

Physics-informed Kolmogorov-Arnold Network with Chebyshev Polynomials for Fluid Mechanics

Chunyu Guo^{a,c}, Lucheng Sun^{a,c}, Shilong Li^{b,c}, Zelong Yuan^{a,c}, Chao Wang^{b,c}

^a*Qingdao Innovation and Development Base, Harbin Engineering University, Sansha Road, West Coast New District, Qingdao, 266000, China*

^b*College of Shipbuilding Engineering, Harbin Engineering University, Nantong Street, Nangang District, Harbin, 150001, China*

^c*Key Laboratory of underwater propulsion Technology, Ministry of Industry and Information Technology, Nantong Street, Nangang District, Harbin, 150001, China*

Abstract

Solving partial differential equations (PDEs) is essential in scientific forecasting and fluid dynamics. Traditional approaches often incur expensive computational costs and trade-offs in efficiency and accuracy. Recent deep neural networks improve accuracy but require quality training data. Physics-informed neural networks (PINNs) effectively integrate physical laws, reducing data reliance in limited sample scenarios. A novel machine-learning framework, Chebyshev physics-informed Kolmogorov-Arnold network (ChebPIKAN), is proposed to integrate the robust architectures of Kolmogorov-Arnold networks (KAN) with physical constraints to enhance calculation accuracy of PDEs for fluid mechanics. We explore the fundamentals of KAN, emphasis on the advantages of using the orthogonality of Chebyshev polynomial basis functions in spline fitting, and describe the incorporation of physics-informed loss functions tailored to specific PDEs in fluid dynamics, including Allen-Cahn equation, nonlinear Burgers equation, two-dimensional Helmholtz equations, two-dimensional Kovasznay flow and two-dimensional Navier-Stokes equations. Extensive experiments demonstrate that the proposed ChebPIKAN model significantly outperforms standard KAN architecture in solving various PDEs by embedding essential physical information more effectively. These results indicate that augmenting KAN with physical constraints can not only alleviate overfitting issues of KAN but also improve extrapolation performance. Consequently, this study highlights the potential of ChebPIKAN as a powerful tool in computational fluid dynamics, proposing a path toward fast and reliable predictions in fluid mechanics and beyond.

Keywords: physics-informed neural network, Kolmogorov-Arnold network, machine learning, computational fluid dynamics, fluid mechanics

1. Introduction

Solving partial differential equations (PDEs) is crucial to make accurate physical predictions of a dynamic system and obtain high-fidelity, effective physical information. In the field of fluid dynamics, computational fluid dynamics (CFD) is an important method for solving dynamic systems consisting of partial differential equations and obtaining fluid dynamics data, including numerical techniques such as finite-difference scheme, finite-volume method, finite-element method [1]. Although classical CFD approaches can be precise enough, they require a significant amount

of computational resources to numerically implement. Although many accelerated algorithms and reduced-order models are proposed to improve the computationa efficiency of CFD [2–8], they also sacrifice the accuracy of PDE calculations. In practical engineering applications, some physical information of the complicated dynamic system, such as initial conditions, boundary conditions and governing equations, is unknown [9], rendering CFD methods unsuitable for calculations. Even when various models are employed to enhance computational efficiency, numerical calculations are still time-consuming. Therefore, alternative methods are needed to achieve fast flow field predictions.

In recent years, rapidly developing machine learning (ML) techniques have quickly gained attention in fluid mechanics research due to its minimal requirements for physical information and fast response capabilities. At present, in the field of mathematics and practical engineering problems, deep neural networks have begun to be widely applied in solving partial differential equations due to their excellent nonlinear fitting and interpolation capabilities. Srinivasan et al.[10] evaluated the ability of deep neural networks to predict temporal turbulent flows, generated training data using a nine-equation shear flow model, and tested multi-layer perceptron (MLP) and long short-term memory (LSTM) networks. The results indicate that LSTM is superior to MLP in predictions of turbulence statistics and dynamic behaviors due to its ability to utilize the sequential nature of data, with relatively small relative errors. This exploratory study establishes the foundation for the future development of accurate and effective data-driven sub-grid scale models, aimed at applying them to more complex large-eddy simulations of wall turbulence. Bhatnagar et al.[11] proposed a convolutional-neural-network (CNN) based model to predict the airfoil flow field, using Reynolds-averaged Navier Stokes (RANS) solutions on airfoil profiles as training data. This model automatically extracts mapping features with minimal human supervision and can predict velocity and pressure fields faster than RANS solvers, while enhancing prediction capabilities through specific convolution operations and parameter sharing. Santos et al.[12] introduced PoreFlow Net, a three-dimensional (3D) convolutional neural network architecture that significantly accelerates the flow field prediction process by extracting the spatial relationship between porous media morphology and fluid velocity field. This approach eliminates the need for costly numerical simulations and demonstrates the successful application of physics-based machine learning models in digital rock physics. Kim et al.[13] employed CNN framework to predict local heat flux through direct-numerical-simulation data, demonstrating that accuracy was preserved even at higher Reynolds numbers. These results not only enhanced the understanding of turbulent heat transfer but also provided a novel method for turbulence modeling. Erichson et al.[14] proposed a data-driven method using shallow neural networks to reconstruct fluid flow fields from limited measurement data, eliminating the requirement for complex preprocessing. Kashefi et al. [15] proposed a deep learning framework based on the PointNet architecture for predicting flow field in irregular domains, utilizing point cloud inputs to map spatial positions to CFD quantities. This method effectively preserves geometric information and trains much faster than traditional CFD solvers, without losing prediction accuracy.

Even though applying deep neural networks to predict fluid mechanics has achieved satisfied results and reduced computational costs, the accuracy of neural networks still relies on the support of a large amount of high fidelity data. Currently, the scarcity of sample data is increasingly hindering the development of neural networks. Physics-informed

neural network (PINN) [16] was proposed and quickly gained widespread attention due to its “benevolent” requirement for data. From the perspective of loss function, PINN framework can be viewed as an extension of artificial neural network, where the incorporation of a physical-information based penalty term significantly enhances the network’s adherence to physical laws [17]. By leveraging physical laws, PINN can perform effectively in scenarios with insufficient measurement data, reducing the excessive dependence of classical data-driven methods on training data [18–22]. In terms of numerical computation, PINN also exhibits excellent capabilities in forward problem solving and inversion. Moreover, PINN’s meshless nature is particularly well-suited for wave equations, facilitating the application to various initial and boundary conditions [23]. PINN possesses seamless encoding and formulation capabilities under various constraint conditions, combined with their meshless nature and simplicity of implementation. These features make it highly effective in solving laminar flows with strong pressure gradients and achieving high accuracy in turbulence modeling [24]. In terms of fluid experiments, Wang H et al.[25] accurately reconstructed high-resolution velocity fields in sparse particle image velocimetry (PIV) particle flow experimental data using PINN constrained by the Navier-Stokes (NS) equations. The impacts of activation function, optimization algorithm and hyperparameters were evaluated systematically through two-dimensional (2D) Taylor decaying vortices and turbulent channel flow, exploring the ability of PINN to reconstruct wall-bounded turbulence.

Based on the excellent capability of PINN in solving physical equations, numerous improved network architectures based on PINN have been proposed successively. Yuan L et al.[26] proposed auxiliary physics-informed neural networks (A-PINN) by introducing auxiliary output variables to represent numerical integrals in governing equations and using automatic differentiation of auxiliary outputs instead of integral operators, avoiding the limitation of using integral discretization. Employing this neural network to solve nonlinear integral differential equations achieves better accuracy than PINN. Rezaei S et al.[27], inspired by the finite element method, sought to improve the performance of existing PINNs by advocating the use of spatial gradients of the principal variables as the output of the separable neural network. By properly designing the network architecture in PINN, deep learning models can solve unknown problems in heterogeneous domains without any available initial data from other sources. The physics-informed neural network based topology optimization (PINNTO) proposed by combining PINN with topology optimization solves several inherent defects due to the addition of PINN [28]. The accuracy of PINN network also highly depends on the selection and distribution of training sample points. Y Liu et al.[29] proposed a novel adaptive sampling algorithm called expected improvement residual based adaptive refinement (EI-RAR), which combines an attention mechanism with sample point generation using a new expected improvement function. This approach addresses the limitation of traditional adaptive sampling algorithms, which typically focus only on sample points within the solution domain. Furthermore, expected improvement Gradient adaptive sampling algorithm, which integrates residual gradient information has been proposed to enhance the nonlinear fitting accuracy of PINN in discontinuous solution regions. J Bai et al.[30] extended the PINN algorithm to the solid mechanics and proposed the LSWR loss function based on the least squares weighted residual (LSWR) method. Comparisons with energy-based and collocation loss functions have demonstrated the potential of the LSWR loss function in accurately predicting stress and displacement fields.

Although numerous neural network architectures have been developed to enhance the performance of PINN for different problems, their fundamental structure remains rooted in the original PINN framework. PINN encounters the issue of gradient vanishing during the training process and is highly sensitive to the selection of various hyperparameters. However, extensive parameters are required manually tuning and currently there is no established optimal parameter selection guideline for PINN [31]. Inspired by the Kolmogorov-Arnold representation theorem, Kolmogorov-Arnold networks (KANs) have fundamentally transformed the traditional MLP architecture by fitting the weight coefficients of the learning activation function with learnable univariate function spline curves at the edges, rather than using fixed activation activation functions at the nodes as in conventional multilayer perceptron (MLP) [32]. As a result, KAN do not require the use of any nonlinear activation functions , bringing them closer to physical laws and aligning better with human intuition. In traditional neural network development, the nonlinear mapping between inputs and outputs is often an obscure mapping that lacks interpretability, and the use of nonlinear activation functions cannot be easily explained from a physical perspective. This implies that traditional neural networks function as “black boxes”, making their results inherently difficult to interpret and less convincing [33]. Moreover, since the learning process of KAN’s spline fitting is grounded in mathematical theorems, the KAN network architecture inherently possesses physical significance. From a machine learning perspective, incorporating physical information as constraints in neural networks is a significant development trend. This approach offers several advantages, such as introducing physical constraints to the loss function, thereby enhancing the physical interpretability of neural networks [34]. However, KAN might still requires a substantial amount of training data to achieve ideal results due to the absence of physical information constraints. Inspired by the concepts of PINN, incorporating machine learning with prior knowledge such as physical information, represents a critical research approach [35]. Although KAN itself possess inherent physical significance, we believe it is essential to incorporate physical information constraints into KAN, similar to the enhancements seen in PINN compared to conventional deep neural networks. In this paper, we propose a novel Kolmogorov-Arnold network architecture based on chebyshev polynomials, namely Chebyshev physics-informed Kolmogorov-Arnold network (ChebPIKAN). The proposed ChebPIKAN framework adopts Kolmogorov-Arnold network as the basic network architecture and incorporate essential physical information based on PINN. KAN can adaptively learn the nonlinear activation functions in comparison with the classical neural network, and automatically prune the basis functions and weight coefficients to determine the optimal network architecture. The proposal of KAN fundamentally solves the problem of poor generalization caused by global optimization of traditional neural networks, since the spline curves learned by KAN are essentially local, and the fitted splines learned by the network are independent [32]. The proposed ChebPIKAN framework leverages KAN as its foundational architecture inherits the advantages of the KAN structure and achieves enhanced performance due to the incorporation of physical information.

The remainder of the paper is organized as follows. In Section II, we will introduce the mathematical foundation of the KAN network architecture and detail the physical information loss incorporated into each physical PDE. Section III compares the results of PDE predictions using KAN with chebyshev basis function and ChebPIKAN, highlighting

the advantages of integrating physical information into neural networks and the benefits of KAN architecture over traditional MLP architectures. ChebPIKAN with added physical information outperforms both KAN and PINN in solving accuracy. Section IV further discusses in detail the relevant work of ChebPIKAN, its impact, and future development directions. Conclusions are drawn in Section V.

2. Methodology

In this section, we will describe in detail how we implement the development of Physics-informed Kolmogorov-Arnold networks with Chebyshev polynomials and used it to solve partial differential equations. An overview of the KAN network architecture is first illustrated. We then described how we use Chebyshev polynomials as alternative basis functions. Finally, we introduced our implementation of Physics-informed KAN networks with Chebyshev polynomials by applying different physical loss functions.

2.1. Kolmogorov-Arnold Networks

The design of Kolmogorov-Arnold networks (KANs) is inspired by the Kolmogorov-Arnold representation theorem. This theorem posits that any multivariate continuous function \mathbf{f} defined on a bounded domain can be expressed as a finite composition of continuous univariate functions \mathbf{a} . For smooth $f : [0, 1]^n \rightarrow \mathbf{R}$, it is represented as

$$f(x) = f(x_1, \dots, x_n) = \sum_{q=1}^{2n+1} \Phi_q \left(\sum_{p=1}^n \phi_{q,p}(x_p) \right), \quad (1)$$

where $\phi_{q,p} : [0, 1] \rightarrow \mathbf{R}$ and $\phi_q : \mathbf{R} \rightarrow \mathbf{R}$. The aim is to identify suitable univariate functions $\phi_{q,p}$ and Φ . We represent the operations of the KAN layer's node as $\phi(x)$, the input of the KAN layer is x , leading to the mathematical formula for the l -th KAN layer

$$x_{l+1} = \underbrace{\begin{pmatrix} \phi_{11}^{(l)} & \phi_{12}^{(l)} & \cdots & \phi_{1N_l}^{(l)} \\ \phi_{21}^{(l)} & \phi_{22}^{(l)} & \cdots & \phi_{2N_l}^{(l)} \\ \vdots & \vdots & & \vdots \\ \phi_{N_{l+1}1}^{(l)} & \phi_{N_{l+1}2}^{(l)} & \cdots & \phi_{N_{l+1}N_l}^{(l)} \end{pmatrix}}_{\Phi_l} x_l, \quad (2)$$

where Φ_l is the operation matrix for the l -th KAN layer, containing learnable parameters.

2.2. Chebyshev Polynomials Basis Functions

In KAN layer calculations, it is essential to select spline basis functions and activation functions for fitting, along with their weight coefficients ω . We define the function as

$$\phi(x) = \omega[b(x) + \text{spline}(x)], \quad (3)$$

where ω is the weight coefficient learned through the neural network. It is important to note that x here represents the input value of the function ϕ , rather than a vector containing multiple arguments as in section 2.1. The $spline(x)$ is modeled as a linear combination of B-splines functions, with the activation function defined by

$$b(x) = \text{silu}(x) = x / (1 + e^{-x}). \quad (4)$$

The goal of neural networks is to learn appropriate weight coefficients ω to approximate the target function. However, B-spline function is not an orthogonal basis function and has a certain linear correlation, which might lead to multicollinearity issues during the fitting process. Multicollinearity can make parameter estimation unstable, increase the variance of the fitting results, and make it difficult to explain the contributions of individual basis function. In practical applications, high-frequency oscillations (such as Runge phenomenon) are likely to occur. To address this, we utilize Chebyshev polynomials, which provide orthogonality and enhance interpolation performance.

Chebyshev polynomials consist of two polynomial sequences related to cosine and sine functions, denoted as $T_n(x)$ and $U_n(x)$. They can be defined in several equivalent ways. The $T_n(x)$ Chebyshev polynomial is derived by the recursive formula

$$T_0(x) = 1, \quad T_1(x) = x, \quad T_{n+1}(x) = 2xT_n(x) - T_{n-1}(x), \quad (5)$$

with $x \in [-1, 1]$, and $T_n(x) \in [-1, 1]$. It is worth noting that, since the initial input x or the output after a single KAN layer can easily exceed the value range of the Chebyshev polynomial, we apply the tanh activation function to the output of the KAN layer and the first input x_0 to limit it within the range of $[-1, 1]$. Therefore, the input of each KAN layer undergoes additional tanh activation

$$\phi_0(x_0) = \omega\{b[\tanh(x_0)]\} + \text{cheb}[\tanh(x_0)], \quad (6)$$

$$\phi_n(x_n) = \omega\{b[\tanh[\phi_{n-1}(x_{n-1})]]\} + \text{cheb}\{\tanh[\phi_{n-1}(x_{n-1})]\}, \quad (7)$$

where $n \geq 1$ and $\text{cheb}(\cdot)$ is the Chebyshev polynomial mentioned in Eq (5).

