nature communications

Article

https://doi.org/10.1038/s41467-025-64624-3

Automatic network structure discovery of physics informed neural networks via knowledge distillation

Received: 21 May 2025

Accepted: 23 September 2025

Published online: 29 October 2025

Check for updates

Ziti Liu[1,2,4], Yang Liu[2,4], Xunshi Yan[3], Wen Liu[2], Han Nie[2], Shuaiqi Guo[2] & Chen-an Zhang[2]

Partial differential equations (PDEs) are fundamental for modeling complex physical processes, often exhibiting structural features such as symmetries and conservation laws. While physics-informed neural networks (PINNs) can simulate and invert PDEs, they mainly rely on external loss functions for physical constraints, making it difficult to automatically discover and embed physically consistent network structures. We propose a physics structure-informed neural network discovery method based on physics-informed distillation, which decouples physical and parameter regularization via staged optimization in teacher and student networks. After distillation, clustering and parameter reconstruction are used to extract and embed physically meaningful structures. Numerical experiments on Laplace, Burgers, and Poisson equations, as well as fluid mechanics, show that the method can automatically extract relevant structures, improve accuracy and training efficiency, and enhance structural adaptability and transferability. This approach offers a new perspective for efficient modeling and automatic discovery of structured neural networks.

The evolution of complex systems is fundamentally governed by high-dimensional, nonlinear, and multiscale partial differential equations (PDEs), which arise in diverse fields such as geosciences, materials science, fluid dynamics, and biological systems[1]. Numerical methods such as finite element[2] and meshless approaches[3] have advanced physical modeling over the past decades, but traditional PDE-solving paradigms are increasingly limited in theory and application when confronted with high-dimensional parameter spaces, extreme computational costs[4], the high trial-and-error cost of inverse problems[5], and practical constraints such as missing boundary or initial conditions. These methods are also limited in their ability to exploit the underlying structural features of the system. Therefore, there is an urgent need for modeling frameworks that integrate data and physics and possess strong applicability to overcome these bottlenecks.

In recent years, machine learning methods, especially deep learning methods[6], has provided new perspectives for scientific modeling. ML can automatically learn couplings and nonlinear mappings between variables from observational data, demonstrating great potential in system identification and equation discovery[7], complex system modeling[8]. However, conventional end-to-end deep learning approaches require large amounts of data and are prone to overfitting, which limits their applicability[1]. To address these issues, physics-informed neural networks (PINNs)[9] have emerged, embedding physical constraints into neural networks (NNs) via loss functions and thus integrating data-driven learning with physical priors. PINNs offer scalability[10], flexibility and mesh-free characteristics[11], making them powerful tools for solving PDEs and showing promise in inverse problems[12]. However, most existing PINN approaches focus on enforcing physical constraints through loss functions, while largely ignoring

[1]School of Advanced Interdisciplinary Sciences, University of Chinese Academy of Sciences, Beijing, China. [2]State Key Laboratory of High Temperature Gas Dynamics, Institute of Mechanics, Chinese Academy of Sciences, Beijing, China. [3]Institute of Nuclear and New Energy Technology, Tsinghua University, Beijing, China. [4]These authors contributed equally: Ziti Liu, Yang Liu. e-mail: yanxs@tsinghua.edu.cn; zhch_a@imech.ac.cn

the explicit structural characteristics of the underlying physical systems. Furthermore, external loss functions only minimize the average inconsistency between model predictions and the physical mechanism[13]. As a result, for problems requiring strong physical consistency, one limitation of PINNs is that their predictions do not strictly adhere to underlying physical conservation laws[14].

It is practical to enhance model performance by adding more domain-specific constraints[15–17]. Some methods impose constraints externally to the network, such as Hamiltonian neural networks[18], Lagrangian neural networks[19], and symplectic networks[20], which maintain conservation laws by learning energy-like scalar quantities. Automatic symmetry discovery methods[21] encourage networks to preserve symmetries by defining generalized symmetry loss functions. In contrast, another class of effective approaches embeds constraints directly within the internal structure of the network[22], thereby ensuring strict satisfaction of these constraints. For example, Rao et al.[23] encoded constraints within the network to optimize solutions for reaction-diffusion equations. The underlying principle is that adjusting a system's internal structure can modify its output to satisfy required constraints[24]. Furthermore, if the network connectivity itself is designed to match the nature of the problem, the network can achieve superior performance in specialized domains[24]. For instance, Zhu et al.[11] enforced relationships among trainable parameters to embed the space-time parity symmetry (ST-symmetry) of the Ablowitz-Ladik (A-L) equation into the network's weight arrangement, enabling the simulation of nonlinear dynamic lattice solutions. However, these parameter-structured data-driven models rely on manual construction based on strong prior knowledge for specific problems, resulting in limited structural patterns and restricted applicability. Therefore, developing algorithms for automatic identification and extraction of network structures–reducing dependence on prior knowledge and manual design–would significantly enhance the applicability of structured network data-driven methods.

One effective approach for extracting network structures in NN techniques is regularization[25,26]. However, applying regularization in PINNs often fails to yield satisfactory results, as shown in Fig. 1. This is not only because PINNs are insensitive to regularization terms in the loss function[27], but also because parameter regularization introduces gradient optimization directions that may conflict with the existing physical constraint regularization in PINNs[9]. Such excessive constraints can actually degrade the accuracy of PINNs[1].

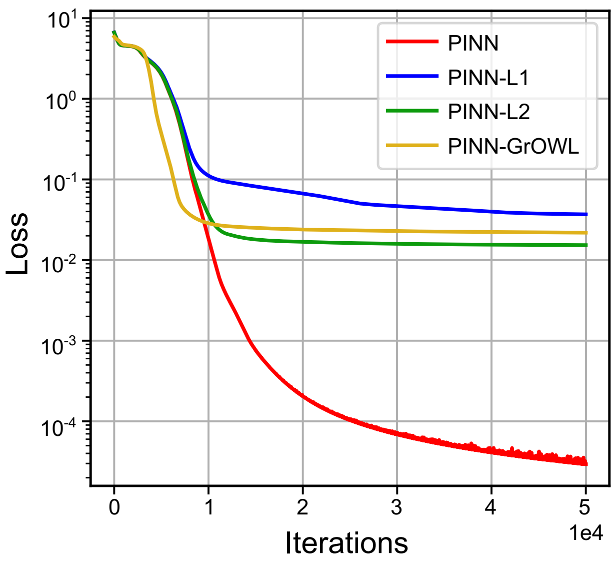


Fig. 1 | The conflict between structure extraction methods and PINN. After adding the regularization term, the loss of the PINN model increases.

Distillation learning[28], as an effective dual-model training scheme, allows two models to be trained with different loss terms, enabling the student model to achieve accuracy comparable to the teacher model. This provides a means to integrate both physical and parameter regularization.

Inspired by knowledge distillation, we propose a physics structure-informed NN discovery framework (Physics structure-informed neural network, pronounced as Psi-NN and abbreviated as *Ψ*-NN.) that enables automatic identification, extraction, and reconstruction of network architectures. In *Ψ*-NN, physical regularization (from governing equations) and parameter regularization are decoupled and applied separately to the teacher and student networks, overcoming the insensitivity to regularization and potential performance degradation observed in conventional PINNs. Physical information is efficiently transferred from the teacher to the student network via distillation, preserving essential physical constraints while expanding the representational capacity of the student model. An optimized structure extraction algorithm then automatically identifies parameter matrices with physical significance, while maximally retaining the feature space of the student network. Finally, a reinitialization mechanism is employed for network reconstruction, ensuring physical consistency in the network structure while endowing the model with applicability. By organically integrating distillation, structure extraction, and network reconstruction, *Ψ*-NN achieves physical consistency, interpretability, and high-accuracy predictions in structured NNs.

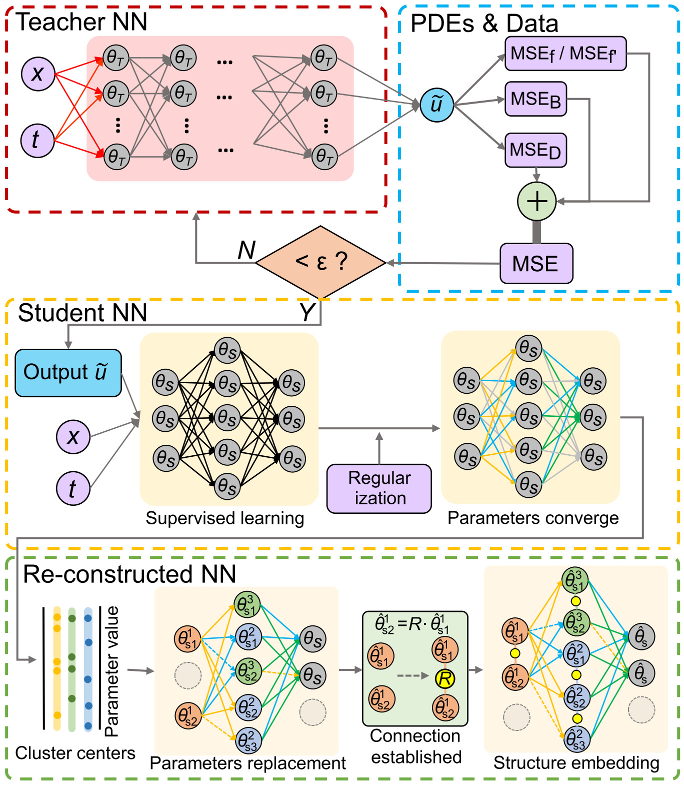


Fig. 2 | The algorithm of the proposed *Ψ*-NN model. The teacher network predicts the computational domain using the PINN approach, while the student network is supervised by the teacher's output, forming a distillation learning process. During the training of the student network, regularization methods are used to naturally drive the parameters into clusters that can be identified by a clustering algorithm under the current physical constraints. Finally, based on the clustering results, parameter matrices related to physical properties are extracted, and ultimatel,y the network structure is reconstructed through structure-embedding (embed the unchanged relation matrix *R* into a new network).

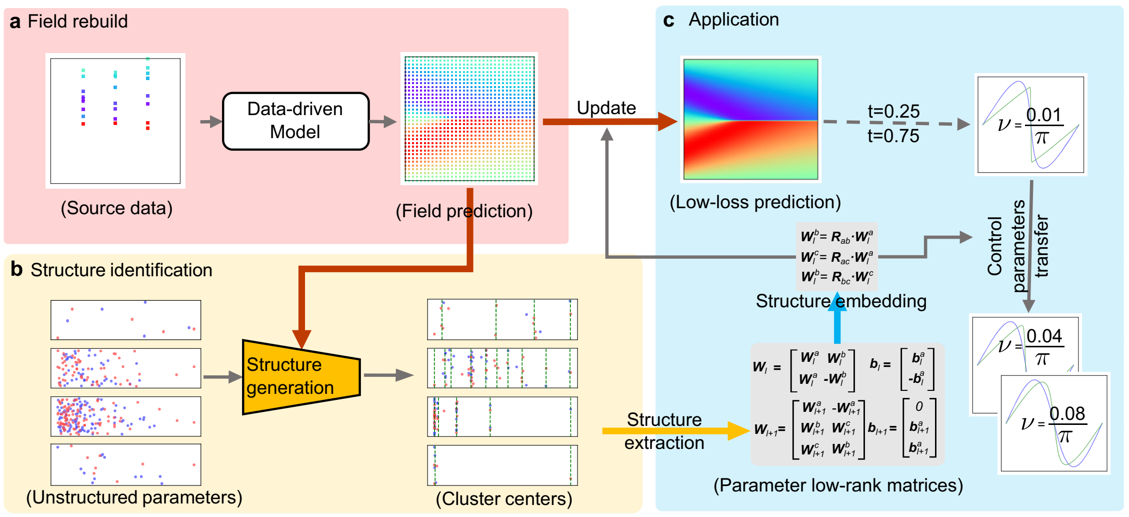


Fig. 3 | The pipeline of the *Ψ*-NN method. Using Burgers equation as a sample. Three bold arrows represent the distillation-extraction-reconstruction process. **a** The whole field data was predicted by data-driven model. **b** The student network is trained with the structure generating method, using the teacher model's output. **c** The structure extraction and reconstruction method is applied to find an optimal network structure for the Burgers equation modeling.

