Nuclear Science and Techniques (2026) 37:23
https://doi.org/10.1007/s41365-025-01836-8

Physics-informed neural network with equation adaption for ^{220}Rn progeny concentration prediction

Shao-Hua Hu[1] · Qi Qiu[2] · De-Tao Xiao[1] · Xiang-Yuan Deng[1] · Xiang-Yu Xu[1] · Peng-Hao Fan[1] · Lei Dai[1] · Zhi-Wen Hu[3] · Tao Zhu[2] · Qing-Zhi Zhou[1]

Received: 28 May 2024 / Revised: 22 August 2024 / Accepted: 31 August 2024 / Published online: 3 January 2026

Abstract

Physics-informed neural networks (PINNs) are vital for machine learning and exhibit significant advantages when handling complex physical problems. The PINN method can rapidly predict ^{220}Rn progeny concentration and is very important for regulating and measuring this property. To construct a PINN model, training data are typically preprocessed; however, this approach changes the physical characteristics of the data, with the preprocessed data potentially no longer directly conforming to the original physical equations. As a result, the original physical equations cannot be directly employed in the PINN. Consequently, an effective method for transforming physical equations is crucial for accurately constraining PINNs to model the ^{220}Rn progeny concentration prediction. This study presents an equation adaptation approach for neural networks, which is designed to improve prediction of ^{220}Rn progeny concentration. Five neural network models based on three architectures are established: a classical network, a physics-informed network without equation adaptation, and a physics-informed network with equation adaptation. The transport equation of the ^{220}Rn progeny concentration is transformed via equation adaption and integrated with the PINN model. The compatibility and robustness of the model with equation adaption is then analyzed. The results show that PINNs with equation adaption converge consistently with classical neural networks in terms of the training and validation loss and achieve the same level of prediction accuracy. This outcome indicates that the proposed method can be integrated into the neural network architecture. Moreover, the prediction performance of classical neural networks declines significantly when interference data are encountered, whereas the PINNs with equation adaption exhibit stable prediction accuracy. This performance demonstrates that the proposed method successfully harnesses the constraining power of physical equations, significantly enhancing the robustness of the resultant PINN models. Thus, the use of a physics-informed network with equation adaption can guarantee accurate prediction of ^{220}Rn progeny concentration.

Keywords Machine learning · Physics-informed neural networks · Equation adaption · ^{220} Rn progeny

1 Introduction

Deep learning has profoundly impacted many areas of modern society [1–4], with significant applications in the fields of image recognition [1], natural language processing [2], cognitive science [3], and genomics [3, 4]. As a core technology of machine learning, neural networks play key roles in those fields. However, traditional neural network methods require considerable volumes of training data when analyzing complex physical, biological, or engineering systems. In complex, specialized cases, the cost of data collection is often high, and uncertainties exist regarding the data collection accuracy; these problems pose many challenges for deep learning applications [5]. Further, when supplied with

This work was supported by the National Natural Science Foundation of China (Nos.12375310, 118750356, and 62006110) and Graduate Research and Innovation Projects of Hunan Province (CX20230964).

✉ Tao Zhu
tzhu@usc.edu.cn

✉ Qing-Zhi Zhou
zhouqingzhi2005@163.com

[1] School of Nuclear Science and Technology, University of South China, Hengyang 421001, China

[2] School of Computer/School of Software, University of South China, Hengyang 421001, China

[3] College of Computer Science and Technology, Hengyang Normal University, Hengyang 421002, China

only partial datasets, most advanced machine learning techniques lack robustness and cannot draw reliable conclusions and make decisions. In recent years, a new deep neural network (DNN) framework, physics-informed neural networks (PINNs), has been developed. A PINN incorporates physical laws into a neural network (e.g., an artificial, recurrent, or convolutional neural network (ANN, RNN or CNN, respectively)) and constitutes a data-physics dual-drive approach. This feature differentiates PINNs from traditional neural networks, which rely solely on data-driven methods [1, 5]. That is, through the use of physical information as prior knowledge, PINNs can be trained with very few or no labeled data as alternative models for accurately solving partial differential equations [5, 6], while also incorporating complex physical laws (that are difficult to describe via theoretical equations) in data form. Thus, PINNs have physics- and data-driven components.

The main traditional neural network models are ANNs, RNNs, and CNNs [5]. Many advanced algorithms have been derived to optimize the performance of these models, such as TRANSFORM-ANN [7], which can simultaneously fine-tune the neural network architecture, adjust the training dataset size, and select the appropriate activation function. To mitigate the risk of overfitting, TRANSFORM-ANN integrates three strategies for determining the training set size, all of which are based on Sobol sampling. TRANSFORM-ANN is suitable for construction of an accurate and concise ANN model. However, limitations exist. Because TRANSFORM-ANN employs multi-objective optimization, it has a high computational cost, especially when applied to high-dimensional datasets, for which the computational complexity increases significantly. In addition to TRANSFORM-ANN, progressive neural architecture search [8] and one-shot neural architecture search (OSNAS) [9] are important methods in the field of model optimization. Boundary-integrated neural networks (BINNs) [10] are among the networks similar to the PINN model. A BINN is a numerical method combining a boundary integral equation (BIE) and neural network and is employed to solve acoustic radiation and scattering problems efficiently and accurately. A BINN requires boundary-node information only as input, which greatly reduces the calculation cost and makes this approach particularly suitable for infinite domain problems. The semi-analytical characteristics of the BIE improves the prediction accuracy of the BINN. However, the application of BINNs to more challenging problems, such as those arising in the fields of high-frequency and nonlinear acoustics and complex geometry, requires further exploration.

The training data supplied to neural networks usually span physical quantities with multiple dimensions, which often exhibit significantly different orders of magnitude; therefore, appropriate data preprocessing is crucial [11]. For example, the physical quantities encountered in the fields of medical microdosimetry [12] and radioactive detection [13] may span hundreds of orders of magnitude or beyond. In those applications, even datasets with narrower ranges contain key information. To improve the sensitivity of neural networks to data with wide ranges of orders of magnitude and ensure that models can fully capture the key information in those data, the data must be processed appropriately using preprocessing functions. For example, logarithmic functions can be used to scale data effectively to an appropriate range. Overall, preprocessing functions are essential and universal for neural network training, but their application changes the dimensions of various features of the training data [1, 14]. Such alterations adversely affect PINN networks. That is, the key concept of the PINN network is that physical equations are combined to guide the training process of the neural network, ensuring that the model predictions not only conform to the data distribution but also follow specific physical laws. When the preprocessing function changes the dimensions of the data features, the physical equations in the PINN network are no longer directly effective, because the equation parameters and variables are usually closely related to the dimensions of the original data.

An ^{220}Rn chamber is an essential scientific device for accurately measuring the radiation dose levels of ^{220}Rn and its progeny [15–17]. The exhaust pipe is a core component of this device. When the concentrations of ^{220}Rn and its progeny in the ^{220}Rn chamber must be reduced, clean air is often injected into the chamber to dilute the indoor radioactive gas concentration. The excess radioactive gas is released into the atmosphere via the exhaust pipe. The emitted radioactive gas concentration can reach several thousand becquerel per meter cubed; thus, the concentration distribution of the emitted radioactive gas must be monitored effectively. To achieve precise control and accurate measurement of ^{220}Rn progeny concentration [15], a rapid prediction model must be established; this is feasible using the PINN approach. However, when a ^{220}Rn concentration prediction model is developed using training data subjected to a preprocessing function, the preprocessing function alters the characteristics of physical quantities such as time, space, and concentration in the training data. To achieve normal functionality of the physical equations in the PINN network, the physical equations must be deformed according to specific preprocessing functions.

Since PINNs were proposed in 2019 [18], these methods have been applied to various fields. In fluid mechanics [19–25], PINNs have proven to be a valuable tool for overcoming the limitations of traditional numerical simulation methods, particularly for noisy data, complex grid generation challenges, and high-dimensional flow problems. In medical diagnostics [26, 27], PINNs precisely simulate biomechanics and biofluid mechanics, elucidating

complex biological fluid phenomena and aiding disease diagnosis, treatment optimization, and medical device design. In materials science [28–35], PINNs have greatly enhanced the prediction accuracy of key physical quantities such as material stress and strain, especially in cases with limited data resources. In the power industry [36–41], PINNs have been used for power system optimization and stability analysis, combining physical laws with data analysis to accurately predict system behavior, optimize energy distribution, enhance grid stability, and improve overall energy efficiency. These applications demonstrate the excellent adaptability and reliability of PINNs. However, in such previous studies, conventional data processing methods were generally adopted and the issue of physical equation deformation and incorporation into the neural network after preprocessing was not explored. In particular, to achieve effective ^{220}Rn progeny concentration prediction, the development of a PINN with equation adaption is highly significant.

This study introduces an equation adaption approach for neural networks, which can accurately deform physical equations for application in PINN model training. The compatibility of this method with neural networks and the robustness of the resultant model are explored. The remainder of this paper is organized as follows. In Sect. 2, five neural network models are established based on three architectures: a classical network, a physics-informed network without equation adaptation, and a physics-informed network with equation adaptation. The equation adaption process is then applied and the deformation of the physical equations based on specific preprocessing functions is demonstrated. Section 3 focuses on the compatibility of the neural network equation adaption approach and the robustness of the PINN network after equation adaption, based on the five models established previously. Section 4 concludes the work.

2 Methodology

When training data are processed by a preprocessing function, the physical equations must be transformed before they can be incorporated into the PINN architecture. This section first establishes five prediction models for ^{220}Rn concentration based on three network architectures. Then, the proposed equation adaptation method is introduced, with the equations being incorporated into the PINN model.

2.1 Physical object

In this study, the exhaust pipe of the ^{220}Rn chamber was taken as the research object and the concentration distribution of the emitted radioactive gas was predicted. The device was cylindrical, with a diameter of $\Phi = 10$ cm and a length L of 40 cm (Fig. 1). The gas entered through the inlet and exited through the outlet, and primarily comprised a mixture of ^{220}Rn progeny and air. The inlet wind speed was taken as the model boundary condition, and the wind speed adjustment range was $0 - 0.1$ m/s [15]. The main decay products of ^{220}Rn are ^{216}Po, ^{212}Pb, and ^{212}Bi. Because ^{216}Po has a short half-life of only 0.145 s, its migration and diffusion capabilities are minimal. In contrast, the half-lives of the latter two are more prolonged, at 10.64 h and 50.55 min, respectively, and their migration and diffusion are more impactful. Thus, ^{212}Pb, and ^{212}Bi are the focus of research attention [15, 16]. As ^{212}Pb and ^{212}Bi exhibit highly similar migration and diffusion patterns in this context, in this study, only the ^{212}Pb concentration distribution was considered.

