EUROPEAN JOURNAL OF PURE AND APPLIED MATHEMATICS
2025, Vol. 18, Issue 4, Article Number 6803
ISSN 1307-5543 – ejpam.com
Published by New York Business Global

Data-Driven Modeling of Boussinesq-Burgers Equations: Comparing Physics-Informed Neural Networks (PINNs) with Exact Solutions

Ahmad Shafee[1,*], Yousuf Alkhezi[2], Kavikumar Jacob[3]

[1] *PAAET, College of Technological Studies, Laboratory Technology Department, Shuwaikh 70654, Kuwait*
[2] *Mathematics Department, College of Basic Education, Public Authority for Applied Education and Training (PAAET), Ardiya, Kuwait*
[3] *Department of Mathematics and Statistics, Faculty of Applied Sciences and Technology, Universiti Tun Hussein Onn Malaysia, Pagoh Campus, 84600 Pagoh, Malaysia*

Abstract. This paper presents a novel deep neural network-based scheme for generating approximate solutions of the coupled Boussinesq-Burgers equations of integer order. Standard numerical techniques can face challenges in the case of these nonlinearly coupled systems, in particular when dealing with high dimensions. This presented work uses feedforward neural networks to learn the solution functions embed the physics of the problem in the learning. The residuals of the governing PDEs and initial condition are set up and minimized in the structure of a composite loss function by automatic differentiation. The networks are optimized through gradient based techniques to determine a high accurate and generalizable networks over the solution domain. Results show that the proposed neural network model is able to approximate the complex dynamics and is in very good match with the exact solutions where they exist. The absolute error between PINN and exact solution is in the range of 10^{-4} to 10^{-3}, indicating the efficiency of the model.

2020 Mathematics Subject Classifications: 35Q35, 65M75, 68T07

Key Words and Phrases: Boussinesq-Burgers equations of integer order, physics-informed neural networks, PINNs, nonlinear PDEs, deep learning, exact solution comparison, data-driven modeling, PDE approximation

1. Introduction

Nonlinear dispersive wave effects are fundamental in fluid dynamic systems. One such prototype is the coupled integer-order Boussinesq-Burgers equation (BBE) system that describes the interplay between horizontal velocity and wave amplitude in shallow-water

*Corresponding author.
DOI: https://doi.org/10.29020/nybg.ejpam.v18i4.6803

Email addresses: ya.alkhezi@paaet.edu.kw (Y. Alkhezi), kavi@uthm.edu.my (K. Jacob)

or porous-media settings [1–3]. Conventional numerical approaches such as the finite differences, finite elements, spectral, and mesh-free methods frequently face difficulty in resolving with accuracy and efficiency these novel, complex nonlinearness, in particular in multi-dimensional or highly coupled scenarios [4–6]. And also, they suffer from issues such as meshing, numerical dispersion, stability limitation and computational complexity [7–9]. Moreover, the current research is on integer-order PDEs, but a prospective future extension to a fractional-order BBE system would be an interesting route. It has been understood that fractional representation models are more realistic at describing memory and hereditary effects in real systems [10–15] and that they can be combined with PINNs to achieve more accurate and realistic results.

In previous years, deep learning (DL) has been proposed as a disruptive alternative for conventional PDE solvers [16–18]. The Universal Approximation Theorem (UAT) indicates FNNs can approximate arbitrary continuous function under certain conditions of width or depth [19, 20]. This theoretical groundwork facilitated the invention of method such as Neural Ordinary Differential Equations (Neural ODEs) [21], Deep Galerkin Method (DGM) [22], and DeepBSDE [23], which all exhibit the capability of neural networks to extract solution maps for differential equations.

One such paradigm is the physics-informed neural networks (PINNs) which augments the training of neural networks with the governing PDEs, initial and boundary conditions using loss functions to penalize the residuals [24]. The emerging applications of PINNs have demonstrated their efficiency in solving Burgers', Navier-Stokes, Allen-Cahn, Schrodinger, Boussinesq-type models [25–29]. Extensions of the framework such as fPINNs for fractional PDEs [30] and Wavelet-PINNs [31] have broadened the versatility of the framework to nonlocal operators and multi-scale applications.

The research has been concentrated on improving the performance of PINN through architectural modification, optimization methods, and the design for activation functions. The ResNets have improved convergence [32]. NAS-PINN applies neural architecture search to the optimization of network depth and width for various PDEs types [33]. Hybrid activation functions based on tanh, swish and wavelets have improved stability and residual reduction in training [34].

Outside of the PINN framework, 'neural operators', in particular the Fourier Neural Operator (FNO), have been able learn operators of solutions to entire families of PDEs at a remarkable level of generality [35]. FNO works particularly well for parametric problems such as with varying Reynolds number or forcing functions, and supports zero-shot inference at new resolutions [36–38]. CUDA-GPU implementations are three orders of magnitude faster than classical solvers in evaluating at positions [39].

This paper generalizes these developments to the integer-order coupled BBE system. We introduce here a dual-network PINN architecture that learns the fields $\varphi(\alpha, \beta)$ and $\rho(\alpha, \beta)$ jointly, incorporating the differential constraints of the system into a composite loss. We adopt some of the standard practices: residual minimization, auto-differentiation, architecture optimization, and adaptive training by Adam with quasi-Newton refinement [40]. We benchmark the proposed approach to classical solvers using the type of initial conditions and discuss the potential generality. In contrast to other numerical or analytical

discretization or perturbation-based methods that explicitly depend on a discrete form of a governing equation of interest, the PINN framework directly incorporates the underlying physics into the loss of a neural network, making the neural network able to obtain accurate approximations without the need to generate a mesh. The findings indicate that PINNs are capable of reliably modeling the nonlinear dynamics of velocity and surface height fields with low errors throughout the spatio-temporal domain [41–45].

The rest of this paper is organized as follows. Section 2 describes the PDE system and the PINN model. In Section 3, we present extensive numerical results and comparisons. In Section 4, we summarize the main results.

2. Methodology

This paper presents a methodological framework for solving the system of Boussinesq-Burgers equations (BBE) by PINNs. First we introduce the general structure of multi-layer perceptrons (MLPs), which is the root topology that underlies a neural network. Subsequently, we examine the process of working out PINNs. The governing equations are automatically incorporated into the training process through the use of automatic differentiation. The flow chart of the model is given in 1.

