Nuclear Science and Techniques (2026) 37:28
https://doi.org/10.1007/s41365-025-01839-5

Residual resampling-based physics-informed neural network for neutron diffusion equations

Heng Zhang[1] · Yun-Ling He[1] · Dong Liu[2,3] · Qin Hang[1] · He-Min Yao[1] · Di Xiang[2,3]

Received: 14 January 2024 / Revised: 20 August 2024 / Accepted: 1 September 2024 / Published online: 4 January 2026

Abstract

The neutron diffusion equation plays a pivotal role in nuclear reactor analysis. Nevertheless, employing the physics-informed neural network (PINN) method for its solution entails certain limitations. Conventional PINN approaches generally utilize a fully connected network (FCN) architecture that is susceptible to overfitting, training instability, and gradient vanishing as the network depth increases. These challenges result in accuracy bottlenecks in the solution. In response to these issues, the residual-based resample physics-informed neural network (R^2-PINN) is proposed. It is an improved PINN architecture that replaces the FCN with a convolutional neural network with a shortcut (S-CNN). It incorporates skip connections to facilitate gradient propagation between network layers. Additionally, the incorporation of the residual adaptive resampling (RAR) mechanism dynamically increases the number of sampling points. This, in turn, enhances the spatial representation capabilities and overall predictive accuracy of the model. The experimental results illustrate that our approach significantly improves the convergence capability of the model and achieves high-precision predictions of the physical fields. Compared with conventional FCN-based PINN methods, R^2-PINN effectively overcomes the limitations inherent in current methods. Thus, it provides more accurate and robust solutions for neutron diffusion equations.

Keywords Neutron diffusion equation · Physics-informed neural network · CNN with shortcut · Residual adaptive resampling

1 Introduction

Nuclear reactor core analysis is crucial to ensure the safe operation of nuclear reactors. The neutron diffusion equation describes the neutron movement within a medium. It is fundamental to this analysis [1]. Numerical methods have been widely applied to numerous physical scenarios. Methods, such as finite difference [2] and finite element [3], have been continuously developed and improved. Many related works are available in the reactor domain. For example, Hamada [4] proposed higher-order compact finite-difference schemes to solve neutron diffusion equations. Yuk [5] utilized the finite-element method to solve a time-dependent neutron diffusion equation. Li [6] designed an algorithm based on the finite volume method to solve multigroup neutron diffusion equations. For the same problem, Matheus Gularte Tavares [7] and K. Zhuang [8] used the source iterative and variational nodal methods, respectively. Additionally, many researchers have successfully employed CFD

This work was supported by the Science and Technology on Reactor System Design Technology Laboratory (No. LRSDT12023108), and supported in part by the Chongqing Postdoctoral Science Foundation (No. cstc2021jcyj-bsh0252), the National Natural Science Foundation of China (No. 12005030), Sichuan Province to unveil the list of marshal industry common technology research projects (No. 23jBGOV0001), and Special Program for Stabilizing Support to Basic Research of National Basic Research Institutes (No. WDZC-2023-05-03-05).

✉ Dong Liu
493159139@qq.com

[1] College of Computer Science and Technology, Chongqing University of Posts and Telecommunications, Chongqing 400065, China

[2] Science and Technology on Reactor System Design Technology Laboratory, Nuclear Power Institute of China, Chengdu 610213, China

[3] CNNC Engineering Research Center of Nuclear Energy Software and Digital Reactor, Chengdu 610213, China

software such as COMSOL [9], ANSYS FLUENT [10], and OpenFOAM [11] to address neutron diffusion problems.

However, these methods require discretization of the solution domain. This can be computationally complex and time-consuming when high-precision physics reconstruction is required [12], also, considering the complex environment in nuclear reactors, which exist in multi-physics fields such as neutron transport and heat transfer. Numerical methods need to simplify and approximate the model to solve the equations. This introduces a certain number of solution errors. Meanwhile, owing to the development of neural networks (NNs), the interest in machine learning-based approaches has been increasing [13]. Training an NN enables the prediction of the physical field to be faster than that with conventional numerical methods. Prior engineering knowledge should then be incorporated into the network to make the NN predictions more consistent with physical laws. PINNs was introduced by Raissi [14]. These incorporate partial differential equations (PDEs) as constraints during network training by increasing the penalties for violating PDE data points. Thus, it yields more precise and physically consistent solutions. To date, PINNs have been applied successfully to various scientific computation problems in engineering fields such as fluids [15, 16], heat transfer [17, 18], flows [19, 20], and solid mechanics [21, 22], and have yielded significant results [23]. The differences between PINNs and conventional numerical methods are listed in Table 1. The Monte Carlo method [24, 25] and CFD [26] can also be used to solve reactor problems in nuclear reactor core analysis. In recent years, deep neural networks (DNNs) have been used increasingly in reactor cores [27, 28]. Dong [29, 30] applied PINNs to solve multiple neutron diffusion benchmark equations and demonstrated highly accurate predictive results. Utilizing FCN to capture the neutron distribution, they achieved a neutron flux distribution solution with an accuracy of 10^{-7} and successfully applied it to the search for critical parameters.

Moreover, FCNs are vulnerable to gradient vanishing and encounter nonconvergence during training. When the gradient vanishes, the NN parameters are not updated, and it is challenging for the network to learn new knowledge [31]. This may cause the network to have difficulty converging to an optimal solution, miss a large amount of information, and affect the expressive capability of the model. In terms of solving the neutron diffusion equation, the error in the predicted result renders the critical assessment inaccurate. Furthermore, it was observed that regions with large gradients exhibited insufficient training under uniform sampling, particularly when the number of sampling points was limited. This resulted in significant errors in regions with large gradients, thereby limiting further improvement in network accuracy. When solving the neutron diffusion equation, the limited accuracy may result in inefficient parameter searches that significantly increase the search time.

Recent studies proposed adaptive sampling based on gradient information [32] and assigned adaptive weights to sample points in loss calculations [33, 34]. These enhancements improved the prediction performance of PINNs in regions with significant gradients and limited sampling points. Furthermore, the problem of gradient disappearance in FCNs remains unresolved. Researchers have evaluated the replacement of FCNs with other neural network techniques [35] such as CNNs [36] and recurrent neural networks (RNNs) [37] to achieve better performance and more precise results.

Table 1 Comparison between PINN and conventional numerical methods

Feature	PINN	Traditional numerical methods
Calculation method	Combines neural networks with physics equations	Uses analytical solutions or numerical methods
Accuracy dependency	Data noise level, physics model accuracy	Grid resolution, numerical methods
Computational efficiency	Low computational demand, training process requires hyperparameter optimization	High computational demand, requires significant computational resources
Reliability	Robust to data noise and uncertainty	Sensitive to initial conditions, boundary conditions, and numerical parameters
Model complexity	Can handle complex nonlinear problems	Typically suitable for specific types and relatively simple problems
Training speed	Relatively high training speed, however, adjusting model hyperparameters is time-consuming	High computational complexity, longer runtime
Grid dependency	Grid-independent	Results are influenced by grid accuracy and partitioning
Parameter tuning	Requires tuning of network architecture and hyperparameters	Requires tuning of grid partitioning and solver parameters
Parallel computing	Efficient parallel computing	Has challenges in parallel computing
Applicability	Suitable for complex nonlinear problems and data scarcity	Suitable for known equations and stable boundary conditions

To address this, this study proposed a novel framework called the R^2-PINN. It combines the S-CNN architecture with the RAR method to solve neutron diffusion equations [38, 39]. The proposed model effectively alleviates the gradient vanishing problem by adding the gradient backpropagation path [40]. Moreover, the network can balance the loss between regions and achieve a higher accuracy by using a resample mechanism. The R^2-PINN was evaluated against the FCN to solve the neutron diffusion equation benchmark problems (introduced in Sect. 2). Additionally, it was demonstrated that our method effectively suppressed the loss function oscillation and achieved high-precision field prediction. In addition, our method significantly reduced the time required for an eigenvalue search. This enabled us to obtain an accuracy of 10^{-5} within 10 min and an accuracy of 10^{-4} in 250 s.

The remainder of this paper is organized as follows: Sect. 2 introduces the benchmark problems used in Sect. 4. Section 3 introduces the basic architectures including the structure of the S-CNN and RAR resample methods and the overall R^2-PINN architecture. Section 4 presents the multiple experiments conducted to optimize the hyperparameters and verify the superiority of the proposed model. In particular, different search algorithms are compared for the parameter search to reduce the search time. It also presents generalizability validation experiments using the same model for multiple benchmark problems. Section 5 analyzes and discusses the experimental results. Finally, Sect. 6 concludes the study and discusses future research directions.

2 Problem Setup

2.1 One-dimensional reactor diffusion equation for a single energy group

In this section, a single-group k-eigenvalue problem is introduced for the criticality calculations. The largest value of k (known as the effective neutron multiplication factor or k_{eff}) should be determined. The single-group neutron diffusion model is given by Eq. (1):

$$\begin{aligned}\frac{1}{v}\frac{\partial\phi(r,E,t)}{\partial t} &= \nabla\cdot D\nabla\phi(r,E,t)-\Sigma_{\text{t}}(r,E)\phi(r,E,t)+\\ \chi(E)\int_0^{\infty}\nu(E')\Sigma_{\text{f}}(r,E')\phi(r,E',t)\mathrm{d}E'&+\\ S(r,E,t)+\int_0^{\infty}\Sigma_{\text{s}}(r,E'\to E)\phi(r,E',t)\mathrm{d}E',&\end{aligned} \tag{1}$$

where $\phi(r,E,t)$ denotes the neutron flux of the energy group E in the r-coordinate at the instant t, v represents the neutron velocity, D denotes the diffusion coefficient, ν represents the neutrons per fission numbers, $\chi(E)$ represents the prompt neutron spectra, Σ_{t} represents total macroscopic cross section, Σ_{s} represents the macroscopic scattering cross section from group E' to group E, Σ_{f} represents the fission macroscopic cross section, and $S(r, E, t)$ represents the neutron source.

In the absence of $S(r, E, t)$, the initial neutron flux density is symmetric along the x-axis. Equation (1) can be simplified into Eq. (2)[1]:

$$\frac{1}{Dv}\frac{\partial\phi(r,t)}{\partial t} = \nabla^2\phi(r,t)+\frac{k_\infty-1}{L^2}\phi(r,t). \tag{2}$$

Here, k_∞ represents the infinite multiplication factor, and L denotes the diffusion length. Consider a uniform bare reactor [41] that is shaped as an infinite-plate bare reactor with dimensions of infinite length and width, and a thickness (including the extrapolation distance) of a. This is illustrated in Fig. 1. The analytical solution for the neutron flux can be obtained using the separation-of-variables method, as shown in Eq. (3):

$$\begin{aligned}\phi(x,t) = \sum_{n=1}^{\infty}\left[\frac{2}{a}\int_{-\frac{a}{2}}^{\frac{a}{2}}\phi_0(x')\cos\frac{(2n-1)\pi}{a}x'\mathrm{d}x'\right]\cdot\\ \cos\frac{(2n-1)\pi}{a}x\mathrm{e}^{(k_n-1)t/l_n}\end{aligned} \tag{3}$$

The critical conditions for a bare reactor in the single-group approximation are given by Eq. (4).

$$k_{\text{eff}} = \frac{k_\infty}{1+L^2B^2} = 1 \tag{4}$$

where k_{eff} is the effective neutron multiplication factor. When the system is in a critical state, the neutron flux-density distribution satisfies the wave equation according to the

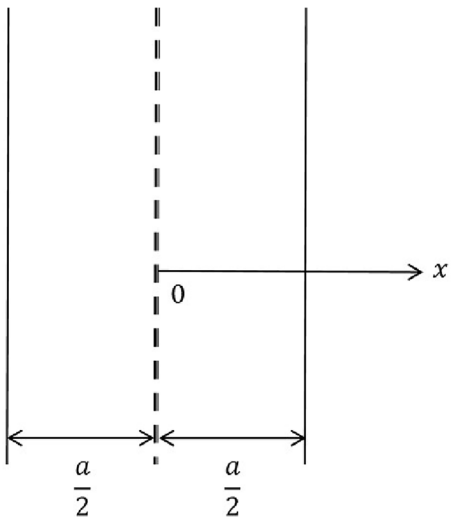


Fig. 1 Infinite Plate Reactor

fundamental eigenfunction corresponding to the minimum eigenvalue B_g^2. This is given by Eq. (5):

$$\nabla^2\phi(r) + B_g^2\phi(r) = 0. \tag{5}$$

If the composition of the system material (i.e., k_∞ and L^2) is given, a unique critical size (denoted by a_0) results in $k_{\text{eff}} = 1$. The critical size a_0 corresponds to the critical state of the reactor. For reactor sizes larger than a_0, $k_{\text{eff}} > 1$, which indicates that the reactor is in a supercritical state. Conversely, for reactor sizes smaller than a_0, $k_{\text{eff}} < 1$, which indicates that the reactor is in a subcritical state [42]. However, if the reactor size is given, it is feasible to determine a fuel enrichment (material composition) that satisfies Eq. (4) and ensures that the reactor attains criticality. When the system is in the critical state, the neutron flux-density distribution within the reactor can be described as follows:

$$\phi(x) = A\cos\frac{\pi}{a}x. \tag{6}$$

2.2 Two-dimensional reactor diffusion equation for a single energy group

Based on Sect. 2.1, consider a two-dimensional neutron diffusion equation formulated as follows:

$$\frac{1}{Dv}\frac{\partial\phi(x,y,t)}{\partial t} = \nabla^2\phi(x,y,t) + \frac{k_\infty - 1}{L^2}\phi(x,y,t). \tag{7}$$

By discretizing Eq. (7) and approximating the Laplace operator and partial derivatives, the equation can be converted into the following form:

$$\begin{aligned}\frac{1}{Dv}\frac{\phi_{i,j}^{n+1} - \phi_{i,j}^{n}}{\Delta t} &= \frac{\phi_{i+1,j}^{n} - 2\phi_{i,j}^{n} + \phi_{i-1,j}^{n}}{\Delta x^2} + \\ &\frac{\phi_{i,j+1}^{n} - 2\phi_{i,j}^{n} + \phi_{i,j-1}^{n}}{\Delta y^2} + \\ &\frac{k_\infty - 1}{L^2}\phi_{i,j}^{n},\end{aligned} \tag{8}$$

where $\phi_{i,j}^{n}$ denotes the value of the neutron flux at the spatial grid point (i, j) at time n. Δt is the time step. Δx and Δy are the spatial steps in the x- and y-directions, respectively. D is the diffusion coefficient, and v is the neutron velocity. k_∞ denotes the infinite multiplication factor, and L is the diffusion length.

According to Eq. (8), the spatial domain meshes use the finite-difference method to solve for the entire domain flux magnitude, and the numerically solved data are used as a test set to verify the model accuracy.

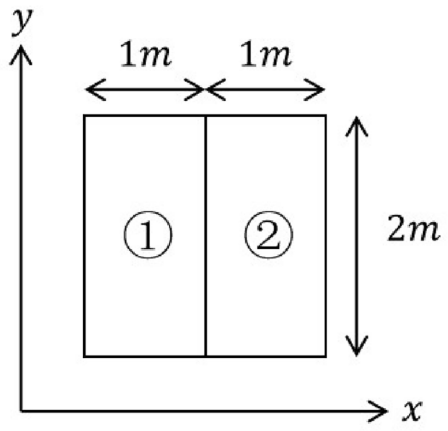


Fig. 2 Material distribution [30]

Table 2 Calculate area material properties

	Material 1		Material 2	
Energy Group	1	2	1	2
D_g(cm)	1.268	0.1902	1.255	0.211
Σ_a (cm^{-1})	0.007181	0.07047	0.008252	0.1003
$\nu\Sigma_f$ (cm^{-1})	0.004609	0.08675	0.004602	0.1091
$\Sigma_{s1\to2}$ (cm^{-1})	0.02767	–	0.02533	–

2.3 Two-dimensional rectangular geometry multigroup multi-material diffusion problem

In a nuclear reactor, the neutron transportation can be described by the multigroup diffusion theory. In this case, the fast- and hot-group neutron fluxes satisfy the following diffusion equations:

$$-D_1\nabla^2\phi_1(r) + \Sigma_{t1}\phi_1(r) = \left[\Sigma_{s1\to1}\phi_1(r) + \Sigma_{s2\to1}\phi_2(r)\right] + \frac{\chi_1}{k_{\text{eff}}}\left[\nu\Sigma_{f1}\phi_1(r) + \nu\Sigma_{f2}\phi_2(r)\right], \tag{9}$$

$$-D_2\nabla^2\phi_2(r) + \Sigma_{t2}\phi_2(r) = \left[\Sigma_{s1\to2}\phi_1(r) + \Sigma_{s2\to2}\phi_2(r)\right] + \frac{\chi_2}{k_{\text{eff}}}\left[\nu\Sigma_{f1}\phi_1(r) + \nu\Sigma_{f2}\phi_2(r)\right], \tag{10}$$

where D_1 and D_2 are the diffusion coefficients of fast- and hot-group neutrons, respectively. ϕ_1 and ϕ_2 are the fast- and hot-group neutron flux densities, respectively. $\nu\Sigma_{f1}$ and $\nu\Sigma_{f2}$ are the neutron production cross sections of the fast- and hot-group neutrons, respectively. $\Sigma_{s1\to2}$ is the fast group to hot-group fission source term.

