mathematics

MDPI

Article

Deep Neural Networks with Spacetime RBF for Solving Forward and Inverse Problems in the Diffusion Process

Cheng-Yu Ku [1], Chih-Yu Liu [2,*], Yu-Jia Chiu [1,*] and Wei-Da Chen [1]

[1] Department of Harbor and River Engineering, National Taiwan Ocean University, Keelung 202301, Taiwan; chkst26@mail.ntou.edu.tw (C.-Y.K.); 11252011@email.ntou.edu.tw (W.-D.C.)
[2] Department of Civil Engineering, National Central University, Taoyuan 320317, Taiwan
* Correspondence: liu20452003@ncu.edu.tw (C.-Y.L.); yjchiu@mail.ntou.edu.tw (Y.-J.C.)

Abstract: This study introduces a deep neural network approach that utilizes radial basis functions (RBFs) to solve forward and inverse problems in the process of diffusion. The input layer incorporates multiquadric (MQ) RBFs, symbolizing the radial distance between the boundary points on the spacetime boundary and the source points positioned outside the spacetime boundary. The output layer is the initial and boundary data given by analytical solutions of the diffusion equation. Utilizing the concept of the spacetime coordinates, the approximations for forward and backward diffusion problems involve assigning initial data on the bottom or top spacetime boundaries, respectively. As the need for discretization of the governing equation is eliminated, our straightforward approach uses only the provided boundary data and MQ RBFs. To validate the proposed method, various diffusion scenarios, including forward, backward, and inverse problems with noise, are examined. Results indicate that the method can achieve high-precision numerical solutions for solving diffusion problems. Notably, only 1/4 of the initial and boundary conditions are known, yet the method still yields precise results.

Keywords: deep neural network; diffusion; multiquadric; radial basis function; spacetime

MSC: 35D35; 65M32

Citation: Ku, C.-Y.; Liu, C.-Y.; Chiu, Y.-J.; Chen, W.-D. Deep Neural Networks with Spacetime RBF for Solving Forward and Inverse Problems in the Diffusion Process. *Mathematics* **2024**, *12*, 1407. https://doi.org/10.3390/math12091407

Academic Editors: Thái Anh Nhan and Istvan Farago

Received: 4 April 2024
Revised: 28 April 2024
Accepted: 1 May 2024
Published: 4 May 2024

1. Introduction

Various applications, such as modeling heat conduction, chemical diffusion, and the spread of pollutants or other substances through a porous medium, can be described by the diffusion equation. Solving the diffusion equation provides insights into the distribution and evolution of the diffusing substance over time and space [1–3]. Different initial and boundary conditions can be applied to represent specific scenarios in practical applications. It mathematically models how a substance's concentration changes in space and time due to the process of diffusion [4,5]. The diffusion equation finds extensive applications across various scientific and engineering disciplines including groundwater, environmental science, geophysics, transport phenomena, heat conduction, and engineering [6,7]. Solving the diffusion equation involves finding the distribution of a quantity over space and time for given initial and boundary conditions. Numerical techniques like finite difference, finite element, spectral, or meshfree methods are commonly used [8,9]. Among these methodologies, the meshfree methods have attracted the attention of researchers from various scientific fields due to their ability to handle diffusion equations with complex and irregular geometries [6,8,10]. The radial basis function (RBF) collocation method is a meshfree approach used to analyze governing equations, where the unknowns are represented by function approximation. The interpolation method using RBF was proposed by Hardy in 1971 [11], who utilized the multiquadric (MQ) RBF for interpolating scattered data to address various problems. Apart from the MQ RBF, several other types of RBFs exist, including the inverse multiquadric (IMQ), Gaussian, and polyharmonic spline (PS)

functions. Among these RBFs, the MQ function has been shown to yield more precise solutions compared to other RBF interpolations, making it a popular choice for addressing two-dimensional forward diffusion problems [12–14].

In addition to the conventional numerical method, the artificial neural network (ANN) for solving diffusion equations leverages the power of ANNs to approximate solutions to diffusion problems [15–17]. This approach is particularly useful when solutions are challenging to obtain due to complex geometries, variable coefficients, or intricate boundary conditions [18]. The neural network architecture includes input, hidden, and output layers. The input layer receives information about spatial coordinates, time, and any other relevant parameters. The hidden layers process the input data through a series of weighted connections, applying activation functions to produce complex transformations. The output layer provides the predicted solution, typically the concentration or quantity at a specific point in space and time [19,20]. The backpropagation neural network is a common type of ANN extensively employed for tasks involving supervised learning [21–23]. The effectiveness of ANNs for solving diffusion problems depends on factors like the choice of architecture, dataset quality, and the complexity of the diffusion problem. A deep neural network (DNN) Is a type of ANN characterized by its multiple layers of interconnected nodes or neurons. It is a key component of deep learning, a subset of machine learning that aims to model and interpret data through the use of complex, hierarchical structures [24,25]. In recent years, the RBF neural network algorithm has been utilized to address the inverse Cauchy problems associated with the Laplace equation. Additionally, the application of RBF neural networks has been suggested for estimating the initial conditions in heat conduction scenarios [26,27]. In recent work by Ku et al. [9], they introduced a collocation technique using spacetime RBFs for addressing inverse heat conduction problems. This method is founded on the spacetime domain concept, where initial and boundary data are prescribed on spacetime boundaries. By allocating the given initial data to either the bottom or top spacetime boundaries, depending on whether it is a forward or backward problem, the numerical solutions for diffusion problems in both directions can be accurately estimated.

This study presents a novel approach employing a DNN with RBFs to address both forward and inverse diffusion problems. In this method, the input layer incorporates collocation technique using MQ RBFs proposed by Ku et al. [9], representing the radial distance between boundary points on the spacetime boundary and external source points. The output layer receives initial and boundary data through analytical solutions of the diffusion equation. Using spacetime coordinates, the approximations for forward and backward diffusion problems involve assigning initial data on the bottom or top spacetime boundaries, respectively. This approach eliminates the need for discretization of the governing equation, offering a direct solution using the provided boundary data and MQ RBFs. To validate the proposed method, various diffusion scenarios, including forward, backward, and inverse problems with noise, are examined.

2. DNN Method with Spacetime RBF

2.1. Problem Statement

The diffusion problem, categorized as a partial differential equation (PDE), involves second-order derivatives and is commonly expressed as follows:

$$\Delta h(\mathbf{x},\ t) = \frac{1}{D}\frac{\partial h(\mathbf{x},\ t)}{\partial t},\ \mathbf{x} \in \Omega, \tag{1}$$

where Δ denotes the Laplacian operator, Ω denotes the domain, h denotes the unknown to be solved or the pressure head in groundwater applications, $\mathbf{x}$ denotes the spatial coordinate, t denotes the time, D denotes the hydraulic diffusivity, expressed as $D = K/S_S$, K denotes the hydraulic conductivity, and Ss denotes the specific storage.

2.1.1. Forward Diffusion Problem

The forward diffusion problem describes the temporal evolution of a physical quantity, such as heat, concentration, or particle diffusion. It is a well-posed problem and frequently employed to forecast how a quantity alters over time, contingent upon its initial and boundary conditions. Generally, it necessitates the specification of initial conditions at the initial time and boundary conditions across the spatial domain, expressed as:

$$h(\mathbf{x},\ t=0) = h_0(\mathbf{x}),\ \mathbf{x} \in \Omega, \tag{2}$$

$$h(\mathbf{x},\ t) = h_b(\mathbf{x},\ t),\ \mathbf{x} \in \partial\Omega. \tag{3}$$

In the preceding equations, $h_0(\mathbf{x})$ denotes the initial value of the pressure head distribution, $h_b(\mathbf{x},\ t)$ denotes the boundary value of the pressure head distribution, and $\partial\Omega$ denotes the boundary of Ω. The forward diffusion problem is described by the governing equation presented in Equation (1), which is solved using the provided initial data in Equation (2) and the specified boundary data in Equation (3), as visualized in Figure 1. Through these provided initial and boundary conditions, it models the diffusion of the quantity from areas with higher concentration to areas with lower concentration.

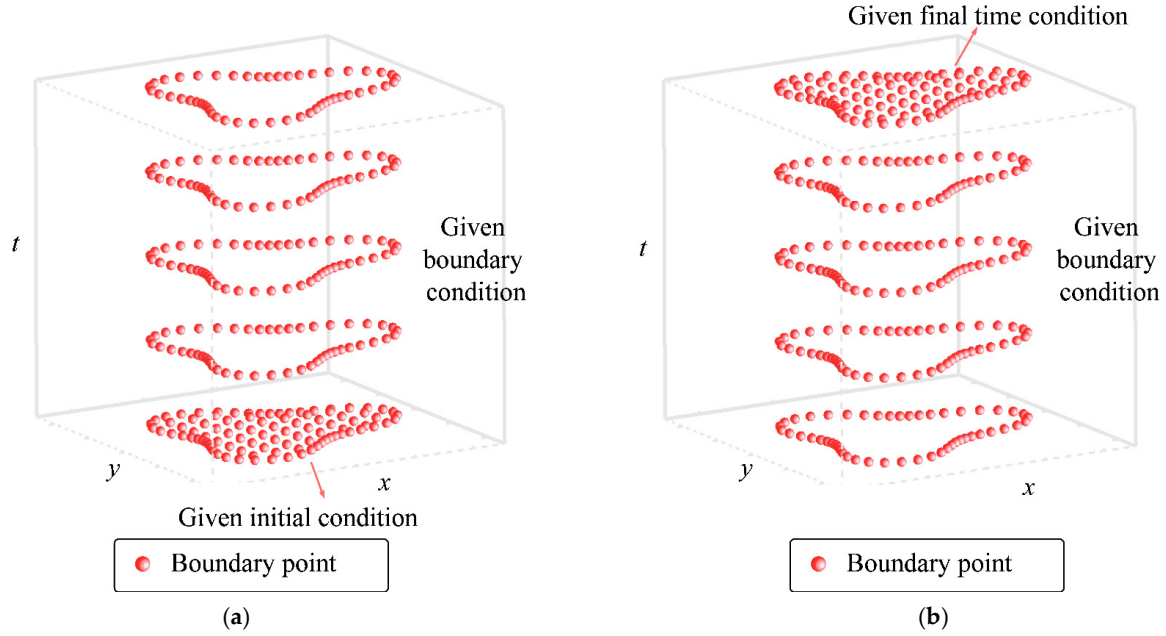


Figure 1. The two-dimensional forward and backward diffusion problems in the spacetime domain: (**a**) Forward diffusion problem and (**b**) backward diffusion problem.