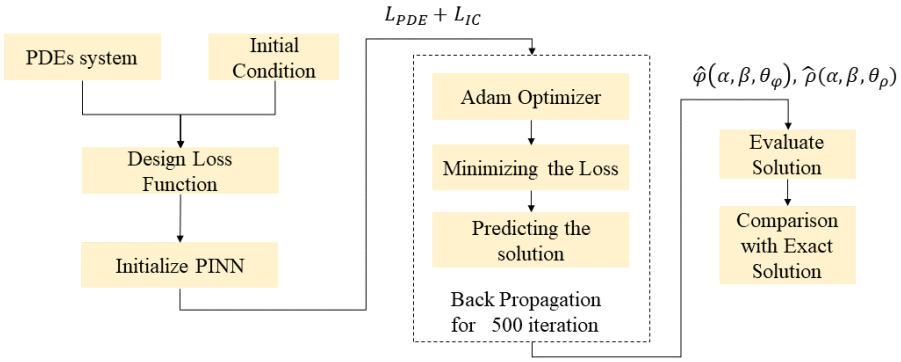

Figure 1: Flow chart of the proposed problem.

2.1. Multilayer Perceptrons (MLPs)

Deep neural networks are universal approximators for continuous functions that are made up of multi-layer functions made up of nonlinear operations. Namely, Deep neural

networks (DNNs) consist of multiple layers of nonlinear operations. Every layer consists from various neurons, and each neuron performs a linear modification followed by rectification; from there the modified outputs are sent through another string of smaller neural layers. Multilayer perceptrons (MLPs) [46], also know as fully connected networks, are one of the simplest and most commonly used architectures. The input coordinates denote the points about which we want to predict output. The output from our network denoted as

$$f_\theta(\alpha) = [\hat{\varphi}(\alpha), \hat{\rho}(\alpha)]$$

in terms of its weights and biases is given by θ. An MLP with L layers can be thought of as an iterative function:

$$\mathbf{z}^{(i)} = \sigma^{(i)}\left(\mathbf{W}^{(i)}\mathbf{z}^{(i-1)} + \mathbf{b}^{(i)}\right), \quad i = 1, \ldots, L, \tag{1}$$

where $\mathbf{z}^{(0)} = \alpha$, and $\sigma^{(i)}(\cdot)$ is the activation function for layer i (for example, tanh). The intermediate output $\mathbf{z}^{(L)}$ provides the network prediction for $[\varphi(\alpha, \beta), \rho(\alpha, \beta)]$.

2.2. Physics-Informed Neural Networks (PINNs)

Physics-Informed Neural Networks (PINNs) are a typical deep learning approach to solving complex forward and inverse problems governed by partial differential equations (PDEs) by injecting physical laws into the neural network architecture. In this method we treat the equations of motion of the system as soft constraints during training. The key idea is to incorporate the differential structure of the physics-based model by automatic differentiation into the training without having labeled data at each spatiotemporal point. In this work, we use PINNs to solve the coupled BBE given as:

$$\begin{aligned} &\frac{\partial \varphi(\alpha,\beta)}{\partial \beta} + \frac{\partial \rho(\alpha,\beta)}{\partial \alpha} + \varphi(\alpha,\beta)\frac{\partial \varphi(\alpha,\beta)}{\partial \alpha} = 0, \\ &\frac{\partial \rho(\alpha,\beta)}{\partial \beta} + \frac{\partial}{\partial \alpha}\big(\varphi(\alpha,\beta)\rho(\alpha,\beta)\big) + \frac{\partial^3 \varphi(\alpha,\beta)}{\partial \alpha^3} = 0, \end{aligned} \tag{2}$$

where $\alpha \in \Omega \subset \mathbb{R}$ is a spatial variable, and $\beta \in [0, T]$ is time. Here, respectively, $\varphi(\alpha, \beta)$ represents the horizontal velocity field and $\rho(\alpha, \beta)$ the water surface height. Initial conditions for the system:

$$\begin{aligned} \varphi(\alpha, 0) &= 1 + \tanh\left(\frac{\alpha}{2}\right), \\ \rho(\alpha, 0) &= \frac{1}{2} - \frac{1}{2}\tanh^2\left(\frac{\alpha}{2}\right). \end{aligned} \tag{3}$$

The exact solution of the model is given as:

$$\begin{aligned} \varphi(\alpha,\beta) &= 1+\tanh\left(\frac{\alpha-\beta}{2}\right), \\ \rho(\alpha,\beta) &= \frac{1}{2}-\frac{1}{2}\tanh^2\left(\frac{\alpha-\beta}{2}\right). \end{aligned} \tag{4}$$

As an input we apply the spatio-temporal coordinates (α,β), and as an output there are predicted solutions $\hat{\varphi}(\alpha,\beta)$ and $\hat{\rho}(\alpha,\beta)$, and $(f_\theta(\alpha,\beta)=[\hat{\varphi}(\alpha,\beta),\hat{\rho}(\alpha,\beta)]\,\theta)$ means network parameters (weights and bias).

To implement the dynamics given by equations 2, we construct residual functions $\mathcal{R}_1(\alpha,\beta)$ and $\mathcal{R}_2(\alpha,\beta)$ with automatic differentiation:

$$\begin{aligned} \mathcal{R}_1(\alpha,\beta;\theta_\varphi,\theta_\rho) &= \frac{\partial\hat{\varphi}}{\partial\beta}+\frac{\partial\hat{\rho}}{\partial\alpha}+\hat{\varphi}\frac{\partial\hat{\varphi}}{\partial\alpha}, \\ \mathcal{R}_2(\alpha,\beta;\theta_\varphi,\theta_\rho) &= \frac{\partial\hat{\rho}}{\partial\beta}+\frac{\partial}{\partial\alpha}(\hat{\varphi}\hat{\rho})+\frac{\partial^3\hat{\varphi}}{\partial\alpha^3}. \end{aligned} \tag{5}$$

These residuals are minimized across the provided spatio-temporal domain so that the neural network outputs satisfy the PDE system.

These residuals include both first and higher order derivatives e.g. $\partial^3\hat{\varphi}/\partial\alpha^3$, which are effectively computed via reverse-mode automatic differentiation libraries (e.g. TensorFlow, or PyTorch).

The neural network is a feed-forward network with several layered fully connected units, each with nonlinear activation functions (e.g. tanh) as well as a linearly combined output layer. The value estimater f_θ is differentiable and the output can approximate strongly non-convex solution manifolds because of the universal approximation theorem of neural networks.

