



Enhancing inverse modeling in groundwater systems through machine learning: a comprehensive comparative study

Junjun Chen^{1,2}, Zhenxue Dai^{2,3}, Shangxian Yin⁴, Mingkun Zhang⁵, and Mohamad Reza Soltanian⁶

¹National and Local Joint Engineering Laboratory of Internet Application Technology on Mine, China University of Mining and Technology, Xuzhou, 221008, China

²College of Construction Engineering, Jilin University, Changchun, 130026, China

³School of Environmental and Municipal Engineering, Qingdao University of Technology, Qingdao, 273400, China

⁴College of Safety Engineering, North China Institute of Science and Technology, Langfang, 065201, China

⁵Shandong Ruyi Technology Group Co., Ltd., Jinan, 250000, China

⁶Departments of Geosciences and Environmental Engineering, University of Cincinnati, Cincinnati, OH 45220, USA

Correspondence: Zhenxue Dai (dzx@jlu.edu.cn) and Shangxian Yin (yinshx03@126.com)

Received: 12 October 2024 – Discussion started: 6 December 2024

Revised: 9 June 2025 – Accepted: 16 June 2025 – Published: 10 September 2025

Abstract. Tandem neural network architecture (TNNA) is a machine learning algorithm that has recently been proposed for estimating uncertain parameters with inverse mappings. However, its reliability has only been validated in limited research scenarios, and its advantages over conventional methods remain underexplored. This study systematically compares the performance of the TNNA algorithm to four traditional metaheuristic algorithms across three heterogeneity scenarios, each employing a specific inversion framework: (i) a surrogate model coupled with an optimization algorithm for cases with eight homogeneous parameter zones, (ii) Karhunen–Loève expansion (KLE)-based dimensionality reduction combined with a surrogate model and an optimization algorithm for a high-dimensional Gaussian random field, and (iii) generative machine-learning-based dimensionality reduction integrated with a surrogate model and an optimization algorithm for a high-dimensional non-Gaussian random field. Additionally, we evaluate algorithm performance under two different noise-level conditions (multiplicative Gaussian noise with standard deviations of 1 % and 10 %) for normalized hydraulic head and solute concentration data in the non-Gaussian random field scenario, which exhibits the most complex parameter characteristics. The results demonstrate that both the TNNA algorithm and the metaheuristic algorithms achieve inversion results that satisfy the convergence accuracy within these machine-learning-based inversion frameworks. Moreover, under the 10 % high-noise con-

dition in the non-Gaussian random field, the inversion results remain robust when sufficient constraints are imposed. Compared to metaheuristic approaches, the TNNA method yields more reliable inversion results with significantly higher computational efficiency, highlighting the considerable advantages of machine learning in advancing groundwater system inversions.

1 Introduction

Numerical models are essential for quantifying flow and mass transport dynamics within aquifers, providing significant insights into hydrological and biogeochemical processes (Steefel et al., 2005; Sanchez-Vila et al., 2010; Sternagel et al., 2021; Xu et al., 2022). However, directly measuring aquifer parameters, such as permeability fields, remains challenging due to limitations in the current hydrogeological exploration techniques and budgetary constraints (Yeh, 1986; Kool et al., 1987; Beven and Binley, 1992; McLaughlin and Townley, 1996; Dai and Samper, 2004; Castaings et al., 2009; Chen et al., 2021). Inverse modeling has become a key approach for estimating these uncertain model parameters, improving the accuracy of numerical simulations (Ginn and Cushman, 1990; Carrera and Glorioso, 1991; Hopmans et al., 2002; Zheng and Samper, 2004; Zhou et al., 2014; Bandai and Ghezzehei, 2022; Abbas et al., 2024; Giudici, 2024).

Inverse modeling within Bayesian theorem-based data assimilation frameworks has garnered significant attention from the hydrogeological community over the past few decades (Scharnagl et al., 2011; Chen et al., 2013; Zhang et al., 2018; Xia et al., 2021). Methods based on the minimization of objective functions or the maximization of posterior distributions require the application of optimization techniques (Tsai et al., 2003; Blasone et al., 2007; Sun, 2013; Vrugt, 2016). One type is local optimization algorithms, which update model parameters from initial guesses towards optimal solutions according to gradient directions, such as the Gaussian–Newton method (Dragonetti et al., 2018; Qin et al., 2022) and the Levenberg–Marquardt method (Schneider-Zapp et al., 2010; Nhu, 2022). These methods are highly efficient but may converge to local optima when dealing with non-convex inversion problems. Another category is to achieve global optima solutions through metaheuristic searches, which typically incorporate processes of exploration (to search the entire parameter space for a diverse range of estimates) and exploitation (to leverage local information to refine estimates). Popular metaheuristic algorithms include the genetic algorithm (GA) (Ines and Droogers, 2002; Lindsay et al., 2016), simulated annealing (SA) (Kirkpatrick et al., 1983; Jaumann and Roth, 2018), differential evolution (DE) (Li, 2019; Yan et al., 2023), and particle swarm optimization (PSO) (Rafiei et al., 2022; Travaš et al., 2023). Nevertheless, their computational efficiency may be reduced by extensive exploration and exploitation processes in achieving globally optimal inversion results. The efficiency of optimization algorithms can be enhanced by integrating them with adjoint methods, particularly when extended to high-dimensional parameter spaces. Adjoint methods are capable of efficiently computing gradients for all parameters simultaneously through solving adjoint equations derived from the original forward model (Plessix, 2006). This gradient information can directly accelerate local optimization algorithms (Epp et al., 2023) and facilitate gradient-enhanced global optimization methods (Kapsoulis et al., 2018), significantly improving efficiency in complex inverse problems. However, the practical implementation of adjoint methods remains challenging due to the complexity associated with deriving adjoint equations, especially for highly non-linear system models (Xiao et al., 2021; Ghelichkhan et al., 2024). The accurate and efficient estimation of uncertain model parameters across various scenarios remains one of the most significant challenges for developing inversion frameworks.

In recent years, machine learning has experienced rapid developments and demonstrated significant performance in addressing complex problems characterized by high dimensionality and non-linearity (Hinton and Salakhutdinov, 2006; LeCun et al., 2015; Bentivoglio et al., 2022; Shen et al., 2023). Integrating conventional inversion methods with cutting-edge machine learning techniques has become increasingly popular in addressing the challenges of inversion

studies. One effective strategy is constructing surrogate models to accelerate forward simulations, ensuring that inversion algorithms perform comprehensive searches across the entire parameter space more efficiently (Razavi et al., 2012). For instance, Zhan et al. (2021) identified lithofacies structures by utilizing a deep octave convolution residual network to construct a surrogate model for predicting solute concentrations and hydraulic heads in heterogeneous aquifers. Wang et al. (2021) constructed a subsurface flow surrogate model under heterogeneous conditions through physically informed neural network methods, specifically for uncertainty quantification and parameter inversion. Liu et al. (2023) constructed a convolutional neural network (CNN) surrogate model to combine with a hierarchical homogenization method to estimate the effective permeability of digital rocks. More related studies can also be found in recent reviews (Yu and Ma, 2021; Luo et al., 2023; Zhan et al., 2023). Additionally, due to their inherent differentiability and continuity, deep neural network (DNN)-based surrogate models can be integrated with adjoint equations, enabling efficient gradient computations and significantly facilitating their practical implementation in high-dimensional and complex scenarios (Xiao et al., 2021).

