Physics-informed neural network for solving coupled Korteweg-de Vries equations

Chaohao Xiao, Xiaoqian Zhu*, Fukang Yin and Xiaoqun Cao

College of Meteorology and Oceanography, National University of Defense Technology, Changsha 410000, China

*Corresponding author’s e-mail: zhuxiaoqian@nudt.edu.cn

Abstract. The studies of coupled partial differential equations are focus of engineering and applied mathematics. Although traditional numerical methods have been widely used, researchers are still looking for new methods to solve coupled partial differential equations. In this paper, physical information neural network (PINN) is introduced to solve one-dimensional coupled Korteweg-de Vries (cKdV) equations. Compared with the traditional neural network, the innovation of PINN is to embed the physical constraints of the equations into the network loss function. Moreover, within the acceptable relative error range, the solution can take a longer single time step than the presently available. The results revealed that PINN can solve the cKdV equations with reasonable errors only by training a small amount of data.

1. Introduction

In recent years, coupled partial differential equations have played a very important role in the fields of engineering and applied mathematics. However, in classical fluid mechanics, coupled partial differential equations have been known for a long time and often appear in some numerical modeling [1]. One-dimensional coupled KdV equations (cKdV) belong to coupled partial differential equations. In this paper, we only focus on the one-dimensional coupled KdV equations as follows:

\[ u_t - 6auu_x - 2bv_v - au_{xxx} = 0 \quad x \in [x_1, x_2], \quad t \in [0, T] \]

\[ v_t + 3uv_x + v_{xxx} = 0 \]

with the initial conditions

\[ u(x, 0) = F(x), v(x, 0) = G(x) \quad x \in [x_1, x_2] \]

and the boundary conditions

\[ u(x_1, t) = u(x_2, t) = 0 \]

\[ v(x_1, t) = v(x_2, t) = 0 \quad t \in [0, T] \]

where \( a \) and \( b \) are constants [2]. It describes the interaction between two long waves with different dispersion relationships and was developed by Hirota and Satsuma [1]. At the same time, in the fluid mechanics of the earth, there are many long waves, such as internal waves and acoustic waves, which are all related to the cKdV equations [3].

The numerical solution of the cKdV equations has attracted much attention of researchers. Halim et al. introduced a general numerical method of the cKdV system [3]. Mehdi Dehghan et al. used the
homotopy analysis method to obtain the numerical solution of cKdV [4]. Khater et al. solved the nonlinear evolution cKdV equations by Lagrangian polynomials spectral method [5]. Assas used the He’s variational iteration method to solve the cKdV equations [6]. For the cKdV equations with periodic initial boundary conditions, Zhu gave a different differentiation scheme [7] and Wazwaz proposed a finite differentiation method [8]. Ma and Zhu combined the Jacobian elliptic function expansion method and Hermite transform to find the exact solution of Wick-type stochastic cKdV equations. Although traditional numerical methods are constantly improving in solving cKdV equations, there has been no satisfactory solution to the convergence and stability of the numerical format.

With the development of deep learning and the increasing of computing power, the application of data-driven methods has been widely used [9-10]. Based on the principle of general function approximation [11], deep learning is good at mining hidden information and has made breakthrough progress in many fields, such as, natural language processing [12] and fault detection [13]. In recent years, more and more researchers have devoted themselves to solving PDEs by deep learning and have achieved the same or even better results as traditional numerical methods. Samuel proposed a method called sparse regression, which uses discrete solutions of the original equation variables as training data for neural networks; through iterative optimization, the partial differential equations corresponding to these data can be found [14]. In these studies, the physical information neural network (PINN) proposed by Raissi has made a very important breakthrough [15]. Different from pure data-driven, its innovation lies in the use of physical constraints in partial differential equations into the loss function of neural networks. In this way, a higher-precision numerical solution can be obtained, and at the same time, the iterative speed of the network can be accelerated. A large number of researches are based on the neural framework of physical information and improved [16-17].

