Research Article

The New Simulation of Quasiperiodic Wave, Periodic Wave, and Soliton Solutions of the KdV-mKdV Equation via a Deep Learning Method

Yong Zhang, Huanhe Dong , Jiuyun Sun, Zhen Wang , Yong Fang , and Yuan Kong

College of Mathematics and Systems Science, Shandong University of Science and Technology, Qingdao 266590, China

Correspondence should be addressed to Yuan Kong; kongyuan@sdust.edu.cn

Received 25 October 2021; Accepted 9 November 2021; Published 26 November 2021

Academic Editor: Heng Liu

Copyright © 2021 Yong Zhang et al. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

How to solve the numerical solution of nonlinear partial differential equations efficiently and conveniently has always been a difficult and meaningful problem. In this paper, the data-driven quasiperiodic wave, periodic wave, and soliton solutions of the KdV-mKdV equation are simulated by the multilayer physics-informed neural networks (PINNs) and compared with the exact solution obtained by the generalized Jacobi elliptic function method. Firstly, the different types of solitary wave solutions are used as initial data to train the PINNs. At the same time, the different PINNs are applied to learn the same initial data by selecting the different numbers of initial points sampled, residual collocation points sampled, network layers, and neurons per hidden layer, respectively. The result shows that the PINNs well reconstruct the dynamical behaviors of the quasiperiodic wave, periodic wave, and soliton solutions for the KdV-mKdV equation, which gives a good way to simulate the solutions of nonlinear partial differential equations via one deep learning method.

1. Introduction

In recent decades, nonlinear partial differential equations (PDEs) are paid more and more attention in optical fiber communication, condensed matter physics, plasma physics, and fluid mechanics [1–3]. As one type of PDEs, the integrable equations attract more attention for their good properties; for example, the KdV-mKdV equation is widely used in nonlinear lattices, wave propagation of bound particles, acoustic waves, and thermal pulses [4–9]. In addition, the KdV-mKdV equation or the Gardner equation,

$$u_t + 6uu_x + 6u^2u_x + u_{xxx} = 0, \quad (1)$$

tends to the classical Korteweg–de Vries (KdV) equation or the mKdV equation in the limits of small or large waves, respectively. Via the complex Miura transformation,

$$v(x, t) = u(1 + u) + i \frac{\partial u}{\partial x}, \quad (2)$$

the KdV-mKdV equation will be transformed into the KdV equation:

$$\frac{\partial v}{\partial t} + 6v \frac{\partial v}{\partial x} + \frac{\partial^3 v}{\partial x^3} = 0, \quad (3)$$

which provides a model for describing surface and internal waves in oceans and the Rossby waves and can be also used to describe ion acoustic waves in plasmas and dust acoustic solitary structures in magnetized dusty plasmas [10–15]. It is usually used to describe the balance between nonlinear wave steepening and linear wave dispersion.

Via linear transformations of the function,

$$w(x, t) = \frac{1}{2} + u \left(x - \frac{3}{2}t, t \right), \quad (4)$$

which is used to describe the acoustic waves in the plasma, the propagation of an elastic quasiplane wave in a lattice, electromagnetic waves in size-quantized films, and internal ocean waves under certain stratification [16–19]. Because PDEs contain more than two nonlinear terms or the

lowest-order nonlinearity is cubic [4, 5, 20, 21] in many physics applications, the exact and numerical solutions of the KdV-mKdV equation can more visually depict the various phenomena of nature and help understand the physical mechanism of the phenomenon.

Recently, with the explosive growth of available data and computing resources, recent advances in machine learning and data analytics have yielded transformative results across diverse scientific disciplines, including image recognition, cognitive science, and genomics [22–25]. Machine learning techniques demonstrate impressive results for a range of highly complex tasks, especially where an accurate mathematical representation of the problem cannot be obtained [26, 27]. Therefore, the application of such techniques to solve PDEs will become a new subfield of machine learning. The physics-informed neural networks (PINNs) are introduced in [28, 29] and have become one of the most popular deep learning methods. PINNs employ a neural network as a solution surrogate and seek to find the best neural network guided by data and physical laws expressed as PDEs. A series of studies have shown the effectiveness of PINNs in fractional PDEs, stochastic differential equations, integrable systems, biomedical problems, and fluid mechanics [30–37]. Therefore, the deep learning method with underlying physical constraints is introduced to obtain the data-driven quasiperiodic wave, periodic wave, and soliton solutions of the KdV-mKdV equation in this paper.