In addition to surrogate models, parameter optimization through machine-learning-based reverse mapping represents another significant advancement in inversion techniques. Previous studies have outlined at least two strategies to achieve reverse mapping models. The first strategy is the data-driven approach, where reverse regressions are trained using datasets that comprise pairs of model outputs and inputs. For example, Sun (2018) developed a regression model from hydraulic heads to heterogeneous conductivity fields using a CNN-based generative adversarial network (GAN) approach. Kuang et al. (2021) succeeded in the real-time identification of earthquake focal mechanisms by training a DNN regression on seismic waveform data. Yang et al. (2022) established the relationship between gravity data and CO₂ plumes to perform real-time inversion for geologic carbon sequestration. Another strategy is to train a reverse network in tandem neural network architecture (TNNA), integrated with a pre-trained surrogate model (i.e., forward network). The TNNA method was introduced with the advent of deep learning and has been successfully applied in computed tomography reconstruction (Adler and Öktem, 2017), nanophotonic structure inverse design (Liu et al., 2018; Yeung et al., 2021), and photonic topological state inverse design (Long et al., 2019). Our previous research expanded the application of the TNNA algorithm in groundwater science, evaluating its performance in reactive transport inverse modeling and improving inversion results by incorporating an adaptive update strategy to reduce local predictive errors of surrogate models. The findings indicated that accurate surrogate model predictive results for the actual parameter values yield dependable TNNA inversion outcomes (Chen et al., 2021).

The TNNA algorithm demonstrates a fundamental advantage by requiring only a single forward simulation to update parameters in each iteration. In contrast, conventional metaheuristic algorithms typically necessitate multiple forward simulations. Despite the innovation of this approach, its applicability in more general groundwater numerical scenarios and its performance compared to conventional metaheuristic algorithms remain uncertain. This study considers three cases with different heterogeneity characteristics to compare the performance of the TNNA algorithm to four conventional metaheuristic algorithms. In case 1, the domain is divided into a finite number of homogeneous zones. The other two cases focus on high-dimensional parameter fields based on the spatial variability of the aquifer. These two cases are essential for revealing the dynamic behaviors of the groundwater system at the discrete grid scale. Depending on the spatial variability of the aquifer structure, the two high-dimensional numerical cases characterize the heterogeneity of aquifer parameters using a Gaussian random field (i.e., case 2) and a non-Gaussian random field (i.e., case 3). The Gaussian random field is suited to aquifers with a single lithofacies and relatively uniform physical structures, where the spatial variation of the parameter values is quite smooth. In contrast, the non-Gaussian random field accounts for the existence of a nugget effect in the aquifer structure, such as when it contains multiple lithofacies with varying hydraulic properties (Mariethoz and Caers, 2014). For a comparative study of the three cases, surrogate models will be used to accelerate the forward simulation. Additionally, dimensionality reduction techniques are necessary for the two high-dimensional cases to reduce the computational complexity associated with high-dimensional parameter spaces. Specifically, the Karhunen–Loève expansion (KLE) method is feasible for Gaussian random fields. It reconstructs the Gaussian random field through a linear combination of orthogonal basis functions, achieving dimensionality reduction by retaining the dominant modes corresponding to the largest eigenvalues (Loève, 1955; Zhang and Lu, 2004; Mariethoz and Caers, 2014). However, the second-order statistics relied upon by KLE are insufficient to fully represent complex characteristics for non-Gaussian random fields. In recent years, generative machine learning methods have demonstrated outstanding performance in parameter field reconstruction (Mo et al., 2020; Zhan et al., 2021; Guo et al., 2023). These methods can establish relationships between low-dimensional standard distributions (e.g., uniform distribution) and high-dimensional distributions, effectively representing non-Gaussian random fields as low-dimensional latent vectors (i.e., parameters after dimensionality reduction). Thus, extending the TNNA framework by integrating KLE and generative machine learning methods, respectively, is a potentially feasible approach for solving the high-dimensional heterogeneous aquifer parameter inversion problems presented in case 2 and case 3. In

summary, the primary contributions of this study are as follows:

1. A novel inversion framework is proposed that integrates the TNNA algorithm with dimensionality reduction techniques, including KLE for Gaussian stochastic processes and generative machine learning methods for non-Gaussian stochastic processes, thereby extending its applicability to high-dimensional heterogeneous fields.
2. A comprehensive comparative analysis is conducted between the TNNA algorithm and four conventional metaheuristic algorithms across three case scenarios, highlighting the advantages of DNN-based reverse mapping over metaheuristic stochastic search strategies for inverse estimation under different heterogeneous conditions.

The sections of this paper are structured as follows: Sect. 2 introduces the fundamental principles of the methodology involved in this study. Section 3 provides detailed information on numerical models for the three cases. Section 4 presents the results and discussions. Finally, Sect. 5 presents a summary and conclusions derived from this research, along with recommendations for future studies.

2 Methodology

The inversion framework, based on non-linear optimization theory, generally consists of two aspects: (1) constructing non-linear constraints for the optimization of uncertain model parameters and (2) establishing optimization algorithms to search for the model parameters that satisfy these constraints. The general form of the non-linear optimization model in this paper is as follows:

$$m^* = \min \sum_{i=1}^{N_{\text{obs}}} \frac{1}{\sigma_i} [y_{\text{obs}}[i] - \hat{y}[i]]^2$$

$$\begin{cases} \hat{y} = \mathbf{F}_{\text{HF}}(\mathbf{m}) \\ \mathbf{m}^{\text{L}} \leq \mathbf{m} \leq \mathbf{m}^{\text{U}}, \end{cases} \quad (1)$$

where $\mathbf{y}_{\text{obs}} \in \mathbf{R}^{N_{\text{obs}} \times 1}$ and $\hat{\mathbf{y}} \in \mathbf{R}^{N_{\text{obs}} \times 1}$ represent the observed data vector and the corresponding model simulation output vector, respectively. $y_{\text{obs}}[i]$ and $\hat{y}[i]$ refer to the i th element of the observed and simulated vectors, respectively, and σ_i denotes the standard deviation of the i th observed data. \mathbf{m} represents the vector of model parameters to be optimized; \mathbf{m}^* denotes the optimal parameter vector obtained through optimization; and \mathbf{m}^{L} and \mathbf{m}^{U} are the vectors representing the lower and upper limit values of the model parameters, respectively. $\mathbf{F}_{\text{HF}}(\cdot)$ represents the high-fidelity numerical model.

In this study, three different inversion frameworks are developed to compare the TNNA algorithms to four meta-heuristic algorithms. In a low-dimensional parameter scenario, a surrogate model $\mathbf{F}_{\text{Forward}}(\cdot)$ is constructed to approximate high-fidelity numerical prediction outputs. Therefore, the objective function of the inversion framework integrated with a surrogate model is as follows:

$$\mathbf{m}^* = \min \sum_{i=1}^{N_{\text{obs}}} \frac{1}{\sigma_i} [\mathbf{y}_{\text{obs}}[i] - \mathbf{F}_{\text{Forward}}(\mathbf{m})[i]]^2. \quad (2)$$

In high-dimensional parameter scenarios, directly optimizing the model parameter \mathbf{m} can lead to computational difficulties due to its high dimensionality. To mitigate this issue, in addition to constructing a surrogate model $\mathbf{F}_{\text{Forward}}(\cdot)$ to improve the computational efficiency of forward simulations, dimensionality reduction algorithms are also integrated into the inversion frameworks. Let $\mathbf{m} = \mathbf{G}(\mathbf{z})$ represent an operator for parameter dimensionality reduction, where \mathbf{z} is a low-dimensional vector whose parameter space is commonly defined as an easily sampled probability distribution (e.g., standard Gaussian or uniform distribution). Specifically, the Karhunen–Loève expansion (KLE) and the octave convolution adversarial autoencoder (OCAAE) are used for representing Gaussian random fields and non-Gaussian random fields, respectively. Once the low-dimensional vector representation of the high-dimensional parameter is obtained, the high-dimensional parameter \mathbf{m} can be indirectly optimized by estimating the low-dimensional vector \mathbf{z} :

$$\begin{aligned} \mathbf{z}^* &= \min \sum_{i=1}^{N_{\text{obs}}} \frac{1}{\sigma_i} [\mathbf{y}_{\text{obs}}[i] - \mathbf{F}_{\text{Forward}}(\mathbf{G}(\mathbf{z}))[i]]^2 \\ \mathbf{m}^* &= \mathbf{G}(\mathbf{z}^*). \end{aligned} \quad (3)$$

The basic mathematical theories of surrogate models, dimensionality reduction techniques, and optimization algorithms are introduced in Sect. 2.1–2.3, respectively.

