© 2025 Institute of Theoretical Physics CAS, Chinese Physical Society and IOP Publishing.
All rights, including for text and data mining, AI training, and similar technologies, are reserved. Printed in China

Commun. Theor. Phys. **77** (2025) 065002 (9pp)
https://doi.org/10.1088/1572-9494/ada3ce
CSTR: 32041.14.CTP.ada3ce

A symmetric difference data enhancement physics-informed neural network for the solving of discrete nonlinear lattice equations

Jian-Chen Zhou, Xiao-Yong Wen and Ming-Juan Guo

School of Applied Science, Beijing Information Science and Technology University, Beijing 100192, China

E-mail: xiaoyongwen@163.com

Received 5 October 2024, revised 7 December 2024
Accepted for publication 30 December 2024
Published 18 March 2025

Abstract

In this paper, we propose a symmetric difference data enhancement physics-informed neural network (SDE-PINN) to study soliton solutions for discrete nonlinear lattice equations (NLEs). By considering known and unknown symmetric points, numerical simulations are conducted to one-soliton and two-soliton solutions of a discrete KdV equation, as well as a one-soliton solution of a discrete Toda lattice equation. Compared with the existing discrete deep learning approach, the numerical results reveal that within the specified spatiotemporal domain, the prediction accuracy by SDE-PINN is excellent regardless of the interior or extrapolation prediction, with a significant reduction in training time. The proposed data enhancement technique and symmetric structure development provides a new perspective for the deep learning approach to solve discrete NLEs. The newly proposed SDE-PINN can also be applied to solve continuous nonlinear equations and other discrete NLEs numerically.

Keywords: symmetric difference data enhancement physics-informed neural network, data enhancement, symmetric point, soliton solutions, discrete nonlinear lattice equations

(Some figures may appear in colour only in the online journal)

1. Introduction

In the rapidly evolving era of artificial intelligence (AI), the application of AI in mathematics and physics, particularly in the study of nonlinear partial differential equations (NPDEs), has become a pivotal issue. The research on NPDEs is of paramount importance because they play a central role in describing natural phenomena and engineering systems. Many scientific fields rely on these equations to model complex dynamical processes, such as fluid dynamics, heat conduction, and biophysical processes. Through emerging technologies such as machine learning, researchers are exploring how to discover and solve these equations more effectively. Many scientific fields rely on these equations to model complex dynamical processes, such as fluid dynamics, heat conduction, and biophysical processes. Researchers have developed a computation method based on physical principles —physics-informed neural network (PINNs) [1], which has garnered significant attention.

Innovations in this field continue to emerge, including the optimization of the architecture of fully connected neural network models [2–4], the improvement of loss functions [5, 6], the optimization of training processes [7, 8] and the enhancement of sampling points based on residuals and gradients [9, 10]. Research has introduced a PINN method for the Hausdorff fractional order derivative Poisson equation, showing enhanced flexibility and accuracy in irregular domains and non-uniform node distributions [11]. Furthermore, a multi-domain PINN has been developed, utilizing domain decomposition to divide the problem into subdomains and applying continuity conditions between sub-networks to solve forward and inverse problems [12, 13].

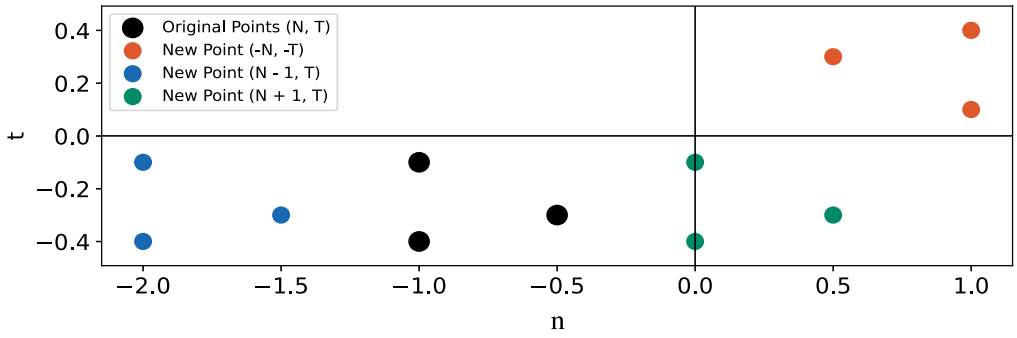

Figure 1. The data enhancement process. Black is the initial point, other colors are new points generated by data enhancement.

To improve the training efficiency and solution accuracy of PINNs in addressing NPDEs, researchers have devised several adaptive enhancement algorithms [4, 14–16]. Certain investigations have focused on employing deep neural networks constrained by predefined control equations to optimize all parameters, thus effectively elucidating solitary waves and their interaction dynamics [14]. One research group introduced a methodology that integrates weighted PINNs with adaptive residual point distribution, which enhances adaptability to steep gradient solutions through the dynamic selection of training points [17]. Furthermore, a separate research team has advanced an algorithm that integrates gradient-enhanced PINNs with adaptive mixed sampling. This algorithm attains high-precision solutions for variable coefficient resonant nonlinear Schrödinger equations under non-uniform parameter conditions by embedding residual gradients into the loss function and adaptively modifying the distribution of training points [18].

Moreover, the macroscopic properties of the solutions, such as symmetry and conservation laws, are crucial for physical constraints, but research in these areas remains limited. Existing studies have proposed integrating PT symmetry into the loss function for optimization [19] and incorporating conservation law optimizations [20].