2.3. Physics-informed KAN Network

Inspired by the principle of physics-informed neural networks training by minimizing the loss function that enforces adherence to physical laws, we apply physical losses to ChebPIKAN. The loss function is defined as

$$\text{Loss} = \text{Loss}_{\text{data}} + \lambda \text{Loss}_{\text{phy}}, \quad (8)$$

where $\text{Loss}_{\text{data}}$ and Loss_{phy} represent the data and physical information loss terms respectively. λ is the weight coefficient for the PDE residual term.

The physics-informed loss(i.e. Loss_{phy}) should be carefully designed to the specific equations being solved. To examine the performance of ChebPIKAN to solve PDEs in fluid mechanics, we take the two-dimensional incompressible Navier-Stokes equations as the solution objective and encode the corresponding physical loss function.

Navier-Stokes (N-S) equations constitute a set of equations that govern the motion of viscous incompressible fluids and describe the conservation of mass and momentum. These equations occupy a pivotal position in fluid dynamics, as they are used to characterize the flow behavior of fluid substances such as liquids and air.

The general form of the Navier-Stokes equation is given by

$$\frac{\partial \mathbf{u}}{\partial t} + \mathbf{u} \cdot \nabla \mathbf{u} = -\frac{1}{\rho} \nabla p + \nu \nabla^2 \mathbf{u}, \quad (9)$$

$$\nabla \cdot \mathbf{u} = 0. \quad (10)$$

Here, ρ represents density, t represents time and \mathbf{u} represents the velocity vector, in the three-dimensional case, it can be expressed as $\mathbf{u} = u\mathbf{i} + v\mathbf{j} + w\mathbf{k}$. ∇ denotes the gradient operator, p represents pressure.

In order to further validate performance of our proposed PIAKN for PDEs in fluid dynamics. We also considered the one-dimensional Allen-Cahn equation, Burgers equation, as well as two-dimensional Helmholtz equation and Kovasznay flow.

The general form of the Allen-Cahn equation is expressed as

$$\frac{\partial u}{\partial t} = d \frac{\partial^2 u}{\partial x^2} + f(u), \quad (11)$$

where u is a function represents the material properties, x is the spatial coordinate, and $f(u)$ is a function of u .

The Burgers equation can be expressed in its general form as follows

$$\frac{\partial u}{\partial t} + u \frac{\partial u}{\partial x} = \nu \frac{\partial^2 u}{\partial x^2}, \quad (12)$$

where u denotes fluid velocity, ν is the viscosity coefficient, and x is the spatial coordinate.

Two-dimensional Helmholtz equation is expressed as

$$\nabla^2 \varphi + k^2 \varphi = 0, \quad (13)$$

where φ represents arbitrary scalar field, and k is wave number.

The Kovasznay flow governing equations can be expressed as follows

$$u \frac{\partial u}{\partial x} + v \frac{\partial u}{\partial y} = -\frac{\partial p}{\partial x} + \frac{1}{Re} \left(\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} \right), \quad (14)$$

$$u \frac{\partial v}{\partial x} + v \frac{\partial v}{\partial y} = -\frac{\partial p}{\partial y} + \frac{1}{Re} \left(\frac{\partial^2 v}{\partial x^2} + \frac{\partial^2 v}{\partial y^2} \right), \quad (15)$$

This equation is a simplified form of the Navier Stokes equation in fluid dynamics, describing the conservation of momentum in fluid motion. Here, Re is the Reynolds number, which characterizes the relative importance of inertia and viscous forces in flow.

We encode the respective physical loss functions based on each physical differential equation, together with the KAN network using Chebyshev polynomials as basis functions, and establish our ChebPIKAN framework.

For each physical equation, we design a varying number of hidden layers from shallow to deep, aiming to compare and select the optimal number of hidden layers. The Adam optimizer is employed for parameter adjustments. The specific parameters are presented in the Table 1, where ILR represents the initial learning rate, $Data_{train}$ and $Data_{test}$ represent the number of data points used for training and testing, respectively.

Table 1: training parameters

Types of PDEs	$Data_{train}$	$Data_{test}$	Number of Layers	Optimizer	ILR	Epoch	λ
Allen-Cahn equation	225	1000	[2] + [5] * (1~4) + [1]	Adam	1e-3	10000	0.1
Burgers equation	225	1000	[2] + [5] * (1~4) + [1]	Adam	1e-3	10000	0.1
Helmholtz equation	225	1000	[2] + [5] * (1~4) + [1]	Adam	1e-3	10000	0.1
Kovasznay flow	225	1500	[2] + [5] * (1~4) + [3]	Adam	1e-3	10000	0.1
Navier-Stokes equations	2000	5000	[3] + [5] * (1~4) + [3]	Adam	1e-3	10000	0.1

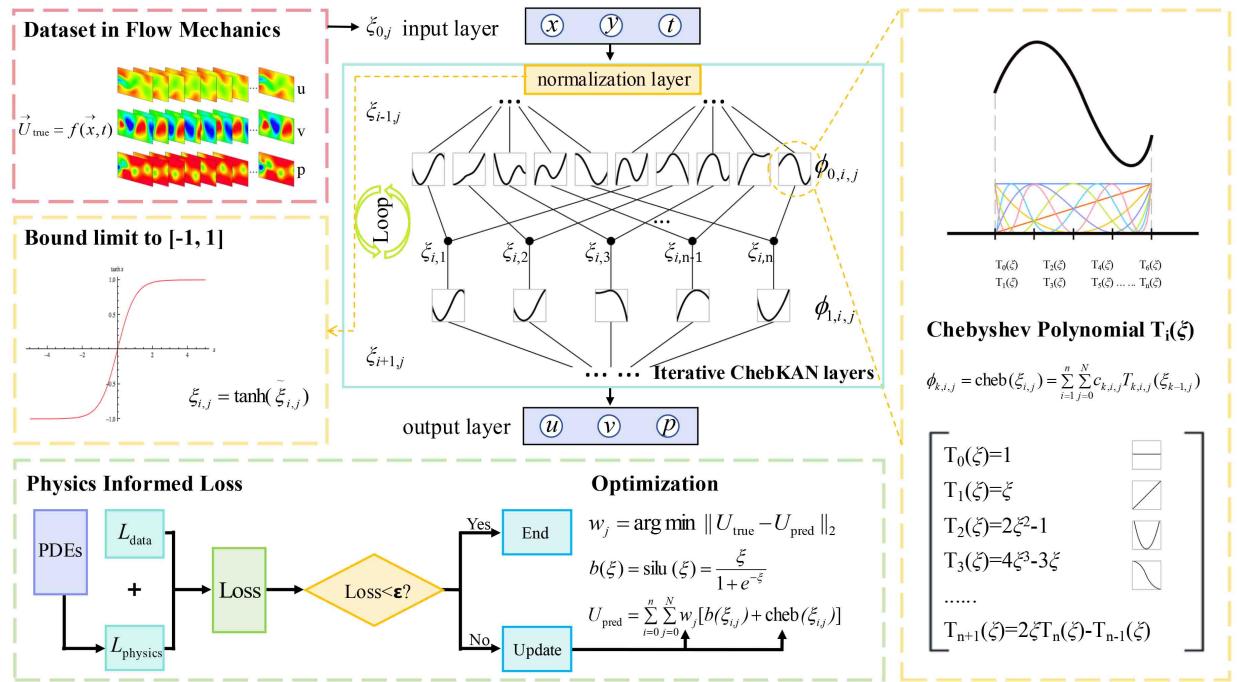


Figure 1: A schematic of ChebPIKAN. ChebPIKAN has physical loss functions and Chebyshev polynomial basis functions. The blue box represents the network architecture of ChebPIKAN. The yellow boxes on both sides represent bound limitation using hyperbolic tangent function and Chebyshev polynomials of the network architecture, respectively. The red box represents the original input data of fluid mechanics for the network. The green box below shows the operation process of the network, which includes the addition of physical constraint.

The experiments are conducted using cloud computing resources, specifically utilizing an NVIDIA GeForce RTX

4090 GPU with 24GB of VRAM. The system operates on a Linux environment.

3. Physical Models and Training Process of ChebPIKAN Models

In this section, we present five physical models used in our study, the one-dimensional Allen-Cahn and Burgers equations, along with the two-dimensional Helmholtz equation, Kovasznay flow, and Navier-Stokes equations. We utilize Kolmogorov-Arnold Networks (KAN) with Chebyshev basis functions, namely our ChebPIKAN approach, to solve these equations, thereby demonstrating the benefits of incorporating physical information into the neural network.

For each equation, we explore five different network architectures for both KAN and ChebPIKAN to assess how varying the number of hidden layers affects performance. Throughout this study, we maintain consistent hyperparameters and training strategies for both networks, ensuring a fair comparison. Each neural network is configured with five neurons per hidden layer.

To evaluate performance, we employ the relative error in the residual contour plot,

$$\mathbf{u}_{\text{res}} = \left(\frac{\sqrt{(\mathbf{u}_{\text{pred}} - \mathbf{u}_{\text{true}})^2}}{\sqrt{\|\mathbf{u}_{\text{true}}\|}} \right) \times 100, \quad (16)$$

where \mathbf{u}_{res} , \mathbf{u}_{true} , \mathbf{u}_{pred} denote the residual, ground truth and prediction of the solution vector. $\|\cdot\|$ represents the overall Euclidean-norm value.

3.1. One-dimensional Allen-Cahn Equation

The Allen-Cahn (AC) equation is a pivotal partial differential equation widely applied in materials science, physical chemistry, and geometry. It effectively models the evolution of interfaces during phase transitions, particularly focusing on single-phase transitions. This study uses the AC equation to assess the network's capability in handling one-dimensional nonlinear problems, with the reference solution derived from computational fluid dynamics (CFD) simulations.

Our investigation focuses on a specific scenario of Eq (11) characterized by

$$\frac{\partial u}{\partial t} = d \frac{\partial^2 u}{\partial x^2} + 5(u - u^3), \quad x \in [-1, 1], \quad t \in [0, 1], \quad (17)$$

with initial and boundary conditions defined as

$$u(x, 0) = x^2 \cos(\pi x), \quad u(-1, t) = u(1, t) = -1. \quad (18)$$

To align our model with the underlying physical process, we develop corresponding loss terms based on the parameters of physical models. By minimizing discrepancies in the equation, we can ensure consistency with the physical model. The physical loss term is defined as

$$Loss_{PDE} = d \frac{\partial^2 u}{\partial x^2} + 5(u - u^3) - \frac{\partial u}{\partial t}. \quad (19)$$

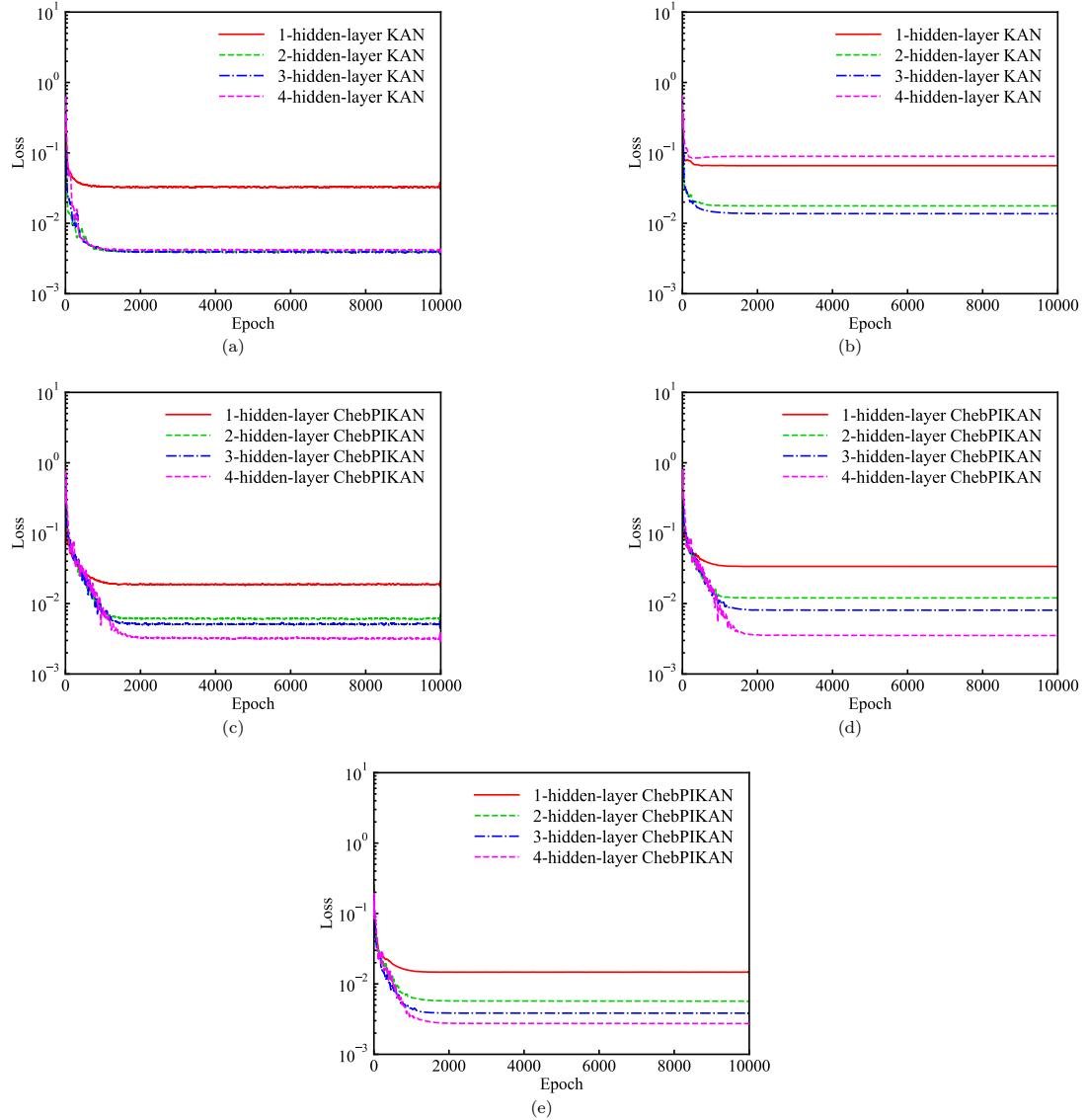


Figure 2: Among them, (a) and (b) respectively represent the train loss and test loss of KANs containing different hidden layers. (c), (d), and (e) respectively represent the train loss, test loss and PDE loss of ChebPIKANs containing different hidden layers.

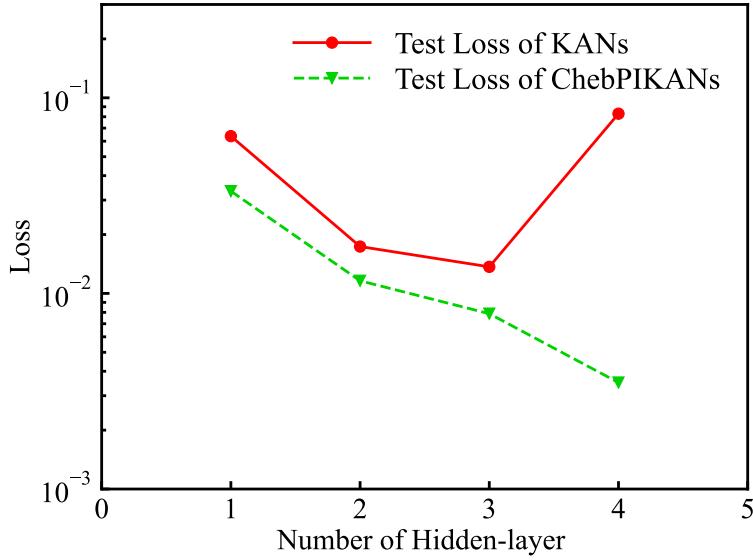


Figure 3: The optimal test loss of KANs and ChebPIKANs with different hidden layers

Figures 2 and Figures 3 reveal that all KAN neural network structures exhibit overfitting, particularly pronounced in those with three hidden layers. This suggests that the extrapolation performance of KAN is suboptimal. In contrast, ChebPIKAN, which incorporates physical information, effectively mitigates overfitting, notably reducing the test loss associated with three hidden layers. By integrating physical information, ChebPIKAN enhances the network's ability to capture the model's essential characteristics, leading to improved extrapolation performance. Additionally, ChebPIKAN consistently shows lower loss values than KAN, aligning with our expectation that the inclusion of physical information strengthens network performance. We also observe that the training loss and PDE loss remain stable, further confirming that this integration helps prevent overfitting. Ultimately, the most effective ChebPIKAN configuration, featuring four hidden layers, achieves loss convergence around 10^{-3} .