Results

Physics structure-informed neural network

To achieve an automatic, physics informed, and interpretable NN structure extraction mechanism, the *Ψ*-NN method consists of three components: (A) physics-informed distillation; (B) network parameter matrix extraction; and (C) structured network reconstruction, as illustrated in Fig. 2. The core idea of *Ψ*-NN is to embed physical information–such as spatiotemporal symmetries and conservation laws–directly into the network architecture. These constraints are encoded by the parameter matrices and reconstructed within the new network structure, thereby endowing the network with physical relevance. Further details are provided in Section 3. An ablation study of the *Ψ*-NN method is presented in Supplementary Information, validating the necessity of each component.

In the numerical experiments, *Ψ*-NN achieves the goal of extracting network structures from data under partially known physical laws. The case studies demonstrate that *Ψ*-NN (A) accurately solves specific physical problems; (B) generalizes across different control parameters within the same problem; and (C) maintains generalizability of the reconstructed network structure across different physical problems. The detailed case settings are provided in Supplementary Information. The results show that *Ψ*-NN can effectively extract high-performance network structures in problems with partially known laws, yielding good fitting accuracy within the problem domain. The overall workflow is illustrated in Fig. 3, and the error results are summarized in Table 1. Control parameter transfer refers to the case where the form of the PDE is fixed during the inverse problem, but certain parameters (such as the viscosity coefficient in the Burgers equation shown in the figure) are varied.

Extraction of network structure from PDEs

We selected several representative PDEs–the Laplace equation, Burgers equation, and Poisson equation–to employ baseline models with prior hard constraints, thereby better demonstrating the generalizability of *Ψ*-NN. These problems are widely used in physics. In the control group, we use PINNs with post-processing hard mapping[13,22](PINN-post) as well as standard PINNs[9]. The former introduces additional enforced constraints by post-processing the network outputs, while the latter serves as a general NN solver for PDEs. The machine used for the case studies is an Intel 12400f CPU, RTX 4080 GPU. In all cases, the Adam optimizer[29] is used for training. To ensure reproducibility, the random seed is fixed at 1234. The computational results are shown in Fig. 4.

Table 1 | Full-field L2 error comparison

Question	PINN	PINN-post	*Ψ*-NN
Laplace (1e−4)	11.59	4.017	0.7422
Burgers (1e−2)	14.47	3.014	1.287
Poisson (1e−2)	2.633	2.563	2.464
Flow *p*(1e−4)	14.89	11.78	7.838
Flow *u*(1e−4)	1.981	1.904	1.854
Flow *v*(1e−5)	1.984	1.896	1.765

PINN-post refers to a model that applies hard-mapping functions as a post-processing step to the results of PINN.

Laplace equation. Laplace's equation has applications in various fields, including electric fields, heat conduction, and fluid statics[30]. With appropriate boundary and initial conditions, this equation can exhibit clear symmetry properties, providing more distinct structural features for the network. The Laplace equation is used to fully illustrate the implementation process of *Ψ*-NN and to demonstrate the interpretability of the *Ψ*-NN structure.

Consider the steady-state Laplace equation in two-dimensional coordinates $\boldsymbol{x} \in \mathbb{R}^2$ with the following control PDE:

$$\mathcal{L}: = \nabla^2 u = 0, \quad \boldsymbol{x} \in [-1, 1]^2 \tag{1}$$

where $\boldsymbol{x} = (x_1, x_2)$. The boundary conditions of the problem is:

$$\mathcal{B}: = u = \begin{cases} -1 + 3x_2^2, & x_1 = -1 \\ 1 - 3x_2^2, & x_1 = 1 \\ x_1^3 - 3x_1, & x_2 = -1 \\ x_1^3 - 3x_1, & x_2 = 1 \end{cases} \tag{2}$$

other settings are provided in Supplementary Information.

A. Extracted structure. The *Ψ*-NN method enables clear extraction of network structures under the guidance of physical laws, whereas other existing methods can negatively impact network accuracy, as detailed

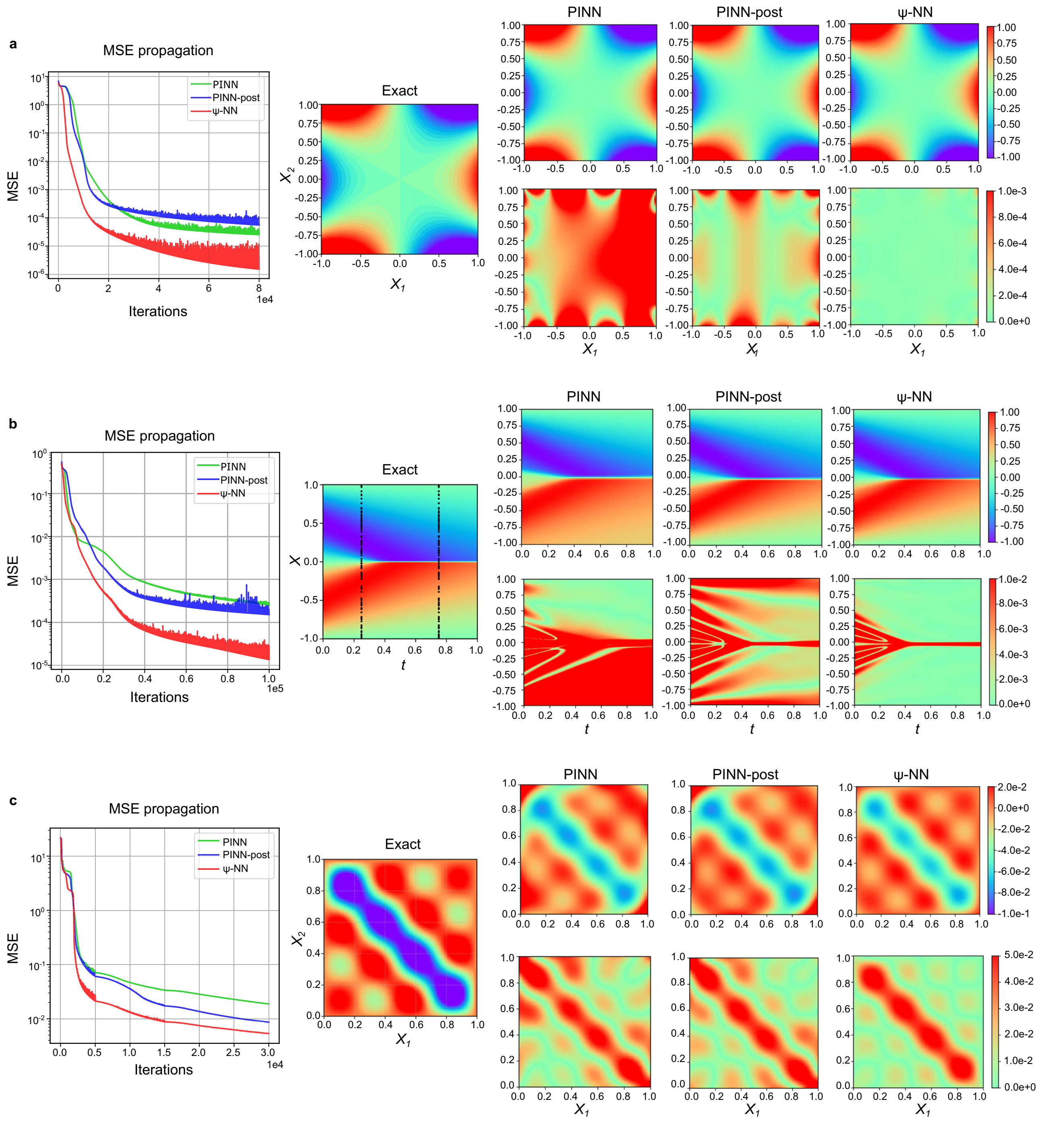
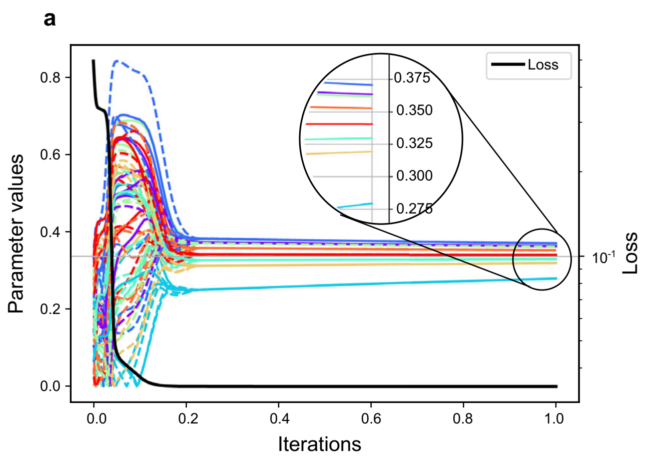


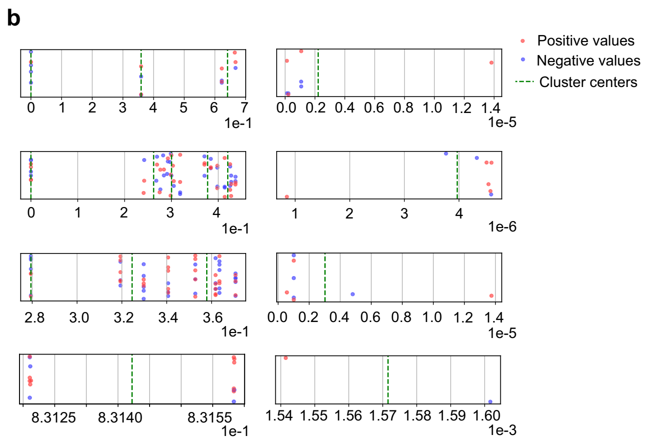
Fig. 4 | Numerical results. a Laplace equation results. **b** Burgers equation results. Black dots in the exact field represent sampling points. **c** Poisson equation results. The first column on the left shows the mean square error (MSE) propagation, which is used to compare the optimization speed (the decrease in error within the same number of steps) and optimization accuracy (the final MSE value) of the representative models; the second column on the left shows the ground truth of the cases. The three columns on the right are the results of PINN, PINN-post, and *Ψ*-NN, respectively. The first row shows the model predictions, and the second row shows the absolute error between the model and the ground truth.

in Supplementary Information. Figure 5a shows the evolution of the first hidden layer parameters during training. As the student network loss stabilizes, parameter convergence becomes more pronounced, resulting in extractable network structures. This convergence phenomenon is observed across different layers, and the final parameter clustering results under *Ψ*-NN are shown in Fig. 5b. The clustering of biases also converges and approaches zero, reducing inter-layer bias features and making the symmetry more evident.