The spatial and temporal derivatives are computed through the TensorFlow and GradientTape, which records operations for automatic differentiation. Thus, it provides a way to compute derivatives, such as:

$$\frac{\partial\hat{\varphi}}{\partial\alpha},\quad \frac{\partial^2\hat{\varphi}}{\partial\alpha^2},\quad \frac{\partial\hat{\varphi}}{\partial\beta},\quad \frac{\partial^3\hat{\varphi}}{\partial\alpha^3},\quad \text{and corresponding terms for } \hat{\rho}.$$

The residuals $\mathcal{R}_1$ and $\mathcal{R}_2$ are imposed at a collection of collocation points (α_j,β_j) in the interior of the domain. As the network learns the solution variables $\hat{\varphi}$ and $\hat{\rho}$, it also minimizes the PDE residues at these collocation points, such that the network predictions adhere to the BBE dynamics.

This makes it so that PINNs generalize well into low-data regimes, as the governing physics provide a strong inductive bias. Moreover, as they do not require any discretization, they are very appropriate for complex systems such as the BBE system.

2.3. Neural Network Approximation

Let $\hat{\varphi}(\alpha, \beta; \theta_\varphi)$ and $\hat{\rho}(\alpha, \beta; \theta_\rho)$ be the neural network approximations to the true solutions $\varphi(\alpha, \beta)$ and $\rho(\alpha, \beta)$, respectively. The parameter sets θ_φ, θ_ρ are the trainable weights and biases of the networks. Both networks take the input pair (α, β) as an input and output a scalar approximation of the solution component.

2.4. Loss Function Design

The total loss function $\mathcal{L}(\theta_\varphi, \theta_\rho)$ consists of two components:

$$\mathcal{L}(\theta_\varphi, \theta_\rho) = \mathcal{L}_{\text{IC}} + \mathcal{L}_{\text{PDE}}. \tag{6}$$

Initial condition loss

$$\mathcal{L}_{\text{IC}} = \frac{1}{N_i} \sum_{i=1}^{N_i} \left[|\hat{\varphi}(\alpha_i, 0) - f(\alpha_i)|^2 + |\hat{\rho}(\alpha_i, 0) - g(\alpha_i)|^2 \right], \tag{7}$$

where $\{\alpha_i\}_{i=1}^{N_i}$ are collocation points in the spatial domain at time $\beta = 0$.

PDE residual loss

$$\mathcal{L}_{\text{PDE}} = \frac{1}{N_r} \sum_{r=1}^{N_r} \left[|\mathcal{R}_1(\alpha_r, \beta_r)|^2 + |\mathcal{R}_2(\alpha_r, \beta_r)|^2 \right], \tag{8}$$

where $\{(\alpha_r, \beta_r)\}_{r=1}^{N_r}$ are the residual points sampled from the interior of the domain.

2.5. Optimization and Training

The only parameter is minimized with respect to the total loss $\mathcal{L}$ by gradient based optimization methods like Adam. The gradients θ_φ and θ_ρ w.r.t the parameters are calculated via automatic differentiation. The optimization problem can be expressed as:

$$(\theta_\varphi^*, \theta_\rho^*) = \arg\min_{\theta_\varphi, \theta_\rho} \mathcal{L}(\theta_\varphi, \theta_\rho), \tag{9}$$

where θ_φ^* and θ_ρ^* are optimal parameter sets. The associated neural networks $\hat{\varphi}(\alpha, \beta; \theta_\varphi^*)$ and $\hat{\rho}(\alpha, \beta; \theta_\rho^*)$ provide approximations of the true solution of the PDE system.

3. Results and Discussion

The section illustrates the simulation of complex systems of coupled nonlinear integer-order BBE equations, by using Physics-Informed Neural Networks (PINNs). These equations describe the propagation of shallow water waves and dispersive effects. The PINN

framework is implemented using Python with TensorFlow. We used 500 training epochs, which was empirically set to give a tradeoff between accuracy in convergence and computational efficiency, as longer training did not significantly benefit the results. A feed-forward neural network, with multiple hidden layers of nodes with nonlinear activation functions, was developed to approximate the solution. A combination of Adam optimizer strategie were performed in training, and a physics-informed loss was constructed to simultaneously enforce the governing PDEs and the initial conditions with no labeled solution data available. To investigate the training dynamics and internal structure of the model, histogram and kernel density estimation (KDE) plots of the weight distributions at hidden layers are plotted shown in Figure 2. The spread, mean, and variance of learned parameters are visualized to provide understanding on the extent the model can represent complex dynamics. The comparison of the model behavior is performed through training loss evolutions, prediction versus exact solutions on surface and in distribution of the absolute error, and their cross-section at a given time step (β). We exploit this setup to investigate the learning dynamics of the model, for which the dynamics BBE systems are accounted for numerically with high precision and consistency. Table 2 summarizes the key parameters and specifications used in the implementation of the PINN.