The prediction results of KAN and ChebPIKAN are presented in Figure 4. The error plots indicate that ChebPIKAN, with its incorporation of physical information, delivers more accurate predictions than KAN, evidenced by significantly reduced error distributions and values. As the number of hidden layers increases, KAN's predictions deteriorate due to increased complexity in learning parameters. In contrast, ChebPIKAN maintains stability, with prediction accuracy improving steadily as more hidden layers are added, achieving optimal performance at four hidden layers.

The average residuals for KAN and ChebPIKAN across different hidden layer configurations are summarized in Table 2. Notably, ChebPIKAN consistently demonstrates lower mean residuals than KAN, particularly with four hidden layers, where KAN shows increased residuals, a phenomenon not observed in ChebPIKAN.

For this study, we considered a computational domain of $[-1, 1]$ and a time interval of $[0, 1]$. Within this domain, 225 real value points were randomly sampled for the training set. For ChebPIKAN, 800 PDE points were randomly selected to calculate the physical loss. The Adam optimizer is employed for its efficiency, utilizing 10^4 epoch iterations

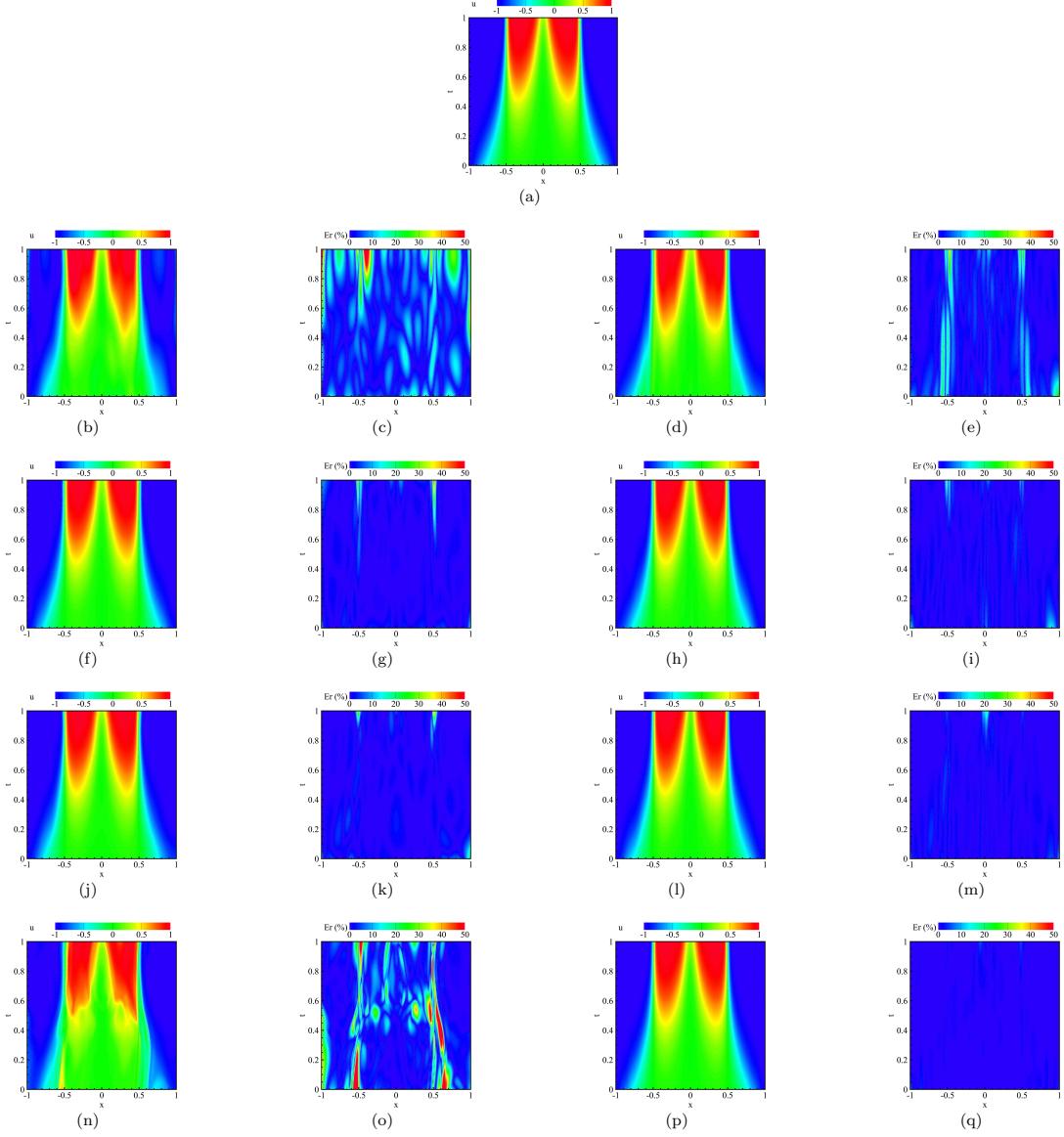


Figure 4: Among them, (a) represents the true value of the AC equation. The left two columns of the remaining figure represent the predicted figure and error figure of KAN containing 1-4 hidden layers, from top to bottom. The two columns on the right, from top to bottom, represent the predicted figure and error figure of ChebPIKAN with 1-4 hidden layers, respectively.

Table 2: The average residuals table of KANs and ChebPIKANs with different hidden layers

Number of Hidden-layer	KANs	ChebPIKANs
1	6.15%	2.69%
2	1.01%	0.94%
3	0.93%	0.73%
4	6.14%	0.34%

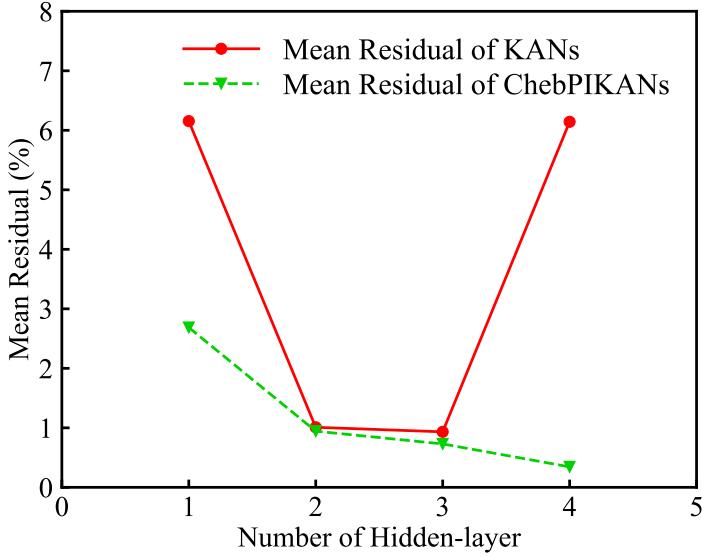


Figure 5: The average residuals of KANs and ChebPIKANs with different hidden layers

with an initial learning rate of 0.01 and a correction coefficient of 0.95. The learning rate is adjusted downwards every 10 steps once the loss curve stabilizes.

We explore the effects of varying KAN and ChebPIKAN structures on network performance by configuring four different network architectures, each containing one to four hidden layers. Theoretically, deeper networks possess stronger nonlinear processing capabilities. However, with four hidden layers, KAN faces challenges due to an increase in parameters, leading to poor training performance. Conversely, ChebPIKAN exhibits a steady decrease in loss with additional hidden layers, indicating that the integration of physical information enhances the network's learning capability. This results in improved overall nonlinear processing ability, demonstrating the benefits of incorporating hidden layers into the architecture.

3.2. One-dimensional Burgers Equation

The Burgers equation is a fundamental equation in soliton theory, representing a diffusion process that includes nonlinear terms.

In our study, we define the spatial and temporal domains of Eq (12) as

$$x \in [-1, 1], \quad t \in [0, 1], \quad (20)$$

and v is set at $\frac{0.01}{\pi}$, with the Dirichlet boundary conditions and initial conditions specified as

$$u(-1, t) = u(1, t) = 0, \quad u(x, 0) = -\sin(\pi x). \quad (21)$$

Given the shock wave characteristics of the Burgers equation, rapid changes in the solution occur in specific regions. Neural networks can effectively address these high-gradient areas by adjusting their predictions accordingly.

The presence of an accurate analytical solution for this boundary condition facilitates performance validation of the neural network when solving the Burgers equation. In ChebPIKAN, the physical loss associated with the Burgers equation is defined as

$$Loss_{PDE} = \nu \frac{\partial^2 u}{\partial x^2} - \frac{\partial u}{\partial t} - u \frac{\partial u}{\partial x}. \quad (22)$$

For the neural network, the inputs of the Burgers equation are position x and time t , while the output is the fluid velocity u . The domain and random sampling strategies for the Burgers equation align with those used for the Allen-Cahn equation, and we apply the Adam optimization strategy consistently.

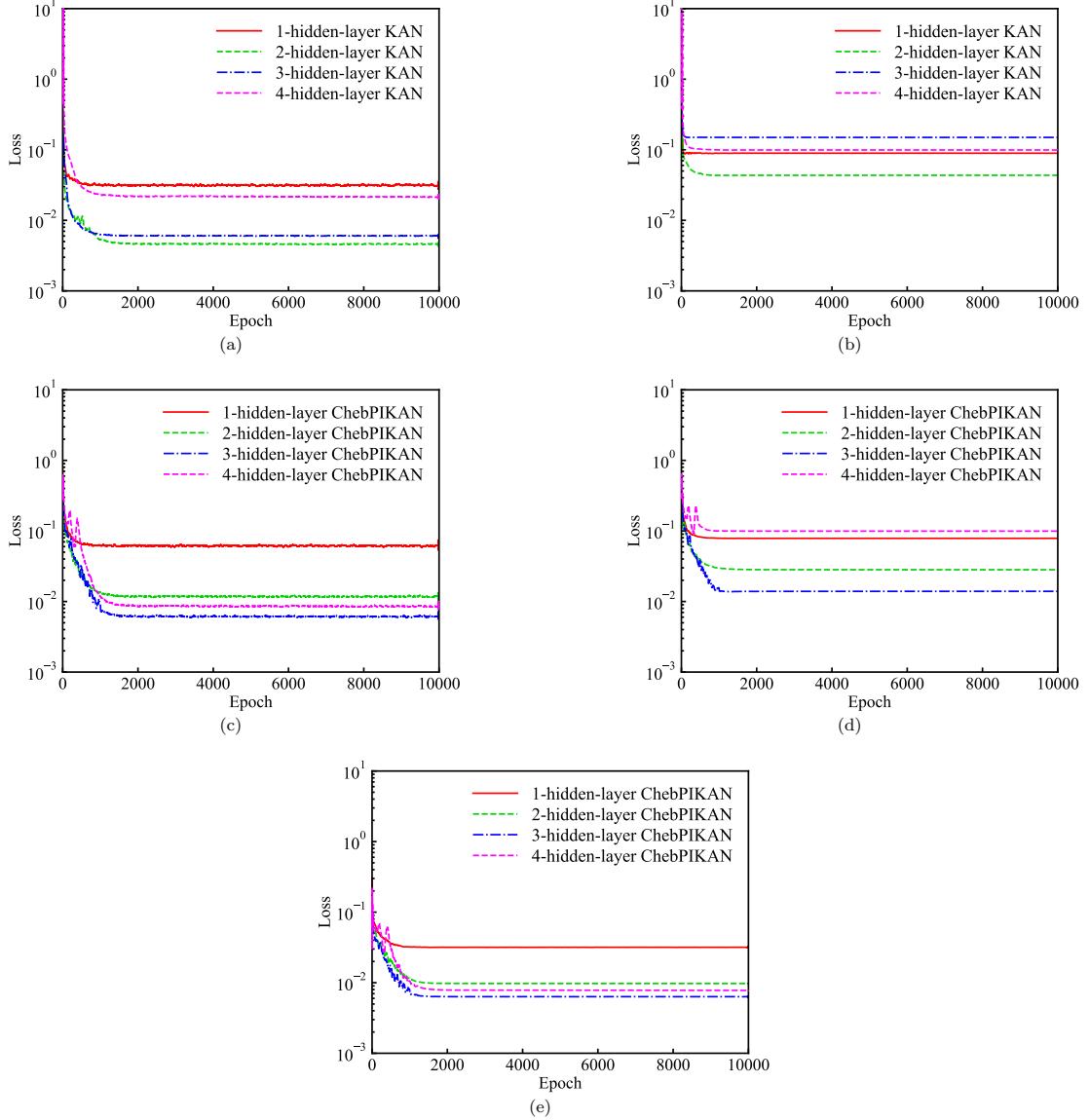


Figure 6: Among them, (a) and (b) respectively represent the train loss and test loss of KANs containing different hidden layers. (c), (d), and (e) respectively represent the train loss, test loss and PDE loss of ChebPIKANs containing different hidden layers.

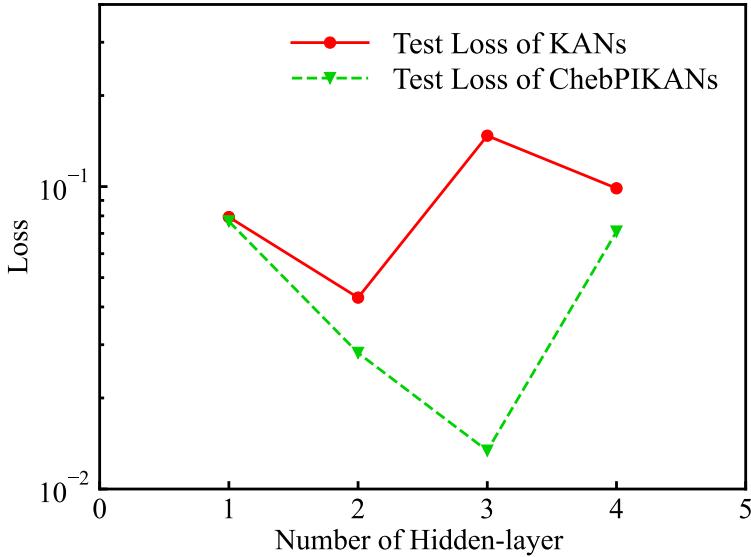


Figure 7: The optimal test loss of KANs and ChebPIKANs with different hidden layers

By observing the loss curve, as shown in the Figure 6 and Figure 7, reveals that the final loss is less than ideal, with most network configurations achieving a loss of around 10^{-2} , close to 10^{-1} . The KAN's overfitting in this context, which is more pronounced than in the Allen-Cahn equation, ChebPIKAN shows resilience to overfitting, except with four hidden layers. The Burgers equation presents substantial challenges due to its strong nonlinear phenomena and potential for shock wave generation at low viscosity coefficients. Consequently, as the network architecture deepens, prediction accuracy improves significantly. ChebPIKAN shows clear advantages with three hidden layers, as this depth enhances its nonlinear processing capabilities, however, the structure with four hidden layers suffers from complications arising from increased learning parameters.

The train loss and PDE loss curves exhibit a high degree of alignment, indicating that the inclusion of physical information mitigates overfitting in ChebPIKAN, thereby enhancing generalization. This suggests that the network effectively learns the characteristics of the partial differential equations involved.

In Figure 8, we observe that error concentrations occur at discontinuities in the shear region. While all networks face challenges in predicting strong shear areas, the error distribution for ChebPIKAN is significantly lower. In the optimal three-layer configuration, predictions are nearly accurate across the board, except for a small region of error in the strong shear area.

Table 3 presents the average residuals for KAN and ChebPIKAN with varying numbers of hidden layers. When employing a single hidden layer, the performances of KAN and ChebPIKAN are comparable, suggesting similar predictive capabilities in solving the Burgers equation. However, as the number of hidden layers increases, ChebPIKAN exhibits significantly improved performance, underscoring its strengths in high-level feature extraction.