2.1 Surrogate modeling methods

In this study, surrogate models $\mathbf{F}_{\text{Forward}}(\cdot)$ are developed using a data-driven strategy, as shown in Fig. 1. The process begins by sampling model parameters from prior distributions. The corresponding system responses for these parameter samples are simulated using a high-fidelity numerical model. Then, a training dataset consisting of paired model parameters and responses is obtained, which is subsequently used to construct surrogate models via supervised machine learning. Specifically, four popular machine learning models with distinct architectural differences are evaluated for surrogate modeling. These are multi-output support vector regression (MSVR), a kernel-based architecture for data mapping; a fully connected deep neural network (FC-DNN), composed of stacked fully connected layers; LeNet, a classical CNN architecture proposed by Yann LeCun; and a deep residual

convolutional neural network (ResNet), which incorporates residual connections into the CNN structure.

The detailed principles of MSVR and the three deep-learning-based methods are illustrated in the following two subsections. The predictive accuracy of four surrogate modeling approaches will be compared in this study, and the best-performing approach among them will subsequently be selected for inversion computations. Before constructing surrogate models, the training datasets are normalized separately for each simulation component using min–max normalization, in which each component is scaled independently based on its minimum and maximum values, ensuring that all normalized values fall within the range [0, 1].

2.1.1 Multi-output support vector regression

MSVR is developed from the original support vector machine (SVM) for realizing multivariate regression (Pérez-Cruz et al., 2002; Tuia et al., 2011). The mathematical expression is given as follows:

$$\hat{\mathbf{y}} = F_{\text{MSVR}}(\mathbf{m}) = \varphi(\mathbf{m})^T \mathbf{W} + \mathbf{B}, \quad (4)$$

where $F_{\text{MSVR}}(\cdot)$ denotes the dataset regression model operator constructed based on MSVR and $\varphi(\mathbf{m})$ is a non-linear regression function that implicitly maps the input vector \mathbf{m} into a high-dimensional feature space. Its inner product defines the kernel function $K(\mathbf{m}, \mathbf{m}_i)$. (Here, we use the Gaussian radial basis function (RBF) kernel with a bandwidth parameter σ : $K(\mathbf{m}, \mathbf{m}_i) = \varphi(\mathbf{m})^T \varphi(\mathbf{m}_i) = \exp(-0.5 \|\mathbf{m} - \mathbf{m}_i\|^2 / \sigma^2)$; \mathbf{m}_i denotes the i th model parameter vector from the surrogate model training dataset.) Assuming N_{samples} denotes the number of surrogate model training samples, the regression coefficients $\mathbf{W} = [\mathbf{w}^1, \dots, \mathbf{w}^{N_{\text{obs}}}]^T \in \mathbb{R}^{N_{\text{obs}} \times N_{\text{samples}}}$ and $\mathbf{B} = [b^1, \dots, b^{N_{\text{obs}}}]^T \in \mathbb{R}^{N_{\text{obs}} \times 1}$ are determined by minimizing the structural risk, as outlined in Eqs. (5) and (6):

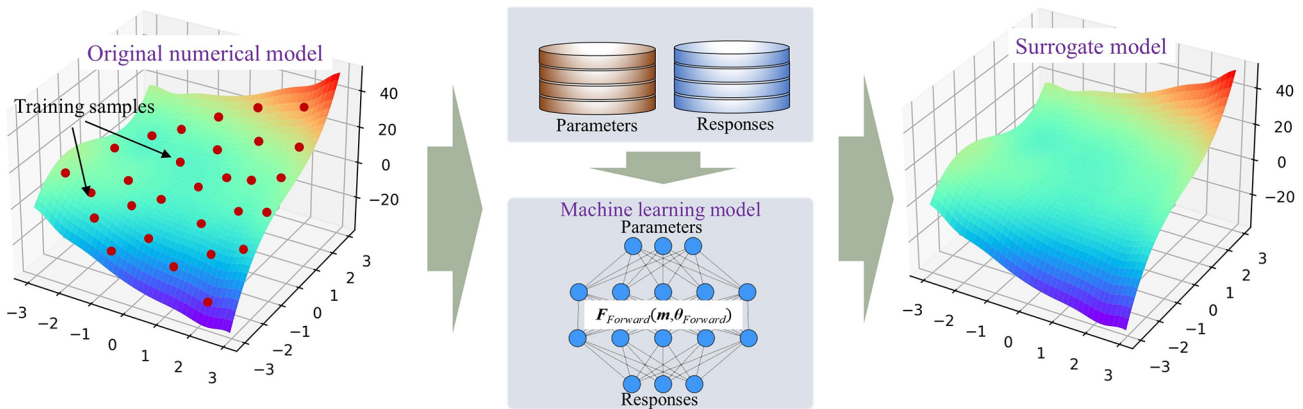
$$\begin{aligned} \mathbf{W}, \mathbf{B} &= \operatorname{argmin} L(\mathbf{W}, \mathbf{B}) \\ &= \frac{1}{2} \sum_{j=1}^{N_{\text{obs}}} \|\mathbf{w}^j\|^2 + C \sum_{j=1}^{N_{\text{samples}}} L(u_i), \end{aligned} \quad (5)$$

where C is a penalty parameter; and $L(u)$ is a quadratic ε -insensitive loss function, expressed as

$$L(u) = \begin{cases} 0, & u < \varepsilon \\ (u - \varepsilon)^2, & u \geq \varepsilon, \end{cases} \quad (6)$$

where $u_i = \|\mathbf{e}_i\| = \sqrt{\mathbf{e}_i^T \mathbf{e}_i}$; $\mathbf{e}_i^T = \mathbf{y}_i^T - \varphi^T(\mathbf{m}_i) \mathbf{W} - \mathbf{B}^T$. For $\varepsilon = 0$, this problem is equivalent to an independent regularized kernel least squares regression for each component. For $\varepsilon \neq 0$, it becomes feasible to develop individual regression functions for each dimension based on the model outputs and to generate their corresponding support vectors. Solving the optimization problem directly is challenging, and the desired solutions for \mathbf{W} and \mathbf{B} are determined using an iterative reweighted least squares (IRWLS) procedure, employing

(a) The framework for data driven based surrogate model construction



(b) Machine learning models

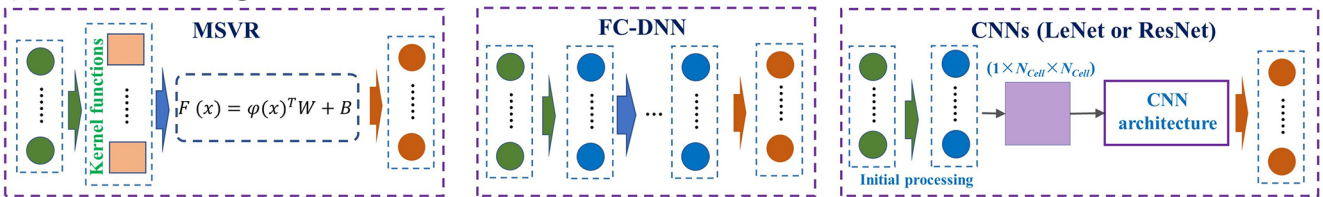


Figure 1. The framework for data-driven surrogate model construction and the machine learning models employed. Note that for CNN-based surrogate models, the initial processing module is activated only for low-dimensional scenarios, whereas in high-dimensional scenarios, the parameter matrix ($1 \times N_{\text{Cell}} \times N_{\text{Cell}}$) is directly input into the CNN architecture.

the quasi-Newton approach. During the IRWLS process, the term $L(u)$ in Eq. (5) is first transformed into a discrete first-order Taylor expansion, and the corresponding quadratic programming approximation is constructed. Meanwhile, a linear expression is derived based on the principle that the first-order derivatives of the objective function with respect to \mathbf{W} and \mathbf{B} are zero. Finally, the optimal values of \mathbf{W} and \mathbf{B} are obtained through a line search. Further details on the IRWLS procedure can be found in Sanchez-Fernandez et al. (2004).