Motivated and inspired by the ongoing research in these areas, a physical neural network is constructed to solve the discretized cKdV equations. The remainder of the paper is organized as follows. In section 2, we introduce the physical information network, and designs the discretization constraint requirements. In section 3, a physical information network model of the cKdV equations is constructed. Finally, the concluding remarks are given in section 4.

2. Methodology

In this section, we first give a brief introduction of feedforward neural network (FNN), and then introduce the discretized equations PINN network with physical constraints.

Since the 1980s, Artificial Neural Network (ANN) has been used as a hot research object of artificial intelligence [18]. It is called ANN because it imitates the neuron transmission information model of the human brain. In this model, neurons are connected to each other to realize the transmission and feedback of complex information. Among them, FNN is a simple network in the field of ANN, and its specific structure is shown in Figure 1. FNN is composed of an input layer, multiple hidden layers, and an output layer. Among them, the parameters from the input layer to the output layer through the hidden layer, and the number of hidden layers, the number of neurons in each layer, and the selection of the activation function all need to be designed with emphasis. In this paper, FNN is used as the approximator of the function.

![Figure 1. A structural diagram of a feedforward neural network (FNN), which consists of an input layer, some hidden layers, and an output layer.](image-url)
Figure 2. The schematic of physics-informed neural network (PINN) for solving one-dimensional cKdV equations.

Different from traditional neural networks rely on pure data, the physical information neural network integrates physical constraints into the network, and forces the output layer to meet the corresponding equations. The ingenious thing is that the equations are embedded in the loss function. So the physical constraints of the equations will all enter the parameter adjustment of the neural network in each back-propagation process. Therefore, it will not only speed up the network training speed, but also increase the interpretability of the network.

The physical information network of one-dimensional cKdV equations is described as:

\[ u_t + N[u;v] = 0 \quad x \in [x_1, x_2], t \in [0, T] \]  \hspace{1cm} (4)

\[ v_t + M[u;v] = 0 \quad x \in (x_1, x_2), \ t \in [0, T] \]  \hspace{1cm} (5)

where \( u, v \) is the solution to be solved and \( N[u;v], M[u;v] \) is a operator (e.g. \( \partial / \partial x, u \partial / \partial x, \partial^2 / \partial^2 x \), etc), and \( B \) denotes the boundary condition of equations.

Then, the general form of the Runge-Kutta method with q-stages [19] is applied to equation (4). In the network, \( f \) and \( g \) are defined as \( -N[u;v] \) and \( -M[u;v] \), respectively ( \( u(x,t) \) as an example). so

\[
\begin{align*}
    u^{n+1} &= u^n - \Delta t \sum_{j=1}^{q} a_j f(u^{n+cj}, v^{n+cj}), i = 1, \ldots, q, \\
    u^{n+1} &= u^n - \Delta t \sum_{j=1}^{q} b_j f(u^{n+cj}, v^{n+cj})
\end{align*}
\]  \hspace{1cm} (6)

Figure 2 shows the schematic diagram of a physics-informed neural network about one-dimensional cKdV equations. The parameters in equation (6) are shared and can be learned by minimizing the sum of squared errors. In the same way, the discrete expressions of \( v(x,t) \) can also obtained. So, the loss function is defined as follows:

\[
\text{Loss} = \text{Loss}_n + \text{Loss}_b
\]  \hspace{1cm} (7)

where

\[
\text{Loss}_n = \sum_{j=1}^{q+1} \sum_{i=1}^{n_f} \left( |u_j^n(x^{i,j}) - u_j^{n,j}|^2 + |v_j^n(x^{i,j}) - v_j^{n,j}|^2 \right)
\]

and
Loss_s = \sum_{i=1}^{q+1} \left( |u^{i+1}(x_i)|^2 + |v^{i+1}(x_i)|^2 \right)