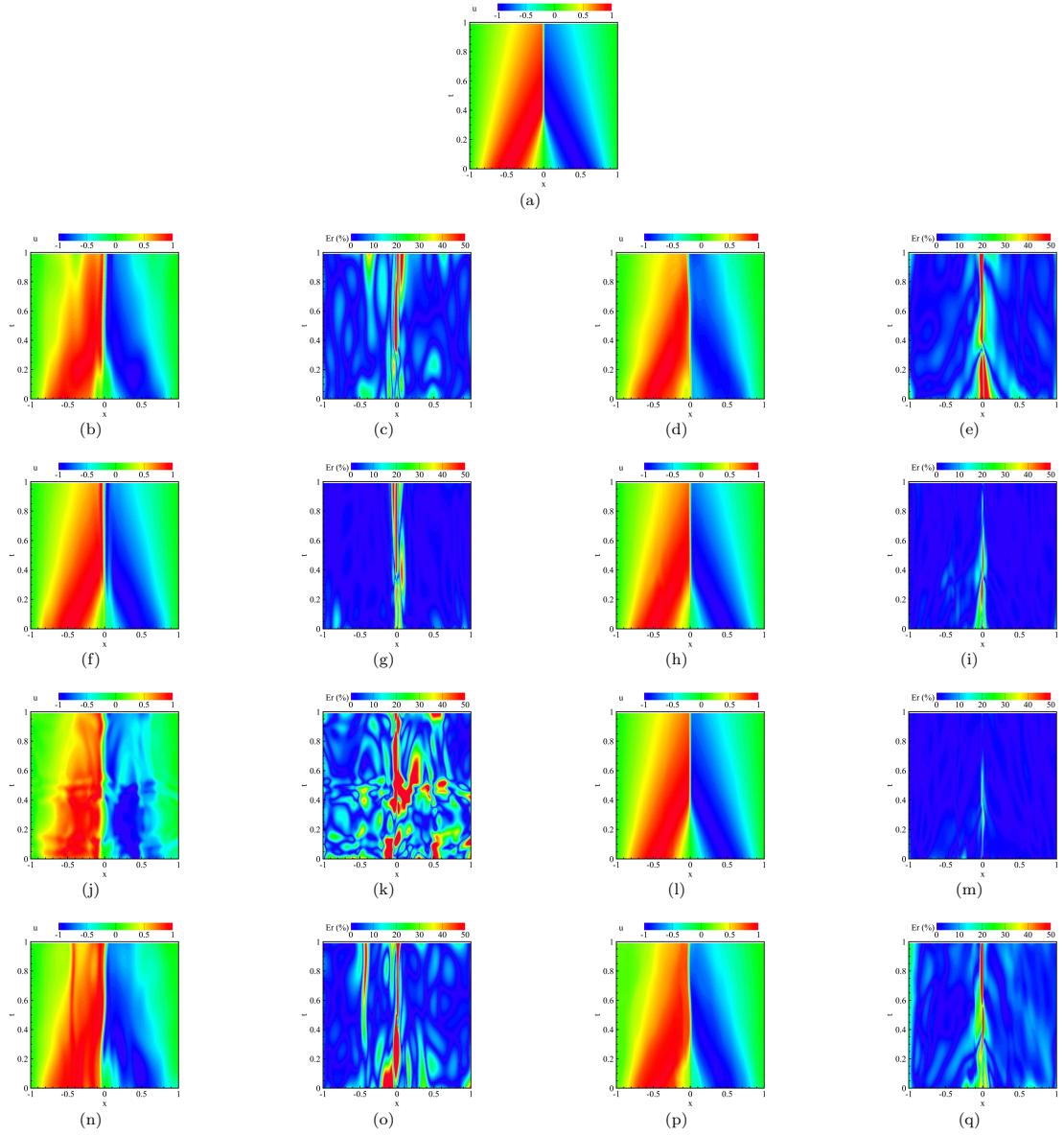


Figure 8: Among them, (a) represents the true value of the Burgers equation. The left two columns of the remaining figure represent the predicted figure and error figure of KAN containing 1-4 hidden layers, from top to bottom. The two columns on the right, from top to bottom, represent the predicted figure and error figure of ChebPIKAN with 1-4 hidden layers, respectively.

Table 3: The average residuals table of KANs and ChebPIKANs with different hidden layers

Number of Hidden-layer	KANs	ChebPIKANs
1	6.36%	6.50%
2	2.86%	1.76%
3	13.67%	1.13%
4	7.72%	6.45%

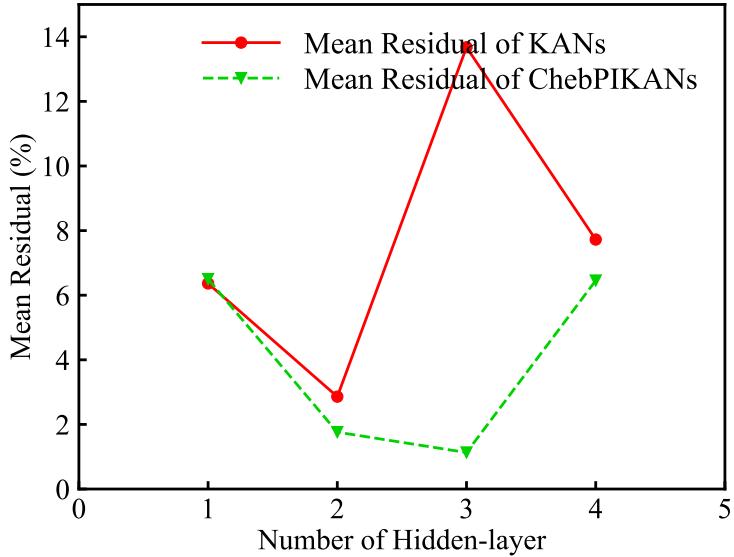


Figure 9: The average residuals of KANs and ChebPIKANs with different hidden layers

3.3. Two-dimensional Helmholtz Equation

The Helmholtz equation is a key partial differential equation used in various fields, including physics, engineering, and computer graphics. It describes how a scalar or vector physical quantity changes in a constant external field, allowing for a better understanding and resolution of wave phenomena.

To streamline computations, we focus on the two-dimensional case of Eq (13) with a wave number defined as $k_0 = 2\pi n$, where $n = 2$. Under this condition, the equation can be simplified to

$$-u_{xx} - u_{yy} - k_0^2 u = f, \quad \Omega = [0, 1]^2, \quad (23)$$

with Dirichlet boundary conditions specified as

$$u(x, y) = 0, \quad (x, y) \in \partial\Omega, \quad (24)$$

and the source term given by

$$f(x, y) = k_0^2 \sin(k_0 x) \sin(k_0 y). \quad (25)$$

In this case, the exact solution of the equation is

$$u(x, y) = \sin(k_0 x) \sin(k_0 y), \quad (26)$$

In physical applications such as heat conduction and wave equations, the terms $-u_{xx}$ and u_{yy} represent changes in local curvature or temperature in the x and y directions, reflecting the system's diffusion or propagation. Meanwhile, $k_0^2 u$ typically affects the system's equilibrium state, acting as a restoring force or potential energy influence, while f is

an externally driven source term that represents the external influence or force applied in space, and can be understood as the distribution of external stimuli or sources.

This framework validates the effectiveness of our ChebPIKAN in solving the Poisson equation. We analyze the spatial domain defined by $x \in [0, 1]$ and $y \in [0, 1]$, with network inputs at these positions and outputs at velocity u . The sampling points and optimization strategies align with the previously discussed methodology.

The physical loss for the Helmholtz equation in ChebPIKAN is defined as

$$Loss_{PDE} = k_0^2 \sin(k_0 x) \sin(k_0 y) + u_{xx} + u_{yy} + k_0^2 u. \quad (27)$$

Figure 10 illustrates the train loss and test loss for KANs and ChebPIKANs with various hidden layers. Notably, KAN networks exhibit more pronounced overfitting in two-dimensional cases, particularly with four hidden layers. Due to the increased complexity of two-dimensional equations compared to one-dimensional equations, both KAN and ChebPIKAN with four hidden layers demonstrate improved performance. The PDE loss and train loss of ChebPIKAN generally show strong consistency, indicating that the network has effectively learned the equation.

As seen in the Figure 12, ChebPIKAN generally yields smaller prediction errors compared to KAN, except for networks with a single hidden layer. This disparity can be attributed to the limitations of one hidden layer in two-dimensional scenarios. Furthermore, the addition of physical information in ChebPIKAN significantly enhances predictive accuracy, although higher error rates in certain areas indicate challenges in capturing non-stationary changes. Increasing the density of sampling points, particularly near vortex edges, may improve this aspect.

Table 4: The average residuals table of KANs and ChebPIKANs with different hidden layers

Number of Hidden-layer	KANs	ChebPIKANs
1	14.39%	12.35%
2	15.46%	5.55%
3	29.90%	4.29%
4	20.01%	5.01%

In assessing performance across various hidden layers, ChebPIKAN maintains stability and low error rates, particularly excelling at the second and third layers. In contrast, KAN's performance fluctuates significantly, especially at higher layers, indicating instability during training and a notable decline at the third layer.

3.4. Two-dimensional Kovasznay Flow

The Kovasznay flow is an idealized model in fluid dynamics, notable for its two-dimensional, steady-state, incompressible, and vortical characteristics. It serves as a vital framework for analyzing flow stability, transitions to turbulence, and flow control. This model reaches a steady state over time, at which point the velocity and pressure fields become constant. The governing equations can be expressed as Eq (14) and Eq (15) with Dirichlet boundary

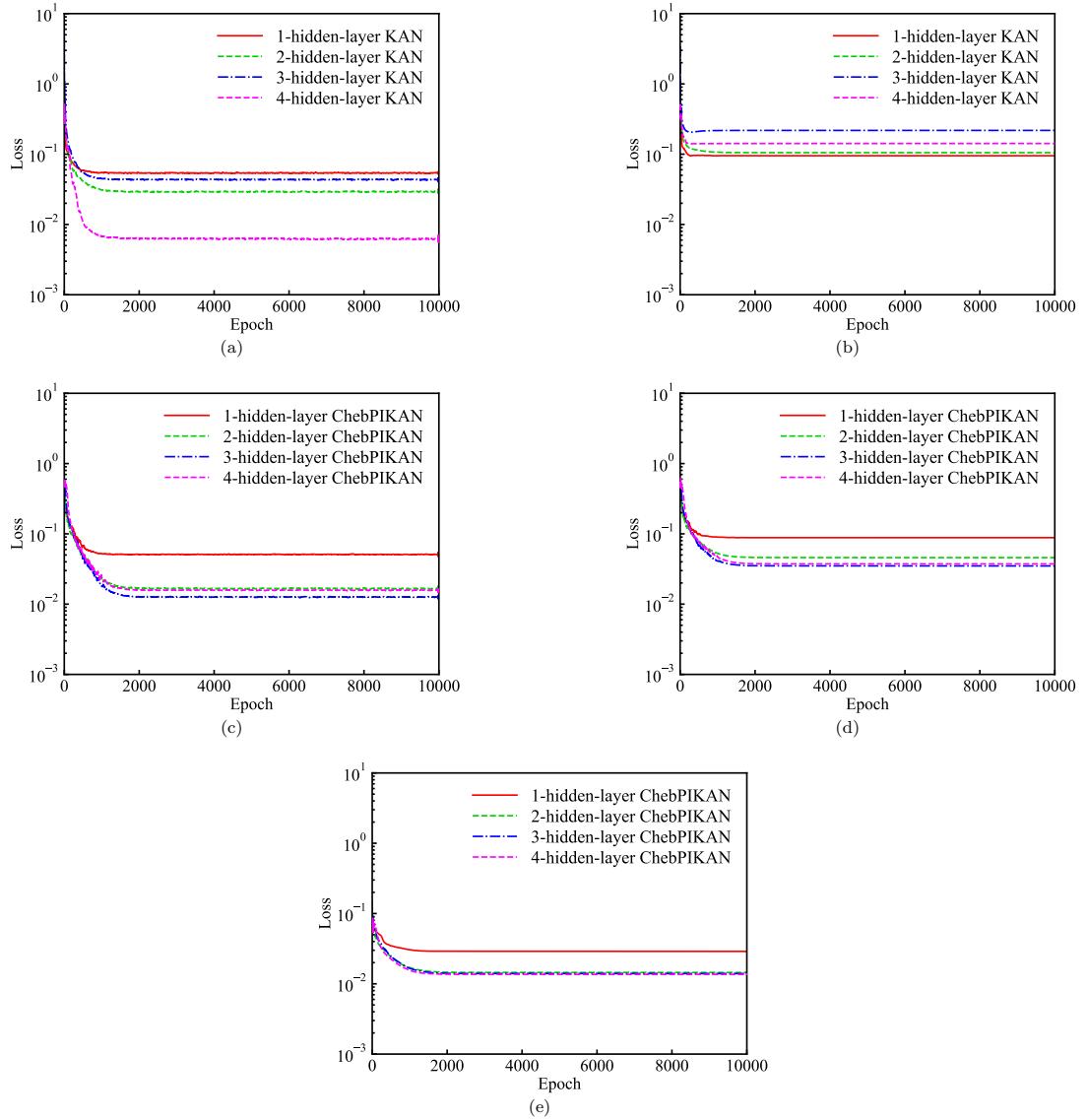


Figure 10: two-dimensional Helmholtz equation loss. Among them, (a) and (b) respectively represent the train loss and test loss of KANs containing different hidden layers. (c), (d), and (e) respectively represent the train loss, test loss and PDE loss of ChebPIKANs containing different hidden layers.

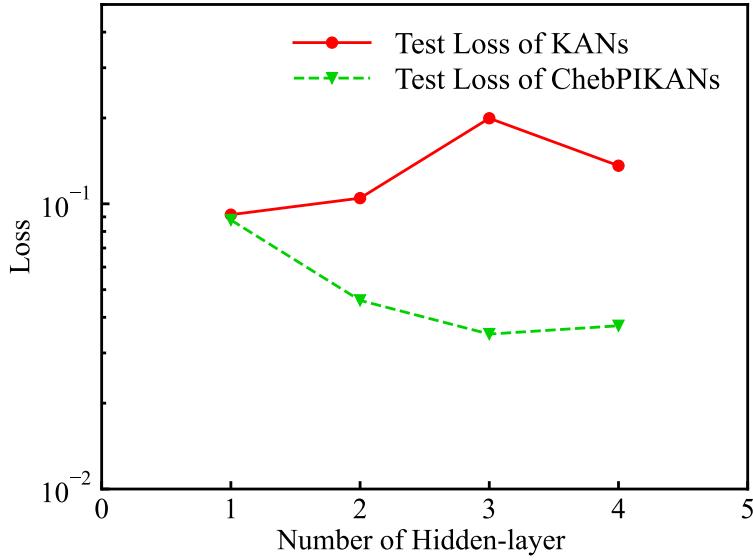


Figure 11: The optimal test loss of KANs and ChebPIKANs with different hidden layers

conditions

$$u(x, y) = 0, \quad (x, y) \in [0, 1]^2. \quad (28)$$

The exact solutions for the velocity components u, v and pressure p are given by

$$u = 1 - e^{\lambda x} \cos(2\pi y), \quad (29)$$

$$v = \frac{\lambda}{2\pi} e^{\lambda x} \sin(2\pi y), \quad (30)$$

$$p = \frac{1}{2} \left(1 - e^{2\lambda x} \right), \quad (31)$$

where $\lambda = \frac{1}{2v} - \sqrt{\frac{1}{4v^2} + 4\pi^2}$.

This equation is a simplified form of the Navier Stokes equation in fluid dynamics, describing the conservation of momentum in fluid motion. Taking the equation (14) as an example, where $u \frac{\partial u}{\partial x}$ and $v \frac{\partial u}{\partial y}$ represents the convective term, it describes how changes in velocity in the fluid affect the momentum of the flow, $\frac{\partial p}{\partial x}$ represents the pressure gradient term, and difference is forced to be one of the main reasons for fluid motion, $\frac{1}{Re} \left(\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} \right)$ represents the viscous dissipation term, indicating the influence of fluid viscosity on momentum transfer. Here, Re is the Reynolds number, which characterizes the relative importance of inertia and viscous forces in flow. This item reflects the momentum diffusion and dissipation caused by shear in fluids.