2.1.2. Backward Diffusion Problem

The backward diffusion problem is associated with the problem of retroactively determining the past states of a physical quantity based on its current state. It essentially models the reverse diffusion, aiming to retrieve the prior distribution of the quantity from a specified final state. Backward diffusion problems are often ill-posed, lacking a unique solution without supplementary constraints. This is due to the inherent difficulty in recovering historical data that might be incomplete or subject to uncertainty.

Equation (1) also serves as the governing equation for the backward diffusion problem. It necessitates defining final time conditions at the desired endpoint and specifying boundary conditions, as detailed in Equation (3). The expression for final time conditions is as follows:

$$h(\mathbf{x},\ t=T) = h_T(\mathbf{x},\ t=T), \tag{4}$$

where $h_T(\mathbf{x},\ t = T)$ is the unknown to be solved, specifically the pressure head in groundwater applications at the final time. Usually, given the ill-posed nature of the backward diffusion problem, even slight random noise in the measurement data can result in a potentially substantial error in the solution [28].

In practical applications, data often encounter noise interference, indicating that the obtained boundary conditions or observed data may not be entirely accurate. Considering noise and developing methods with noise resilience is crucial when dealing with the backward diffusion problem. Hence, to investigate the noise resistance capability of this study, when considering the backward diffusion problem, both boundary and final time data are subjected to noise interference [29,30], as expressed below.

$$\widetilde{h}_b(\mathbf{x},\ t) = \left[\frac{\delta(2 \times rand - 1)}{100} + 1\right] h_b(\mathbf{x},\ t), \tag{5}$$

$$\widetilde{h}_T(\mathbf{x},\ t) = \left[\frac{\delta(2 \times rand - 1)}{100} + 1\right] h_T(\mathbf{x},\ t), \tag{6}$$

where $\widetilde{h}_b(\mathbf{x},\ t)$ is the boundary data with noise, $h_b(\mathbf{x},\ t)$ is the boundary data that have not been contaminated by noise, representing the true boundary data, $\widetilde{h}_T(\mathbf{x},\ t)$ is the noisy final time data, $h_T(\mathbf{x},\ t)$ is the true final time data, δ is the level of noise data, and *rand* signifies a set of randomly sampled numbers exhibiting no discernible pattern or sequence. These numbers are produced using a random number generator primarily intended to enhance the system's unpredictability and randomness.

2.2. Data Preparation

Data preparation is a pivotal phase in the DNN method [31,32]. To solve diffusion problems, it involves the gathering and arrangement of input and output pairs essential for training the neural network. In this research, we have introduced a simplified MQ RBF [33]. The RBF employed in this study does not require the determination of a shape parameter, nor does it involve the setup of center points. Instead, the center points in this study are external source points positioned outside the spacetime computational domain. The RBF used in this study is represented by the following formula:

$$\phi(r_i) = r_i, \tag{7}$$

where $\phi(r_i)$ represents the RBF, r_i represents the spacetime distance described as $r_i = \left|\mathbf{x} - \mathbf{x}_i^s\right|$, and $\mathbf{x}_i^s$ represents the ith source point, defined as $\mathbf{x}_i^s = \left(x_i^s, y_i^s, t_i^s\right)$. The above RBF serves as the training data for the DNN.

The output data involve employing either analytical solutions or simulation data to determine the corresponding boundary data values. Analytical solutions are used to calculate the corresponding solution values for each chosen collocation point. These pairs, which consist of the RBF values and corresponding solution values, constitute the training data for the DNN.

Our approach only requires the placement of boundary and source points. Notably, the introduced simplified RBF in our study incorporates the concept of spacetime, treating time as a virtual spatial dimension. Therefore, the original two-dimensional spatial coordinates, when extended with a one-dimensional time dimension, can be transformed into three-dimensional spacetime coordinates. The collocation scheme is visualized in Figure 2. Figure 2 depicts the source points strategically positioned outside the computational domain enclosed by the boundary points. These source points are skillfully arranged in a spiral pattern within the three-dimensional spacetime coordinates. This particular source point configuration has demonstrated enhanced accuracy and increased computational precision in addressing transient issues [9]. An important advantage of the proposed DNN with the spacetime RBF approach is that it obviates the necessity of discretizing the

diffusion problem. As a result, this method offers a straightforward way to solve diffusion problems, relying solely on provided RBF, initial data, and boundary conditions.

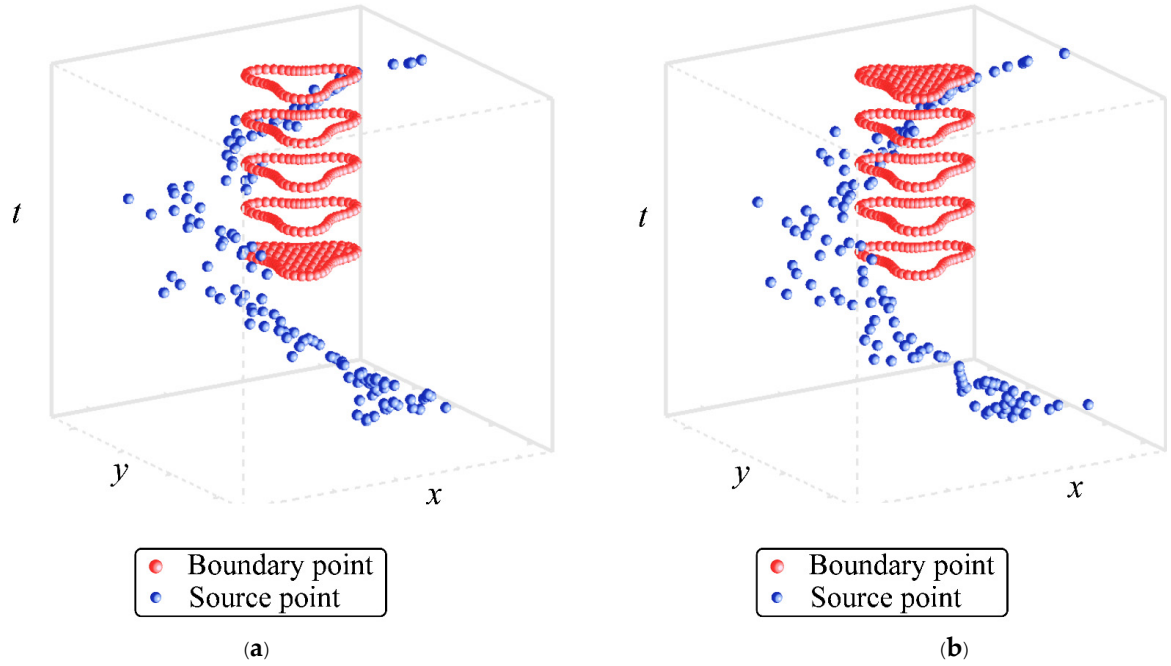


Figure 2. Collocation points for the proposed DNN with spacetime RBF: (**a**) Forward diffusion problem and (**b**) backward diffusion problem.

2.3. DNN Architecture Development

The creation of the DNN architecture initiates with the weights composition for tackling the forward and backward diffusion problems. The linear combination of bias term and the inputs represent the weights [34,35].

The estimation for each neuron within a hidden layer of a neural network is a numerical value that represents the output or activation of that neuron. This estimation is determined by computing the weighted sum of the inputs to the neuron and, subsequently, it undergoes an activation function to generate the ultimate output, as defined by:

$$W_j = s_j + \sum_{i=1}^{m} [\alpha_{ij}\phi(r_i)], \tag{8}$$

where W_j represents an intermediate value, which is essentially the weighted sum, used to calculate the output of neuron j. This value is further processed by applying an activation function to it, where s_j represents the bias, m represents the total input elements originating from the input layer, and α_{ij} is the weight between neuron i and j in the hidden layer.

As depicted in Equation (8), the estimation for each neuron in a hidden layer of a neural network is obtained through the computation of a dot product between its input values and the corresponding weights, followed by the application of an activation function to the result. Subsequently, this estimation is propagated to the neurons in the subsequent layer. The precise value for each neuron's estimation is contingent upon factors such as the network's architecture, the assigned weights for connections, and the activation function employed, expressed by the following formula:

$$W_{ij} = s_{ij} + \sum_{j=1}^{n}\sum_{i=1}^{m} v_{ij} f(W_j), \tag{9}$$

In the preceding equations, W_{ij} represents the total weighted sum of neuron i in the jth hidden layer, s_{ij} represents the bias, v_{ij} represents the weight connected to the activation function, n represents the neuron number, and $f(W_j)$ represents the activation function, specifically a hyperbolic tangent function, defined as $f(W_j) = (e^{W_j} - e^{-W_j})/(e^{W_j} + e^{-W_j})$.