For pressurized water reactors, the example is divided into two different material regions, which is shown in Fig. 2. The material parameters for each region are listed in Table 2.

Meanwhile, considering that the boundary energy between the fast- and hot-group is low enough, under such circumstances, no hot neutrons are directly produced by nuclear fission. As a result, $\chi_1 = 1$, $\chi_2 = 0$, and $\Sigma_{s2\to1} = 0$. Equations (9) and (10) can be simplified to Eqs. (11) and (12).

$$-D_1\nabla^2\phi_1(r) + \Sigma_{r1}\phi_1(r) = \frac{1}{k_{eff}}\left[\nu\Sigma_{f1}\phi_1(r) + \nu\Sigma_{f2}\phi_2(r)\right] \quad (11)$$

$$-D_2\nabla^2\phi_2(r) + \Sigma_{a2}\phi_2(r) = \Sigma_{s1\to2}\phi_1(r) \quad (12)$$

Based on the multigroup diffusion theory, the distribution of the neutron flux in the core is obtained by iteratively solving the discretized diffusion equation. The obtained dataset was used as a test set in the experiment to evaluate the model prediction accuracy.

2.4 2D-IAEA benchmark problem

The 2D-IAEA PWR benchmark problem is a two-dimensional static problem with two neutron groups but without delayed neutron precursors [43]. It is modeled by the following two-dimensional two-group diffusion equations:

$$\begin{cases} -D_1\nabla^2\phi_1 + (\Sigma_{a1} + \Sigma_{s1\to2})\phi_1 = \lambda\chi_1(\nu\Sigma_{f1}\phi_1 + \nu\Sigma_{f2}\phi_2) \\ -D_2\nabla^2\phi_2 + \Sigma_{a2}\phi_2 - \Sigma_{s1\to2}\phi_1 = \lambda\chi_2(\nu\Sigma_{f1}\phi_1 + \nu\Sigma_{f2}\phi_2) \end{cases} \quad (13)$$

The reactor has a two-zone core containing 177 fuel assemblies with a width of 20 cm. The core is radially reflected by 20 cm of water. Owing to the symmetry along the x- and y-axes, this one-quarter reactor domain is denoted by Ω. It is composed of four sub-regions of different physical properties $\Omega_{1,2,3,4}$. The reactor is shown in Fig. 3. Neumann boundary conditions were enforced at the left and bottom boundaries. The group constants for this problem are listed in Table 3.

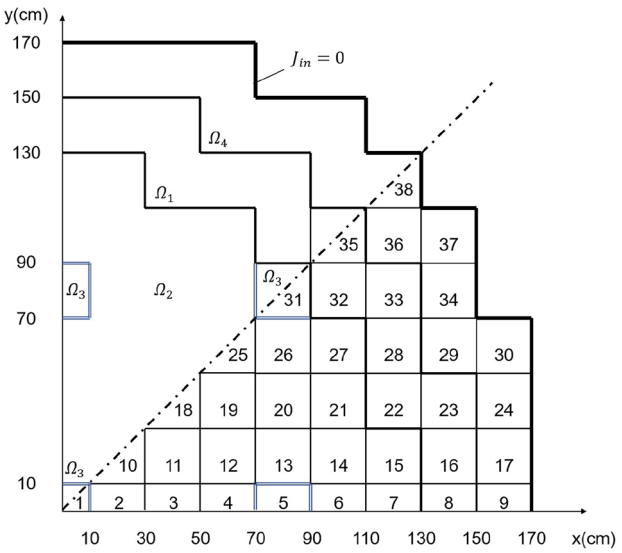


Fig. 3 Geometric layout of the 2D-IAEA benchmark Problem [43]

Table 3 Group constants of the 2D-IAEA benchmark

Region	Group	D_g (cm)	Σ_{ag} (cm^{-1})	$\nu\Sigma_{fg}$ (cm^{-1})	$\Sigma_{1\to2}$ (cm^{-1})
Ω_1	1	1.5	0.010	0.000	0.02
	2	0.4	0.080	0.135	
Ω_2	1	1.5	0.010	0.000	0.02
	2	0.4	0.085	0.135	
Ω_3	1	1.5	0.010	0.000	0.02
	2	0.4	0.130	0.135	
Ω_4	1	2.0	0.000	0.000	0.04
	2	0.3	0.010	0.000	

3 Methods

3.1 PINN loss formulation

In PINN, considering the general form of a parameterized and nonlinear PDE,

$$F(u, x, y, t, :) = 0, (x, y) \in \Omega, t \in [0, T], \quad (14)$$

where u represents the latent solution and Ω represents the solution domain. This formula can express PDEs in almost all the fields of mathematical physics.

N_f data points were sampled to measure the physical consistency. This type of loss is collectively referred to as the PDE loss. It has the following form:

$$Loss_{PDE} = \frac{1}{N_f}\sum_{i=1}^{N_f} F(u, x, y, t, :); x \in \Omega. \quad (15)$$

Subsequently, the initial and boundary losses owing to the known initial and boundary conditions are introduced. N_i and N_b data points are collected to calculate $Loss_{Initial}$ and $Loss_{Boundary}$, respectively. In addition, a few data points should be used for training.

$$Loss_{Initial} = \frac{1}{N_i}\sum_{i=1}^{N_i}(\phi_{truth} - \phi_{predict}); t = 0, x \in \Omega \quad (16)$$

$$Loss_{Boundary} = \frac{1}{N_b}\sum_{i=1}^{N_b}(\phi_{truth} - \phi_{predict}); x \in \partial\Omega \quad (17)$$

$$Loss_{\text{Data}} = \frac{1}{N_{\text{d}}} \sum_{i=1}^{N_{\text{d}}} (\phi_{\text{truth}} - \phi_{\text{predict}}) \tag{18}$$

Incorporating a small amount of labeled data provides information regarding the correct order of magnitude. This enables the PINN to better calibrate its predictions and prevent unrealistic or divergent solutions. The inclusion of such labeled data significantly contributes to the stability and accuracy of the training process for PINNs. By combining the above losses, networks can be penalized, and the training process is constrained effectively. This ensures that the obtained solutions adhere more closely to the fundamental laws of physics. Therefore, the final network loss formulation is as follows:

$$\begin{aligned} Loss_{\text{Total}} = & Loss_{\text{PDE}} + \\ & (Loss_{\text{Initial}} + Loss_{\text{Boundary}} + Loss_{\text{Data}}) \cdot w. \end{aligned} \tag{19}$$

The weights of the different loss functions and number of data points used to calculate these losses significantly affect the convergence speed and accuracy of the model. The final sampling ratio was determined through multiple experiments. The sampling ratio for the PDE loss, boundary loss, initial loss, and data loss was approximately 30:10:10:1. Multiple losses can be balanced by adjusting the weights w. This prevents the network from prioritizing a single loss, particularly when significant differences in the magnitude between losses exist.

3.2 S-CNN architecture

When solving the neutron diffusion equation, the performance of the NN can be affected significantly by the vanishing gradient problem as the network depth increases. This problem can straightforwardly result in training failure and render the results unreliable. Moreover, it significantly limits the increase in the number of layers, reduces the expressive power of the network, and limits the prediction accuracy of the network. Inspired by the effective alleviation of the gradient vanishing problem in the ResNet architecture [44] in the image domain (which introduces the concept of residual learning), networks can learn residual mappings (the difference between the input and desired output). This facilitates the training of very deep neural networks. Thus, a skip-connection mechanism was introduced into the PINN architecture.

The input sampling features are the coordinates, namely, x, y, and t. These are independent. Therefore, a separate filter was applied to each feature in the experiment. A one-dimensional convolution was used as the basis for each network layer. Each hidden layer is expressed as follows:

$$z_l = f_l(W_l z_{l-1} + b_l), \tag{20}$$

where z_l denotes hidden layer l between the input and output layers; W_l and b_l are the weight and bias, respectively; and $f_l(\cdot)$ denotes the activation function (e.g., the Tanh function).

On this basis, skip connections were added between different layers. The corresponding hidden layer is expressed as

$$z_l = f_l((W_l z_{l-1} + b_l) + z_{l-n-1}), l > n + 1, \tag{21}$$

where n represents the skip distance, that is, the number of crossed hidden layers. To determine where a skip connection should be added, the contribution of each layer to the training of the entire network was measured by calculating the gradient norm for each layer. Here, the gradient norm was computed as in [40]:

$$||g|| = \text{sqrt}(\text{sum}(g_i^2)). \tag{22}$$

For example, in the 10-layer network, the gradient contribution was calculated for each layer of the network (Fig. 4).

When the gradient is significant, the parameters of the layer can be updated conveniently. A large gradient results

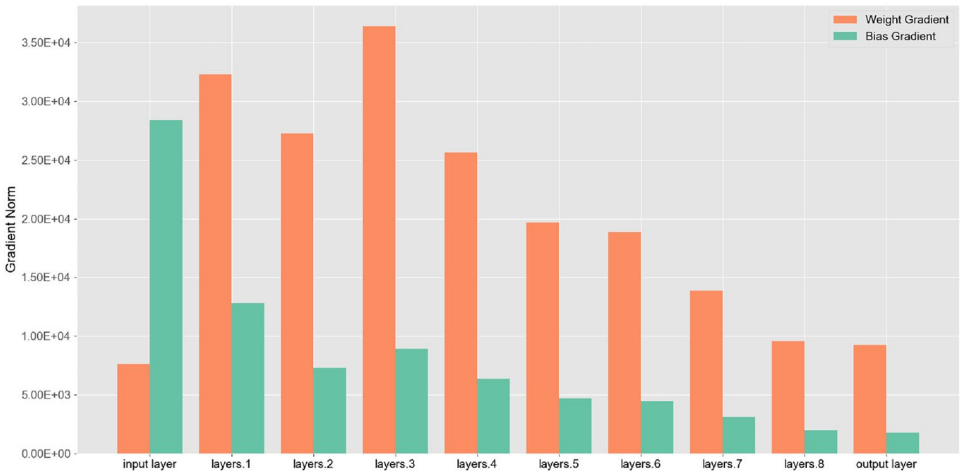


Fig. 4 (Color online) Gradient norm of each layer

in an unstable model, and a small gradient indicates that the layer may need to undergo more iterations to attain the optimal state and may encounter a gradient vanishing problem. Thus, the network increases the shallow gradient paths by establishing shallow-to-deep skip connections. This provides more gradient paths in the shallow layers and prevents vanishing gradients in the deep layers. The detailed 10-layer S-CNN architecture used in Sect. 4 is presented in Table 4. Here, B, C, and L represent the batch size, channel, and length, respectively.

In the experiment, the skip distance n was set to 3 for larger gradient propagation spans. It is the best parameter for shallow networks. When $n < 3$, the jump distance is insufficient. This may limit the network's capability to learn complex physical fields and does not prevent the gradient vanishing problem. When $n > 3$, the number of jumping layers is excessively large. This may pose a challenge to gradient propagation and, thus, increase the difficulty of optimizing the network. Experiments were conducted on S-CNN models with varying depths, and high-precision results were obtained consistently. This validated the effectiveness of the residual module in addressing the gradient problem.

3.3 RAR Mechanism

Owing to specific regions with large gradients in the overall solution domain, if sampled uniformly, the network displays inadequate fitting of the local regions. This issue can be addressed by increasing the weights of specific sampling points (as shown in Eq. (23)) or by resampling using Eq. (24):

$$X_{\text{new}} = X_{\text{raw}} + w \cdot X_{\text{f}} \tag{23}$$

$$X_{\text{new}} = X_{\text{raw}} + X_{\text{resample}} \tag{24}$$

where X_{raw} represents the original dataset, X_{new} represents the new dataset, X_{f} represents the set of sampling points with more significant PDE residuals, X_{resample} represents the set of resampling points, and w represents the penalty weight.

Meanwhile, referring to the grid division of conventional numerical computation methods, fine-grained samples should be collected from regions with large gradients to improve the prediction accuracy of the network in these regions.

Based on this, we consider using Eq. (24) to update our dataset, that is, introducing RAR [38] to improve the distribution of residual points during the training process of PINNs to address the bottleneck phenomenon that originates from the difficulty of reducing the PDE residuals in certain regions. This ultimately enhances the predictive accuracy of the model.

By selectively sampling more points in regions where the PDE residuals are significant, this approach enables the network to focus on challenging areas and adjust the sampling density accordingly. This, in turn, results in improved learning and prediction capabilities. The method is particularly effective for capturing the complex behavior of PDE solutions and identifying sharp gradient regions.

The pseudocode for the RAR method is as follows:

Algorithm 1 RAR Algorithm.

Input : Set S with randomly sampled initial points
Output: Updated set S
Divide the solution domain into α^2 subintervals;
Train PINN for n iterations;
repeat
  Compute $Loss_{\text{PDE}}$ for points in set S; Calculate the average residual of each subinterval;
  Randomly sample S' from the subinterval with the highest average residual. Update set S: $S = S \cup S'$;
  Train PINN for n iterations;
until *the maximum number of iterations is attained or the total number of points attains the limit*;

This algorithm divides the solution domain into α^2 subintervals with α subdivisions in both x- and t-directions. The Latin hypercube sampling (LHS) method [39] was used for random sampling. This ensured that sample points covered the entire space without clustering or bias.

As shown in Fig. 5, by adaptively adding points to regions with more significant residuals, more intensive sampling was conducted in one of the subdomains of the entire domain. This enabled the network to better capture the PDE solution's behavior. This adaptive sampling approach improved the distribution of residual points and mitigated the bottleneck phenomenon. Ultimately, it enhanced the accuracy and performance of the model and accelerated its convergence.

Table 4 S-CNN structure

Layer name	Input size(B,C,L)	Output size(B,C,L)	Param
input layer	(None,2,1)	(None,26,1)	78
conv1 layer	(None,26,1)	(None,26,1)	702
conv2 layer	(None,26,1)	(None,26,1)	702
conv3 layer	(None,26,1)	(None,26,1)	702
conv4 layer	(None,26,1)	(None,26,1)	702
conv5 layer	(None,26,1)	(None,26,1)	702
conv6 layer	(None,26,1)	(None,26,1)	702
conv7 layer	(None,26,1)	(None,26,1)	702
conv8 layer	(None,26,1)	(None,26,1)	702
output layer	(None,26,1)	(None,1,1)	27

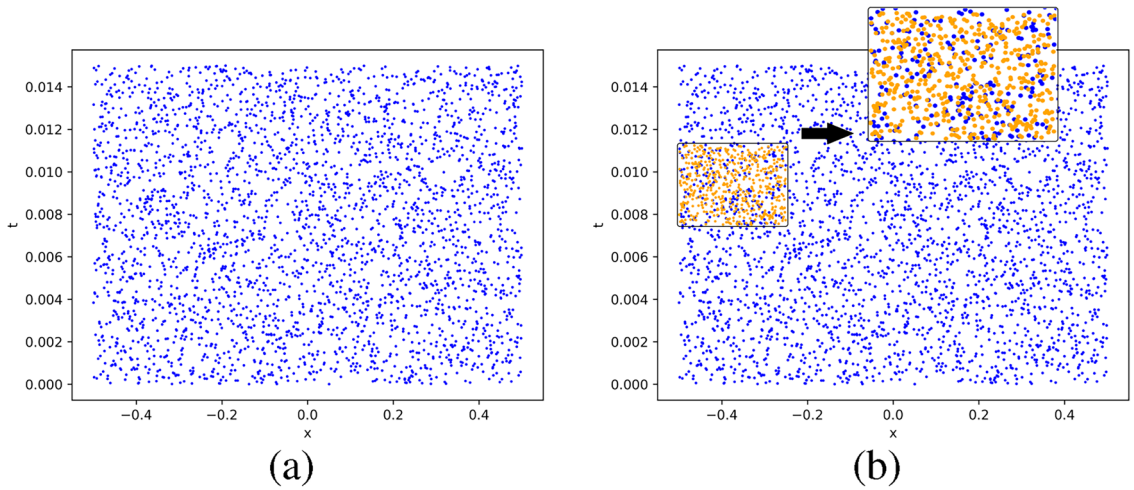


Fig. 5 (Color online) PDE points distribution

3.4 Proposed framework: R^2-PINN

Our study proposed an R^2-PINN architecture. It combines the PINN structure, S-CNN, and RAR mechanisms to solve PDEs with improved accuracy and computational efficiency.