In our analysis, we define the spatial domain as $x \in [0, 1]$ and $y \in [0, 1]$. The same random sampling and optimization strategies are employed as previously mentioned. The neural network model for the Kovasznay flow utilizes two inputs, x and y , and three outputs, Pressure p , velocity u and velocity v . In the ChebPIKAN framework, the physical

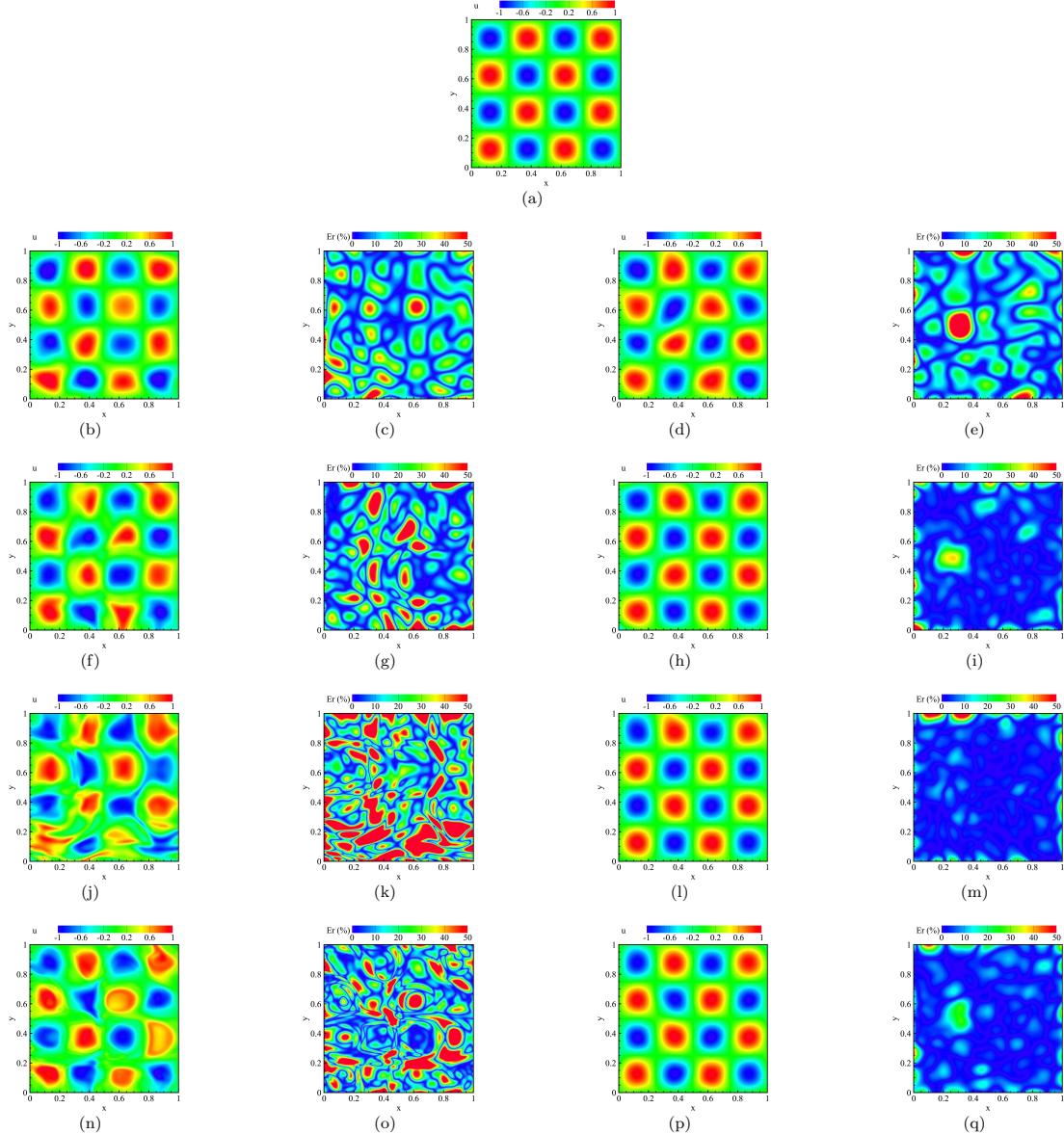


Figure 12: two-dimensional Helmholtz equation contour. Among them, (a) represents the true value of the Helmholtz equation. The left two columns of the remaining figure represent the predicted figure and error figure of KAN containing 1-4 hidden layers, from top to bottom. The two columns on the right, from top to bottom, represent the predicted figure and error figure of ChebPIKAN with 1-4 hidden layers, respectively.

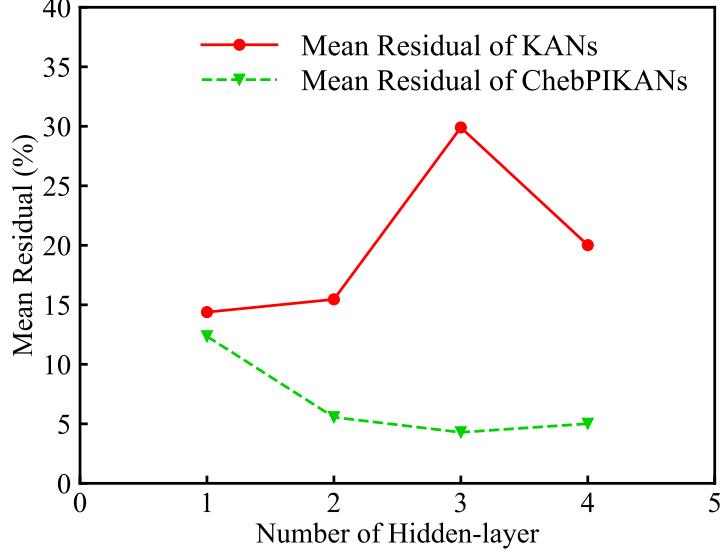


Figure 13: The average residuals of KANs and ChebPIKANs with different hidden layers

loss associated with the Kovasznay flow is expressed as

$$\begin{aligned}
 Loss_{PDE} = & u \frac{\partial u}{\partial x} + v \frac{\partial u}{\partial y} + \frac{\partial p}{\partial x} - \frac{1}{Re} \left(\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} \right) \\
 & + u \frac{\partial v}{\partial x} + v \frac{\partial v}{\partial y} + \frac{\partial p}{\partial y} - \frac{1}{Re} \left(\frac{\partial^2 v}{\partial x^2} + \frac{\partial^2 v}{\partial y^2} \right) \\
 & + \nabla \cdot \vec{u}.
 \end{aligned} \tag{32}$$

Given the complexity of Kovasznay flow solutions, prediction errors are often more pronounced at the boundaries. To address this, boundary constraints are incorporated into the ChebPIKAN framework by selecting random boundary coordinates and applying the governing equations of Kovasznay flow. This integration of boundary constraints enhances the physical information used to compute the PDE loss, directly improving the accuracy of boundary forecasts and influencing internal predictions.

Figure 14 illustrating the loss curves for Kovasznay flow demonstrate that neither the KAN nor ChebPIKAN models exhibit severe overfitting, with ChebPIKAN performing better, particularly in architectures with three or four hidden layers.

The contour of the flow velocity u predicted by the network is illustrated in Figure 16. It vividly shows that both networks accurately predict the mainstream u , with ChebPIKAN, enhanced by additional physical information, exhibiting virtually no prediction error. In contrast, KAN shows a slight error at the boundary of the strong shear region on the left. The contour of the non-mainstream flow velocity v predicted by the network is shown in Figure 17. Given that v has a smaller magnitude compared to the mainstream u and exhibits a more complex solution form than both u and p , both networks demonstrate reduced accuracy in predicting v compared to u and p . However, the error in ChebPIKAN is significantly smaller than that in KAN. Notably, after incorporating boundary constraints, the

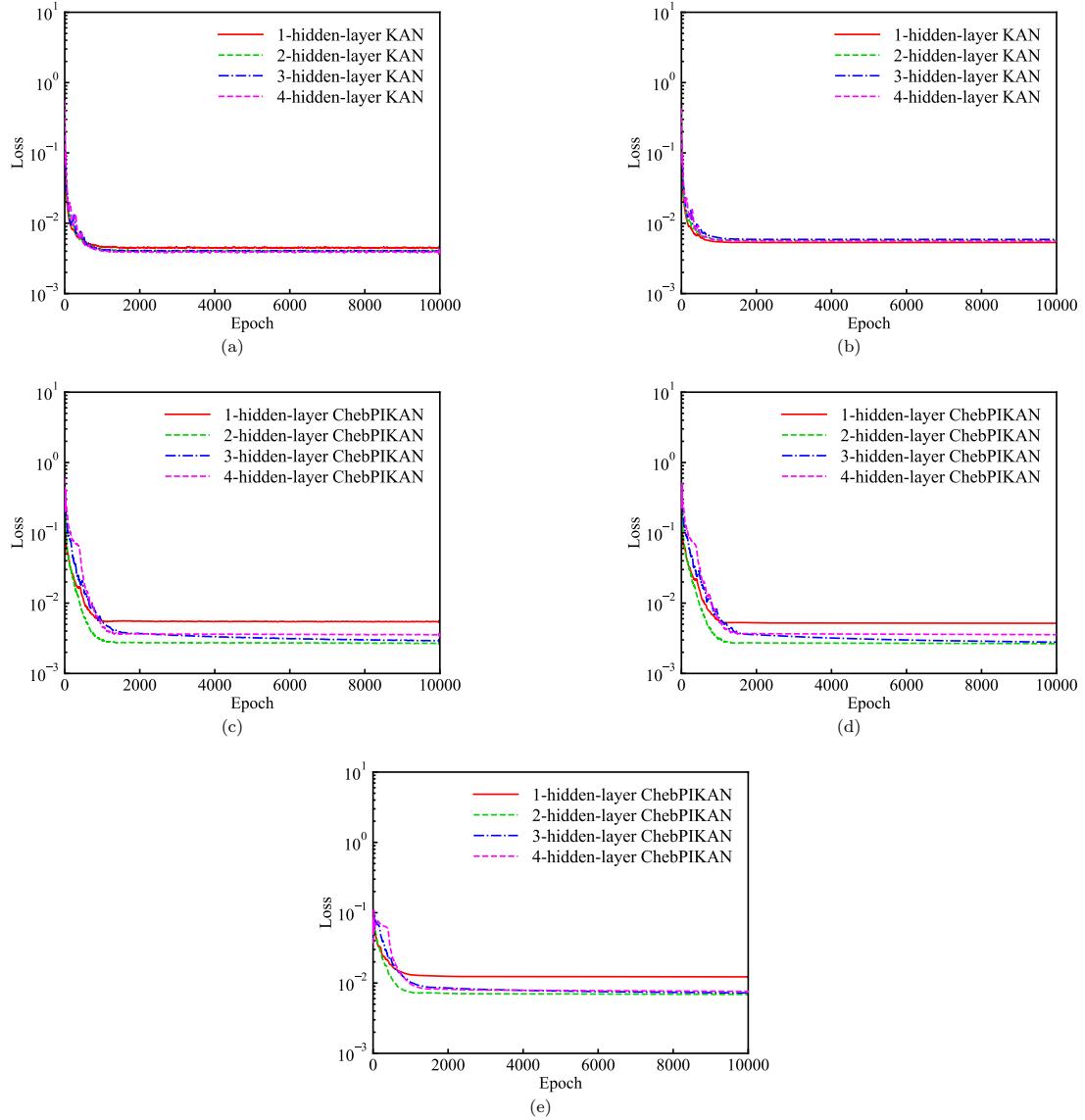


Figure 14: two-dimensional Kovasznay flow equation loss. Among them, (a) and (b) respectively represent the train loss and test loss of KANs containing different hidden layers. (c), (d), and (e) respectively represent the train loss, test loss and PDE loss of ChebPIKANs containing different hidden layers.

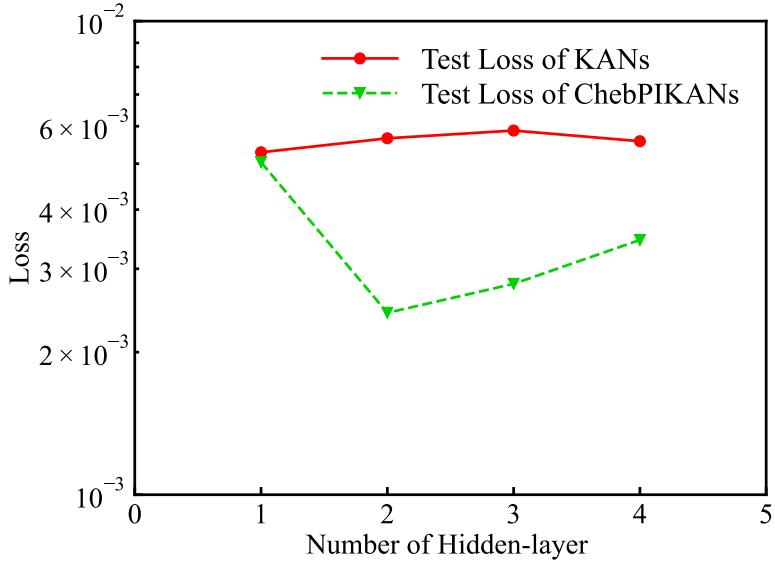


Figure 15: The optimal test loss of KANs and ChebPIKANs with different hidden layers

boundary error in ChebPIKAN shows considerable improvement.

Table 5: The average residuals table of KANs and ChebPIKANs with different hidden layers

Number of Hidden-layer	KANs_u	ChebPIKANs_u	KANs_v	ChebPIKANs_v	KANs_p	ChebPIKANs_p
1	0.33%	0.42%	4.05%	3.37%	0.64%	1.08%
2	0.40%	0.19%	3.36%	1.77%	0.72%	0.48%
3	0.35%	0.20%	3.74%	1.41%	0.65%	0.67%
4	0.32%	0.18%	3.34%	2.71%	0.80%	0.81%

For pressure prediction, the simpler form of the pressure solution results in relatively ideal predictions from both networks. Nonetheless, ChebPIKAN demonstrates a significantly smaller overall error, particularly at the boundary, compared to KAN. The accurate solution of Kovasznay flow underscores the strong capability of the KAN architecture, and the addition of physical information sets a solid foundation for tackling other fluid mechanics equations in the future.

In the results for u and p , both network types exhibit similar performance. However, for v , which exhibits larger average residuals and presents a greater challenge for prediction, ChebPIKAN significantly outperforms KAN, particularly with two and three layers. This highlights ChebPIKAN's robust model stability, as it better adapts to the complexities introduced by additional hidden layers, especially in predicting both u and v . In contrast, KAN's performance shows minimal improvement and can become unstable in some cases.

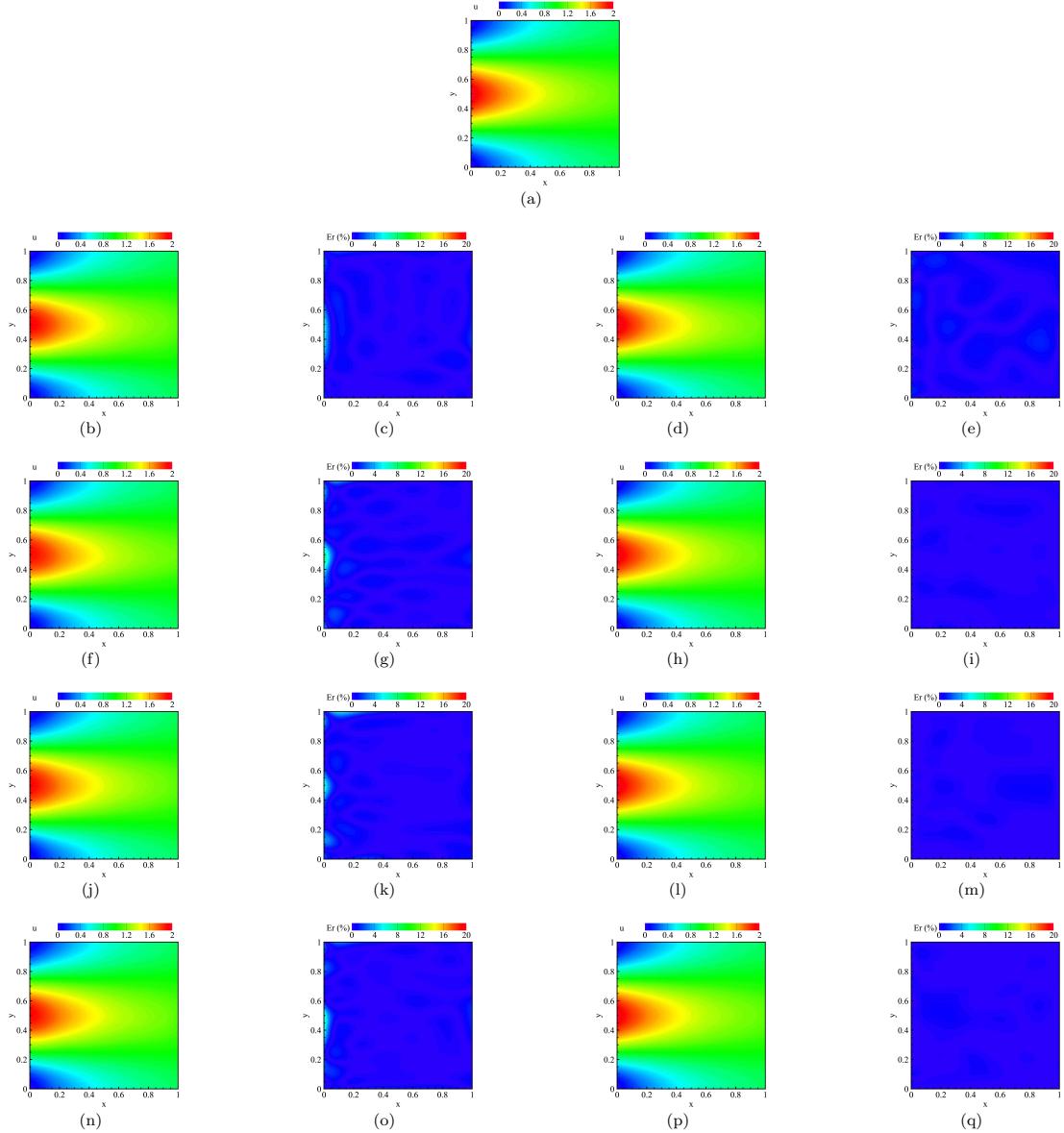


Figure 16: two-dimensional Kovasznay flow velocity of the x direction, u contour. Among them, (a) represents the true value of the Helmholtz equation. The left two columns of the remaining figure represent the predicted figure and error figure of KAN containing 1-4 hidden layers, from top to bottom. The two columns on the right, from top to bottom, represent the predicted figure and error figure of ChebPIKAN with 1-4 hidden layers, respectively.

