RESEARCH ARTICLE | JULY 08 2022

Orthogonal grid physics-informed neural networks: A neural network-based simulation tool for advection–diffusion–reaction problems

Special Collection: Artificial Intelligence in Fluid Mechanics

Qingzhi Hou ; Zewei Sun ; Li He ; Alireza Karemat

Physics of Fluids 34, 077108 (2022)
https://doi.org/10.1063/5.0095536

Physics of Fluids ARTICLE scitation.org/journal/phf

Orthogonal grid physics-informed neural networks: A neural network-based simulation tool for advection–diffusion–reaction problems

Cite as: Phys. Fluids **34**, 077108 (2022); doi: 10.1063/5.0095536
Submitted: 11 April 2022 · Accepted: 21 June 2022 ·
Published Online: 8 July 2022

Qingzhi Hou,[1] Zewei Sun,[2] Li He,[1] and Alireza Karemat[3,a)]

AFFILIATIONS

[1]State Key Laboratory of Hydraulic Engineering Simulation and Safety, Tianjin University, Tianjin 300350, China
[2]College of Intelligence and Computing, Tianjin University, Tianjin 300350, China
[3]Department of Civil and Environmental Engineering, The Hong Kong Polytechnic University, Hongkong, China

Note: This paper is part of the special topic, Artificial Intelligence in Fluid Mechanics.
[a)]**Author to whom correspondence should be addressed:** alireza.keramat@polyu.edu.hk

ABSTRACT

Stable and accurate reconstruction of pollutant transport is a crucial and challenging problem, including the inverse problem of identifying pollution sources and physical coefficients and the forward problem of inferring pollutant transport. Governed by advection, diffusion, and reaction processes, this transport phenomenon can be represented by the advection–diffusion–reaction (ADR) equation. In this paper, the physics-informed neural networks (PINNs) are applied to solve the forward and inverse ADR problems. To further enhance the stability and accuracy of the original PINN, two improvements are developed. The first adjusts the orthogonal grid (OG) point selection method and the other suggests adding an additional regulation function, namely, first derivative constraint (FDC). The new method is referred to as OG-PINN with FDC. To verify the effectiveness of the proposed method, five forward and inverse ADR problems are solved, and the results are compared with the analytical and reference solutions. For forward problems, the improved method can solve various ADR problems accurately and stably. For inverse problems, the ability of the OG-PINN for model parameter learning and initial distribution prediction is demonstrated and analyzed. The former gives the missed physical information in the ADR equation from the data, and the latter is used to trace the source of pollutants. The proposed method is quantitatively reliable for investigating various advection–diffusion–reaction processes.

I. INTRODUCTION

Estimating the distribution and diffusion of unknown pollution sources in water is an important measure to formulate the best remediation strategy after surface water or groundwater is polluted. Pollutant transport is mainly affected by advection, diffusion, and chemical or biological reactions, and the advection–diffusion–reaction (ADR) equation is mostly used to describe the pollutant transport process. The distribution of unknown pollution sources and the physical coefficients in the transport process are the key points to solve the inverse problems.[1] This corresponds, respectively, to tracing the initial distribution and identifying diffusion–reaction coefficients. The reconstruction of source characteristics is not the final goal of inverse problems, improving the prediction also needs to be considered.[2] In this case, an accurate and stable method is needed to solve the forward and inverse ADR problems.

Under the advection-dominated situations, the ADR behaves more like a hyperbolic equation, whereas under the diffusion-dominated situations, the ADR behaves more like a parabolic equation. This brings some challenges to general numerical methods.[3] A variety of numerical methods can be used to solve the forward ADR equation, including Eulerian,[4,5] semi-Lagrangian,[6] and Lagrangian[7] methods. However, due to the numerical discrete calculation format, none of them is thought to be really space-time continuous. There are also works to approximate the solution by simplifying the equation. The advection–diffusion equation is simplified into a pure diffusion equation with a complex-valued potential, and then, parabolic partial differential equation (PDE) solvers are identified.[8] Recently, Heifetz and Zucker[9] pointed out that it can be equivalently described as a pressureless, compressible, and irrotational flow dynamics, whose effective force is governed by the divergence of the viscous stress tensor.

Moreover, this kind of equation can be explained by the random movement of particles.[10] The advection–diffusion equation is reinterpreted at the microscopic level, and a random differential equation controlling the behavior of individual particles of the species transported by the fluid is obtained. Then, a linearized version of the convection coefficient of the advection-dominated equation is derived and solved.[11] On the other hand, to solve the inverse problem, there are many methods such as genetic algorithms,[12,13] random walk particle tracking methods,[14] the backward probability method,[15] and the minimum relative entropy approach.[16]

However, traditional methods are computationally expensive in solving forward and inverse ADR problems, for which deep learning (DL) technology can be an attractive alternative. In the last decade, DL has made great achievements in many fields of computer science. As a general function approximator,[17] the DL method with deep neural network (DNN) as the main form shows the characteristics of versatility and flexibility in various tasks. Recently, DL technology has been applied to solve PDE.[12,18–20] Based on the governing equation, and initial and boundary conditions, which are essential in scientific computing, continuous field information is given by the trained DL PDE solver, which is the major difference from traditional numerical methods.

At the same time, automatic derivation differentiates the whole network according to the input coordinates and network parameters. The complex operator approximations in the traditional numerical calculation of PDEs are, hence, replaced.[21] Using the automatic derivation ability of Neural Network (NN), physics-informed neural network (PINN)[22,23] not only solves the forward problem according to the governing equation and initial boundary conditions but also solves the inverse problem according to the observed data. Based on the PINN structure, Hidden Fluid Mechanics (HFM)[24] was then developed to deduce the velocity and pressure fields according to the spatial and temporal distribution of passive scalar (e.g., concentration). Nowadays, many problems difficult for traditional numerical methods have been solved by using the PINN-based methods.[21,25–27]

A few works have focused on improving the accuracy of PINN from different perspectives, including residual-based adaptive refinement,[23] balance different loss terms,[28] decompose large domain,[29] and adaptive activation functions.[30] However, the results might be unstable when it is used for inverse problems with sparse noisy data. Li *et al.*[31] pointed out that using a few observation points will cause problems in the reconstruction process, which are not easy to identify. The stability of solving inverse ADR problems with PINN on sparse noisy data is a less concerned issue. In addition, the inverse problem is sensitive to grid irregularities, dispersion coefficients, and velocity gradients.[32] Therefore, factors from two aspects are considered in this work: the distribution of training points and the additional constraints for each order derivative. For the first factor, a simple idea is to use orthogonal grid (OG) points, while the points used in each training are consistent, which can reduce the fluctuation of generalization error. For the second one, the first derivative of the governing equation is obtained to generate a new governing equation to further restrict the advection term and diffusion term.

The rest of the paper is organized as follows. In Sec. II, the original PINN method for solving forward and inverse problems is introduced. In Sec. III, the specific improvements to the above two factors are presented. The improved method is named as orthogonal grid-PINN (OG-PINN) with the first derivative constraint (FDC). In Sec. IV, OG-PINN with FDC is tested by four different ADR cases, including forward and inverse problems. In Sec. V, an ADR inverse problem with observation data that is sparse with noise is implemented to test the reliability of the proposed method for real applications. Conclusions are given in Sec. VI.

II. PINN for ADR

A. Governing equations

The partial differential equations (PDEs) for advection–diffusion–reaction (ADR) problems in domain Ω with initial and boundary conditions considered in this paper are as follows:

$$\text{Governing equation: } \frac{\partial u}{\partial t} + v\frac{\partial u}{\partial x} - k\frac{\partial^2 u}{\partial x^2} = f, \quad x \in \Omega,\ t \in [0, T], \tag{1}$$

$$\text{Initial condition: } u(x,t) = u_0(x), \quad x \in \Omega,\ t = 0, \tag{2}$$

$$\text{Boundary condition: } \mathscr{B}(u,x,t) = 0, \quad x \in \partial\Omega,\ t \in [0,T], \tag{3}$$

where $u(x, t)$ is a field variable (e.g., concentration), v is the advection velocity, k is the diffusion coefficient, and f is the reaction function. The boundary conditions $\mathscr{B}(u, X, t)$ can be Dirichlet, Neumann, or Robin types.