The structure of R^2-PINN is shown in Fig. 6. In R^2-PINN, S-CNN is used as the backbone of the network. It can better capture the features of the PDE solution and improve the training efficiency. The RAR mechanism is employed to balance $Loss_{\mathrm{PDE}}$ across regions of the domain. This ensures that the network focuses on regions with large errors and adaptively refines the mesh.

By combining the PINN structure, S-CNN, and RAR mechanisms, the R^2-PINN can effectively solve PDEs with improved accuracy and computational efficiency. It leverages the capability of deep learning and adaptive refinement to capture the complex features of a PDE solution. Thereby, it enables a more precise representation of the physical phenomena. Section 4 describes the verification

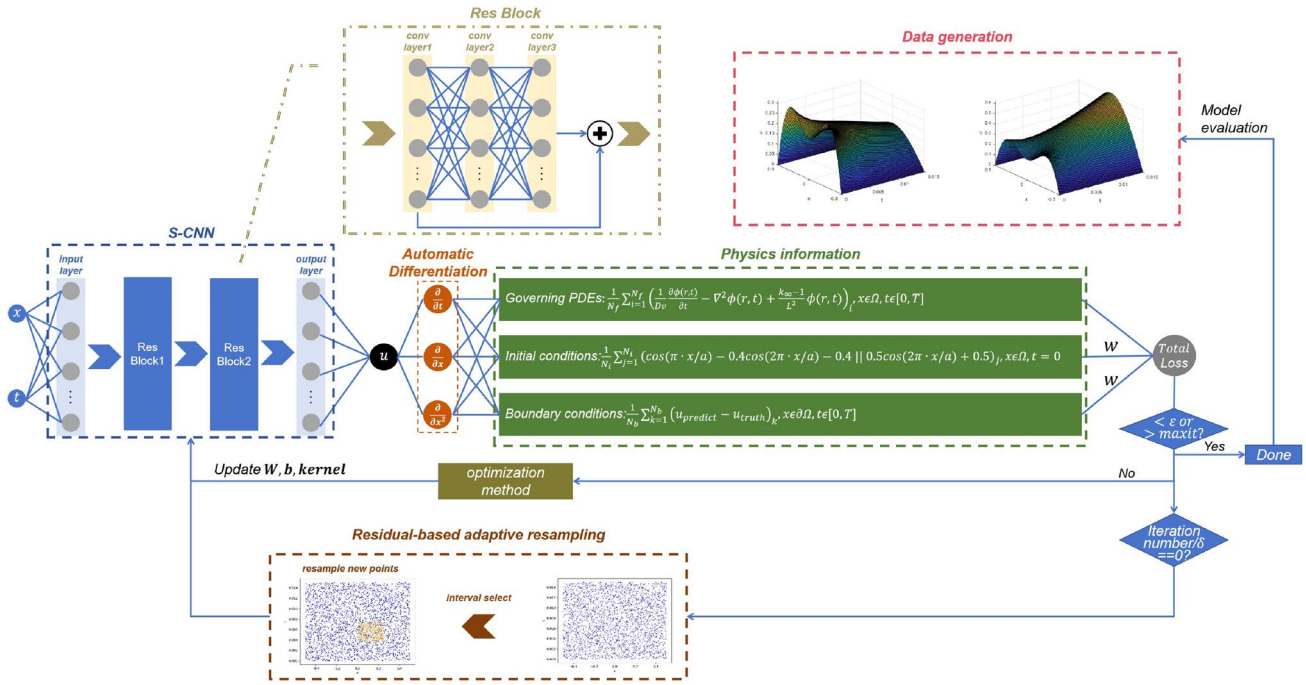


Fig. 6 (Color online) Overall architecture

of the superiority of the proposed R^2-PINN network and each mechanism through a series of experiments.

4 Experiments

In Sect. 4.1, the generation of datasets is introduced, specifically for testing the accuracy. Section 4.2 compares S-CNN networks with different depths to validate the superiority of the improved PINN network architecture. Section 4.3 investigates how resampling affects the model accuracy. In Sect. 4.4, R^2-PINN is implemented to search for k_∞, and its search efficiency is further enhanced in Sect. 4.5. Finally, in Sect. 4.6, the generalization capabilities of the improved PINN architecture model are presented. Furthermore, Sects. 4.7 to 4.9 evaluate the generalizability of the models in two-dimensional neutron diffusion, including multigroup and multi-material scenarios. Finally, Sect. 4.10 explains the strategy for selecting optimal hyperparameters for the R^2-PINN.

4.1 Dataset preparation

4.1.1 One-dimensional reactor diffusion equation for a single energy group

According to Eq. (3) and assuming symmetric initial conditions, two initial distributions were used for the dataset.

$$\begin{aligned} &\phi_1(x,0) = \cos(\pi \cdot x/a) - 0.4\cos(2\pi \cdot x/a) - 0.4; \\ &\phi_2(x,0) = 0.5\cos(2\pi \cdot x/a) + 0.5; \\ &x \in [-0.5, 0.5] \end{aligned} \tag{25}$$

The numerical values used in Eq. (2) are as follows: $v = 2.2 \times 10^3$ m/s , $D = 0.211 \times 10^{-2}$ m , $L^2 = 2.1037 \times 10^{-4}$ m^2, and $a = 1$ m. The boundary follows the delicacy boundary and predicts the flux distribution for 0.015 s. Data on how the flux varies with time for different parameter settings can be obtained by modifying the k_∞ parameter value.

Our test dataset consists of 10,000 data points with 100 grid points in both x- and t-directions. The experimental section tests the accuracy of the model prediction on this dataset.

4.1.2 Two-dimensional reactor diffusion equation for a single energy group

To ensure that the boundary flux was zero and that the full-domain flux was continuous, we set the initial flux distribution at time $t = 0$ to the following equation:

$$\begin{aligned} &\phi = (\exp(-(x^2 + y^2)/20) - \exp(-100)) \\ &x \in [-10, 10], y \in [-10, 10]. \end{aligned} \tag{26}$$

An iterative solution was obtained using 100 grids in the x-, y-, and t-directions to solve the flux distribution for 1 s, which was used as a dataset for testing the model.

4.1.3 Two-dimensional rectangular geometry multigroup multi-material diffusion problem

A source iteration method was used to solve the dual-group neutron flux and test the accuracy of the model. The ranges of the values of x and y are shown in Fig. 2. The total dataset consists of 10,000 data points with 100 grid points in both x- and y-directions. The boundary follows the delicacy boundary.

4.1.4 2D-IAEA problem

The reference solution for the two-dimensional two-group diffusion equations was obtained using the high-quality general-purpose finite-element solver FreeFem++ [45]. The eigenvalue problem was solved using Arpack ++. It is an object-oriented version of the ARPACK eigenvalue package [46]. The total number of samples in the dataset was 12,286 [47], with 76 data points as data fed into the network. Finally, all the samples in the dataset were used to validate the model accuracy.

4.2 S-CNN depth and kernel size ablation Experiment

All the parameters except for the base network were maintained consistent to compare the FCN and S-CNN architectures. The total number of sampled data points was 5000, with 3000 data points used to calculate $Loss_{\mathrm{PDE}}$, 1000 data points sampled for $Loss_{\mathrm{Initial}}$, and 1000 data points sampled for $Loss_{\mathrm{Boundary}}$. Network training does not involve feeding the data to calculate $Loss_{\mathrm{Data}}$. The Tanh activation function and LBFGS optimizer were used. The Gaussian distribution random sampling method was used to initialize the network weights and biases. The number of hidden neurons per layer in the network was set to 26.

To investigate the impact of the layer number and kernel size on the model performance, ablation experiments were conducted on S-CNN networks with different depths and kernel sizes. The padding was set to zero for kernel size one and to one for kernel size three. The detailed experimental results are listed in Table 5 and Fig. 7.

In Table 5 and Fig. 7, Ω MSE refers to the mean squared error (MSE) score over the entire domain, and Ω_1 MSE refers to the MSE score at t=0.015 s, which is calculated

Table 5 Ablation study on S-CNN depth and kernel size

	$k = 1.0041\,(\phi_0 = \phi_1)$				$k = 1.0001\,(\phi_0 = \phi_2)$			
	Kernel size=1		Kernel size=3		Kernel size=1		Kernel size=3	
S-CNN Layer	Ω MSE	Ω_1 MSE	Ω MSE	Ω_1 MSE	Ω MSE	Ω_1 MSE	Ω MSE	Ω_1 MSE
Baseline	8.9×10^{-7}	3.8×10^{-6}	8.9×10^{-7}	3.8×10^{-6}	4.4×10^{-6}	1.8×10^{-5}	4.4×10^{-6}	1.8×10^{-5}
6	1.2×10^{-7}	3.1×10^{-8}	1.4×10^{-7}	6.2×10^{-8}	2.4×10^{-7}	1.6×10^{-7}	1.4×10^{-7}	2.1×10^{-7}
7	1.5×10^{-7}	5.4×10^{-8}	1.4×10^{-7}	6.5×10^{-8}	1.8×10^{-7}	1.3×10^{-7}	1.7×10^{-7}	1.6×10^{-7}
8	9.6×10^{-8}	2.6×10^{-8}	1.4×10^{-7}	6.5×10^{-8}	1.8×10^{-7}	1.3×10^{-7}	1.7×10^{-7}	1.6×10^{-7}
9	9.9×10^{-8}	3.9×10^{-8}	3.9×10^{-7}	6.6×10^{-8}	1.1×10^{-7}	3.1×10^{-8}	3.6×10^{-7}	2.0×10^{-8}
10	8.3×10^{-8}	4.0×10^{-8}	2.2×10^{-7}	3.9×10^{-8}	5.8×10^{-8}	1.9×10^{-8}	8.8×10^{-7}	3.4×10^{-7}
11	1.6×10^{-7}	1.4×10^{-7}	2.2×10^{-7}	7.4×10^{-8}	5.8×10^{-8}	2.8×10^{-8}	6.8×10^{-6}	6.3×10^{-6}
12	2.1×10^{-7}	1.2×10^{-7}	3.5×10^{-7}	6.1×10^{-7}	2.3×10^{-7}	3.2×10^{-7}	2.6×10^{-7}	2.7×10^{-7}
F13	1.3×10^{-7}	1.1×10^{-7}	5.1×10^{-7}	6.0×10^{-7}	8.1×10^{-8}	9.3×10^{-8}	1.8×10^{-7}	3.1×10^{-8}
14	1.5×10^{-6}	1.4×10^{-7}	4.3×10^{-7}	1.2×10^{-6}	1.8×10^{-7}	2.7×10^{-7}	1.3×10^{-7}	1.4×10^{-7}
15	1.6×10^{-7}	1.1×10^{-7}	4.4×10^{-7}	1.3×10^{-6}	1.3×10^{-7}	2.9×10^{-8}	1.4×10^{-7}	5.2×10^{-8}
16	1.5×10^{-7}	9.7×10^{-8}	1.7×10^{-7}	1.7×10^{-7}	1.9×10^{-7}	2.6×10^{-8}	3.0×10^{-7}	5.8×10^{-8}

separately to assess the capability of the model to extrapolate in time. The baseline refers to the results of FCN.

From Table 5 and Fig. 7, it can be observed that when the number of layers is less than 10, the accuracy shows a positive correlation with the number of layers. However, as the number of layers increases further, the accuracy does not increase. Hence, it is preferable to select fewer layers to make the NN train faster and achieve a high accuracy. This is because although the expressive capability of the neural network increases with the number of layers, beyond a certain threshold, an increase in the number of layers does not significantly improve the expressive capability of the network. An increase in the number of layers only causes an increase in the training time, and the model experiences overfitting problems. Moreover, because the input features are coordinates and not related to each other, the kernel size set to one can consider each channel as an independent feature to be addressed. This operation is more in line with reality. Therefore, a kernel size set to one can achieve a higher accuracy than a kernel size set to three. Based on the above analysis, the 10-layer S-CNN with a kernel size of one exhibited the best predictive accuracy with the minimum number of parameters. Consequently, this configuration was used in the subsequent experiments.

4.3 Ablation experiment on resampling parameters in S-CNN

According to the RAR algorithm, the resampling granularity was set to α to define the granularity of the subdomains in the solution domain. The solution domain was divided into α subintervals along the x and t dimensions. This resulted in α^2 subintervals. Following the RAR algorithm, in every 1000 epochs, the network calculates and compares the MSEs of each subdomain and performs resampling in the subinterval with the largest MSE. The number of resampling points is denoted by m.

The initial PDE sampling point was set to 3000 to prevent excessive sampling and training. Moreover, 2000 data points were sampled, with 1000 points to calculate the initial loss and 1000 points to calculate the boundary loss. The maximum number of PDE samples was set to 5000. Resampling was stopped when the total number of PDE samples attained 5000 after multiple iterations. The LHS method was used to resample.

Ablation experiments were conducted in two dimensions (resampling granularity and number of resampling points) while maintaining an identical S-CNN architecture (10 layers) and the other hyperparameters. The S-CNN network was compared with an FCN (baseline) using identical initial conditions, that is, $\phi_0 = \phi_1$. The detailed experimental results are presented in Tables 6 and 7.

From Tables 6 and 7, it is evident that under different sampling hyperparameters, our network consistently achieved a better Ω_1 MSE result. It was two orders of magnitude higher than that of the FCN baseline. When using the optimal hyperparameters, the R^2-PINN achieved an accuracy of 10^{-8} or even 10^{-9}. This indicates that our method has a significant advantage in determining whether the flux values attain a steady state at a specific instant. This advantage was demonstrated during the k_∞ search described in Sect. 4.4.