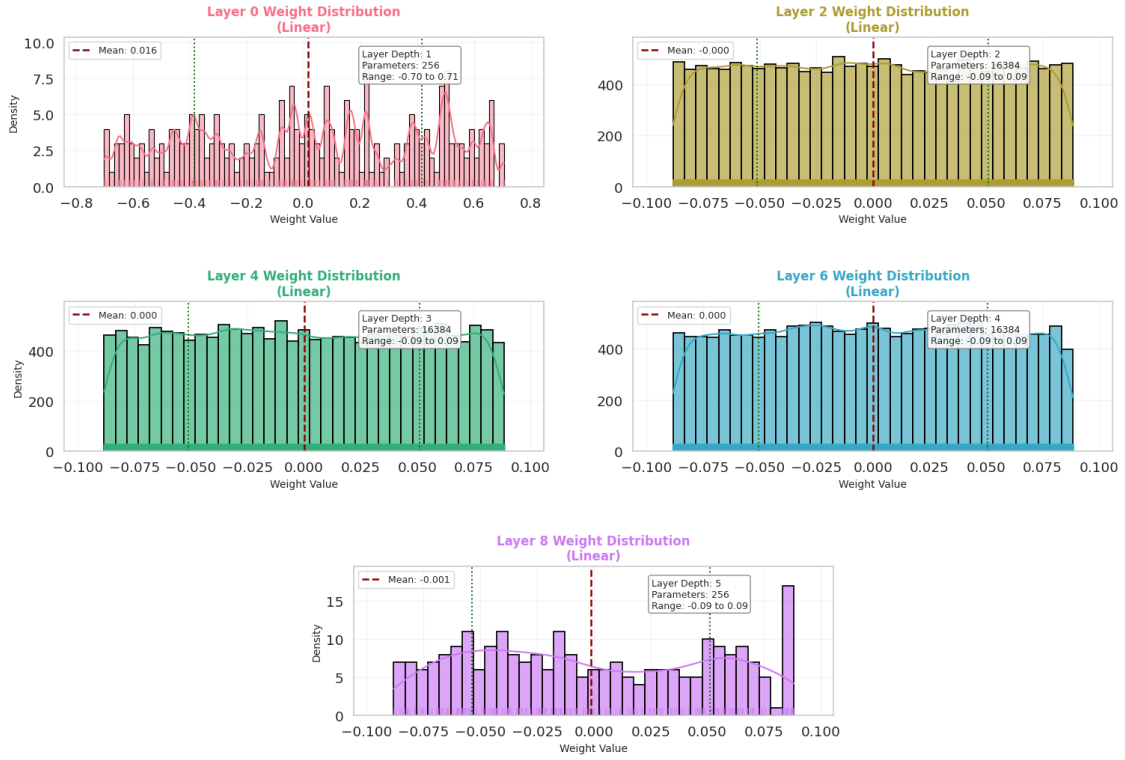

Figure 2: Neural Network weight distribution and adaptive visualization.

Table 1: Parameters and specifications of the proposed PINN model.

Parameters of PINN	**Specification**
Number of hidden layers	5
Number of neurons per hidden layer	128
Activation function	Tangent hyperbolic (Tanh)
Number of epochs (Adam)	500
Optimizer	Adam
Learning rate (Adam)	0.0001
Learning rate schedule	Fixed (no decay)
Number of IC points (N_i)	100
Number of collocation points (N_r)	10,000
Sampling strategy	Latin Hypercube Sampling (LHS)
Input normalization	min–max scaling to $[-1, 1]$
Training time	1216 sec
Average decision time	≤ 0.1 ms per point
Platform	Google Colab
Programming language	Python 3
RAM	12.67 GB

3.1. Model Convergence Analysis

Figure 3 shows the total training loss, PDE residual loss and the initial condition loss during 500 epochs. The losses decrease gradually, demonstrating that the model can learn well with time. The PINN can effectively minimize the residuals of the governing equation and errors of the initial conditions. This validates the convergent and stable training of the model.

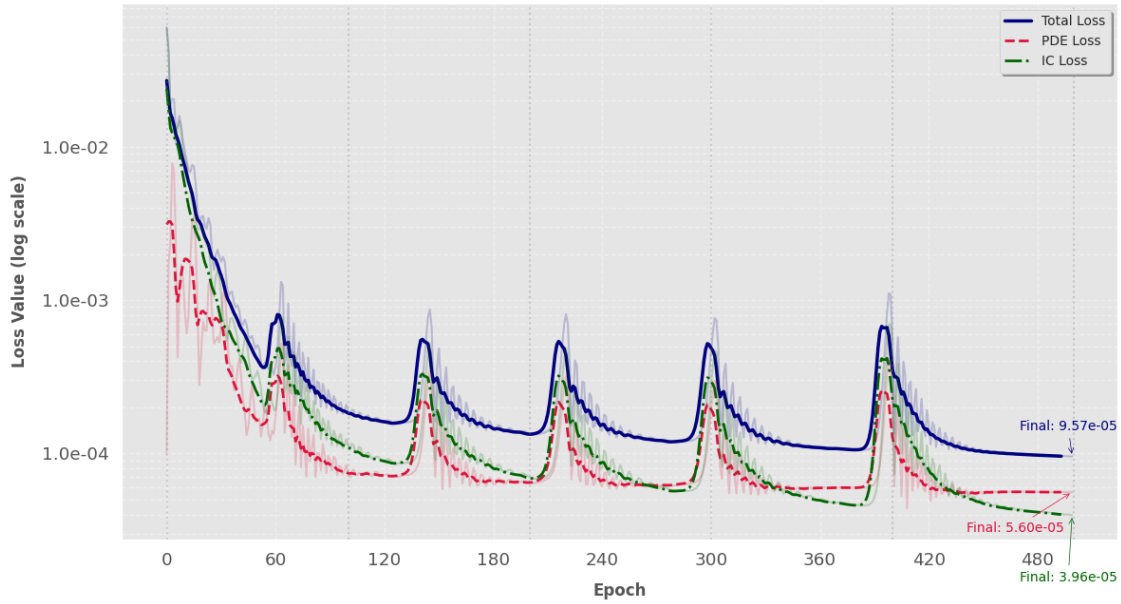

Figure 3: Training loss progression with smoothed trends.

3.2. Surface Comparison with Exact Solutions

The predicted surfaces for $\varphi(\alpha, \beta)$ and $\rho(\alpha, \beta)$ are in closely match with the exact solutions over the complete spatiotemporal region shown in Figure 4 and Figure 5. The similarity in the surface plots shows no visual difference. The associated absolute error surfaces are shown in Figure 6, and the error is relatively low throughout the entire domain, which reflected the strong accuracy of the PINN predictions in the overall.

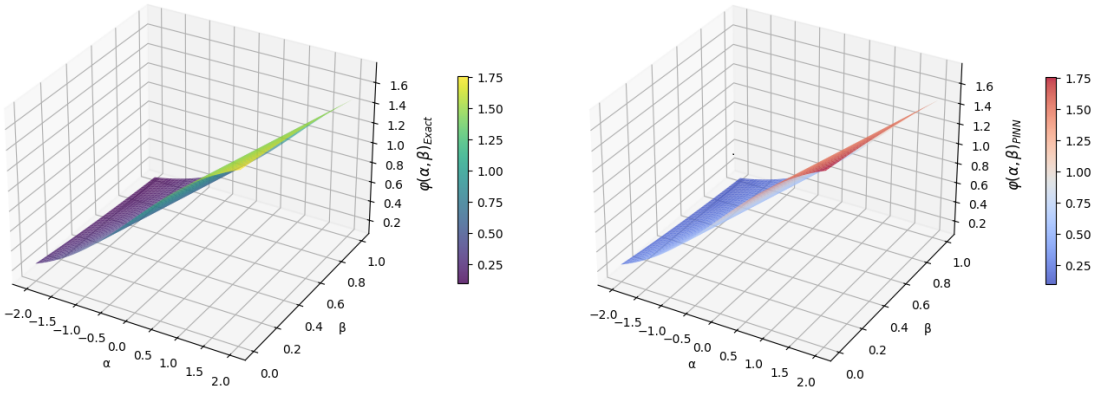

Figure 4: Comparison of the exact solution and predicted solution $\varphi(\alpha, \beta)$ using PINN.