The rest of this paper is organized as follows: Section 2 introduces the PINNs and briefly presents some problem setups. Section 3 obtains data-driven quasiperiodic wave, periodic wave, and soliton solutions for the KdV-mKdV equation by the PINNs and discusses the effects of different numbers of initial points sampled, residual collocation points sampled, network layers, and neurons per hidden layer. Conclusion is given in the last section.

2. PINNs Method

In this section, the general architecture of PINNs is introduced and applied into the KdV-mKdV equation. Following notation similar to [33], the general form of the functions that the PINNs can approximate is

$$\frac{\partial u}{\partial t} = -\mathcal{N}[u; \lambda], x \in \Omega, t \in [0, T], \quad (5)$$

where $u(t, x)$ is the solution and $\mathcal{N}[u; \lambda]$ is a nonlinear operator connecting the state variables u with the system parameters λ . The term t denotes time, and x denotes the system input. The domain Ω can be bounded based on prior knowledge of the dynamical system, and $[0, T]$ is the time interval of system evolution. The model parameters λ can be constant or unknown. In case that λ is unknown, approximating function (6) becomes a problem of system identification, where we seek the parameters λ for which the expression in (6) is satisfied. In order to describe the physical laws of the dynamical system, the physics-informed neural network $f(t, x)$ is defined as

$$f(t, x) = \frac{\partial u}{\partial t} + \mathcal{N}[u; \lambda]. \quad (6)$$

Note that if the system parameters λ are known, the nonlinear operator $\mathcal{N}[u; \lambda]$ simplifies to $\mathcal{N}[u]$. A neural network is used to predict $u(t, x)$ based on the inputs t and x . To determine $f(t, x)$, automatic differentiation [38] of the components of the neural network $u(t, x)$ is used. Based on this, the required derivatives of $u(t, x)$ with respect to time t and system inputs x are computed. As a result, the neural network $f(t, x)$ has the same parameters compared to the neural network $u(t, x)$, but different activation functions. The shared parameters of these two neural networks are optimized by minimizing the loss function:

$$\text{MSE} = \text{MSE}_u + \text{MSE}_f, \quad (7)$$

where $\text{MSE}_u = (1/N_u) \sum_i^{N_u} |u(t_u^i, x_u^i) - u^i|^2$ denotes the mean squared error loss corresponding to the initial data, N_u is the total number of training data, $\text{MSE}_f = (1/N_f) \sum_i^{N_f} |f(t_f^i, x_f^i)|^2$ denotes the mean squared error at a finite set of collocation points, and N_f is the total number of collocation points. The number of collocation points and training data influence the prediction accuracy and the computational time to optimize the loss function. The error MSE_u enforces the boundary conditions of the independent variables x , and MSE_f enforces the physics of the dynamical system imposed by condition (5), i.e., it penalizes deviations of the predicted physical law. Given a training data set and known system parameters λ , the parameters (weights and biases) of the neural networks are obtained to minimize (7). If the parameters λ are unknown, the same objective is trained but the system parameters are regarded as additional variables.

In this paper, the L-BFGS algorithm is used to optimize all loss functions. The algorithm is a full-batch gradient-based optimization algorithm based on a quasi-Newton method [39]. All codes in this article are based on Python 3.7 and TensorFlow 1.15, and all numerical examples reported here are run on a Lenovo Legion Y7000P 2020H computer with 2.60 GHz 6-core Intel(R) Core(TM) i7-10750H CPU and 16 GB memory. The hidden layers of the neural network use hyperbolic tangent activation functions. All codes can be mainly referred to <https://github.com/maziarraissi/PINNs>.

3. Data-Driven Solutions to the KdV-mKdV Equation

In the following, we consider the $(1+1)$ -dimensional KdV-mKdV equation along with initial-boundary value conditions:

$$\begin{cases} u_t + 6uu_x + 6u^2u_x + u_{xxx} = 0, x \in [0, 0.5], & t \in [-10, 10], \\ u(t_0, x) = a, \\ u(t, 0) = u_1(t), u(t, 0.5) = u_2(t), \end{cases} \quad (8)$$

where a is the constant and $u_1(t)$ and $u_2(t)$ are certain boundary functions.