The performance of the MSVR model is influenced by three hyperparameters: C , σ , and ε (Ma et al., 2022). This study optimizes these hyperparameters by minimizing the root mean square error (RMSE) using the four metaheuristic algorithms introduced in this study.

2.1.2 Deep-learning-based surrogate models

(1) DNN architectures

The three DNN models are all feedforward neural networks, which are generally constructed by stacking multiple hidden layers. The structure can be expressed as $\mathbf{F}_{\text{DNN}}(\mathbf{m}, \boldsymbol{\theta}_{\text{DNN}}) = f_{L_{\text{NN}}}(\dots f_1(\dots f_1(\mathbf{m})))$. Specifically, $\mathbf{F}_{\text{DNN}}(\cdot)$ and $\boldsymbol{\theta}_{\text{DNN}}$ represent the DNN-based surrogate model operator and the corresponding trainable parameters, respectively. $f_l(\cdot)$ denotes the non-linear transformation function of the l th layer, and L_{NN} indicates the total number of neural network layers. In DNN model construction, various neural network layers

can yield diverse DNN models, resulting in different predictive performances (LeCun et al., 2015). For the DNN models adopted in this study, the involved neural network types are the fully connected layer, the convolutional layer, and the residual block layer.

In fully connected layers, both input and output layers are in vector forms. Assume $\mathbf{X}_{\text{input}} \in \mathbb{R}^{n \times 1}$ is the input vector and $\mathbf{X}_{\text{output}} \in \mathbb{R}^{m \times 1}$ is the output vector of the l th fully connected layer $f_l(\cdot)$. The transformation in this fully connected layer is expressed as

$$\mathbf{X}_{\text{output}} = f_l(\mathbf{X}_{\text{input}}, \boldsymbol{\theta}_l) = f_{\sigma-l}(\mathbf{W}_{\text{DNN}} \mathbf{X}_{\text{input}} + \mathbf{B}_{\text{DNN}}), \quad (7)$$

where $f_{\sigma-l}(\cdot)$ is a non-linear active function, $\mathbf{W}_{\text{DNN}} \in \mathbb{R}^{m \times n}$ is the weight matrix, and $\mathbf{B}_{\text{DNN}} \in \mathbb{R}^{m \times 1}$ is the bias vector.

In a convolutional layer, both the input and output are in matrix forms. A convolutional layer transfers information through sparse connections by several convolution kernels, essentially small matrices. The mathematical formula of a convolutional layer is as follows (Wang et al., 2019; Jardani et al., 2022):

$$h_{u,v}^q(x_{u,v}) = f_{\sigma-l} \left(\sum_{i=1}^{k'_i} \sum_{j=1}^{k'_j} w_{i,j}^q x_{u+i,v+j} + b \right), \quad (8)$$

where $x_{u,v}$ is the pixel value at position (u, v) of the input matrix and $h_{u,v}^q(x_{u,v})$ is the output feature calculated by employing the q th ($q = 1, \dots, N_{\text{out}}$) convolutional kernel filter

$\mathbf{w}^q \in \mathbb{R}^{k'_i \times k'_j}$. In a convolutional layer with N_{out} filters, the output matrix contains N_{out} feature layers. The output size (S_{out}) of each convolutional layer is determined by the input size (S_{in}) and the hyperparameters (i.e., zero padding p , kernel size k' , and stride s). A pooling layer is often used after a convolutional layer to remove redundant information from the extracted features and to improve the efficiency of model training (Chen et al., 2021).

The residual block is a fundamental component of residual networks (ResNets), designed primarily to mitigate the vanishing and exploding gradient problems commonly encountered during DNN training. A residual block learns a residual mapping, defined as

$$R(\mathbf{X}_{\text{input}}, \boldsymbol{\theta}_R) = H(\mathbf{X}_{\text{input}}) - T(\mathbf{X}_{\text{input}}), \quad (9)$$

where $\boldsymbol{\theta}_R$ represents the trainable parameters of a residual block, $R(\cdot)$ is the residual function, $H(\cdot)$ denotes the target mapping that the residual block aims to approximate, and $T(\cdot)$ is chosen as an identity transformation (i.e., $T(\mathbf{X}_{\text{input}}) = \mathbf{X}_{\text{input}}$) or another suitable transformation depending on the network architecture. The output of the residual block is computed as

$$\mathbf{X}_{\text{output}} = f_{\sigma-R}(R(\mathbf{X}_{\text{input}}, \boldsymbol{\theta}_R) + T(\mathbf{X}_{\text{input}})), \quad (10)$$

where $f_{\sigma-R}(\cdot)$ is the activation function of the rectified linear unit (ReLU). Such a design ensures that the output of the residual block at least approximates the input, effectively addressing the vanishing gradient problem. When stacking multiple residual blocks, the relationship between the L th residual block in a deeper layer and the l th residual block is expressed as follows (He et al., 2016):

$$\mathbf{X}_{\text{output}(L)} = \mathbf{X}_{\text{input}(l)} + \sum_{i=l}^{L-1} R(\mathbf{X}_{\text{output}(i)}, \boldsymbol{\theta}_{R(i)}), \quad (11)$$

where $\mathbf{X}_{\text{input}(i)}$ and $\boldsymbol{\theta}_{R(i)}$ denote the input data and trainable parameters of the i th residual block, respectively, and $\mathbf{X}_{\text{output}(L)}$ represents the output from the L th residual block. According to the chain rule in derivatives, the gradient of the loss function \mathbf{J}_{Res} with respect to $\mathbf{X}_{\text{input}(l)}$ can be given by

$$\frac{\partial \mathbf{J}_{\text{Res}}}{\partial \mathbf{X}_{\text{input}(l)}} = \frac{\partial \mathbf{J}_{\text{Res}}}{\partial \mathbf{X}_{\text{output}(L)}} \times \left(1 + \frac{\partial}{\partial \mathbf{X}_{\text{input}(l)}} \sum_{i=l}^{L-1} R(\mathbf{X}_{\text{output}(i)}, \boldsymbol{\theta}_{R(i)}) \right). \quad (12)$$

This formulation highlights two key properties of the residual network. First, the gradient does not vanish during network training processes because the term

$$\frac{\partial}{\partial \mathbf{X}_{\text{input}(l)}} \sum_{i=l}^{L-1} F(\mathbf{X}_{\text{input}(i)}, \boldsymbol{\theta}_{R(i)})$$

is never equal to -1 . Second, the gradient of the deepest residual block $\partial \mathbf{J}_{\text{Res}} / \partial \mathbf{X}_{\text{output}(L)}$ can directly affect all preceding layers, ensuring the effective transmission of gradients throughout the network (Chang et al., 2022).