Current research primarily focuses on continuous NPDEs, with relatively less exploration in discrete nonlinear lattice equations (NLEs). However, recent studies have begun to explore architectures based on pseudo-grids for the study of discrete NLEs [21]. To further delve into discrete NLEs and improve the solution accuracy, we propose a streamlined processing architecture termed symmetric difference data enhancement physics-informed neural network (SDE-PINN). The core concepts underlying this architecture are as follows:

- Employing data enhancement techniques to create datasets that reflect both differential and symmetric demands, which are subsequently used as inputs for the neural network, can significantly enrich the data without being restricted to discrete equidistant sampling points.
- The outputs generated by the neural network facilitate efficient differential computations by converting the differentiation operations within the continuous model's loss function into forward and backward difference operations in the discrete model. This approach reduces computational complexity and enhances efficiency.
- Building upon the symmetrical properties inherent in the solutions of physical equations, this study proposes an adaptive symmetric point learning method. By dynamically selecting symmetric points and imposing symmetry constraints on the network outputs, the model's solutions are ensured to adhere to the principles of physical symmetry.

The structure of this paper is as follows: section 2 will provide a detailed introduction to the SDE-PINN. Section 3 will discuss the prediction and extrapolation of the discrete KdV equation under both known and unknown symmetric points. Section 4 will discuss our model and existing discrete neural network models for an already solved discrete Toda lattice equation. The final section will sum up the results and provide conclusions.

2. Model description

In this section, we describe how to solve the discrete NLEs by building and evaluating the SDE-PINN. The process is divided into three main parts.

2.1. Data enhancement

In terms of input, discrete NLEs require training and prediction on integer grid points. Given the irregular distribution of solitary points near peaks, additional prior information is needed. Inspired by [9], we use firstly the Hammersley method to generate data NT. Then we map it to create symmetrical $-NT$ using symmetric point P_s and average both inputs to achieve outputs. This method satisfies the equation's symmetry requirements. The symmetric point P_s can be preset or learned through a neural network. Different from the space discretization for NPDEs, we need to create additional datasets via shift operations (i.e., $Ef(n, t) = f(n + 1, t)$ represents the forward shift, while $E^{-1}f(n, t) = f(n - 1, t)$ denotes the backward shift) and simulate spatial differences using neural network outputs. The data enhancement process is depicted in Figure 1.

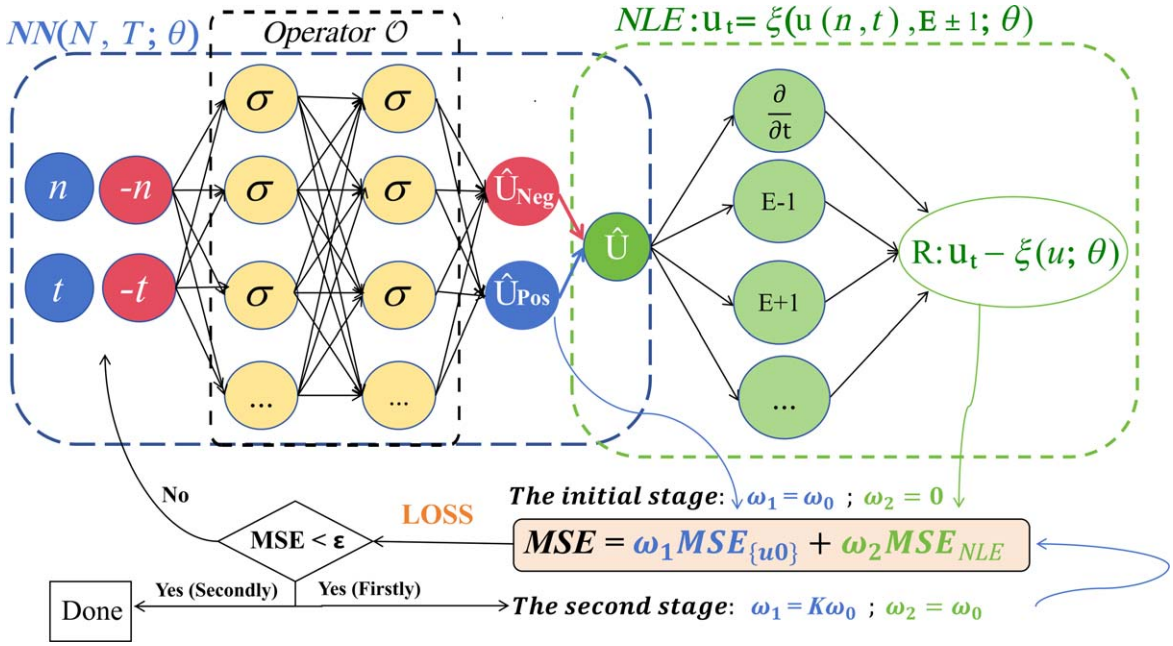

Figure 2. SDE-PINN architecture diagram.

2.2. Training process and optimization strategies

The first step is to impose the hard constraint on the boundary conditions. During the training process, the hard constraints on the boundary conditions cause their corresponding loss components to approach zero, considering the inherent errors in data values. Therefore, the discussion henceforth only focuses on the initial and internal points.

This diagram illustrates the architecture of the neural network, where the neural network operator $\mathcal{O}$ processes inputs NT and its negative counterpart $-NT$. The parameters of the neural network, denoted by θ, include the weights and biases that are learned during training to optimize the network's performance. The architecture is designed to leverage symmetry in the input data, which can enhance the model's performance by effectively capturing underlying patterns. Below are the key equations defining the operation and components of the system. The combined output of the neural network is defined as

$$\hat{U}_{(N,T;\theta)} = \frac{1}{2}[\mathcal{O}(NT) + \mathcal{O}(-NT)]. \tag{1}$$

The term $E \pm i$ represents a differential operator applied to the system's evolution, specifically defined as

$$E \pm i = \hat{U}_{(N\pm i,T;\theta)} - \hat{U}_{(N,T;\theta)}, \quad i = 1, 2, 3,.... \tag{2}$$

The loss function used to optimize the neural network parameters θ is structured as follows

$$\text{Loss}(\theta) = w_1 \cdot \text{MSE}_{u0} + w_2 \cdot \text{MSE}_{\text{NLE}}, \tag{3}$$

where w_1 and w_2 are the weights for the respective mean squared error (MSE) terms. The two MSE terms are defined as follows

$$\text{MSE}_{u0}(\theta) = \frac{1}{N_0}\sum_{i=1}^{N_0} |\hat{U}(n_i, t_i; \theta) - U_i|^2,$$
$$\text{MSE}_{\text{NLE}}(\theta) = \frac{1}{N_f}\sum_{j=1}^{N_f} |u_t - \xi(u, E \pm i)|^2, \tag{4}$$

in which N_0 is the number of initial condition samples, N_f is the number of samples at which the NLEs residual is evaluated, $\hat{U}(t_i, n_i)$ represents the predicted output at the initial condition, U_i is the actual output at the initial condition, and $\xi(u, E \pm i)$ represents a nonlinear function.