2.2 Establishment of neural network

Computational fluid dynamics (CFD) was used to establish a numerical simulation of the physical object (i.e., the exhaust pipe) discussed in Sect. 2.1; hence, the database needed to train the neural network was obtained. Note that this database can be used to train and validate subsequent neural network models. The physical equations were incorporated into the loss function to jointly constrain the training of the neural network model. Finally, the data and physical equation loss function were used in combination to train the

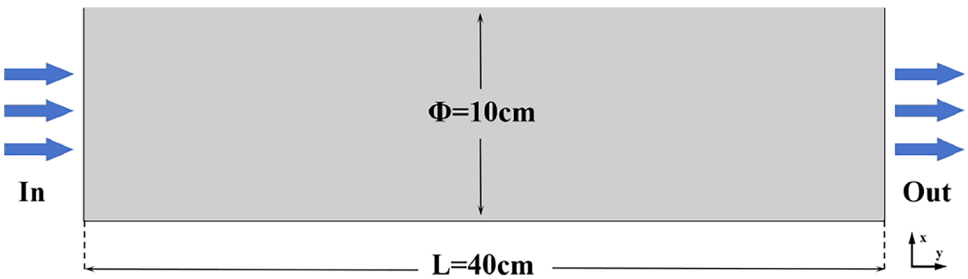

Fig. 1 (Color online) Exhaust pipeline structure

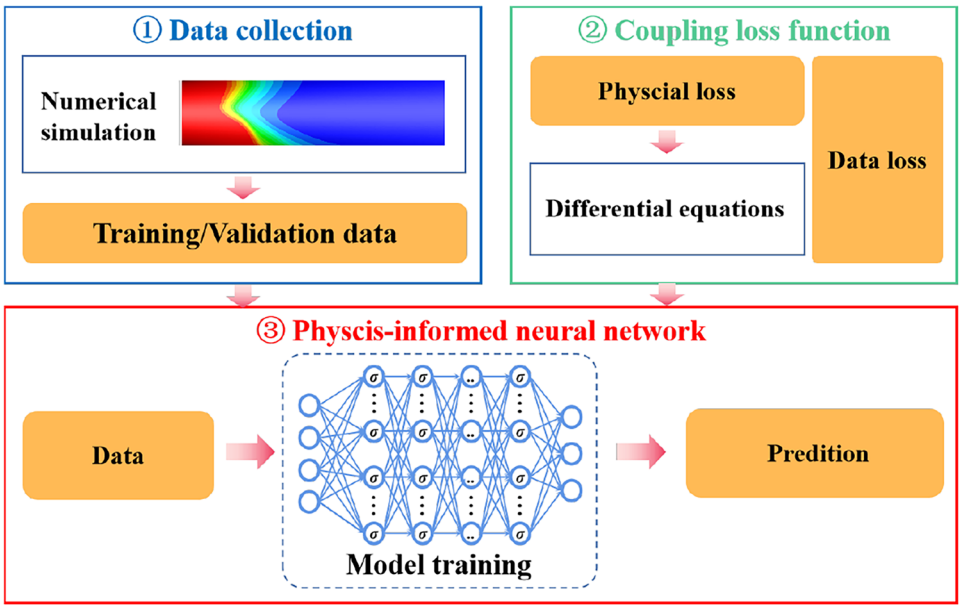

Fig. 2 (Color online) Framework for establishing neural network model

neural network model. The model construction flowchart is shown in Fig. 2.

The physical structure considered in this study was a cylinder, which is highly symmetrical. Therefore, in experiment, only the ^{212}Pb concentration distribution in the *xoy* plane area is often required. In addition, because the physical structure is highly symmetrical and the fluid velocity is minimal, the Reynolds number is far less than 2000; thus, a laminar flow is established, which is characterized by a linear and uniform velocity distribution with a stable pressure gradient. Considerable similarity exists between the two- and three-dimensional flows; that is, the characteristics of a three-dimensional flow field can be described by a two-dimensional simulation. Thus, the difficulty of establishing a three-dimensional model can be avoided [42]. Therefore, the neural network model established in this study was designed to predict the ^{212}Pb concentration distribution on the *xoy* plane.

(1) Data collection PINN models exhibit high robustness and can effectively handle data with significant errors [43, 44]. In this study, to test the robustness of the PINN model established using the proposed method, two databases were established: one without and one with interference (labeled Data-01 and Data-02, respectively). Data-02 comprised Data-01 with the addition of a small volume of data containing significant errors. The databases contained ^{212}Pb concentrations spanning an extensive range, from 10^{-50} Bq/m^3 to 10^3 Bq/m^3. Data-01 comprised N_{data} = 987,135 normal data points, whereas Data-02 comprised N_{data} = 987,135 normal data points and N_{error} = 50 erroneous data points. A random 0.2 % of the data from Data-01 was selected as the validation set ($N_{\text{validation}}$ = 2,000 data points), labeled "data-validation." Finally, these two databases were used to separately train three networks: NN, PINN-EA, and PINN-f. Hence, five models were obtained: NN, NN-ERR, PINN-EA, PINN-EA-ERR, and PINN-f. The correspondence between the different training databases and training models is shown in Fig. 3. NN, PINN-EA, and PINN-f are introduced in detail in the following subsection. Note that Data-02 had 50 additional data points compared to Data-01, accounting for 0.005% of the total data, which was a tiny proportion. Therefore, the data volumes of Data-02 and Data-01 were considered identical, and the influence of the additional 50 data points on the model training was ignored.

(2) Neural networks A PINN is essentially a DNN that can approximate a solution determined from data and PDEs [45]; its architecture is shown in Fig. 4. In this study, three neural networks were constructed: NN, PINN-EA, and PINN-f. NN was a classical neural network without physical laws, whereas PINN-EA and PINN-f were PINNs with physical laws. PINN-EA and PINN-f differed in terms of their preprocessing functions. The equation adaption approach proposed in this study was adopted for PINN-EA; that is, the physical law $F(X, Y, U, V, C)$ was used in the network. No preprocessing function was used for PINN-f, with the physical law $f(x, y, u, v, c)$ being directly integrated into the network. The physical laws $F(X, Y, U, V, C)$ and $f(x, y, u, v, c)$ are explained in detail in Sect. 2.3.

A residual neural network [46, 47] was adopted in this study, for which the relationship between the inputs and outputs can be expressed as

$$(u, v, c) = \mathcal{T}_{\text{NN}}(x, y, t, q; \boldsymbol{\Theta}). \tag{1}$$

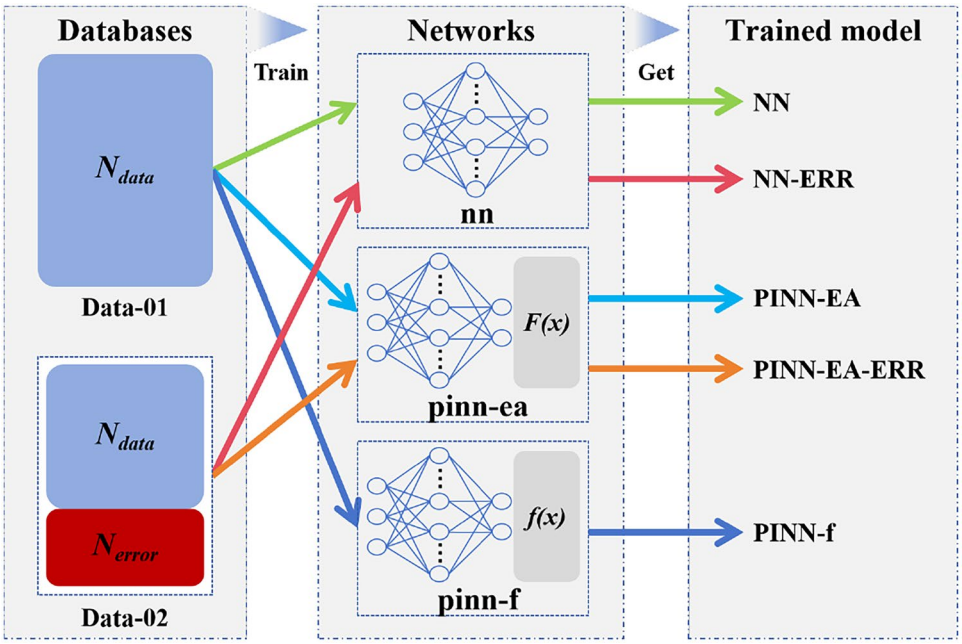

Fig. 3 (Color online) Correspondence between training databases and trained models

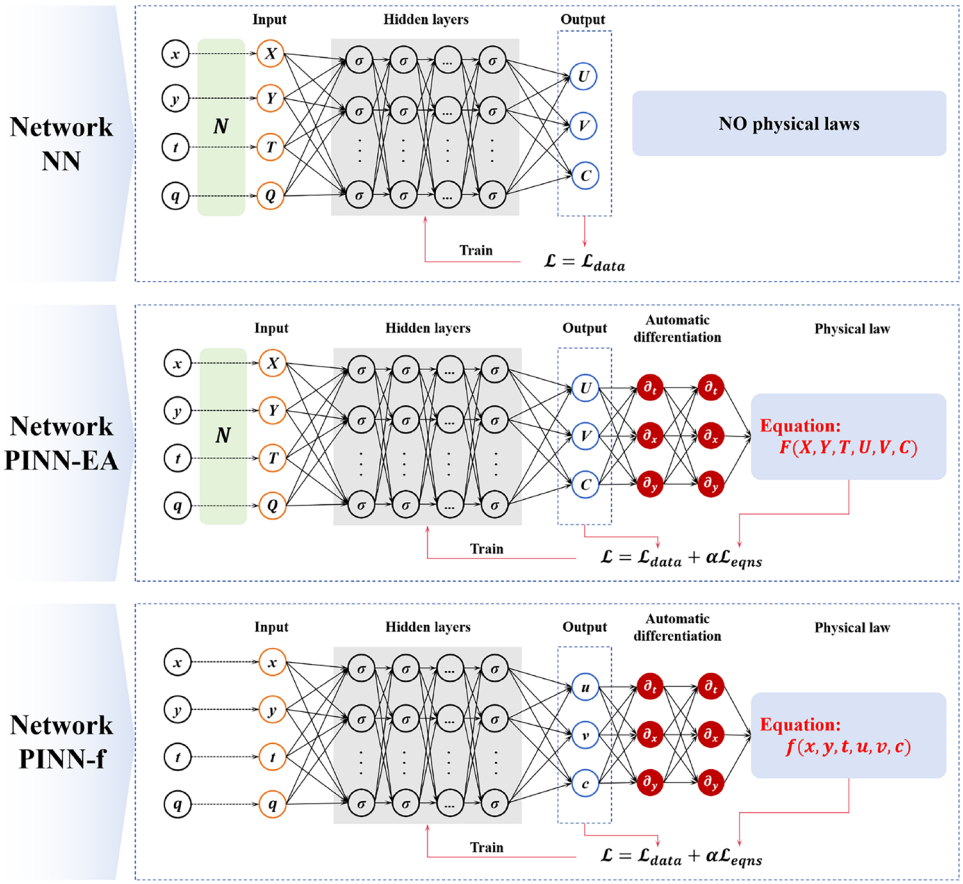

Fig. 4 (Color online) NN, PINN-EA and PINN-f architectures

Here, $\mathcal{T}_{\mathrm{NN}}$ represents the neural network, the inputs of which are the space coordinates (x and y), time (t), and wind speed (q). The neural network outputs are the velocity vectors (u and v) and concentration (c). The parameter $\boldsymbol{\Theta}$ represents the trainable variables. Through this network, the relationship

between the inputs and outputs is constructed. The k-th hidden layer for the residual neural network is expressed as

$$H^k = \sigma(H^{k-1}W^{k-1} + b^{k-1}), \tag{2}$$

where W and b are the weights and biases, respectively; H represents the output of each hidden layer; and σ is the activation function.