By following the steps outlined in the proposed DNN with spacetime RBF architecture, numerical approximations can be obtained, as expressed below:

$$h_{net} = \sum_{j=1}^{n} \beta_j f(W_{ij}), \tag{10}$$

where h_{net} represents the approximations, and β_j represents the weights related to the output layer. The DNN architecture is illustrated in Figure 3, with the links between nodes symbolizing the weighted values responsible for managing signal transmission. The neural network emulates signal processing by incorporating weights for each input before entering a neural unit. Every node in the network is linked to an activation function, shaping the network's output [36]. In this study, we utilize backpropagation neural networks within the DNN framework. During the computation, we enhance the training of the DNN model with the inclusion of the provided data and the spacetime distances between the boundary and source points for constructing the simplified RBF. This process contributes to achieving higher accuracy in the model's predictions.

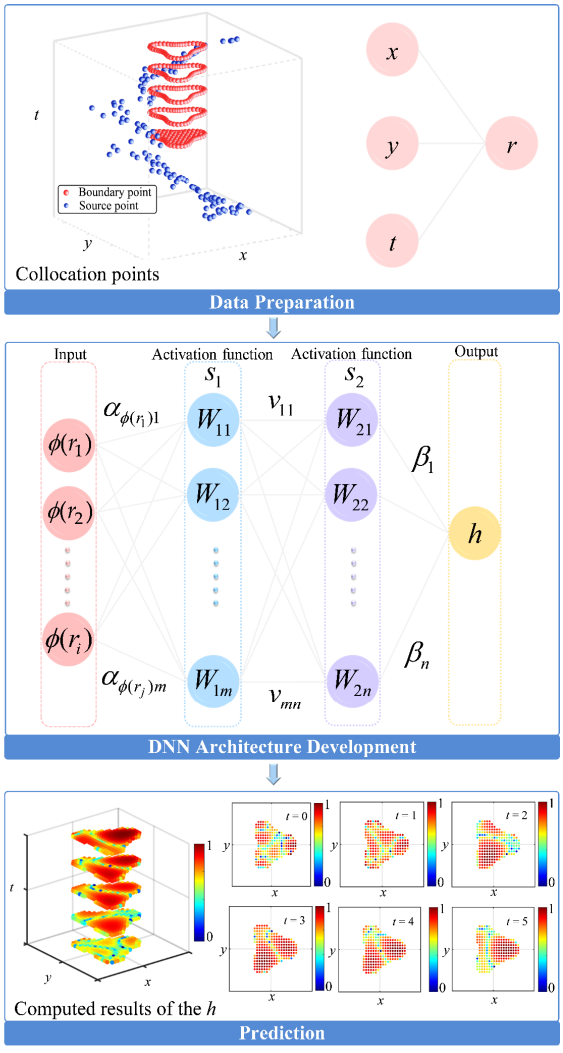


Figure 3. Structure of the proposed DNN with spacetime RBF method.

2.4. Loss Function Assessment

The main objective in DNN computations is to achieve optimal training by minimizing the objective function through adjustments to the network's parameters, which encompass weights and biases. The loss function employed in this study is then used to quantify the dissimilarity between the predicted output values and the actual target values as:

$$Loss = \frac{\sum_{i=1}^{M} (h_{net,i} - h_i)^2}{M}, \tag{11}$$

where *Loss* represents the loss function, M represents the data point number, h_i represents the target solutions. The Levenberg–Marquardt (LM) algorithm serves as the optimizer tasked with minimizing Equation (11), which quantifies the discrepancy between the predicted and actual solutions at individual data points. Through iterative weight and bias updates, the algorithm strives to identify the parameter combination that yields the minimal loss function value.

2.5. Training Phase

The optimization algorithm aims to minimize the loss function, as shown in Equation (11), by modifying the network's parameters. This iterative process persists until the network's predictions closely match the intended target outputs. During each iteration, the algorithm revises the biases and weights to minimize the error. The formula for these updates is as follows:

$$\Delta w = -(J + \lambda^k H)^{-1} \nabla Loss^k, \tag{12}$$

In the preceding equation, w represents the weights, J represents the Jacobian matrix, H represents the Hessian matrix, λ represents the damping parameter, ∇ represents the gradient, and k represents the iteration number. In this study, through Equation (12), and in combination with the LM formula, the computational process iteratively adjusts the weights and biases to reevaluate the computed loss. Throughout this iterative process, the computation continues until it reaches the specified number of iterations or achieves an appropriate minimum change in error, satisfying the convergence conditions set above, at which point, the computation terminates.

2.6. Prediction

During the prediction phase, the network utilizes the connections between the input and output variables to make predictions or execute the intended tasks. Remarkably, it accomplishes these tasks without the need for any further adjustments to its weights or biases. To evaluate the accuracy of the method, this study employs three types of error assessment metrics in the subsequent analysis cases, including root mean square error (RMSE), maximum relative error (MRE), and maximum absolute error (MAE), as evaluation criteria.

This study utilizes DNNs to train neural network models for the direct approximation of solutions to the diffusion problem. As a result, through the following analysis, this study aims to demonstrate that the proposed method is particularly suitable for complex domains characterized by irregular and intricate geometric shapes.

3. Validation Example

In the first numerical case of this study, we initially investigate the two-dimensional forward diffusion problem, as depicted in Equation (1). The computational domain in this study exhibits an irregular amoeba-like shape as follows:

$$\begin{aligned} &\partial\Omega = \{(x, y, t) | x = \rho(\theta)\cos\theta,\ y = \rho(\theta)\sin\theta\},\ 0 \le \theta \le 2\pi,\ 5 \le t \le 10, \\ &\rho(\theta) = \left[e^{\cos(\theta)}\cos^2(2\theta) + \sin^2(2\theta)\right] e^{\sin(\theta)}. \end{aligned} \tag{13}$$

As this study pertains to a forward problem, the initial conditions and boundary conditions must be provided to solve Equation (1), expressed as:

$$u(x, y, t = 5) = x^2 - y^2 + e^{-10/D} \sin(x) \sin(y), \tag{14}$$

$$u(x, y, t) = x^2 - y^2 + e^{-2t/D} \sin(x) \sin(y). \tag{15}$$

In this study, with a hydraulic diffusivity of 1 and a total computation time of 5, two numerical methods are employed for analysis. For the DNN with spacetime RBF, only two categories of collocation points need to be positioned, the source and boundary points, as depicted in Figure 4.

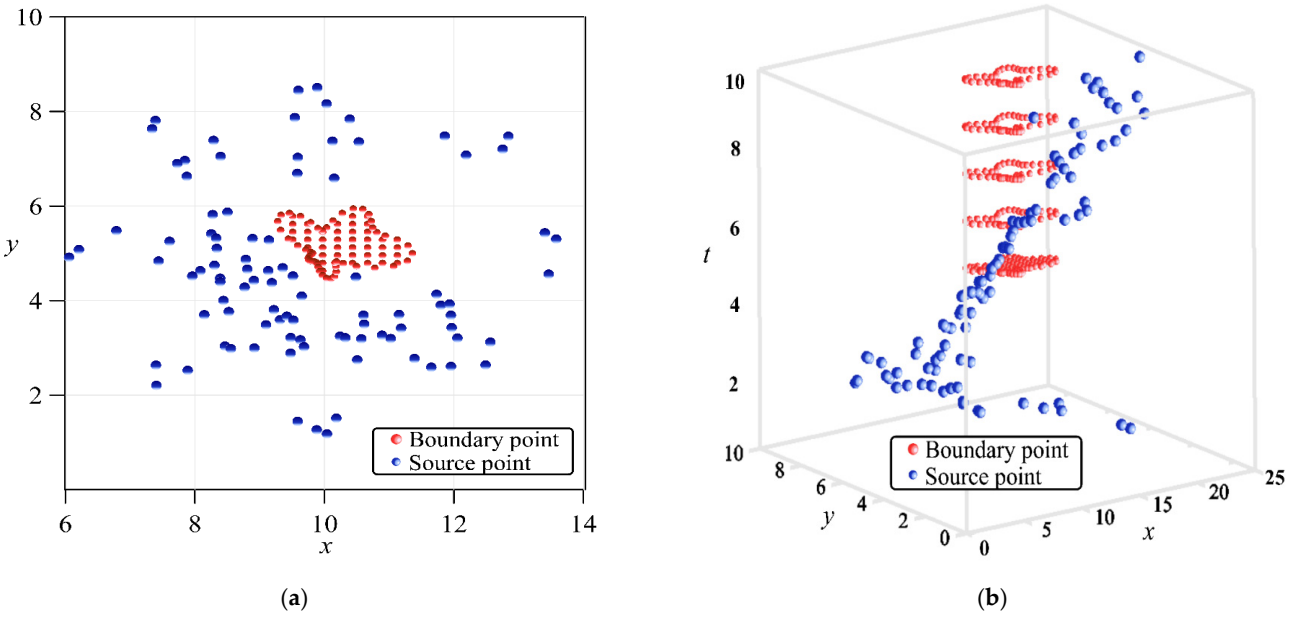


Figure 4. Placement of collocation point for the two-dimensional diffusion problem: (**a**) x–y plane projection of the spacetime domain and (**b**) the three-dimensional spacetime domain.

In this example, a total of 110 source points and 1001 boundary points are used. The arrangement of source points is presented in the following formula.

$$\begin{aligned} &\partial\Omega^s = \left\{ (x_j^s, y_j^s, t) \middle| x_j^s = \eta\rho_j^s(\theta_j^s)\cos(\theta_j^s),\ y_j^s = \eta\rho_j^s(\theta_j^s)\sin(\theta_j^s) \right\},\ 0 \le \theta_j^s \le 2\pi,\ 0 \le t \le 10, \\ &\rho_j^s(\theta_j^s) = 2[1 + \cos^2(4\theta)]. \end{aligned} \tag{16}$$

where θ_j^s and ρ_j^s represents the angle and radius of source point, η represents the dilation factor used to describe the distance between source and boundary points.