Figure 5 c shows the network structure after replacing the original parameters with the cluster centers.

Since the second hidden layer structure involves reuse, sign reversal, and swapping, we take it as an example to describe in detail the formation of the relation matrix $\boldsymbol{R}_2$ during the "structure extraction" process and its role in the "network reconstruction" process of *Ψ*-NN. The subscript indicates the second layer. First, after replacing the trainable parameters of the student network with cluster centers, the





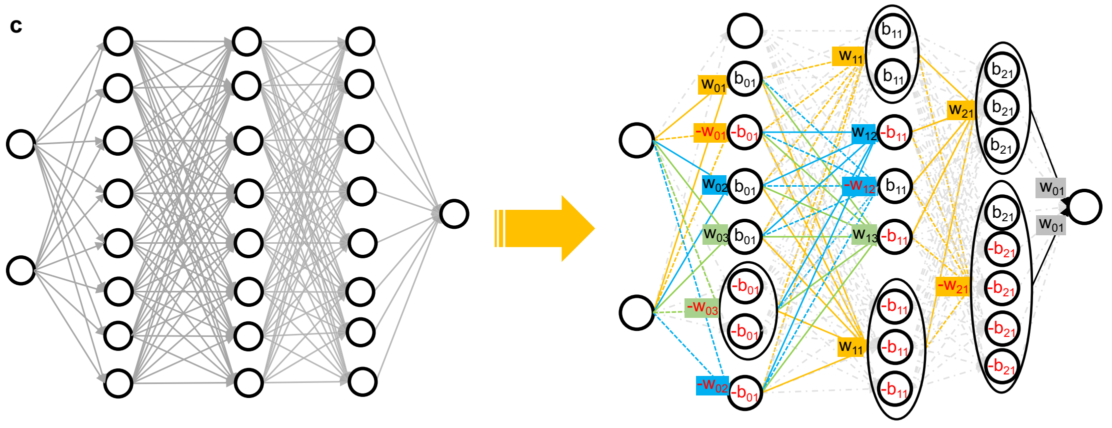


Fig. 5 | Burgers case results. a The evolution of parameters under *Ψ*-NN method. The loss curve here serves as an indicator of residual stability rather than final accuracy. **b** Cluster centers of *Ψ*-NN in Laplace equation. The *x*-axis represents the absolute value of the weights, and the *y*-values are given randomly in order to better visualize the distribution. Negative values are shown as red dots, and positive values are shown as blue dots. The cluster distance is set to 0.1. The right column contains distributions of biases, left column contains distributions of weights. The first to fourth rows correspond to the clustering results of the network parameters for the first to third hidden layers and the output layer, respectively. **c** The structure of student NN after parameter replacement in Laplace equation.

parameter matrix c_2 is:

$$c_2 = \begin{bmatrix} -c_2^a & -c_2^b \\ c_2^b & c_2^a \end{bmatrix} \quad (3)$$

parameter matrix c_2 is constructed as a diagonal matrix with different cluster center parameters arranged on the diagonal and denoted by superscripts as:

$$c_2 = \begin{bmatrix} c_2^a & 0 \\ 0 & c_2^b \end{bmatrix} \quad (4)$$

After flattening c_{S2}, selecting cluster centers using one-hot vectors, and incorporating sign relationships, the relation matrix $\boldsymbol{R}_2$ is represented as:

$$\boldsymbol{R}_2 = \begin{bmatrix} -1 & 0 \\ 0 & -1 \\ 0 & 1 \\ 1 & 0 \end{bmatrix} \quad (5)$$

In this matrix, rows with the same parameters are duplicated, rows with opposite signs are negated, and the swapping of rows 1,2 and 4,3 (which includes both row swapping and sign reversal in this case) represents the swapping relationship of parameters. The relation matrix $\boldsymbol{R}_2$ stores the relationships between network parameters, with each row representing a selected cluster center. Thus, in the reconstruction process of the new network, the trainable parameter matrix of the first hidden layer $\hat{\boldsymbol{\theta}}_2$ is constrained by $\boldsymbol{R}_2$ as follows:

$$\hat{\boldsymbol{\theta}}_2 = \boldsymbol{R}_2 \cdot \begin{bmatrix} \hat{\boldsymbol{\theta}}_2^a & 0 \\ 0 & \hat{\boldsymbol{\theta}}_2^b \end{bmatrix} = \begin{bmatrix} -\hat{\boldsymbol{\theta}}_2^a & 0 \\ 0 & -\hat{\boldsymbol{\theta}}_2^b \\ 0 & \hat{\boldsymbol{\theta}}_2^b \\ \hat{\boldsymbol{\theta}}_2^a & 0 \end{bmatrix} \quad (6)$$

After arranging the selected non-zero trainable parameters according to the node order, the following is obtained:

$$\hat{\boldsymbol{\theta}}_2 = \begin{bmatrix} -\hat{\boldsymbol{\theta}}_2^a & -\hat{\boldsymbol{\theta}}_2^b \\ \hat{\boldsymbol{\theta}}_2^b & \hat{\boldsymbol{\theta}}_2^a \end{bmatrix} \quad (7)$$

converting to matrix form, $\boldsymbol{W}_2$ is as follows:

$$\boldsymbol{W}_2 = \begin{bmatrix} -\boldsymbol{W}_2^a & -\boldsymbol{W}_2^b \\ \boldsymbol{W}_2^b & \boldsymbol{W}_2^a \end{bmatrix} \quad (8)$$

The other layers follow similarly. This structure is further represented using low-rank parameter matrices, with weight matrices being denoted as $\boldsymbol{W}_i^j$, where i indicates the layer, and j is the label for the same submatrix. The bias is denoted as $\boldsymbol{b}$. The architecture is expressed as follows::

$$\boldsymbol{W}_1 = \begin{bmatrix} \boldsymbol{W}_1^a & \boldsymbol{W}_1^b \\ -\boldsymbol{W}_1^a & -\boldsymbol{W}_1^b \end{bmatrix}, \quad \boldsymbol{b}_1 = \begin{bmatrix} \boldsymbol{b}_1^a \\ -\boldsymbol{b}_1^a \end{bmatrix} \tag{9}$$

$$\boldsymbol{W}_2 = \begin{bmatrix} -\boldsymbol{W}_2^a & -\boldsymbol{W}_2^b \\ \boldsymbol{W}_2^b & \boldsymbol{W}_2^a \end{bmatrix}, \quad \boldsymbol{b}_2 = \begin{bmatrix} \boldsymbol{b}_2^a \\ \boldsymbol{b}_2^a \end{bmatrix} \tag{10}$$

$$\boldsymbol{W}_3 = \begin{bmatrix} \boldsymbol{W}_3^a & \boldsymbol{W}_3^a \end{bmatrix}, \tag{11}$$

This low-rank matrix is reconstructed through structure-embedding (embed the unchanged relation matrix $\boldsymbol{R}$ into a new network)., where the parameters of d (parameter submatrix dimension) nodes are represented by $\boldsymbol{W}_1^a = [w_{11}^a, w_{12}^a, \cdots, w_{1d}^a]^T$.

By expanding the expressions of the final hidden layer's two sets of nodes, the relationship between the two-dimensional input $\boldsymbol{x} = (x_1, x_2)$ and the node outputs $\boldsymbol{u}_a$ and $\boldsymbol{u}_b$ can be obtained:

$$\begin{aligned} \boldsymbol{u}_a(x_1, x_2) = tanh\Big(\boldsymbol{W}_2^a \cdot \big(tanh(\boldsymbol{W}_1^a \cdot x_1 + \boldsymbol{W}_1^b \cdot x_2 + \boldsymbol{b}_1^a) \\ + \boldsymbol{W}_2^b \cdot (tanh(\boldsymbol{W}_1^a \cdot x_1 - \boldsymbol{W}_1^b \cdot x_2 + \boldsymbol{b}_1^a) + \boldsymbol{b}_2^a)\big)\Big) \end{aligned} \tag{12}$$

$$\begin{aligned} \boldsymbol{u}_b(x_1, x_2) = tanh\Big(\boldsymbol{W}_2^b \cdot \big(tanh(\boldsymbol{W}_1^a \cdot x_1 + \boldsymbol{W}_1^b \cdot x_2 + \boldsymbol{b}_1^a) \\ + \boldsymbol{W}_2^a \cdot (tanh(\boldsymbol{W}_1^a \cdot x_1 - \boldsymbol{W}_1^b \cdot x_2 + \boldsymbol{b}_1^a) + \boldsymbol{b}_2^a)\big)\Big) \end{aligned} \tag{13}$$

From the expressions of both, it follows that:

$$\boldsymbol{u}_a(x_1, x_2) = \boldsymbol{u}_b(x_1, -x_2) \tag{14}$$

finally, the output expression is:

$$\boldsymbol{u}(x_1, x_2) = \boldsymbol{W}_3^a(\boldsymbol{u}_a(x_1, x_2) + \boldsymbol{u}_b(x_1, x_2)) \tag{15}$$

where $\boldsymbol{W}_3^a$ is the parameter matrix. This expression can also be written as:

$$\boldsymbol{u}(x_1, x_2) = \boldsymbol{W}_3^a(\boldsymbol{u}_a(x_1, x_2) + \boldsymbol{u}_a(x_1, -x_2)) \tag{16}$$

Therefore, this structured method implicitly embeds the symmetry contained in PINN-post

Previous structured PINN approaches[11,31] require symmetry priors and rely on manually imposing group equivariance, targeting specific problems, while the network structure in Ψ-NN is automatically extracted from data and physical constraints, reducing reliance on manual design and strong prior knowledge.

B. Comparison between Ψ-NN and PINN. The results are shown in Fig. 4a, and the full-field L2 errors are summarized in Table 1. Compared to PINN, Ψ-NN reduces the number of iterations required to reach the same loss magnitude (1e−3) by approximately 50%, and decreases the final L2 error by about 95%. As illustrated in Fig. 6a and d, PINN does not exhibit consistent symmetry during training, whereas the structural constraints in Ψ-NN, which are consistent with the features of the PDE, enable a reduced search space and allow the solution to be found more quickly and accurately in the early stages of training.

C. Comparison between Ψ-NN and PINN-post. The PINN-post incorporates spatial symmetry into the network output layer through explicit constraints, resulting in outputs that better satisfy symmetric physical properties–reducing the full-field L2 error by approximately 65% compared to conventional PINN, especially within the computational domain. However, the converged MSE of PINN-post is higher than that of Ψ-NN, indicating that the minimum loss value in the PINN framework does not fully reflect the true accuracy of the network, but only the average fit to the available data and PDEs. The rate of loss reduction reflects the convergence speed: to reach a loss of 1e-3, PINN-post requires 5e3 fewer iterations than PINN.