In this work, the partial derivatives ∂_x, ∂_y, and ∂_t were computed based on the chain rule, which has previously been implemented via automatic differentiation in both TensorFlow and PyTorch [45]. In this study, PyTorch was employed for this computation. Note that higher-order derivatives can be estimated through multiple calling of this function. Based on the ^{220}Rn progeny transport equation, which is detailed in Eqs. (9) and (18) of Sect. 2.3, the residual equation of the transport equation can be derived as follows:

$$\begin{aligned} e^{\mathrm{f}} = &\frac{\partial c}{\partial t} + \left(\frac{\partial uc}{\partial x} + \frac{\partial vc}{\partial y}\right) \\ &- D_{\mathrm{e}}\left(\frac{\partial^2}{\partial x^2} + \frac{\partial^2}{\partial y^2}\right)c + \lambda c, \end{aligned} \tag{3}$$

$$e^{\mathrm{F}} = A' + B'' - E'' + G', \tag{4}$$

where e^{f} and e^{F} pertain to PINN-f and PINN-F, respectively. Further, D_{e} is the diffusion coefficient, $D_{\mathrm{e}} = 2.88 \times 10^{-5}\ \mathrm{m^2/s}$ [15], and λ is the decay constant of ^{212}Pb. To ensure the network adheres to the transport equations, the loss function within the PINN is defined as follows:

$$\mathcal{L} = \mathcal{L}_{\mathrm{data}} + \alpha\mathcal{L}_{\mathrm{eqns}}, \tag{5}$$

where α is a weighting coefficient, the value of which is discussed in Sect. 3.3. Further, $\mathcal{L}_{\mathrm{data}}$ and $\mathcal{L}_{\mathrm{eqns}}$ are computed as

$$\begin{aligned} \mathcal{L}_{\mathrm{data}} = &\frac{1}{N_{\mathrm{data}}}\sum_{i=1}^{N_{\mathrm{data}}}\{\left|u^i_{\mathrm{data}} - u^i_{\mathrm{pred}}\right| \\ &+ \left|v^i_{\mathrm{data}} - v^i_{\mathrm{pred}}\right| + \left|c^i_{\mathrm{data}} - c^i_{\mathrm{pred}}\right|\}, \end{aligned} \tag{6}$$

$$\mathcal{L}_{\mathrm{eqns}} = \frac{1}{N_{\mathrm{data}}}\sum_{\mathrm{i}=1}^{N_{\mathrm{data}}}|e_i|. \tag{7}$$

Here, u^i_{data}, v^i_{data}, and c^i_{data} are the measured data; u^i_{pred}, v^i_{pred}, and c^i_{pred} are the predicted data; $\mathcal{L}_{\mathrm{data}}$ represents the loss between the measured and predicted data; and e represents e^{f} or e^{F}. Recall that, in PINN-F, $e = e^{\mathrm{F}}$, whereas in PINN-f, $e = e^{\mathrm{f}}$. The variables $\boldsymbol{\Theta}$ are optimized by minimizing the loss function. Here, the training variables were updated using the adaptive moment estimation (Adam) optimizer with an initial learning rate of 0.001. The learning rate decreased stepwise every 40 epochs to 0.9 times the original learning rate. Five models were obtained by training the three neural networks with different databases. The hyperparameter settings of the five models are detailed in Table 1. Note that the specified number of iterations was sufficient to decrease the model training error to a stable condition. The numbers of hidden units and layers (N_{cell} and N_{layer}, respectively) were determined as discussed in Sect. 3.3.

(3) Activation functions The activation function is pivotal to the neural network ability to approximate data. Without an activation function, the network would perform linear transformations [45]. Given that a PINN incorporates a derivation process, selection of an appropriate activation function is essential for practical model training. Five representative activation functions were used to construct and optimize our model training process in this study: Sigmoid, Tanh, ReLU, Leaky ReLU, and Hardswish. These activation functions exhibit unique nonlinear mapping characteristics, such as a sharp contrast between the saturated (Sigmoid, Tanh) and unsaturated (ReLU, Leaky ReLU, Hardswish) types, and also span the diversity of parameterized (e.g., the negative slope parameter of Leaky ReLU) and nonparametric design. These five activation functions are representative and widely used in the field of neural networks [48, 49].

2.3 Equation adaption

The migration and diffusion behaviors of ^{212}Pb within the device follow the transport equation. That is,

$$\frac{\partial c}{\partial t} + \left(\frac{\partial uc}{\partial x} + \frac{\partial vc}{\partial y}\right) = D_{\mathrm{e}}\left(\frac{\partial^2}{\partial x^2} + \frac{\partial^2}{\partial y^2}\right)c - \lambda c, \tag{8}$$

where x and y are spatial coordinates; u and v are the flow field velocities at these coordinates; c is the ^{212}Pb concentration at these coordinates; D_{e} is the diffusion coefficient; and λ is the ^{212}Pb decay constant. First, the $f(x, y, u, v, c)$ expression is established:

$$\begin{aligned} f(x,y,t,u,v,c) = &\frac{\partial c}{\partial t} + \left(\frac{\partial uc}{\partial x} + \frac{\partial vc}{\partial y}\right) \\ &- D_{\mathrm{e}}\left(\frac{\partial^2 c}{\partial x^2} + \frac{\partial^2 c}{\partial y^2}\right) + \lambda c. \end{aligned} \tag{9}$$

Here, the preprocessing function and its inverse are represented by N and FN, respectively. Further, X, Y, T, Q, U, V, and C are the parameters obtained by transforming x,

y, t, q, u, v, and c through the preprocessing function. Their relationships can be expressed as

$$\begin{aligned} x &= FN_x(X) \\ y &= FN_y(Y) \\ t &= FN_t(T) \\ q &= FN_t(Q) \\ u &= FN_u(U) \\ v &= FN_v(V) \\ c &= FN_c(C) \end{aligned} \tag{10}$$

From Eq. (10), the physical Eq. (9) can be transformed as follows:

$$\begin{aligned} F(X,Y,T,U,V,C) = & \frac{\partial FN_c(C)}{\partial FN_t(T)} \\ & + \left(\frac{\partial FN_u(U)FN_c(C)}{\partial FN_x(X)} + \frac{\partial FN_v(V)FN_c(C)}{\partial FN_y(Y)} \right) \\ & - D_e \left(\frac{\partial^2}{\partial FN_x(X)^2} + \frac{\partial^2}{\partial FN_y(Y)^2} \right) FN_c(C) \\ & + \lambda FN_c(C) \end{aligned} \tag{11}$$

To improve the transparency of the subsequent equation derivation, Eq. (11) is decomposed into four parts: A, B, E, and G:

$$\begin{aligned} A &= \frac{\partial FN_c(C)}{\partial FN_t(T)} \\ B &= \frac{\partial FN_u(U)FN_c(C)}{\partial FN_x(X)} + \frac{\partial FN_v(V)FN_c(C)}{\partial FN_y(Y)} \\ E &= D_e\left(\frac{\partial^2}{\partial FN_x(X)^2} + \frac{\partial^2}{\partial FN_y(Y)^2} \right) \\ G &= \lambda FN_c(C) \end{aligned} \tag{12}$$

Then, Eq. (11) can be transformed into the following form:

$$F(X,Y,T,U,V,C) = A + B - E + G. \tag{13}$$

By applying the chain rule for the differentiation of composite functions [50], Eq. (12) can be expanded as follows:

$$\begin{aligned} A &= \frac{\partial FN_c(C)}{\partial C} \frac{\partial C}{\partial T} \frac{\partial T}{\partial FN_t(T)} \\ B &= \left[\frac{\partial FN_u(U)}{\partial U} \frac{\partial U}{\partial X} \frac{\partial X}{\partial FN_x(X)} \right. \\ &\quad \left. + \frac{\partial FN_v(V)}{\partial V} \frac{\partial V}{\partial Y} \frac{\partial Y}{\partial FN_y(Y)} \right] FN_c(C) \\ &\quad + \frac{\partial FN_c(C)}{\partial C} \left[FN_u(U) \frac{\partial C}{\partial X} \frac{\partial X}{\partial FN_x(X)} \right. \\ &\quad \left. + FN_v(V) \frac{\partial C}{\partial Y} \frac{\partial Y}{\partial FN_y(Y)} \right] \\ E &= D_e \frac{\partial^2 FN_c(C)}{\partial C^2} \left[\left(\frac{\partial C}{\partial X} \frac{\partial X}{\partial FN_x(X)} \right)^2 + \left(\frac{\partial C}{\partial Y} \frac{\partial Y}{\partial FN_y(Y)} \right)^2 \right] \\ &\quad + D_e \frac{\partial FN_c(C)}{\partial C} \left[\frac{\partial^2 C}{\partial X^2} \left(\frac{\partial X}{\partial FN_x(X)} \right)^2 \right. \\ &\quad + \frac{\partial^2 C}{\partial Y^2} \left(\frac{\partial Y}{\partial FN_y(Y)} \right)^2 \\ &\quad \left. + \frac{\partial C}{\partial X} \frac{\partial^2 X}{\partial FN_x(X)^2} + \frac{\partial C}{\partial Y} \frac{\partial^2 Y}{\partial FN_y(Y)^2} \right] \\ G &= \lambda FN_c(C) \end{aligned} \tag{14}$$

Eq. (14) is a transformed form of the physical Eq. (9). If the preprocessing function is determined, a further precise transformation of Eq. (14) can be performed. The normalization function was employed for preprocessing in this study. The normalization function and its inverse are, respectively, expressed as

$$N(\eta) = \frac{\eta - \eta_{\min}}{\eta_{\max} - \eta_{\min}}, \tag{15}$$

$$FN(\theta) = \theta(\eta_{\max} - \eta_{\min}) + \eta_{\min}, \tag{16}$$

where η represents the parameters x, y, t, q, u, v and c; $\eta_{\min}$ and $\eta_{\max}$ represent the minimum and maximum parameter values, respectively; and θ represents the normalized parameters X, Y, T, Q, U, V, and C. Therefore, based on

Table 1 Hyperparameters used in five models

Hyperparameters	Model				
	NN	NN-ERR	PINN-EA	PINN-EA-ERR	PINN-f
Network	NN	NN	PINN-EA	PINN-EA	PINN-f
Learning rate	Stepped descent	Stepped descent	Stepped descent	Stepped descent	Stepped descent
Optimizer	Adam	Adam	Adam	Adam	Adam
Iteration	1000	2000	1000	2000	3000
Training data	Data-01	Data-02	Data-01	Data-02	Data-01
Validation data	Data-validation	Data-validation	Data-validation	Data-validation	Data-validation

Eqs. (15) and (16), Eq. (14) can be transformed to obtain the following:

$$
\begin{aligned}
A' &= \frac{(c_{\max}-c_{\min})}{t_{\max}-t_{\min}}\frac{\partial C}{\partial T}\\
B' &= \left(\frac{u_{\max}-u_{\min}}{x_{\max}-x_{\min}}\frac{\partial U}{\partial X}+\frac{v_{\max}-v_{\min}}{y_{\max}-y_{\min}}\frac{\partial V}{\partial Y}\right)(C(c_{\max}-c_{\min})\\
&\quad + c_{\min}) + (c_{\max}-c_{\min})\left[\frac{U(u_{\max}-u_{\min})+u_{\min}}{x_{\max}-x_{\min}}\frac{\partial C}{\partial X}\right.\\
&\quad \left.+\frac{V(v_{\max}-v_{\min})+v_{\min}}{y_{\max}-y_{\min}}\frac{\partial C}{\partial Y}\right]\\
E' &= D_{\mathrm{e}}(c_{\max}-c_{\min})\left[\frac{\partial^2 C}{\partial X^2}\left(\frac{1}{x_{\max}-x_{\min}}\right)^2\right.\\
&\quad \left.+\frac{\partial^2 C}{\partial Y^2}\left(\frac{1}{y_{\max}-y_{\min}}\right)^2\right]\\
G' &= \lambda(C(c_{\max}-c_{\min})+c_{\min})
\end{aligned}
\tag{17}
$$

Then, Eq. (13) can be transformed into the following form:

$$
F(X,Y,T,U,V,C) = A' + B' - E' + G'. \tag{18}
$$

Therefore, after the preprocessing function is determined, Eq. (12) can be transformed into Eq. (18), which means that Eq. (9) is transformed into Eq. (18), following implementation of the equation adaption technique.