Now, $f(t, x)$ is defined as

$$f := u_t + 6uu_x + 6u^2u_x + u_{xxx}. \quad (9)$$

The quasiperiodic wave, periodic wave, and soliton solutions of the KdV-mKdV equation have been obtained by many different methods, such as variational iteration method [40], tanh-function method [41], Jacobi elliptic function method [42–44], and so on. Here, the quasiperiodic wave, periodic wave, and soliton solutions are simulated by using the PINNs and compared with the known exact solutions, so as to prove the effectiveness of solving the numerical solutions $u(t, x)$ by neural networks.

3.1. Data-Driven Quasiperiodic Wave Solution. Firstly, we introduce the quasiperiodic wave solution which have been derived the Jacobi elliptic function method in [44], and the quasiperiodic wave solution is formed as

$$u(t, x) = -\frac{1}{2} + k^2 \text{cn}\left(kx + \left(\frac{3}{2} - 2k^4 + k^2\right)kt, k\right), \quad (10)$$

where $\text{cn}(\cdot)$ is the elliptic cosine function and $0 \leq k \leq 1$.

In order to numerically construct data-driven quasiperiodic wave solution of equation (8), quasiperiodic wave solution (10) is reduced to

$$u(t, x) = -\frac{1}{2} + \frac{81 \text{cn}((9/10)x + (44901/50000)t, (9/10))}{100}, \quad (11)$$

when $k = 0.9$. The corresponding initial condition is obtained by substituting a specific initial value into equation (11):

$$u(t, x) = -\frac{1}{2} + \frac{81 \text{cn}((9/10)x - (44901/50000)t, (9/10))}{100}, \quad (12)$$

with taking $[x_0, x_1]$ and $[t_0, t_1]$ as $[0, 0.5]$ and $[-10, 10]$, respectively.

The traditional finite difference model on even grids in MATLAB with initial data (12) is adopted to generate the training data. Specifically, by dividing space $[0, 0.5]$ into 51 points and time $[-10, 10]$ into 401 points, quasiperiodic wave solution $u(t, x)$ is discretized into 401 snapshots accordingly. We subsample a small training dataset that contain initial-boundary subsets by randomly extracting $N_u = 120$ from original initial-boundary data and $N_f = 10000$ collocation points which are generated by LHS [45]. After giving a dataset of initial and boundary points, the latent quasiperiodic wave solution $u(t, x)$ is successfully learned by using Python and TensorFlow [46] with 4 hidden layers and 60 neurons per hidden layer to tune all learnable parameters of the neural network and regulating the loss function (8). The model achieves a relative $\mathbb{L}_2$ error of $7.880803e^{-03}$ in about 1441 seconds, and the number of iterations is 8080.

Figure 1(a) shows the density diagrams of the quasiperiodic wave solution and clearly compares the exact solution with the learned spatiotemporal solution. The comparisons of different spatial locations $x = 0.1, 0.25$, and 0.4 are presented in the bottom panel of Figure 1(a). Figure 1(b)

presents the error diagram about the difference between the exact quasiperiodic wave solution. Combining with Figure 1(b), it is visible that the error between the numerical solution and the exact solution is very small. The error is mainly concentrated in the area where the solitary wave solution is generated; that is to say, the oscillation has a certain influence on the PINNs. The three-dimensional motion of the predicted solution and the loss curve at different iterations are given out in detail in Figures 1(c) and 1(d). From Figure 1(d), it is obvious that there are some obvious fluctuations in the training only at the start. The results show that the loss curve is very smooth when the number of iterations is more than 400, which proves the effectiveness and stability of the PINNs.