Furthermore, the results were obtained by adopting another initial condition, where $\phi_0 = \phi_2$. A boxplot analysis of the resampling hyperparameters for all the results is shown in Figs. 8 and 9. When the number of resamples is

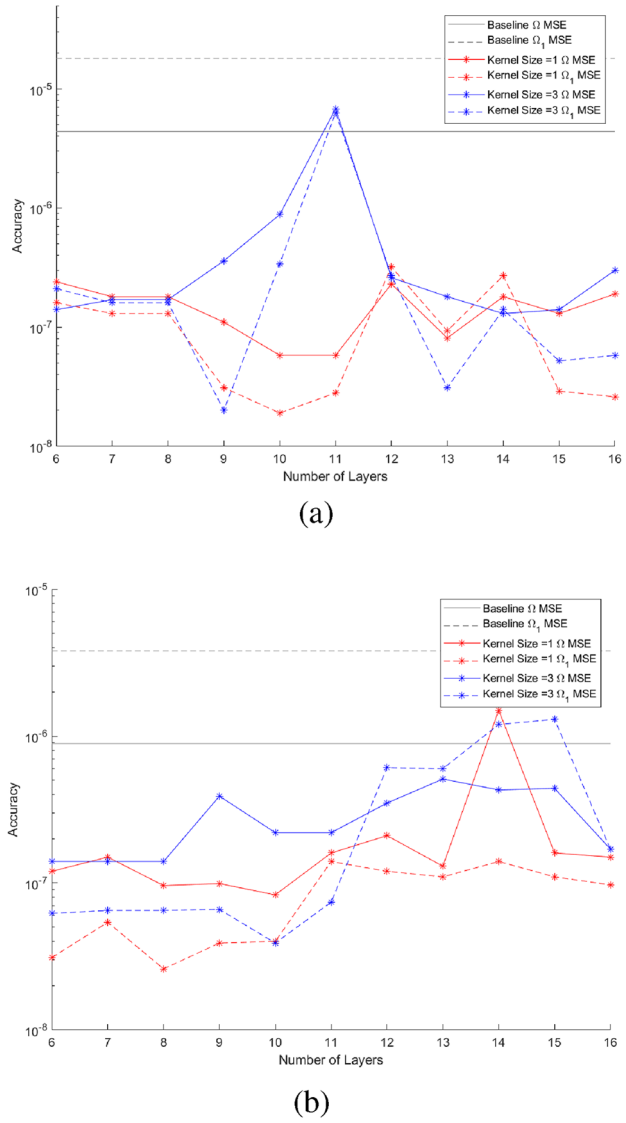


Fig. 7 Comparative accuracy with different model parameters. **a** $\phi_0 = \phi_1$; **b** $\phi_0 = \phi_2$

Table 6 Ablation study on resampling numbers

Resample numbers m	Ω MSE	Ω_1 MSE
Baseline	8.9×10^{-7}	3.8×10^{-6}
0	9.1×10^{-8}	2.1×10^{-8}
100	1.3×10^{-7}	8.4×10^{-8}
200	9.5×10^{-8}	9.8×10^{-9}
300	1.3×10^{-7}	5.4×10^{-8}
400	9.8×10^{-8}	4.1×10^{-8}
500	8.0×10^{-8}	4.9×10^{-9}
600	1.3×10^{-7}	3.1×10^{-8}
700	1.0×10^{-7}	2.0×10^{-8}
800	3.9×10^{-8}	1.8×10^{-8}

Table 7 Ablation study on resampling granularity

Resample granularity α	Ω MSE	Ω_1 MSE
Baseline	8.9×10^{-7}	3.8×10^{-6}
2	8.0×10^{-8}	4.9×10^{-9}
3	1.5×10^{-7}	8.5×10^{-8}
4	1.2×10^{-8}	9.1×10^{-8}
5	6.4×10^{-7}	3.5×10^{-8}
6	1.0×10^{-7}	8.8×10^{-8}

500 and the resample granularity is set to two, the model achieves the best accuracy.

Given the introduction of the RAR mechanism, it is necessary to consider whether there is a significant time overhead. Hence, time comparisons were performed for different numbers of samples. The results are listed in Table 8. It can be observed that when the number of resamplings was set to 500, the training time required by the model reduced significantly compared with that when it was set to zero. By calculating the time consumed per cycle, we determined that this trend did not increase significantly. Thus, setting an appropriate number of resamplings effectively reduced the overall network training time. This shows that the RAR method involves an increase in the number of sampling points. This results in a different density of samples in each region, as well as a relatively large number of sampling points in complex regions. This contributes substantially to the convergence speed of the network.

The model uses the optimal resampling parameter configuration to predict the flux distribution under different k_∞ values where $\phi_0 = \phi_1$. The error plot is shown in Fig. 10. Except for the relatively large errors at $t = 0$ s, the errors in the other regions were relatively flat. Significantly, as the time increased, the errors increased marginally. In addition, Fig. 11 shows the distribution of the values predicted by the model. When t approaches zero, there is a specific deviation between the predicted result in the boundary region and zero. This guided us to appropriately increase the weights of the $Loss_{\text{Boundary}}$ and $Loss_{\text{Initial}}$.

To evaluate the performance of the model and measure the difference between the losses, Fig. 12 shows the training and testing losses, respectively. R^2-PINN converged in 2000 epochs. In Fig. 12, $Loss_{\text{PDE}}$ is larger than $Loss_{\text{Boundary}}$ and $Loss_{\text{Initial}}$. This may result in $Loss_{\text{PDE}}$ dominating the optimization process, whereas the other losses cannot be optimized effectively. To address this issue, w in Eq. (19) was set to 100 to impose higher penalties on the boundary and initial region error.

Based on the experimental results in Sects. 4.2 and 4.3, an optimized S-CNN architecture with an optimized RAR mechanism was used to compose the R^2-PINN for further

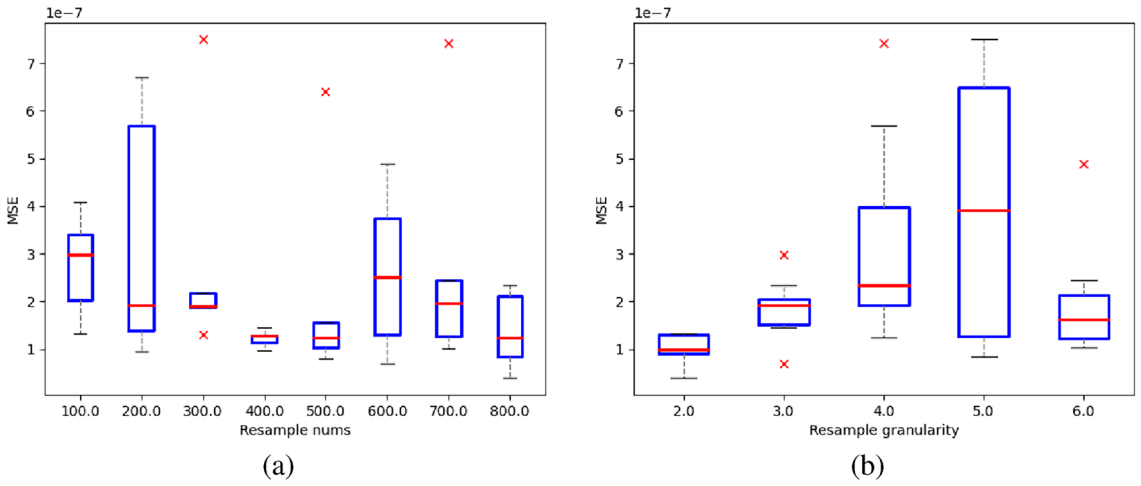


Fig. 8 MSE Comparison Between Parameters. **a** m Comparison; **b** α Comparison

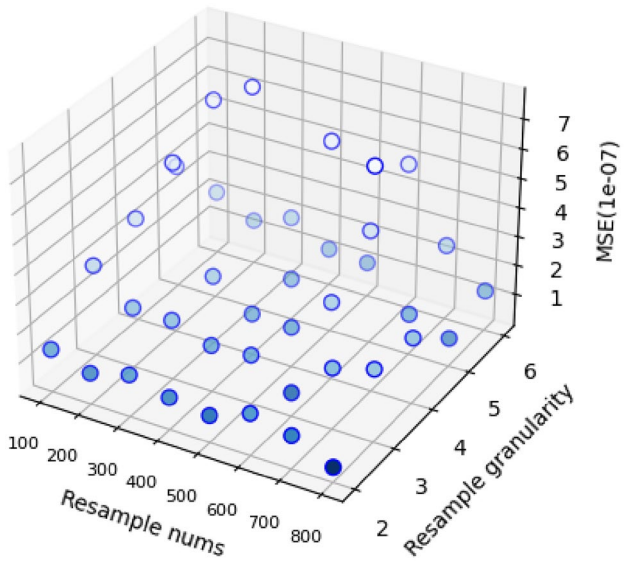


Fig. 9 Resample parameter 3D visualization

use in searching critical parameters. This is discussed in Sects. 4.4 and 4.5.

4.4 k_∞ Search with R^2-PINN

For a given geometric shape and volume of the reactor core, k_∞ and L^2 can be adjusted by modifying the reactor core size or modifying the composition of the materials within the reactor such that k_{eff} is one. When the system attains a steady state after a sufficient period, the neutron flux density follows the distribution described by Eq. (6), and the reactor is in a critical state.

Table 8 Training performance with different resample numbers

m	Training epochs	Training time (s)	Avg. time per epoch (s)
0	3262	275.28	0.084
100	3875	300.36	0.078
200	1005	84.25	0.084
300	2004	174.32	0.087
400	2544	164.16	0.065
500	**1004**	**84.48**	**0.084**
600	3004	240.74	0.080
700	3004	287.95	0.096
800	2005	140.02	0.070

In this experiment, the L^2 parameter size was not altered. Moreover, different networks were trained by adjusting k_∞ to predict the evolution of the neutron flux at this parameter. By continuously adjusting the value of k_∞, we attempted to adjust k_{eff} to one, thereby attaining the critical state. Initially, the parameter search range was set as [1.0001, 1.0041]. It has been verified that when k_∞ is 1.0001, $k_{\text{eff}} < 1$. This indicates a subcritical state in which $\phi(x,t)$ decays exponentially with time t. When k_∞ is 1.0041, $k_{\text{eff}} > 1$ indicates a supercritical state in which $\phi(x,t)$ increases continuously. The specific distributions are shown in Fig. 13.

In the absence of delayed neutrons, when the system approaches criticality it converges to a critical state within a significantly short time (5 – 8 ms) regardless of the boundary conditions. Here, ϕ does not vary. At this point,

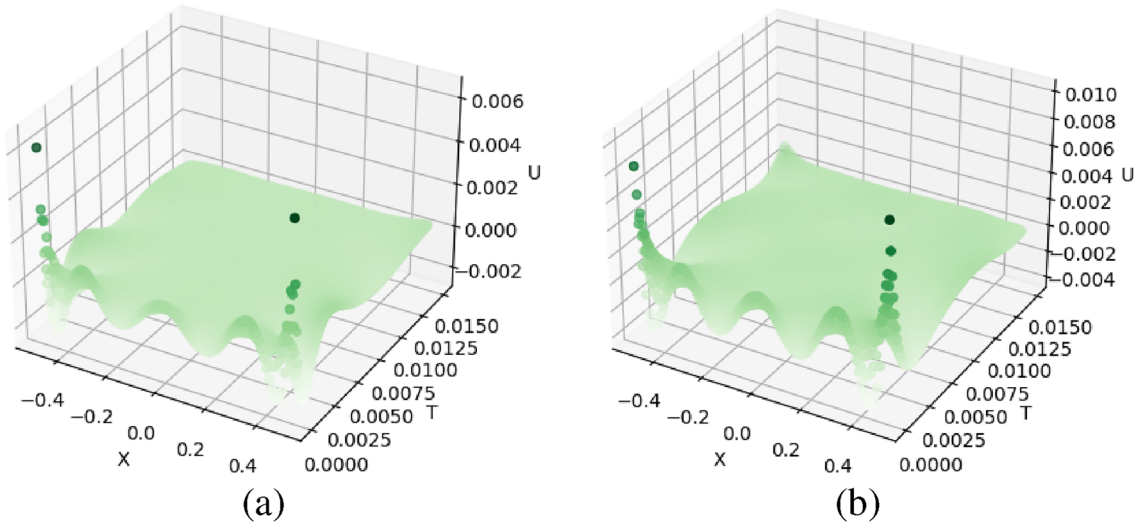


Fig. 10 (Color online) Error field. **a** $k_\infty = 1.0001$; **b** $k_\infty = 1.0041$

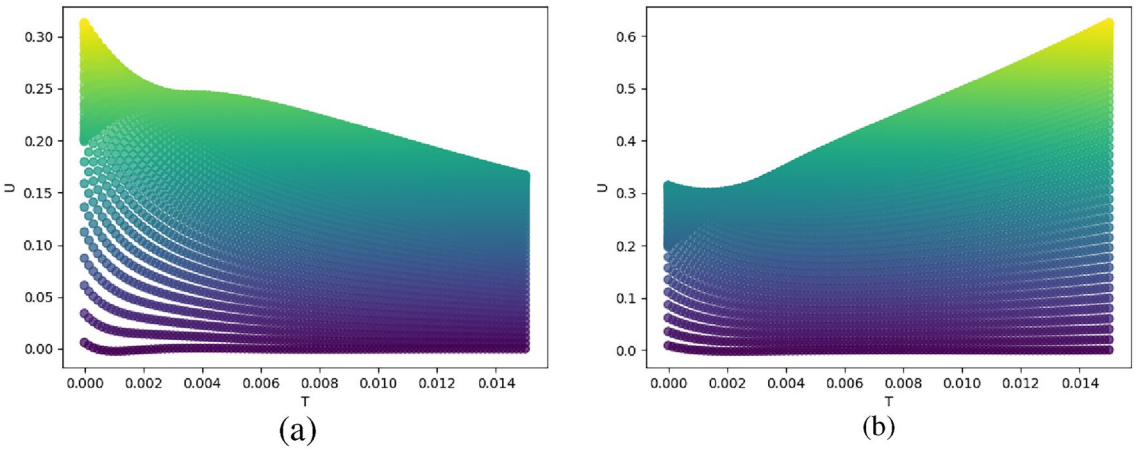


Fig. 11 (Color online) Predict result. **a** $k_\infty = 1.0001$; **b** $k_\infty = 1.0041$

the ϕ-distribution is a steady-state analytical solution. The parameter search interval is partitioned into n equal parts, yielding multiple k_∞ values. The network is trained for each value, calculates ϕ_t after a specific time t_τ, and performs a search until ϕ_t approaches zero within a reasonable accuracy range. The process is illustrated in Fig. 14.

In each network, Eq. (2) is used as the equation for $Loss_{\mathrm{PDE}}$. Equation (25) is used as the equation for $Loss_{\mathrm{Initial}}$. The MSE between the boundary predicted values and zero is used to compute $Loss_{\mathrm{Boundary}}$.

Using the automatic differentiation mechanism of the NN, the partial derivative of ϕ for t was calculated after obtaining the predicted values. Given the network's capability to predict flux variations within a time interval of 0.015 s, the experiment used the last five time points to calculate $\partial\phi(r,t)/\partial t$ and to determine whether the system was in a critical state [29].

Using Fig. 14, searches for k_∞ when the initial distribution follows ϕ_1 and ϕ_2 were conducted. Each search result was recorded. ϕ_t as a function of k_∞ is shown in Fig. 15. When $n = 2$ and the grid method degenerates into a binary search, only approximately 20 iterations are required to obtain the search results with an accuracy level of 10^{-5}. The total runtime of the program was approximately 30 min.

From Fig. 15, ϕ_t varies exponentially with k_∞. This further indicates that gradient-based methods such as gradient descent or Newton's method can be evaluated to search for parameter values more rapidly and efficiently. Alternatively, using curve-fitting techniques to fit the scattered data and the intersection of the resulting curve with the x-axis yields the value of k_∞ at the critical state. This process can

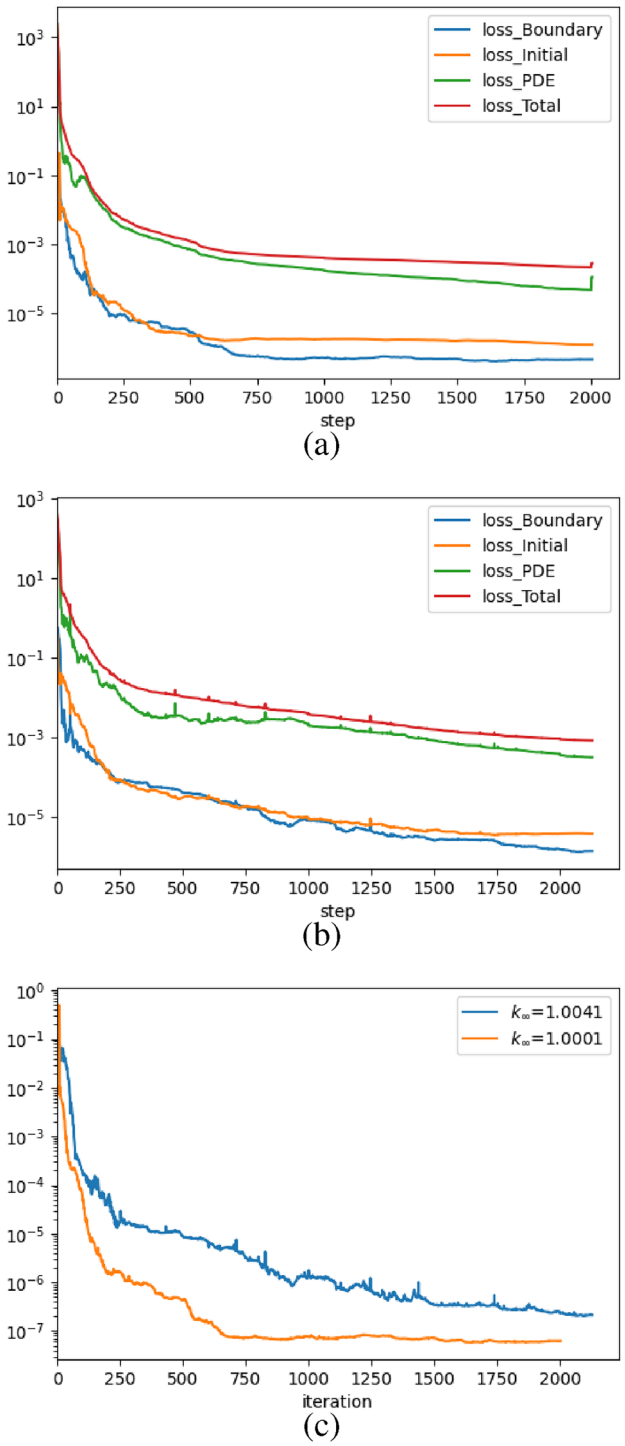


Fig. 12 Loss over iterations. **a** Training loss ($k_\infty = 1.0001$); **b** Training loss ($k_\infty = 1.0041$); **c** Test loss

be completed in a small number of search iterations. The k_∞ value corresponding to the critical state is identified by progressively refining the interval. The search results are presented in Table 9. Among these, $\Delta\phi$ at the last five time points is recorded to determine whether ϕ attains a steady state. Compared with the results of the FCN [29], the R^2-PINN has a smaller $\Delta\phi$. This implies that the R^2-PINN search has a higher accuracy than FCN. Furthermore, the search results were validated by generating the ϕ distribution for the obtained k_∞ values, as shown in Fig. 16.