The second is the optimisation of the training process. Initially, the training optimizes solely for the initial points, adhering to the causal sequence. This approach not only trains points that are already accurately determined but also substantially speeds up the training process, rapidly reducing the loss function within a very short period. Once the initial conditions are satisfied through preliminary training, the second phase begins. This phase introduces a regularization penalty term for NLEs, focusing on training the internal points. Additionally, a significantly larger weight ω is assigned to the loss function term for initial conditions to ensure that after satisfying the initial conditions, the NLEs penalty term is utilized to further optimize training for internal points. This method significantly reduces time and conforms to the actual causal training sequence.

The complete SDE-PINN architecture is shown in Figure 2.

In terms of setting the weights of the components of the loss function, ω_0 denotes an initial weight, typically set to 1, while K represents a large weight, often chosen as 1×10^3 in the model.

2.3. Network settings and effectiveness evaluation

In the presented numerical outcomes for the discrete NLEs, the network setup used includes a two-dimensional input layer that processes spatiotemporal data. In the intermediate hidden layers, four fully connected layers with eight neurons each are chosen. The optimizers chosen are Adam and LBFGS, which adaptively switch based on changes in the loss function. The learning rates are set at 1×10^{-3} for Adam and 1 for LBFGS.

The model's performance is assessed using the relative error (RE) as the evaluation metric. The RE is defined in a discrete structure as follows

$$\mathrm{RE}=\frac{\sum_{i=0}^{P_t-1}\sum_{j=0}^{P_n-1}|u_{i,j}-\hat{u}_{i,j}|}{\sum_{i=0}^{P_t-1}\sum_{j=0}^{P_n-1}|u_{i,j}|}, \tag{5}$$

where P_t is the total number of time points, P_n is the total number of spatial points, $u_{i,j}$ is the actual value at time t_i and space n_j, $\hat{u}_{i,j}$ is the predicted value at time t_i and space n_j.

We utilize the time interval $[-t_{\mathrm{Tra}}, t_{\mathrm{Tra}}]$ for training and prediction. To evaluate the model's predictive performance, we select the time intervals $[-t_{\mathrm{Ext}}, -t_{\mathrm{Tra}}]$ and $[t_{\mathrm{Tra}}, t_{\mathrm{Ext}}]$ for extrapolation. Within the training range, the RE is denoted as $\mathrm{RE}_{\mathrm{Int}}$. For the extrapolation range, the RE is denoted as $\mathrm{RE}_{\mathrm{Ext}}$.

The code is run in a Python 3.11.7 setting, utilizing PyTorch version 2.3.1+cu121, on a high-performance computer outfitted with an NVIDIA GeForce RTX 3060 Laptop GPU.

3. Numerical simulation of discrete KdV equation

The first example in this section numerically simulates the discrete KdV equation in [22], which can correspond to the famous KdV equation through the continuous limit. The discrete KdV equation and its initial boundary problem are considered below

$$\begin{cases} u_{n,t}=u_n^2(u_{n+1}-u_{n-1}), \\ u_n(t_0)=u(n,t_0), \\ u_1=u_{b1}, \quad u_N=u_{b2}. \end{cases} \tag{6}$$

3.1. Known symmetric point: one-soliton solution of equation (6)

The initial values are obtained from the exact analytical solution in [22], from which one-soliton solution is given by

$$u_n=U(n,t)=1+\frac{(\lambda_1^2-4)}{4} \times \mathrm{sech}^2\left[n\ln\left(\frac{\lambda_1+\sqrt{\lambda_1^2-4}}{2}\right)+\frac{\lambda_1\sqrt{\lambda_1^2-4}}{2}t\right]. \tag{7}$$

Here we choose $\lambda_1=4$ and $N=[-10, 10]$ representing a finite one-dimensional lattice. We set $t_{\mathrm{Tra}}=1$ and $t_{\mathrm{Ext}}=1.5$, initializing the system at $t=-1$ as per equation (6) to establish initial conditions. The boundary values for the potential function u_n are set as $u_{b1}=u_{b2}=1$. Training and prediction occur within $R_{\mathrm{in}}=\{(n,t)|n\in[-10,10], t\in[-1,1]\}$, while extrapolation tests are conducted in $R_{\mathrm{out}}=\{(n,t)|n\in[-10,10], t\in[-1.5,-1]\cup[1,1.5]\}$ to evaluate the model's extrapolative capabilities.

In the first phase, the Adam optimizer is initially used, and after 300 training iterations, it is switched to the LBFGS optimizer. Subsequently, the loss function value rapidly decreases to 1.54273×10^{-6}, with a total of 400 epochs completed in 10.0 seconds.

In the second phase, the Adam optimizer is used for the first five iterations. Due to the application of a large initial weight and the addition of an NLEs penalty term, the loss function initially increases. After 405 iterations, the optimizer is switched to LBFGS, and after 410 epochs, the loss function reaches 9.12466×10^{-7}, with the initial value residual being 5.76309×10^{-10} and the NLEs penalty term value being 3.36157×10^{-7}. The second phase involves 50 epochs and takes 119.6 seconds.