Both Ψ-NN and PINN-post embed spatial symmetry into the network, but the key difference is that the reconstructed Ψ-NN architecture inherently contains this physical property, whereas PINN-post applies hard mapping as a post-processing step at the output layer. In terms of computation time: the computation time for PINN-post is 32.68 minutes, longer than Ψ-NN's 29.87 minutes. Furthermore, Ψ-NN reaches a loss of 1e−4 with 1.5e4 fewer iterations, and its average convergence speed is about twice that of PINN-post. The L2 error is reduced from 1.159e−3 to 7.422e−5. As shown in Fig. 4a, the hard mapping constraint in PINN-post does not reduce the large computational errors near the boundaries. This suggests that, due to the rich implicit constraints in physical fields, manually embedding features via post-processing provides only a necessary but not sufficient constraint, and its applicability may be limited to a certain range. In contrast, Ψ-NN discovers network structures entirely based on observational data and PDEs, aiming to automatically embed all known information about the physical problem into the network structure, thereby reducing errors over a broader computational domain.

Burgers equation. The Burgers equation, as an important tool for describing nonlinear wave phenomena, is frequently used to study complex systems such as fluid dynamics and wave behavior[32]. We select the Burgers equation due to its pronounced shock formation and resulting antisymmetric properties, which serve to validate (a) the performance advantages of structured networks and (b) their applicability capability across a wide range of parameter variations.

In the inverse problem, the viscosity coefficient in the Burgers equation is replaced by an unknown parameter λ_1, and the governing PDE becomes:

$$\mathcal{L} := u_t - uu_x - \lambda_1 u_{xx} = 0, \quad x \in [-1, 1], \quad t \in [0, 1] \tag{17}$$

A. Extraction result and performance comparison. In the Ψ-NN extraction process, taking the third hidden layer as an example, the parameter variations are shown in Supplementary Information (Fig. 6). The extracted low-rank parameter matrices are:

$$\boldsymbol{W}_1 = \begin{bmatrix} \boldsymbol{W}_1^a & \boldsymbol{W}_1^b \\ \boldsymbol{W}_1^a & -\boldsymbol{W}_1^b \end{bmatrix}, \quad \boldsymbol{b}_1 = \begin{bmatrix} \boldsymbol{b}_1^a \\ -\boldsymbol{b}_1^a \end{bmatrix} \tag{18}$$

$$\boldsymbol{W}_2 = \begin{bmatrix} \boldsymbol{W}_2^a & -\boldsymbol{W}_2^a \\ \boldsymbol{W}_2^b & \boldsymbol{W}_2^c \\ \boldsymbol{W}_2^c & \boldsymbol{W}_2^b \end{bmatrix}, \quad \boldsymbol{b}_2 = \begin{bmatrix} \boldsymbol{0} \\ \boldsymbol{b}_2^a \\ \boldsymbol{b}_2^a \end{bmatrix} \tag{19}$$

$$\boldsymbol{W}_3 = \begin{bmatrix} \boldsymbol{W}_3^a & \boldsymbol{W}_3^b & -\boldsymbol{W}_3^b \end{bmatrix}, \quad \boldsymbol{b}_3 = \boldsymbol{0} \tag{20}$$

Similarly, the relationship between the two-dimensional input $\boldsymbol{x} = (x_1, x_2)$ and the node outputs $\boldsymbol{u}_a$, $\boldsymbol{u}_b$, $\boldsymbol{u}_c$ of the final hidden layer can be obtained:

$$\begin{aligned} \boldsymbol{u}_a(x_1, x_2) = tanh\Big(\boldsymbol{W}_2^a \cdot \big(tanh(\boldsymbol{W}_1^a \cdot x_1 + \boldsymbol{W}_1^b \cdot x_2 + \boldsymbol{b}_1^a) \\ - \boldsymbol{W}_2^a \cdot (tanh(\boldsymbol{W}_1^a \cdot x_1 - \boldsymbol{W}_1^b \cdot x_2 - \boldsymbol{b}_1^a))\big)\Big) \end{aligned} \tag{21}$$

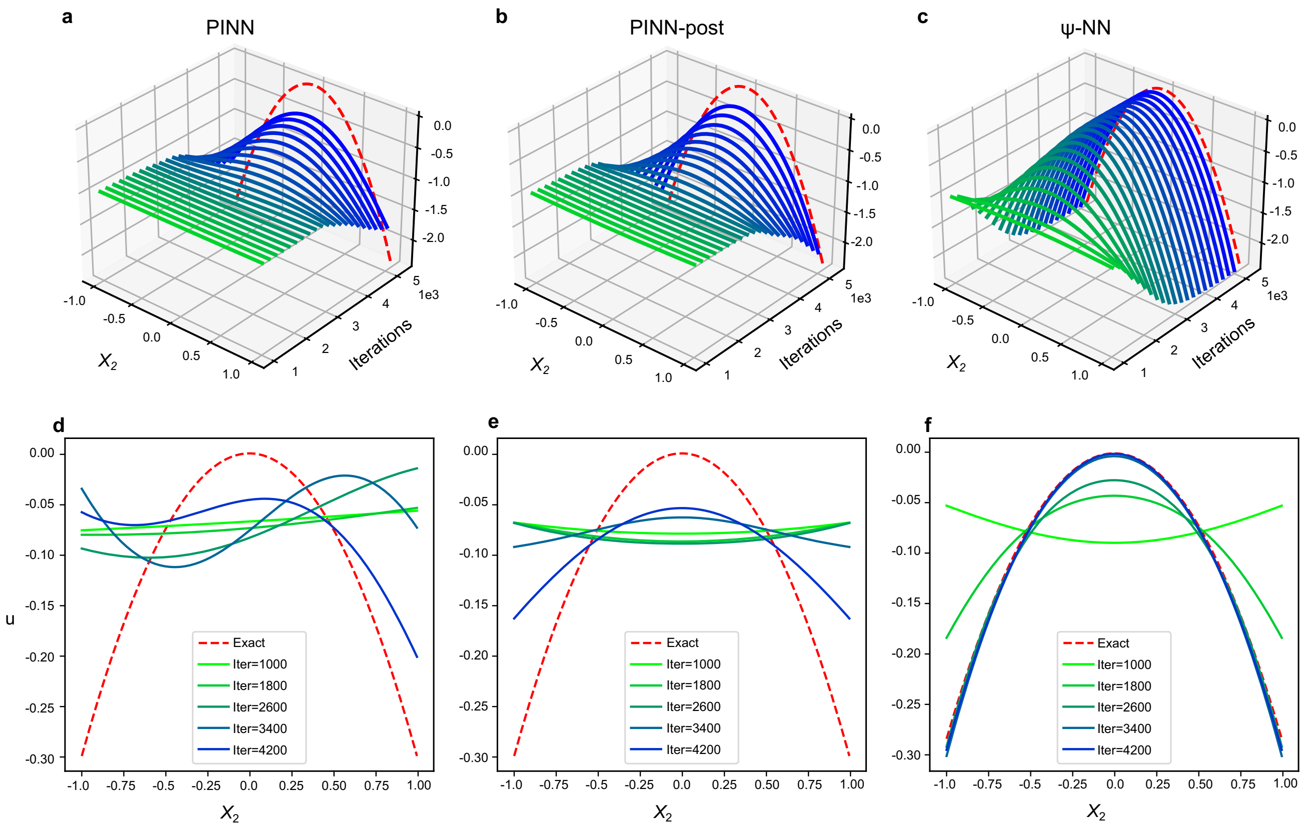


Fig. 6 | Prediction comparison by iterations. a, **d**, PINN predictions; **b**, **e**, PINN-post predictions; **c** and **f**, *Ψ*-NN predictions. The optimization tendency of different models. In the figures of the first row, the red dashed line represents the true value of u as x_2 varies when $x_1 = 0.8$. Each green-blue gradient curve represents the network output at training steps from 1000 to 5000, with an interval of 300. In the second row of figures, the results at several training steps are plotted on a two-dimensional coordinate plane to illustrate the symmetry breaking in PINN during the iterative process, as well as the symmetry preservation in PINN-post and *Ψ*-NN.

Table 2 | Burgers equation control parameter comparison

Model	Truth	PINN	PINN-post	*Ψ*-NN
$v_1 \cdot \pi$(1e−2)	1	2.820	1.898	1.465
$v_2 \cdot \pi$(1e−2)	4	6.625	4.960	4.084
$v_3 \cdot \pi$(1e−2)	8	9.705	0.885	6.673

$$\boldsymbol{u}_b(x_1,x_2) = tanh\Big(\boldsymbol{W}_2^b \cdot \big(tanh(\boldsymbol{W}_1^a \cdot x_1 + \boldsymbol{W}_1^b \cdot x_2 + \boldsymbol{b}_1^a) + \boldsymbol{W}_2^c \cdot (tanh(\boldsymbol{W}_1^a \cdot x_1 - \boldsymbol{W}_1^b \cdot x_2 - \boldsymbol{b}_1^a) + \boldsymbol{b}_2^a)\big)\Big) \tag{22}$$

$$\boldsymbol{u}_c(x_1,x_2) = tanh\Big(\boldsymbol{W}_2^c \cdot \big(tanh(\boldsymbol{W}_1^a \cdot x_1 + \boldsymbol{W}_1^b \cdot x_2 - \boldsymbol{b}_1^a) + \boldsymbol{W}_2^b \cdot (tanh(\boldsymbol{W}_1^a \cdot x_1 - \boldsymbol{W}_1^b \cdot x_2 + \boldsymbol{b}_1^a) + \boldsymbol{b}_2^a)\big)\Big) \tag{23}$$

From the expressions of the three, it follows that:

$$\lim_{b_1^a \to 0} \boldsymbol{u}_a(x_1, -x_2) = -\boldsymbol{u}_a(x_1,x_2) \tag{24}$$

$$\boldsymbol{u}_b(x_1,x_2) = \boldsymbol{u}_c(x_1, -x_2) \tag{25}$$

The expression of the final output $\boldsymbol{u}(x_1, x_2)$ is:

$$\boldsymbol{u}(x_1,x_2) = \boldsymbol{W}_3^a \cdot \boldsymbol{u}_a(x_1,x_2) + \boldsymbol{W}_3^b \cdot (\boldsymbol{u}_b(x_1,x_2) - \boldsymbol{u}_c(x_1,x_2)) \tag{26}$$

where $\boldsymbol{W}_3^a, \boldsymbol{W}_3^b$ are parameter matrices.

Therefore, the first half of $\boldsymbol{u}$, $\boldsymbol{W}_3^a \cdot \boldsymbol{u}_a(x_1,x_2)$, contains the symmetry of PINN-post under the condition $\lim_{b_1^a \to 0}$, while the second half, $\boldsymbol{W}_3^b \cdot (\boldsymbol{u}_b(x_1,x_2) + \boldsymbol{u}_c(x_1,x_2))$, directly contains the symmetry of PINN-post. Additionally, the properties of $\boldsymbol{u}_a$ in the second hidden layer also indicate that the emergence of symmetry does not strictly depend on the number of network layers.