From Fig. 16, the flux tends to stabilize as t increases. This implies that the system has attained a critical state. This indicates that our network can effectively search for optimal parameters corresponding to the critical state.

4.5 k_∞ Search efficiency improvement

When searching for k_∞, the initial search interval is large. The k_∞ value to be searched differs considerably from the critical state k value. Thus, the accuracy of prediction is not highly required. Only the prediction is used to compute ϕ_t to serve as a priori information for the next interval refinement. Therefore, during the training section, we examined whether the rate of variation in the fluxes converged at regular intervals. When ϕ_t converges, the training section is stopped, and the next k-value network training begins. This results in a faster parameter search without loss of accuracy. The equation for determining when to stop network iteration is as follows:

$$((\phi_t)_{i+1} - (\phi_t)_i)/((\phi_t)_i - (\phi_t)_{i-1}) < \lambda. \tag{27}$$

The parameter λ was set to 0.01. At every 200 iterations, Eq. (27) was computed to determine if the network had stopped iterating. The comparative results for $\phi_0 = \phi_1$ are listed in Table 10. The early termination mechanism can significantly reduce the search time (0.56 times the original one).

Furthermore, as shown in Fig. 15, ϕ_t is exponentially related to k_∞. Therefore, it can be used to determine the critical value of k_∞ by fitting a quadratic function. To optimize the search algorithms, an experiment was conducted to compare three search methods: binary, grid, and quadratic fitting. Using R^2-PINN with an early termination mechanism, the results are shown in Fig. 17.

The quadratic fitting search method determined k_∞ rapidly in 250 s and attained an accuracy of 10^{-4}. Meanwhile, the grid search method determined k_∞ in 522 s and attained an accuracy of 10^{-5}. Different methods can be selected for parameter searches based on the actual requirement for time and accuracy.

4.6 Network prediction for different k_∞: R^2-PINN's robustness

The network was trained at different k_∞ values. The results were compared with the analytical solution generated to

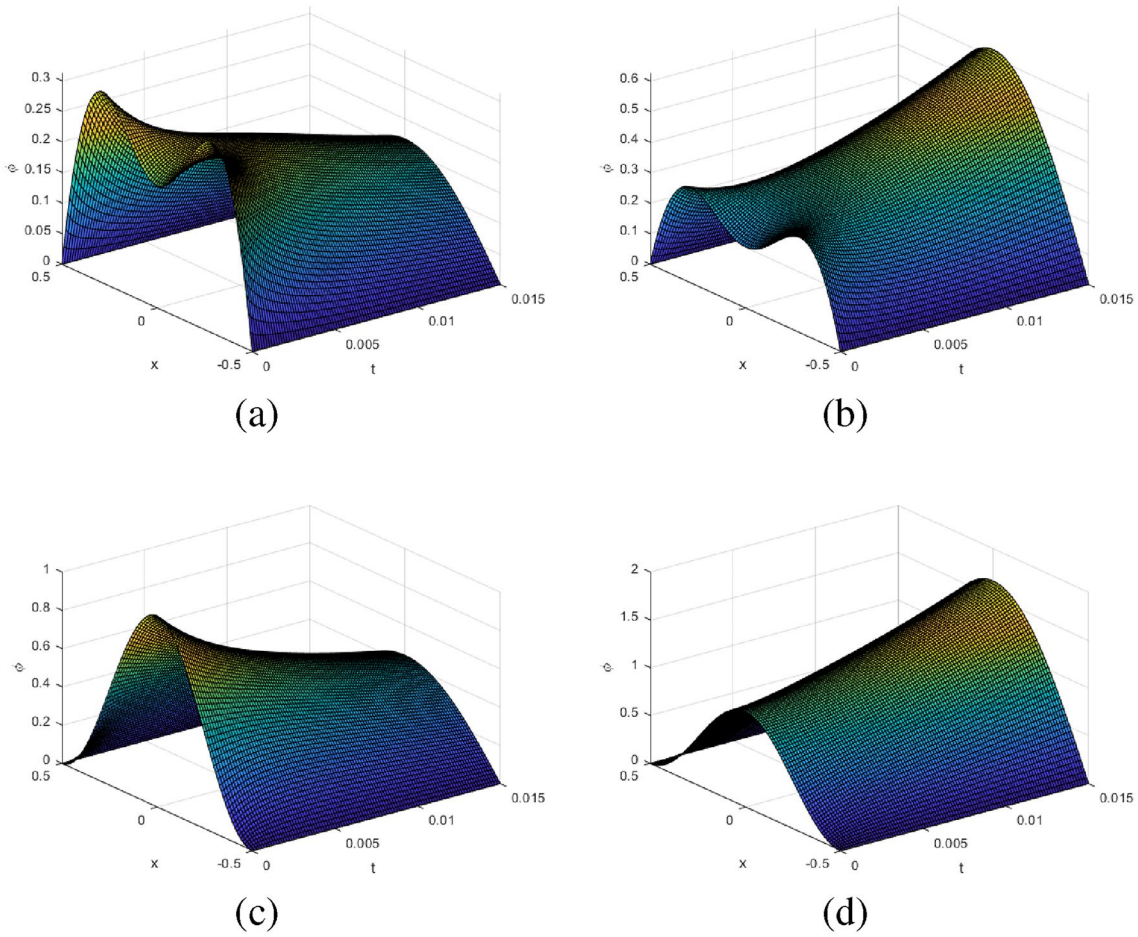
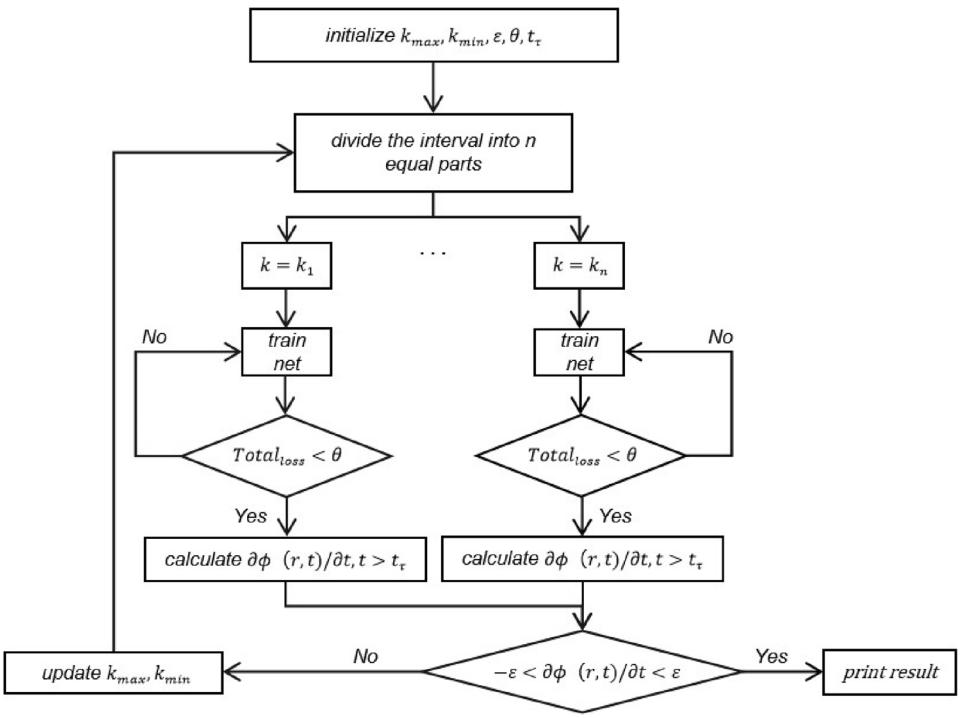


Fig. 13 (Color online) ϕ Distribution. **a** $\phi_0 = \phi_1$, $k_\infty = 1.0001$; **b** $\phi_0 = \phi_1$, $k_\infty = 1.0041$; **c** $\phi_0 = \phi_2$, $k_\infty = 1.0001$; **d** $\phi_0 = \phi_2$, $k_\infty = 1.0041$

Fig. 14 Search flowchart [29]



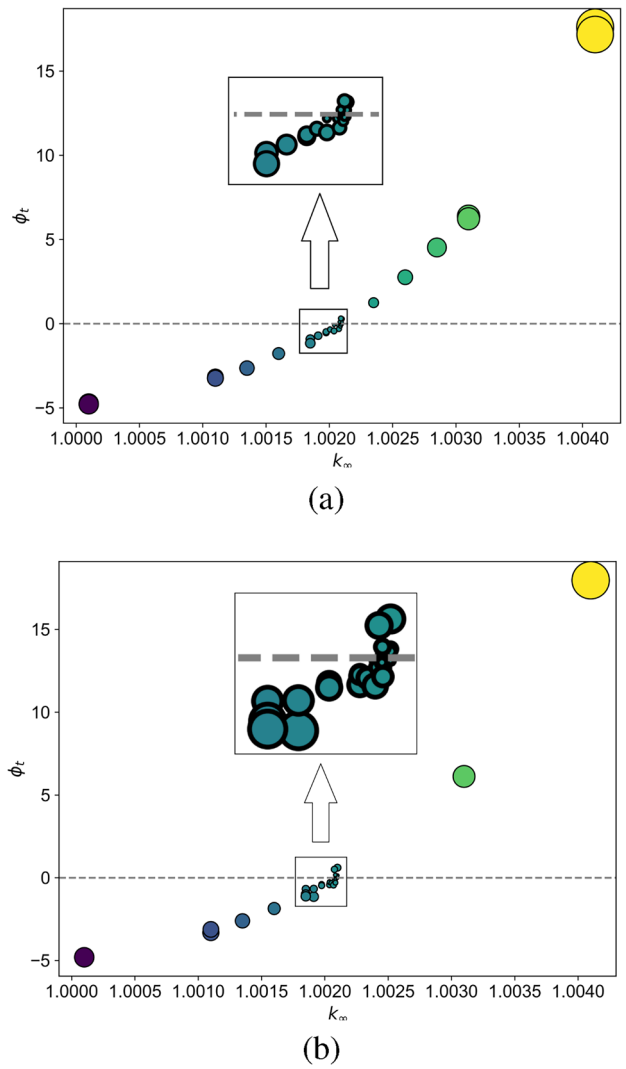


Fig. 15 (Color online) Search result of ϕ_t. **a** $\phi_0 = \phi_1$; **b** $\phi_0 = \phi_2$

Table 9 Search result

	$\phi_0 = \phi_1$		$\phi_0 = \phi_2$	
	k_∞	$\Delta\phi$	k_∞	$\Delta\phi$
FCN	1.00202	−0.017973	0.99803	−0.788704
R^2-PINN	1.0020822	−0.001522	1.0020824	−0.001519

obtain the accuracy at each k_∞ value, as shown in Fig. 18, to validate the robustness of the R^2-PINN.

The robustness of R^2-PINN is illustrated in Fig. 18. Across the tested values, the network consistently maintained a high accuracy of at least 10^{-7}. This indicated its capability to handle various scenarios and maintain reliable predictions. The observed differences in MSE across parameter values were relatively small. This further highlights the stability and effectiveness of the network. The robustness of the R^2-PINN network is evident from its consistently high accuracy for different parameter values.

4.7 R^2-PINN for solving a two-dimensional reactor diffusion equation for a single energy group

In this experiment, an 11-layer S-CNN network was used for training. The remaining hyperparameters were selected as described in Sect. 4.4.

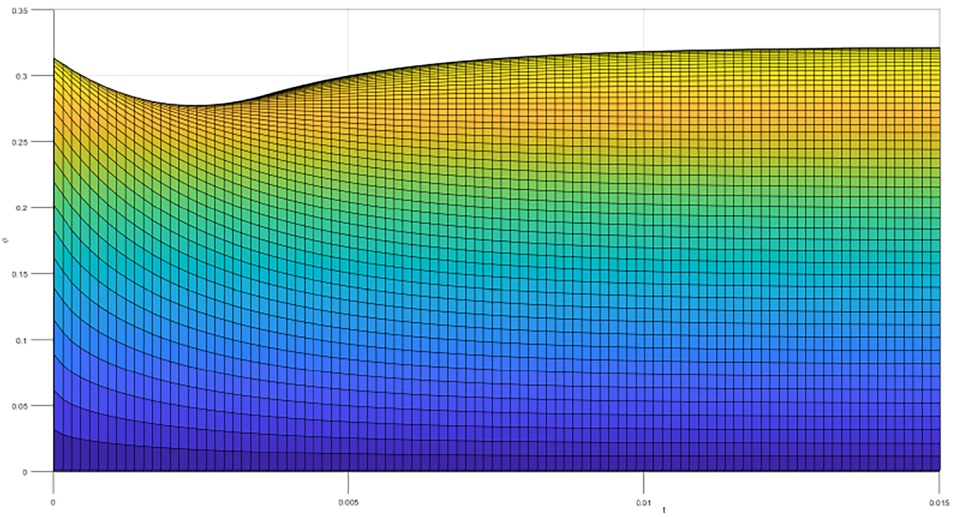
To validate the generalizability of the models, an experiment was conducted between different k_∞. The results are shown in Table 11. Here, the network achieved an accuracy of 10^{-5} under different parameters, and the MSE for the extrapolation region (that is, the region with $t = 1$ s) still achieved an accuracy of 10^{-5}. This demonstrates that the model performs well in terms of temporal extrapolation capability and can be used to search for parameters corresponding to the critical state.

Searching for the k_∞ parameter in the manner described in Sect. 4.4 yields $k_\infty = 1.1378$. Using the source iteration method, we determined that $k_\infty = 1.1202$. The calculation error is approximately 1.5%. The results of the prediction and reference truths are presented in Fig. 19. The error plots are shown in Fig. 20.

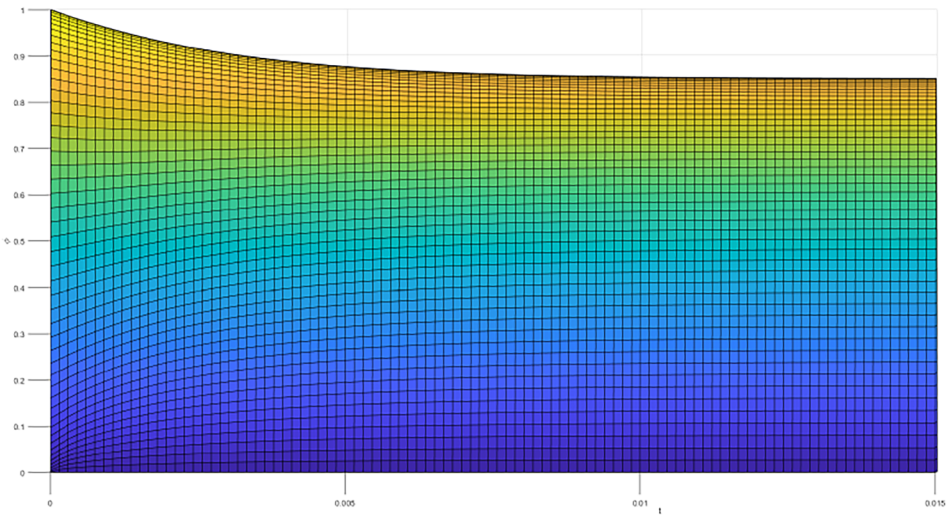
4.8 R^2-PINN for solving two-dimensional rectangular geometry multigroup multi-material diffusion problem

In this two-group diffusion problem, each neutron group's flux is predicted independently to enhance the accuracy. Specifically, we separately model the fast group neutron flux (ϕ_1) and hot-group neutron flux (ϕ_2) using dual PINNs with a criticality factor of $k_{\text{eff}} = 0.9693$. Given the interconversion relationship between the fluxes of these two energy groups, dual PINNs are designed to share loss functions and are optimized sequentially. This shared loss structure ensures that the interaction between the energy groups is captured effectively while allowing for customized optimization paths for each flux. To train the models, a four-layer S-CNN was employed. The other PINN-related parameters, such as the sampling ratios, activation functions, and optimization strategies, remained consistent with those detailed in Sect. 4.4.