The final REs are $\mathrm{RE}_{\mathrm{Int}}=6.1314 \times 10^{-5}$ in R_{in}, and $\mathrm{RE}_{\mathrm{Ext}}=5.6612 \times 10^{-5}$ in R_{out}. The spatiotemporal domain for model training is illustrated in figure 3. It is evident that under prior symmetric point conditions, the SDE-PINN model achieves very high accuracy with rapid convergence.

3.2. Unknown symmetric point: two-soliton solution of equation (6)

In more cases, we do not know the prior symmetric points, at which point the coordinates of the symmetric points can be added as parameters to the model training. For the selection of the initial point, a preliminary model can be trained to calculate or estimate an approximate symmetric point coordinate. We train our model by using a two-soliton solution of equation (6) in [22] as follows

$$u_n=U(n,t)=1-\frac{2(\lambda_1^2-\lambda_2^2)[(\lambda_2^2-4)\cosh(2\xi_1)+(\lambda_1^2-4)\cosh(2\xi_2)-\lambda_1^2+\lambda_2^2]}{[(\lambda_1\sqrt{\lambda_2^2-4}-\lambda_2\sqrt{\lambda_1^2-4})\cosh(\xi_1+\xi_2)+(\lambda_1\sqrt{\lambda_2^2-4}+\lambda_2\sqrt{\lambda_1^2-4})\cosh(\xi_1-\xi_2)]^2}, \tag{8}$$

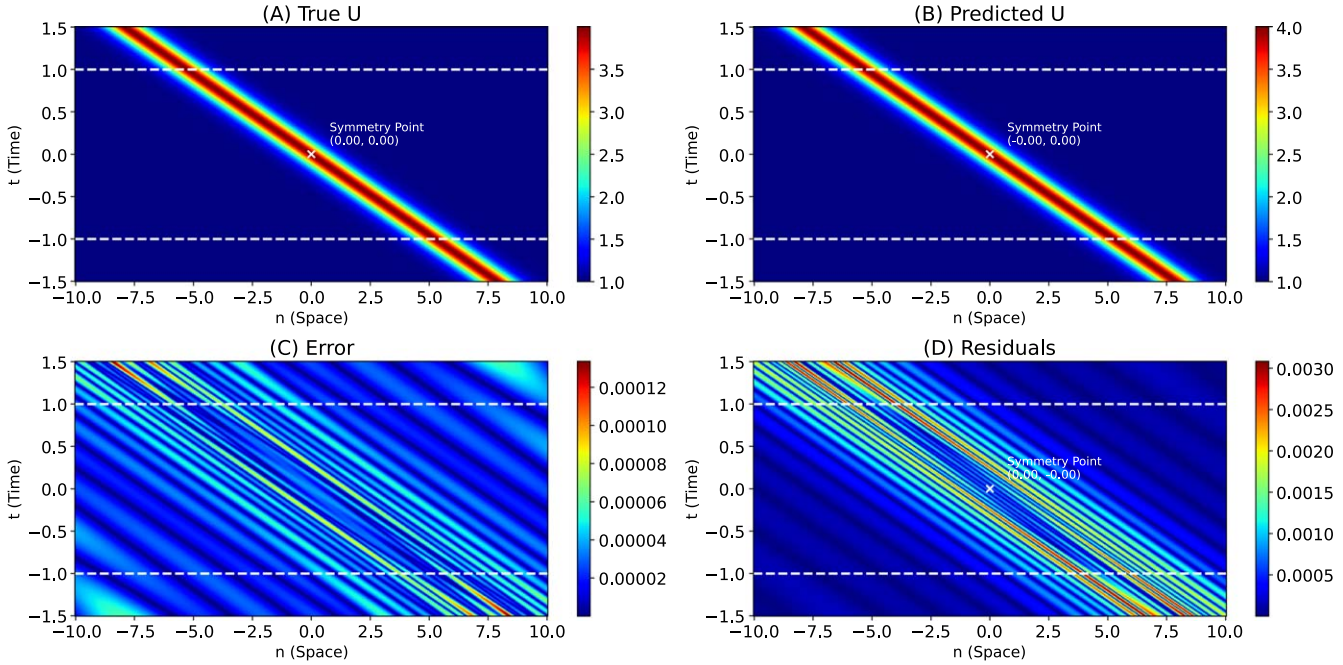

Figure 3. Numerical solution of equation (6) for one-soliton solution with known symmetric point. (A) shows the true density map of u_n; (B) exhibits the model output of u_n, where inside the white dashed line is the predicted range and outside the white dashed line is the extrapolated range; (C) shows the absolute value of the difference between the true and predicted values; (D) shows the value of the NLEs penalty term in the loss function.

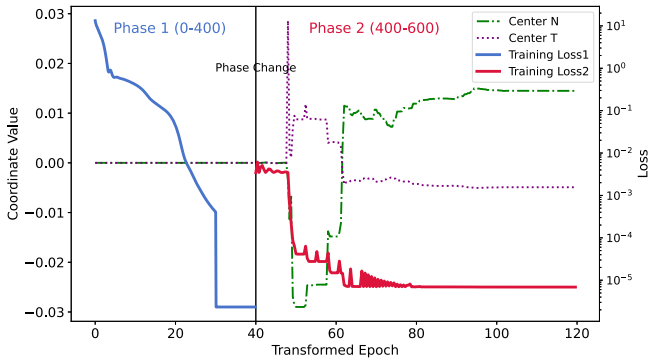

Figure 4. The training process diagram, where the straight line is the change value of the loss function, the dashed line is the unknown symmetric point coordinate evolution, and the vertical line indicates the optimizer transformation, as well as the stage transformation during the training process.