Based on the three unique network layer structures described above, the FC-DNN, LeNet, and ResNet models are constructed. The FC-DNN of this study is constructed using fully connected layers, and each hidden layer consists of 512 neurons. The activation function for the output layer is the Sigmoid function, which constrains outputs within the range of 0 to 1. Note that other activation functions whose outputs ranges include $[0, 1]$ as a subset, such as the hyperbolic tangent (-1 to 1) and ReLU (0 to $+\infty$), could also be adopted. However, we specifically selected the Sigmoid function to strictly constrain initial model outputs within the target range (0 to 1), thereby reducing the risk of occasional extreme or anomalous predictions, particularly in the early stages of training. For hidden layers, the Swish activation function is adopted due to its smooth form with non-monotonic and continuously differentiable properties, which helps to improve the DNN training procedures (Elfwing et al., 2018). The performance of the FC-DNN is sensitive to the number of hidden layers, whose optimal value is determined based on specific case studies presented in the application section. For the LeNet and ResNet models, when dealing with low-dimensional scenarios, an initial processing module consisting of a fully connected layer followed by a reshaping operation is added to convert the input vector into a fixed-size matrix. In contrast, for high-dimensional parameter scenarios, the discrete grid matrix of the parameter field is directly input into the CNN architecture (see Fig. 1b). Specifically, LeNet consists of two convolutional blocks and two fully connected layers. Each convolutional block consists of a convolutional layer followed by a max-pooling layer. The fully connected layers have 1024 and 512 neurons, respectively. ResNet consists of four stages, and two different Res blocks are adopted. The first stage includes two residual units without down-sampling, while the remaining three stages each have one residual unit with down-sampling and one residual unit without down-sampling. Activation functions in all layers are rectified linear units (ReLUs), except for the output layer, where Sigmoid activation is used. Detailed architecture information for LeNet and ResNet is provided in Figs. S1 and S2 in the Supplement, respectively.

(2) DNN model training

The surrogate models are trained by minimizing the difference between the predicted outputs $\hat{\mathbf{y}}_i = \mathbf{F}_{\text{DNN}}(\mathbf{m}_i, \boldsymbol{\theta}_{\text{DNN}})$ and the corresponding numerical model outputs \mathbf{y}_i in the training datasets ($i = 1, \dots, N_{\text{samples}}$). Following prior studies (Mo et al., 2019, 2020; Chen et al., 2021), the L1 norm-

based loss function is adopted and formulated as

$$\theta_{\text{DNN}}^* = \underset{\theta_{\text{DNN}}}{\operatorname{argmin}} \frac{1}{N_{\text{samples}}} \sum_{i=1}^{N_{\text{samples}}} |\mathbf{F}_{\text{DNN}}(\mathbf{m}_i, \theta_{\text{DNN}}) - \mathbf{y}_i| + \frac{w_d}{2} \theta_{\text{DNN}}^T \theta_{\text{DNN}}, \quad (13)$$

where w_d is the weight decay to avoid overfitting, referred to as the regularization coefficient. It should be noted that the L2 norm can also be employed as a loss function in constructing surrogate model tasks. Due to its squared-error formulation, the L2 norm provides smoother gradients and more stable parameter updates near convergence compared to the L1 norm; however, this formulation also makes it more sensitive to extreme outliers. When the sampled parameters sparsely cover the parameter space, adopting the L1 norm loss function can improve the robustness of surrogate model predictions. This study implemented the DNN models using PyTorch (<https://pytorch.org/>, last access: 10 September 2024). The neural network weights were initialized using the default initialization method of PyTorch and optimized using the stochastic gradient descent method via the Adam algorithm. Specifically, the hyperparameter of weight decay can be set directly in the Adam optimizer without including it explicitly in the loss function.

When conducting DNN training, the hyperparameter selection primarily influences the update process of trainable parameters. Besides the weight decay mentioned above, the learning rate and the number of epochs are two other crucial hyperparameters directly affecting the training stability and convergence speed. A higher learning rate accelerates initial convergence but may lead to oscillations near the optimal solution, whereas a lower learning rate tends to improve final accuracy but requires more epochs to achieve convergence. In this study, we first set a relatively high number of epochs to ensure that the trainable parameters are adequately updated. Subsequently, appropriate learning rates and weight decay values for different scenarios are determined through a trial-and-error approach.

2.2 Dimensionality reduction methods

2.2.1 Karhunen–Loève expansion for Gaussian random field

Let $\mathbf{Y}_G(s) \sim \mathbf{N}(\boldsymbol{\mu}_G(s)\mathbf{C}(\cdot, \cdot))$ represent a Gaussian random field, where $\boldsymbol{\mu}_G$ denotes the mean of the random field and $\mathbf{C}(\cdot, \cdot)$ represents the exponential covariance function between two arbitrary spatial points $s = (s_x, s_y)$ and $s' = (s'_x, s'_y)$. The covariance function for these two spatial locations is given by

$$\mathbf{C}(s, s') = \sigma_G^2 \exp \left(-\sqrt{\left(\frac{s_x - s'_x}{\lambda_x} \right)^2 + \left(\frac{s_y - s'_y}{\lambda_y} \right)^2} \right), \quad (14)$$

where σ_G^2 is the variance and λ_x and λ_y are the correlation lengths along the x and y directions, respectively. Since the covariance matrix is symmetric and positive definite, the exponential covariance function in Eq. (14) can be decomposed into an eigenvalue–eigenfunction representation. By solving the second-kind Fredholm integral equation and performing eigenvalue decomposition, the Gaussian random field can be expressed through the Karhunen–Loève expansion (KLE) as follows:

$$\mathbf{Y}_G(s) = \boldsymbol{\mu}_G(s) + \sum_{i=1}^{\infty} \mathbf{z}_i \sqrt{\lambda_i} \phi_i(s), \quad (15)$$

where \mathbf{z}_i represents a random variable following a Gaussian distribution of $\mathbf{z}_i \sim \mathcal{N}(0, 1)$, also known as a KL term; and $\phi_i(s)$ and λ_i denote the eigenfunction and eigenvalue, respectively. For discretized numerical models, the index i takes values from 1 to n , representing the number of discrete grid points (e.g., in Eq. 15, ∞ is replaced by n). Dimensionality reduction using KLE is achieved through a truncated expansion (Loève, 1955; Zhang and Lu, 2004; Mariethoz and Caers, 2014).

2.2.2 Octave convolution adversarial autoencoder for non-Gaussian random field

The octave convolutional adversarial autoencoder (OCAAE) is a generative machine learning approach that combines the variational autoencoder (VAE) with adversarial learning, leveraging octave convolution neural networks (Zhan et al., 2021). It consists of three main components: an encoder, a decoder, and a discriminator. The encoder maps a high-dimensional parameter field \mathbf{m}_I to a low-dimensional latent vector \mathbf{z}_i . The distribution of the latent vectors $\{\mathbf{z}_1, \dots, \mathbf{z}_N\}$, obtained by mapping the N prior model parameter samples $\{\mathbf{m}_1, \dots, \mathbf{m}_N\}$, is denoted as $\mathbf{z} \sim \mathbf{q}(\mathbf{z})$. Specifically, the encoder outputs two low-dimensional vectors: the mean vector $\boldsymbol{\mu}_z$ and the log-variance vector $\ln(\sigma_z^2)$ of the latent vector \mathbf{z} . Then, a vector \mathbf{z}' is randomly drawn from a standard normal distribution $\mathcal{N}(0, \mathbf{I})$, and the latent vector is produced as $\mathbf{z} = \boldsymbol{\mu}_z + \sigma_z \times \mathbf{z}'$. The decoder reconstructs the high-dimensional parameter field $\tilde{\mathbf{m}}$ by taking the latent vector \mathbf{z} as input. The discriminator enforces adversarial training, ensuring that the encoded latent vector distribution $\mathbf{z} \sim \mathbf{q}(\mathbf{z})$ approximates a prior Gaussian distribution $\mathbf{z} \sim \mathbf{p}(\mathbf{z})$. It receives input from the latent vectors generated by the encoder $\mathbf{z} \sim \mathbf{q}(\mathbf{z})$ or from the prior distribution $\mathbf{z} \sim \mathbf{p}(\mathbf{z})$, and it discriminates which distribution the input latent vector originates from.