In addition, based on the same initial and boundary values of the quasiperiodic wave solution in the case of $N_u = 120$ and $N_f = 10000$, the control variable method is used to study the influence of different numbers of network layers and neurons per hidden layer on the quasiperiodic wave solution dynamics of the KdV-mKdV equation. The relative $\mathbb{L}_2$ errors of different network layers and different neurons per hidden layer are given in Table 1. The relative $\mathbb{L}_2$ errors with 4 network layers and 40 neurons per hidden layer when taking different numbers of subsampling points N_u in the initial-boundary data and collocation points N_f are shown in Table 2. From the data in Table 1, it can be seen that the number of network layers and single-layer neurons have no obvious regularity in the influence of the relative $\mathbb{L}_2$ error and both have certain influence. However, the influence of the number of single-layer neurons is greater. To sum up, it is eyeable that the network layers and the single-layer neurons jointly determine the relative $\mathbb{L}_2$ error to some extent. In the case of the same training data set, from Table 2, it is obvious that the influence of N_u on the relative $\mathbb{L}_2$ error of the network is not obvious, which also indicates the network model with physical constraints can uncover accurate predicted solutions with smaller initial-boundary data and relatively many sampled collocation points.

3.2. Data-Driven Periodic Wave Solution. In this section, we study the periodic wave solution.

$$u(t, x) = -\frac{1}{2} + \frac{9}{25} \text{cn}\left(\frac{3}{5}x + \frac{6003}{6250}t, \frac{3}{5}\right), \quad (13)$$

when $k = 0.6$ in equation (10). Then, $[x_0, x_1]$ and $[t_0, t_1]$ are taken as $[0, 0.5]$ and $[-10, 10]$, respectively. The corresponding initial condition is obtained by substituting a specific initial value into equation (14).

$$u(t, x) = -\frac{1}{2} + \frac{9}{25} \text{cn}\left(\frac{3}{5}x - \frac{6003}{6250}t, \frac{3}{5}\right). \quad (14)$$

The periodic wave solution is simulated numerically by using the same data generation and sampling method as the quasiperiodic wave solution. Then, $N_u = 200$ from original initial-boundary data and $N_f = 1000$ collocation points are sampled randomly. After training by Python and TensorFlow with 8 hidden layers and 20 neurons per hidden layer,

FIGURE 1: The quasiperiodic wave solution $u(x, t)$: (a) the density diagram and profile diagrams of three different positions; (b) the error density diagram; (c) the three-dimensional diagram; (d) the loss curve.

TABLE 1: The quasiperiodic wave solution of the KdV-mKdV equation: relative final learned error estimations in the $\mathbb{L}_2$ norm for different numbers of network layers and neurons per hidden layer.

Layers	Neurons			
	20	40	60	80
2	$3.602283e-2$	$1.466837e-2$	$1.780898e-2$	$1.560210e-2$
4	$9.527266e-3$	$8.251322e-3$	$7.880803e-3$	$8.563890e-3$
6	$8.885163e-3$	$8.344743e-3$	$8.318624e-3$	$8.183045e-3$
8	$9.012644e-3$	$8.507917e-3$	$8.578885e-3$	$8.395397e-3$

the model achieves a relative $\mathbb{L}_2$ error of $3.289852e^{-03}$ in about 873 seconds and the number of iterations is 7392.

Similar to Figure 1, Figure 2 shows the density diagrams, the error diagram about the difference between exact and learned periodic wave solutions, the three-dimensional figure, and the loss curve figure, respectively. In addition, the relative $\mathbb{L}_2$ errors of different network layers and different neurons per hidden layer are given in Table 3. At the same

time, the relative $\mathbb{L}_2$ errors of different numbers of N_u and N_f in the case of 4 network layers and 40 neurons per hidden layer are shown in Table 4. By comparing with Table 1, it is found that the influence of the number of network layers and the single-layer neurons on the relative $\mathbb{L}_2$ error is exactly the opposite. Obviously, combining Figure 1(b) and Tables 1 and 2, it is eyeable that due to the simpler structure of the periodic solution, the relative $\mathbb{L}_2$

TABLE 2: The quasiperiodic wave solution of the KdV-mKdV equation: relative final learned error measurements in the $\mathbb{L}_2$ norm for different numbers of N_u and N_f .