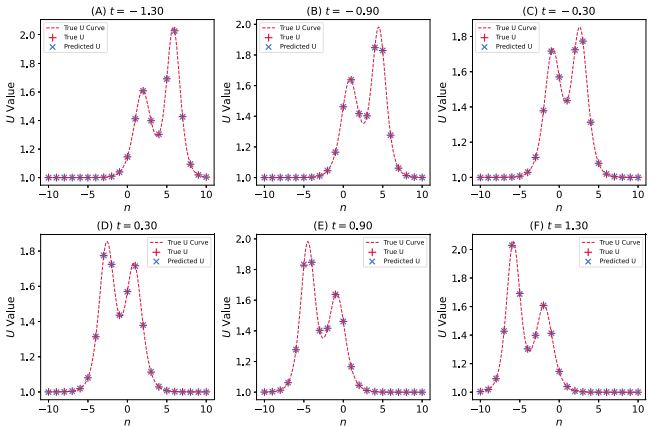

Figure 5. The time evolution of a two-soliton solution at six different space-time points.

where $\xi_i = n \ln\left(\frac{\lambda_i + \sqrt{\lambda_i^2 - 4}}{2}\right) + \frac{\lambda_i \sqrt{\lambda_i^2 - 4}}{2} t$ for $i = 1, 2$. Here we choose $\lambda_1 = \frac{5}{2}$, $\lambda_2 = 3$ and keep other settings consistent with those described in section 3.1.

Firstly, we train a SDE-PINN model without a symmetric point as a preprocessing step, then observe and estimate that the symmetric point P_s of the two-soliton solution is near the origin. In this stage, we initially set P_s at (0,0), considering that the first phase only involves training initial values, so the symmetric point parameter is not included in the model. However, upon entering the second phase, this symmetric point will be integrated into the model as an important parameter of the neural network.

The training process is illustrated in figure 4. Observation of the centroid coordinate changes reveals that incorporating the coordinates as neural network parameters into the training can be more effective in adaptively finding symmetric point P_s. The actual symmetric point is (0,0) and the adaptively found symmetric point is (0.0144, -0.0049). For ease of observation, the epochs of the first phase are scaled by dividing by 10, compressing the range from [0,400] to [0,40], while the second phase epochs are represented as actual counts, from [400,600] to [40,120]. The first phase takes 4.4s and the second phase takes 81.7s. The final REs are $\mathrm{RE_{Int}} = 7.5230 \times 10^{-4}$ in R_{in}, and $\mathrm{RE_{Ext}} = 9.6901 \times 10^{-4}$ in R_{out}.

Six space-time points are equidistantly selected for visualization in figure 5, and it is found that the model's

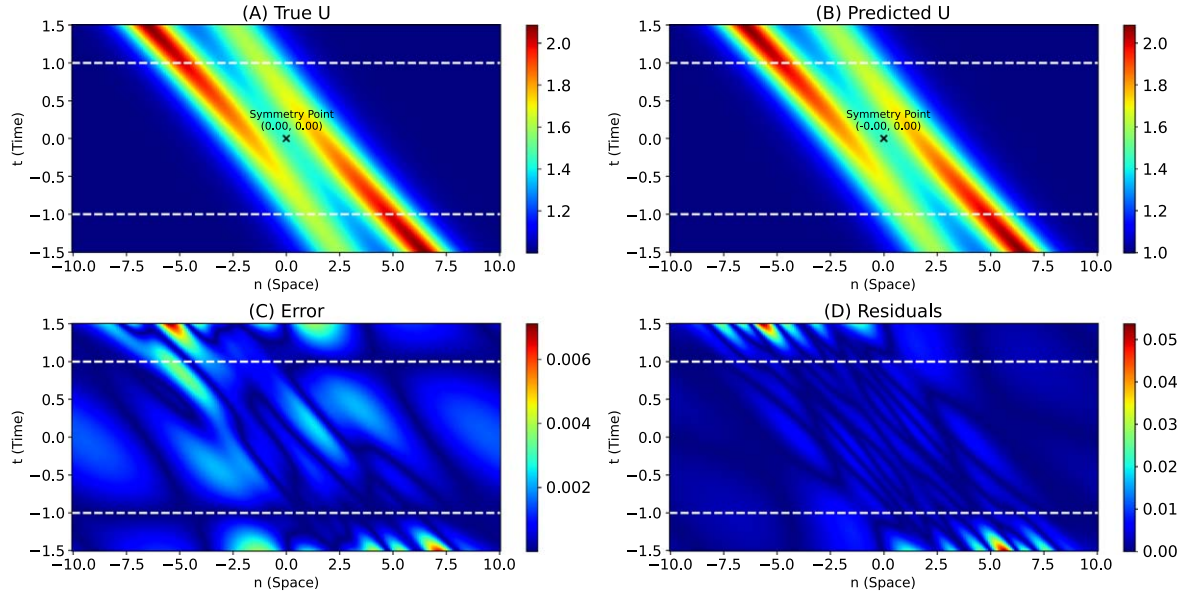

Figure 6. Numerical solution for a two-soliton solution with unknown symmetric point of equation (6).

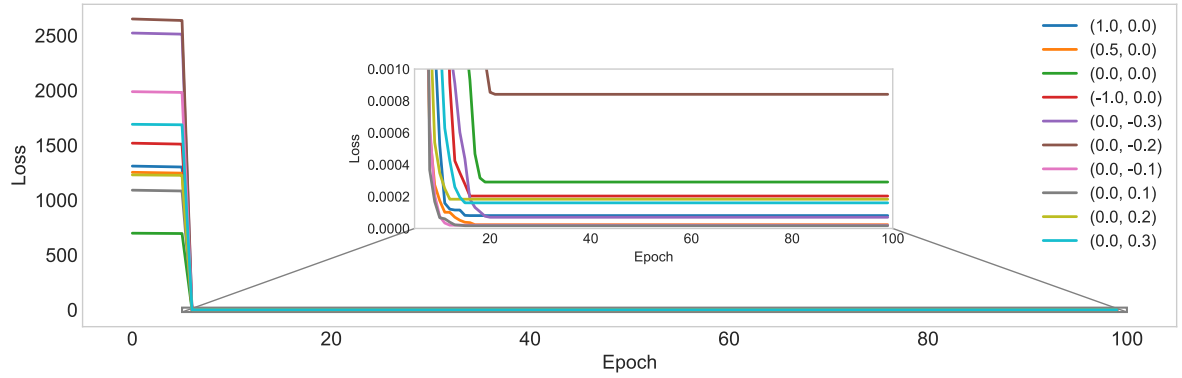

Figure 7. The impact of different initial symmetric point selections on the loss function evolution.

fitting effect is also very good in the case of a two-soliton solution. The spatiotemporal domain for model training is illustrated in figure 6. As observed in figure 6 (C), the predictive performance within the designated range is notably robust, with the potential for minor extrapolations.