2.4 Model evaluation indexes

In this study, three key indicators were used to gauge the performance of the trained model: the training loss, validation loss, and relative standard deviation (*RSD*). The *RSD* measures the relative discrepancy between the predicted and true values. The *RSD* calculation formula is as follows:

$$
RSD = \frac{1}{N_{\mathrm{data}}}\sum_{i=1}^{N_{\mathrm{data}}}\left|\frac{y_{\mathrm{tr}}^{(i)}-y_{\mathrm{pre}}^{(i)}}{y_{\mathrm{tr}}^{(i)}}\right|\times 100\%, \tag{19}
$$

where y_{tr} and y_{pre} are the true and predicted values, respectively.

To facilitate analysis and comparison of different models, the final training and validation losses (FT and FV, respectively) were adopted as metrics. Consequently, the FT and FV for the NN and PINN models were denoted as FT_{NN}, FV_{NN}, FT_{PINN} and FV_{PINN}, respectively. The FT and FV ratios between the two models were $K_{\mathrm{FT}} = \frac{FT_{\mathrm{NN}}}{FT_{\mathrm{PINN}}}$ and $K_{\mathrm{FV}} = \frac{FV_{\mathrm{NN}}}{FV_{\mathrm{PINN}}}$, respectively.

3 Results and Discussion

Section 2 presents the basic conditions for establishing the PINN model. This section details the optimization of the model parameters and the subsequent performance analysis of the optimal model. The performance analysis of the optimal model is reported first, in Sect. 3.1 and 3.2. The basis for determining the model parameters is discussed subsequently, in Sect. 3.3.

3.1 Convergence and predictive performance of PINNs without equation adaption

This section discusses the convergence and predictive performance of the PINN network with no equation adaption trained on the Data-01 database. The PINN model discussed in this section corresponds to the PINN-f model in Sect. 2.2. Figure 5 shows the training and validation loss convergence during training of the PINN-f model. Following application of the Leaky ReLU and Hardswish activation functions, the model training and validation losses exceeded orders of magnitude of 10^{23} and 10^{8}, respectively, and convergence to smaller values did not occur. However, for the models with the ReLU, Sigmoid, and Tanh activation functions, these values converged to 10^{-3}. Therefore, compared to the Leaky ReLU and Hardswish activation functions, the ReLU, Sigmoid, and Tanh activation functions yielded better convergence of the model training and validation losses.

Although the training and validation losses are fundamental metrics for evaluating the effectiveness of model training, the accuracy with which physical quantities are predicted is also a crucial indicator. In this study, the model accuracy was assessed using the predictive RSD, which measures the RSD between the predicted and true values, as expressed in Eq. (19). Figure 6 illustrates the RSD between the predicted and true values for the x- and y-velocity components and the ^{212}Pb concentration for the PINN-f model. The models employing the Leaky ReLU and Hardswish activation functions exhibited RSDs for the predictive values of the three considered physical quantities that significantly exceeded 10^{6}. In particular, the RSD for the predictive ^{212}Pb concentration was as high as the order of 10^{27}. The models using the ReLU, Sigmoid, and Tanh activation functions exhibited smaller predictive RSDs. However, the values for the three considered physical quantities were relatively high, all exceeding 1, with the predictive RSD for the ^{212}Pb concentration reaching the order of 10^{11}. Therefore, the PINN-f models trained with these five activation functions did not achieve satisfactory prediction accuracy.

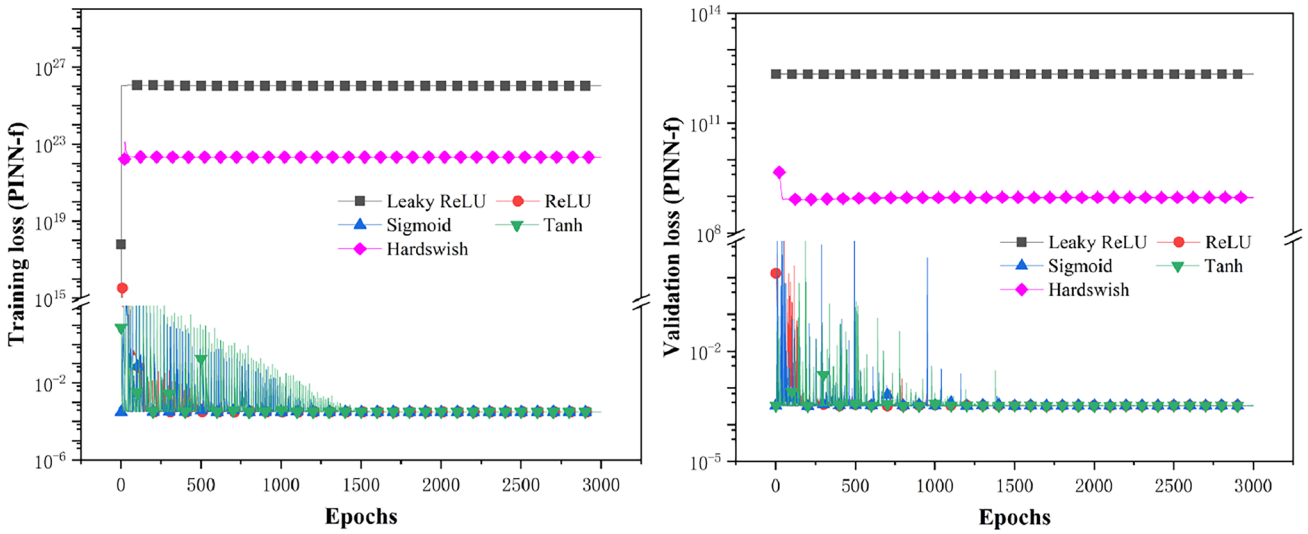

Fig. 5 (Color online) Training and validation losses of PINN-f model for different activation functions

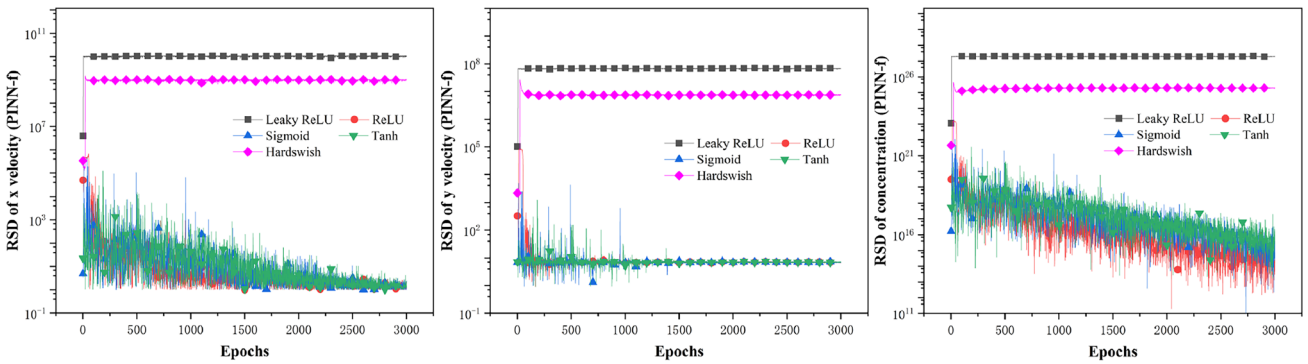

Fig. 6 (Color online) RSDs between predicted and true values for PINN-f models with different activation functions

3.2 Convergence and predictive performance of PINNs with equation adaption

As indicated in Sect. 3.1, the neural network method without equation adaption failed to accurately predict the x- and y-velocity components and the ^{212}Pb concentration. This section compares the neural network model with equation adaption to that without equation adaption and verifies the compatibility and robustness of the proposed method within neural networks.

3.2.1 Comparative analysis of PINN and NN models without interference data

(1) Neural network convergence

This section discusses the training and predictive performance of the PINN network with the equation adaption and the classical neural network, both of which were trained without interference data (i.e., on the Data-01 database). These models correspond to the PINN-EA and NN models described in Sect. 2.2, respectively. Figure 7 shows the training and validation loss convergence for both models under different activation function conditions. Without interference data, the training- and validation loss convergence patterns of the two models were consistent and reached the order of 10^{-6}. Compared to the results reported in Sect. 3.1, these results indicate that preprocessing of the database effectively reduce the model training difficulty and facilitates training and validation loss convergence. This outcome further demonstrates the necessity of equation adaption in the training of PINN models.