N_f	N_u			
	80	120	160	200
5000	$1.520107e-2$	$3.494261e-2$	$8.795864e-3$	$9.298087e-3$
10000	$8.991697e-3$	$8.251322e-3$	$9.737098e-3$	$8.227559e-3$
15000	$1.910283e-2$	$1.097170e-2$	$2.638983e-2$	$8.782078e-3$
20000	$1.358711e-2$	$1.017950e-2$	$9.225287e-3$	$8.386114e-3$

FIGURE 2: The periodic wave solution $u(x, t)$: (a) the density diagram and profile diagrams of three different positions; (b) the error density diagram; (c) the three-dimensional diagram; (d) the loss curve.

error of the periodic solution is significantly smaller. At the same time, from Table 4, it is obvious that N_u and N_f have a certain influence on the relative $\mathbb{L}_2$ error of the network, but the regularity is not obvious. From Figure 2(d), it is visible that there are some obvious fluctuations at the start and end of training, but they have little effect on the overall training, so the PINNs is still effective and stable.

3.3. Data-Driven Soliton Solution. Similar to Section 3.2, the soliton solution,

$$u(t, x) = -\frac{1}{2} + \operatorname{sech}\left(x + \frac{1}{2}t\right), \quad (15)$$

is obtained by equation (10) when $k=1$. Then, the corresponding initial condition,

TABLE 3: The periodic wave solution of the KdV-mKdV equation: relative final learned error estimations in the $\mathbb{L}_2$ norm for different numbers of network layers and neurons per hidden layer.

Layers	Neurons			
	20	40	60	80
2	$3.005433e-2$	$2.388090e-3$	$2.946261e-2$	$2.826414e-2$
4	$2.674150e-2$	$3.407698e-3$	$2.894121e-2$	$3.767917e-3$
6	$3.397864e-3$	$3.812189e-3$	$3.316271e-3$	$3.612304e-3$
8	$3.289852e-3$	$1.668124e-3$	$3.814982e-3$	$4.327971e-2$

TABLE 4: The periodic wave solution of the KdV-mKdV equation: relative final learned error measurements in the $\mathbb{L}_2$ norm for different numbers of N_u and N_f .

N_f	N_u			
	80	120	160	200
5000	$8.001687e-3$	$7.878526e-3$	$5.472321e-3$	$4.188364e-3$
10000	$6.470068e-3$	$4.159932e-3$	$4.096255e-3$	$3.407698e-3$
15000	$7.194250e-3$	$4.526040e-3$	$4.095404e-3$	$3.566130e-3$
20000	$1.824714e-2$	$4.229454e-3$	$6.915370e-3$	$5.180385e-3$

TABLE 5: The soliton solution of the KdV-mKdV equation: relative final learned error estimations in the $\mathbb{L}_2$ norm for different numbers of network layers and neurons per hidden layer.

Layers	Neurons			
	20	40	60	80
2	$9.314110e-3$	$7.436345e-3$	$2.595096e-2$	$1.772005e-2$
4	$9.154403e-3$	$3.070442e-3$	$9.465375e-3$	$9.040695e-3$
6	$1.120311e-2$	$7.512318e-3$	$6.778210e-3$	$5.110427e-3$
8	$8.747611e-3$	$5.329763e-3$	$8.411414e-3$	$8.499836e-3$

TABLE 6: The soliton solution of the KdV-mKdV equation: relative final learned error measurements in the $\mathbb{L}_2$ norm for different numbers of N_u and N_f .

N_f	N_u			
	80	120	160	200
5000	$7.829712e-3$	$3.070442e-3$	$6.715074e-3$	$4.582525e-3$
10000	$9.306359e-3$	$7.676940e-3$	$6.886371e-3$	$6.832231e-3$
15000	$8.039516e-3$	$7.248955e-3$	$9.451375e-3$	$5.664678e-3$
20000	$6.343377e-3$	$8.873952e-3$	$5.490009e-3$	$9.278955e-3$

$$u(t, x) = -\frac{1}{2} + \text{sech}(x - 5), \quad (16)$$

is obtained with taking $[x_0, x_1]$ and $[t_0, t_1]$ as $[0, 0.5]$ and $[-10, 10]$, respectively.

The same data generation and sampling method as in Section 3.1 are used to simulate numerically the soliton solution. $N_u = 120$ from original initial-boundary data and $N_f = 5000$ collocation points are sampled randomly, which builds up a small training dataset. After giving a dataset of initial and boundary points, the latent quasiperiodic wave solution $u(t, x)$ is successfully learned by Python and TensorFlow with 4 hidden layers and 40 neurons per hidden layer to tune all learnable parameters of the neural network and regulating loss function (7). The model achieves a

relative $\mathbb{L}_2$ error of $3.070442e^{-03}$ in about 82 seconds, and the number of iterations is 4119.