To quantitatively analyze the impact of the initial point selection, we conducted a grid-based numerical experiment, focusing on points near the true symmetric point (0, 0). In the experiment, the first five training iterations employed the Adam optimization method, while the subsequent iterations utilized the LBFGS method. Each group was trained for 100 iterations. The variations in the loss function are depicted in figure 7. It can be observed that, with the exception of the initial point (0.0, − 0.2), which results in a higher loss value compared to the true value 0, 0, the loss values for other initial points are relatively smaller. It is noteworthy that the loss function values for three particular initial points are of the magnitude E-6, specifically: for the point (0.5, 0.0), the value is 2.35×10^{-5}; for the point (0.0, 0.1), the value is 2.06×10^{-5}; and for the point (0.0, -0.1), the value is 1.72×10^{-5}.

Further predictions and extrapolations of the model are presented, with the results visualized in figure 8. It is evident that the selection of the initial symmetric point has a substantial impact on the training performance of the model. Specifically, when the initial symmetric point is aligned with the true symmetric point, the model's performance, given the current parameters and training conditions, is less favorable compared to scenarios where alternative initial points are utilized. Furthermore, the model's performance demonstrates a heightened sensitivity to the selection of time coordinates, as indicated by the significant fluctuations in relative error observed across the seven distinct time points.

4. Model comparison through the Toda lattice equation

To further evaluate the performance of the SDE-PINN, we continue to compare to existing discrete soliton learning models, specifically against the PG-PhyCRNet proposed in [21] under the no pseudo-grid condition through the Toda

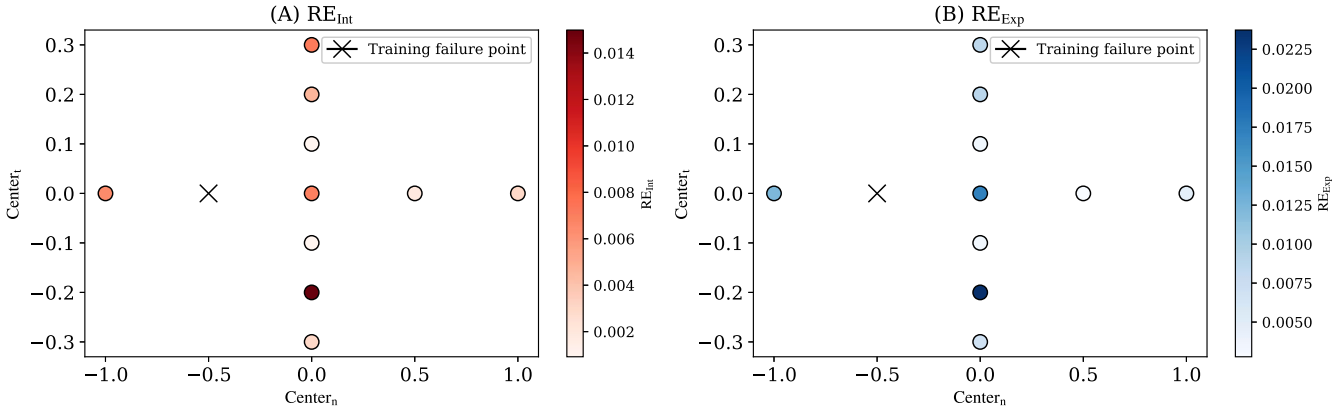

Figure 8. The RE_{Int} and RE_{Exp} under different initial symmetry point selection conditions. (A) depicts the heatmap of prediction errors at various coordinates. (B) shows the heatmap of extrapolation errors at different coordinates. In both panels, lighter colors represent smaller relative errors, indicating better model performance during training.

Table 1. Comparison of RE for different models.

Model	RE_{Int}			RE_{Ext}			Times
	u_n	v_n	Total	u_n	v_n	Total	
PG-PhyCRNet	**2.0567724E-05**	**2.2417906E-04**	**2.4474679E-04**	1.0692835E-03	1.3986321E-02	1.5055604E-02	1003 s
SDE-PINN(1)	1.0501234E-04	1.2644633E-03	1.3694756E-03	1.2041807E-04	1.5070488E-03	1.6274669E-03	**156 s**
SDE-PINN(2)	4.4054545E-05	5.0257038E-04	5.4662493E-04	**7.9619436E-05**	**9.1176821E-04**	**9.9138765E-04**	622 s

lattice equation

$$\begin{cases} u_{n,t} = u_n(v_n - v_{n-1}), \\ v_{n,t} = u_{n+1} - u_n, \\ u_n(t_0) = u(n, t_0), \quad v_n(t_0) = v(n, t_0), \\ u_1 = u_{b1}, \quad u_N = u_{b2}, \quad v_1 = v_{b1}, \quad v_N = v_{b2}. \end{cases} \tag{9}$$

Next, we adopt the same initial-boundary conditions and space-time range used in handling equation (9) by that model in [21], where the space-time domain NT: $[-12, 13] \times [-0.5, 0]$ is utilized for model training and prediction, while NT: $[-12, 13] \times [0, 0.5]$ is used for extrapolation. Considering that the symmetric points in the solutions for u_n and v_n are not consistent, we initially set both symmetric points P_{su} and P_{sv} to $(0, -0.25)$. Subsequently, these two distinct symmetric points are incorporated as parameters into the model.