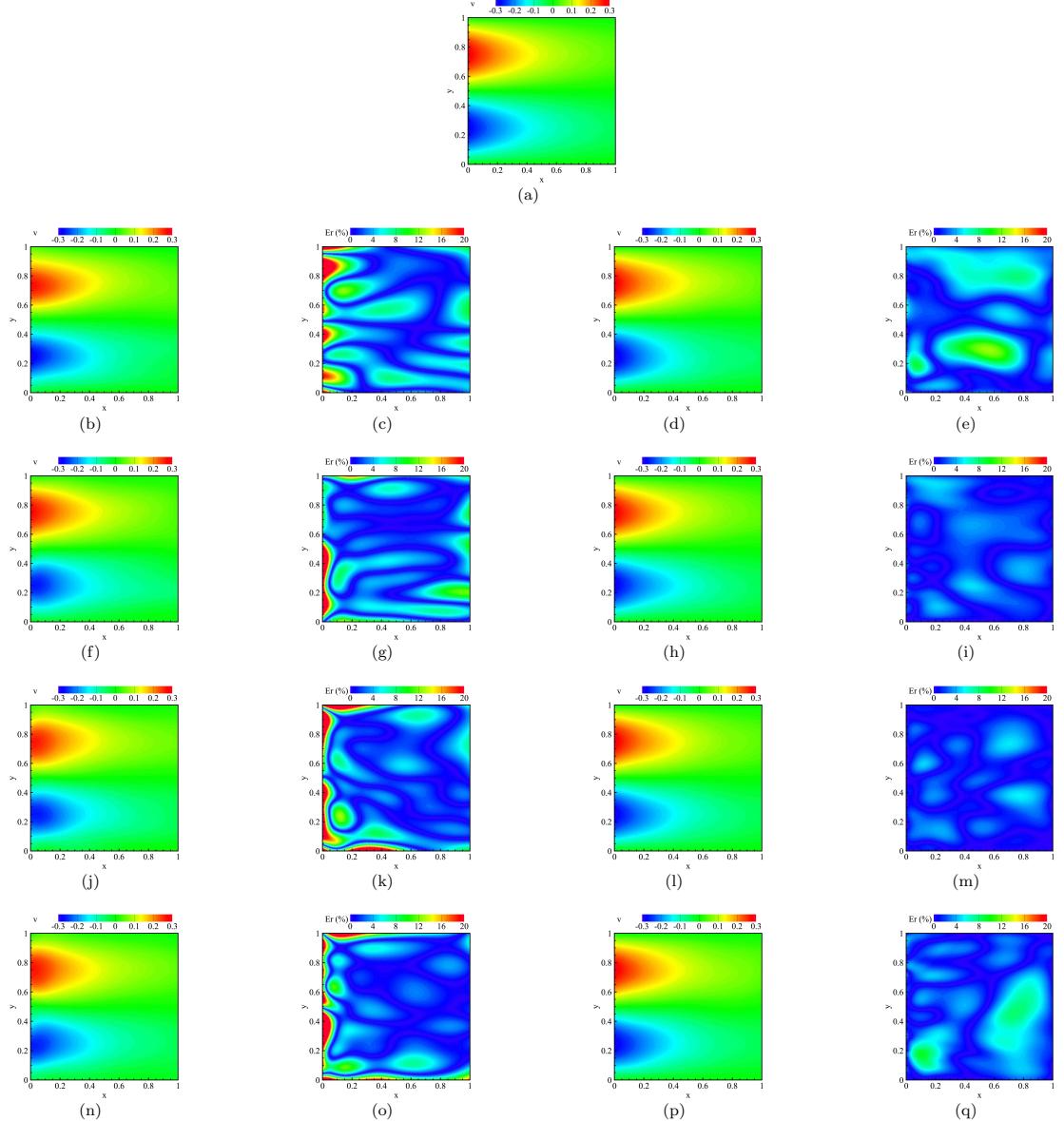


Figure 17: two-dimensional Kovasznay flow velocity of the y direction, v contour. Among them, (a) represents the true value of the Helmholtz equation. The left two columns of the remaining figure represent the predicted figure and error figure of KAN containing 1-4 hidden layers, from top to bottom. The two columns on the right, from top to bottom, represent the predicted figure and error figure of ChebPIKAN with 1-4 hidden layers, respectively.

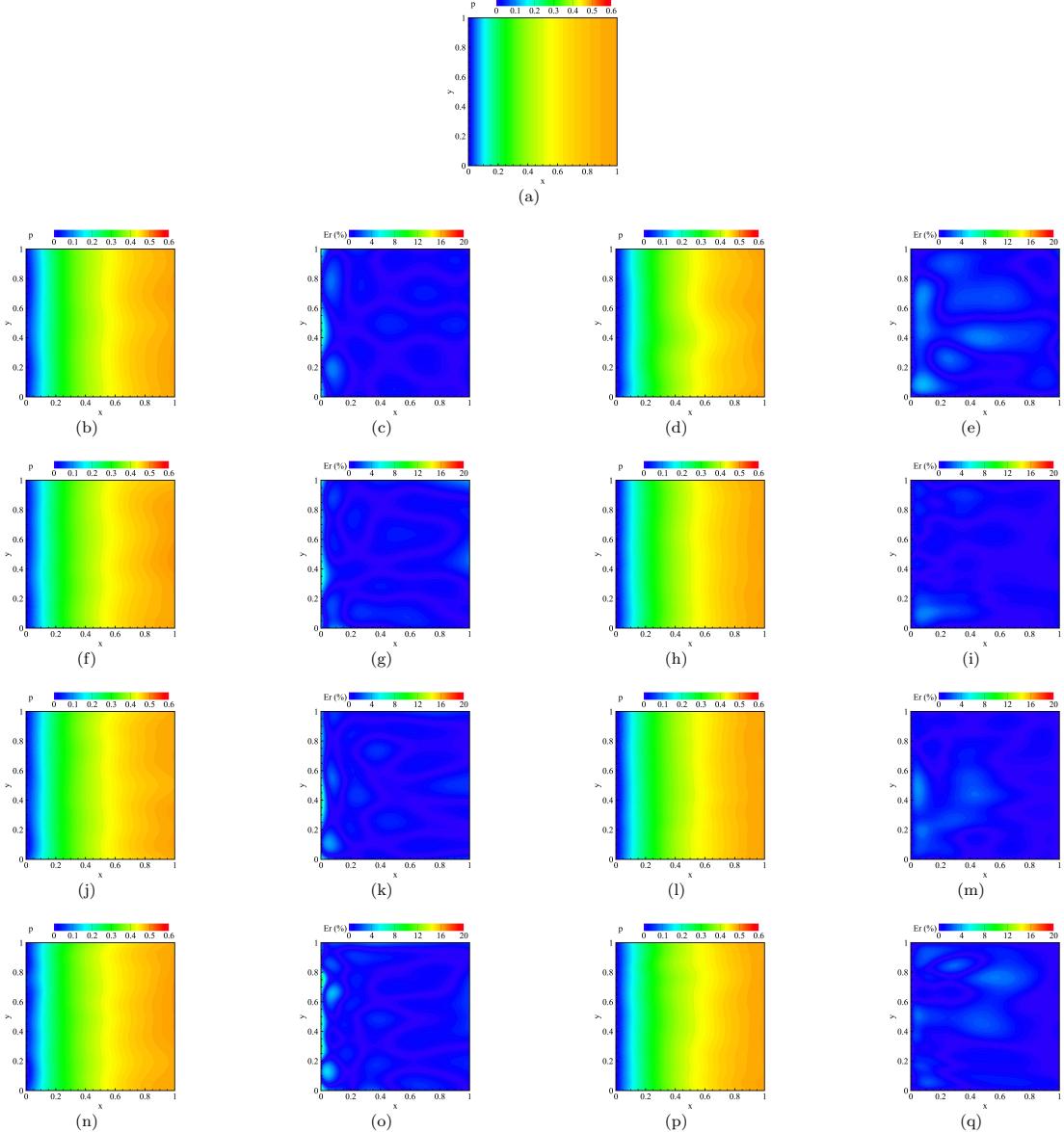


Figure 18: two-dimensional Kovasznay flow pressure, p contour. Among them, (a) represents the true value of the Helmholtz equation. The left two columns of the remaining figure represent the predicted figure and error figure of KAN containing 1-4 hidden layers, from top to bottom. The two columns on the right, from top to bottom, represent the predicted figure and error figure of ChebPIKAN with 1-4 hidden layers, respectively.

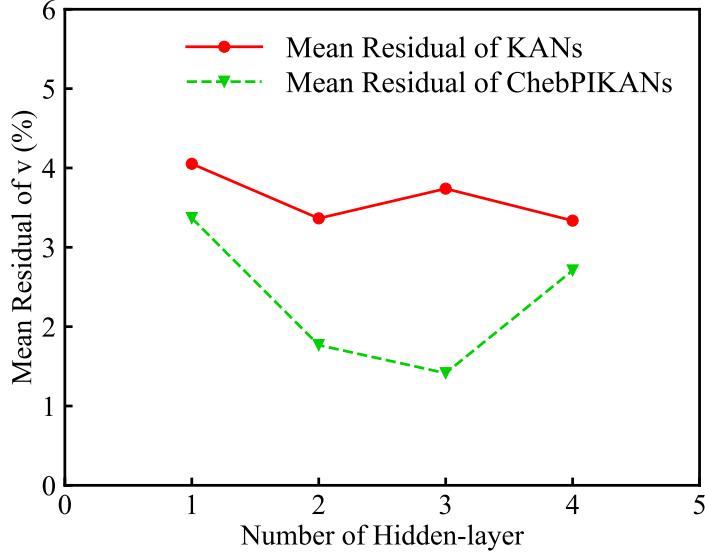


Figure 19: The average residuals of velocity v of KANs and ChebPIKANs with different hidden layers

3.5. Two-dimensional Navier-Stokes Equation

This section focuses on the flow dynamics around a two-dimensional cylinder, analyzed within a horizontal plane where external forces are negligible. The governing equations can be simplified as follows

$$\frac{\partial u}{\partial t} + u \frac{\partial u}{\partial x} + v \frac{\partial u}{\partial y} = - \frac{\partial p}{\partial x} + v \left(\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} \right), \quad (33)$$

$$\frac{\partial v}{\partial t} + u \frac{\partial v}{\partial x} + v \frac{\partial v}{\partial y} = - \frac{\partial p}{\partial y} + v \left(\frac{\partial^2 v}{\partial x^2} + \frac{\partial^2 v}{\partial y^2} \right). \quad (34)$$

The cylindrical turbulence is examined within the spatial domain defined by $x \in [1, 8]$, $y \in [-2, 2]$, and the temporal domain $t \in [0, 7]$. Given the complexity of these time-dependent equations, we generated 2,000 training data points and 5,000 test data points. For the ChebPIKAN model, an additional 8,000 points were sampled to compute the physical loss. The optimizer and optimization strategy are the same as studies for other equations. The network has three inputs, x , y and t , and three outputs, pressure p , velocity u and velocity v . In two-dimensional Navier-Stokes equation, the physical loss is expressed as

$$\begin{aligned} Loss_{PDE} = & \frac{\partial u}{\partial t} + u \frac{\partial u}{\partial x} + v \frac{\partial u}{\partial y} + \frac{\partial p}{\partial x} - v \left(\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} \right) \\ & + \frac{\partial v}{\partial t} + u \frac{\partial v}{\partial x} + v \frac{\partial v}{\partial y} + \frac{\partial p}{\partial y} - v \left(\frac{\partial^2 v}{\partial x^2} + \frac{\partial^2 v}{\partial y^2} \right) \\ & + \nabla \cdot \vec{u}. \end{aligned} \quad (35)$$

Due to the challenges in predicting time-dependent two-dimensional equations and the variations in the specific formulation of the loss function, the predicted loss for KAN and ChebPIKAN is approximately one order of magnitude

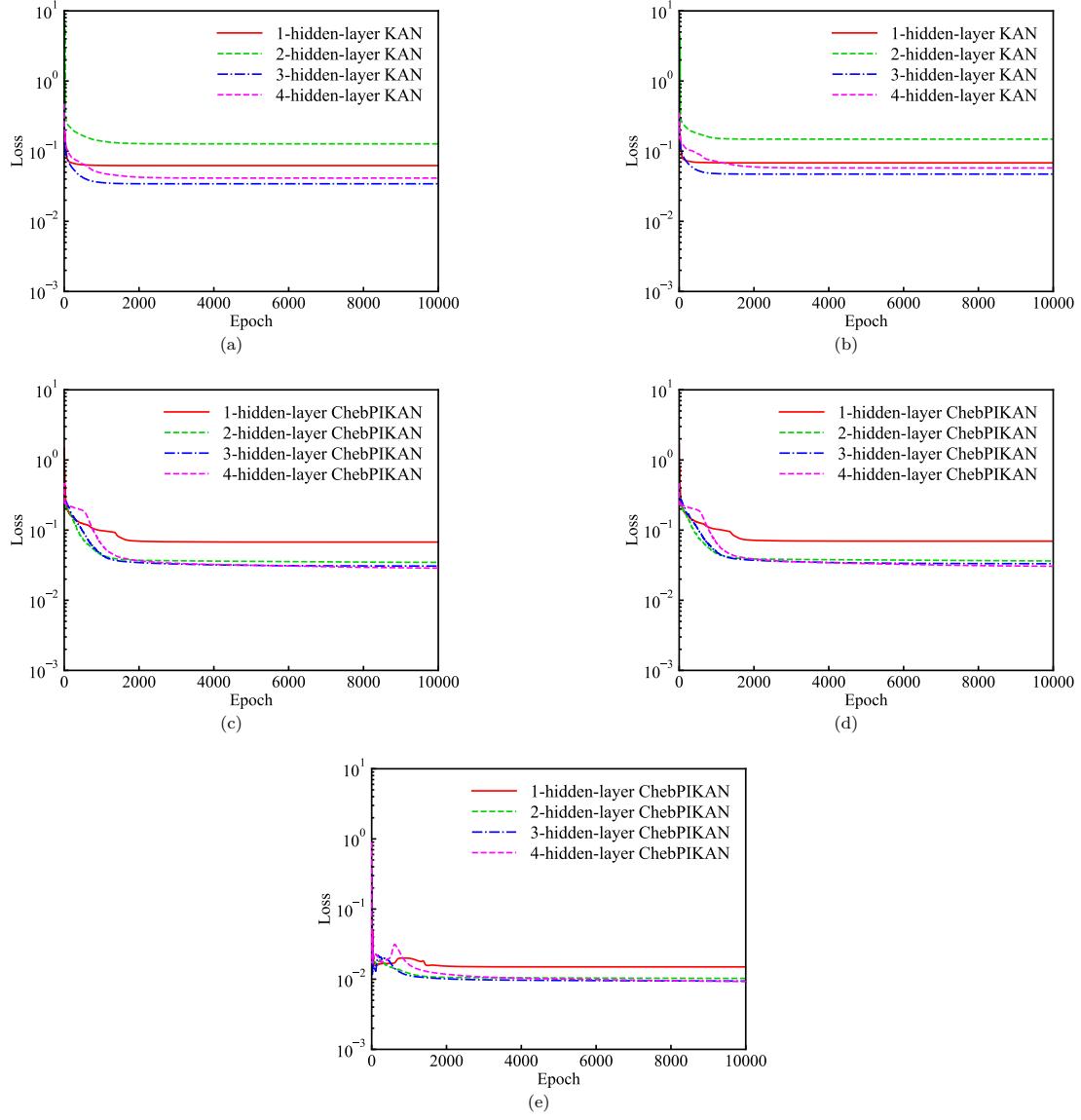


Figure 20: two-dimensional Navier-Stokes equation loss. Among them, (a) and (b) respectively represent the train loss and test loss of KANs containing different hidden layers. (c), (d), and (e) respectively represent the train loss, test loss and PDE loss of ChebPIKANs containing different hidden layers.