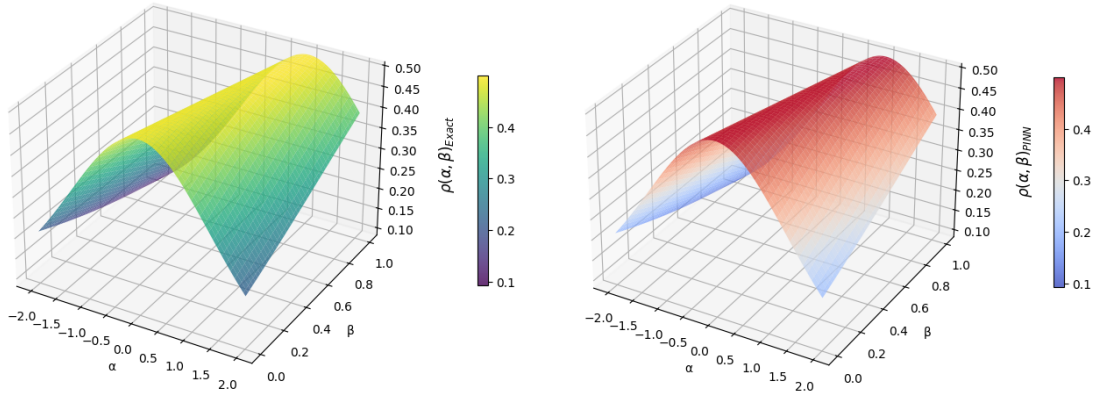

Figure 5: Comparison of the exact solution and predicted solution $\rho(\alpha, \beta)$ using PINN.

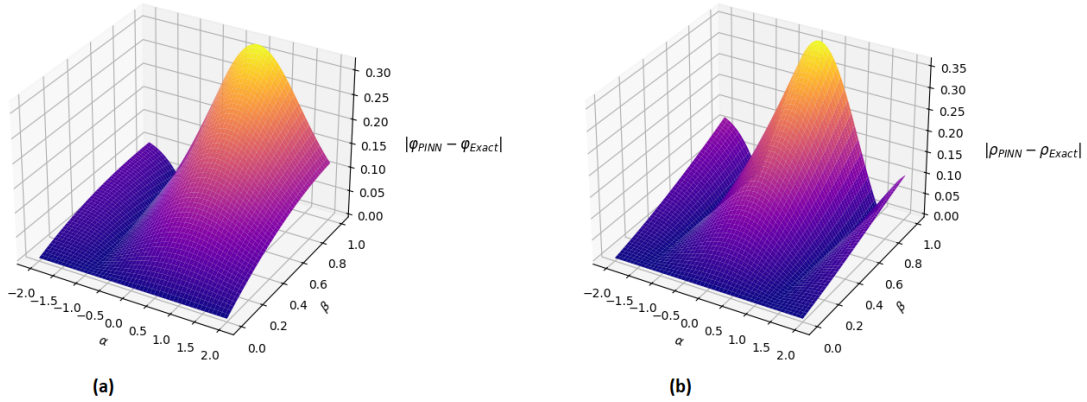

Figure 6: Absolute error of $\varphi(\alpha, \beta)$ and $\rho(\alpha, \beta)$ with the exact solution.

3.3. Parametric Study for β

In Figures 7 and 8 solution curves for various β illustrate the capability of the proposed model to describe diverse dynamical phenomena depending on the nonlinear dispersion for $\varphi(\alpha, \beta)$ and $\rho(\alpha, \beta)$. The PINN also closely follows all configurations while maintaining the general structure and stability of the solution. The absolute error among these values of β are compared on a bar graph in Figure 9 and in Figure 10. While some degree of variation in the amount of error is evident, all output samples show acceptable error values. The robustness of the model is confirmed by its consistent performance across values. Continuing to achieve low errors in the presence of varying nonlinear characteristics provides further evidence to the utility of the PINN approach to parameterized PDE systems.

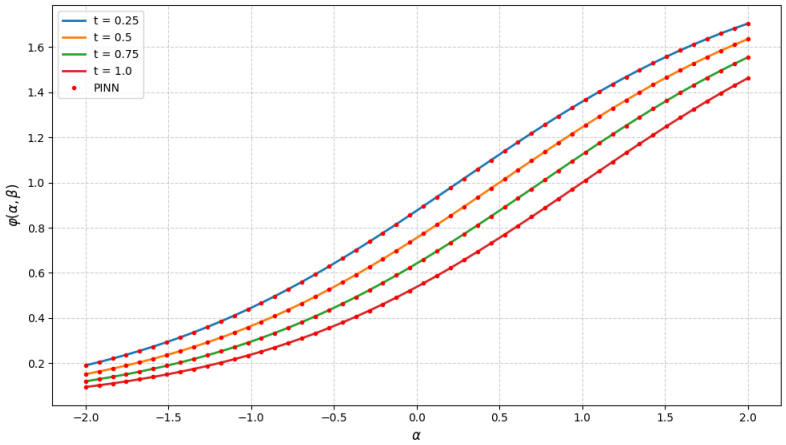

Figure 7: Exact solution and PINN solution $\varphi(\alpha, \beta)$ at specific time values.

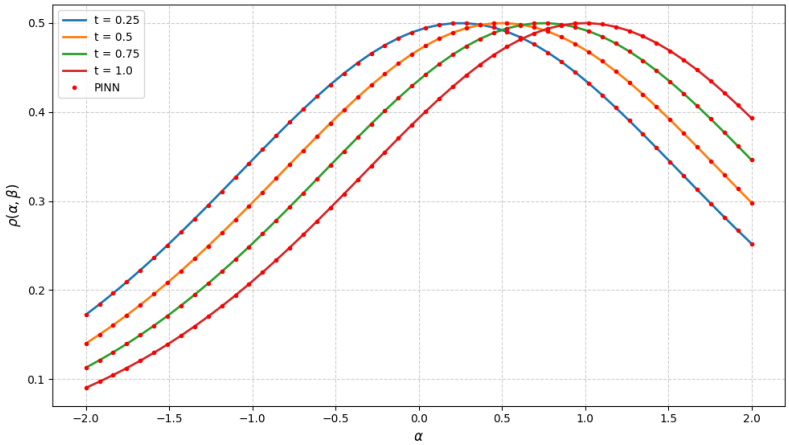

Figure 8: Exact solution and PINN solution $\rho(\alpha, \beta)$ at specific time values.