As shown in table 1, we compare the three models: PG-PhyCRNet, SDE-PINN(1) (with only differential enhancement), and SDE-PINN(2) (with symmetric differential enhancement for unknown symmetric points). We observe that SDE-PINN(1), which employs differential enhancement alone, achieves a total relative error ($RE_{Int,\ Total}$) of 1.3695E-03 for interior points, which is higher than PG-PhyCRNet's 2.4475E-04, but still reasonable. In extrapolation, SDE-PINN(1) achieves a total relative error ($RE_{Ext,\ Total}$) of 1.6275E-03, which is significantly smaller than PG-PhyCRNet's 1.5056E-02. This shows that SDE-PINN(1) performs better in extrapolation with nearly no significant overfitting phenomena and consistent results between interior prediction and extrapolation.

Further incorporating symmetric enhancement, SDE-PINN (2) approaches the predictive performance of PG-PhyCRNet. SDE-PINN(2) achieves a total relative error ($RE_{Int,\ Total}$) of 5.4662E-04 for interior points, which is still slightly larger than PG-PhyCRNet's 2.4475E-04, but performs significantly better in extrapolation, achieving a total extrapolation relative error ($RE_{Ext,\ Total}$) of 9.9139E-04, much lower than PG-PhyCRNet's 1.5056E-02.

Regarding the time cost, the SDE-PINN models employing data augmentation demonstrate superior performance compared to the PG-PhyCRNet. Notably, SDE-PINN(1) achieves the shortest training duration of 156 seconds, whereas SDE-PINN(2) requires 622 seconds. In contrast, PG-PhyCRNet necessitates 1003 seconds for training. The spatiotemporal domain utilized for training the SDE-PINN(2) model is depicted in figure 9. The neural network is capable of determining the coordinates of P_{su} and P_{sv} as $(1.260, -0.25)$ and $(0.759, -0.25)$, respectively. In comparison, the true coordinates are $(1, -0.25)$ for P_{su} and $(0.5, -0.25)$ for P_{sv}. It can be observed that when symmetric point coordinates are used as parameters in SDE-PINN, the network accurately learns the true coordinates, which not only speeds up the training process but also enhances the precision of the training.

Compared with the existing discrete soliton learning method PG-PhyCRNet, the SDE-PINN demonstrates a significant advantage in terms of time efficiency and extrapolation capabilities. While the prediction accuracy remains comparable, our model exhibits minimal signs of overfitting.

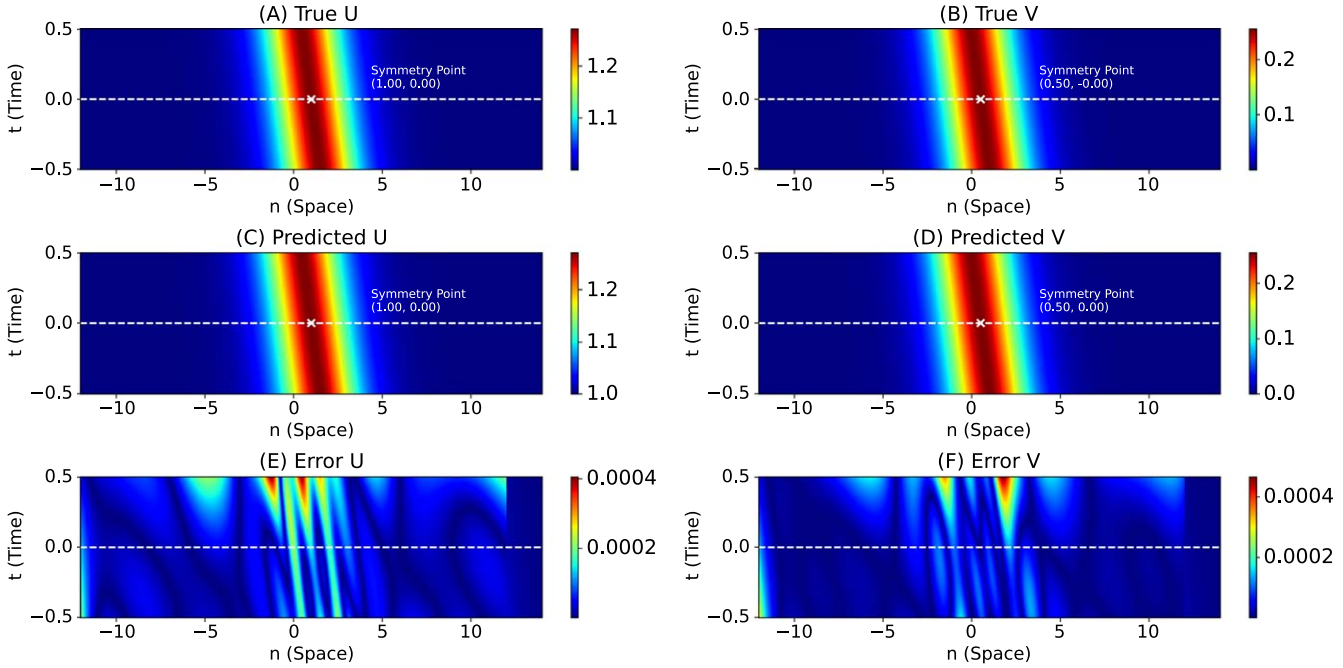

Figure 9. Numerical solution of equation (9) for one-soliton solution with SDE-PINN(2), where the spatial and temporal scales below the white dotted line are training and prediction, and above is the extrapolation.