The trend of the loss function is shown in Fig. 4b. The reconstructed structured network of *Ψ*-NN exhibits significantly faster iteration speed during training compared to PINN and PINN-post, achieving a minimum loss function accuracy of 1*e*-5, which is lower than the other two models.

The full-field L2 errors are summarized in Table 1. Both PINN-post and *Ψ*-NN achieve lower errors than PINN, demonstrating the effectiveness of embedding equation features into the network structure for improving fitting accuracy. As shown in Fig. 4b, after shock formation at $t > 0.4$, both PINN and PINN-post exhibit large error distributions, whereas *Ψ*-NN uniformly reduces errors on both sides of the shock. Furthermore, PINN-post, due to its post-processing symmetry, enforces a symmetric error distribution across the shock but does not reduce the actual error. The structure discovery capability of *Ψ*-NN provides a more precise and matching feature space, resulting in the lowest error.

For the inverse problem, in addition to reconstructing the entire field, another key task is to estimate the unknown parameter λ_1 in Eq. (17), with the true value $\lambda_1 = 0.01/\pi$. The final results are given in Table 2, where *Ψ*-NN achieves the closest value to the ground truth. The evolution of this parameter during training is shown in Fig. 7. Since λ_1 is included in the trainable parameter vector of the NN, these curves can be interpreted as optimization trajectories[6]; a shorter path indicates a clearer search direction and faster convergence. Thus, the

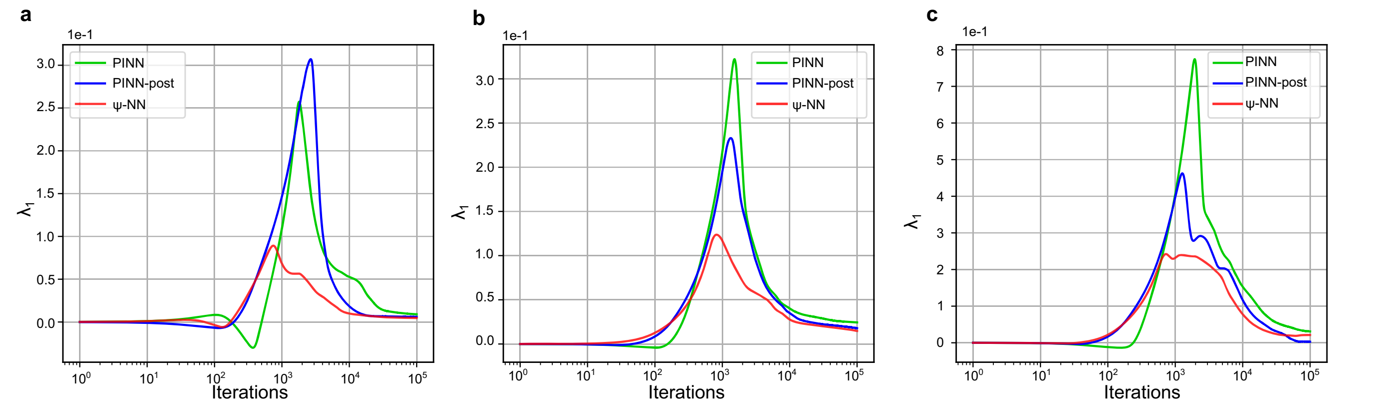


Fig. 7 | Prediction comparison under different control parameters. a $\nu_1 = \frac{0.01}{\pi}$. **b** $\nu_2 = \frac{0.04}{\pi}$. **c** $\nu_3 = \frac{0.08}{\pi}$. Burgers problem parameters comparison. The shorter the path, the clearer the search direction in the parameter space and the faster the convergence. The structural features discovered by *Ψ*-NN reduce the output space and make it easier to find the correct solution. Moreover, the network structure found in $\nu_1 = \frac{0.01}{\pi}$ can successfully generalize to $\nu_2 = \frac{0.04}{\pi}$ and $\nu_3 = \frac{0.08}{\pi}$, yielding good results.

structural features discovered by *Ψ*-NN reduce the output space and make it easier to find the correct solution.

B. *Ψ*-NN structure performance in different parameter cases. The viscosity term ν to be solved in the inverse problem is represented by the unknown parameter λ_1. This allows us to validate the applicability capability of the structure reconstructed for a specific problem under different parameters. We conducted experiments using the same *Ψ*-NN-reconstructed structure without modifying other configurations, specifically at $\nu_2 = 0.04/\pi$ and $\nu_3 = 0.08/\pi$. The computational results are shown in Fig. 7, where the *Ψ*-NN method maintains shorter paths across different parameters. Shorter path across different parameters indicates that *Ψ*-NN has a more efficient optimization process in parameter space[6]. The final prediction results are summarized in Table 2, with values closest to the ground truth.

Poisson equation. Poisson's equation plays a crucial role in various computations, including heat conduction, electromagnetism, and gravitational fields[33]. Here, we select a Poisson problem in a unit square domain with a smooth source term $f(x_1, x_2)$ that contains four increasing frequencies. This choice allows us to demonstrate the applicability and performance of the *Ψ*-NN method across different parameters. High-frequency physical systems often exhibit inherent symmetric structures[34]. To address this, we employ *Ψ*-NN to discover and leverage the symmetric patterns present in the problem, effectively alleviating the challenges associated with high-frequency characteristics and enhancing modeling efficiency and accuracy. Specifically, Poisson's equation satisfies the constraint:

$$\mathcal{L} : = \nabla^2 u = f(x_1, x_2), \quad x_1, x_2 \in [0,1] \tag{27}$$

where the source term is given by:

$$f(x_1, x_2) = \frac{1}{4}\sum_{k=1}^{4}(-1)(k+1)2k^2 sin(k\pi x_1)sin(k\pi x_2) \tag{28}$$

The source term exhibits permutation equivariance, which can be reformulated with the equivariant group $H = \mathbb{Z}_2 \times \mathbb{Z}_2$ formed by the 2-order cyclic group $\mathbb{Z}_2 : \{0, 1\}$. The 2 transformations can be stated as $\forall\ (h_1, h_2) \in H$:

$$\begin{cases} \mathcal{T}_{X_f}^{h_1,h_2}(x_1, x_2) = ((-1)^{h_1}x_2, (-1)^{h_2}x_1) \\ \mathcal{T}_{Y_f}^{h_1,h_2}(f) = (-1)^{h_1+h_2} f \end{cases} \tag{29}$$

where $\mathcal{T}_X^h, \mathcal{T}_Y^h$ are the group actions on domain X and codomain Y, respectively.

Similarly, we construct a low-frequency solution to the PDEs (27) with the source term $f = 0$, ensuring the permutation property of the solution. The permutation property can be simplified using the permutation equivariant group $H_e = \mathbb{Z}_2$ as $\forall\ h \in H_e$:

$$\begin{cases} \mathcal{T}_{X_{lf}}^{h}(x_1, x_2) = (-x_2, -x_1) \\ \mathcal{T}_{Y_{lf}}^{h}(u) = (-1)^h u \end{cases} \tag{30}$$

The constructed solution is $u = x_1^2 - x_2^2$. The low-rank parameter matrix extracted for this low-frequency function is:

$$\boldsymbol{W}_1 = \begin{bmatrix} \boldsymbol{W}_1^a & \boldsymbol{0} \\ \boldsymbol{0}_1 & \boldsymbol{W}_1^a \end{bmatrix}, \quad \boldsymbol{b}_1 = \begin{bmatrix} \boldsymbol{b}_1^a \\ -\boldsymbol{b}_1^a \end{bmatrix} \tag{31}$$

$$\boldsymbol{W}_2 = \begin{bmatrix} \boldsymbol{W}_2^a & \boldsymbol{W}_2^b \\ \boldsymbol{W}_2^b & \boldsymbol{W}_2^a \end{bmatrix}, \quad \boldsymbol{b}_2 = \begin{bmatrix} \boldsymbol{b}_2^a \\ -\boldsymbol{b}_2^a \end{bmatrix} \tag{32}$$

$$\boldsymbol{W}_3 = \begin{bmatrix} \boldsymbol{W}_3^a & \boldsymbol{W}_3^a \end{bmatrix}, \quad \boldsymbol{b}_3 = \boldsymbol{0} \tag{33}$$

Similarly, the relationship between the two-dimensional input $\boldsymbol{x} = (x_1, x_2)$ and the node outputs $\boldsymbol{u}_a$, $\boldsymbol{u}_b$ of the final hidden layer can be obtained:

$$\begin{aligned} \boldsymbol{u}_a(x_1, x_2) = tanh\Big(&\boldsymbol{W}_2^a \cdot tanh(\boldsymbol{W}_1^a \cdot x_1 + \boldsymbol{b}_1^a) \\ &+ \boldsymbol{W}_2^b \cdot tanh(\boldsymbol{W}_1^a \cdot x_2 - \boldsymbol{b}_1^a) + \boldsymbol{b}_2^a\Big) \end{aligned} \tag{34}$$

$$\begin{aligned} \boldsymbol{u}_b(x_1, x_2) = tanh\Big(&\boldsymbol{W}_2^b \cdot tanh(\boldsymbol{W}_1^a \cdot x_1 + \boldsymbol{b}_1^a) \\ &+ \boldsymbol{W}_2^a \cdot tanh(\boldsymbol{W}_1^a \cdot x_2 - \boldsymbol{b}_1^a) - \boldsymbol{b}_2^a\Big) \end{aligned} \tag{35}$$

From the expressions of both, it follows that:

$$\boldsymbol{u}_a(x_1, x_2) = -\boldsymbol{u}_b(-x_2, -x_1) \tag{36}$$

Since the expression of the final output $\boldsymbol{u}(x_1, x_2)$ is:

$$\boldsymbol{u}(x_1, x_2) = \boldsymbol{W}_3^a \cdot (\boldsymbol{u}_a(x_1, x_2) + \boldsymbol{u}_b(x_1, x_2)) \tag{37}$$

this result contains the symmetry defined in (30).

In order to match the network structure results with the characteristics of the high-frequency solution, by setting the sign of the values in $\boldsymbol{b}_1$ to be the same, we can obtain:

$$\begin{aligned}\tilde{\boldsymbol{u}}_a(x_1, x_2) = tanh\big(&\boldsymbol{W}_2^a \cdot tanh(\boldsymbol{W}_1^a \cdot x_1 + \boldsymbol{b}_1^a) \\ &+ \boldsymbol{W}_2^b \cdot tanh(\boldsymbol{W}_1^a \cdot x_2 + \boldsymbol{b}_1^a) + \boldsymbol{b}_2^a\big)\end{aligned} \tag{38}$$

$$\begin{aligned}\tilde{\boldsymbol{u}}_b(x_1, x_2) = tanh\Big(&\boldsymbol{W}_2^b \cdot tanh(\boldsymbol{W}_1^a \cdot x_1 + \boldsymbol{b}_1^a) \\ &+ \boldsymbol{W}_2^a \cdot tanh(\boldsymbol{W}_1^a \cdot x_2 + \boldsymbol{b}_1^a) + \boldsymbol{b}_2^a\Big)\end{aligned} \tag{39}$$

from which we have:

$$\tilde{\boldsymbol{u}}_a(x_1, x_2) = \tilde{\boldsymbol{u}}_b(x_2, x_1) \tag{40}$$

thus, the final expression:

$$\boldsymbol{u}(x_1, x_2) = \boldsymbol{W}_3^a \cdot (\tilde{\boldsymbol{u}}_a(x_1, x_2) + \tilde{\boldsymbol{u}}_b(x_1, x_2)) \tag{41}$$

which satisfies the symmetry of the high-frequency solution.