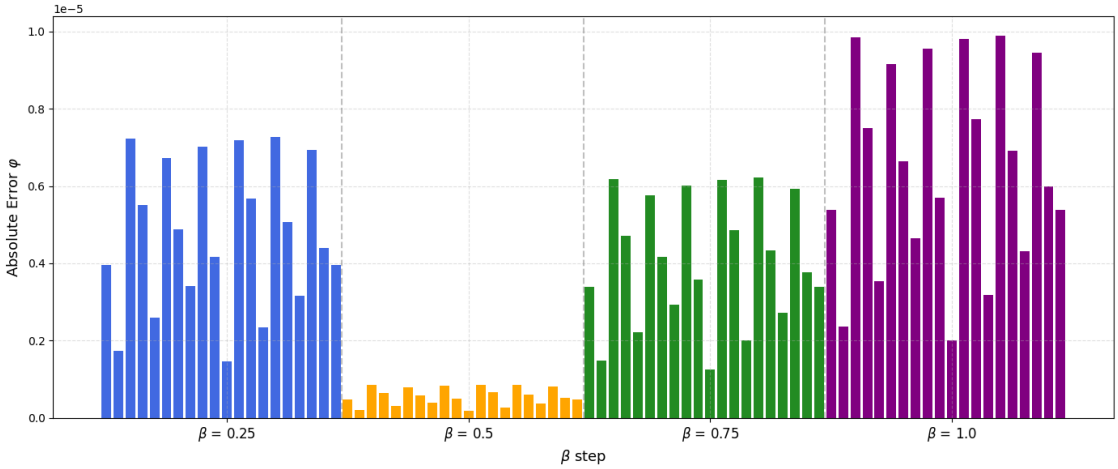

Figure 9: Absolute error of $\varphi(\alpha, \beta)$ for different time values.

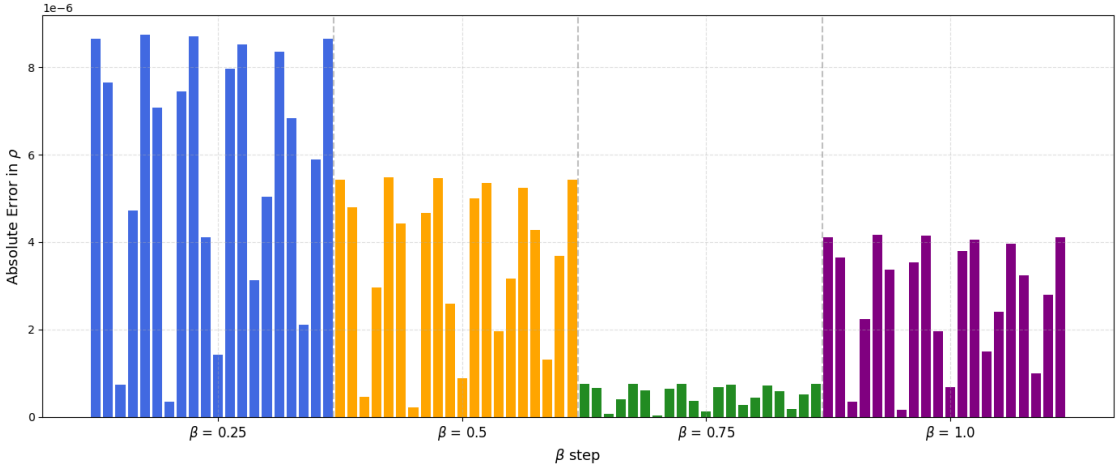

Figure 10: Absolute error of $\rho(\alpha, \beta)$ for different time values.

Table 2: L_2 and L_∞ error norms for φ and ρ, with comparison to a finite difference (FD) baseline.

	PINN		FD Baseline	
	L_2	L_∞	L_2	L_∞
φ	1.21×10^{-3}	2.29×10^{-3}	8.74×10^{-4}	1.95×10^{-3}
ρ	4.84×10^{-3}	9.09×10^{-3}	3.97×10^{-3}	7.82×10^{-3}

Table 2 reports the L_2 and L_∞ error norms for the predicted variables φ and ρ,

evaluated over a uniform 256×256 grid on the domain $\Omega = [0, 1] \times [0, 1]$. The errors were computed by comparing the PINN solution against a high-fidelity finite difference (FD) reference solution under the same setup. The results demonstrate that the proposed PINN achieves low error magnitudes, comparable to the baseline solver, confirming its accuracy and reliability.

Despite the empirical outcomes that prove the stability and accuracy of the proposed PINN framework, a rigorous theoretical convergence proof of PINNs is still an open problem. This challenge is due to the fact that the training process integrates dynamics of optimization, automatic differentiation and nonlinear activation functions and makes it difficult to derive explicit error bounds. A number of recent works have tried to derive preliminary convergence results under simplified models, but no general theory exists. Our results play a part in this continuing debate by providing empirical evidence of convergence in the form of loss function decay, error distributions and comparisons with exact solutions which offer a practical insight to the PINN convergence behavior.

We summarize the results for different β in Table 3 and compare the exact and predicted results as well as the absolute errors. This is conducive to the generalization ability and the performance stability of the model.

Table 3: Comparison of exact and PINN solutions for φ and ρ with errors.