Here, \( \{x^{n,i}, u^{n,i}, v^{n,i}\}_{i=1}^{N_n} \) corresponds to the data at time \( t^n \) and \( N_n \) represents the number of selected training data. \( \{u^{n+1,c}(x_b), v^{n+1,c}(x_b)\} \) is the boundary condition \((x = x_b)\). The gradient optimizer to minimize the loss function (7) by L-BFGS is adopted in this paper, and the following parameters in the physical information neural network are defined as:

\[
\omega = \arg\min_{\omega} L(\omega) \quad b = \arg\min_{b} L(b)
\]  

From the initial data at time \( t^n \) and the boundary data \((x = x_b)\), the neural network about the equation (6) is trained by using the loss function (7), so that the value at time \( t^{n+1} \) can be predicted. After above steps, the neural network of the one-dimensional nonlinear coupled equations solution is developed. Finally, we can update the data in time by using the initial data in \( \{u^{n+2,i}, v^{n+2,i}\}, \{u^{n+3,i}, v^{n+3,i}\}, \) etc.

3. Model and Solution

In this section, we consider the problem that the initial state is the soliton wave solution. The equation (1) has the following exact solution [20]:

\[
u(x,t) = 2\lambda^2 \sech^2 \left( \frac{x}{2\lambda^2 t} \right), \quad \nu(x,t) = \frac{1}{2\lambda^2 \sech \left( \frac{x}{\lambda^2 t} \right)}
\]

where

\[
\xi = \lambda \left( x - \lambda^2 t \right) + \frac{1}{2\lambda^2 \sech \left( \frac{x}{\lambda^2 t} \right)}, \quad \omega = -\frac{b}{8(4a+1)\lambda^2}
\]

In this paper, we take \( a = -1/8, b = -3, \lambda = 0.5 \) for equations (1),(9),(10) and construct the training data set from this.

Then, in the physical information neural network introduced in section 2, \( f, g \) in equation [1] is:

\[
f = -1/8u_{xst} - 3/4uv_{x} - 6v_{xx}, \quad g = -v_{xxt} - 3uv_{x}
\]

Table 1. Pseudo code of physical information neural network for learning the solution of the cKdV equations, among them, self.neural_net0 is the definition of the FNN network required in the network, and U1 and V1 are the results of the output layer of the FNN network. F and G respectively represent the non-linear operators that \( u \) and \( v \) conform to, and U0 and V0 are the solutions after physical constraints.

```python
def net_U0(self, x):
    A = self.neural_net0(x, self.weights, self.biases)
    U1 = A[:, 0:q + 1]
    U = U1[:, :-1]
    U_x = self.fwd_gradients_0(U, x)
    U_xx = self.fwd_gradients_0(U_x, x)
    U_xxx = self.fwd_gradients_0(U_xx, x)
    F = -1/8 * U_xxx - 3/4 * U * U_x - 6 * V * V_x
    U0 = U1 - self.dt * tf.matmul(F, self.IRK_weights.T)
    return U0, V0
```
Table 1 shows the python code snippets of physical information neural network for learning the solution of the cKdV equations. The results of applying this physical information neural network are presented in Figure 3. The training data is made up of 100 initial data points and is given by Equation (9). These points are randomly selected from interval \([x_1, x_2]\) at time \(t = 1.01\). The goal is to predict and solve the value of \((u, v)\) at \(t = 8.08\), the single time step \(\Delta t = 7.07\). The discretized time physical information neural network we use has 3 hidden layers with 50 neurons in each layer. The output layer is respectively \((q+1)\) values of \((u, v)\), corresponding to \((u^{n+c_1}(x), v^{n+c_1}(x))\), \(i = 1, \ldots, q\) of \(q = 500\) Runge-Kutta stages and \((u^{n+1}(x), v^{n+1}(x))\) at the predicted moment. At a single time-step, the resulting prediction error of \((u, v)\) is measured at \((3.7 \times 10^{-4}, 1.2 \times 10^{-3})\) in the relative L2-norm. The reason why such a small error can be obtained is mainly due to the approximate ability of the physical information neural network, and the loss allows the data to be interpolated.