B. PINN for forward problems

PINN is constructed by the NN structure and a physics-informed loss function, where the solution of the PDE is approximated by NN. In this paper, we choose fully connected neural networks in the original PINN to solve the ADR equation, and it is straightforward to change the network structure into other forms. Here, the neural network is defined as

$$\begin{aligned} \text{Input layer: } & \mathscr{N}^0 = X, \\ \text{Hidden layers: } & \mathscr{N}^k = \sigma(\mathbf{W}^k\mathscr{N}^{k-1} + \mathbf{b}^k), \quad \text{for} \quad 1 \le \mathrm{k} \le \mathrm{L}-1, \\ \text{Output layer: } & u_\Theta(X) := Y = \mathscr{N}^L = \sigma(\mathbf{W}^L\mathscr{N}^{L-1} + \mathbf{b}^L), \end{aligned}$$

where $\mathscr{N}^k$, $\mathbf{W}^k$, and $\mathbf{b}^k$ are the neuron, weight, and bias of the kth layer, respectively; $\sigma(\cdot)$ is the activation function; and $u_\Theta(\cdot)$ is the solution approximated by NN. The structure of the PINN is shown in Fig. 1, and its training process is divided into two steps. The input X is first transformed into the output Y by nonlinear transformation with W and b as parameters. Then, the gradient of W and b on each layer is calculated according to the objective loss function, and the optimization algorithm is used to update W and b. The above operation is repeated until the loss function is reduced below the expected threshold.

Remark 1. *In the DNN, as the parameters are randomly initialized, the training results cannot be guaranteed to be globally optimal. This will bring some uncertainty to the PINN prediction, which is considered to be the instability of the model. The* ***mean*** *and* ***standard deviation*** *of error between multiple prediction results and the exact solution will be used in this paper as the standard to evaluate the* ***stability*** *of the methods.*

1. Physics-informed loss

In the DNN, the losses are used to measure the difference between $u_\Theta(\cdot)$ and $u(\cdot)$. The PINN is implemented by imposing two

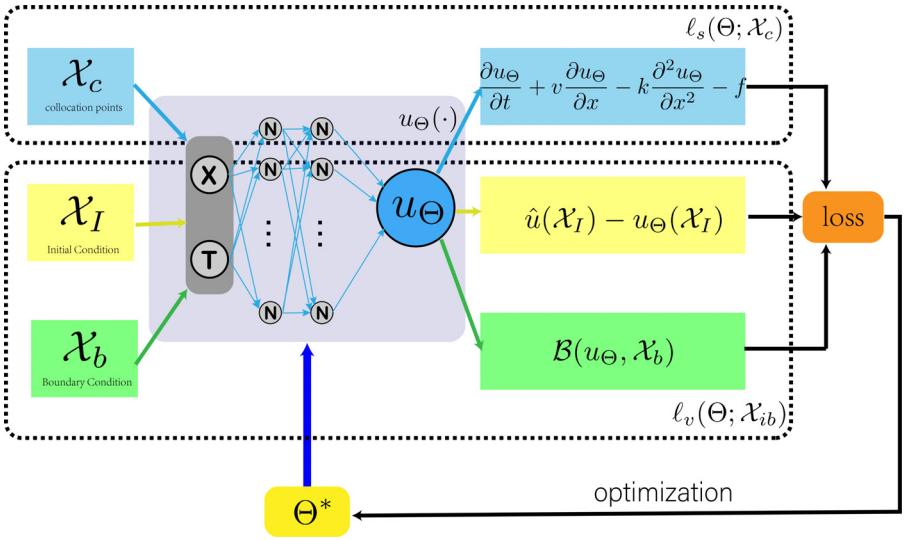

FIG. 1. Structure of the PINN for solving a partial differential equation $u_t + vu_x - ku_{xx} - f = 0$ with initial and boundary conditions.

types of loss. One is the ℓ_s controlled by the collocation points $\mathcal{X}_c$ and the other is the ℓ_v calculated on the initial-boundary points $\mathcal{X}_{ib}$. Consequently, the PINN classifies the training points into two categories. One kind is the points in the space-time domain, which are called collocation points. This kind of points provides control constraints to the equation. The other kind is the initial-boundary points, which provide value constraint. Different from traditional numerical methods, value constraint is used for PINN to train NN to fit the initial and boundary conditions, which means that there are errors in the prediction of the initial and boundary conditions. Two kinds of points are selected during the initialization process, the positions of which remain unchanged in each training. The training dataset $\mathcal{X}$ is composed of $\mathcal{X}_c \subset \Omega \times [0, T]$ and $\mathcal{X}_{ib} \subset \Omega \times [0, T]$, i.e., $\mathcal{X} = \mathcal{X}_c \cup \mathcal{X}_{ib}$.

The discrepancy between $u_\Theta(\cdot)$ and $u(\cdot)$ is measured on the collocation points and boundary points using the loss function defined by the L_2 norm as

$$\ell(\Theta;\mathcal{X}) = \omega_s \ell_s(\Theta;\mathcal{X}_c) + \omega_v \ell_v(\Theta;\mathcal{X}_{ib}), \tag{4}$$

where

$$\begin{aligned} \ell_s(\Theta;\mathcal{X}_c) &= \frac{1}{|\mathcal{X}_c|}\sum_{X\in\mathcal{X}_c}\left\|\frac{\partial u_\Theta}{\partial t}+v\frac{\partial u_\Theta}{\partial x}-k\frac{\partial^2 u_\Theta}{\partial x^2}-f\right\|_2, \\ \ell_v(\Theta;\mathcal{X}_{ib}) &= \frac{1}{|\mathcal{X}_{ib}|}\sum_{X\in\mathcal{X}_{ib}}||\mathcal{B}(u_\Theta,\mathcal{X}_{ib})||_2, \end{aligned} \tag{5}$$

in which $|\mathcal{X}_c|$ and $|\mathcal{X}_{ib}|$ represent the number of points in $\mathcal{X}_c$ and $\mathcal{X}_{ib}$, and ω_s and ω_v are the weights. The above expressions can be calculated quickly due to the automatic derivation property of the NN.

Remark 2. *In PINN, the training results are affected by the selection of points in the initialization process. Although the locations of the collocation points are generated using a space-filling Latin Hypercube Sampling (LHS) strategy*[33] *and initial-boundary points are selected randomly in initialization, the final neural network solution $u_\Theta(\cdot)$ will be different if different initialization is applied.*[23] *This may also cause instability.*

The optimal model parameters Θ^* are obtained by minimizing the loss function $\ell(\Theta;\mathcal{X})$. In this paper, the Adam optimization algorithm[34] is used to avoid the training process falling into the local optimum.

C. PINN for inverse problems

PINN can also be used for inverse problems. With little modifications, the parameters of PDEs and the initial distribution can be learned. For inverse problems, observed data are required and the loss function should also be modified.

Unlike solving PDEs, inverse problems no longer require initial-boundary values but apply additional information about the equations in the space-time domain. The information here is provided by the distribution of concentrations $\hat{u}$ and the spatiotemporal position (x, t) in the observed data. These points make up the point set $\mathcal{X}_v \subset \Omega \times [0, T]$.

The loss function is the same as before, but ℓ_v is no longer calculated on $\mathcal{X}_{ib}$, but on $\mathcal{X}_v$

$$\begin{aligned} \ell_s(\Theta;\mathcal{X}_c) &= \frac{1}{|\mathcal{X}_c|}\sum_{X\in\mathcal{X}_c}\left\|\frac{\partial u_\Theta}{\partial t}+v_\theta\frac{\partial u_\Theta}{\partial x}-k_\theta\frac{\partial^2 u_\Theta}{\partial x^2}-f\right\|_2, \\ \ell_v(\Theta;\mathcal{X}_v) &= \frac{1}{|\mathcal{X}_v|}\sum_{X\in\mathcal{X}_v}||\hat{u}(\mathcal{X}_v)-u_\Theta(\mathcal{X}_v)||_2, \end{aligned} \tag{6}$$

where v_θ, k_θ are the model parameters to be learned.