In this example, the DNN with spacetime RBF developed in this study and the Kansa method, for the purpose of comparing numerical results, are both adopted for solving this example. As for the Kansa method, it requires the placement of three categories of points, including the boundary, inner, and center points. The quantities of the boundary, inner, and center points are 81, 191, and 272, respectively. Both center and inner points are situated within the domain, and their locations are entirely identical.

3.1. The Impact of the Dilation Parameter on Accuracy

Based on the information presented in Figure 4, which illustrates the positioning of collocation points for the two-dimensional diffusion problem, it becomes evident that when source points are placed outside the computational domain, an adjustment of the distance between the source points and the boundary line is required through the dilation parameter.

The source points are located at a certain distance outside the boundary points (i.e., the dilation parameter).

Given that the locations of the source points may impact result accuracy, we conduct a convergence analysis to assess MAE accuracy relative to the dilation parameter value. This analysis involves placing sources at various positions outside the three-dimensional spacetime region. With the number of hidden layers set at 9, we have 81 boundary points, 191 inner points, and 272 center points. Figure 5 shows the results of the dilation parameter versus the MAE. The convergence analysis outcomes show that approximations in the range of 10^{-3} to 10^{-4} are attainable when the dilation parameter is between 1 and 50. Hence, a dilation parameter of 5 is consistently applied in subsequent analyses.

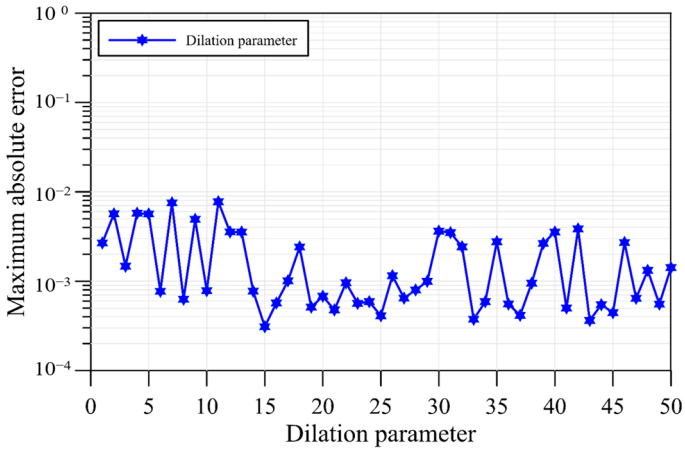
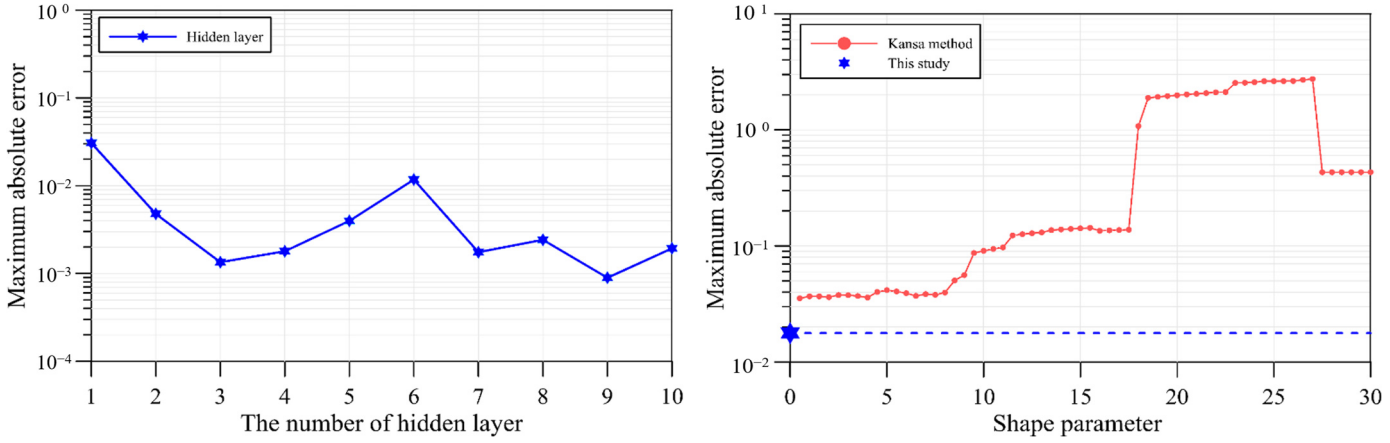


Figure 5. The dilation parameter versus the MAE.

3.2. The Impact of Number of the Hidden Layer on Accuracy

Recognizing the pivotal role of hidden layers in a DNN for the extraction and comprehension of intricate patterns and data representations, which were subsequently applied in making predictions or classifications in the output layer, this study conducts a sensitivity analysis of the parameters to assess the impact of varying the number of hidden layers. The dataset is divided into three parts within the DNN, which include the training, testing, and validation sets. Specifically, 70% of the dataset is designated for the training set, while both the testing and validation sets each account for 15% of the data.

The LM optimization technique was utilized. During the computational process, the DNN with spacetime RBF reached a mean squared error of 10^{-7} after the 1000th epoch. Additionally, the correlation coefficients for the training, validation, and testing phase were all close to 1, indicating a high degree of correlation between the actual physical quantities and the predictions. Subsequently, this study explores the impact and sensitivity of the number of hidden layers on the accuracy of the computational results. As indicated by the results in Figure 6a, our analysis reveals that the number of hidden layers, within the range of 3 to 10, yields precision levels ranging from approximately 10^{-2} to 10^{-4}. Consequently, based on these findings from the convergence analysis, we adopt a consistent configuration of 9 hidden layers for all subsequent cases in this study.



(**a**) (**b**)

Figure 6. The number of hidden layer and shape parameter versus the MAE: (**a**) Number of hidden layers versus the MAE; (**b**) Shape parameter versus the MAE.

3.3. Comparison of the Accuracy

Since the proposed DNN with the spacetime RBF belongs to the ANN, the weights and biases of the neural network are randomly initialized during the computation process. Additionally, different random mini batches of data are used in each iteration, introducing randomness into the training process. Therefore, in this example, the training was conducted 50 times, and the results were averaged to reduce the variance and increase reliability. The average computational error of the proposed DNN with the spacetime RBF was computed across these 50 cases.

Furthermore, within conventional method such as the Kansa method [37], the shape parameter significantly influences computational accuracy. Another key advantage of the approach used in this study is the elimination of the need for the determination of the shape parameter. As indicated by the analysis in Figure 6b, the Kansa method exhibits sensitivity to the shape parameter. When the shape parameter falls within the range of 0.5 to 18, it achieves computational accuracy in the order of 10^{-1} to 10^{-2}. However, beyond a shape parameter value of 18, it fails to provide accurate numerical solutions. In contrast, the approach employed in this study offers substantial advantages as it does not necessitate the determination of the shape parameter and still delivers high-precision solutions.

In this study, RMSE, MAE, and MRE were utilized as the three-error metrics, and their results are summarized in Table 1. The results in Table 1 highlight the notable accuracy achieved by the proposed DNN using spacetime RBF with exterior source points. The RMSE, MAE, and MRE values obtained are in the order of 10^{-3}, 10^{-5}, and 10^{-3}, respectively. In contrast, the Kansa method, even with optimized shape parameters, can only achieve an accuracy of around 10^{-2}. This illustrates that the proposed method can achieve greater accuracy when compared to conventional approaches, all without the determination of a shape parameter and the discretization of the governing equation.

Table 1. Comparison of errors for the first validation case.

	RMSE	**MAE**	**MRE**
Kansa method (c = 4)	1.01×10^{-1}	5.84×10^{-2}	3.68×10^{-2}
This study (across 50 cases)	3.24×10^{-3}	2.11×10^{-3}	2.22×10^{-5}

4. Applications

4.1. Two-Dimensional Backward Diffusion Problem

The backward diffusion problem describes the behavior of a diffusive process in reverse, often used for problems involving the reconstruction of past events or conditions based on present or final data. It is the time-reversed version of the standard diffusion problem. In the context of time-dependent diffusion problems, the forward diffusion problem models how a quantity (such as temperature, concentration, or pressure) evolves over time due to diffusion. The backward diffusion problem, on the other hand, seeks to determine the history of the quantity by using present or final data. Typically, the backward diffusion problem is expressed as a PDE, as shown in Equation (1), involving spatial and temporal variables. In this case, the computational domain utilizes a boundary with a peanut-like shape, and the equation describing its boundary is represented as follows:

$$\begin{array}{l}\partial\Omega = \{(x,y,t)|x = \rho(\theta)\cos\theta,\ y = \rho(\theta)\sin\theta\},\ 0 \le \theta \le 2\pi,\ 5 \le t \le 10,\\ \rho(\theta) = 0.5e^{\cos(\theta)}\{2 + [0.5\sin(8\theta)]\}.\end{array} \quad (17)$$

To address the backward diffusion problem, one must define the final time conditions at the desired endpoint and boundary conditions, as outlined in the following equations:

$$u(x,y,t) = \frac{1}{4\pi t}e^{-\frac{D(x^2+y^2)}{4t}}, \quad (18)$$

$$u(x,y,t=5) = \frac{1}{20\pi}e^{-\frac{D(x^2+y^2)}{20}}. \quad (19)$$

The hydraulic diffusivity is 0.01. When dealing with real-world data, it is a common scenario to encounter boundary conditions that are measured or observed with noise or inaccuracies. These noisy boundary conditions can be incorporated into the forward and backward diffusion problem to account for uncertainties in the input data. Hence, the boundary conditions in this example are considered with noise level of 0.1, encompassing scenarios with and without noise. The collocation point distribution for two-dimensional forward and backward diffusion problems is depicted in Figure 7. When solving the two-dimensional forward diffusion problem, it typically necessitates the specification of the initial conditions at the starting time and the boundary conditions throughout the spatial domain. As shown in Figure 7, the original two-dimensional problem, when transformed into the spacetime coordinate system, becomes a three-dimensional spacetime coordinate problem.

