Karakoç, S.B.G., Zeybek, H.: A septic B-spline collocation method for solving the generalized equal width wave equation. Kuwait J. Sci. **43**(3), 20–31 (2016)
37. Karakoc, S.B.G., Omrani, K., Sucu, D.: Numerical investigations of shallow water waves via generalized equal width (GEW) equation. Appl. Numer. Math. **162**, 249–264 (2021)
38. Karniadakis, G.E., Kevrekidis, I.G., Lu, L., Perdikaris, P., Wang, S., Yang, L.: Physics-informed machine learning. Nat. Rev. Phys. **3**(6), 422–440 (2021)
39. Karta, M.: A numerical algorithm for solitary wave solutions of the GEW equation. Afr. Mat. **34**(4), 90 (2023)
40. Kaya, D.: A numerical simulation of solitary-wave solutions of the generalized regularized long-wave equation. Appl. Math. Comput. **149**(3), 833–841 (2004)
41. Kharazmi, E., Zhang, Z., Karniadakis, G.E.: hp-vpinns: variational physics-informed neural networks with domain decomposition. Comput. Methods Appl. Mech. Eng. **374**, 113547 (2021)
42. Kissas, G., Yang, Y., Hwuang, E., Witschey, W.R., Detre, J.A., Perdikaris, P.: Machine learning in cardiovascular flows modeling: Predicting arterial blood pressure from non-invasive 4D flow MRI data using physics-informed neural networks. Comput. Methods Appl. Mech. Eng. **358**, 112623 (2020)
43. Lagaris, I.E., Likas, A., Fotiadis, D.I.: Artificial Neural Networks for solving ordinary and partial differential equations. IEEE Trans. Neural Netw. **9**(5), 987–1000 (1998)
44. Lou, Q., Meng, X., Karniadakis, G.E.: Physics-informed neural networks for solving forward and inverse flow problems via the Boltzmann-BGK formulation. J. Comput. Phys. **447**, 110676 (2021)
45. Lucor, D., Agrawal, A., Sergent, A.: Physics-aware deep neural networks for surrogate modeling of turbulent natural convection. arXiv preprint arXiv:2103.03565
46. Ma, Y.-L., Wazwaz, A.-M., Li, B.-Q.: Soliton resonances, soliton molecules, soliton oscillations and heterotypic solitons for the nonlinear maccari system. Nonlinear Dyn. **111**(19), 18331–18344 (2023)
47. Mahmoudabadbozchelou, M., Caggioni, M., Shahsavari, S., Hartt, W.H., Em Karniadakis, G., Jamali, S.: Data-driven physics-informed constitutive metamodeling of complex fluids: a multifidelity neural network (MFNN) framework. J. Rheol. **65**(2), 179–198 (2021)
48. Mao, Z., Jagtap, A.D., Karniadakis, G.E.: Physics-informed neural networks for high-speed flows. Comput. Methods Appl. Mech. Eng. **360**, 112789 (2020)
49. McClenny, L.D., Braga-Neto, U.M.: Self-adaptive physics-informed neural networks. J. Comput. Phys. **474**, 111722 (2023)
50. Meng, X., Karniadakis, G.E.: A composite neural network that learns from multi-fidelity data: application to function approximation and inverse PDE problems. J. Comput. Phys. **401**, 109020 (2020)
51. Mohebbi, A., Abbaszadeh, M., Dehghan, M.: The use of a meshless technique based on collocation and radial basis functions for solving the time fractional nonlinear Schrödinger equation arising in quantum mechanics. Eng. Anal. Bound. Elem. **37**(2), 475–485 (2013)
52. Morrison, P., Meiss, J., Cary, J.: Scattering of regularized-long-wave solitary waves. Physica D **11**(3), 324–336 (1984)
53. Moukalled, F., Mangani, L., Darwish, M., Moukalled, F., Mangani, L., Darwish, M.: The Finite Volume Method. Springer, Berlin (2016)
54. Nasir, M., Jabeen, S., Afzal, F., Zafar, A.: Solving the generalized equal width wave equation via sextic-spline collocation technique. Int. J. Math. Comput. Eng. **1**(2), 229–242 (2023)
55. Oruç, Ö.: Delta-shaped basis functions-pseudospectral method for numerical investigation of nonlinear generalized equal width equation in shallow water waves. Wave Motion **101**, 102687 (2021)
56. Peregrine, D.H.: Calculations of the development of an undular bore. J. Fluid Mech. **25**(2), 321–330 (1966)
57. Peregrine, D.H.: Long waves on a beach. J. Fluid Mech. **27**(4), 815–827 (1967)