Moreover, the structure can be adjusted by sign to satisfy other forms of symmetry. For example, after (36), adjusting the sign of the weights in the last hidden layer can yield:

$$\boldsymbol{u}_r = \boldsymbol{W}_3^a \cdot (\boldsymbol{u}_a - \boldsymbol{u}_b) = \boldsymbol{u}_a(x_1, x_2) + \boldsymbol{u}_a(-x_1, -x_2) \tag{42}$$

that is, the Poisson equation result can be adjusted by sign to satisfy the central symmetry $\boldsymbol{u}_r(x_1, x_2) = \boldsymbol{u}_r(-x_1, -x_2)$.

The numerical results are shown in Fig. 4c. The iteration step size used in this problem is 1e−3, which is reduced to 0.2 times the original value at steps 5e3 and 1.5e4. The *Ψ*-NN re-constructed NN outperforms both PINN and PINN-post models in terms of speed and accuracy. Their L2 errors are summarized in Table 1. All three models exhibit peak errors along the line $x_2 = -x_1 + 1$, but *Ψ*-NN maintains a lower overall error, particularly at the boundaries. In contrast, the other models in the control group show large gradient errors at local boundaries, highlighting *Ψ*-NN's superior performance in high-frequency fitting.

Steady flow passing a circular cylinder

Re-constructed structured NNs by *Ψ*-NN not only perform well in their specific problems but also exhibit good applicability across different problems with similar characteristics. Here, we utilize the structures reconstructed from the Laplace problem and the Burgers equation to validate this applicability.

In the field of fluid mechanics, the two-dimensional incompressible laminar cylinder flow case[35] can be used as a complex case with multiple outputs and multiple constraints to test the performance of *Ψ*-NN under multiple output conditions. The outputs selected in this case contain two completely opposite symmetries at the same time, which better reflects the transfer ability of *Ψ*-NN structures. The control equations are:

$$\mathcal{L}_1 := u_x + v_y = 0, \tag{43}$$

$$\mathcal{L}_2 := \rho(uu_x + vu_y) + p_x - \mu(u_x x + u_y y) = 0 \tag{44}$$

$$\mathcal{L}_3 := \rho(uv_x + vv_y) + p_y - \mu(v_x x + v_y y) = 0 \tag{45}$$

the inlet flow rate is:

$$u(0, y) = 4U_{max}(H - y)y/H^2 \tag{46}$$

that is:

$$\mathcal{B} := u(0, y) - 4U_{max}(H - y)y/H^2 = 0 \tag{47}$$

The specific settings are shown in Fig. 8a. The results are manifested in Fig. 8, with the loss iteration curve shown in Fig. 8b. The *Ψ*-NN method demonstrates superior performance in both convergence speed and accuracy, especially around the cylinder. The final L2 errors are summarized in Table 1.

Discussion

This paper presents a novel physics structure-informed network extraction method, termed *Ψ*-NN. First, *Ψ*-NN employs a three-step distillation-structure extraction-reconstruction framework to automatically discover and extract network structures consistent with physical constraints from limited sampled data, thereby linking physical information in PDEs with network architecture. This approach overcomes the reliance on prior knowledge and manual design in traditional structured PINN construction, enabling automated structural embedding. Second, *Ψ*-NN integrates knowledge distillation and parameter regularization-based sparsification through a staged training strategy, introducing a new method for automatic structure extraction and reconstruction, and expanding the application of distillation and regularization in physics-based modeling. Unlike neural architecture search methods focused on hyperparameter optimization[36], *Ψ*-NN emphasizes the automatic identification and representation of physical features among trainable parameters, surpassing the limitations of conventional structured sparsity and achieving structured embedding of physical information. The extracted network structures not only demonstrate strong physical relevance and applicability in numerical experiments, but are also supported by parameter convergence theorems and mathematical proofs, ensuring theoretical rigor and interpretability. Numerical results show that *Ψ*-NN achieves significantly improved fitting performance and reduced model complexity and computational cost compared to conventional PINNs and post-processed models (PINN-post).

Moreover, *Ψ*-NN offers a new perspective for network transfer learning. In the case of the Poisson equation, for example, by simplifying the original problem into low- and high-frequency components and further increasing complexity based on the extracted structure, *Ψ*-NN can efficiently regress from low-frequency to high-frequency solutions. During low-frequency simplification, the *Ψ*-NN structure remains interpretable, effectively reducing problem complexity and prioritizing computational efficiency. This structural transfer process provides an effective and flexible approach for extracting simple feature structures and performing complex regression across different problems, achieving low resource consumption and high computational accuracy.

Ψ-NN encodes the symbolic relationships of PDEs through network connectivity. While *Ψ*-NN effectively discovers interpretable structures from partially known problems, it still has some limitations that require further investigation:

1. The *Ψ*-NN method has been validated on physical problems with known forms of PDEs, and the extracted physical features demonstrate a certain degree of generalizability (for example, in

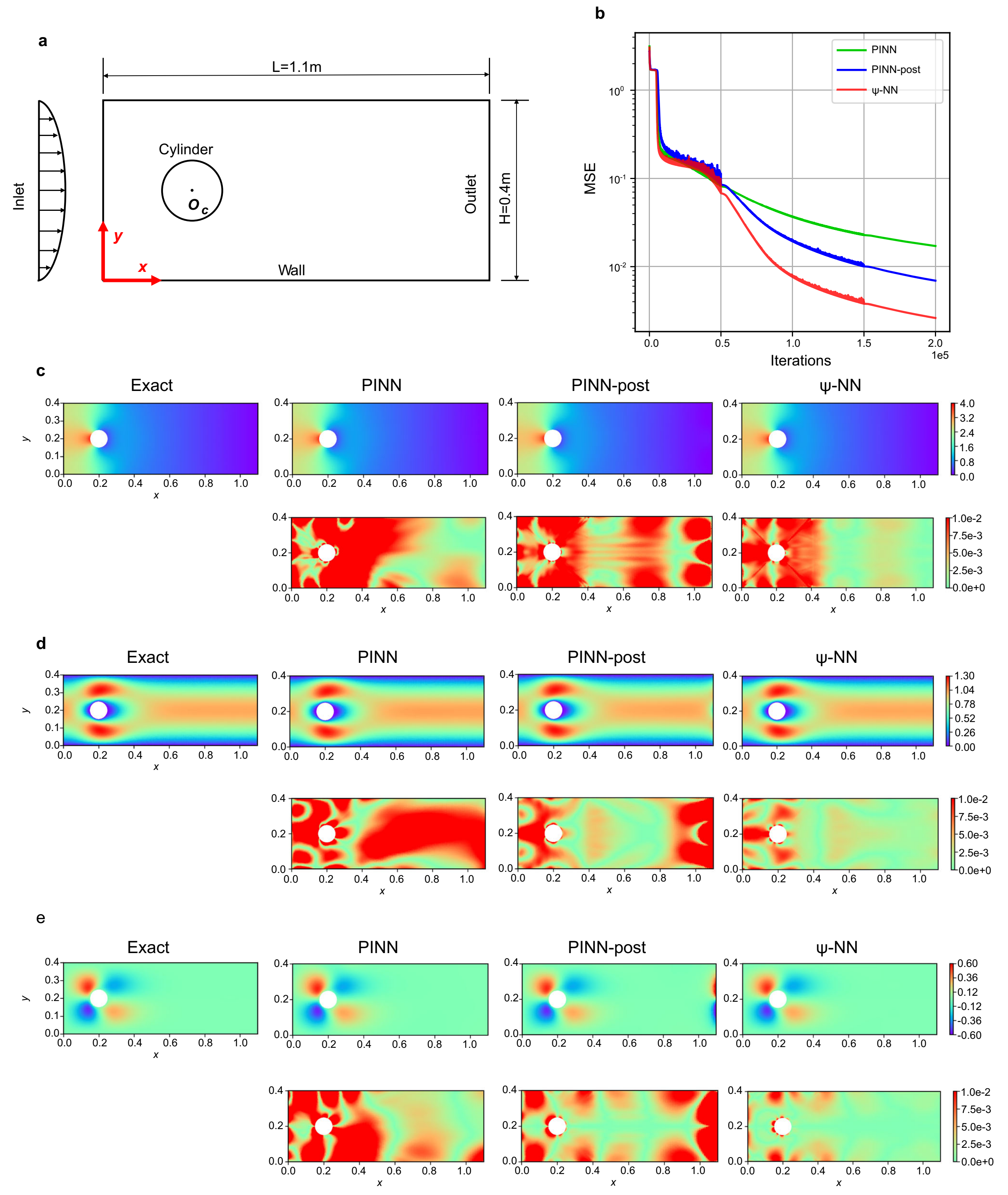


Fig. 8 | Flow case results. a Cylinder flow setting[35]. The cylinder center O_c is located at $(0.2, 0.2)(m)$ and the radius is 0.05 m. **b** Flow field loss comparison. **c** Flow field pressure results and error. **d**, u (x-axis velocity) results and error. **e**, v (y-axis velocity) results and error.

the Poisson equation case, adjusting the sign of hidden layer parameters can achieve central symmetry or anti-symmetry). However, scenarios involving real observational data or genuinely uncertain or incomplete physical constraints may entail more complex parameter-feature relationships, such as time translation symmetry or rotational invariance at arbitrary angles. These properties are often associated with conservation laws via Noether's theorem[37], and may require more sophisticated network architectures or additional exploration in parameter space. We will investigate these potential applications in future research.

2. Since the calculations in Theorems 1 and 2 are based on the properties of MLPs, the current Distillation-Extraction-Reconstruction framework of the Ψ-NN method has been validated using a three-layer fully connected multilayer perceptron (MLP) architecture. The construction of structures involving sign transformations (such as symmetry transformations) relies on the odd symmetry of the tanh activation function, while for permutation transformations (e.g., in the Poisson equation case), this property is not required. However, for more expressive architectures in specialized domains, such as Transformers[38] for sequential data, the multi-module and attention mechanisms make it difficult to establish a direct one-to-one correspondence between network parameters and physical features. Nevertheless, these models have the potential to be integrated with the distillation-extraction-reconstruction framework, which will require the development of new structure extraction and reconstruction methods tailored to the characteristics of each architecture. We will explore the integration of different network architectures with the Ψ-NN framework in future work to broaden its applicability.

Methods

Ψ-NN method consists of three main components: distillation, structure extraction, and network reconstruction. The distillation process enables the transfer of physical information without additional intervention, decoupling the optimization of physical and parameter directions by separating the high-order derivative losses from the PDEs. The structure extraction method then automatically identifies low-rank parameter matrices with physical consistency while preserving physical information. Finally, the low-rank parameter matrices are reconstructed to form network structures with physical relevance.