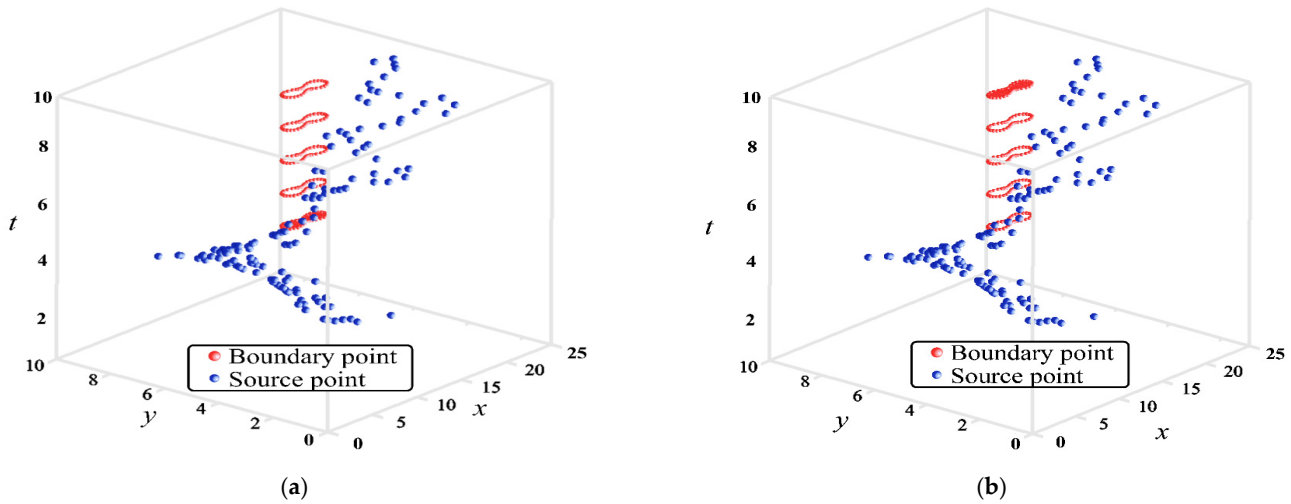


Figure 7. Collocation points for the two-dimensional forward and backward diffusion problem: (**a**) Forward diffusion problem and (**b**) backward diffusion problem.

The boundary points are positioned not only on the borders of the three-dimensional spacetime coordinate system but also at the bottom spacetime boundary of the three-dimensional spacetime computational domain, which represents the initial conditions. The source points are distributed irregularly outside the three-dimensional spacetime computational domain.

Similarly, when solving the two-dimensional backward diffusion problem, it requires specifying final time conditions at the end time and boundary conditions throughout the spatial domain. The boundary points are situated on the borders of the three-dimensional spacetime coordinate system and at the top spacetime boundary of the three-dimensional spacetime computational domain, representing the final time conditions. The source points are positioned outside the three-dimensional spacetime computational domain and exhibit an irregular, spiral-like arrangement. In this case, the point distribution for the two-dimensional forward and backward diffusion problems in the DNN-based RBF framework involves approximately 119 to 137 source and 1070 boundary points.

The DNN-based RBF architecture utilized in this study involves data partitioning into training, testing, and validation subsets. Specifically, the training subset accounts for 70% of the dataset, while the testing and validation subsets each use the remaining 15% of the data for analysis. For the comparison, a comprehensive assessment of accuracy is conducted, utilizing three different error metrics—RMSE, MAE, and MRE. The results of this analysis are presented in Table 2. As shown in Table 2, the analysis of the proposed DNN-based RBF method demonstrates its capability to provide high-precision numerical solutions, even when considering uncertainties. The obtained RMSE, MAE, and MRE values are in the order of 10^{-2}, 10^{-3}, and 10^{-1}, illustrating the method's accuracy. The findings indicate that the proposed DNN with spacetime RBF effectively tackles the two-dimensional backward diffusion problem with remarkable accuracy. Even when considering the interference of noisy boundary conditions, this research method continues to deliver precise results, illustrating its robustness against noise.

Table 2. Comparative results of the first application example for 50 runs.

Noise Level		δ **= 0**			δ **= 0.1**	
Error	RMSE	MAE	MRE	RMSE	MAE	MRE
Forward problem	1.35×10^{-2}	2.14×10^{-3}	1.67×10^{-1}	1.21×10^{-2}	2.79×10^{-3}	2.78×10^{-1}
Backward problem	2.05×10^{-2}	2.39×10^{-3}	1.85×10^{-1}	2.93×10^{-2}	3.68×10^{-3}	2.94×10^{-1}

4.2. Two-Dimensional Inverse Diffusion Problem

The inverse diffusion problem pertains to the challenge of determining the unknown parameters or initial conditions of a diffusion process based on observed data. Specifically, it involves the identification of diffusion coefficients, boundary conditions, or initial conditions that best describe the observed diffusion behavior. Solving the inverse diffusion problem often entails the utilization of observed data to estimate the unknown conditions. A gear-like shape is adopted, with the shape described by the following equation:

$$\begin{aligned} &\partial\Omega = \{(x,y,t)|x = \rho(\theta)\cos\theta,\ y = \rho(\theta)\sin\theta\},\ 0 \le \theta \le 2\pi,\ 5 \le t \le 10, \\ &\rho(\theta) = 0.5\{1 + 0.1\tanh[10 \times \sin(12\theta)]\}. \end{aligned} \tag{20}$$

In this study, a hydraulic diffusivity of 0.1 is employed. The analytical solution presented in the following equation serves as the spacetime boundary condition for this case:

$$u(x,y,t) = x^2 - y^2 + e^{-2t/D}\sin(x)\sin(y) \tag{21}$$

We explore five distinct scenarios, denoted as Case A, Case B, Case C, Case D, and Case E. These scenarios are schematically illustrated in Figure 8. In the first scenario, Case A, we focus on the forward diffusion problem, where both initial conditions and

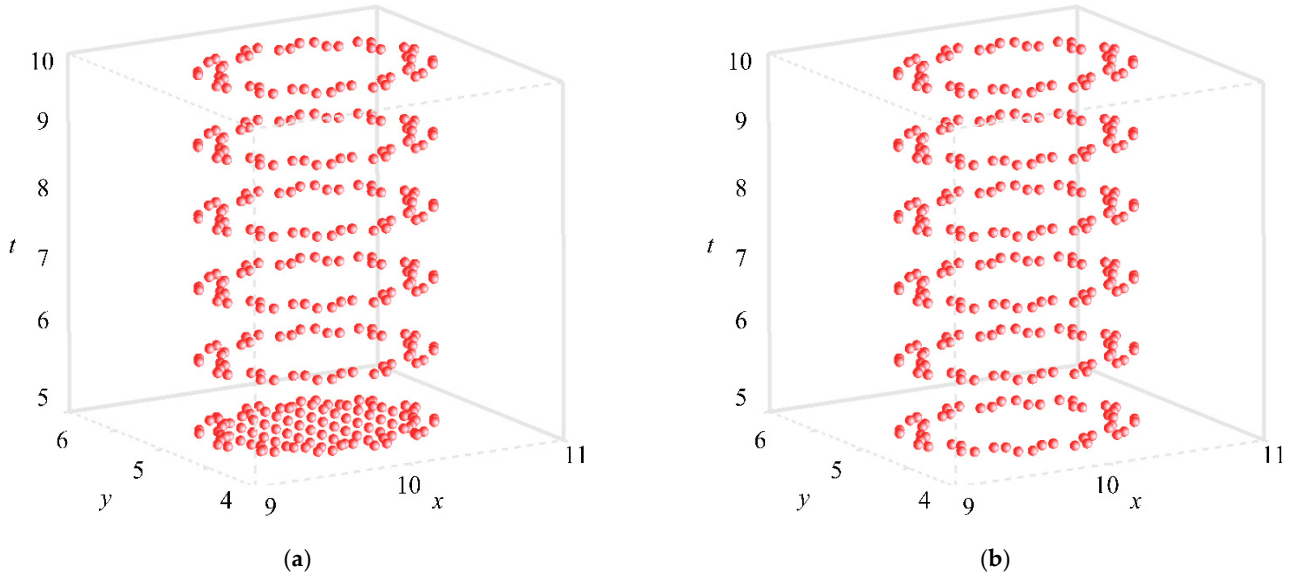
complete boundary conditions are provided. In the second scenario, Case B, the initial condition is unspecified, while all boundary conditions are known. The third scenario, Case C, involves only partial initial conditions and partial boundary conditions, with the given conditions accounting for half of the overall spacetime boundary conditions. Moving on to the fourth scenario, Case D, it shares similarities with Case C, as it also considers partial initial conditions and partial boundary conditions. However, the specific regions where the conditions are provided differ from those in Case C. Lastly, in the fifth scenario, Case E, only 1/4 of the initial conditions and 1/4 of the boundary conditions are known. It is clear that Case E represents the most challenging scenario with the fewest known conditions.