Using R^2-PINN, the MSE of ϕ_1 attains 1.36×10^{-6} and that of ϕ_2 attains 2.49×10^{-7}. To better illustrate the flux distribution, particularly the abrupt variations in the neutron flux at the material interfaces for different



(a)



(b)

Fig. 16 (Color online) Steady-state verification. **a** $\phi_0 = \phi_1$; **b** $\phi_0 = \phi_2$

Table 10 Time Consume Comparation

	Cost time (s)	Search result
FCN	7841	1.00202
R^2-PINN	1837	1.00208
R^2-PINN(Early termination)	1027	1.00207

neutron groups, cross-sectional views of the neutron flux were adopted with slices extracted from the x-z and y-z planes. The results and reference solutions are presented in Figs. 21 and 22.

The error fields for each energy group are presented in Fig. 23. Owing to the small value of the flux, to clarify the error representation, we calculated the loss rate. This is shown in Fig. 23. The specific equations are given in Eq. (28).

$$\text{loss rate} = ABS(\phi_{\text{predict}} - \phi_{\text{truth}})/\phi_{\text{truth}} \tag{28}$$

The results show that we can effectively determine the flux distribution under the specified k_{eff} according to the steady-state equations. Additionally, according to

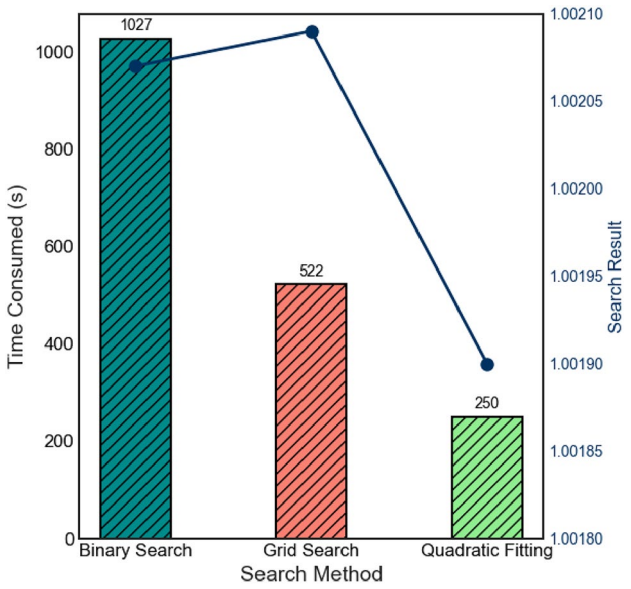


Fig. 17 Comparison of search method

the experiments presented in Sects. 4.4 and 4.7, we can determine the critical parameters according to the transient equations. Thus, the model can be adapted well to solve the two major problems of nuclear reactors, that is, solving for the fluxes and searching for the steady-state parameters.

4.9 R²-PINN for solving 2D-IAEA problem

Using a six-layer R²-PINN structure, we incorporated k_{eff} as a parameter for iterative optimization in neural network training. The model utilized 18,000 points for computing $Loss_{\text{PDE}}$, 500 points for $Loss_{\text{Boundary}}$, and 76 points for $Loss_{\text{Data}}$.

Finally, the relative error and relative L_∞ error were used to evaluate the accuracy of the R²-PINN. Herein, the relative L_∞ error is particularly important in the nuclear engineering domain. The specific equations are expressed in Eqs. (29) and (30). The prediction results and reference truth are presented in Fig. 24. The absolute error plots are shown in Fig. 25.

$$e_{\text{r}} = ABS(\phi_{\text{predict}} - \phi_{\text{truth}})/\phi_{\text{truth}} \tag{29}$$

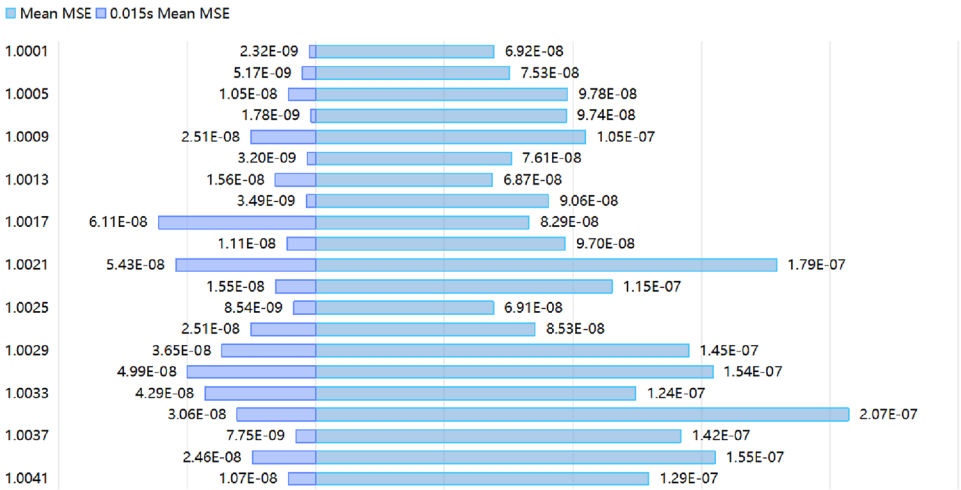
$$e_{\infty} = \frac{\left\|\phi_{\text{predict}} - \phi_{\text{truth}}\right\|_{\infty}}{\left\|\phi_{\text{truth}}\right\|_{\infty}} \tag{30}$$

Considering the engineering acceptance criteria for the 2D IBP, the flux calculation error in fuel assemblies with a relative flux higher than 0.9 should be less than 5%. In fuel assemblies with a relative flux less than 0.9, the flux calculation error should be less than 8%. In addition, the

Table 11 MSE results in different k_∞

k_∞	MSE	t=1 s MSE
1.1	4.4×10^{-6}	9.0×10^{-6}
1.11	1.1×10^{-6}	1.7×10^{-6}
1.12	3.2×10^{-6}	5.2×10^{-6}
1.13	1.2×10^{-6}	2.2×10^{-6}
1.14	1.8×10^{-6}	3.0×10^{-6}
1.15	8.5×10^{-7}	1.6×10^{-6}
1.16	2.4×10^{-6}	4.4×10^{-6}
1.17	1.1×10^{-5}	2.2×10^{-5}
1.18	5.5×10^{-6}	9.3×10^{-6}
1.19	2.2×10^{-6}	4.7×10^{-6}
1.2	4.5×10^{-6}	8.2×10^{-6}

Fig. 18 MSE Accuracy Comparison for Different k_∞



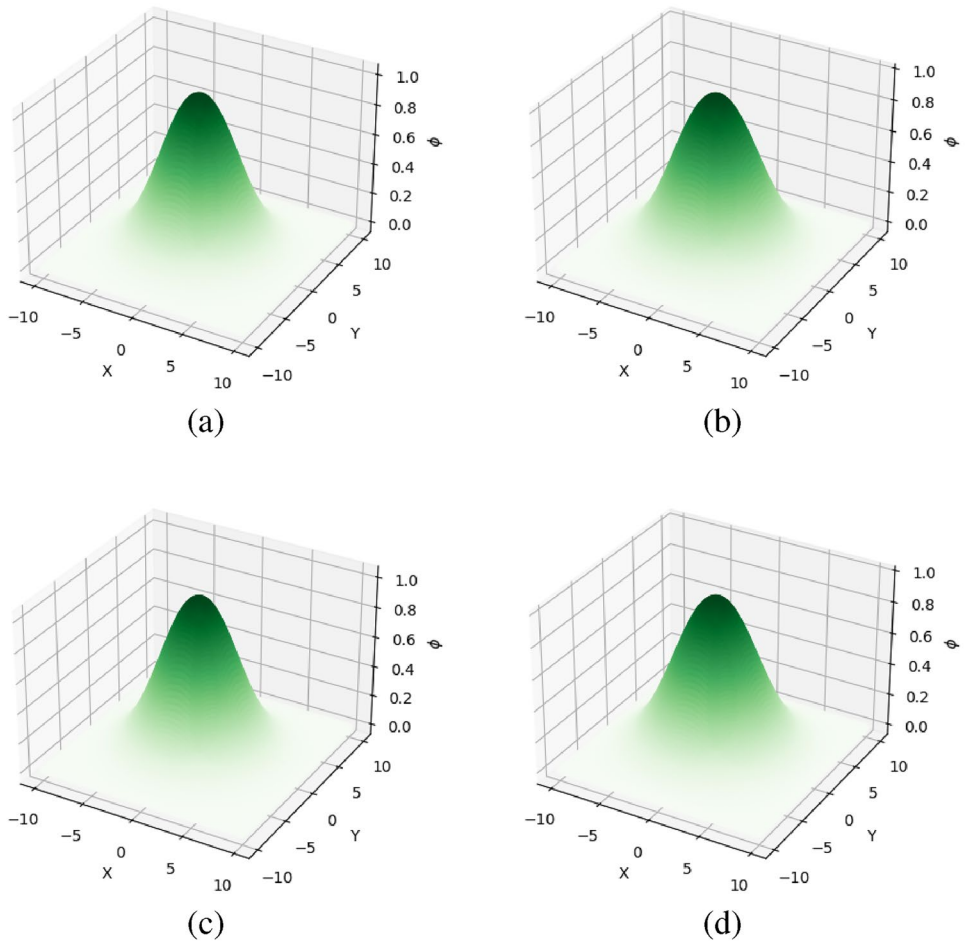


Fig. 19 (Color online) Comparison between result and truth. **a** Prediction result (t=0 s); **b** Prediction result (t=1 s); **c** Truth (t=0 s); **d** Truth (t=1 s)

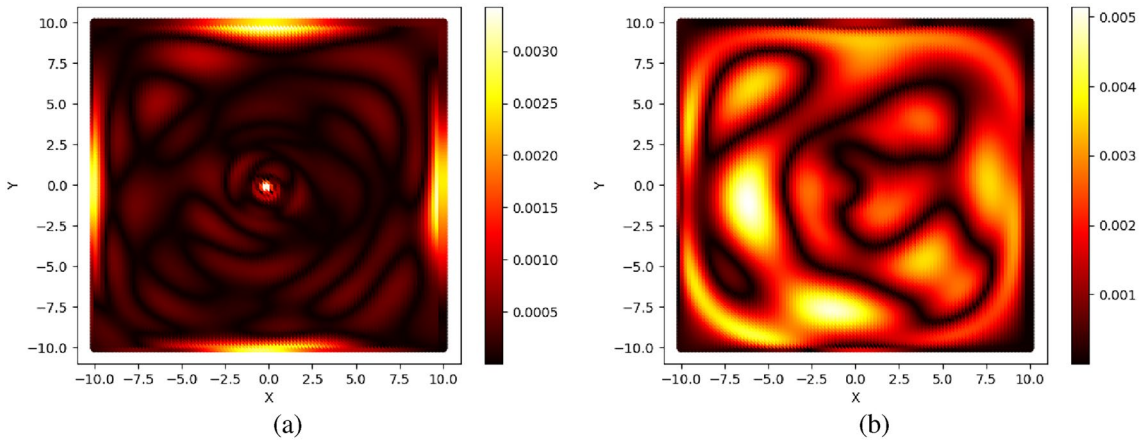


Fig. 20 (Color online) Loss field. **a** $t = 0$ s; **b** $t = 1$ s

relative error of k_{eff} should be less than 0.005 [47]. The predicted results shown in Table 12 satisfy these acceptance criteria. This indicates that R2-PINN also holds practical engineering application value.

4.10 The strategy for hyperparameter selection

The hyperparameter selection strategy was formulated meticulously to balance the model expressiveness, training stability, and prediction accuracy across the computational

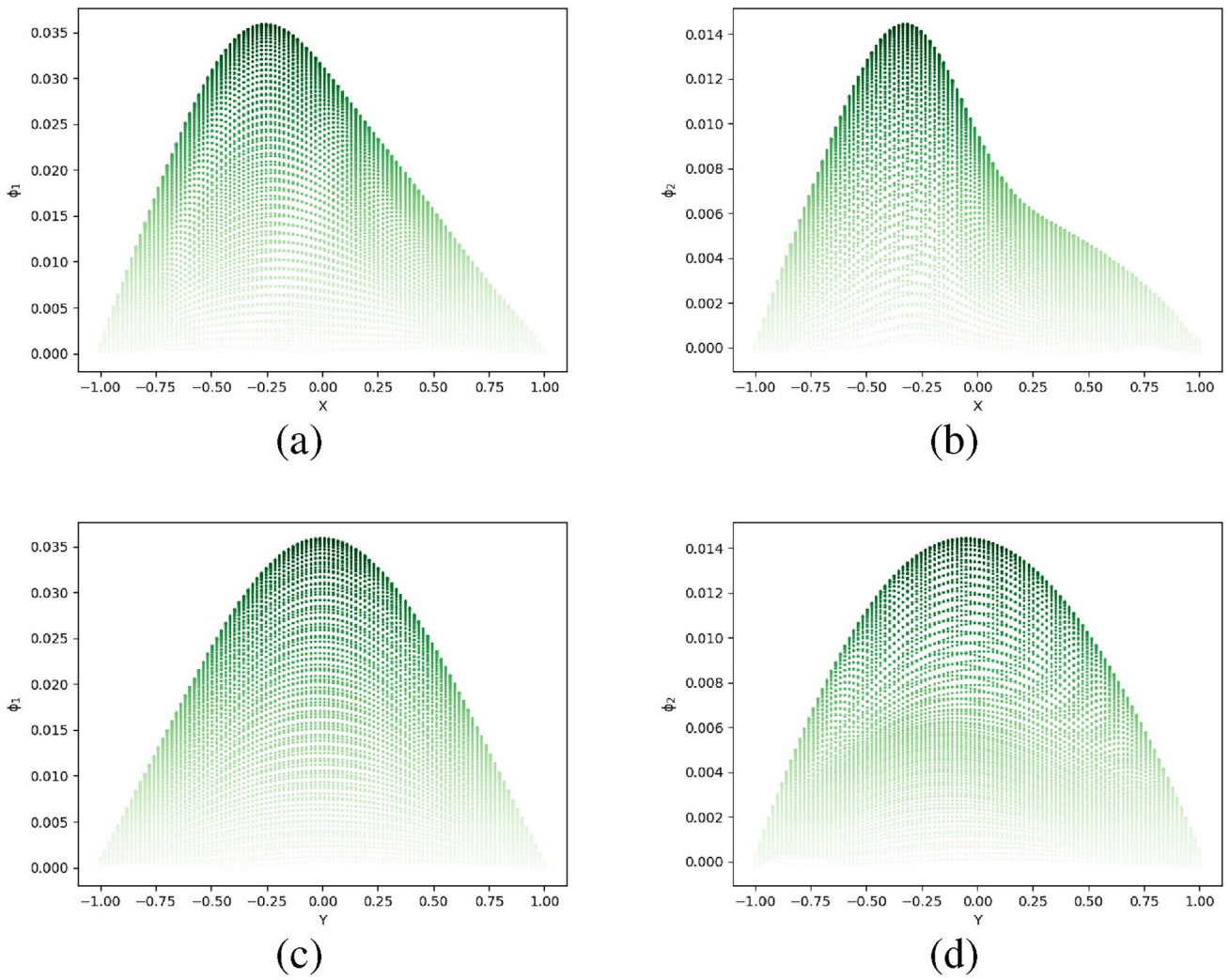


Fig. 21 (Color online) Predict result. **a** ϕ_1 Distribution in x-z plane; **b** ϕ_2 distribution in x-z plane; **c** ϕ_1 Distribution in y-z plane; **d** ϕ_2 distribution in y-z plane

domain. This subsection explains the important hyperparameter selections for neural networks.