Figure 8 compares the magnitudes to which the two models' training and validation losses converged after 1000 training epochs under different activation function conditions. As the activation function changed, the two models exhibited a consistent trend in both FT and FV, with

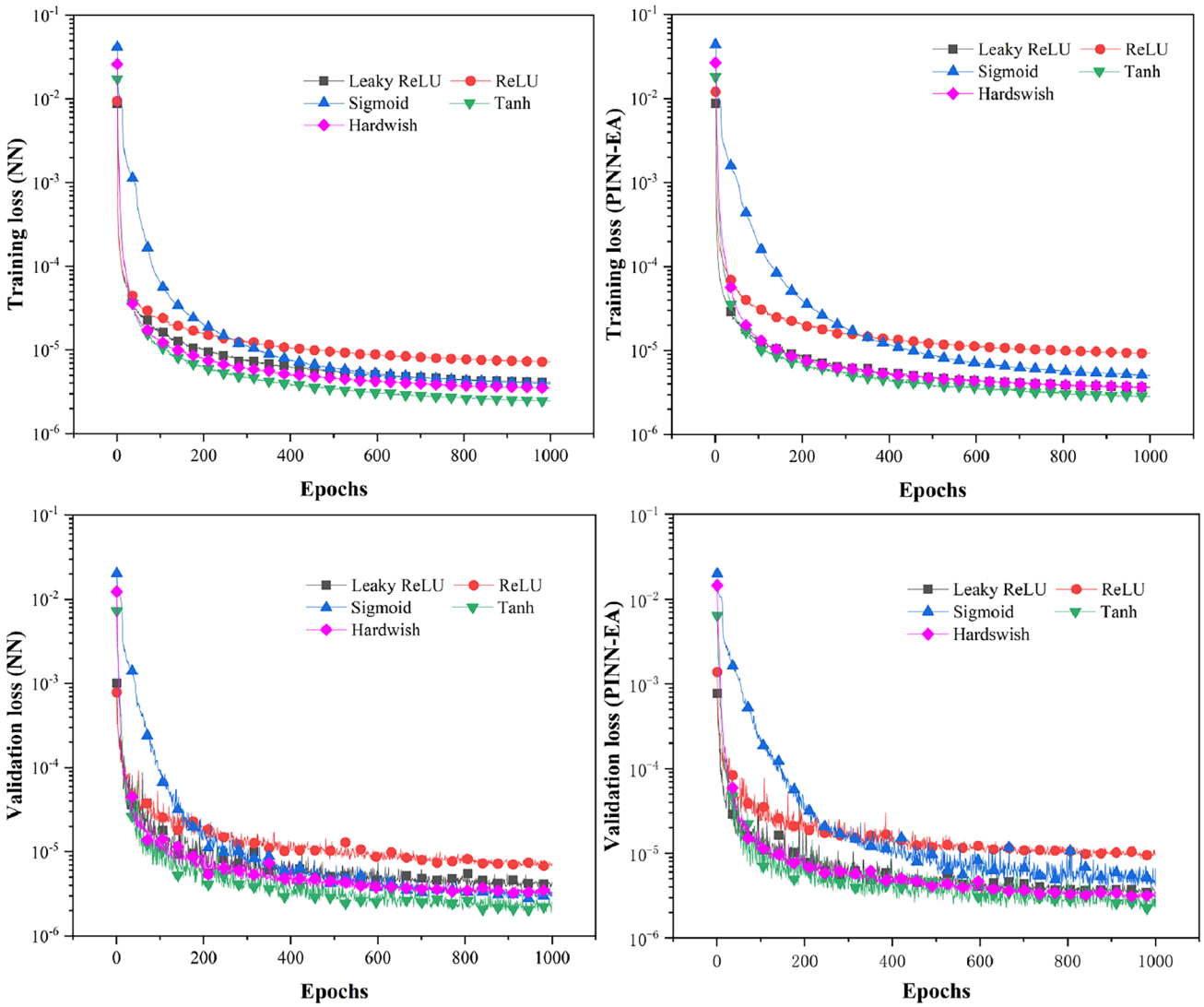

Fig. 7 (Color online) Training and validation losses of NN and PINN-EA models for different activation functions without interference data

the K values fluctuating around 1. This indicates that there was almost no difference between the FT values of the two models, and similarly, that the FV values of the two models were almost identical. Therefore, without interference data, a comprehensive analysis of the FT, FV, and K values suggests that the neural network model with equation adaption (PINN-EA) and the classical neural network model (NN) exhibit good consistency.

(2) Model prediction accuracy

Figure 9 illustrates the change pattern of the predictive RSD for the x-velocity and y-velocity components and the ^{212}Pb concentration as the NN and PINN-EA models underwent continuous training. When the ReLU activation function was used, the RSD for the x-velocity component exceeded 100%. For the other activation functions, however, the RSD values for these properties remained below 100%. The lowest RSD values were obtained for the y-velocity component, with all values below 10%, whereas those for the x-velocity and ^{212}Pb concentration fell between 10% and 100%. Figure 9 also displays the RSD change pattern for the x-velocity and y-velocity components and the ^{212}Pb concentration as the PINN-EA model underwent continuous training. The RSD change pattern for the three physical properties were essentially the same for both the PINN-EA and NN models.

To more intuitively illustrate the relationship between the ^{212}Pb concentrations predicted by the NN and PINN-EA models and the true values, a scatter plot of the predicted and true values is shown in Fig. 10. The true and predicted values were normalized, with the maximum and minimum values in the normalization being the $\eta_{\min}$ and $\eta_{\max}$ of Eq. (15). The true and predicted ^{212}Pb concentration values

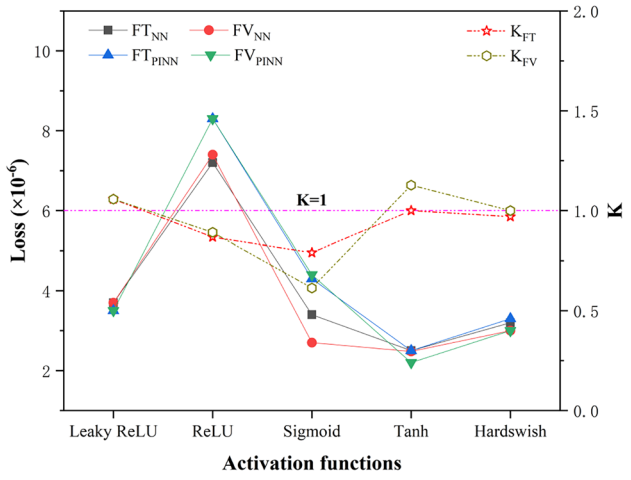

Fig. 8 (Color online) FT, FV, and K values of NN and PINN-EA models for different activation functions without interference data

are plotted on the horizontal and vertical axes, respectively. The closer the scatter points are to the $y = x$ line, the closer the predicted values are to the true values, which indicates better prediction accuracy. The scatter points in Fig. 10 are essentially on the $y = x$ line, suggesting that both the NN and PINN-EA models had good prediction accuracy.

In summary, analysis of the training and validation loss convergence patterns, as well as the model predictive accuracy revealed that, without interference data, the training and validation losses of the NN and PINN-EA models exhibit consistent convergence patterns during training. Moreover, compared to the models in Sect. 3.1, the two models in this section achieved higher prediction accuracy for the ^{212}Pb concentration. These results demonstrate that the equation adaption technique can be effectively integrated into neural networks without conflict, with good compatibility.

3.2.2 Comparative analysis of PINN and NN models with interference data

To verify the robustness of the PINN model following adoption of the equation adaption technique, both the classical neural network and the PINN network were trained using data containing interference (the Data-02 database). The effects of training were examined for the two models. The models discussed in this section correspond to the NN-ERR and PINN-EA-ERR models described in Sect. 2.2.

(1) Convergence of neural networks

Figure 11 shows the training and validation loss convergence with epochs for the two models under different activation functions. When the model training was stable, although the NN-ERR training loss was considerably smaller than that of PINN-EA-ERR, the PINN-EA-ERR validation loss was smaller than that of NN-ERR, by approximately

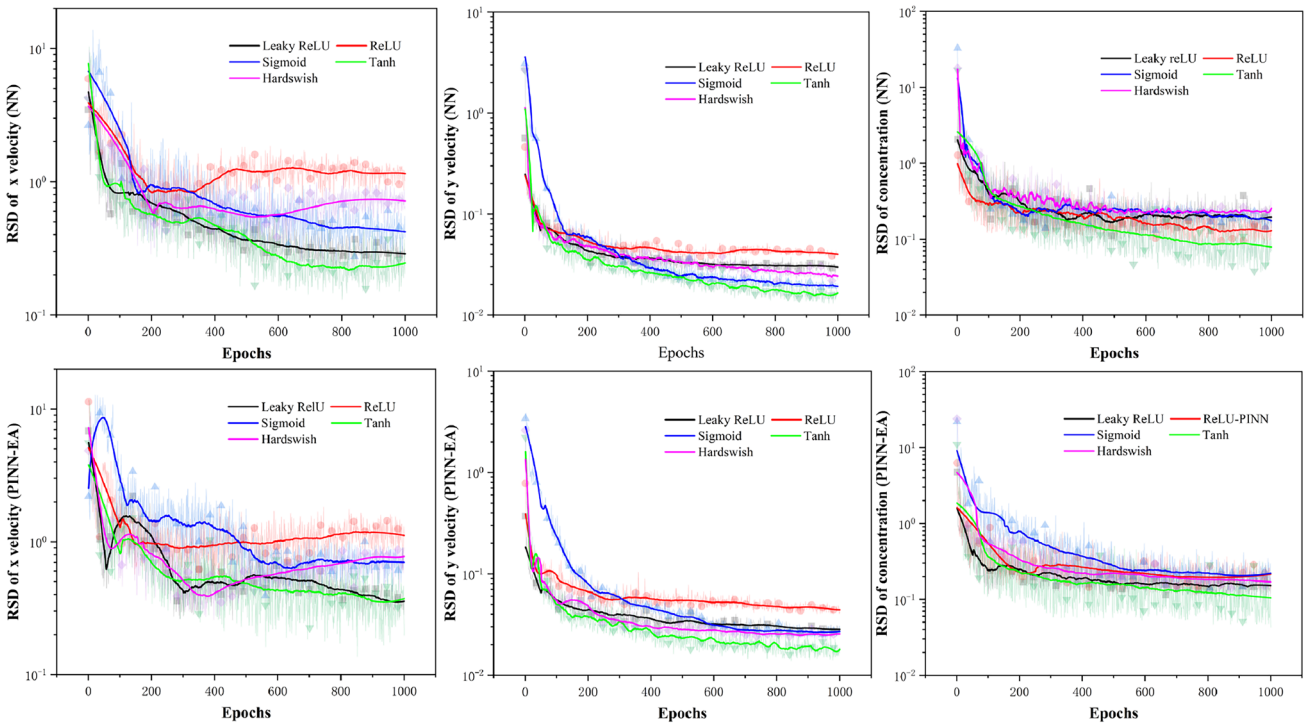

Fig. 9 (Color online) RSD between predicted and true values for NN and PINN-EA models for different activation functions without interference data

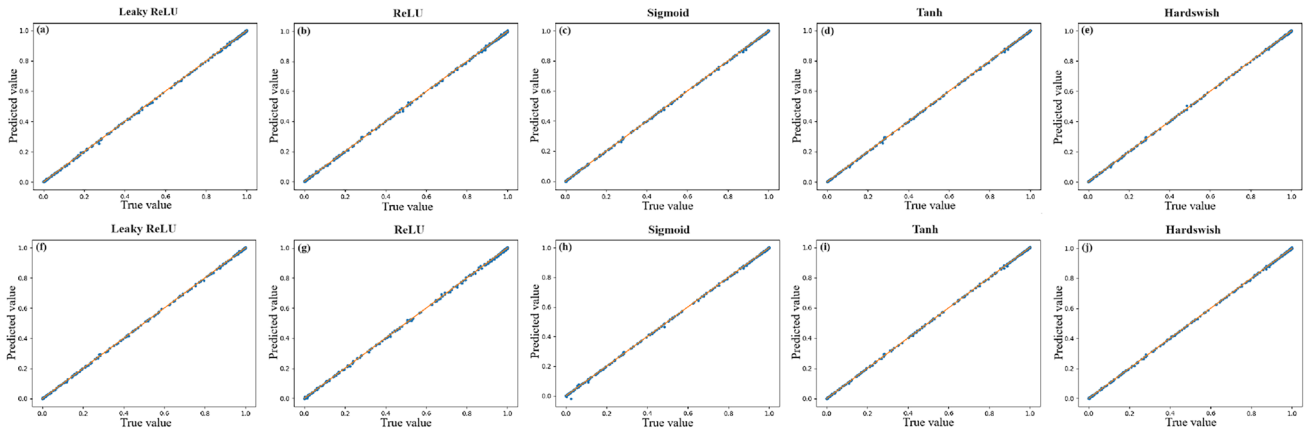

Fig. 10 (Color online) Scatter plots of predicted and true values for NN (a–e) and PINN-EA (f–j) models without interference data