In order to analysis the performance of different network architectures, Table 2 gives relative L2 error for different number of hidden layers and neurons per layer when the stage of Runge-Kutta \(q = 500\) and the time step \(\triangle t = 8.08\). It can be concluded that the prediction accuracy improves as the number of layers and neurons increases (the ability of neural networks to approximate more complex functions increases).

![Figure 3. The top two: Solution \((u(t, x), v(t, x))\)along with the location of the initial training snapshot at \(t = 1.01\) and the final prediction snapshot at \(t = 8.08\). The bottom two: Initial training data and final prediction at the snapshots depicted by the white vertical lines in the top panel.](image)

| Neurons | Layers | 10     | 25     | 50     | 10     | 25     | 50     |
|---------|--------|--------|--------|--------|--------|--------|--------|
| 1       | 1.93e-02 | 8.28e-03 | 1.47e-03 | 4.71e-02 | 1.05e-02 | 3.17e-02 |
| 2       | 2.03e-03 | 1.03e-03 | 7.76e-04 | 3.13e-03 | 1.51e-03 | 1.51e-03 |
| 3       | 2.22e-03 | 7.46e-04 | 1.36e-03 | 2.04e-03 | 1.69e-03 | 2.09e-03 |

The parameters Runge-Kutta stages \(q\) and the time step \(t\) are very important. Table 3 lists relative L2 error of PINN, which has 3 hidden layers and each has 50 neurons, for different numbers of Runge-Kutta stages \(q\) and the time step \(t\). It can be found that relative L2 error increases with the increase of the time step and quickly converges with the increase of \(q\).
Table 3. Relative L2 error of PINN for different numbers of Runge-Kutta stages q and the time step t which has 3 hidden layers and each layer has 50 neurons.

| q   | 2.02 | 4.04 | 6.06 | 8.08 | 2.02 | 4.04 | 6.06 | 8.08 |
|-----|------|------|------|------|------|------|------|------|
| 1   | 3.74e-03 | 2.48e-02 | 7.75e-02 | 1.38e+00 | 5.99e-03 | 1.85e-02 | 5.36e-02 | 2.45e+00 |
| 16  | 3.51e-04 | 1.36e-03 | 2.31e-03 | 1.36e-03 | 1.82e-03 | 2.87e-03 | 2.35e-03 | 3.66e-03 |
| 100 | 3.52e-04 | 9.56e-04 | 7.56e-04 | 6.62e-04 | 1.05e-03 | 1.14e-03 | 1.04e-03 | 1.83e-03 |
| 500 | 4.17e-04 | 2.78e-04 | 9.54e-04 | 4.07e-04 | 1.39e-03 | 1.02e-03 | 3.96e-03 | 1.13e-03 |

4. Conclusion

In this paper, we have introduced the physical information neural network (PINN) method to find the solution of one-dimensional coupled KdV equations and discussed the performance of PINN with different neural network architectures and parameters. The PINN method can solve the cKdV equations with reasonable errors by training a small amount of data. When we use the physical information neural network containing 3 hidden layers with 50 neurons in each layer, relative L2 errors of (u, v) are (3.7∙10^-4, 1.2∙10^-3) from the initial time at t =1.01 to the forecast time at t =8.08. In addition, we systematically studied the performance of different network structures and the influence of Runge-Kutta stages q and the time step t.

Different from the traditional neural network, PINN method adds the equations to the training of the network instead of pure data driving, which can obtain accurate solutions and strengthen the interpretability of the neural network. At the same time, compared with the previous numerical methods, the Runge-Kutta stages can be increased to 500, and the single time step can be significantly increased. It can be believed that the PINN method will provide a new way for solving various coupled differential equations, which greatly promote the development of the field of computational science. This paper only solves the cKdV equations with single soliton solution. In the future, we will conduct further research on the cKdV equations with multiple soliton solution.