58. Perrone, N., Kao, R.: A general finite difference method for arbitrary meshes. Comput. Struct. **5**(1), 45–57 (1975)
59. Raissi, M., Perdikaris, P., Karniadakis, G.E.: Physics-informed neural networks: a deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations. J. Comput. Phys. **378**, 686–707 (2019)
60. Raissi, M., Wang, Z., Triantafyllou, M.S., Karniadakis, G.E.: Deep learning of vortex-induced vibrations. J. Fluid Mech. **861**, 119–137 (2019)
61. Raissi, M., Yazdani, A., Karniadakis, G.E.: Hidden fluid mechanics: learning velocity and pressure fields from flow visualizations. Science **367**(6481), 1026–1030 (2020)
62. Raslan, K.R.: A computational method for the equal width equation. Int. J. Comput. Math. **81**(1), 63–72 (2004)
63. Raslan, K.R.: Collocation method using cubic B-spline for the generalized equal width equation. Int. J. Simul. Process Model. **2**(1–2), 37–44 (2006)

64. Roshan, T.: A Petrov–Galerkin method for solving the generalized equal width (GEW) equation. J. Comput. Appl. Math. **235**(6), 1641–1652 (2011)
65. Shin, Y., Darbon, J., Karniadakis, G.E.: On the convergence and generalization of physics informed neural networks. arXiv e-prints (2020) arXiv (2004)
66. Shukla, K., Jagtap, A.D., Karniadakis, G.E.: Parallel physics-informed neural networks via domain decomposition. J. Comput. Phys. **447**, 110683 (2021)
67. Tian, S., Niu, Z., Li, B.: Mix-training physics-informed neural networks for high-order rogue waves of cmKdV equation. Nonlinear Dyn. **111**(17), 16467–16482 (2023)
68. Wang, S., Teng, Y., Perdikaris, P.: Understanding and mitigating gradient flow pathologies in physics-informed neural networks. SIAM J. Sci. Comput. **43**(5), A3055–A3081 (2021)
69. Wang, S., Yu, X., Perdikaris, P.: When and why pinns fail to train: a neural tangent kernel perspective. J. Comput. Phys. **449**, 110768 (2022)
70. Wazwaz, A.-M.: A sine-cosine method for handlingnonlinear wave equations. Math. Comput. Model. **40**(5–6), 499–508 (2004)
71. Wazwaz, A.-M.: New solitary wave solutions to the Kuramoto–Sivashinsky and the Kawahara equations. Appl. Math. Comput. **182**(2), 1642–1650 (2006)
72. Wazwaz, A.-M.: Partial Differential Equations and Solitary Waves Theory. Springer, Berlin (2010)
73. Wazwaz, A.-M.: Painlevé integrability and lump solutions for two extended (3+ 1)-and (2+ 1)-dimensional Kadomtsev–Petviashvili equations. Nonlinear Dyn. **111**(4), 3623–3632 (2023)
74. Wight, C.L., Zhao, J.: Solving Allen-Cahn and Cahn-Hilliard equations using the adaptive physics informed neural networks, arXiv preprint arXiv:2007.04542
75. Xiao, C., Zhu, X., Yin, Fukang, Cao, X.: Physics-informed neural network for solving coupled Korteweg–de Vries equations. J. Phys. Conf. Ser. **2031**(1), 012056 (2021)
76. Yang, X., Zafar, S., Wang, J.-X., Xiao, H.: Predictive large-eddy-simulation wall modeling via physics-informed neural networks. Phys. Rev. Fluids **4**(3), 034602 (2019)
77. Yin, M., Zheng, X., Humphrey, J.D., Karniadakis, G.E.: Non-invasive inference of thrombus material properties with physics-informed neural networks. Comput. Methods Appl. Mech. Eng. **375**, 113603 (2021)
78. Yuan, L., Ni, Y.-Q., Deng, X.-Y., Hao, S.: A-PINN: Auxiliary physics informed neural networks for forward and inverse problems of nonlinear integro-differential equations. J. Comput. Phys. **462**, 111260 (2022)
79. Zeybek, H., Karakoç, S.B.G.: Application of the collocation method with Bsplines to the GEW equation. Electron. Trans. Numer. Anal. **46**, 71–88 (2017)
80. Zhang, L.: A finite difference scheme for generalized regularized long-wave equation. Appl. Math. Comput. **168**(2), 962–972 (2005)
81. Zhu, J., Chen, Y.: Data-driven solutions and parameter discovery of the nonlocal mKdV equation via deep learning method. Nonlinear Dyn. **111**(9), 8397–8417 (2023)