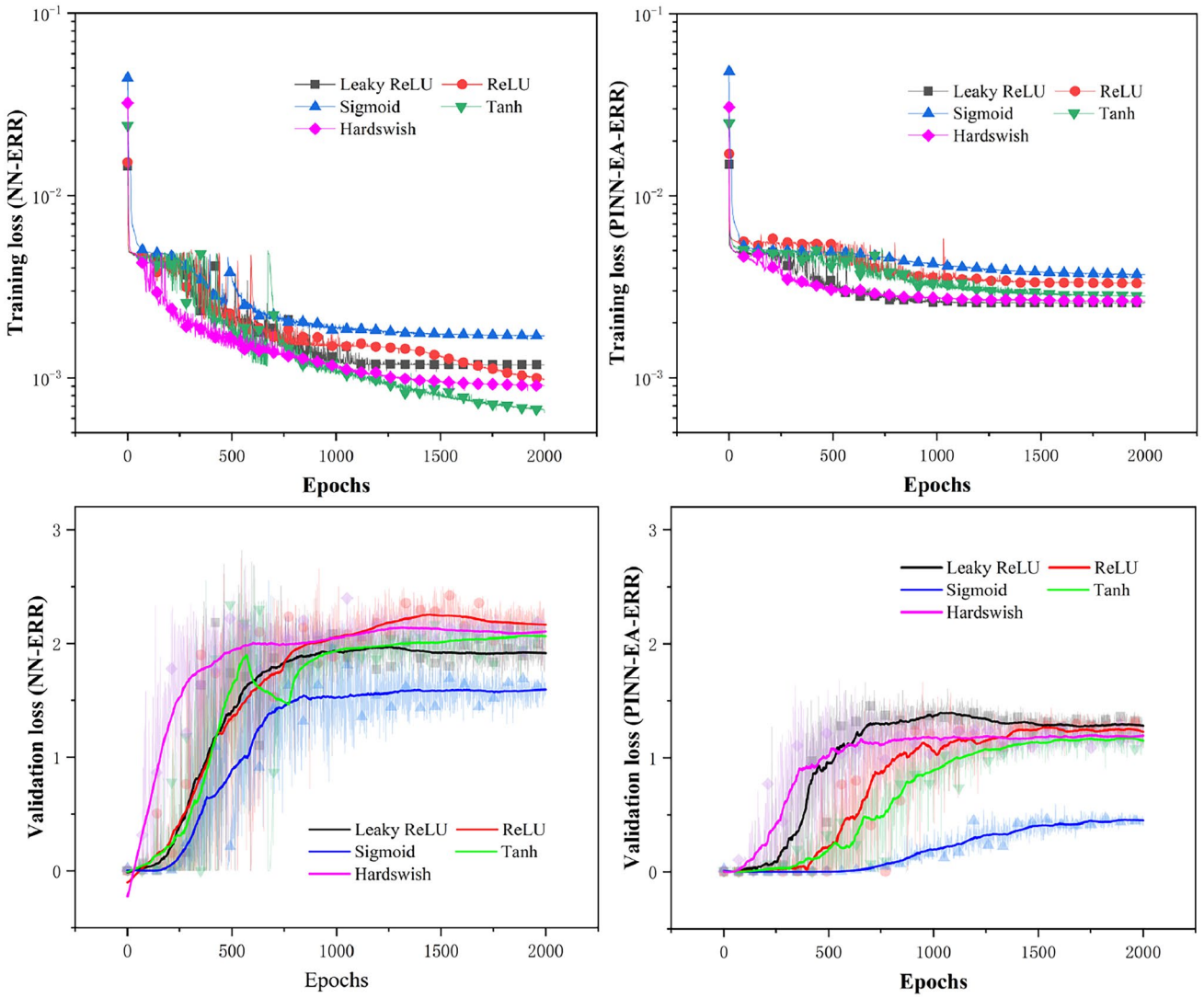

Fig. 11 (Color online) Training and validation losses of NN-ERR and PINN-EA-ERR models for different activation functions with interference data

half. Further, both the NN-ERR and PINN-EA-ERR models exhibited overfitting at approximately 200 epochs. Table 2 reports the magnitudes to which the training and validation losses of the classic neural network and PINN network converged after 2000 training epochs. From Table 2, the FT values of both models were within the range of 1×10^{-4} to 1×10^{-3}, and the FV values exceeded 1×10^{-1}. Under the condition of no interference data, as shown in Fig. 8, the FT and FV values of the two models were within the range of 1×10^{-6} to 1×10^{-5}. This indicates that, under the influence of interference data, the FT increased by two to three orders of magnitude, and the FV increased by 6. Therefore, interference data has a significant and adverse effect on model training.

Figure 12 shows the ratios of the FT or FV values between NN-ERR and PINN-EA-ERR under different activation functions. The FT ratios of the two models were less than 1. In contrast, the FV ratios were larger than 1. This result indicates that the PINN-EA-ERR model with the equation adaption constraint can recognize erroneous data, further enhancing the robustness of the neural network.

(2) Model prediction accuracy

Figure 13 shows the RSD values between the predicted and true values for the *x*- and *y*-velocity components and the ^{212}Pb concentration for the NN-ERR and PINN-EA-ERR models under different activation function conditions. Comparison of the NN-ERR and PINN-EA-ERR models reveals that, for the *x* and *y* fluid velocity vectors, the RSD of the PINN-EA-ERR model predictions were half those of the NN-ERR model; as regards the RSD of the predicted ^{212}Pb concentration values, those of the PINN-EA-ERR model were approximately 1/400th that of the NN-ERR model. This indicates that the PINN-EA-ERR model had higher prediction accuracy than the NN-ERR model, especially for the ^{212}Pb concentration. The main reason for this performance is that the physical equation used in this study is the ^{212}Pb concentration transport equation, which effectively constrained the ^{212}Pb concentration prediction. This equation also incorporates flow field physical quantities; that is, the *x*- and *y*-velocity components. Some improvement in prediction accuracy was observed for those quantities; however, the constraining force was insufficient and there was a stark contrast in accuracy.

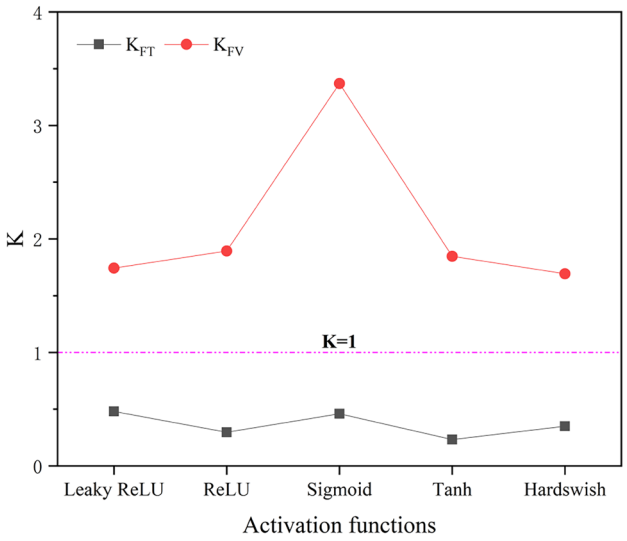

Fig. 12 (Color online) *K* values of NN-ERR and PINN-EA-ERR models for different activation functions with interference data

Figure 14 depicts scatter plots of the ^{212}Pb concentrations predicted by the NN-ERR and PINN-EA-ERR models against the true values. The true and predicted values were uniformly normalized in the same manner as for the results shown in Fig. 10. Figure 14(a)–(e) and (f)–(j) shows scatter plots of the ^{212}Pb concentration predictions of the NN-ERR and PINN-EA-ERR models, respectively. In Figs. 14(a)–(e), most of the scatter points lie on the $y = x$, except for those close to 0. This result indicates a significant deviation in the NN-ERR model predictions for values close to 0, which markedly decreased the prediction accuracy. Moreover, compared to the NN-ERR model, the predicted scatter points (for the PINN-EA-ERR model (Figs. 14(f)-(j)) are mainly on the $y = x$ line. and the model maintained good prediction accuracy for values close to 0. This further demonstrates the effectiveness of the practical constraint of the proposed equation adaption on the neural network model.

3.3 N_{data}, N_{cell}, and α parameters

Parameter optimization is typically performed for neural networks with different activation functions. From Sect. 3.1 and 3.2, the optimal performance was obtained for the tanh

Table 2 Comparison of training and validation losses of NN-ERR and PINN-EA-ERR models for different activation functions

		Activation functions				
		Leaky ReLU	ReLU	Sigmoid	Tanh	Hardswish
NN	FT_{NN}	1.2×10^{-3}	9.8×10^{-4}	1.7×10^{-3}	6.5×10^{-4}	9.1×10^{-4}
	FV_{NN}	1.90	2.14	1.55	2.05	2.10
PINN	FT_{PINN}	2.5×10^{-3}	3.3×10^{-3}	3.7×10^{-3}	2.8×10^{-3}	2.6×10^{-3}
	FV_{PINN}	1.09	1.13	0.46	1.11	1.24