References

[1] Roy, P. K.. (1998). On coupled kdv equations. Physics Letters A, 249(1-2), 55-58.
[2] Hirota, R., & Satsuma, J.. (1981). Soliton solutions of a coupled korteweg-de vries equation. Physics Letters A, 85(8-9), 407-408.
[3] Halim, A. A., Kshevetskii, S. P. and Leble, S. B.. (2003). Numerical integration of a coupled Korteweg-de Vries system, Computers & Mathematics with Applications, 45 (4-5), 581-591.
[4] Dehghan, M., Manafian, J. , & Saadatmandi, A. . (2010). Solving nonlinear fractional partial differential equations using the homotopy analysis method. Numerical Methods for Partial Differential Equations, 26(2), 448-479.
[5] Khater A H, Temsah R S, Callebaut D K . (2008). Numerical solutions for some coupled nonlinear evolution equations by using spectral collocation method[J]. Mathematical & Computer Modelling, 48(7-8):1237-1253.
[6] Assas, L.. (2008). Variational iteration method for solving coupled-kdv equations. Chaos Solitons & Fractals, 38(4), 1225-1228.
[7] Zhu, S.. (1999). A difference scheme for the coupled kdv equation. Communications in Nonlinear Science and Numerical Simulation, 4(1), 60-63.
[8] Wazwaz, A. M.. (2008). Handbook of Differential Equations Evolutionary Equations, Elsevier, The Netherlands.
[9] Lecun, Y., Bengio, Y., & Hinton, G.. (2015). Deep learning. Nature, 521(7553), 436-444.
[10] Bai, Z.; Brunton, S.L.; Brunton, B.W.; Kutz, J.N.; Kaiser, E.; Spohn, A.; Noack, B.R..(2017). Data-driven methods in fluid dynamics: Sparse classification from experimental data. In
Whither Turbulence and Big Data in the 21st Century? Springer: Berlin/Heidelberg, Germany, 2017; pp. 323–342.

[11] Lu, Y.; Lu, J. (2020). A Universal Approximation Theorem of Deep Neural Networks for Expressing Distributions, arXiv:2004.08867.

[12] Goldberg, Y. (2016). A primer on neural network models for natural language processing. Computer Science, 57, 345–420.

[13] Helbing, G.; Ritter, M. (2018). Deep Learning for fault detection in wind turbines. Renew. Sustain. Energy Rev. 98, 189–198.

[14] Rudy, S. H., Brunton, S. L., Proctor, J. L., & Kutz, J. N. (2016). Data-driven discovery of partial differential equations. Science Advances, 3(4).

[15] Raissi, M., Perdikaris, P., & Karniadakis, G. E. (2018). 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.

[16] Wight, C. L., & Zhao, J. (2020). Solving Allen-Cahn and Cahn-Hilliard equations using the adaptive physics informed neural networks. Communications in Computation Physics. 29, 930-954.

[17] Lu, L., Jin, P., Pang, G., Zhang, Z., & Karniadakis, G. E. (2021). Learning nonlinear operators via deepnet based on the universal approximation theorem of operators. Nature Machine Intelligence, 3(3), 218-229.

[18] Shanmuganathan, S. (2016). Artificial neural network modelling: an introduction. Applied Mathematics and Computation. 203(3), 482-492.

[19] Iserles, A. (2009). A First Course in the Numerical Analysis of Differential Equations (Cambridge Texts in Applied Mathematics). Cambridge University Press. London.

[20] Oru, M., Bulut, F., & Esen, A. (2017). A numerical treatment based on haar wavelets for coupled kdv equation. An International Journal of Optimization and Control Theories & Applications (IJOCTA), 7(2), 195.