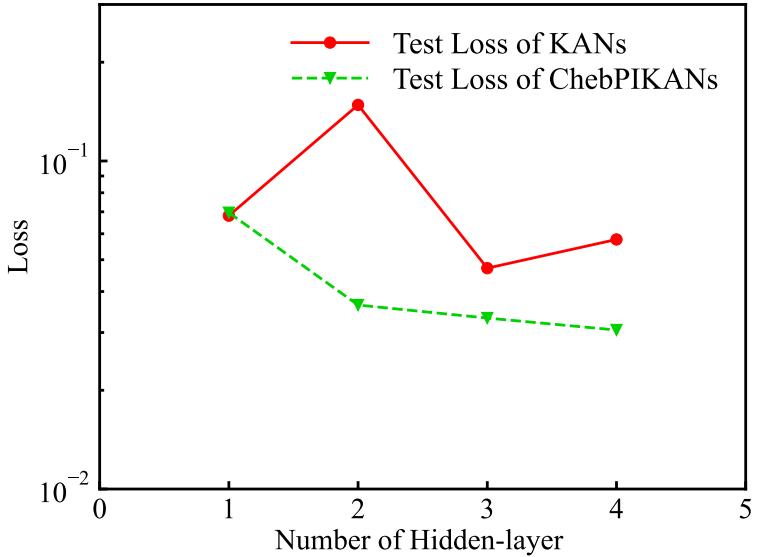


Figure 21: The optimal test loss of KANs and ChebPIKANs with different hidden layers

higher than that of previous models (around 10^{-1}). Notably, the loss for ChebPIKAN is significantly lower than that of KAN, indicating that ChebPIKAN demonstrates superior predictive ability when addressing more complex two-dimensional time-dependent flow problems. The PDE loss is consistently much smaller than the training and test losses across all models, indicating that its weight in the training process is relatively minor. Nevertheless, it contributes to improved network performance, demonstrating that the incorporation of physical information effectively enhances the learning capacity of neural networks. Meanwhile, the KAN network's performance deteriorates due to an excess of learning parameters arising from too many hidden layers. In contrast, the ChebPIKAN architecture, even with more hidden layers, exhibits greater predictive accuracy when tackling complex problems after integrating physical information.

The flow velocity contour of u at the 3rd second is illustrated in Figure 22. For the KAN model, prediction performance is inadequate with up to two hidden layers. In networks with 2, 3, and 4 hidden layers, there are noticeable wavy errors in the strong shear region near the cylindrical wall. This indicates the KAN network's unstable learning ability with low-density data, leading to inaccuracies in areas with sharp velocity gradient changes. In contrast, ChebPIKAN, which incorporates physical information, does not exhibit this issue, its prediction errors near the wall and in other regions are significantly lower than those of KAN. Even with just one hidden layer, ChebPIKAN demonstrates satisfactory predictive performance, markedly differing from KAN's results.

The prediction of the flow velocity v in the y -direction of cylindrical turbulence is relatively small compared to u , making it challenging to minimize errors. Even a slight discrepancy can significantly increase the relative error (due to a small denominator, as shown in eq (16)). Consequently, both networks exhibit considerable errors in the final prediction of v compared to u . However, it is evident that ChebPIKAN, which incorporates physical

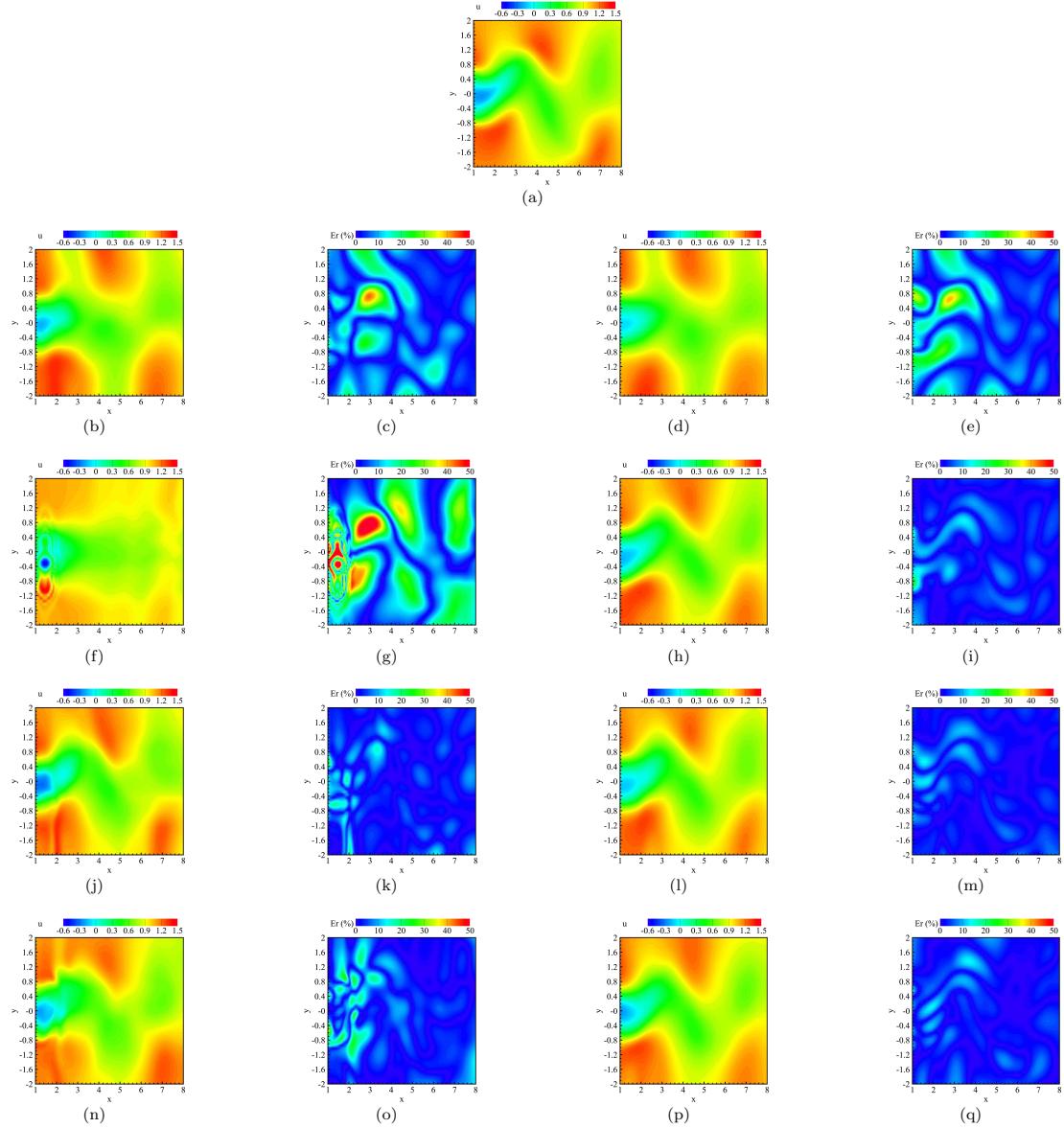


Figure 22: two-dimensional Navier-Stokes velocity of the x direction, u contour. Among them, (a) represents the true value of the NS equation. The left two columns of the remaining figure represent the predicted figure and error figure of KAN containing 1-4 hidden layers, from top to bottom. The two columns on the right, from top to bottom, represent the predicted figure and error figure of ChebPIKAN with 1-4 hidden layers, respectively.

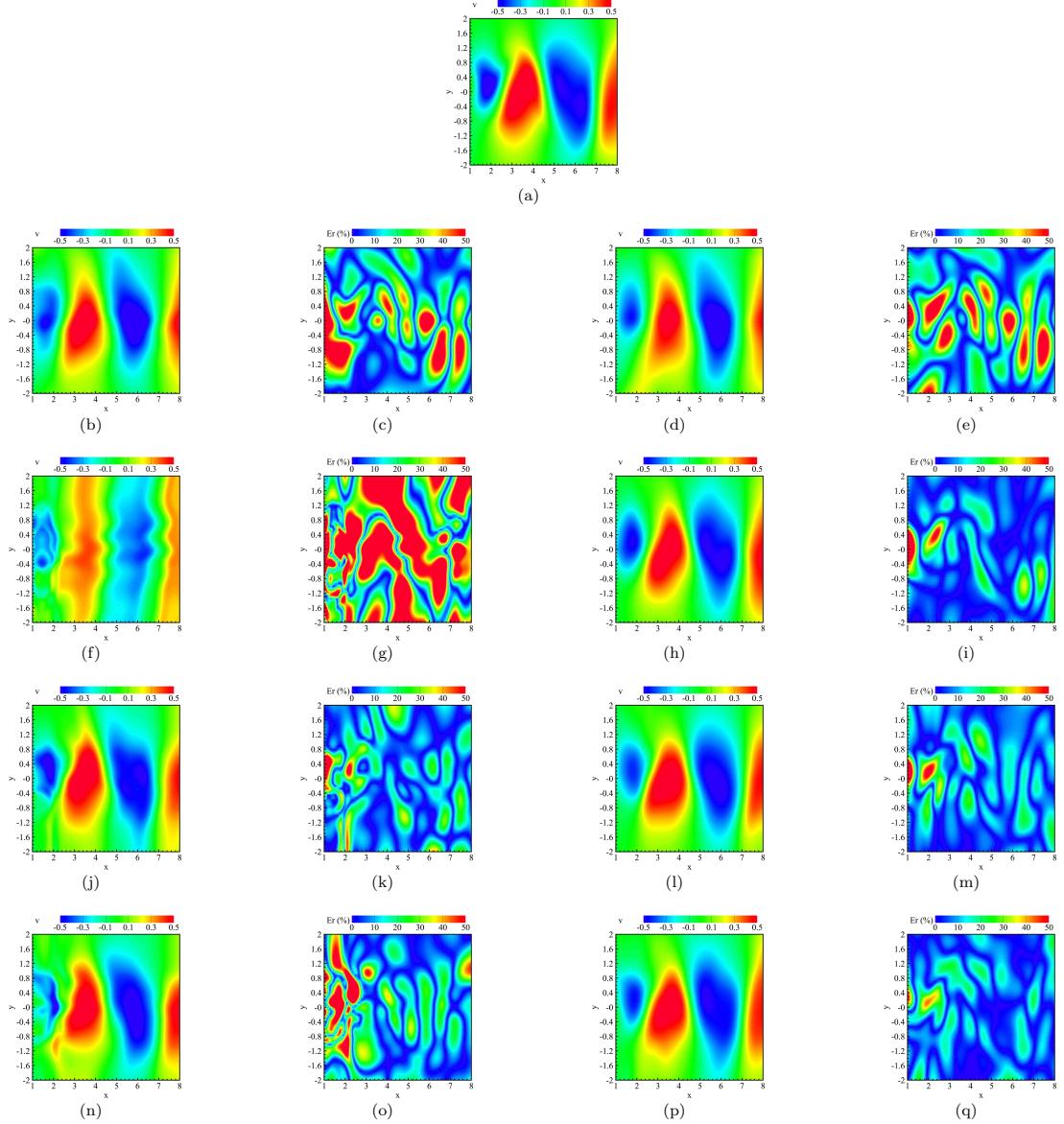


Figure 23: two-dimensional Navier-Stokes velocity of the y direction, v contour. Among them, (a) represents the true value of the NS equation. The left two columns of the remaining figure represent the predicted figure and error figure of KAN containing 1-4 hidden layers, from top to bottom. The two columns on the right, from top to bottom, represent the predicted figure and error figure of ChebPIKAN with 1-4 hidden layers, respectively.

information, achieves lower errors and demonstrates greater stability. Notably, when the network has two hidden layers, it showcases a strong learning ability.

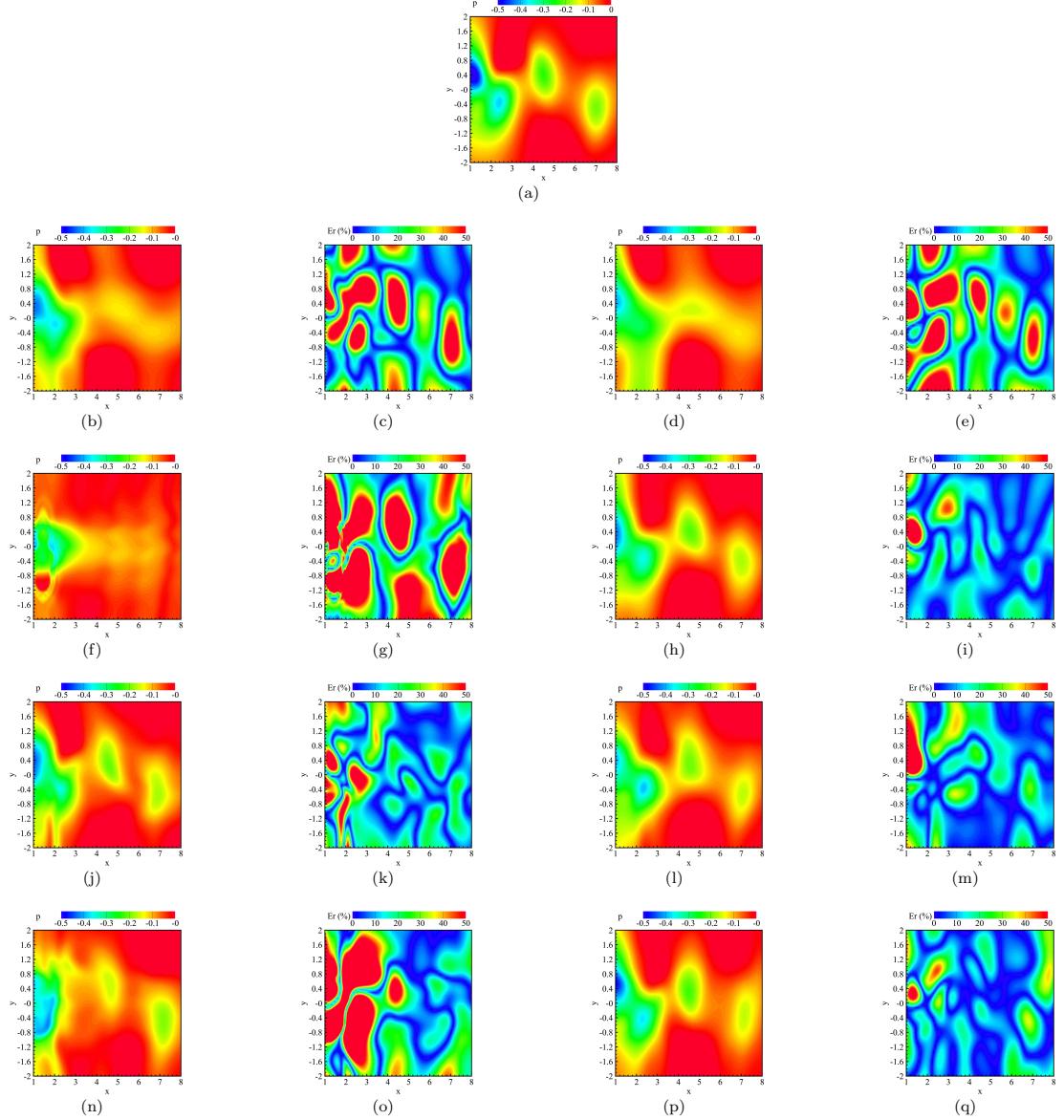


Figure 24: two-dimensional Navier-Stokes pressure, p contour. Among them, (a) represents the true value of the NS equation. The left two columns of the remaining figure represent the predicted figure and error figure of KAN containing 1-4 hidden layers, from top to bottom. The two columns on the right, from top to bottom, represent the predicted figure and error figure of ChebPIKAN with 1-4 hidden layers, respectively.