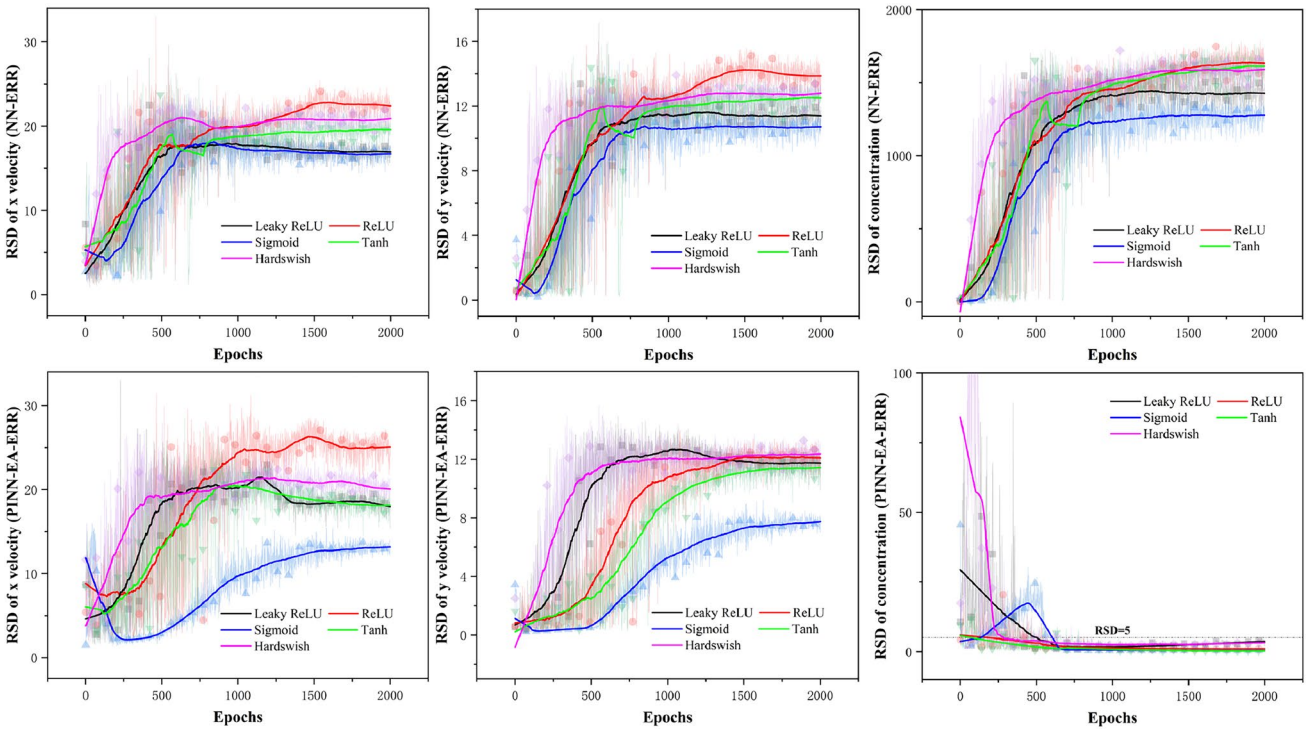

Fig. 13 (Color online) RSD between predicted and true values of NN-ERR and PINN-EA-ERR models for different activation functions with interference data

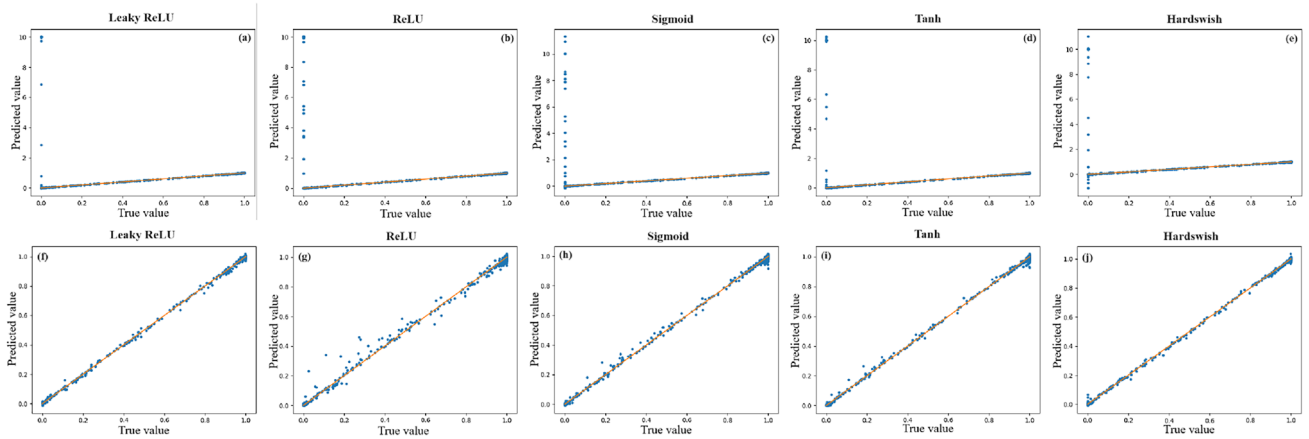

Fig. 14 Scatter plot of predicted and true values for NN-ERR (a–e) and PINN-EA-ERR (f–j) models with interference data. (Color figure online)

activation function. Therefore, in this section, we take the neural network model with the tanh activation function as an example and discuss the parameter optimization process in detail. We primarily consider the influence of N_{data}, N_{cell}, and α on the prediction accuracy without interference data. The ^{212}Pb concentration predicted by the PINN for different N_{data} is presented in Fig. 15. The RSD of the ^{212}Pb concentration prediction was used to characterize the model prediction accuracy. The layer number was set to 5 for all cases, and the number of neurons in each layer was varied from 16 to 128.

Three points can be summarized from Fig. 15. ① Data dependence on PINN prediction accuracy: With increasing N_{data}, the prediction accuracy for the ^{212}Pb concentration improved significantly. When N_{data} reached 10^4, the prediction accuracy did not change considerably. ② Determination of N_{cell}: From Figs. 15(a)–(e), with increasing N_{cell}, the prediction accuracy for the ^{212}Pb

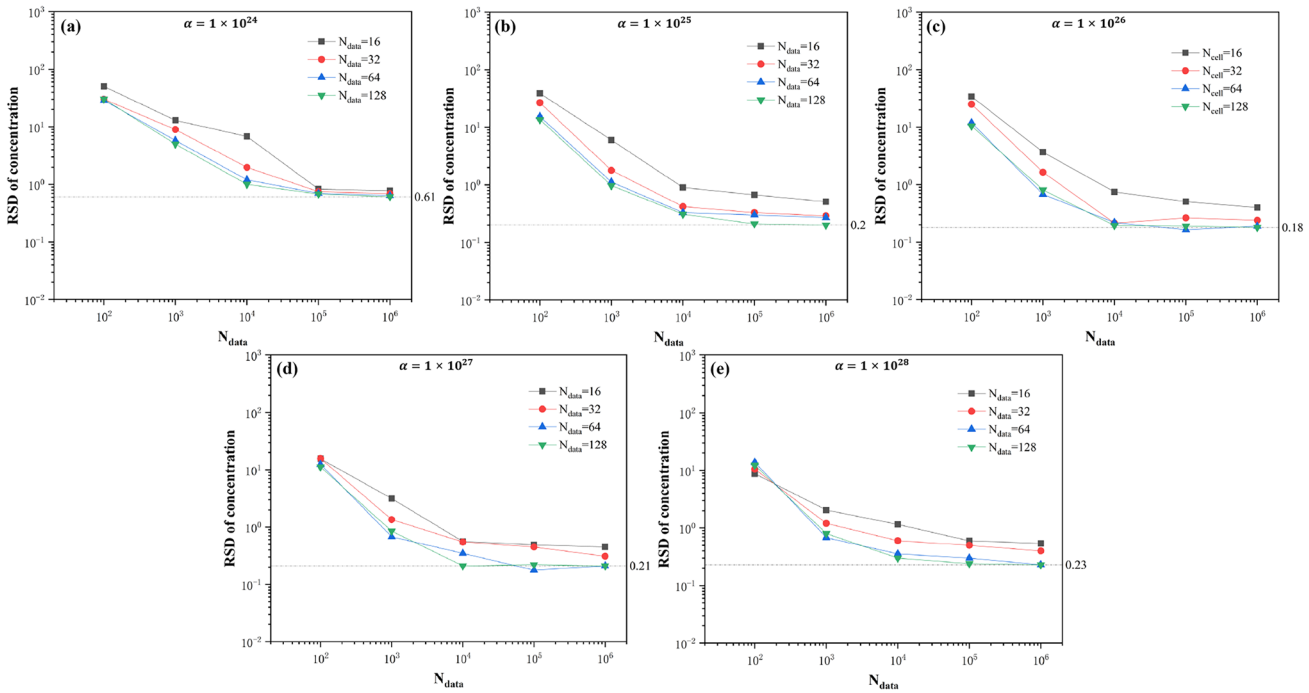

Fig. 15 (Color online) RSD of ^{212}Pb concentration for varying N_{data} and N_{cell}. N_{layer} was fixed at 5. In (a)–(e), $\alpha = 1 \times 10^{24}$, 1×10^{25}, 1×10^{26}, 1×10^{27}, and 1×10^{28}, respectively

concentration increased gradually; the prediction accuracies for $N_{cell} = 64$ and 128 were close. Considering the calculation resources, the optimal N_{cell} for this model was 64. ③ Determination of α: In the training experiment for the PINN neural network model, the order of magnitude of αL_{eqns} was approximately 10^{-3}, and the order of magnitude of L_{data} was approximately 10^{-40}. To facilitate supervision of the training of neural network models for both, in this paper study similar orders of magnitude were set for αL_{eqns} and L_{data}, so the change in prediction accuracy was analyzed when α was 10^{24}–10^{28}. As apparent from Fig. 15, for sufficient training data and with increasing α, the model prediction accuracy first increased rapidly and then decreased slightly, and optimal performance was obtained when α was 10^{26}. Therefore, in this study, $N_{cell} = 64$ and $\alpha = 10^{28}$ were selected for the optimal model.

This study had some limitations. A high level of accuracy was not achieved for the flow field prediction when interference data were considered (Sect. 3.2.2). This indicates that the conditions of the physical equations should be adjusted to constrain the model and enhance its prediction accuracy. As regards noise in the training stage of this research model, the training data were preprocessed to a certain extent and a good signal-to-noise ratio was obtained.

4 Conclusion

In this study, a PINN with equation adaption was established for ^{220}Rn progeny concentration prediction. A PINN without equation adaption was examined; this model failed to yield the desired outcomes. This failure underscores the critical role of the equation adaption in training neural networks. For training without interference data, the PINN with equation adaption exhibited performance consistent with that of a classical neural network model, achieving high accuracy when predicting ^{220}Rn concentrations. This outcome emphasizes the excellent compatibility of the equation adaption technique with neural networks. When interference data were considered, the PINN model with equation adaption retained good prediction accuracy, especially for ^{220}Rn concentration prediction. This outcome highlights the effectiveness of equation adaption in constraining neural networks with physical equations, thereby improving the robustness of the neural network model.

In future work, different types of noise will be added to the model, based on factors such as the background radioactivity level and detection method. Additionally, the equation adaption technique will be used to model specific physical objects with PINNs; for example, precise and rapid prediction of ^{220}Rn and its progeny concentrations will be explored. Overall, the equation adaptation approach presented in this study has good universality and provides

a theoretical foundation for the widespread application of neural networks in various fields.

Author Contributions All authors contributed to the study conception and design. Material preparation, data collection and analysis were performed by Shao-Hua Hu, Qi Qiu, De-Tao Xiao, Xiang-Yuan Deng, Xiang-Yu Xu, Peng-hao Fan, Lei Dai, Zhi-Wen Hu, Tao Zhu, and Qing-Zhi Zhou. The first draft of the manuscript was written by Shao-Hua Hu, and 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.29256 and https://www.doi.org/10.57760/sciencedb.29256..