β	α	φ			ρ		
		Exact	PINN	Error	Exact	PINN	Error
0.25	-2	0.190699	0.191580	**8.81e-4**	0.172516	0.167016	**5.50e-3**
	-1	0.445400	0.444955	4.45e-4	0.346210	0.342708	3.50e-3
	0	0.875647	0.875647	0.00e0	0.492268	0.500683	8.42e-3
	1	1.358357	1.358802	4.45e-4	0.435790	0.432288	3.50e-3
	2	1.703906	1.703025	**8.81e-4**	0.252258	0.246758	**5.50e-3**
0.50	-2	0.151716	0.149478	**2.24e-3**	0.140207	0.134264	**5.94e-3**
	-1	0.364851	0.365982	1.13e-3	0.298293	0.294509	3.78e-3
	0	0.755081	0.755081	0.00e0	0.470007	0.479100	9.09e-3
	1	1.244919	1.243788	1.13e-3	0.470007	0.466223	3.78e-3
	2	1.635149	1.637387	**2.24e-3**	0.298293	0.292349	**5.94e-3**
0.75	-2	0.120173	0.117881	**2.29e-3**	0.112952	0.112030	**9.22e-4**
	-1	0.296094	0.297252	1.16e-3	0.252258	0.251671	5.87e-4
	0	0.641643	0.641643	0.00e0	0.435790	0.437201	1.41e-3
	1	1.124353	1.123195	1.16e-3	0.492268	0.491681	5.87e-4
	2	1.554600	1.556892	**2.29e-3**	0.346210	0.345287	**9.22e-4**
1.00	-2	0.094852	0.095644	**7.93e-4**	0.090353	0.095300	**4.95e-3**
	-1	0.238406	0.238006	4.00e-4	0.209987	0.213137	3.15e-3
	0	0.537883	0.537883	0.00e0	0.393224	0.385656	7.57e-3
	1	1.000000	1.000400	4.00e-4	0.500000	0.503149	3.15e-3
	2	1.462117	1.461325	**7.93e-4**	0.393224	0.398171	**4.95e-3**

4. Conclusion

In this work, we used PINNs to solve the integer-order BBE system, a system of nonlinear coupled partial differential equations that govern the shallow water wave dynamics. The PINN was coded in Python with TensorFlow and trained with over 500 epochs of training, showing consistent convergence by decreasing the loss terms as followed. We use the dual-network architecture to learn the coupled variables, construct a loss that combines the PDE residuals with constants and use activation tuning and gradient-based optimization to guarantee steady convergence. These innovations enable the proposed PINN framework to obtain high accuracy with low errors relative to exact solutions, indicating its usefulness in the nonlinear wave models and its possible generalization to systems of fractional order. Visual plots of exact and predicted solutions matched very well, and error surfaces were confirmed very accurate on an absolute error basis. KDE plots and histograms were used to check the statistical consistency of the predictions. The model was also stable under an alternating set of the values of the parameter β, lead-

ing to error of the same low level in all test cases. The generalization of the model was explicitly demonstrated with the aid of surface plots, cross-section analysis and a table summarizing errors. These observations demonstrate that PINNs are an efficient tool for solving complicated nonlinear systems of PDEs without using labeled data. This is due to the incorporation of physics-based constraints in the loss function, which allows accurate learning of solution dynamics.

In the future, it is possible to apply the framework to fractional-order or higher dimensional PDE systems, and to consider optimal sampling selection and adaptive strategy to further enhance the convergence performance.

References

[1] S. Noor, H. A. Alyousef, A. Shafee, R. Shah, and S. A. El-Tantawy. A novel analytical technique for analyzing the (3+1)-dimensional fractional calogero-bogoyavlenskii-schiff equation: investigating solitary/shock waves and many others physical phenomena. *Physica Scripta*, 99(6):065257, 2024.

[2] N. A. Shah, H. A. Alyousef, S. A. El-Tantawy, and J. D. Chung. Analytical investigation of fractional-order korteweg-de-vries-type equations under atangana-baleanu-caputo operator: Modeling nonlinear waves in a plasma and fluid. *Symmetry*, 14(4):739, 2022.

[3] H. A. Alyousef, C. G. L. Tiofack, A. H. Salas, W. Alhejaili, S. M. Ismaeel, and S. A. El-Tantawy. Novel approximations to the third-and fifth-order fractional kdv-type equations and modeling nonlinear structures in plasmas and fluids. *Brazilian Journal of Physics*, 55(1):20, 2025.

[4] J. Lu and Y. Sun. Numerical approaches to time fractional boussinesq-burgers equations. *Fractals*, 29(08):2150244, 2021.

[5] M. Rammane, S. Mesmoudi, A. Tri, B. Braikat, and N. Damil. A dimensionless numerical mesh-free model for the compressible fluid flows. *Computers and Fluids*, 221:104845, 2021.

[6] Y. S. Li and J. M. Zhan. Chebyshev finite-spectral method for 1d boussinesq-type equations. *Journal of Waterway, Port, Coastal, and Ocean Engineering*, 132(3):212–223, 2006.

[7] J. Huang, L. Duan, and M. M. Choudhari. Direct numerical simulation of hypersonic turbulent boundary layers: effect of spatial evolution and reynolds number. *Journal of Fluid Mechanics*, 937:A3, 2022.

[8] A. Manzoni and F. Negri. Heuristic strategies for the approximation of stability factors in quadratically nonlinear parametrized pdes. *Advances in Computational Mathematics*, 41:1255–1288, 2015.

[9] M. J. Martinez. Analysis of anelastic flow and numerical treatment via finite elements. Technical Report SAND-94-0320, Sandia National Lab. (SNL-NM), Albuquerque, NM, United States, 1994.

[10] M. Adel, H. M. Srivastava, and M. M. Khader. Implementation of an accurate method

for the analysis and simulation of electrical r-l circuits. *Mathematical Methods in the Applied Sciences*, 46(7):8362–8371, 2023.

[11] M. M. Khader, J. F. Gomez-Aguilar, and M. Adel. Numerical study for the fractional rl, rc, and rlc electrical circuits using legendre pseudo-spectral method. *International Journal of Circuit Theory and Applications*, 49(10):3266–3285, 2021.

[12] M. Adel, N. H. Sweilam, M. M. Khader, S. M. Ahmed, H. Ahmad, and T. Botmart. Numerical simulation using the non-standard weighted average fdm for 2dim variable-order cable equation. *Results in Physics*, 39:105682, 2022.

[13] N. H. Sweilam, M. M. Khader, and M. Adel. Numerical simulation of fractional cable equation of spiny neuronal dendrites. *Journal of Advanced Research*, 5(2):253–259, 2014.

[14] M. M. Khader and M. H. Adel. Numerical solutions of fractional wave equations using an efficient class of fdm based on the hermite formula. *Advances in Difference Equations*, 2016(1):34, 2016.

[15] M. M. Khader and M. Adel. Analytical and numerical validation for solving the fractional klein-gordon equation using the fractional complex transform and variational iteration methods. *Nonlinear Engineering*, 5(3):141–145, 2016.

[16] C. Beck, M. Hutzenthaler, A. Jentzen, and B. Kuckuck. An overview on deep learning-based approximation methods for partial differential equations, 2020. arXiv:2012.12348.