The summary of the number of collocation points for the five scenarios mentioned above is presented in Table 3. The analysis results of the average RMSE, MAE, and MRE obtained after conducting 50 runs are outlined in Table 4. The results indicate that the RMSE for the five scenarios are as follows: 10^{-3}, 10^{-2}, 10^{-2}, 10^{-3}, and 10^{-1}. The MRE at the final time (Figure 9) and within the spacetime domain (Figure 10) is visually represented. The MRE values for these scenarios are 10^{-5}, 10^{-5}, 10^{-4}, 10^{-5}, and 10^{-3}.

Furthermore, to illustrate the capability of the proposed DNN with the spacetime RBF in solving the inverse diffusion problem, while considering the influence of noise, a noise level of 0.1 is taken into account. The analysis results show that, even with the presence of noise in the provided conditions, the RMSE values for the five scenarios are as follows: 10^{-3}, 10^{-3}, 10^{-2}, 10^{-3}, and 10^{-1}. It is noteworthy that, even in the scenario where only 1/4 of the initial conditions and 1/4 of the boundary conditions are known (the fifth scenario), the proposed method can still yield high-precision results, as shown in Figure 11.

Table 3. The quantity of collocation points employed in the five scenarios.

	Number of Boundary Points	Number of Source Points
Case A	1038	100 to 135
Case B	810	100 to 135
Case C	534	100 to 135
Case D	544	100 to 135
Case E	277	100 to 135



(**a**) (**b**)

Figure 8. *Cont.*

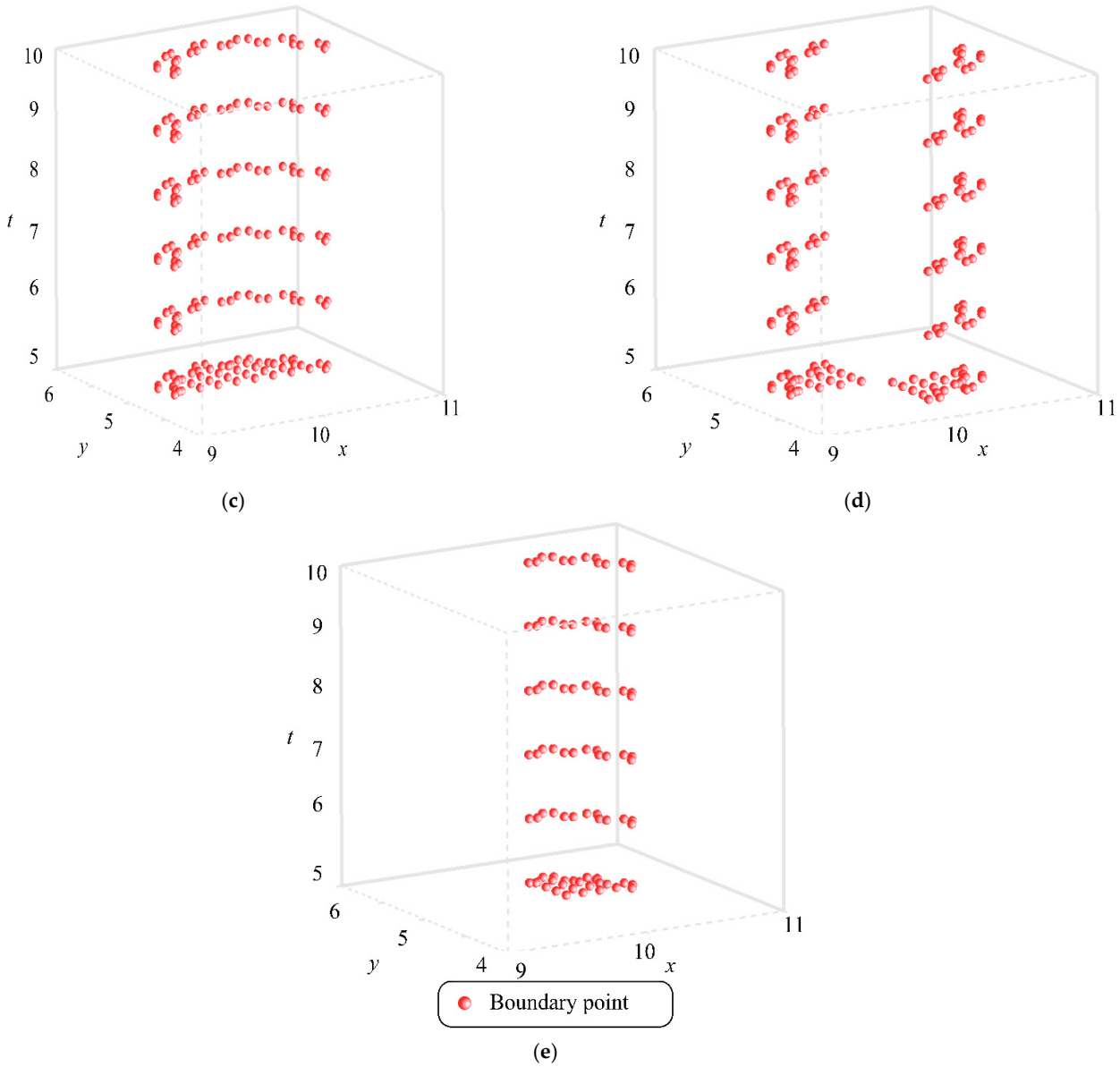


Figure 8. Boundary points used in the two-dimensional inverse diffusion problem: (**a**) Case A; (**b**) Case B; (**c**) Case C; (**d**) Case D; and (**e**) Case E.

Table 4. Computed error of the inverse diffusion problem for 50 runs.

Noise Level		δ **= 0**			δ **= 0.1**	
Error	RMSE	MAE	MRE	RMSE	MAE	MRE
Case A	4.93×10^{-3}	8.83×10^{-4}	1.17×10^{-5}	7.16×10^{-3}	2.17×10^{-3}	2.9×10^{-5}
Case B	1.05×10^{-2}	1.64×10^{-3}	2.18×10^{-5}	7.41×10^{-3}	2.37×10^{-3}	3.18×10^{-5}
Case C	9.50×10^{-2}	3.81×10^{-2}	4.84×10^{-4}	7.34×10^{-2}	3.49×10^{-2}	4.44×10^{-4}
Case D	9.55×10^{-3}	2.95×10^{-3}	4.12×10^{-5}	9.99×10^{-3}	3.62×10^{-3}	4.99×10^{-5}
Case E	7.65×10^{-1}	2.30×10^{-1}	3.18×10^{-3}	7.13×10^{-1}	1.86×10^{-1}	2.58×10^{-3}

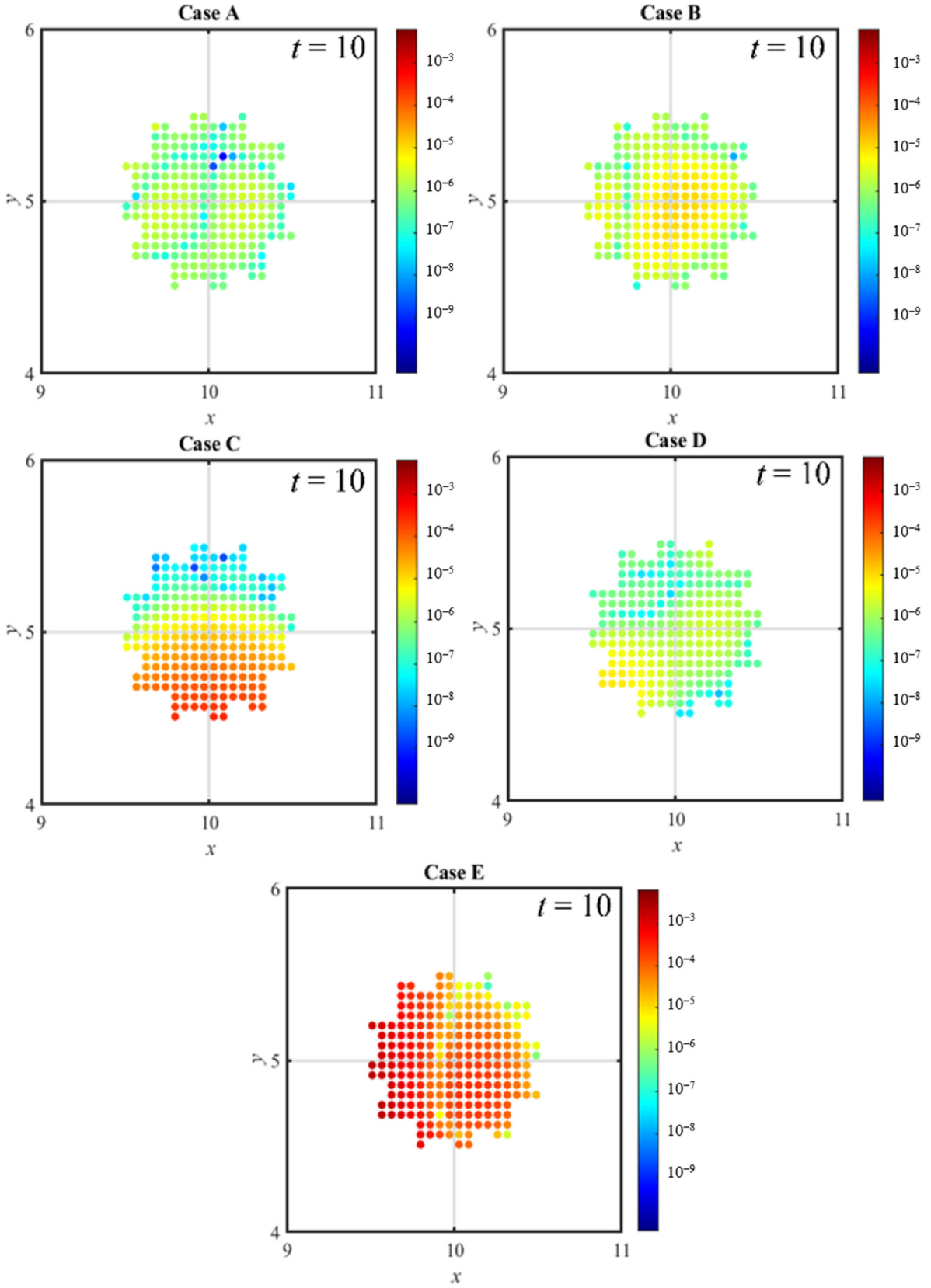


Figure 9. The maximum relative error at the final time (excluding noise).