Physics-informed distillation

Regularization, as an effective approach for parameter sparsification, cannot be efficiently applied in PINNs[27,39]. Moreover, parameter regularization introduces gradient optimization directions that may conflict with the existing physical constraint regularization in PINNs[9], and these excessive constraints can actually degrade the accuracy of PINNs[1]. This makes it challenging to discover network structures related to physical constraints. To address this issue, it is necessary to appropriately decouple the processes of physical constraint enforcement and parameter sparsification. In classification tasks, distillation learning[28] has proven to be a successful multi-model training strategy, enabling the student network to be trained without compromising the fitting accuracy of the teacher network. Therefore, we introduce a specialized distillation mechanism that allows learning bias and regularization to coexist.

Li et al.[40] improved the distillation method for regression problems originally proposed by Muhamad et al.[41], and found that self-distillation can extract and utilize the rich physical information contained in datasets generated by PINN. Inspired by this, we extend the distillation approach to a physics-informed distillation framework, enabling the separation of learning bias while transferring physical information across networks with different architectures.

Consider a PDE with temporal coordinate t and spatial coordinates $\boldsymbol{x} \in \mathbb{R}^n$, whose solution is denoted as $\boldsymbol{u}(t, \boldsymbol{x}) \in \mathbb{R}^k$:

$$\mathcal{L}(\boldsymbol{u}, t, \boldsymbol{x}) = 0, \quad \boldsymbol{x} \in \Omega, \quad t \in [0, T] \tag{48}$$

Teacher and student networks are defined as:

$$\mathcal{N}_T(\boldsymbol{x}, t; \boldsymbol{\theta}_T) = \tilde{\boldsymbol{u}}_T \tag{49}$$

$$\mathcal{N}_S(\boldsymbol{x}, t; \boldsymbol{\theta}_S) = \tilde{\boldsymbol{u}}_S \tag{50}$$

where $\boldsymbol{\theta}$ is the vector of trainable parameters in the network, and the output $\boldsymbol{u}$ is the predicted solution to the PDE (48), with subscripts denoting teacher T and student S, respectively.

To achieve physics-informed supervision, the teacher model is trained following the PINN framework, with details provided in Supplementary Information.

Essentially, the student model is designed to replicate the outputs of the teacher model[28]. The distillation loss function is given by:

$$MSE_T = \frac{1}{M_T} \sum_{i=1}^{M_T} |\tilde{\boldsymbol{u}}_T^i - \tilde{\boldsymbol{u}}_S^i|^2 \tag{51}$$

where M_T denotes the number of configuration points for the computational field.

Consequently, as a staged training strategy (First, the teacher network is used to predict the computational field, and then the student network is trained to learn the results of the teacher network.), the teacher network bears the learning bias of physical information containing high-order gradient terms (like second-order or higher-order derivatives in PDEs), while the student network is allowed to shift towards the gradient direction of parameter regularization. To train and extract meaningful network structures from the student network, further parameter analysis techniques are required.

Structure extraction method

Regularization has been widely used as an effective parameter sparsification technique in network pruning and structural simplification methods[42], effectively optimizing network structures by reducing complexity[43]. However, extracting physically meaningful and generalizable network structures–such as translation equivariance in convolutional networks or rotational invariance–requires a deeper understanding of parameter relationships[26,27]. To address this, Ψ-NN refines the student network's parameter matrices through a specialized clustering approach.

Our structure extraction is based on L2 regularization (parameter smoothing), whose mathematical essence can be derived using the Lagrange multiplier method, promoting parameter convergence within the same layer, as shown in Supplementary Information. The L2 regularization is only applied to the student network, while the teacher network does not use L2 regularization. First, consider L2 regularization on the student network parameters:

$$\Omega(\boldsymbol{\theta}_S) = \sum_{l=1}^{L} \omega_l ||\boldsymbol{\theta}_l||_2^2 \tag{52}$$

where L is the number of layers in the network, ω_l is the regularization weight, and $\boldsymbol{\theta}_l$ is the nonlinear affine transformation parameters of the l-th layer. Under L2 regularization, the parameter vector is stretched along its principal eigenvector direction, thereby enhancing major features while suppressing minor ones[6].

For the evolution trend of parameters under L2 regularization, we have the following theorem:

Theorem 1. For n trainable parameters $\theta_1, \theta_2, ..., \theta_n$ in the same hidden layer, if they play equivalent roles in the network, they will converge under L2 regularization.

Theorem 2. For n trainable parameter values $|\theta_1|, |\theta_2|, ..., |\theta_n|$ in the same hidden layer, if parameter symmetry exists among them, they will converge under regularization.

The proofs are provided in Supplementary Information. Based on the effect of L2 regularization on parameter values (see Appendix D for details), to ensure that parameters representing similar correlations within each neuron share the same value and to further compress the NN, hierarchical agglomerative clustering (HAC)[44] is performed on the absolute values of the weights in each layer. HAC does not require a preset number of clusters and can adaptively retain the necessary cluster centers; it also generates a hierarchical dendrogram, allowing for the selection of an appropriate clustering level through evaluation. We use Euclidean distance to measure the distribution of the data. Details of the clustering algorithm are provided in Supplementary Information.

For the weights $\boldsymbol{\theta}_l$ in the l-th layer, we first compute their absolute values:

$$\boldsymbol{\theta}_l^{\text{abs}} = |\boldsymbol{\theta}_l| \tag{53}$$

Next, we treat $\boldsymbol{\theta}_l^{\text{abs}}$ as feature vectors and apply the HAC clustering algorithm. The clustering process is given by:

$$\{C_1, C_2, \ldots, C_K\} = \text{HAC}(\boldsymbol{\theta}_l^{\text{abs}}, K) \tag{54}$$

where K is the number of clusters and C_k denotes the k-th cluster. After clustering, the absolute values of the weights in each cluster C_k are replaced by the cluster center μ_k:

$$\mu_k = \frac{1}{|C_k|} \sum_{\boldsymbol{\theta}_{l,i} \in C_k} \boldsymbol{\theta}_{l,i} \tag{55}$$

Finally, the updated weights $\boldsymbol{\theta}_l^{\text{new}}$ are given by:

$$\boldsymbol{\theta}_{l,i}^{\text{new}} = \text{sgn}(\boldsymbol{\theta}_{l,i}) \cdot \mu_k \quad \text{if} \quad \boldsymbol{\theta}_{l,i} \in C_k \tag{56}$$

In this way, the n-dimensional trainable parameter vector $\boldsymbol{\theta}_l$ in the l-th layer is reduced to a K-dimensional vector of cluster centers and a two-dimensional sign vector, achieving maximal structural refinement.

After clustering and replacement, the network exhibits a new ordered matrix structure, which may still contain some parameter redundancy and reuse, requiring further analysis. Due to the nature of clustering, using only structured sparsity[26] may overlook the relationships between parameters of adjacent hidden layers, thereby failing to identify physically relevant network structures encoded in these parameter relationships. Therefore, instead of pure low-rank decomposition, we adopt a hybrid compression strategy that combines low-rank constraints with structured parameter sharing. Essentially, this approach compresses and simplifies the weight matrix through parameter sharing and structured design, mapping the high-dimensional weight matrix to a low-dimensional structured subspace. Specifically, Ψ-NN not only reduces the rank of the parameter matrix, but also avoids overly complex network structures by identifying redundant reuse in the form of repeated submatrix basis vectors. The detailed analysis is presented in Section 3.

Different from structured pruning nodes (i.e., complete network substructures of weight-activation-weight), the main goal of Ψ-NN in the parameter matrix extraction process is to identify parameter relationships. To maximize the refinement of parameter relationships, Ψ-NN retains the sign features of parameters and refines the clustering objects to the trainable parameter values between each pair of nodes. On this basis, Ψ-NN can transcend traditional sparse strategies that merely merge repeated nodes, resulting in parameter matrices with physical relevance.

Network reconstruction

The core objective of the network reconstruction stage is to enhance the applicability capability of the network while preserving physical relevance. By structurally reconstructing the extracted parameter relationship matrix, Ψ-NN yields a NN architecture that not only incorporates physical constraints but also adapts to new problems. Unlike approaches that rely solely on parameter pruning or zeroing, Ψ-NN reconstruction emphasizes not just parameter compression, but more importantly, the preservation and explicit representation of physical relationships among parameters. As a result, the reconstructed network both reflects underlying physical laws and enables structural transferability and applicability across different problems.

Specifically, we first evaluate the clustered parameter matrix and sparsify redundant or insignificant parameters by zeroing them out, thereby improving parameter efficiency. For the salient parameter subsets (such as cluster centers), we perform reinitialization to restore their trainability. Meanwhile, the structural relationship matrix $\boldsymbol{R}$ is used to enforce consistency of numerical relationships and physical constraints among nodes in the reconstructed network. This approach ensures that the network retains sufficient degrees of freedom for learning while preserving the extracted physical structure.

The relation matrix $\boldsymbol{R}$ encodes the structural relationships among trainable parameters, with each row essentially derived from a transformed one-hot vector. Specifically: (1) If two sets of parameters are identical, the corresponding rows in $\boldsymbol{R}$ are the same, indicating parameter sharing; (2) If the parameters have equal magnitudes but opposite signs, the corresponding row elements in $\boldsymbol{R}$ are −1, representing a sign-reversal relationship; (3) If there is a permutation relationship between parameters, the relevant rows in $\boldsymbol{R}$ are swapped accordingly, reflecting parameter permutation. Thus, the elements of $\boldsymbol{R}$ are typically −1, 0, or 1, corresponding to inverse, unrelated, and direct relationships, respectively. In this way, $\boldsymbol{R}$ systematically expresses structural information such as parameter sharing, sign, and arrangement, and is used during the structure reconstruction stage to constrain the parameter representation of the new network, thereby achieving structured embedding of physical information. A concrete example of the formation and reconstruction process of the relation matrix is provided in the Laplace case study.

This method preserves the iterative fitting capability of the new network and embeds the physics structure into the neural network via the relation matrix $\boldsymbol{R}$, thus balancing physical consistency with model expressiveness. In this way, Ψ-NN's network reconstruction not only achieves parameter compression and preservation of physical relationships but also significantly enhances the network's applicability and interpretability through a structured adaptive fusion mechanism.

Whole implementation

Through the complete process of distillation, structure extraction, and network reconstruction, the Ψ-NN method ultimately achieves the intrinsic design of network structures with physical constraints. Pseudocode is provided in Supplementary Information to illustrate the entire implementation process. The key steps are summarized as follows:

1. Distillation: In the distillation stage, the choice of teacher network is not unique and depends on the characteristics of the problem. When sufficient understanding of the physical problem exists and a state-of-the-art (SOTA) model is available, selecting the SOTA model is preferable, as its outputs more accurately reflect the true physical scenario. This more precise reconstruction incorporates richer physical information into the generated data, facilitating the extraction of structures that are more physically relevant for the student model. However, when the understanding of the physical problem is limited or an SOTA model is not applicable, more general networks such as PINN or PI-CNN (Physics-informed Convolutional Neural Network[45]) can be used as the teacher network. In this paper, PINNs are used as the teacher network.
2. Structure extraction: The extraction method not only preserves the learned physical structure but also greatly simplifies the network architecture. During extraction, due to the convergence of structural parameters, the HAC clustering algorithm compresses parameter vectors into smaller cluster center vectors, thereby transforming physical features into network parameter matrices and ensuring the physical relevance of the network framework.