Compared with the abovementioned figures and tables, it is obvious that the relative $\mathbb{L}_2$ error of the soliton solution in Table 5 is larger than that of the quasiperiodic wave solution and smaller than that of the periodic wave solution. Similarly, the error is mainly concentrated in the place where the solitary solution is generated. At the same time, the influence of the number of network layers and the single-layer neurons on the relative $\mathbb{L}_2$ error in Table 6 is similar to that of periodic wave solution. From Figure 3(d), it is visible that there are some obvious fluctuations only when the number of iterations is between 0 and 300 which proves the effectiveness and stability of the PINNs.

FIGURE 3: The soliton solution $u(x, t)$: (a) the density diagram and profile diagrams of three different positions; (b) the error density diagram; (c) the three-dimensional diagram; (d) the loss curve.

4. Conclusion

In this paper, the data-driven quasiperiodic wave, periodic wave, and soliton solutions of the KdV-mKdV equation are gained by the PINNs. Specifically, we discussed the influence of different numbers of network layers and neurons per hidden layer to solve the data-driven solutions of the KdV-mKdV equation. It is visible that the network layers and the single-layer neurons jointly determine the relative $\mathbb{L}_2$ error to some extent and the error is mainly concentrated in the place where the solitary solution is generated. At the same time, according to the structures of the solutions, the applicability of the PINNs is also different. In other words, the more complex and regular the structure of the solutions, the better the effect of the PINNs. Remarkably, these results show that the PINNs can exactly recover different dynamical

behaviors and obtain the data-driven solutions more quickly and efficiently for some nonlinear partial differential equations which cannot be solved by the traditional methods. Moreover, due to the physical constraints, the network is trained with just few data and has a better physical interpretability.

The PINNs obtain a series of results about various problems in the interdisciplinary fields of applied mathematics and computational science, which opens a new path for using deep learning to simulate unknown solutions and correspondingly discover the parametric equations. In this paper, we do not discuss the influence of noise on the neural network model, more complex boundary conditions, and more sampling methods. In the future work, we will focus on studying these problems to make the PINNs more universal and general.

Data Availability

The data used to support the findings of this study are included within the article.

Conflicts of Interest

The authors declare that there are no conflicts of interest regarding the publication of this paper.

Acknowledgments

The authors thank M. Raissi, P. Perdikaris, and G. E. Karniadakis for sharing the CODE about PINNs (<https://github.com/maziarraissi/PINNs>). In addition, the authors are very grateful to Professor Y. Chen for his report and helpful discussions on PINNs at Shandong University of Science and Technology (<http://export.arxiv.org/pdf/2011.04949>). This work was supported by the National Natural Science Foundation of China (Grant nos. 11975143, 12105161, and 61602188), Natural Science Foundation of Shandong Province (Grant no. ZR2019QD018), CAS Key Laboratory of Science and Technology on Operational Oceanography (Grant no. OOST2021-05), and Scientific Research Foundation for Recruited Talents (Grant nos. 2017RCJJ068 and 2017RCJJ069).