First, a 10-layer network was selected to achieve an optimal trade-off between complexity and expressiveness. Compared with shallower networks, deeper networks more effectively approximate complex nonlinear functions. This is typical in physical modeling. As shown in Table 5, our evaluation across configurations indicated that a 10-layer S-CNN structure consistently outperforms the others. This makes it the optimal option. This selection method was similarly applied to different benchmark cases to ensure robustness.

Second, the Tanh activation function was selected over ReLU and Sigmoid because of its smoother nonlinearity, which is critical for approximating continuous functions in physical problems. Although ReLU is effective in mitigating vanishing gradients, it has insufficient stability in capturing smooth physical fields and generally encounters difficulty with highly nonlinear PDEs. In contrast, Tanh enhances numerical stability. This makes it more suitable for PINNs.

Third, the LBFGS optimization algorithm was selected for its faster convergence in high-dimensional problems and capability to prevent gradient explosions. Compared with first-order optimizers such as Adam and SGD, LBFGS provides a second-order approximation. This results in a more stable training and better performance, particularly in data-limited scenarios. During testing, Adam and SGD were vulnerable to early convergence and local minima, thereby yielding suboptimal predictions. Meanwhile, LBFGS provided better stability and training completeness.

Finally, the selection of 26 hidden neurons per layer achieved a balance between the model capacity and generalization. Excessively few neurons result in underfitting, whereas excessively many neurons risk overfitting and

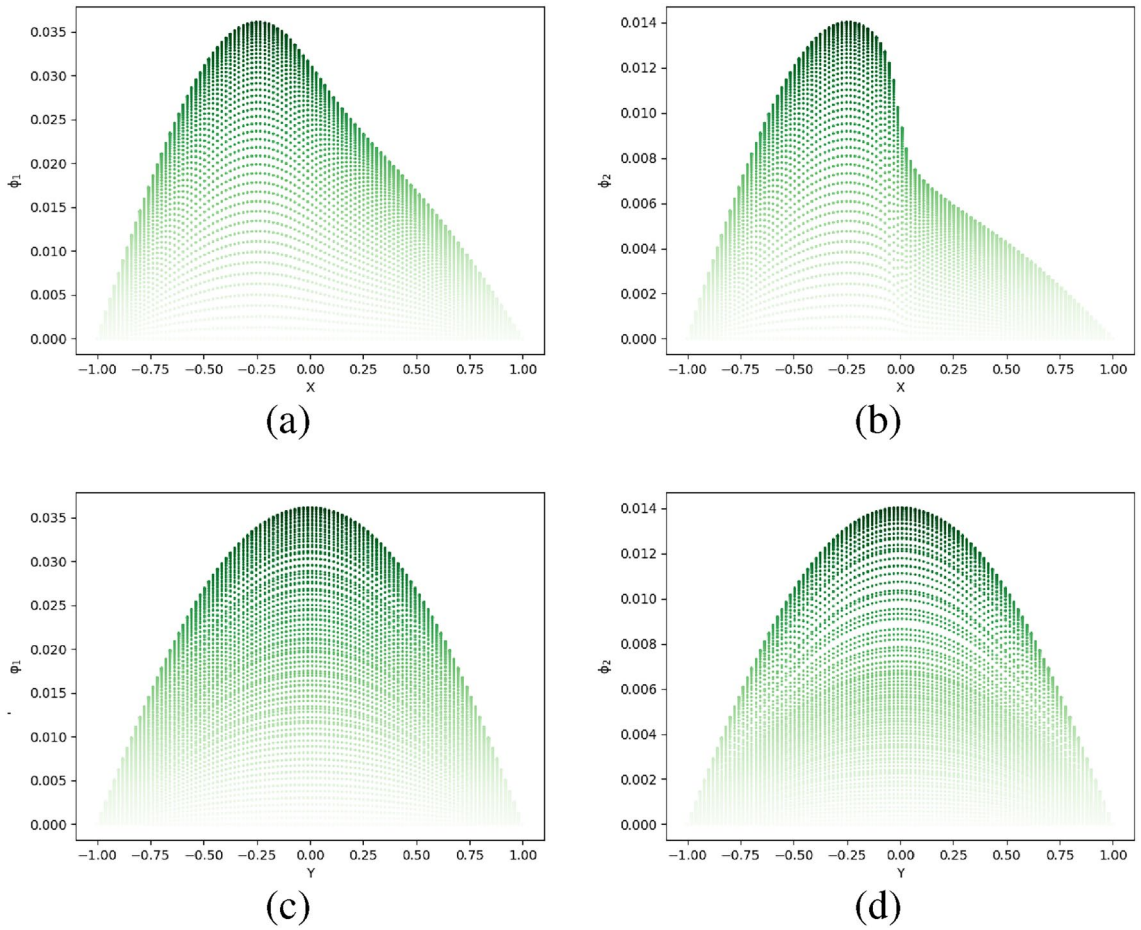


Fig. 22 (Color online) Truth result. **a** ϕ_1 distribution in x-z plane; **b** ϕ_2 distribution in x-z plane; **c** ϕ_1 distribution in y-z plane; **d** ϕ_2 distribution in y-z plane

instability. Through extensive experimentation, 26 neurons per layer were observed to provide an optimal trade-off. This ensured accurate and stable predictions.

5 Results and discussion

An ablation study on the number of layers revealed that an increase in the depth of the S-CNN architecture improved the accuracy. Significantly, a kernel size of one yielded superior results. However, an accuracy bottleneck makes excessively deep layers unnecessary. As described in Sect. 4.2, the S-CNN outperformed the FCN architecture across layer configurations. Even when the number of layers increased to two-fold, no vanishing gradients were observed. The addition of skip connections effectively mitigated the vanishing gradients and enhanced the stability, robustness, and overall accuracy of the model.

In Sect. 4.3, a sensitivity analysis of the resample parameter revealed that the optimal R^2-PINN configuration is a resample granularity of 2 and 500 resamples. Setting the loss weight coefficient w to 100 achieved the best prediction performance, thereby resulting in lower losses than the PDE loss. The test loss in Fig. 12 illustrates a smooth training process without significant fluctuations. At approximately 2000 epochs, the LBFGS optimizer automatically stopped training, thereby achieving an MSE accuracy of 10^{-7}. This indicates a high precision.

Furthermore, k_∞ was determined by searching for the critic state in Sect. 4.4 using the adjusted R^2-PINN. R^2-PINN converged rapidly in 1000 epochs and attained an accuracy of 10^{-8}. The fast convergence and high convergence accuracy of the model indicate that it is suitable for parameter search goals that require the training of multiple networks.

Then, multiple search methods are compared in Sect. 4.5 to improve the search efficiency. From the experimental results, the quadratic fitting search method can rapidly identify k_∞ in only two k-value search processes, with an

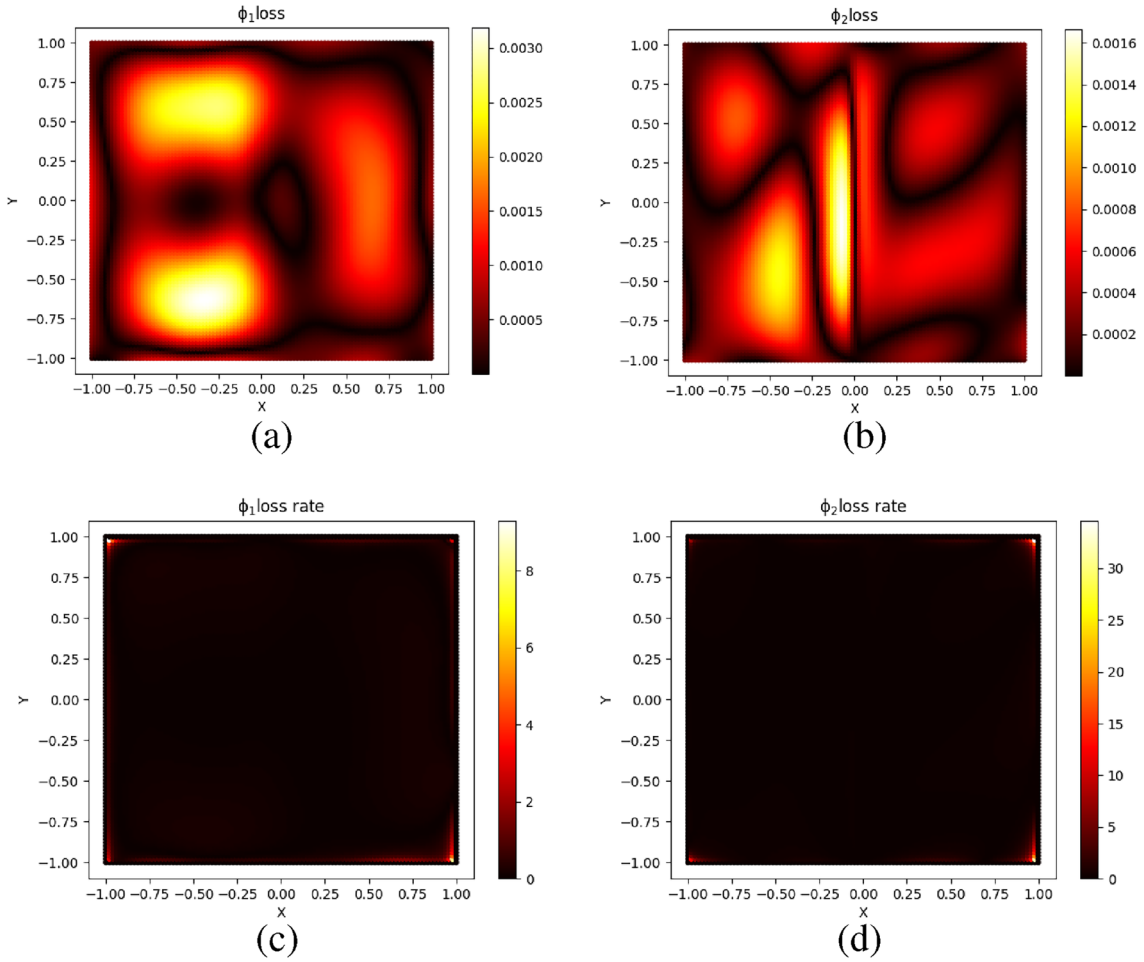


Fig. 23 (Color online) Loss field and loss rate field

accuracy of 10^{-4} in 250 s. This is of significant value for scenarios with high real-time requirements. The grid method can also achieve a parameter search with an accuracy of 10^{-5} within 10 min. When a high accuracy is required, the grid method can set the optimal grid refinement numbers to attain a reasonable duration with a higher accuracy.

The results of experiments conducted at different k_∞ values (described in Sect. 4.6) show that our R^2-PINN network exhibited exceptional performance in capturing the system dynamics within the domain Ω_1. This region exhibited a significant improvement in accuracy compared with FCN networks. By accurately representing the intricate features and sharp gradients of this specific region, our network enables a more precise determination of whether the system is in a steady state. This enhanced accuracy is highly effective when conducting parameter searches because it allows for a higher precision and more reliable results.

Finally, to verify the generalizability of the model, experiments were conducted using the S-CNN architecture for a 2D single cluster (Sect. 4.7). The parameter search error was approximately 1.6%. A solution with an accuracy of e-06 was obtained by using S-CNN to solve 2D multi-cluster multi-materials (Sect. 4.8). The standardized test problem sets–2D-IAEA benchmark for search k_{eff} attained an accuracy of 10^{-4} (Sect. 4.9). These results show that the model can effectively predict the variation in physical quantities in the physical field under multiple equations, different initial conditions, and boundaries. Moreover, it has good generalization under different scenarios.

To summarize, the R^2-PINN network performed better than the FCN network in solving the neutron diffusion equations. The accuracy improvement of one–two points is noteworthy considering the computational efficiency achieved by our framework. This efficiency reduces the computational load and maintains a high accuracy. Thus, it is a potential approach for practical applications. Using a suitable search method, the proposed architecture exhibits a good real-time performance.

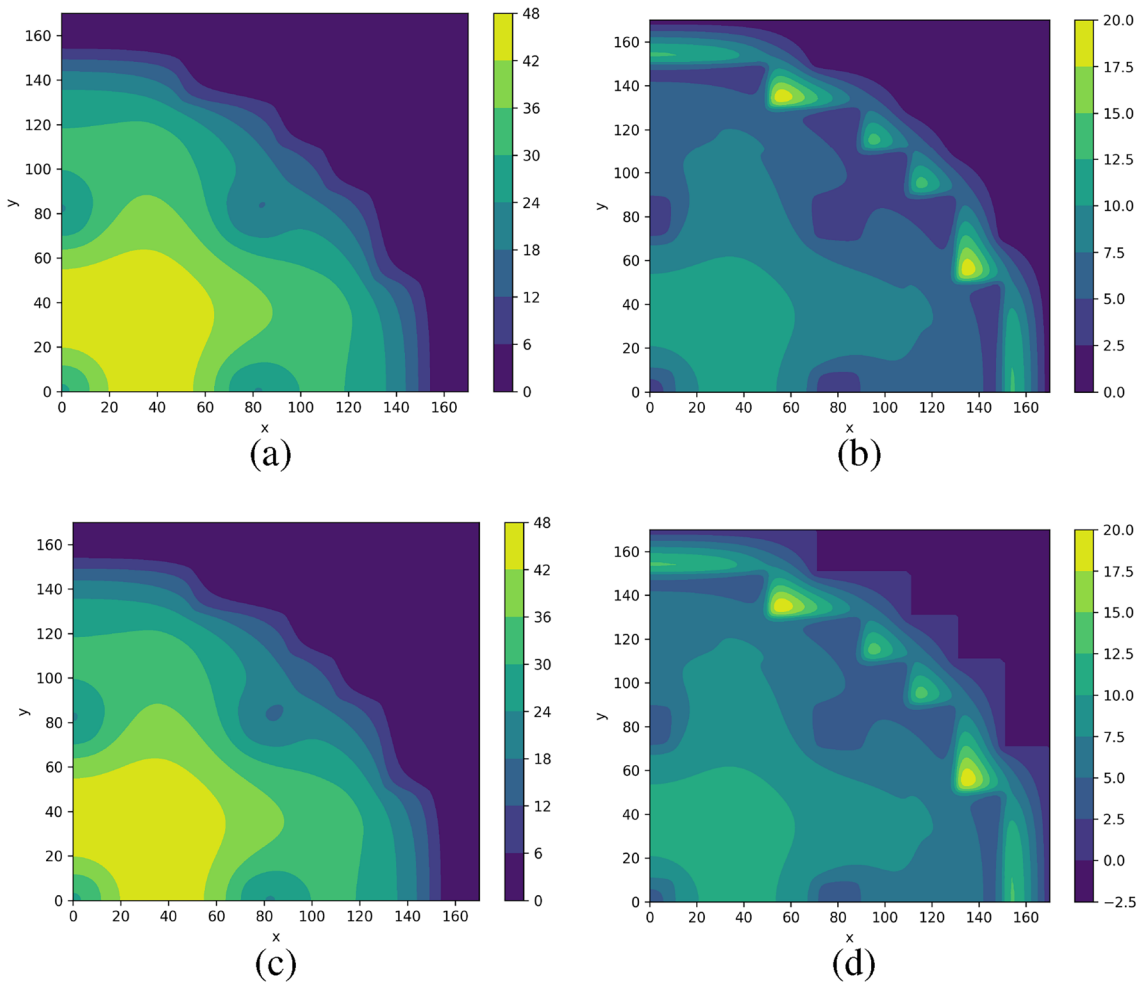


Fig. 24 (Color online) Comparison between results and reference. **a** R^2-PINN result of ϕ_1; **b** R^2-PINN result of ϕ_2; **c** FreeFem++ result of ϕ_1; **d** FreeFem++ result of ϕ_2