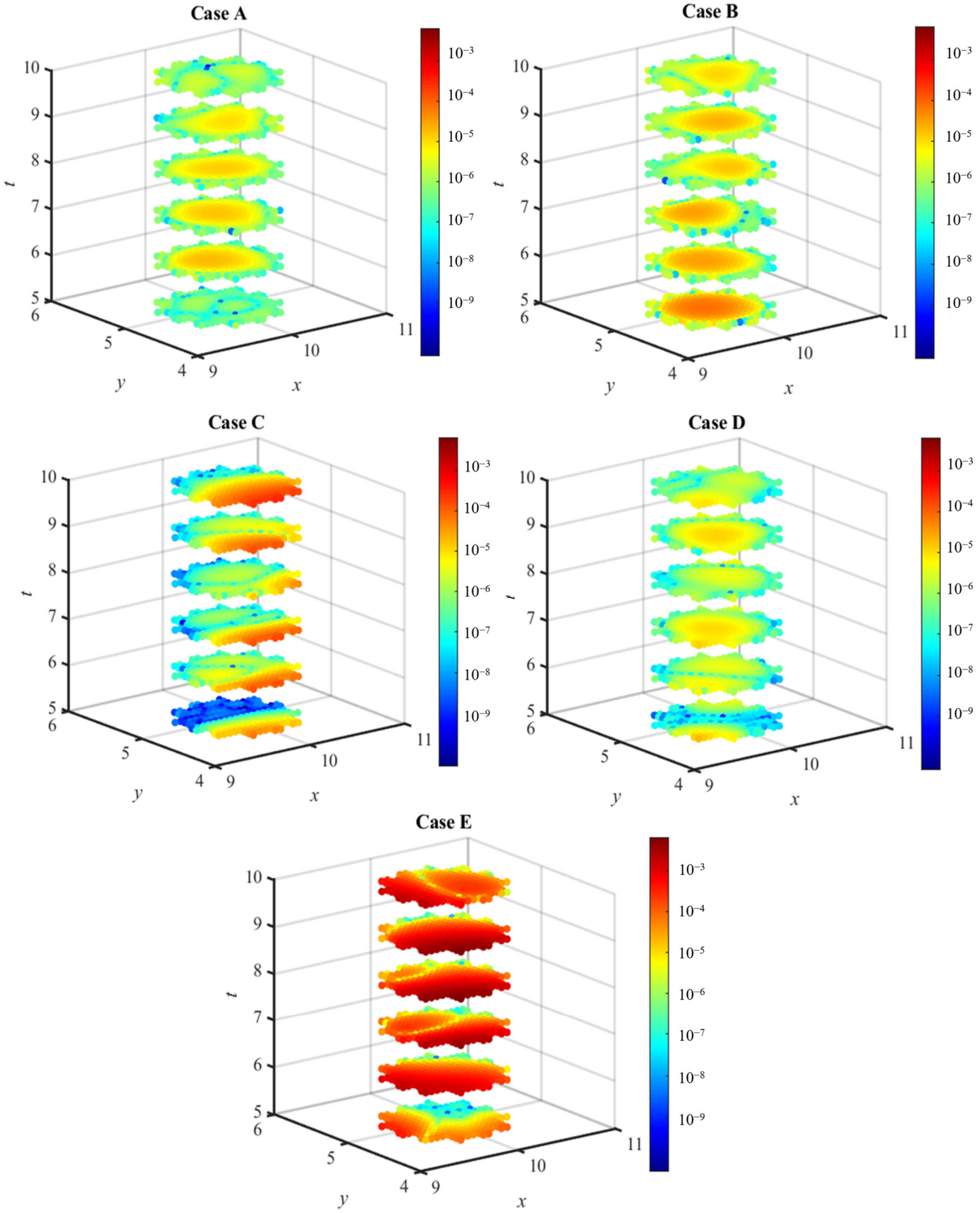


Figure 10. The maximum relative error within the spacetime domain (excluding noise).

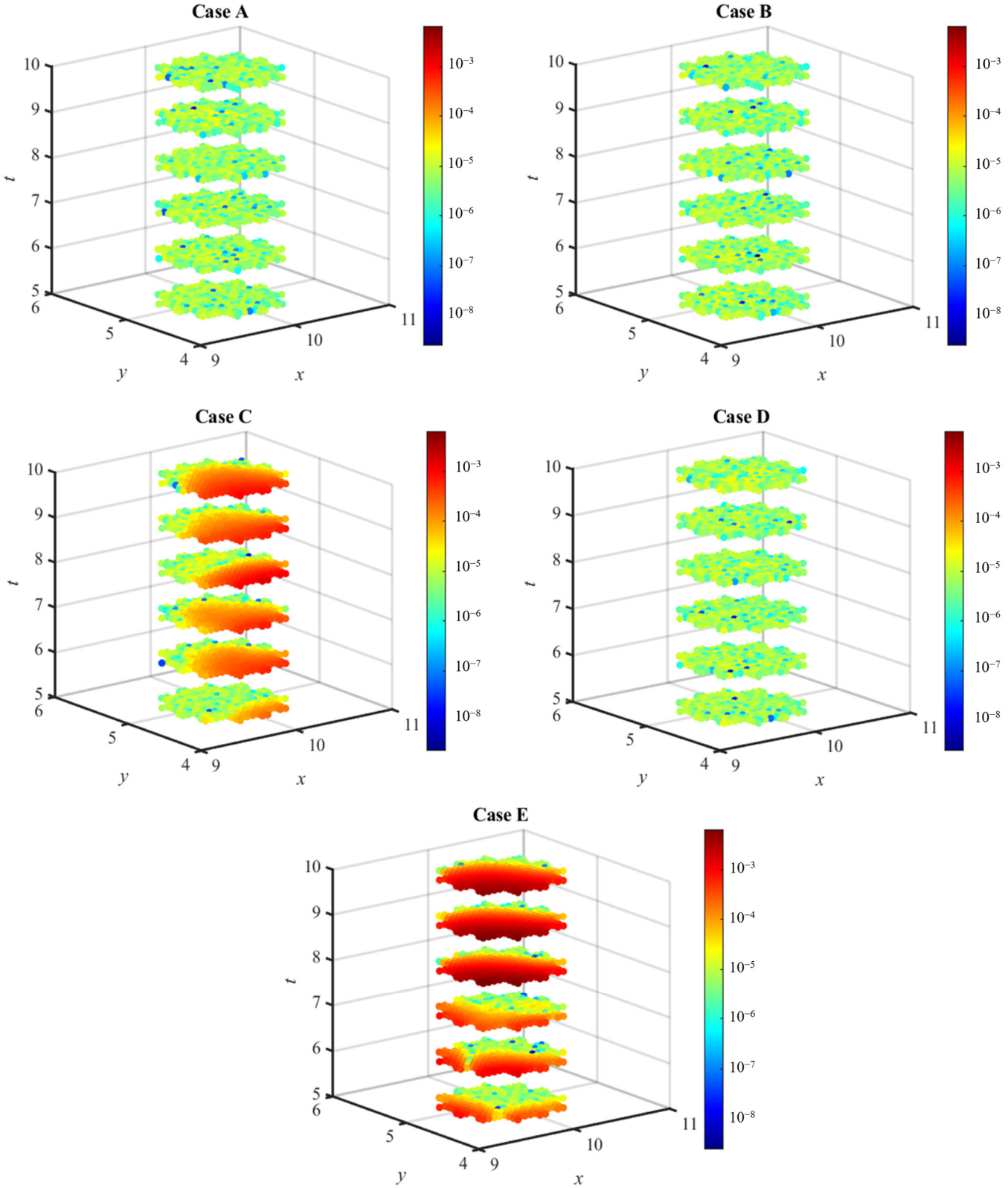


Figure 11. The maximum relative error within the spacetime domain (considering noise).

5. Conclusions

This study introduces a groundbreaking application of RBFs in a novel DNN approach designed to solve forward, backward, and inverse diffusion problems. This study em-

ploys backpropagation neural networks to enhance accuracy through training data. The significant findings are summarized as follows.

The conceptualization of spacetime coordinates is employed in the proposed DNN with the spacetime RBF to transform transient problems into inverse boundary value problems. Consequently, initial and boundary data are treated as spacetime boundary conditions. By placing boundary points on spacetime boundaries and configuring source points outside of spacetime boundaries, the diffusion problems can be solved. Unlike conventional numerical methods, this approach not only reduces the complexity of mathematical equation derivation as there is no need for the discretization of the temporal and spatial terms of the diffusion problem but also eliminates the need for the determination of the shape parameter, resulting in high-precision solutions.

This study addresses various types of diffusion problems, including forward, backward, and inverse diffusion problems, to validate the proposed methodology. The proposed method demonstrates advantages in achieving high-precision numerical solutions for inverse diffusion problems using RBFs and partial boundary data. Even under conditions involving noise, this study consistently exhibits a significant highlight.