5. Conclusions

In traditional neural network learning methods for NPDEs, the focus is predominantly on continuous NPDEs. This paper discusses the discrete soliton deep learning method for the discrete NLEs through SDE-PINN. In this approach, the shift operator substitutes the higher-order dispersion terms observed in continuous NLEs, naturally fulfilling symmetry by constructing symmetric structures. The discussions cover known and unknown symmetric points, with numerical simulations conducted for one-soliton and two-soliton solutions of equation (6). The numerical analysis demonstrates that, within the designated space-time domain and over a short interval, both internal predictions and extrapolations show outstanding accuracy. Meanwhile, compared with the existing discrete soliton deep learning method PG-PhyCRNet through equation (9), we find that the SDE-PINN has an undeniable advantage in time efficiency as well as in the extrapolation effect without a pronounced overfitting phenomenon when the prediction effect is similar. The concepts of data enhancement and symmetric structures introduced in this paper also provide a novel approach for addressing the continuous NPDEs through this newly proposed discrete learning technique.

In addition, some novel findings are observed during the training process. For instance, if the initial symmetric point are selected near the range of the initial values, adjusting parameters can result in better training accuracy than using the exact given symmetric point directly. This may be due to the increased sampling near the initial values and symmetric point evolving towards true symmetry. Determining the initial symmetric points and improving the learning of true symmetric points during the training process will serve as a future research direction. Furthermore, embedding attributes of discrete NLEs such as the discrete symmetry and conservation laws into neural networks will remain a focal point of future research.

Acknowledgments

This work is supported by the National Natural Science Foundation of China (Grant No. 12071042), the Beijing Natural Science Foundation (Grant No. 1202004) and Promoting the Development of University Classification-Student Innovation and Entrepreneurship Training Programme (Grant No. 5112410857).

ORCID iDs

Xiao-Yong Wen https://orcid.org/0000-0003-1657-9064

References

[1] Raissi M, Perdikaris P and Karniadakis G E 2019 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
[2] Bandai T and Ghezzehei T A 2021 Physics-informed neural networks with monotonicity constraints for Richardson-Richards equation: estimation of constitutive relationships and soil water flux density from volumetric water content measurements *Water Resour. Res.* **57** 027642

[3] Gao H, Zahr M J and Wang J X 2022 Physics-informed graph neural Galerkin networks: a unified framework for solving PDE-governed forward and inverse problems *Comput. Methods Appl. Mech. Eng.* **390** 114502
[4] Li J and Chen Y 2020 Solving second-order nonlinear evolution partial differential equations using deep learning *Commun. Theor. Phys.* **72** 105005
[5] Sun L N, Gao H, Pan S W and Wang J X 2020 Surrogate modeling for fluid flows based on physics-constrained deep learning without simulation data *Comput. Methods Appl. Mech. Eng.* **361** 112732
[6] Dong S and Ni N 2021 A method for representing periodic functions and enforcing exactly periodic boundary conditions with deep neural networks *J. Comput. Phys.* **435** 110242
[7] Meng X, Li Z, Zhang D and Karniadakis G E 2020 PPINNs: Parareal physics-informed neural network for time-dependent PDEs *Comput. Methods Appl. Mech. Eng.* **370** 113250
[8] Ren P, Rao C P, Liu Y, Wang J X and Sun H 2022 PhyCRNet: Physics-informed convolutional-recurrent network for solving spatiotemporal PDEs *Comput. Methods Appl. Mech. Eng.* **389** 114399
[9] Wu C X, Zhu M, Tan Q Y, Kartha Y and Lu L 2023 A comprehensive study of non-adaptive and residual-based adaptive sampling for physics-informed neural networks *Comput. Methods Appl. Mech. Eng.* **403** 115671
[10] Mao Z and Meng X 2023 Physics-informed neural networks with residual/gradient-based adaptive sampling methods for solving partial differential equations with sharp solutions *Appl. Math. Mech.* **44** 1069–84
[11] Wu G, Wang F and Qiu L 2023 Physics-informed neural network for solving Hausdorff derivative Poisson equations *Fractals* **31** 2340103
[12] Zhang B, Wu G, Gu Y, Wang X and Wang F 2022 Multi-domain physics-informed neural network for solving forward and inverse problems of steady-state heat conduction in multilayer media *Phys. Fluids* **34** 116116
[13] Zhang B, Wang F and Qiu L 2023 Multi-domain physics-informed neural networks for solving transient heat conduction problems in multilayer materials *J. Appl. Phys.* **133** 245103
[14] Li J and Chen Y 2020 A deep learning method for solving third-order nonlinear evolution equations *Commun. Theor. Phys.* **72** 115003
[15] Wu G Z, Fang Y, Wang Y Y, Wu G C and Dai C Q 2021 Predicting the dynamic process and model parameters of the vector optical solitons in birefringent fibers via the modified PINN *Chaos Solitons Fract.* **152** 111393
[16] Li J and Chen Y 2021 A physics-constrained deep residual network for solving the sine-Gordon equation *Commun. Theor. Phys.* **73** 015001
[17] Qin S M, Li M, Xu T and Dong S Q 2023 A-WPINN algorithm for the data-driven vector-soliton solutions and parameter discovery of general coupled nonlinear equations *Phys.* D **443** 133562
[18] Qin S M, Li M, Xu T and Dong S Q 2023 AM-GPINN algorithm and its application in a variable-coefficient resonant nonlinear Schrödinger equation *Phys. Scr.* **98** 025219
[19] Peng W Q and Chen Y 2024 PT-symmetric PINNs for integrable nonlocal equations: Forward and inverse problems *Chaos* **34** 043124
[20] Lin S and Chen Y 2022 A two-stage physics-informed neural network method based on conserved quantities and applications in localized wave solutions *J. Comput. Phys.* **457** 111053
[21] Lin Z and Chen Y 2024 Pseudo grid-based physics-informed convolutional-recurrent network solving the integrable nonlinear lattice equations *Phys.* D **468** 134304
[22] Liu X K and Wen X Y 2022 A discrete KdV equation hierarchy: continuous limit, diverse exact solutions and their asymptotic state analysis *Commun. Theor. Phys.* **74** 065001