During the training process, by changing the unknown parameters v and k in the equations into the trainable parameters v_θ and k_θ in PINN, the inference of missing physics is realized. At the same time, the solution of the PDE that fits the data is also obtained.

III. IMPROVEMENTS

A. Point selection method in space-time domain

Using LHS[33] for collocation points and random selection for initial-boundary points can solve basic differential equations and has shown strong robustness in various tasks for solving PDEs by PINN. One major drawback of this approach is that the random selection of the training points will lead to large differences in model predictions

between two independent pieces of training. Especially when the PDE contains complicated initial boundary conditions, the precision of its solution will be greatly reduced. Therefore, to improve the stability of the model, a new method for point selection is needed.

Inspired by the traditional numerical methods for PDEs, orthogonal grids for collocation points and uniform dividing for initial-boundary points are used to improve the fitting effect of the NN and the stability. The following selection strategies for collocation and initial-boundary points are studied:

Strategy 1: Use the LHS method for collocation points and the random selection method for initial-boundary points.

Strategy 2: Use the LHS method for collocation points and uniform selection method for initial-boundary points.

Strategy 3: Use orthogonal grids for collocation points and a uniform selection method for initial-boundary points.

Remark 3. *The prediction is tested on different orthogonal grids, the grid spacing of which is generally different from that used in the training process. This means that most points used in the test process are not the same as those in the training process.*

The training points selected by three strategies are displayed in Fig. 2. PINN with strategy 3 is referred to as the orthogonal grid-PINN (OG-PINN). For collocation points, the orthogonal grid can uniformly cover the space-time domain. Compared with the LHS method in

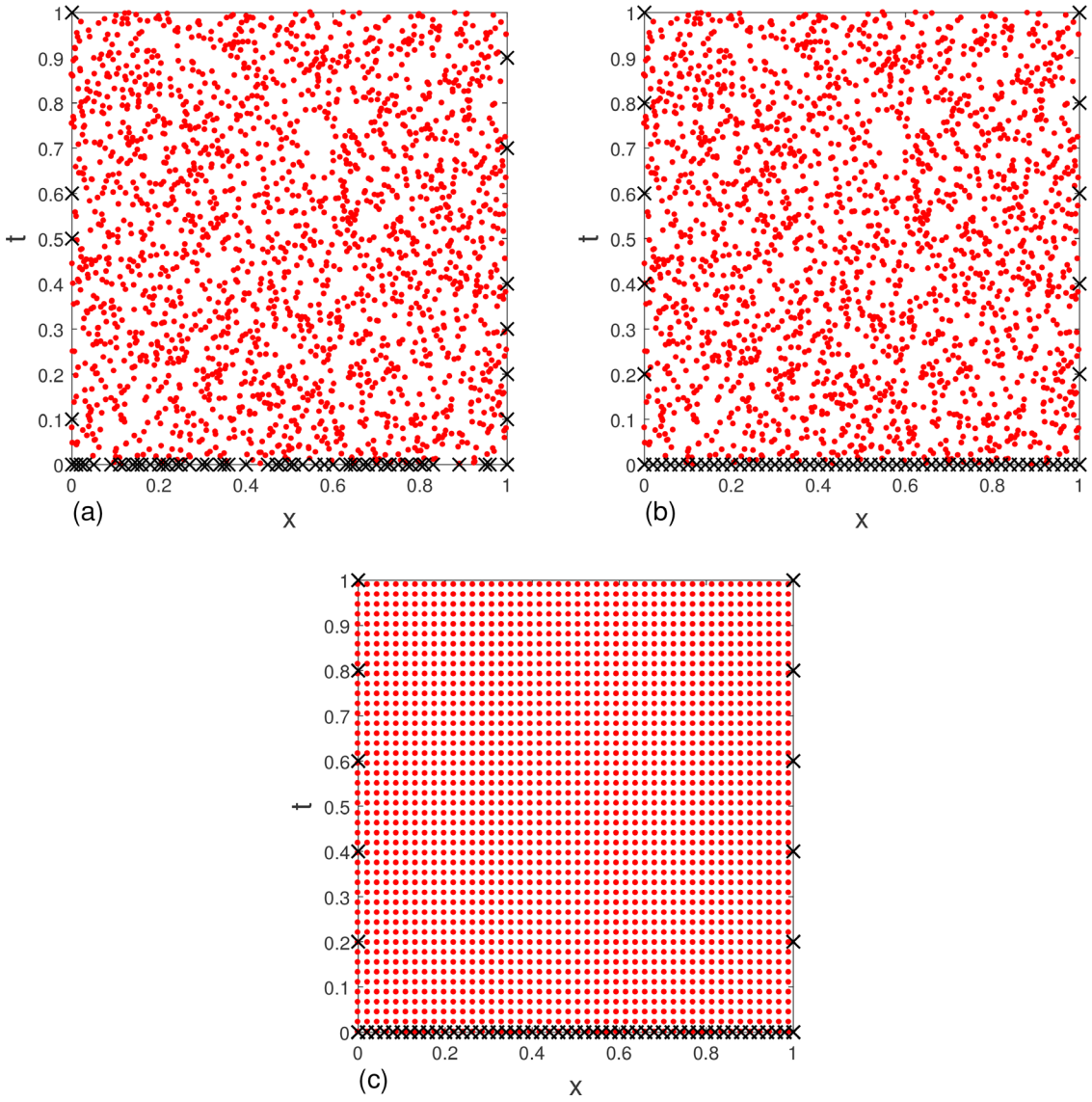

FIG. 2. Strategies for point selection. The red points represent collocation points in the space-time domain, and the black "×" represents initial-boundary points. (a) Strategy 1: use LHS for collocation points and random for initial-boundary points; (b) strategy 2: use LHS for collocation points and uniform for initial-boundary points; (c) strategy 3: use orthogonal grids for collocation points and uniform for initial-boundary points. All these three strategies have the same number of collocation points and initial-boundary points.

which random points are selected in each initialization process, the positions of orthogonal grids are fixed among each training process. At the same time, for problems with complex boundary conditions, the use of uniformly distributed initial-boundary points reduces the fitting errors and decreases the instability caused by the random selection of initial-boundary points.

B. First derivative constraint (FDC)

For general basic differential equations, the original PINN can obtain good accuracy with only the initial and boundary conditions. However, for complex differential equations, high-order derivative terms without constraints are difficult to be well fitted. Moreover, the ADR problems are sensitive to changes in each process. Therefore, it is necessary to introduce new constraints into the NN training process to improve the accuracy of the fitted solution of the governing equation.

For equations with advection, diffusion, and reaction terms, only the initial and boundary values are provided, without knowing any derivative information in the space-time domain. To better solve this problem, a new constraint on the derivative needs to be added to the loss function. Considering that there are no additional high-order derivatives, the first derivative of the governing equation is used as a constraint to the second-order derivative, i.e., diffusion term.

By adding the first derivative of the governing equation as an additional constraint to the algorithm, the loss function is

$$\ell_{f_x}(\Theta;\mathscr{X}_f)=\frac{1}{|\mathscr{X}_f|}\sum_{X\in\mathscr{X}_f}\left\|\frac{\partial\left(\frac{\partial u_\Theta}{\partial t}+v\frac{\partial u_\Theta}{\partial x}-k\frac{\partial^2 u_\Theta}{\partial x^2}-f\right)}{\partial x}\right\|_2, \tag{7}$$

where $\mathscr{X}_f$ is the point set for calculating the first derivative constraint of the governing equation. This constraint is named as the first derivative constraint (FDC). In this paper, we take $\mathscr{X}_f=\mathscr{X}_c$.