References

- [1] Z. Fu, S. Liu, and S. Liu, "New kinds of solutions to gardner equation," *Chaos, Solitons & Fractals*, vol. 20, no. 2, pp. 301–309, 2004.
- [2] G. Q. Xu, Z. B. Li, and Y. P. Liu, "Exact solutions to a large class of nonlinear evolution equations," *Chinese Journal of Physics*, vol. 41, pp. 232–241, 2003.
- [3] Z. Yan, "Jacobi elliptic function solutions of nonlinear wave equations via the new sinh-Gordon equation expansion method," *Journal of Physics A: Mathematical and General*, vol. 36, no. 7, pp. 1961–1972, 2003.
- [4] M. Wadati, "Wave propagation in nonlinear lattice. I," *Journal of the Physical Society of Japan*, vol. 38, no. 3, pp. 673–680, 1975.
- [5] M. Wadati, "Wave propagation in nonlinear lattice. II," *Journal of the Physical Society of Japan*, vol. 38, no. 3, pp. 681–686, 1975.
- [6] M. W. Coffey, "On series expansions giving closed-form solutions of Korteweg-de Vries-like equations," *SIAM Journal on Applied Mathematics*, vol. 50, pp. 1580–1592, 1989.
- [7] M. N. B. Mohamad, "Exact solutions to the combined KdV and mKdV equation," *Mathematical Methods in the Applied Sciences*, vol. 15, no. 2, pp. 73–78, 1992.
- [8] S.-Y. Lou and L.-L. Chen, "Solitary wave solutions and cnoidal wave solutions to the combined KdV and mKdV equation," *Mathematical Methods in the Applied Sciences*, vol. 17, no. 5, pp. 339–347, 1994.
- [9] Y. Z. Peng, "New exact solutions to the combined KdV and mKdV equation," *International Journal of Theoretical Physics*, vol. 42, pp. 863–868, 2003.
- [10] P. D. Lax, "Periodic solutions of the KdV equation," *Communications on Pure and Applied Mathematics*, vol. 28, pp. 141–188, 1975.
- [11] K. Sawada and T. Kotera, "A method for finding n-soliton solutions of the K.d.V. equation and K.d.V.-like equation," *Progress of Theoretical Physics*, vol. 51, no. 5, pp. 1355–1367, 1974.
- [12] R. M. Miura, "Korteweg-de Vries equation and generalizations. I. A remarkable explicit nonlinear transformation," *Journal of Mathematical Physics*, vol. 9, no. 8, pp. 1202–1204, 1968.
- [13] W. X. Ma, "Complexiton solutions to the Korteweg-de Vries equation," *Physics Letters A*, vol. 301, no. 1-2, pp. 35–44, 2002.
- [14] D. D. Bhatta and M. I. Bhatti, "Numerical solution of KdV equation using modified Bernstein polynomials," *Applied Mathematics and Computation*, vol. 174, no. 2, pp. 1255–1268, 2006.
- [15] A. R. Seadawy, "New exact solutions for the KdV equation with higher order nonlinearity by using the variational method," *Computers & Mathematics with Applications*, vol. 62, no. 10, pp. 3741–3755, 2011.
- [16] F. Gesztesy and B. Simon, "Constructing solutions of the mKdV-equation," *Journal of Functional Analysis*, vol. 89, no. 1, pp. 53–60, 1990.
- [17] Z. Fu, S. Liu, and S. Liu, "New solutions to mKdV equation," *Physics Letters A*, vol. 326, no. 5-6, pp. 364–374, 2004.
- [18] C. Dai, J. Zhu, and J. Zhang, "New exact solutions to the mKdV equation with variable coefficients," *Chaos, Solitons & Fractals*, vol. 27, no. 4, pp. 881–886, 2006.
- [19] S. Zhang, J.-L. Tong, and W. Wang, "A generalized -expansion method for the mKdV equation with variable coefficients," *Physics Letters A*, vol. 372, no. 13, pp. 2254–2257, 2008.
- [20] H. W. Schürmann, "Traveling-wave solutions of the cubic-quintic nonlinear Schrödinger equation," *Physical Review E - Statistical Physics, Plasmas, Fluids, and Related Interdisciplinary Topics*, vol. 54, p. 4312, 1996.
- [21] Y. Zhang, H.-H. Dong, and D.-S. Wang, "Riemann-Hilbert problems and soliton solutions for a multi-component cubic-quintic nonlinear Schrödinger equation," *Journal of Geometry and Physics*, vol. 149, p. 103569, 2020.
- [22] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," *Advances in Neural Information Processing Systems*, vol. 25, pp. 1097–1105, 2012.
- [23] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, vol. 521, no. 7553, pp. 436–444, 2015.
- [24] B. M. Lake, R. Salakhutdinov, and J. B. Tenenbaum, "Human-level concept learning through probabilistic program induction," *Science*, vol. 350, no. 6266, pp. 1332–1338, 2015.
- [25] B. Alipanahi, A. Delong, M. T. Weirauch, and B. J. Frey, "Predicting the sequence specificities of DNA- and RNA-binding proteins by deep learning," *Nature Biotechnology*, vol. 33, no. 8, pp. 831–838, 2015.
- [26] M. Raissi, P. Perdikaris, and G. E. Karniadakis, "Inferring solutions of differential equations using noisy multi-fidelity data," *Journal of Computational Physics*, vol. 335, pp. 736–746, 2017.
- [27] M. Raissi, P. Perdikaris, and G. E. Karniadakis, "Machine learning of linear differential equations using Gaussian processes," *Journal of Computational Physics*, vol. 348, pp. 683–693, 2017.
- [28] M. Raissi, P. Perdikaris, and G. E. Karniadakis, "Physics informed deep learning (part I): Data-driven solutions of nonlinear partial differential equations," 2017, <https://arxiv.org/abs/1711.10561>.
- [29] M. Raissi, P. Perdikaris, and G. E. Karniadakis, "Physics informed deep learning (part II): Data-driven solutions of