3. Network reconstruction: In the reconstruction process, Ψ-NN maximally retains the original parameter relationships while only reinitializing the trainable parameters, resulting in a final network structure that incorporates physical relevance. This approach allows the Ψ-NN structure to fully utilize trainable parameters, enhancing parameter efficiency and enabling applicability across a broader problem space.

Essentially, the Ψ-NN method can be regarded as a specialized form of regularization–integrating physical constraints directly into the internal network structure to produce problem-specific architectures. Ψ-NN offers several advantages: (A) intrinsic structural features–by constraining the structure of the parameter matrix, the model is guided to learn patterns consistent with specific physical laws, ensuring that outputs naturally satisfy physical constraints; (B) interpretability–the combination of submatrices reveals the underlying composition of input features, providing mathematical consistency; and (C) parameter efficiency–parameter sharing reduces model complexity and improves parameter utilization.

Data availability
The sample data used in this study are available in the Github database under accession code https://github.com/ZitiLiu/Psi-NN.

Code availability
All code accompanying this manuscript is publicly available[46].

References

1. Karniadakis, G. E., Kevrekidis, I. G., Lu, L., Perdikaris, P., Wang, S. & Yang, L. Physics-informed machine learning. *Nat. Rev. Phys.* **3**, 422–440 (2021).
2. Zienkiewicz, O.C., Taylor, R.L. The Finite Element Method: Its Basis and Fundamentals. Butterworth-Heinemann, Oxford (2000).
3. Belytschko, T., Krongauz, Y., Organ, D., Fleming, M. & Krysl, P. Meshless methods: An overview and recent developments. *Comput. Methods Appl. Mech. Eng.* **139**, 3–47 (1996).
4. Moin, P. & Mahesh, K. Direct numerical simulation: A tool in turbulence research. *Annu. Rev. Fluid Mech.* **30**, 539–578 (1998).
5. Lookman, T., Balachandran, P. V., Xue, D. & Yuan, R. Active learning in materials science with emphasis on adaptive sampling using uncertainties for targeted design. *npj Comput. Mater.* **5**, 21 (2019).
6. Goodfellow, I., Bengio, Y., Courville, A. Deep Learning. MIT Press, Cambridge, MA (2016).
7. Schmidt, M. & Lipson, H. Distilling free-form natural laws from experimental data. *Science* **324**, 81–85 (2009).
8. Brunton, S. L., Proctor, J. L. & Kutz, J. N. Discovering governing equations from data: Sparse identification of nonlinear dynamical systems. *Proc. Natl Acad. Sci.* **113**, 3932–3937 (2016).
9. Raissi, M., Perdikaris, P. & Karniadakis, G. E. Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations. *J. Comput. Phys.* **378**, 686–707 (2019).
10. Meng, X., Li, Z., Zhang, D. & Karniadakis, G. E. PPINN: Parareal physics-informed neural network for time-dependent PDEs. *Comput. Methods Appl. Mech. Eng.* **370**, 113250 (2020).
11. Zhu, W., Khademi, W., Charalampidis, E. G. & Kevrekidis, P. G. Neural networks enforcing physical symmetries in nonlinear dynamical lattices: The case example of the Ablowitz-Ladik model. *Phys. D: Nonlinear Phenom.* **434**, 133264 (2022).
12. Raissi, M., Yazdani, A. & Karniadakis, G. E. Hidden fluid mechanics: Learning velocity and pressure fields from flow visualizations. *Science* **367**, 1026–1030 (2020).
13. Chen, Y., Huang, D., Zhang, D., Zeng, J., Wang, N., Zhang, H. & Yan, J. Theory-guided hard constraint projection (HCP): A knowledge-based data-driven scientific machine learning method. *J. Comput. Phys.* **445**, 110624 (2021).
14. Shin, Y., Zhang, Z., Karniadakis, G.E. Error estimates of residual minimization using neural networks for linear PDES. J. *Mach. Learn. Model. Comput.* **4**(4) (2023).
15. Zhang, Z.-Y., Zhang, H., Zhang, L.-S. & Guo, L.-L. Enforcing continuous symmetries in physics-informed neural network for solving forward and inverse problems of partial differential equations. *J. Comput. Phys.* **492**, 112415 (2023).
16. Wang, N., Zhang, D., Chang, H. & Li, H. Deep learning of subsurface flow via theory-guided neural network. *J. Hydrol.* **584**, 124700 (2020).
17. Lin, S. & Chen, Y. A two-stage physics-informed neural network method based on conserved quantities and applications in localized wave solutions. *J. Comput. Phys.* **457**, 111053 (2022).
18. Greydanus, S., Dzamba, M., Yosinski, J. Hamiltonian neural networks. Advances in neural information processing systems **32** (2019).
19. Cranmer, M., Greydanus, S., Hoyer, S., Battaglia, P., Spergel, D. & Ho, S. Lagrangian neural networks. *ICLR 2020 Workshop on Integration of Deep Neural Models and Differential Equations* (2019).
20. Chen, Z., Zhang, J., Arjovsky, M. & Bottou, L. Symplectic recurrent neural networks. *ICLR 2020* (2019).
21. Liu, Z. & Tegmark, M. Machine-learning hidden symmetries. *Phys. Rev. Lett.* **128**, 180201 (2022).
22. Hendriks, J., Jidling, C., Wills, A., Schön, T. Linearly Constrained Neural Networks. arXiv. arXiv:2002.01600, 2020. (2021).
23. Rao, C., Ren, P., Wang, Q., Buyukozturk, O., Sun, H. & Liu, Y. Encoding physics to learn reaction-diffusion processes. *Nat. Mach. Intell.* **5**, 765–779 (2023).
24. Michel, A. N., Farrell, J. A. & Porod, W. Qualitative analysis of neural networks. *IEEE Trans. Circuits Syst.* **36**, 229–243 (1989). Conference Name: IEEE Transactions on Circuits and Systems.
25. Hinton, G. E. & Salakhutdinov, R. R. Reducing the dimensionality of data with neural networks. *Science* **313**, 504–507 (2006).
26. Zhang, D., Wang, H., Figueiredo, M., Balzano, L. Learning to share: Simultaneous parameter tying and sparsification in deep learning. In: International Conference on Learning Representations (2018).
27. Fuhg, J. N., Jones, R. E. & Bouklas, N. Extreme sparsification of physics-augmented neural networks for interpretable model discovery in mechanics. *Comput. Methods Appl. Mech. Eng.* **426**, 116973 (2024).
28. Hinton, G., Vinyals, O., Dean, J. Distilling the Knowledge in a Neural Network. arXiv. arXiv:1503.02531 (2015).
29. Kingma, D.P., Ba, J. Adam: A Method for Stochastic Optimization. arXiv. arXiv:1412.6980, 2014. (2017).
30. Shortley, G. H. & Weller, R. The numerical solution of Laplace's Equation. *J. Appl. Phys.* **9**, 334–348 (2004).
31. Liu, Z., Liu, Y., Yan, X., Liu, W., Guo, S. & Zhang, C.-a AsPINN: Adaptive symmetry-recomposition physics-informed neural networks. *Comput. Methods Appl. Mech. Eng.* **432**, 117405 (2024).
32. Hon, Y. C. & Mao, X. Z. An efficient numerical scheme for Burgers' equation. *Appl. Math. Comput.* **95**, 37–50 (1998).
33. Jackson, J.D. Classical Electrodynamics, 3rd edn. Wiley, New York (1999).
34. Xu, Z.-Q. J., Zhang, Y., Luo, T., Xiao, Y. & Ma, Z. Frequency Principle: Fourier Analysis Sheds Light on Deep Neural Networks. *Commun. Comput. Phys.* **28**, 1746–1767 (2020).
35. Rao, C., Sun, H. & Liu, Y. Physics-informed deep learning for incompressible laminar flows. *Theor. Appl. Mech. Lett.* **10**, 207–212 (2020).
36. Elsken, T., Metzen, J. H. & Hutter, F. Neural architecture search: A survey. *J. Mach. Learn. Res.* **20**, 1–21 (2019).
37. Noether, E. Invariant variation problems. *Transp. theory Stat. Phys.* **1**, 186–207 (1971).

38. Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, Ł., Polosukhin, I. Attention is all you need. Advances in neural information processing systems **30**, (2017).
39. Xu, C., Lu, C., Liang, X., Gao, J., Zheng, W., Wang, T. & Yan, S. Multi-loss regularized deep neural network. *IEEE Trans. Circuits Syst. Video Technol.* **26**, 2273–2283 (2016).
40. Li, Y., Yang, J., Wang, D. Self-knowledge distillation enhanced universal framework for physics-informed neural networks. *Nonlinear Dyn.* (2025).
41. Saputra, M.R.U., Gusmao, P.P.B.d., Almalioglu, Y., Markham, A., Trigoni, N. Distilling knowledge from a deep pose regressor network, pp. 263–272 (2019).
42. Wen, W., Wu, C., Wang, Y., Chen, Y., Li, H. Learning structured sparsity in deep neural networks. *Adv. Neural Inf. Process. Syst.* **29**, (2016).
43. LeCun, Y., Denker, J., Solla, S. Optimal brain damage. *Adv. Neural Inf. Process. Syst.* **2**, (1989).
44. Ward Jr, J. H. Hierarchical grouping to optimize an objective function. *J. Am. Stat. Assoc.* **58**, 236–244 (1963).
45. Zhu, Y., Zabaras, N., Koutsourelakis, P.-S. & Perdikaris, P. Physics-constrained deep learning for high-dimensional surrogate modeling and uncertainty quantification without labeled data. *J. Comput. Phys.* **394**, 56–81 (2019).
46. Liu, Z., Liu, Y., Yan, X., Liu, W., Nie, H., Guo, S., Zhang, C.-A. Automatic network structure discovery of physics informed neural networks via knowledge distillation. Psi-NN code repository, https://doi.org/10.5281/zenodo.17098398 (2025).

Acknowledgements
This work is supported by the Chinese Academy of Sciences Project for Young Scientists in Basic Research (No.YSBR-107), the Strategic Priority Research Program of Chinese Academy of Science (No.XDB0620402), and the Youth Innovation Promotion Association Chinese Academy of Science (No.2023023). The authors would also like to acknowledge Dr. Yang Xuan for his creation and design of the Featured Image.

Author contributions
Ziti Liu: Conceptualization, Data curation, Investigation, Methodology, Software, Writing - original draft. Yang Liu: Data curation, Writing - original draft. Xunshi Yan: Software, Formal analysis, Visualization, Supervision. Wen Liu: Investigation, Supervision. Han Nie: Supervision, Validation. Shuaiqi Guo: Visualization, Validation. Chen-an Zhang: Funding acquisition, Investigation, Supervision.

Competing interests
The authors declare no competing interests.

Additional information
Supplementary information The online version contains supplementary material available at https://doi.org/10.1038/s41467-025-64624-3.

Correspondence and requests for materials should be addressed to Xunshi Yan or Chen-an Zhang.

Peer review information : *Nature Communications* thanks Kiran Bacsa and the other anonymous reviewer(s) for their contribution to the peer review of this work. A peer review file is available.