Remark 4. *Other point set can be chosen for $\mathscr{X}_f$, and $\mathscr{X}_f=\mathscr{X}_c$ is selected here just for the convenience of calculation.*

The loss function (4) in the training process becomes

$$\ell(\Theta;\mathscr{X})=\omega_s\ell_s(\Theta;\mathscr{X}_c)+\omega_{f_x}\ell_{f_x}(\Theta;\mathscr{X}_f)+\omega_v\ell_v(\Theta;\mathscr{X}_{ib}). \tag{8}$$

Carrying out the above constraints not only provides an additional high-order derivative constraint but also smoothes the fitted solution of the governing equation. A smooth trained governing equation means accurate prediction and good stability of the model.

IV. NUMERICAL RESULTS AND DISCUSSION

In this section, four examples are tested to demonstrate the effectiveness of the improvements proposed in Sec. III. The selected examples are likely to cause spurious oscillations or numerical dissipation for traditional grid methods. In all the tests, the activation function is tanh, the number of hidden layers in the network is five, and the number of neurons in each layer is 50. The weights ω_s and ω_{f_x} are one, and ω_v is 0.001, in order to ensure that each part of the loss is in the same order of magnitude. The Adam method[34] is used as the optimization algorithm in the training process with the learning rate of 0.001.

A. Steep tracer

In this case, the advection–diffusion equation with steep distribution is considered. The governing equation for this case is

$$\frac{\partial c}{\partial t}+v\frac{\partial c}{\partial x}=k\frac{\partial^2 c}{\partial x^2},\quad x\in[0,10],\quad t\in[0,t_{max}], \tag{9}$$

with the initial condition

$$\begin{cases}c(x,0)=1, & 0\le x\le b,\\ c(x,0)=0, & \text{otherwise}\end{cases} \tag{10}$$

and boundary conditions:

$$c(0,t)=1,\quad c(2.6,t)=0, \tag{11}$$

where b defines the position of the steep.

The exact solution of this case is

$$c(x,t)=\frac{c_0}{2}erfc\left(\frac{x-b-vt}{\sqrt{4kt}}\right)+\frac{c_0}{2}erfc\left(\frac{x-b+vt}{\sqrt{4kt}}\right)\exp\left(\frac{v(x-b)}{k}\right). \tag{12}$$

In this test, we take $c_0=1$ g/m, $v=0.5$ m/s, $k=0.005$ m^2/s, and $b=0.1$ m. In Fig. 3, the results of OG-PINN with FDC are compared with the exact and the original PINN solutions at three time levels of

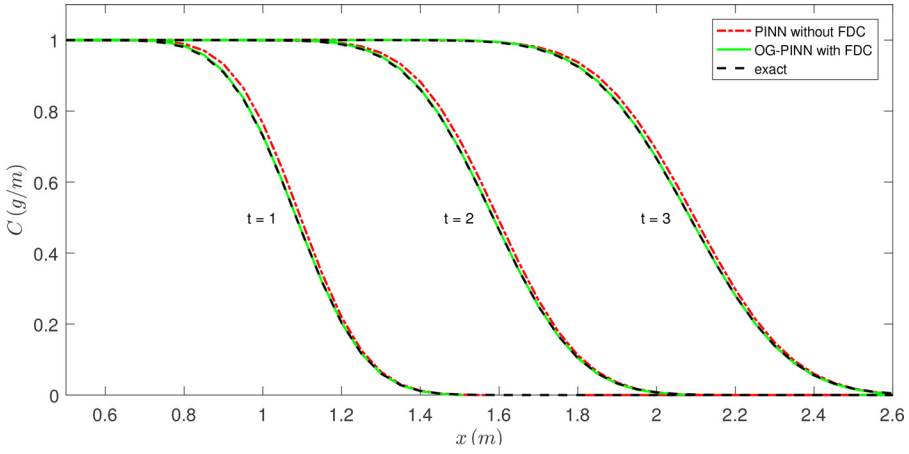

FIG. 3. Comparison of numerical results obtained by original PINN, OG-PINN with FDC, and exact solution.

TABLE I. Experimental mean and standard deviation of the error in the steep distribution case.

Case	Strategy 1	Strategy 2	Strategy 3
Without FDC	0.041 84 ± 0.036 22	0.033 99 ± 0.020 68	0.041 59 ± 0.015 66
With FDC	0.029 13 ± 0.014 61	0.021 65 ± 0.020 66	0.003 19 ± 0.001 32

$t = 1$, 2, and 3 s, respectively. Both the original PINN and the OG-PINN with FDC can give correct solutions, but the original PINN shows a misfit at the position of the steep.

Table I presents the numerical results of PINN with/without FDC using three point selection methods defined in Sec. III A. Compared with the original PINN, by better fitting the initial and boundary values, the method of using uniform initial-boundary points effectively improves the accuracy. The use of orthogonal grids largely reduces the fluctuations in the forecast. For all three selection methods, the use of FDC improves accuracy and reduces fluctuations. Both the mean and standard deviation of the error in OG-PINN with FDC are one magnitude lower than those in the original PINN.

To further demonstrate the effect of FDC, the loss of the governing equation and its first derivative are compared in Fig. 4. By comparing the losses, it can be seen that the method of adding FDC can effectively smoothen the derivative of the governing equation fitted by OG-PINN and, hence, reduce the error.

B. Advection–diffusion case with location-sensitive source term

In this case, to further demonstrate the effectiveness of the developed method, we consider the steady-state location-sensitive advection–diffusion equation with both Dirichlet and Neumann boundary conditions. The governing equation for this case is

$$(vc - kc')' = s, \quad 0 \le x \le 1, \tag{13}$$

with boundary conditions

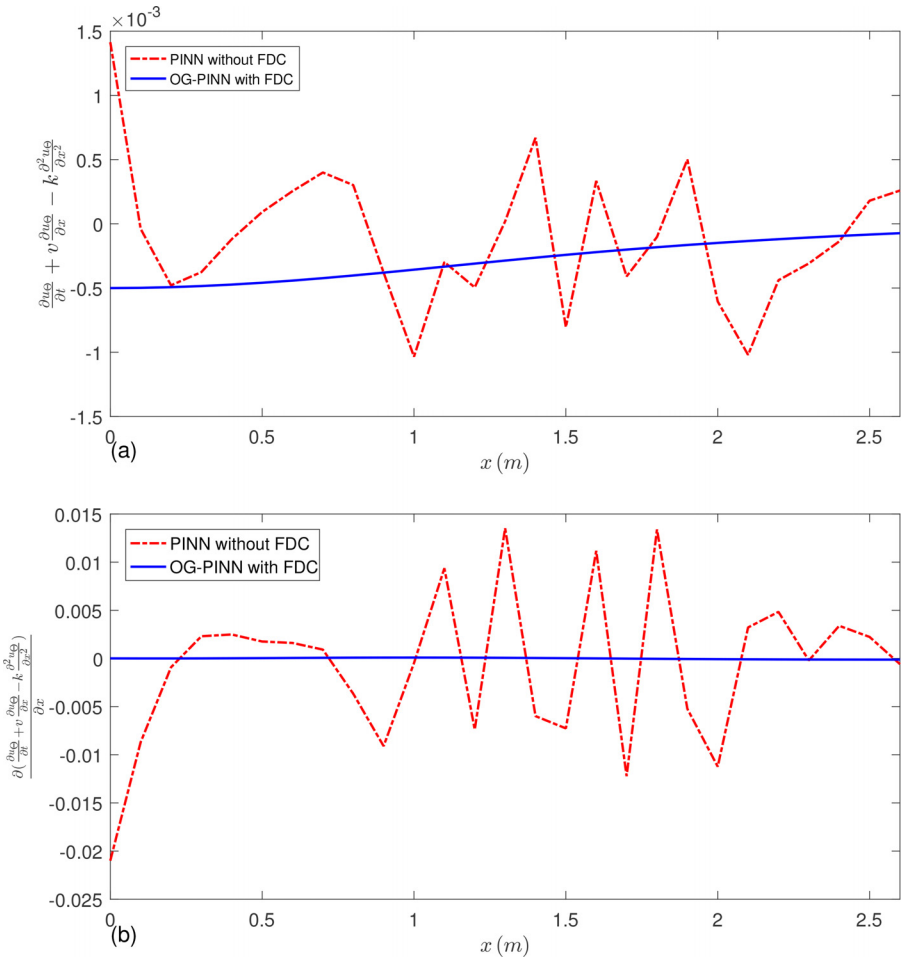

FIG. 4. The error in the trained (a) governing equation and (b) its first-derivative for the steep tracer problem.

$$c(0) = c'(1) = 0, \tag{14}$$

where v, s, and k are given by

$$v(x) = (1+x)^3, \quad s(x) = \frac{S_{max}}{1 + S_{max}(2x-1)^2}, \quad S_{max} = 10^2, \quad k = 10^{-5}. \tag{15}$$

As this case has no exact solution, the high-precision reference solution obtained by Ten Thije Boonkkamp and Anthonissen[35] is used as a benchmark for comparison.