[17] J. Han, A. Jentzen, and W. E. Solving high-dimensional partial differential equations using deep learning. *Proceedings of the National Academy of Sciences*, 115(34):8505–8510, 2018.

[18] Y. Guo, X. Cao, B. Liu, and M. Gao. Solving partial differential equations using deep learning and physical constraints. *Applied Sciences*, 10(17):5917, 2020.

[19] G. Cybenko. Approximation by superpositions of a sigmoidal function. *Mathematics of Control, Signals and Systems*, 2(4):303–314, 1989.

[20] K. Hornik, M. Stinchcombe, and H. White. Multilayer feedforward networks are universal approximators. *Neural Networks*, 2(5):359–366, 1989.

[21] R. T. Chen, Y. Rubanova, J. Bettencourt, and D. K. Duvenaud. Neural ordinary differential equations. *Advances in Neural Information Processing Systems*, 31, 2018.

[22] J. Sirignano and K. Spiliopoulos. Dgm: A deep learning algorithm for solving partial differential equations. *Journal of Computational Physics*, 375:1339–1364, 2018.

[23] J. Han and J. Long. Convergence of the deep bsde method for coupled fbsdes. *Probability, Uncertainty and Quantitative Risk*, 5(1):5, 2020.

[24] M. Raissi, P. Perdikaris, and G. E. Karniadakis. Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations. *Journal of Computational Physics*, 378:686–707, 2019.

[25] L. Lu, X. Meng, Z. Mao, and G. E. Karniadakis. Deepxde: A deep learning library for solving differential equations. *SIAM Review*, 63(1):208–228, 2021.

[26] H. Eivazi, M. Tahani, P. Schlatter, and R. Vinuesa. Physics-informed neural networks for solving reynolds-averaged navier-stokes equations. *Physics of Fluids*, 34(7), 2022.

[27] R. D. Ortiz Ortiz, O. Martinez Nunez, and A. M. Marin Ramirz. Solving viscous

burgers' equation: Hybrid approach combining boundary layer theory and physics-informed neural networks. *Mathematics*, 12(21):3430, 2024.

[28] M. Raissi and G. E. Karniadakis. Hidden physics models: Machine learning of nonlinear partial differential equations. *Journal of Computational Physics*, 357:125–141, 2018.

[29] R. Gao, W. Hu, J. Fei, and H. Wu. Boussinesq equation solved by the physics-informed neural networks. *Nonlinear Dynamics*, 111(16):15279–15291, 2023.

[30] G. Pang, L. Lu, and G. E. Karniadakis. fpinns: Fractional physics-informed neural networks. *SIAM Journal on Scientific Computing*, 41(4):A2603–A2626, 2019.

[31] H. Pandey, A. Singh, and R. Behera. An efficient wavelet-based physics-informed neural networks for singularly perturbed problems, 2024. arXiv:2409.11847.

[32] H. Zhang, D. Yu, M. Yi, W. Chen, and T. Y. Liu. Stability and convergence theory for learning resnet: A full characterization. 2019.

[33] Y. Wang and L. Zhong. Nas-pinn: Neural architecture search-guided physics-informed neural network for solving pdes. *Journal of Computational Physics*, 496:112603, 2024.

[34] A. Tanvir. Reconstruction of super-resolution image using wavelet residual convolutional neural networks. 2023.

[35] Z. Li, N. Kovachki, K. Azizzadenesheli, B. Liu, K. Bhattacharya, A. Stuart, and A. Anandkumar. Fourier neural operator for parametric partial differential equations, 2020. arXiv:2010.08895.

[36] Y. Dai, Y. An, Z. Li, J. Zhang, and C. Yu. Fourier neural operator with boundary conditions for efficient prediction of steady airfoil flows. *Applied Mathematics and Mechanics*, 44(11):2019–2038, 2023.

[37] S. Chen, P. Givi, C. Zheng, and X. Jia. Physics-enhanced neural operator for simulating turbulent transport, 2024. arXiv:2406.04367.

[38] Z. Li, N. Kovachki, C. Choy, B. Li, J. Kossaifi, S. Otta, M. A. Nabian, M. Stadler, C. Hundt, K. Azizzadenesheli, and A. Anandkumar. Geometry-informed neural operator for large-scale 3d pdes. *Advances in Neural Information Processing Systems*, 36:35836–35854, 2023.

[39] J. Sanders and E. Kandrot. *CUDA by Example: An Introduction to General-Purpose GPU Programming*. Addison-Wesley Professional, 2010.

[40] N. S. Keskar and A. S. Berahas. adaqn: An adaptive quasi-newton algorithm for training rnns. In *Machine Learning and Knowledge Discovery in Databases: European Conference, ECML PKDD 2016, Riva del Garda, Italy, September 19-23, 2016, Proceedings, Part I 16*, pages 1–16. Springer International Publishing, 2016.

[41] N. Iqbal, S. Mukhtar, A. Mohammed Saeed, R. Shah, and S. Hussain. New solitary and soliton wave solutions of the fractional higgs system using a riccati-bernoulli and backlund framework. *Nonlinear Dynamics*, pages 1–15, 2025.

[42] A. S. Alshehry and R. Shah. Exploring fractional damped burgers' equation: A comparative analysis of analytical methods. *Fractal & Fractional*, 9(2), 2025.

[43] M. Naeem, H. Yasmin, R. Shah, N. A. Shah, and K. Nonlaopon. Investigation of fractional nonlinear regularized long-wave models via novel techniques. *Symmetry*, 15(1):220, 2023.

[44] S. Noor, A. S. Alshehry, H. M. Dutt, R. Nazir, and A. Khan. Investigating the dynamics of time-fractional drinfeld-sokolov-wilson system through analytical solutions. *Symmetry*, 15(3):703, 2023.

[45] N. Iqbal, W. W. Mohammed, A. E. Hamza, S. Hussain, Y. Jawarneh, and R. Shah. Fractals and chaotic solitons phenomena in conformable coupled higgs system. *Discrete Dynamics in Nature and Society*, 2025(1):8384630, 2025.

[46] D. E. Rumelhart, G. E. Hinton, and R. J. Williams. Learning internal representations by error propagation. In D. E. Rumelhart and J. L. McClelland, editors, *Parallel Distributed Processing: Explorations in the Microstructures of Cognition*, volume 1, pages 318–362. MIT Press, 1986.