This adversarial framework enhances the generative capability and ensures smooth transitions between different field realizations. In the adversarial autoencoder method, the encoder $\mathcal{G}(\cdot)$ (which also acts as the generator of the adversarial network), decoder, and discriminator $D(\cdot)$ are trained jointly in two phases during each iteration: the reconstruction phase and the regularization phase.

In the reconstruction phase, the encoder and decoder are updated using the following loss function:

$$\mathcal{L}_{ED} = \frac{1}{N} \sum_{i=1}^N \|\mathbf{m}_i - \tilde{\mathbf{m}}_i\|_1 - w_{\text{adv}} \left(\frac{1}{N} \sum_{i=1}^N \log\{D[\mathcal{G}(\mathbf{m}_i)]\} \right), \quad (16)$$

where w_{adv} is a weight balancing the reconstruction and adversarial losses (set to 0.01 in this study), $\tilde{\mathbf{m}}_i$ is the reconstructed sample of \mathbf{m}_i , and N is the number of training samples.

In the regularization phase, the discriminator is trained to distinguish real latent vectors from the prior distribution based on the following loss function:

$$\mathcal{L}_D = -\frac{1}{N} \sum_{i=1}^N \{\log[D(\mathbf{z}_i)] + \log[1 - D[\mathcal{G}(\mathbf{m}_i)]]\}. \quad (17)$$

This loss function helps the discriminator to distinguish between the latent vector \mathbf{z}_i (from the true distribution $p(\mathbf{z})$) and the fake latent vector produced by the encoder $\mathcal{G}(\mathbf{m}_i)$.

The constraint loss functions in the adversarial auto-encoder framework ensure that the reconstructed high-dimensional parameter field $\tilde{\mathbf{m}}$ closely matches the original field \mathbf{m} , while also making sure that the distribution of the low-dimensional latent vector \mathbf{z} approximates a pre-defined standard normal distribution $p(\mathbf{z})$. After finishing the training process, it is possible to sample from the low-dimensional space of $p(\mathbf{z})$ and to use the decoder to generate corresponding high-dimensional parameter fields. Then, the high-dimensional parameter field can be reconstructed by indirectly estimating the low-dimensional latent vectors (Makhzani et al., 2015; Mo et al., 2020).

2.3 Optimization algorithms

2.3.1 Metaheuristic algorithms

The four metaheuristic algorithms used in this paper essentially update model parameters through distinct heuristic stochastic search strategies. Specifically, particle swarm optimization (PSO) updates the model parameters \mathbf{m} based on the personal best position of the particles and the global best position of the swarm (Eberhart and Kennedy, 1995). Genetic algorithm (GA) encodes the initial model parameter samples using binary encoding, then iteratively updates them through crossover (combining portions of encoded solutions to generate new candidate solutions), mutation (randomly altering encoded information to introduce diversity), and selection (choosing candidate solutions based on objective function evaluations) (Holland John, 1975). Differential evolution (DE) initializes a population of real-valued parameter vectors and iteratively updates them through differential mutation (generating trial solutions based on vector differences among population members), crossover (probabilisti-

cally combining components from original and mutated vectors), and greedy selection (retaining solutions with better objective function values) (Storn and Price, 1997; Tran et al., 2022). Simulated annealing (SA) starts from a random initial solution and iteratively explores neighboring solutions, accepting them probabilistically based on the Metropolis criterion, while gradually decreasing the temperature parameter until convergence (Metropolis et al., 1953; Kirkpatrick et al., 1983).

A common characteristic of all the methods described above is that each iterative update of model parameters requires multiple evaluations of the objective function, and sufficient iterations are necessary to balance local exploitation and global exploration. Detailed implementation procedures and theoretical foundations of these methods are provided in the supplementary materials. The metaheuristic algorithms used in this study were implemented using the open-source Python package scikit-opt (<https://scikit-opt.github.io/>, last access: 10 September 2024).

2.3.2 TNNA algorithm

The TNNA algorithm aims to obtain a reverse network $\mathbf{F}_{\text{Reverse}}(\cdot)$ that maps the observation vector to model parameters, as shown in Eq. (18).

$$\mathbf{m} = \mathbf{F}_{\text{Reverse}}(\mathbf{y}_{\text{obs}}, \boldsymbol{\theta}_{\text{Reverse}}), \quad (18)$$

where $\boldsymbol{\theta}_{\text{Reverse}}$ is the trainable parameters of $\mathbf{F}_{\text{Reverse}}$. Since \mathbf{m} also serves as the input to the established surrogate model $\mathbf{F}_{\text{Forward}}(\cdot)$, by substituting the parameter \mathbf{m} in the inversion objective function of Eq. (2) with the expression from Eq. (18), we obtain the objective function constraint for $\boldsymbol{\theta}_{\text{Reverse}}$ (i.e., the loss function for training $\mathbf{F}_{\text{Reverse}}$):

$$\boldsymbol{\theta}_{\text{Reverse}}^* = \underset{\boldsymbol{\theta}_{\text{Reverse}}}{\text{argmin}} \sum_{i=1}^{N_{\text{obs}}} \frac{1}{\sigma_i} [\mathbf{y}_{\text{obs}}[i] - \mathbf{F}_{\text{Forward}}(\mathbf{F}_{\text{Reverse}}(\mathbf{y}_{\text{obs}}, \boldsymbol{\theta}_{\text{Reverse}}))[i]]^2. \quad (19)$$

After obtaining the optimal trainable parameters $\boldsymbol{\theta}_{\text{Reverse}}^*$ through backpropagation-based stochastic gradient descent within the PyTorch framework, the final inversion results for the model parameters can be computed by $\mathbf{m}^* = \mathbf{F}_{\text{Reverse}}(\mathbf{y}_{\text{obs}}, \boldsymbol{\theta}_{\text{Reverse}}^*)$. The required training data here are the normalized observation data. Specifically, the reverse network for this study is designed using an FC-DNN with three hidden layers, each containing 512 neurons.

During the reverse network training processes, each iteration of updating the trainable parameters $\boldsymbol{\theta}_{\text{Reverse}}$ involves two main steps: first, the observation vector \mathbf{y}_{obs} is input into the reverse network $\mathbf{F}_{\text{Reverse}}$ to obtain the parameter prediction vector $\tilde{\mathbf{m}}$. Next, the predicted parameter $\tilde{\mathbf{m}}$ is input into the forward network $\mathbf{F}_{\text{Forward}}$ to generate corresponding forward prediction results. Subsequently, the trainable parameters $\boldsymbol{\theta}_{\text{Reverse}}$ of the reverse network are updated through standard DNN model training based on the error feedback from

the loss function in Eq. (19). This process demonstrates that $\mathbf{F}_{\text{Reverse}}$ and $\mathbf{F}_{\text{Forward}}$ are integrated through a tandem connection, which is why this method is named TNNA. Upon completing the training of $\mathbf{F}_{\text{Reverse}}$, the final optimal parameters are predicted by inputting observation data into $\mathbf{F}_{\text{Reverse}}$. Further details on TNNA can be found in Chen et al. (2021).