Figure 5(a) shows the results of the original PINN, the OG-PINN with FDC, and the reference solutions. Through comparison, we find that when the number of points selected in the space-time domain is 20, the OG-PINN with FDC can correctly predict the results, but the original PINN has large prediction errors. In Fig. 5(b), the predicted results of the original PINN using different numbers of collocation points are compared. It is found that the agreement between the predicted results and the reference solution improves with the increasing number of collocation points. This means that for location-sensitive advection–diffusion problems, OG-PINN with FDC can reduce the number of collocation points required in the training process, to save computing overhead.

C. ADR case

In this case, a steady-state ADR is tested. The governing equation for this case is

$$\lambda \frac{\partial c}{\partial x} = \frac{\partial^2 c}{\partial x^2} - \phi^2 c, \quad x \in [0,1], \tag{16}$$

with boundary conditions

$$c(0) = c(1) = 1. \tag{17}$$

The exact solution of this case is

$$c(x) = \frac{\exp(-m_1 + m_1 x) - \exp(m + m_1 x) + \exp(m_2 x) - \exp(m_2 x - m_1)}{1 - \exp(m)}, \tag{18}$$

where m, m_1, and m_2 are given by

$$m = -\sqrt{\lambda^2 + 4\phi^2}, \quad m_1 = \frac{\lambda + \sqrt{\lambda^2 + 4\phi^2}}{2}, \quad m_2 = \frac{\lambda - \sqrt{\lambda^2 + 4\phi^2}}{2}. \tag{19}$$

In this test, we take $\lambda = 1, 5, 10$ and $\phi = 2, 5$. In Fig. 6, the results of OG-PINN with FDC are compared with the exact and the original PINN solutions. Both the original PINN and the OG-PINN with FDC can give correct solutions.

Table II presents the numerical results of PINN with/without FDC using two point selection methods defined in Sec. III A. For this case, there is no difference between strategy 1 and strategy 2; therefore, only two strategies are compared in Table II. Therefore, compared with the original PINN, the use of orthogonal grids largely reduces the fluctuations in the forecast. It is seen that both the mean and standard deviation of the error in OG-PINN with FDC are lower than those in the original PINN.

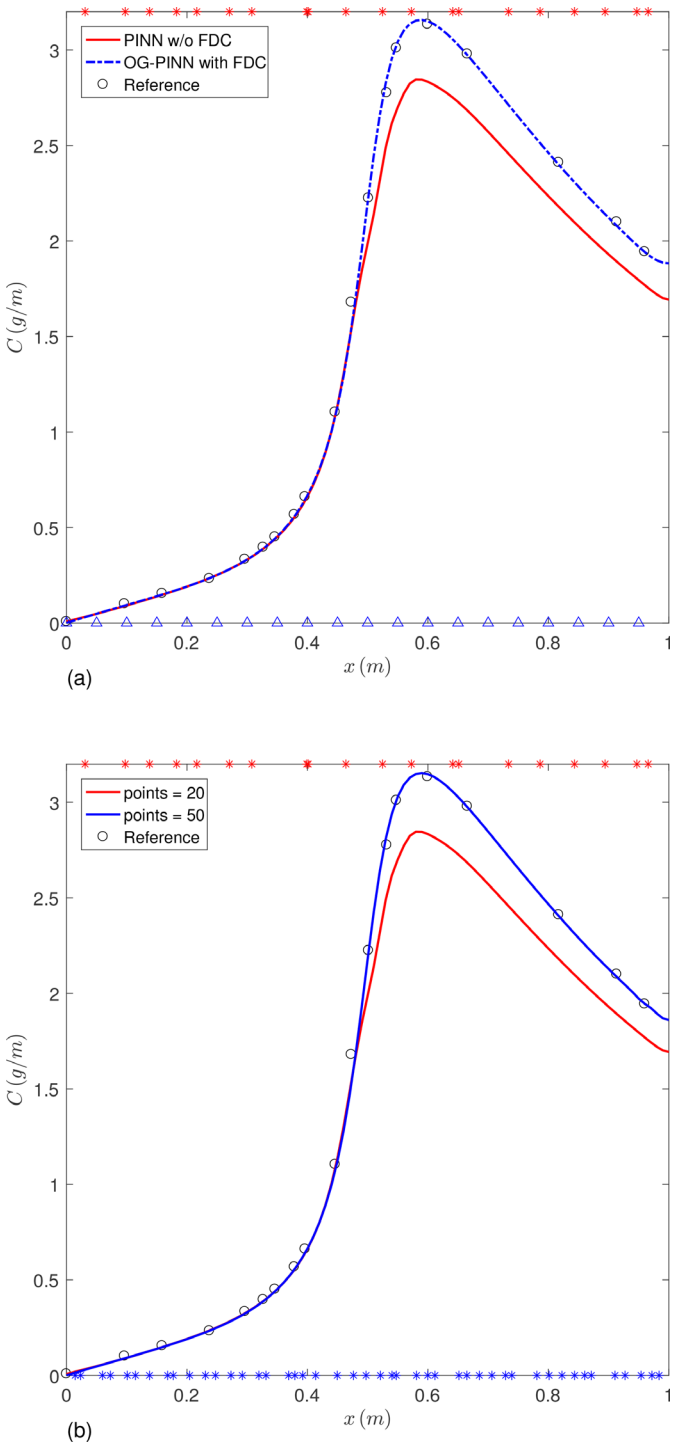

FIG. 5. Validation of the solutions trained by original PINN and OG-PINN with FDC against the reference solution[35] for the location-sensitive advection–diffusion problem. (a) Comparison of the two point selection methods with 20 collocation points (those on the coordinate axis represent the boundary points selected by this method in the training process) and (b) comparison of the results with a different number of collocation points used in the original PINN.

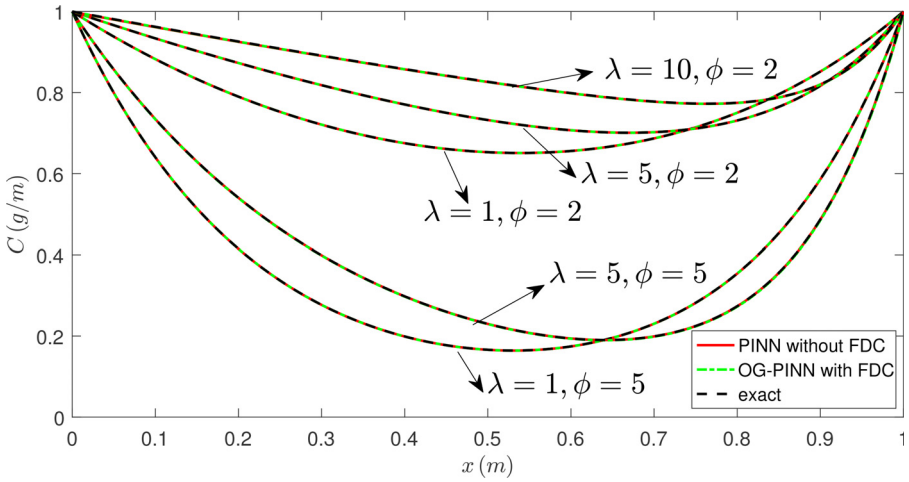

FIG. 6. Comparison of the results obtained by original PINN, OG-PINN with FDC, and exact solution.