The error in pressure prediction is primarily concentrated in regions of strong pressure pulsation, presenting a significant challenge for neural networks. This pressure largely arises from the negative pressure generated by the Bernoulli effect in the flow. Additionally, since pressure is much smaller in magnitude compared to u , even a slight prediction error can lead to a substantial increase in relative error. When comparing the prediction results of the two

neural networks, it is clear that ChebPIKAN demonstrates significantly better accuracy than KAN.

Table 6: The average residuals table of KANs and ChebPIKANs with different hidden layers

Number of Hidden-layer	KANs_u	ChebPIKANs_u	KANs_v	ChebPIKANs_v	KANs_p	ChebPIKANs_p
1	7.30%	7.38%	17.88%	17.51%	22.09%	23.52%
2	15.09%	3.34%	38.39%	8.46%	39.45%	10.85%
3	3.38%	2.76%	11.25%	9.68%	15.33%	13.12%
4	4.61%	2.96%	15.39%	7.93%	32.55%	11.13%

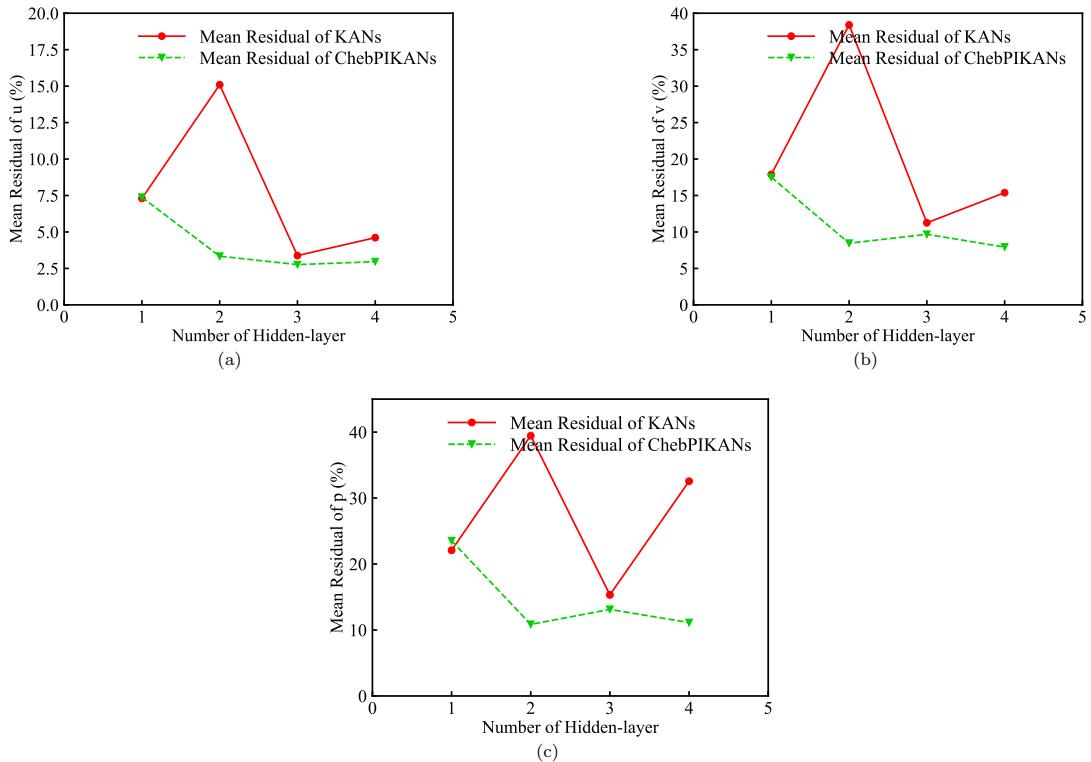


Figure 25: The average residuals of velocity u , v and p an of KANs and ChebPIKANs with different hidden layers

The superiority of ChebPIKAN is evident across multiple levels, particularly in scenarios with a high number of hidden layers, where it effectively manages the mean residuals. This highlights its adaptability and effectiveness in complex models, while the performance of KAN fails to meet expectations.

4. Conclusions and Discussion

In this section, a summary of the proposed ChebPIKAN is provided. The conversion effect similar to ANN to PINN is achieved by adding physical information (PDE loss) to KAN. In this article, the performance improvement

of ChebPIKAN with added physical information relative to KAN is verified by solving partial differential equations, ChebPIKAN has obvious advantages. It is reflected in the higher stability and generalization of the network compared to KAN without adding physical information. It also has higher learning ability. Generally, the prediction accuracy of ChebPIKAN with two hidden layers is higher than that of KANs with 3 or 4 layers, indicating that ChebPIKAN can achieve better results with fewer training parameters. All equation solving experiments use low resolution data with few sampling points. ChebPIKAN has satisfactory prediction results, which proves the potential application of ChebPIKAN. ChebPIKAN has demonstrated excellent solving ability for both one-dimensional and two-dimensional equations. Two-dimensional equations often require more epochs and training time compared. To one-dimensional equations. This can be understood as two-dimensional equations are more complex. Although spline fitting is very time-consuming and computationally intensive. Although spline fitting is very time-consuming and computationally resource intensive, using the KAN architecture requires a smaller model, significantly saving computational resources and reducing computation time.

Because of the localization of KAN fitting, ChebPIKAN has better generalization performance. This advantage is evident not only in the continuous learning performance of the network but also in its resilience during debugging and training across various tasks, effectively mitigating the risk of catastrophic forgetting. Furthermore, when future tasks involve overlapping or similar samples, the KAN structure is poised to offer significant benefits.

Despite these strengths, there are several areas for improvement within ChebPIKAN that warrant further exploration:

Mathematical Considerations: Although we excel in adopting more flexible network shapes for different scenarios, the mathematical visualization of the network architecture has not been fully developed due to algorithmic optimizations. As a result, its mathematical interpretability remains questionable. However, compared to traditional multilayer perceptrons (MLPs), KAN inherently possesses stronger physical significance, and the addition of physical information through PIKAN enhances this further. Thus, ChebPIKAN's physical meaning and interpretability are notably superior to those of conventional neural networks.

Algorithm Enhancements: We have integrated a PDE loss into the KAN framework, forming the basis of ChebPIKAN. This addition, which incorporates physical information, results in three types of loss, PDE loss, test loss and train loss. Properly adjusting the weights of these losses is crucial, as it directly influences the neural network's learning trajectory. In particular, fine-tuning the correction coefficient for the PDE loss is essential, inappropriate values can lead to imbalanced learning, with one loss dominating and others becoming ineffective. Thus, selecting suitable weight coefficients tailored to specific problems is vital for enhancing ChebPIKAN's predictive performance.

Practical Application: ChebPIKAN's automatic pruning function is not yet fully developed. Currently, PIKAN's automatic pruning model cannot be directly applied due to hidden parameters lacking a one-to-one correspondence. Consequently, we manually define network architectures for performance testing. Nonetheless, the shape of the automatically trimmed model can serve as a valuable reference, guiding manual network configurations. Although setting network parameters manually is manageable, the ongoing refinement of the automatic pruning function remains

an important area for future enhancement.

Overall, while ChebPIKAN demonstrates promising capabilities, continuous improvements in mathematical interpretability, algorithmic precision, and practical application will be essential for its evolution.

Acknowledgments

This work was supported by the National Natural Science Foundation of China (NSFC Grants No. 52425111), by the Shandong Province Youth Fund Project (Grants No. ZR2024QA050), by the International Science and Technology Cooperation Project of the Fundamental Research Funds for the Central Universities of Harbin Engineering University Grants (No. 3072024GH2602), and by the Young Scientist Cultivation Fund of Qingdao Innovation and Development Base of Harbin Engineering University.

References

- [1] Parviz Moin and Krishnan Mahesh. Direct numerical simulation: a tool in turbulence research. *Annual review of fluid mechanics*, 30(1):539–578, 1998.
- [2] Sergio Pirozzoli and Paolo Orlandi. Natural grid stretching for dns of wall-bounded flows. *Journal of Computational Physics*, 439:110408, 2021.
- [3] E.M.J. Komen, L.H. Camilo, A. Shams, B.J. Geurts, and B. Koren. A quantification method for numerical dissipation in quasi-dns and under-resolved dns, and effects of numerical dissipation in quasi-dns and under-resolved dns of turbulent channel flows. *Journal of Computational Physics*, 345:565–595, 2017.
- [4] Mohammad Shoeybi, Magnus Svärd, Frank E. Ham, and Parviz Moin. An adaptive implicit-explicit scheme for the dns and les of compressible flows on unstructured grids. *Journal of Computational Physics*, 229(17):5944–5965, 2010.
- [5] Kiran Bhaganagar, Dietmar Rempfer, and John Lumley. Direct numerical simulation of spatial transition to turbulence using fourth-order vertical velocity second-order vertical vorticity formulation. *Journal of Computational Physics*, 180(1):200–228, 2002.
- [6] Yu Zhang, Richard P. Dwight, Martin Schmelzer, Javier F. Gómez, Zhonghua Han, and Stefan Hickel. Customized data-driven rans closures for bi-fidelity les-rans optimization. *Journal of Computational Physics*, 432:110153, 2021.
- [7] F. Clerici, P.R. Spalart, and F. Alauzet. Turbulence driven goal-oriented anisotropic mesh adaptation for rans simulations in aerodynamics. *Journal of Computational Physics*, 514:113191, 2024.
- [8] Heng Xiao and Patrick Jenny. A consistent dual-mesh framework for hybrid les/rans modeling. *Journal of Computational Physics*, 231(4):1848–1865, 2012.
- [9] Xiaowei Jin, Shengze Cai, Hui Li, and George Em Karniadakis. Nsfnets (navier-stokes flow nets): Physics-informed neural networks for the incompressible navier-stokes equations. *Journal of Computational Physics*, 426:109951, 2021.
- [10] Prem A Srinivasan, L Guastoni, Hossein Azizpour, PHILIPP Schlatter, and Ricardo Vinuesa. Predictions of turbulent shear flows using deep neural networks. *Physical Review Fluids*, 4(5):054603, 2019.
- [11] Saakaar Bhatnagar, Yaser Afshar, Shaowu Pan, Karthik Duraisamy, and Shailendra Kaushik. Prediction of aerodynamic flow fields using convolutional neural networks. *Computational Mechanics*, 64:525–545, 2019.
- [12] Javier E Santos, Duo Xu, Honggeun Jo, Christopher J Landry, Maša Prodanović, and Michael J Pyrcz. Poreflow-net: A 3d convolutional neural network to predict fluid flow through porous media. *Advances in Water Resources*, 138:103539, 2020.
- [13] Junhyuk Kim and Changhoon Lee. Prediction of turbulent heat transfer using convolutional neural networks. *Journal of Fluid Mechanics*, 882:A18, 2020.

- [14] N Benjamin Erichson, Lionel Mathelin, Zhewei Yao, Steven L Brunton, Michael W Mahoney, and J Nathan Kutz. Shallow neural networks for fluid flow reconstruction with limited sensors. *Proceedings of the Royal Society A*, 476(2238):20200097, 2020.
- [15] Ali Kashefi, Davis Rempe, and Leonidas J Guibas. A point-cloud deep learning framework for prediction of fluid flow fields on irregular geometries. *Physics of Fluids*, 33(2), 2021.
- [16] Maziar Raissi, Paris Perdikaris, and George E Karniadakis. Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations. *Journal of Computational physics*, 378:686–707, 2019.
- [17] Laith Alzubaidi, Jinshuai Bai, Aiman Al-Sabaawi, Jose Santamar’ia, Ahmed Shihab Albahri, Bashar Sami Nayyef Al-dabbagh, Mohammed A Fadhel, Mohamed Manoufali, Jinglan Zhang, Ali H Al-Timemy, et al. A survey on deep learning tools dealing with data scarcity: definitions, challenges, solutions, tips, and applications. *Journal of Big Data*, 10(1):46, 2023.
- [18] Amirhossein Arzani, Jian-Xun Wang, and Roshan M D’Souza. Uncovering near-wall blood flow from sparse data with physics-informed neural networks. *Physics of Fluids*, 33(7), 2021.
- [19] Zhao Chen, Yang Liu, and Hao Sun. Physics-informed learning of governing equations from scarce data. *Nature communications*, 12(1):6136, 2021.
- [20] Amuthan A Ramabathiran and Prabhu Ramachandran. Spinn: sparse, physics-based, and partially interpretable neural networks for pdes. *Journal of Computational Physics*, 445:110600, 2021.
- [21] S Hanrahan, M Kozul, and RD Sandberg. Studying turbulent flows with physics-informed neural networks and sparse data. *International Journal of Heat and Fluid Flow*, 104:109232, 2023.
- [22] Shengfeng Xu, Zhenxu Sun, Renfang Huang, Dilong Guo, Guowei Yang, and Shengjun Ju. A practical approach to flow field reconstruction with sparse or incomplete data through physics informed neural network. *Acta Mechanica Sinica*, 39(3):322302, 2023.
- [23] Majid Rasht-Behesht, Christian Huber, Khemraj Shukla, and George Em Karniadakis. Physics-informed neural networks (pinns) for wave propagation and full waveform inversions. *Journal of Geophysical Research: Solid Earth*, 127(5):e2021JB023120, 2022.
- [24] Hamidreza Eivazi, Mojtaba Tahani, Philipp Schlatter, and Ricardo Vinuesa. Physics-informed neural networks for solving reynolds-averaged navier–stokes equations. *Physics of Fluids*, 34(7), 2022.
- [25] Hongping Wang, Yi Liu, and Shizhao Wang. Dense velocity reconstruction from particle image velocimetry/particle tracking velocimetry using a physics-informed neural network. *Physics of fluids*, 34(1), 2022.
- [26] Lei Yuan, Yi-Qing Ni, Xiang-Yun Deng, and Shuo Hao. A-pinn: Auxiliary physics informed neural networks for forward and inverse problems of nonlinear integro-differential equations. *Journal of Computational Physics*, 462:111260, 2022.
- [27] Shahed Rezaei, Ali Harandi, Ahmad Moeineddin, Bai-Xiang Xu, and Stefanie Reese. A mixed formulation for physics-informed neural networks as a potential solver for engineering problems in heterogeneous domains: Comparison with finite element method. *Computer Methods in Applied Mechanics and Engineering*, 401:115616, 2022.
- [28] Hyogu Jeong, Jinshuai Bai, Chanaka Prabuddha Batuwatta-Gamage, Charith Rathnayaka, Ying Zhou, and YuanTong Gu. A physics-informed neural network-based topology optimization (pinnto) framework for structural optimization. *Engineering Structures*, 278:115484, 2023.
- [29] Yanbing Liu, Liping Chen, Jianwan Ding, and Yu Chen. An adaptive sampling method based on expected improvement function and residual gradient in pinns. *IEEE Access*, 2024.
- [30] Jinshuai Bai, Timon Rabczuk, Ashish Gupta, Laith Alzubaidi, and Yuantong Gu. A physics-informed neural network technique based on a modified loss function for computational 2d and 3d solid mechanics. *Computational Mechanics*, 71(3):543–562, 2023.
- [31] Salah A Faroughi, Nikhil M Pawar, Celio Fernandes, Maziar Raissi, Subashish Das, Nima K Kalantari, and Seyed Kouros Mahjour. Physics-guided, physics-informed, and physics-encoded neural networks and operators in scientific computing: Fluid and solid mechanics. *Journal of Computing and Information Science in Engineering*, 24(4):040802, 2024.
- [32] Ziming Liu, Yixuan Wang, Sachin Vaidya, Fabian Ruehle, James Halverson, Marin Soljavić, Thomas Y Hou, and Max Tegmark. Kan: Kolmogorov-arnold networks. *arXiv preprint arXiv:2404.19756*, 2024.
- [33] Fabr’icio Olivetti de Francca. A greedy search tree heuristic for symbolic regression. *Information Sciences*, 442:18–32, 2018.
- [34] Jared Willard, Xiaowei Jia, Shaoming Xu, Michael Steinbach, and Vipin Kumar. Integrating scientific knowledge with machine learning for

- engineering and environmental systems. *ACM Computing Surveys*, 55(4):1–37, 2022.
- [35] Jichao Li, Xaosong Du, and Joaquim RRA Martins. Machine learning in aerodynamic shape optimization. *Progress in Aerospace Sciences*, 134:100849, 2022.