In the above process, each backpropagation step involves only a single forward calculation of the loss function. After establishing the computational graph, gradients of the trainable parameters θ_{Reverse} are computed through backpropagation combined with automatic differentiation. These gradients are then used to update the trainable parameters θ_{Reverse} . Thus, only one forward simulation is executed during each epoch of the reverse network $\mathbf{F}_{\text{Reverse}}$ training procedure. This presents a marked computational advantage of TNNA compared to the four selected metaheuristic algorithms, which require numerous forward simulations for parameter updates at each iteration.

3 Case study

This study considers three synthetic cases based on previous research, covering different model sizes and hydraulic gradient combinations (Jose et al., 2004; Zhang et al., 2018; Mo et al., 2019) to evaluate the performance of the TNNA algorithm against conventional metaheuristic algorithms. Both case 1 and case 2 are approximately tens of meters in size, with a simulation time of 60 d. Their hydraulic gradients are 0.05 and 0.1, respectively. These scenarios are typically found in large sand tank experiments, aquifers with natural slopes, or in-situ experimental areas where flow conditions are enhanced through pumping wells. Case 3 simulates contaminant plume migration, with a size of approximately 1 km and a simulation time of several years (up to 30 years). It uses a hydraulic gradient of 0.00625, representing a smaller natural gradient typically found in alluvial aquifers. Regarding the differences in heterogeneity conditions among these cases, case 1 features a low-dimensional zoned permeability field scenario, case 2 involves a high-dimensional Gaussian random permeability field parameterized through the Karhunen–Loève expansion (KLE), and case 3 uses a high-dimensional non-Gaussian binary random permeability field parameterized by a decoder trained with OCAAE. The numerical models of the three cases are established using TOUGHREACT, which employs an integral finite difference method with sequential iteration procedures and adaptive time stepping to solve the flow and transport equations. In all the three cases, the relative error tolerance for the conservation equations was uniformly set to 10^{-5} , ensuring that the maximum imbalance of conserved quantities within each discrete grid cell remains below 1 part in 100 000 of the total quantity in that cell. Dispersion effects are inherently incorporated through molecular diffusion and numeri-

cal dispersion induced by upstream weighting and grid discretization (Xu et al., 2011).

After developing numerical models for the three scenarios, we first evaluate four surrogate models in case 1, and the optimal surrogate model will be integrated into the inversion framework. Subsequently, hypothetical observation scenarios are used to systematically compare the inversion accuracy of TNNA against four metaheuristic algorithms across the three cases. The observation data (hydraulic heads and solute concentrations) for the model parameter inversion are generated by adding Gaussian noise perturbations to the numerical model simulation results. Specifically, observational noise is introduced by multiplying the min–max normalized simulated data by the random noise factor $\epsilon \sim N(1, \sigma^2)$, where σ represents the ratio of observational noise to the observed values. In this study, we conduct a comparative analysis of inversion performance across the three cases under a noise level of $\sigma = 0.01$. Additionally, our previous study (Chen et al., 2021) examined the effects of higher observational noise levels ($\sigma = 0.05$ and 0.1) and real-world noise conditions on inversion accuracy in low-dimensional parameter scenarios. To further investigate the impact of increased observational noise on inversion performance in high-dimensional parameter scenarios, we conducted an extended analysis on case 3 – the most complex scenario – by increasing the noise level to 10 % ($\sigma = 0.1$). This analysis also provides insights into the stability of the TNNA algorithm when integrated with a generative machine-learning-based inversion framework for high-dimensional parameter estimation. Here, we applied the multiplicative noise to ensure that all perturbed observation values remain non-negative, which is particularly important in regions near plume boundaries where concentrations are close to zero. Generally, observation errors are assumed to be independent of the measured values, whereas the multiplicative noise model introduces value-proportional perturbations, resulting in a positive correlation between the standard deviation of observation noise and the true values. This type of error dependence may also exist in real-world studies when certain measurement techniques are used. For example, in hydraulic head monitoring, pressure transducers may exhibit drift (i.e., a persistent deviation in output not caused by actual pressure changes) due to the aging and fatigue of components such as the diaphragm or strain gauge, leading to reduced measurement accuracy (Sorensen and Butcher, 2011). A variation in hydraulic pressure can lead to different levels of drift among transducers, with those installed at higher pressure (i.e., higher hydraulic head) environments tending to experience more significant drift, which in some cases may result in elevated observation noise. For the analysis of solute concentrations in laboratory settings, when the concentrations of water samples exceed the detection range of the instrument, a common approach is to dilute these samples prior to measurement. While analytical instruments may introduce additive errors at a relatively fixed level, the rescaling process fol-

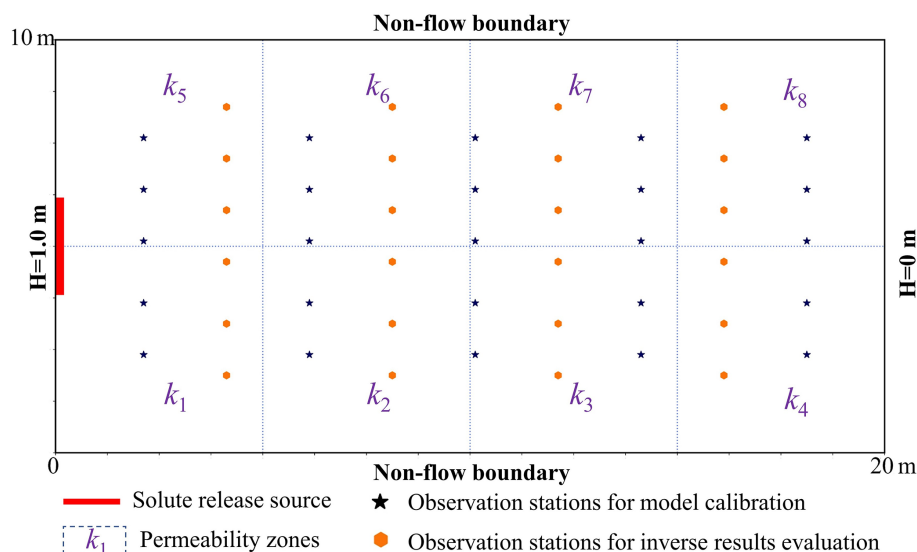


Figure 2. Flow domain of the solute transport model for the low-dimensional scenario.

lowing dilution (i.e., multiplying the measured value by the dilution factor) amplifies these errors. As a result, the final measurement error becomes approximately proportional to the original solute concentration (Kabala and Skaggs, 1998). Given that the goal of this study is to evaluate the robustness of five inversion algorithms under different noise levels, both additive and multiplicative noise models are suitable for representing observational uncertainty. Prior work by Neupauer et al. (2000) demonstrated that the choice between these two noise types has minimal influence on the comparative performance of inversion methods. The details of these three cases are provided in Sects. 3.1–3.3.

3.1 Case 1: low-dimensional zoned permeability field scenario

As shown in Fig. 2, the numerical model for the low-dimensional scenario focuses on conservative solute transport in a zoned permeability field. The model domain is a two-dimensional rectangular area measuring 10 m × 20 m. The left and right boundaries feature Dirichlet boundary conditions, with a hydraulic head difference of 1 m. The heterogeneous permeability is divided into eight homogeneous permeability zones, denoted as k_1 to k_8 . The prior range for these eight permeabilities is from 1×10^{-12} to $9.9 \times 10^{-12} \text{ m}^2$. The contaminant source is located at the left boundary, with a fixed release concentration ranging from 1×10^{-3} to 1 mol L^{-1} . The simulation area is uniformly discretized into 3200 (40×80) cells, and the simulation time is set to 20 d.