- nonlinear partial differential equations,” 2017, <https://arxiv.org/abs/1711.10566>.
- [30] X. Meng, Z. Li, D. Zhang, and G. E. Karniadakis, “PPINN: parareal physics-informed neural network for time-dependent PDEs,” *Computer Methods in Applied Mechanics and Engineering*, vol. 370, Article ID 113250, 2020.
 - [31] Z. Mao, A. D. Jagtap, and G. E. Karniadakis, “Physics-informed neural networks for high-speed flows,” *Computer Methods in Applied Mechanics and Engineering*, vol. 360, Article ID 112789, 2020.
 - [32] G. S. Misyris, A. V. enzke, and S. Chatzivasileiadis, “Physics-informed neural networks for power systems,” 2019, <https://arxiv.org/abs/1911.03737>.
 - [33] 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*, vol. 378, pp. 686–707, 2019.
 - [34] J. Li and Y. Chen, “A physics-constrained deep residual network for solving the sine-Gordon equation,” *Communications in Theoretical Physics*, vol. 73, no. 1, Article ID 015001, 2021.
 - [35] J. Li and Y. Chen, “Solving second-order nonlinear evolution partial differential equations using deep learning,” *Communications in Theoretical Physics*, vol. 72, no. 10, Article ID 105005, 2020.
 - [36] J. Li and Y. Chen, “A deep learning method for solving third-order nonlinear evolution equations,” *Communications in Theoretical Physics*, vol. 72, no. 11, Article ID 115003, 2020.
 - [37] J. C. Pu, J. Li, and Y. Chen, “Soliton, breather and rogue wave solutions for solving the nonlinear Schrödinger equation using a deep learning method with physical constraints,” 2020, <https://arxiv.org/abs/2011.04949>.
 - [38] A. G. Baydin, B. A. Pearlmutter, A. A. Radul, and J. M. Siskind, “Automatic differentiation in machine learning: a survey,” *Journal of Machine Learning Research*, vol. 18, pp. 5595–5637, 2017.
 - [39] D. C. Liu and J. Nocedal, “On the limited memory BFGS method for large scale optimization,” *Mathematical Programming*, vol. 45, no. 1-3, pp. 503–528, 1989.
 - [40] J. He, “A new approach to nonlinear partial differential equations,” *Communications in Nonlinear Science and Numerical Simulation*, vol. 2, no. 4, pp. 230–235, 1997.
 - [41] E. Fan, “Extended tanh-function method and its applications to nonlinear equations,” *Physics Letters A*, vol. 277, no. 4-5, pp. 212–218, 2000.
 - [42] S. Liu, Z. Fu, S. Liu, and Q. Zhao, “Jacobi elliptic function expansion method and periodic wave solutions of nonlinear wave equations,” *Physics Letters A*, vol. 289, no. 1-2, pp. 69–74, 2001.
 - [43] Z. Yan, “Abundant families of Jacobi elliptic function solutions of the $(2 + 1)$ -dimensional integrable Davey-Stewartson-type equation via a new method,” *Chaos, Solitons & Fractals*, vol. 18, no. 2, pp. 299–309, 2003.
 - [44] A. B. Emad and S. Abdel, “Quasi-periodic, periodic waves, and soliton solutions for the combined KdV-mKdV equation,” *Zeitschrift für Naturforschung A*, vol. 64, pp. 639–645, 2009.
 - [45] M. Stein, “Large sample properties of simulations using Latin hypercube sampling,” *Technometrics*, vol. 29, pp. 143–151, 2012.
 - [46] M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen et al., “Tensorflow: large-scale machine learning on heterogeneous distributed systems,” 2016, <https://arxiv.org/abs/1603.04467>.