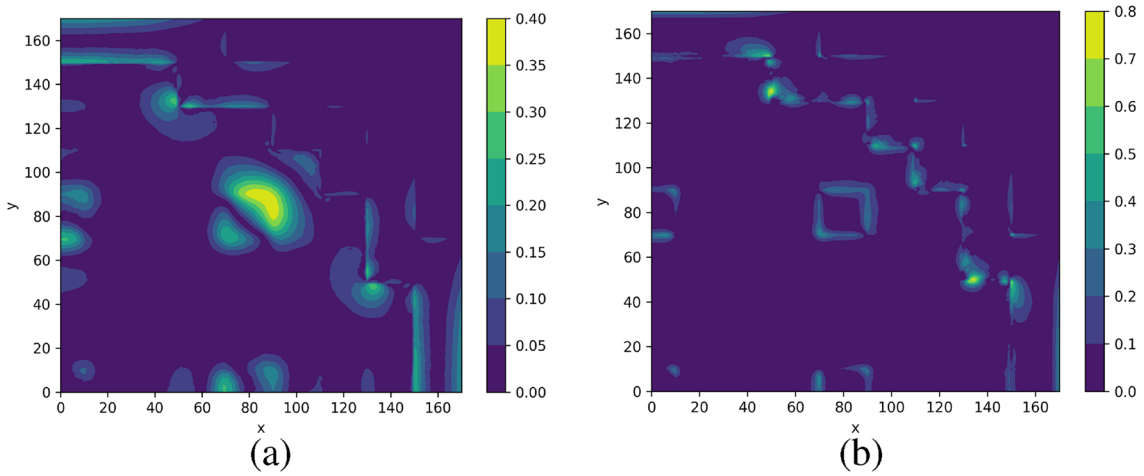


Fig. 25 (Color online) Absolute loss field. **a** Absolute error of ϕ_1; **b** Absolute error of ϕ_2

Table 12 Results of 2D-IAEA benchmark

k_{eff}	e_{r} of k_{eff}	e_{∞} of ϕ_1	e_{∞} of ϕ_2
1.02977	1.797×10^{-4}	0.008	0.038

6 Conclusion

This study introduced a novel and innovative framework called R^2-PINN. It addresses the persistent challenge of the disappearing gradient phenomenon in DNN. In addition, our framework is designed to enhance the accuracy and computational efficiency of PINNs when solving neutron diffusion equations. The R^2-PINN can determine k_{∞} with an accuracy of 10^{-4} in 250 s. The single NN accuracy attained 10^{-8} on an average. This is an order of magnitude higher than that for FCN. The S-CNN architecture is integrated into the R^2-PINN framework to overcome vanishing gradients. By leveraging cross-layer connections, our model effectively learns the residual information. This improves the depth and expressive power of the network. This architectural enhancement ensures that the network can effectively propagate gradients through the layers, thereby enabling more accurate and stable learning. Furthermore, the RAR method is introduced to enhance the representation and sampling strategies within the network. The RAR method allows for adaptive collection of sample points. Thereby, it ensures that the model captures the essential features and gradients in the solution space. This refinement strategy dramatically improves the capacity of the network to handle sharp gradients and intricate features in the PDE solutions.

As described in the experimental section, comprehensive comparative experiments were conducted to optimize the R^2-PINN framework. Through meticulous parameter tuning, including adjusting the number of layers, kernel size, and resampling hyperparameters, R^2-PINN achieved significant accuracy improvements of one–two units compared with FCN. This demonstrated its effectiveness in enhancing PDE solutions. The parameter search capability of the R^2-PINN was validated by efficiently determining the corresponding value of k_{∞} when it entered the critical state within the specified search interval using high-precision network predictions. To evaluate the robustness and accuracy of the model under varying parameters, MSE validation experiments with different values of k_{∞} were conducted. These consistently yielded highly accurate results. Finally, for complex models such as the two-dimensional single-group neutron diffusion equation, two-group two-material neutron diffusion equation models, and 2D-IAEA benchmark, the search for effective value-added coefficients and steady-state flux distribution solutions was conducted successfully.

Overall, the experimental results verify that the integration of S-CNN and the RAR mechanism in R^2-PINN enables the network to capture intricate features and sharp gradients in PDE solutions. The adaptive refinement strategy significantly improves the distribution of residual points. This, in turn, enhances the capability of the network to accurately represent complex physical systems. Our observations provide compelling evidence of the potential for the R^2-PINN to advance the field of deep learning-based PDE solving. Our framework outperforms existing methods and exhibits potential for application in real-world physical systems. The contributions of this study have substantial implications for the development of more accurate and efficient models in various scientific and engineering domains.

Author Contributions All authors contributed to the study conception and design. Heng Zhang contributed to conceptualization, formal analysis, and methodology. Dong Liu was responsible for funding acquisition and project administration. Yun-Ling He handled data curation, investigation, methodology, and visualization. Qin Hang took part in project administration. He-Min Yao participated in investigation. Xiang Di was in charge of Supervision. All authors commented on previous versions of the manuscript. All authors read and approved the final manuscript.

Data Availability The data that support the findings of this study are openly available in Science Data Bank at https://cstr.cn/31253.11.sciencedb.j00186.00825 and https://www.doi.org/10.57760/sciencedb.j00186.00825..

Declarations

Conflict of interest The authors declare that they have no conflict of interest.

References

1. Z.S. Xie, *Physical Analysis of Nuclear Reactors* (Xi'an Jiaotong University Press, Xi'an, 2004), pp. 49–74. (in Chinese)
2. M.N. Özişik, H.R. Orlande, M.J. Colaço et al., *Finite difference methods in heat transfer*, 2nd edn. (CRC press, Inc, 2017)
3. A. Younes, P. Ackerer, F. Delay, Mixed finite elements for solving 2-d diffusion-type equations. Rev. Geophys. **48**, 2008RG000277 (2010). https://doi.org/10.1029/2008RG000277
4. Y.M. Hamada, Higher-order compact finite difference schemes for steady and transient solutions of space-time neutron diffusion model. Ann. Nucl. Energy **175**, 109177 (2022). https://doi.org/10.1016/j.anucene.2022.109177
5. S. Yuk, J. Cho, C. Jo et al., Time-dependent neutron diffusion analysis using finite element method for a block-type vhtr core design. Nucl. Eng. Des. **360**, 110512 (2020). https://doi.org/10.1016/j.nucengdes.2020.110512
6. X.Y. Li, K. Cheng, T. Huang et al., Research on neutron diffusion equation and nuclear thermal coupling method based on gradient updating finite volume method. Ann. Nucl. Energy. **195**, 110158 (2024). https://doi.org/10.1016/j.anucene.2023.110158
7. M.G. Tavares, C.Z. Petersen, M. Schramm et al., Solution for the multigroup neutron space kinetics equations by source iterative method. Braz. J. Radiat. Sci. (2021). https://doi.org/10.15392/bjrs.v9i2A.731

8. K. Zhuang, W. Shang, T. Li et al., Variational nodal method for three-dimensional multigroup neutron diffusion equation based on arbitrary triangular prism. Ann. Nucl. Energy **158**, 108285 (2021). https://doi.org/10.1016/j.anucene.2021.108285
9. Z. Huang, Y. Yuan, G.M. Liu et al., Solution of neutron diffusion problem based on COMSOL multiphysics and its application analysis on micro gas-cooled reactor. Atomic Energy Sci. Technol. (in Chinese) **57**, 565–575 (2023). https://doi.org/10.7538/yzk.2022.youxian.0354
10. K. Sidi-Ali, E.M. Medouri, D. Ailem et al., Neutronic calculations and thermalhydraulic application using CFD for the nuclear research reactor NUR at steady state mode. Prog. Nucl. Energ. **159**, 104640 (2023). https://doi.org/10.1016/j.pnucene.2023.104640
11. Y. Ma, Y.H. Wang, J.H. Yang, ntkFoam: An OpenFOAM based neutron transport kinetics solver for nuclear reactor simulation. Comput. Math. Appl. **81**, 512–531 (2021). https://doi.org/10.1016/j.camwa.2019.09.015
12. G.E. Karniadakis, I.G. Kevrekidis, L. Lu et al., Physics-informed machine learning. Nat. Rev. Phys. **3**, 422–440 (2021). https://doi.org/10.1038/s42254-021-00314-5
13. S. Cuomo, V.S. Di Cola, F. Giampaolo et al., Scientific machine learning through physics-informed neural networks: Where we are and what's next. J. Sci. Comput. **92**, 88 (2022). https://doi.org/10.1007/s10915-022-01939-z
14. M. Raissi, P. Perdikaris, G.E. Karniadakis, 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). https://doi.org/10.1016/j.jcp.2018.10.045
15. S.Z. Cai, Z.P. Mao, Z.C. Wang et al., Physics-informed neural networks (PINNs) for fluid mechanics: a review. Acta. Mech. Sinica. **37**, 1727–1738 (2021). https://doi.org/10.1007/s10409-021-01148-1
16. B.C. Xu, X.Z. Zhang, Y.X. Wang et al., Self-adaptive physical information neural network model for prediction of two-phase flow annulus pressure. Acta. Petrol. Sin. **44**, 545 (2023). https://doi.org/10.7623/syxb202303012
17. S.Z. Cai, Z.C. Wang, S.F. Wang et al., Physics-informed neural networks for heat transfer problems. J. Heat. Transf. **143**(6), 060801 (2021). https://doi.org/10.1115/1.4050542
18. B. Yu, Z.Y. Gan, S.L. Zhang et al., Prediction of 2d/3d unsteady-state temperature fields and heat sources upon the physics-informed neural networks. Engineer. Mechan. **41**, 1–13 (2023). https://doi.org/10.6052/j.issn.1000-4750.2023.04.0282. (in Chinese)
19. Z. Mao, A.D. Jagtap, G.E. Karniadakis, Physics-informed neural networks for high-speed flows. Comput. Method. Appl. M. **360**, 112789 (2020). https://doi.org/10.1016/j.cma.2019.112789
20. C. Chen, G.T. Zhang, Deep learning method based on physics informed neural network with resnet block for solving fluid flow problems. Water **13**(4), 423 (2021). https://doi.org/10.3390/w13040423
21. J.S. Bai, T. Rabczuk, A. Gupta et al., A physics-informed neural network technique based on a modified loss function for computational 2d and 3d solid mechanics. Comput. Mech. **71**, 543–562 (2023). https://doi.org/10.1007/s00466-022-02252-0
22. T.S.J. Feng, W. Liang, The buckling analysis of thin-walled structures based on physics-informed neural networks. Chin. J. Theor. Appl. Mech. **55**(11), 2539–2553 (2023). https://doi.org/10.6052/0459-1879-23-277. (in Chinese)
23. H. Zhang, X. Lyu, D. Liu et al., Nuclear power AI applications: Status, challenges and opportunities. Nuclear Power Eng. **44**(1), 1–8 (2023). https://doi.org/10.13832/j.jnpe.2023.01.0001. (in Chinese)
24. P. Liu, D.F. Shi, R. Li et al., Simulation of core physics benchmark VERA based on Monte Carlo code JMCT. Atomic Energy Sci. Technol. **57**, 1131–1139 (2023). https://doi.org/10.7538/yzk.2022.youxian.0593. (in Chinese)
25. H. Guo, Y.W. Wu, Q.F. Song et al., Development of multi-group Monte-Carlo transport and depletion coupling calculation method and verification with metal-fueled fast reactor. Nucl. Sci. Tech. **34**, 163 (2023). https://doi.org/10.1007/s41365-023-01310-3
26. D.H. Daher, M. Kotb, A.M. Khalaf et al., Simulation of a molten salt fast reactor using the COMSOL Multiphysics software. Nucl. Sci. Tech. **31**, 115 (2020). https://doi.org/10.1007/s41365-020-00833-3
27. Q.H. Yang, Y. Yang, Y.T. Deng et al., Physics-constrained neural network for solving discontinuous interface K-eigenvalue problem with application to reactor physics. Nucl. Sci. Tech. **34**, 161 (2023). https://doi.org/10.1007/s41365-023-01313-0
28. Y.S. Hao, Z. Wu, Y.H. Pu et al., Research on inversion method for complex source-term distributions based on deep neural networks. Nucl. Sci. Tech. **34**, 195 (2023). https://doi.org/10.1007/s41365-023-01327-8
29. D. Liu, Q. Luo, L. Tang et al., Solving multi-dimensional neutron diffusion equation using deep machine learning technology based on pinn model. Nuclear Power Eng. **43**(2), 1–8 (2022). https://doi.org/10.13832/j.jnpe.2022.02.0001. (in Chinese)
30. D. Liu, L. Tang, P. An et al., The deep learning method to search effective multiplication factor of nuclear reactor directly. Nuclear Power Engineering **44**(5), 6–14 (2023). https://doi.org/10.13832/j.jnpe.2023.05.0006. (in Chinese)
31. C. Shen, L. Yan, Recent development of hydrodynamic modeling in heavy-ion collisions. Nucl. Sci. Tech. **31**, 122 (2020). https://doi.org/10.1007/s41365-020-00829-z
32. L. Lu, X.H. Meng, Z.P. Mao et al., Deepxde: A deep learning library for solving differential equations. SIAM Rev. **63**(1), 208–228 (2021). https://doi.org/10.1137/19M1274067
33. L.D. McClenny, U.M. Braga-Neto, Self-adaptive physics-informed neural networks. J. Comput. Phys. **474**, 111722 (2023). https://doi.org/10.1016/j.jcp.2022.111722
34. Z. Wang, H. Xia, S. Zhu et al., Cross-domain fault diagnosis of rotating machinery in nuclear power plant based on improved domain adaptation method. Nucl. Sci. Tech. **59**(1), 67–77 (2022). https://doi.org/10.1080/00223131.2021.1953630
35. Z.K. Lawal, H. Yassin, D.T.C. Lai et al., Physics-informed neural network (pinn) evolution and beyond: a systematic literature review and bibliometric analysis. Big Data Cogn. Comput. **6**(4), 140 (2022). https://doi.org/10.3390/bdcc6040140
36. Z.W. Fang, A high-efficient hybrid physics-informed neural networks based on convolutional neural network. IEEE. T. Neur. Net. Lear. **33**(10), 5514–5526 (2021). https://doi.org/10.1109/TNNLS.2021.3070878
37. X.Y. Yang, Z.X. Zhou, L.H. Li et al., Collaborative robot dynamics with physical human–robot interaction and parameter identification with PINN. Mech. Mach. Theory **189**, 105439 (2023). https://doi.org/10.1016/j.mechmachtheory.2023.105439
38. C.X. Wu, M. Zhu, Q.Y. Tan et al., A comprehensive study of non-adaptive and residual-based adaptive sampling for physics-informed neural networks. Comput. Method. Appl. M. **403**, 115671 (2023). https://doi.org/10.1016/j.cma.2022.115671
39. M.D. McKay, R.J. Beckman, W.J. Conover, A comparison of three methods for selecting values of input variables in the analysis of output from a computer code (JSTOR Abstract). Technometrics **21**(2), 239–245 (1979). https://doi.org/10.2307/1268522
40. M. Liu, L.M. Chen, X.H. Du et al., Activated gradients for deep neural networks. IEEE. T. Neur. Net. Lear. **34**(4), 2156–2168 (2023). https://doi.org/10.1109/TNNLS.2021.3106044
41. M.W. Stacey, *Nuclear Reactor Physics*, 2nd edn. (Wiley, Weinheim, 2010), pp.43–60

42. J.J. Duderstadt, *Nuclear Reactor Analysis* (Wiley, New York, 1976), pp.74–88
43. N. None, *Argonne Code Center: Benchmark problem book* (Argonne National Lab.(ANL), Argonne, 1977), pp.277–284
44. K.M. He, X.Y. Zhang, S.Q. Ren et al., Deep residual learning for image recognition. 2016 Proc. CVPR. IEEE. Las Vegas, USA 2016, pp. 770–778 (2016). https://doi.org/10.1109/CVPR.2016.90
45. F. Hecht, New development in FreeFem++. J. Numer. Math. **20**(3–4), 251–265 (2012). https://doi.org/10.1515/jnum-2012-0013
46. R.B. Lehoucq, D.C. Sorensen, C. Yang, *ARPACK Users' Guide: Solution of Large-Scale Eigenvalue Problems with Implicitly Restarted Arnoldi Methods* (SIAM, 1998), pp.21–40. https://doi.org/10.1137/1.9780898719628
47. Y. Yang, H. Gong, S. Zhang et al., A data-enabled physics-informed neural network with comprehensive numerical study on solving neutron diffusion eigenvalue problems. Ann. Nucl. Energy **183**, 109656 (2023). https://doi.org/10.1016/j.anucene.2022.109656