It is clear that the use of uniform distribution on the initial and boundary can effectively strengthen the NN's fitting of the initial boundary, the use of orthogonal grids in the space-time domain can greatly reduce the fluctuations in the prediction, and the use of FDC can provide strong constraint for the training process.

D. Gaussian distribution inverse problems

In this case, the inverse problems of the advection–diffusion equation with Gaussian distribution are tested. For inverse problems, two main issues are investigated: model parameter learning and initial distribution prediction. Model parameter learning is used to infer missing physics in the ADR equation from the data. Initial distribution prediction is used to trace the source of pollutants.

The governing equation, for this case, is

$$\frac{\partial c}{\partial t} + \frac{\partial c}{\partial x} = k_\theta \frac{\partial^2 c}{\partial x^2}, \quad x \in [0,1], t \in [0,1], \tag{20}$$

where k_θ is the model parameter that needs to be learned. At the same time, the initial distribution of the equation also needs to be predicted. In this case, there is no difference between strategy 1 and strategy 2; therefore, only two strategies are compared.

Different from the inverse problems learning condition set in the original PINN: randomly obtaining points in the space-time domain, we consider an arrangement that is closer to the actual situation. We arrange monitoring points at four positions (0.2, 0.4, 0.6, 0.8), and the monitoring time interval is 0.05 s. The total number of the data pairs $\langle x, t, \hat{u} \rangle$ is 80, which means that the observed data are sparse, and the initial distribution is unknown. In this case, we take $k_\theta = 0.05$. For different training strategies, we use the same training data and repeat the training ten times for demonstration.

TABLE II. Experimental mean and standard deviation of the error in the ADR case.

Case	Strategy 1	Strategy 3
Without FDC	0.000 46 ± 0.000 64	**0.000 22 ± 0.000 16**
With FDC	0.000 73 ± 0.001 28	**0.000 14 ± 0.000 11**

Remark 5. *The amount of data used in the training process is small for neural network methods, but considering the limitations of real-world measurement situations, it is necessary to choose a reasonable sensor monitoring location and sampling frequency. We tested other numbers of monitoring points and found that four is the minimum value for a favorable training.*

To approximate the real test environment, concentration-independent Gaussian additive noise is added to the observed data. Three different levels of noise are considered: without noise, the noise amplitude is 0.05% (1% of k_θ), and the noise amplitude is 1% (20% of k_θ). For model parameters, the error is used to demonstrate the learning effect. For the initial value distribution, the predicted concentration distribution is used to demonstrate the effectiveness of the method.

First, for model parameter learning, Table III presents the model parameter prediction error of PINN with/without FDC using two point selection methods. When the noise is zero, both strategies give correct model parameter predictions. The use of both improvements enhances the accuracy of parameter learning and reduces fluctuations.

The prediction of the initial distribution is shown in Fig. 7. Both strategies correctly predict the initial distribution. By comparing the prediction regions formed by ten independent training processes, the prediction fluctuation of OG-PINN with FDC is smaller than the original PINN.

Second, when the noise is 0.05%, both strategies perform better in model parameter learning than when the noise is zero. Table IV presents the model parameter error of PINN with/without FDC using two point selection methods. At this time, the use of orthogonal grids reduces the fluctuations, and the use of FDC improves accuracy and reduces fluctuations.

TABLE III. Experimental mean and standard deviation of the error of the model parameter in the AD Gaussian distribution case without noise.

Case	Strategy 1	Strategy 3
Without FDC	0.033 77 ± 0.012 48	**0.029 03 ± 0.009 81**
With FDC	**0.025 78** ± 0.007 65	0.034 37 ± **0.006 68**

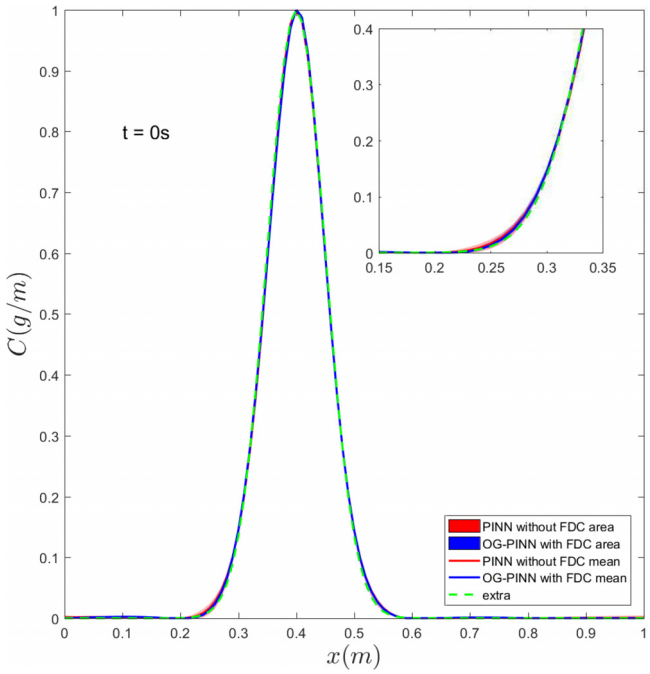

FIG. 7. Comparison of the initial distribution predicted by the original PINN and OG-PINN with FDC and the exact solution without noise.

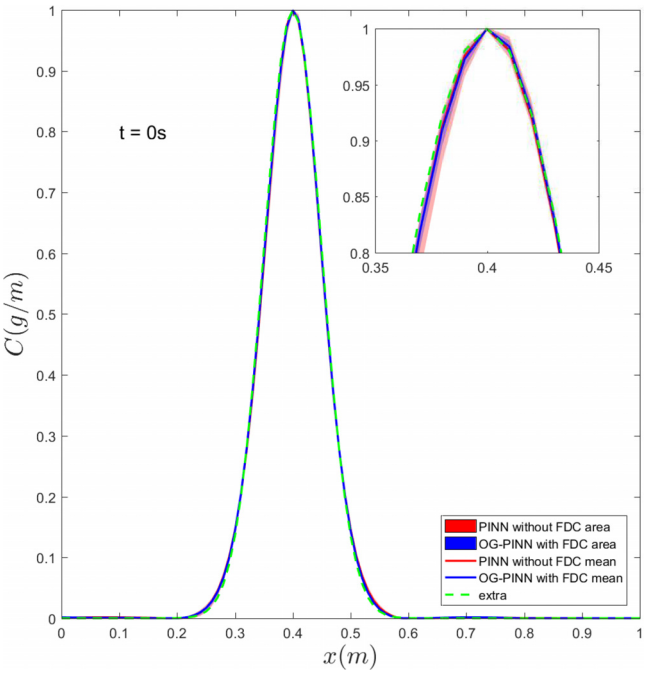

FIG. 8. Comparison of the initial distribution predicted by the original PINN and OG-PINN with FDC and the exact solution when the noise is 0.05%.

Figure 8 shows the initial distribution predicted by the original PINN and the OG-PINN with FDC together with the exact initial distribution. It is obvious that with the emergence of noise, the prediction of the original PINN has fluctuation compared to the OG-PINN with FDC.

Third, when the noise is 1%, both strategies perform better in model parameter learning than when the noise is small. Table V presents the model parameter error of PINN with/without FDC using two point selection methods. At this time, the use of orthogonal grids reduces the fluctuations, and the use of FDC improves accuracy and reduces fluctuations.

Figure 9 shows the initial distribution predicted by the original PINN and the OG-PINN with FDC together with the exact initial distribution. When the noise is 1%, the obtained results are considered acceptable by comparing the predicted center position, magnitude, and phase of the initial distribution.