Declarations

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

References

1. A. Krizhevsky, I. Sutskever, G.E. Hinton et al., ImageNet classification with deep convolutional neural networks. Commun. ACM **60**, 84–90 (2017). https://doi.org/10.1145/3065386
2. Y. LeCun, Y. Bengio, G. Hinton et al., Deep learning. Nature **521**, 436–444 (2015). https://doi.org/10.1038/nature14539
3. B.M. Lake, R. Salakhutdinov, J.B. Tenenbaum et al., Human-level concept learning through probabilistic program induction. Science **350**, 1332–1338 (2015). https://doi.org/10.1126/science.aab3050
4. B. Alipanahi, A. Delong, M.T. Weirauch et al., Predicting the sequence specificities of DNA-and RNA-binding proteins by deep learning. Nat. Biotechnol. **33**, 831–838 (2015). https://doi.org/10.1038/nbt.3300
5. G.E. Karniadakis, I.G. Kevrekidis, L. Lu et al., Physics-Informed Mach. Learn. Nat. Rev. Phys. **3**, 422–440 (2021). https://doi.org/10.1038/s42254-021-00314-5
6. L. Yuan, Y.Q. Ni, X.Y. Deng et al., A-PINN: Auxiliary physics informed neural networks for forward and inverse problems of nonlinear integro-differential equations. J. Comput. Phys. **462**, 111260 (2022). https://doi.org/10.1016/j.jcp.2022.111260
7. S.S. Miriyala, V.R. Subramanian, K. Mitra et al., TRANSFORM-ANN for online optimization of complex industrial processes: Casting process as case study. Eur. J. Oper. Res. **264**, 294–309 (2018). https://doi.org/10.1016/j.ejor.2017.05.026
8. C. Jin, J. Huang, Y. Chen, Neural architecture search via progressive partial connection with attention mechanism. Sci Rep **14**, 6462 (2024). https://doi.org/10.1038/s41598-024-57236-2
9. X. Xie, X. Song, Z. Lv et al., Efficient evaluation methods for neural architecture search: A survey (2023). arXiv:2301.05919
10. W. Qu, Y. Gu, S. Zhao et al., Boundary integrated neural networks and code for acoustic radiation and scattering. Int. J. Mech. Syst. Dynam. **4**, 2767–1399 (2024). https://doi.org/10.1002/msd2.12109
11. S. García, J. Luengo, F. Herrera et al., *Data preprocessing in data mining* (Springer, Switzerland, 2015), pp.59–139. https://doi.org/10.1007/978-3-319-10247-4
12. M.V. Nuland, H. Rosing, A.D. Huitema et al., Predictive value of microdose pharmacokinetics. Clin. Pharmacokinet. **58**, 1221–1236 (2019). https://doi.org/10.1007/s40262-019-00769-x
13. N. Keat, J. Kenny, K. Chen et al., A microdose PET study of the safety, immunogenicity, biodistribution, and radiation dosimetry of 18F-FB-A20FMDV2 for imaging the integrin $\alpha_v\beta_6$. J. Nucl. Med. Technol. **46**, 136–143 (2018). https://doi.org/10.2967/jnmt.117.203547
14. L. Pouchard, K.G. Reyes, F.J. Alexander et al., A rigorous uncertainty-aware quantification framework is essential for reproducible and replicable machine learning workflows. Digital Discovery **2**, 1251–1258 (2023). https://doi.org/10.1039/D3DD00094J
15. S.H. Hu, Y.J. Ye, Z.Z. He et al., Analysis and optimization of performance parameters of the ^{220}Rn chamber in flow-field mode using computational fluid dynamics method. Nucl. Sci. Tech. **35**, 175 (2024). https://doi.org/10.1007/s41365-024-01526-x
16. Z. He, D. Xiao, L. Lv et al., Stable control of thoron progeny concentration in a thoron chamber for calibration of active sampling monitors. Radiat. Meas. **102**, 27–33 (2017). https://doi.org/10.1016/j.radmeas.2017.02.013
17. W. Li, Q. Zhou, Z. He et al., Optimization of the thoron progeny compensation system of a thoron calibration chamber. J. Radioanal. Nucl. Ch. **324**, 1255–1263 (2020). https://doi.org/10.1007/s10967-020-07180-y
18. M. Raissi, P. Perdikaris, G.E. Karniadakis et al., 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
19. S. Cai, Z. Mao, Z. Wang et al., Physics-informed neural networks (PINNs) for fluid mechanics: A review. Acta Mech. Sinica-PRC **37**, 1727–1738 (2021). https://doi.org/10.1007/s10409-021-01148-1
20. Q. He, D. Barajas-Solano, G. Tartakovsky et al., Physics-informed neural networks for multiphysics data assimilation with application to subsurface transport. Adv. Water Resour. **141**, 103610 (2020). https://doi.org/10.1016/j.advwatres.2020.103610
21. S. Falas, C. Konstantinou, M.K.Michael et al., Physics-informed neural networks for securing water distribution systems (2020). arXiv:2009.08842
22. C. Cheng, G.T. Zhang, Deep learning method based on physics informed neural network with resnet block for solving fluid flow problems. Water-Sui **13**, 423 (2021). https://doi.org/10.3390/w13040423
23. Z. Mao, A.D. Jagtap, G.E. Karniadakis et al., Physics-informed neural networks for high-speed flows. Comput. Method. Appl. M. **360**, 112789 (2020). https://doi.org/10.1016/j.cma.2019.112789
24. Q. Zhu, Z. Liu, J. Yan et al., Machine learning for metal additive manufacturing: Predicting temperature and melt pool fluid dynamics using physics-informed neural networks. Comput. Mech. **67**, 619–635 (2021). https://doi.org/10.1007/s00466-020-01952-9
25. J. Du, J. Zheng, Y. Liang et al., Deeppipe: A two-stage physics-informed neural network for predicting mixed oil concentration distribution. Energy **276**, 127452 (2023). https://doi.org/10.1016/j.energy.2023.127452
26. A. Arzani, J.X. Wang, R.M.D' Souza et al., Uncovering near-wall blood flow from sparse data with physics-informed neural networks. Phys. Fluids **33**, 071905 (2021). https://doi.org/10.1063/5.0055600
27. F.S. Costabal, Y. Yang, P. Perdikaris et al., Physics-informed neural networks for cardiac activation mapping. Front. Phys-Lausanne **8**, 42 (2020). https://doi.org/10.3389/fphy.2020.00042
28. Y. Chen, L. Lu, G.E. Karniadakis et al., Physics-informed neural networks for inverse problems in nano-optics and metamaterials. Opt. Express **28**, 11618–11633 (2020). https://doi.org/10.1364/OE.384875
29. S. Goswami, C. Anitescu, S. Chakraborty et al., Transfer learning enhanced physics informed neural network for phase-field modeling of fracture. Theor. Appl. Fract. Mec. **106**, 102447 (2020). https://doi.org/10.1016/j.tafmec.2019.102447

30. E. Zhang, M. Yin, G.E. Karniadakis et al., Physics-informed neural networks for nonhomogeneous material identification in elasticity imaging (2020). arXiv:2004.04525
31. M. Yin, X. Zheng, J.D. Humphrey et al., Non-invasive inference of thrombus material properties with physics-informed neural networks. Comput. Method. Appl. M. **375**, 113603 (2021). https://doi.org/10.1016/j.cma.2020.113603
32. R. Zhang, Y. Liu, H. Sun et al., Physics-informed multi-LSTM networks for metamodeling of nonlinear structures. Comput. Method. Appl. M. **369**, 113226 (2020). https://doi.org/10.1016/j.cma.2020.113226
33. E. Haghighat, M. Raissi, A. Moure et al., A physics-informed deep learning framework for inversion and surrogate modeling in solid mechanics. Comput. Method. Appl. M. **379**, 113741 (2021). https://doi.org/10.1016/j.cma.2021.113741
34. Q. Zhang, Y. Chen, Z. Yang, Data-driven solutions and discoveries in mechanics using physics informed neural network. Preprints **2020**, 2020060258 (2020). https://doi.org/10.20944/preprints202006.0258.v1
35. E. Zhang, M. Dao, G.E. Karniadakis et al., Analyses of internal structures and defects in materials using physics-informed neural networks. Sci. Adv. **8**(7), eabk0644 (2022). doi: https://doi.org/10.1126/sciadv.abk0644
36. T. Zhu, W. Luo, C. Bu et al., Accelerate population-based stochastic search algorithms with memory for optima tracking on dynamic power systems. IEEE T. Power Syst. **31**(1), 268–277 (2015). https://doi.org/10.1109/TPWRS.2015.2407899
37. H. Gong, T. Zhu, Z. Chen et al., Parameter identification and state estimation for nuclear reactor operation digital twin. Ann. Nucl. Energy **180**, 109497 (2023). https://doi.org/10.1016/j.anucene.2022.109497
38. Q.H. Ngo, B.L. Nguyen, T.V. Vu et al., Physics-informed graphical neural network for power system state estimation. Appl. Energ. **358**, 122602 (2024). https://doi.org/10.1016/j.apenergy.2023.122602
39. M.E. Bento, Physics-guided neural network for load margin assessment of power systems. IEEE T. Power Syst. **39**(1), 564–575 (2023). https://doi.org/10.1109/TPWRS.2023.3266236
40. R. Nellikkath, S. Chatzivasileiadis, Physics-informed neural networks for ac optimal power flow. Electr. Pow. Syst. Res. **212**, 108412 (2022). https://doi.org/10.1016/j.epsr.2022.108412
41. P.R. Bana, M. Amin, Control for grid-connected VSC with improved damping based on physics-informed neural network. IEEE J. Emerg. Sel. Top. Ind. Electron. **4**(3), 878–888 (2023). doi: https://doi.org/10.1109/jestie.2023.3258339
42. W. Wang, M. Yang, The nonlinear flow characteristics within two-dimensional and three-dimensional counterflow models within symmetrical structures. Energies **17**, 3176 (2024). https://doi.org/10.3390/en17133176
43. M. Raissi, P. Perdikaris, G.E. Karniadakis et al., 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
44. C. Rackauckas, Y. Ma, J. Martensen et al., Universal differential equations for scientific machine learning. arXiv:2001.04385 (2020)
45. H. Wang, Y. Liu, S. Wang et al., Dense velocity reconstruction from particle image velocimetry/particle tracking velocimetry using a physics-informed neural network. Phys. Fluids. **34** (2022). doi: https://doi.org/10.1063/5.0078143
46. S.H. Rudy, S.L. Brunton, J.L. Proctor et al., Data-driven discovery of partial differential equations. Sci. Adv. **3**, e1602614 (2017). https://doi.org/10.1126/sciadv.1602614
47. K. He, X. Zhang, S. Ren et al., Deep residual learning for image recognition. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR). IEEE 770–778 (2016). https://doi.org/10.1109/CVPR.2016.90
48. P. Ramachandran, B. Zoph, Q.V. Le, Searching for activation functions. arXiv:1710.05941 (2017)
49. S. Markidis, The old and the new: Can physics-informed deep-learning replace traditional linear solvers? Front. Big Data. **4**, 669097 (2021). https://doi.org/10.3389/fdata.2021.669097
50. L. Ambrosio, G.D. Maso, A general chain rule for distributional derivatives. P. Am. Math. Soc. **108**, 691–702 (1990). https://doi.org/10.1090/S0002-9939-1990-0969514-3