Moreover, to illustrate the robustness of the proposed DNN with the spacetime RBF in addressing the inverse diffusion problem while considering the impact of noise, a noise level is incorporated. The analysis results indicate that, even with noise affecting the provided conditions, the RMSE values for the five scenarios remain highly accurate. Notably, in the fifth scenario, where only 1/4 of the initial conditions and 1/4 of the boundary conditions are known, the method still yields precise results. These research findings illustrate the high accuracy achieved by the proposed method in solving inverse diffusion problems.

Author Contributions: Design, C.-Y.K.; methodology, writing and editing, C.-Y.L.; data curation, Y.-J.C.; formal analysis and visualization, W.-D.C. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Data Availability Statement: Data can be obtained upon request.

Conflicts of Interest: The authors declare no conflicts of interest.

References

1. Lin, J.; Bai, J.; Reutskiy, S.; Lu, J. A novel RBF-based meshless method for solving time-fractional transport equations in 2D and 3D arbitrary domains. *Eng. Comput.* **2023**, *39*, 1905–1922. [CrossRef]
2. Tartakovsky, A.M.; Marrero, C.O.; Perdikaris, P.; Tartakovsky, G.D.; Barajas-Solano, D. Physics-informed deep neural networks for learning parameters and constitutive relationships in subsurface flow problems. *Water Resour. Res.* **2020**, *56*, e2019WR026731. [CrossRef]
3. Camacho, M.; Edwards, B.; Engheta, N. Achieving asymmetry and trapping in diffusion with spatiotemporal metamaterials. *Nat. Commun.* **2020**, *11*, 3733. [CrossRef] [PubMed]
4. Li, P.W.; Hu, S.; Zhang, M. Numerical solutions of the nonlinear dispersive shallow water wave equations based on the space–time coupled generalized finite difference scheme. *Appl. Sci.* **2023**, *13*, 8504. [CrossRef]
5. Sun, L.; Qiu, H.; Wu, C.; Niu, J.; Hu, B.X. A review of applications of fractional advection–dispersion equations for anomalous solute transport in surface and subsurface water. *Wiley Interdiscip. Rev. Water* **2020**, *7*, e1448. [CrossRef]
6. Ku, C.Y.; Liu, C.Y.; Xiao, J.E.; Yeih, W.; Fan, C.M. A spacetime meshless method for modeling subsurface flow with a transient moving boundary. *Water* **2019**, *11*, 2595. [CrossRef]
7. Zheng, X.; Wang, H. Optimal-order error estimates of finite element approximations to variable-order time-fractional diffusion equations without regularity assumptions of the true solutions. *IMA J. Numer. Anal.* **2021**, *41*, 1522–1545. [CrossRef]
8. Li, P.W.; Fan, C.M.; Grabski, J.K. A meshless generalized finite difference method for solving shallow water equations with the flux limiter technique. *Eng. Anal. Bound. Elem.* **2021**, *131*, 159–173. [CrossRef]
9. Ku, C.Y.; Liu, C.Y.; Xiao, J.E.; Hsu, S.M.; Yeih, W. A collocation method with space–time radial polynomials for inverse heat conduction problems. *Eng. Anal. Bound. Elem.* **2021**, *122*, 117–131. [CrossRef]
10. Ku, C.Y.; Liu, C.Y. A Novel Spacetime Boundary-Type Meshless Method for Estimating Aquifer Hydraulic Properties Using Pumping Tests. *Mathematics* **2023**, *11*, 4497. [CrossRef]
11. Hardy, R.L. Multiquadric equations of topography and other irregular surfaces. *J. Geophys. Res.* **1971**, *76*, 1905–1915. [CrossRef]

12. Kaennakham, S.; Chuathong, N. Solution to a convection-diffusion problem using a new variable inverse-multiquadric parameter in a collocation meshfree scheme. *Int. J. Multiphys.* **2017**, *11*, 359–374.
13. Rashidinia, J.; Khasi, M.; Fasshauer, G.E. A stable Gaussian radial basis function method for solving nonlinear unsteady convection–diffusion–reaction equations. *Comput. Math. Appl.* **2018**, *75*, 1831–1850. [CrossRef]
14. Gunderman, D.; Flyer, N.; Fornberg, B. Transport schemes in spherical geometries using spline-based RBF-FD with polynomials. *J. Comput. Phys.* **2020**, *408*, 109256. [CrossRef]
15. Geist, M.; Petersen, P.; Raslan, M.; Schneider, R.; Kutyniok, G. Numerical solution of the parametric diffusion equation by deep neural networks. *J. Sci. Comput.* **2021**, *88*, 22. [CrossRef]
16. Li, A.; Chen, R.; Farimani, A.B.; Zhang, Y.J. Reaction diffusion system prediction based on convolutional neural network. *Sci. Rep.* **2020**, *10*, 3894. [CrossRef] [PubMed]
17. Wu, H.; Fang, W.Z.; Kang, Q.; Tao, W.Q.; Qiao, R. Predicting effective diffusivity of porous media from images by deep learning. *Sci. Rep.* **2019**, *9*, 20387. [CrossRef] [PubMed]
18. Chen, Z.; Liu, Y.; Sun, H. Physics-informed learning of governing equations from scarce data. *Nat. Commun.* **2021**, *12*, 6136. [CrossRef] [PubMed]
19. Zobeiry, N.; Humfeld, K.D. A physics-informed machine learning approach for solving heat transfer equation in advanced manufacturing and engineering applications. *Eng. Appl. Artif. Intell.* **2021**, *101*, 104232. [CrossRef]
20. Granik, N.; Weiss, L.E.; Nehme, E.; Levin, M.; Chein, M.; Perlson, E.; Shechtman, Y. Single-particle diffusion characterization by deep learning. *Biophys. J.* **2019**, *117*, 185–192. [CrossRef]
21. Apaydin, H.; Feizi, H.; Sattari, M.T.; Colak, M.S.; Shamshirband, S.; Chau, K.W. Comparative analysis of recurrent neural network architectures for reservoir inflow forecasting. *Water* **2020**, *12*, 1500. [CrossRef]
22. Basha, S.S.; Dubey, S.R.; Pulabaigari, V.; Mukherjee, S. Impact of fully connected layers on performance of convolutional neural networks for image classification. *Neurocomputing* **2020**, *378*, 112–119. [CrossRef]
23. Alabsi, B.A.; Anbar, M.; Rihan, S.D.A. CNN-CNN: Dual Convolutional Neural Network Approach for Feature Selection and Attack Detection on Internet of Things Networks. *Sensors* **2023**, *23*, 6507. [CrossRef]
24. Bashar, D.A. Survey on evolving deep learning neural network architectures. *J. Artif. Intell. Capsul. Netw.* **2019**, *1*, 73–82. [CrossRef]
25. Asghari, V.; Leung, Y.F.; Hsu, S.C. Deep neural network based framework for complex correlations in engineering metrics. *Adv. Eng. Inform.* **2020**, *44*, 101058. [CrossRef]
26. Mostajeran, F.; Mokhtari, R. DeepBHCP: Deep neural network algorithm for solving backward heat conduction problems. *Comput. Phys. Commun.* **2022**, *272*, 108236. [CrossRef]
27. Mostajeran, F.; Hosseini, S.M. Radial basis function neural network (RBFNN) approximation of Cauchy inverse problems of the Laplace equation. *Comput. Math. Appl.* **2023**, *141*, 129–144. [CrossRef]
28. Ruan, Z.; Wang, Z. A backward problem for distributed order diffusion equation: Uniqueness and numerical solution. *Inverse Probl. Sci. Eng.* **2021**, *29*, 418–439. [CrossRef]
29. Liu, C.S.; Kuo, C.L.; Jhao, W.S. The multiple-scale polynomial Trefftz method for solving inverse heat conduction problems. *Int. J. Heat Mass Transf.* **2016**, *95*, 936–943. [CrossRef]
30. Ku, C.Y.; Liu, C.Y.; Yeih, W.; Liu, C.S.; Fan, C.M. A novel space–time meshless method for solving the backward heat conduction problem. *Int. J. Heat Mass Transf.* **2019**, *130*, 109–122. [CrossRef]
31. Wunsch, A.; Liesch, T.; Broda, S. Deep learning shows declining groundwater levels in Germany until 2100 due to climate change. *Nat. Commun.* **2022**, *13*, 1221. [CrossRef] [PubMed]
32. Grohs, P.; Herrmann, L. Deep neural network approximation for high-dimensional elliptic PDEs with boundary conditions. *IMA J. Numer. Anal.* **2022**, *42*, 2055–2082. [CrossRef]
33. Liu, C.Y.; Ku, C.Y. A Novel ANN-Based Radial Basis Function Collocation Method for Solving Elliptic Boundary Value Problems. *Mathematics* **2023**, *11*, 3935. [CrossRef]
34. Stelzer, F.; Röhm, A.; Vicente, R.; Fischer, I.; Yanchuk, S. Deep neural networks using a single neuron: Folded-in-time architecture using feedback-modulated delay loops. *Nat. Commun.* **2021**, *12*, 5164. [CrossRef] [PubMed]
35. Aspri, M.; Tsagkatakis, G.; Tsakalides, P. Distributed training and inference of deep learning models for multi-modal land cover classification. *Remote Sens.* **2020**, *12*, 2670. [CrossRef]
36. de Jesús Rubio, J. Stability analysis of the modified Levenberg–Marquardt algorithm for the artificial neural network training. *IEEE Trans. Neural Netw. Learn. Syst.* **2020**, *32*, 3510–3524. [CrossRef]
37. Kansa, E.J. Multiquadrics—A scattered data approximation scheme with applications to computational fluid-dynamics—II solutions to parabolic, hyperbolic and elliptic partial differential equations. *Comput. Math. Appl.* **1990**, *19*, 147–161. [CrossRef]