As long as the error is controllable, although the error of the prediction for the initial distribution becomes larger as the noise in the observed data becomes larger, the results are still acceptable. At the same time, learning about model parameters is accurate.

TABLE IV. Experimental mean and standard deviation of the error for the model parameter in the AD Gaussian distribution case when the noise is 0.05%.

Case	Strategy 1	Strategy 3
Without FDC	**0.024 75** ± 0.010 85	0.025 98 ± **0.006 84**
With FDC	0.020 34 ± 0.007 04	**0.018 75** ± **0.005 49**

V. ADR INVERSE PROBLEMS

The above discussion of ADR forward and inverse problems shows that under the same conditions, the OG-PINN with FDC is more accurate and stable than the original PINN, with smaller errors and lower variance. In this section, the developed OG-PINN with FDC is further applied to an ADR case with noise data, which can verify the accuracy of the developed method for practical applications.

In this case, an inverse problem for steady-state ADR similar to that studied in Sec. IV C is tested. The governing equation is

$$\hat{\lambda}\frac{\partial c}{\partial x} = \frac{\partial^2 c}{\partial x^2} - \hat{\phi}c, \quad x \in [0,1], \tag{21}$$

where $\hat{\lambda}$ and $\hat{\phi}$ are the two model parameters to be identified. The boundary conditions are the same as (17).

For this case, two main issues are shown: the model parameters learning process and the steady-state distribution. Consistent with the previous considerations, only four points of concentration with 1% noise are used. The exact model parameters are $\hat{\lambda} = 10$ and $\hat{\phi} = 4$, which means that this equation is under the advection-dominated situation, and it becomes steep close to the right end. In this case, it is

TABLE V. Experimental mean and standard deviation of the error of the model parameter in the AD Gaussian distribution case when the noise is 1%.

Case	Strategy 1	Strategy 3
Without FDC	0.022 35 ± 0.011 66	**0.016 19** ± **0.008 87**
With FDC	0.011 69 ± 0.010 66	**0.011 12** ± **0.008 80**

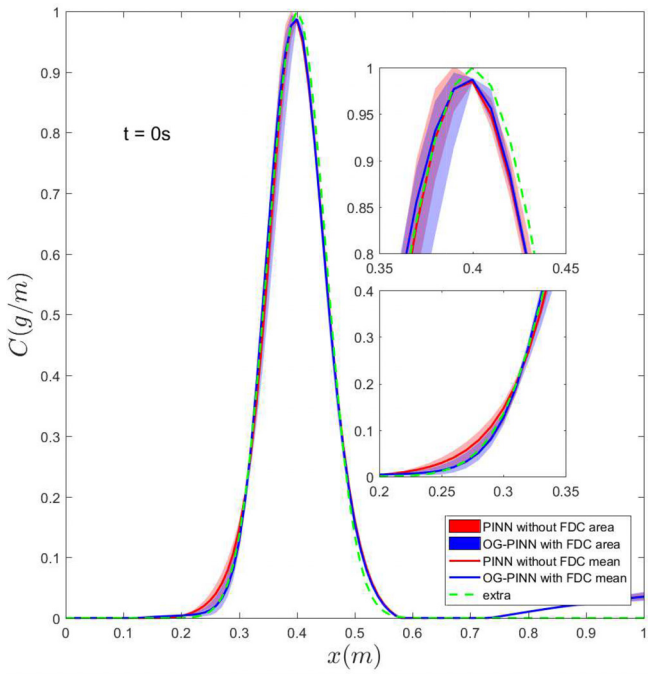

FIG. 9. Comparison of the initial distribution predicted by the original PINN and OG-PINN with FDC and the exact solution when the noise is 1%.

difficult to solve the forward problem because it is an intense ADR system. Coupled with the sparse observed data, small noise may cause large errors, so this inverse case is considered a representative of a real practice with low-quality data.

OG-PINN with FDC successfully reconstructs the ADR system as shown in Fig. 10. Even if the data are far from the exact value, the responding peak position is in good agreement with the exact solution.

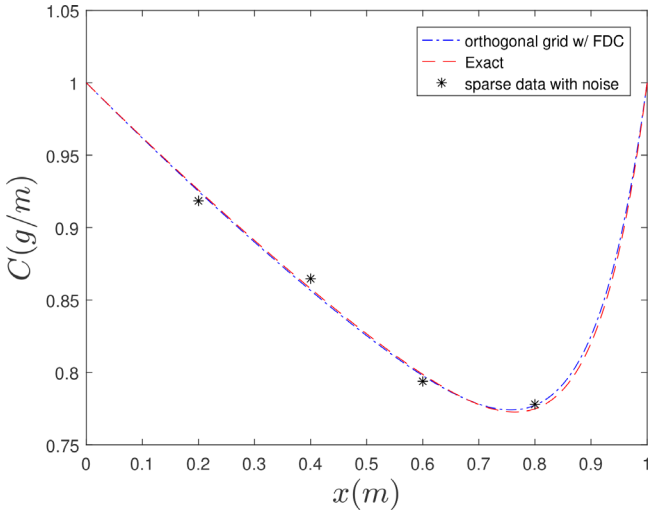

FIG. 10. Comparison of the steady-state distribution predicted by OG-PINN with FDC and the exact solution with noise.

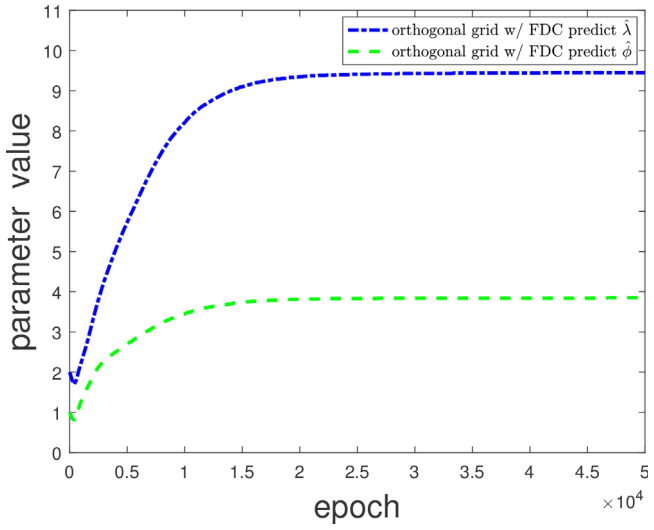

FIG. 11. Parameter prediction process by OG-PINN with FDC.

The learning process of model parameters is shown in Fig. 11. With the training, the predicted model parameters gradually approach the accurate values.

VI. CONCLUSIONS

In this paper, the orthogonal grid PINN (OG-PINN) with the first derivative constraint (FDC) has been presented. The main conclusions are as follows:

For the forward problems, the strategy of using orthogonal grids instead of LHS for collocation points and uniform instead of random for initial-boundary points is demonstrated as an effective method for point selection in PINN to solve the ADR equation. An additional high-order derivative constraint is added to the loss function, which differentiates the governing equation to improve the accuracy of each term in the governing equation. The effectiveness of these improvements is verified by numerical experiments. Both the mean and standard deviation of the error in OG-PINN with FDC are lower than those in the original PINN. These improvements have improved the accuracy and stability of the model without increasing the need for additional data during the training process.

For the inverse problems, the effectiveness of the improved strategy is verified by simulating monitored data similar to real scenarios. Even with sparse and noisy observed data, PINN is still able to learn model parameters and predict initial distributions. Compared with the original PINN, OG-PINN with FDC can reduce the error and maintain the stability of model parameters learning and initial distribution prediction when noise appears in the observed data.

There are also some limitations. Although FDC can effectively improve the stability and accuracy of the proposed OG-PINN, it also increases the training time. To reduce the computation cost, a better way to add FDC will be considered by choosing more suitable training points. The extension of the OG-PINN with FDC to nonlinear ADR systems and high-dimensional ADR problems also deserves separate investigations.

ACKNOWLEDGMENTS

This study has been partly funded by the National Key Research and Development Program of China (Grant No. 2020YFC1807905); the National Natural Science Foundation of China (Grant No. 52079090); and the Basic Research Program of Qinghai Province (Grant No. 2022-ZJ-704).

AUTHOR DECLARATIONS

Conflict of Interest

The authors have no conflicts to disclose.

Author Contributions

Qingzhi Hou: Conceptualization (lead); Project administration (lead); Resources (equal); Writing – review and editing (equal). **Zewei Sun:** Investigation (equal); Writing – original draft (equal). **Li He:** Resources (equal). **Alireza Keramat:** Writing – review and editing (equal).

DATA AVAILABILITY

The data that support the findings of this study are available from the corresponding author upon reasonable request.

REFERENCES

[1]E. Essouayed, E. Verardo, A. Pryet, R. L. Chassagne, and O. Atteia, "An iterative strategy for contaminant source localisation using GLMA optimization and Data Worth on two synthetic 2D Aquifers," J. Contam. Hydrol. **228**, 103554 (2020).
[2]M. Barati Moghaddam, M. Mazaheri, and J. Mohammad Vali Samani, "Inverse modeling of contaminant transport for pollution source identification in surface and groundwaters: A review," Groundwater Sustainable Dev. **15**, 100651 (2021).
[3]L. A. Khan and P. L. Liu, "An operator splitting algorithm for the three-dimensional advection-diffusion equation," Int. J. Numer. Methods Fluids **28**, 461–476 (1998).
[4]Y. Wang and K. Hutter, "Comparisons of numerical methods with respect to convectively dominated problems," Int. J. Numer. Methods Fluids **37**, 721–745 (2001).
[5]K. Alhumaizi, "Comparison of finite difference methods for the numerical simulation of reacting flow," Comput. Chem. Eng. **28**, 1759–1769 (2004).
[6]D. L. Stefanovic and H. G. Stefan, "Accurate two-dimensional simulation of advective-diffusive-reactive transport," J. Hydraul. Eng. **127**, 728–737 (2001).
[7]B. H. Devkota and J. Imberger, "Lagrangian modeling of advection-diffusion transport in open channel flow," Water Resour. Res. **45**, 1–14, https://doi.org/10.1029/2009WR008364 (2009).
[8]N. Karedla, J. C. Thiele, I. Gregor, and J. Enderlein, "Efficient solver for a special class of convection-diffusion problems," Phys. Fluids **31**, 023606 (2019).
[9]E. Heifetz and S. Zucker, "Fluid-like representation of Fickian diffusion," Phys. Fluids **34**, 011701 (2022).
[10]E. M. LaBolle, J. Quastel, and G. E. Fogg, "Diffusion theory for transport in porous media: Transition-probability densities of diffusion processes corresponding to advection-dispersion equations," Water Resour. Res. **34**, 1685–1693, https://doi.org/10.1029/98WR00319 (1998).
[11]Y. Sun, A. S. Jayaraman, and G. S. Chirikjian, "Approximate solutions of the advection–diffusion equation for spatially variable flows," Phys. Fluids **34**, 033318 (2022).
[12]S. Leichombam and R. K. Bhattacharjya, "Identification of unknown groundwater pollution sources and determination of optimal well locations using ANN-GA based simulation-optimization model," J. Water Resour. Prot. **08**, 411–424 (2016).
[13]S. Zhang and X. Xin, "Pollutant source identification model for water pollution incidents in small straight rivers based on genetic algorithm," Appl. Water Sci. **7**, 1955–1963 (2017).
[14]G. Sole-Mari, D. Fernàndez-Garcia, P. Rodríguez-Escales, and X. Sanchez-Vila, "A KDE-based random walk method for modeling reactive transport with complex kinetics in porous media," Water Resour. Res. **53**, 9019–9039, https://doi.org/10.1002/2017WR021064 (2017).
[15]H. T. Hwang, S. W. Jeen, D. Kaown, S. S. Lee, E. A. Sudicky, D. T. Steinmoeller, and K. K. Lee, "Backward probability model for identifying multiple contaminant source zones under transient variably saturated flow conditions," Water Resour. Res. **56**, 1–20, https://doi.org/10.1029/2019WR025400 (2020).
[16]F. Cupola, M. G. Tanda, and A. Zanini, "Contaminant release history identification in 2-D heterogeneous aquifers through a minimum relative entropy approach," SpringerPlus **4**, 1–19 (2015).
[17]K. Hornik, M. Stinchcombe, and H. White, "Multilayer feedforward networks are universal approximators," Neural Networks **2**, 359–366 (1989).
[18]L. Lyu, K. Wu, R. Du, and J. Chen, "Enforcing exact boundary and initial conditions in the deep mixed residual method," arXiv:2008.01491 (2020).
[19]B. Liu, J. Tang, H. Huang, and X. Lu, "Deep learning methods for super-resolution reconstruction of turbulent flows," Phys. Fluids **32**, 025105 (2020).
[20]Z. Long, Y. Lu, and B. Dong, "PDE-Net 2.0: Learning PDEs from data with a numeric-symbolic hybrid deep network," J. Comput. Phys. **399**, 108925 (2019).
[21]G. E. Karniadakis, I. G. Kevrekidis, L. Lu, P. Perdikaris, S. Wang, and L. Yang, "Physics-informed machine learning," Nat. Rev. Phys. **3**, 422–440 (2021).
[22]M. Raissi, P. Perdikaris, and 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).
[23]L. Lu, X. Meng, Z. Mao, and G. E. Karniadakis, "DeepXDE: A deep learning library for solving differential equations," SIAM Rev. **63**, 208–228 (2021).
[24]M. Raissi, A. Yazdani, and G. E. Karniadakis, "Hidden fluid mechanics: Learning velocity and pressure fields from flow visualizations," Science **367**, 1026–1030 (2020).
[25]Y. Li and F. Mei, "Deep learning-based method coupled with small sample learning for solving partial differential equations," Multimedia Tools Appl. **80**, 17391–17413 (2021).
[26]Z. Zhou and Z. Yan, "Solving forward and inverse problems of the logarithmic nonlinear Schrödinger equation with PT-symmetric harmonic potential via deep learning," Phys. Lett. A **387**, 127010 (2021).
[27]S. Cai, Z. Wang, S. Wang, P. Perdikaris, and G. E. Karniadakis, "Physics-informed neural networks for heat transfer problems," J. Heat Transfer **143**, 1–15 (2021).
[28]S. Wang, X. Yu, and P. Perdikaris, "When and why PINNs fail to train: A neural tangent kernel perspective," J. Comput. Phys. **449**, 110768 (2022).
[29]A. D. Jagtap and G. E. Karniadakis, "Extended physics-informed neural networks (XPINNs): A generalized space-time domain decomposition based deep learning framework for nonlinear partial differential equations," Commun. Comput. Phys. **28**, 2002–2041 (2020).
[30]A. D. Jagtap, K. Kawaguchi, and G. E. Karniadakis, "Adaptive activation functions accelerate convergence in deep and physics-informed neural networks," J. Comput. Phys. **404**, 109136 (2020).
[31]Z. Li, X. Z. Mao, T. S. Li, and S. Zhang, "Estimation of river pollution source using the space-time radial basis collocation method," Adv. Water Resour. **88**, 68–79 (2016).
[32]W. P. Cheng and Y. Jia, "Identification of contaminant point source in surface waters based on backward location probability density function method," Adv. Water Resour. **33**, 397–410 (2010).
[33]M. Stein, "Large sample properties of simulations using Latin hypercube sampling," Technometrics **29**, 143–151 (1987).
[34]D. P. Kingma and J. L. Ba, "Adam: A method for stochastic optimization," in *3rd International Conference on Learning Representations, ICLR 2015-Conference Track Proceedings* (IEEE, 2015), pp. 1–15.
[35]J. H. Ten Thije Boonkkamp and M. J. Anthonissen, "The finite volume-complete flux scheme for advection-diffusion-reaction equations," J. Sci. Comput. **46**, 47–70 (2011).