According to these model conditions, there are nine model parameters to be estimated: eight permeability parameters (k_1 to k_8) and the source release concentration. As shown in Fig. 2, these parameters will be estimated using the ob-

servation data of hydraulic heads and solute concentrations collected from 25 locations, denoted by black stars. Additionally, observation data from another 24 locations, denoted by orange hexagons and not included in the calibration process, will be used to evaluate the prediction accuracy of the calibrated numerical model.

3.2 Case 2: high-dimensional Gaussian random permeability field scenario

The numerical model for the high-dimensional scenario features a domain size of 10 m × 10 m, with impervious upper and lower boundaries and constant head boundaries on the left (1 m) and right (0 m) sides. The domain is discretized into 4096 (64×64) cells. The log-permeability field follows a Gaussian distribution, and the permeability value of the i th mesh is defined as follows:

$$k_i = \alpha_i k_{\text{ref}}, \quad (20)$$

where k_{ref} is the reference permeability, set to $2 \times 10^{-13} \text{ m}^2$. The heterogeneity of k_i is controlled by the modifier α_i . The geostatistical parameters for this Gaussian field are $m = 0$, $\sigma_G^2 = 2$, and $\lambda_x = \lambda_y = 2.5 \text{ m}$. Under this heterogeneous condition, 100 KLE terms are used to preserve more than 92.67 % of the field variance. Consequently, estimating the permeability field is equivalent to identifying these 100 KLE terms.

The observational data used for inverse modeling include hydraulic heads from a stationary flow field and solute concentrations measured every 2 d over 40 d, starting from the second day to the 40th day (day: $t = 2i$, $i = 1, \dots, 20$). It should be noted that in high-dimensional parameter scenarios, the increased degrees of freedom typically result in greater parameter uncertainty. Insufficient observational in-

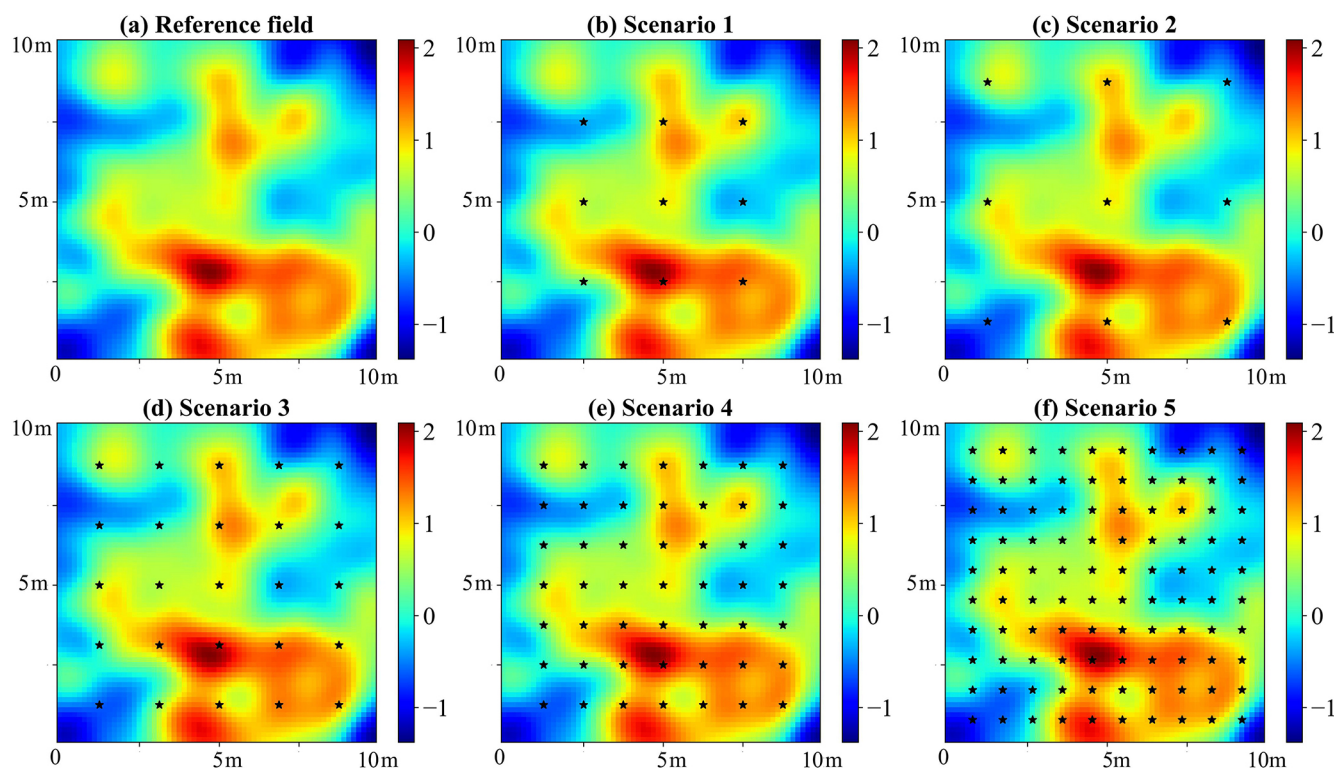


Figure 3. The reference log-permeability field and locations of observation stations for five scenarios. The observation stations are represented by black stars.

formation may fail to effectively constrain parameter estimation, resulting in potential uncertainty and equifinality (Beven and Binley, 1992; McLaughlin and Townley, 1996; Zhang et al., 2015; Cao et al., 2025). Therefore, this study includes actual permeability values at observed locations as regularization constraints to mitigate inversion errors arising from equifinality. Since identical regularization conditions are uniformly applied across all algorithms, introducing these constraints ensures the stability and robustness of the inversion outcomes without affecting the inherent performance characteristics of the five optimization algorithms compared in this study.

As the degrees of freedom significantly increase in high-dimensional models, the influence of observation data on inversion results becomes increasingly significant. Five scenarios with different monitoring networks are considered to comprehensively evaluate the performance of different inversion algorithms using various observations. Figure 3 displays the monitoring station locations for each scenario.

3.3 Case 3: high-dimensional non-Gaussian random permeability field scenario

This case focuses on an estimation of a binary non-Gaussian permeability field. The numerical model features a domain size of $800\text{m} \times 800\text{m}$, with impervious upper and lower boundaries and constant head boundaries on the left (5 m)

and right (0 m) sides. The domain is discretized into 6400 (80×80) cells. The permeability field is a channelized random field composed of two lithofacies, with permeability values of 1.0×10^{-13} and $5.46 \times 10^{-12} \text{ m}^2$ for the two media, respectively. The reference field (Fig. 4b) is generated from a training image (Fig. 4a) using the direct sampling (DS) method proposed by Mariethoz et al. (2010). The contaminant release source is located on the entire left boundary, with a concentration of 1 mol L^{-1} . The observational data used for inversion are generated through numerical simulation, including steady-state hydraulic head data and solute concentration data at 12 time points (from 2 to 24 years, with 2-year intervals). This case focuses on a high-dimensional binary inverse problem aimed at identifying the lithofacies type of each discrete grid cell within the domain. Note that the permeability values of the two lithofacies in this case are fixed.

To achieve a low-dimensional representation of permeability fields, a training dataset comprising 2000 stochastic realizations is generated using multi-point statistics (MPS). Then, an octave convolution adversarial autoencoder (OCAAE) is developed, where the decoder network learns non-linear mapping from 100-dimensional Gaussian latent vectors to 6400-dimensional binary non-Gaussian permeability fields. Thus, the non-Gaussian permeability field is

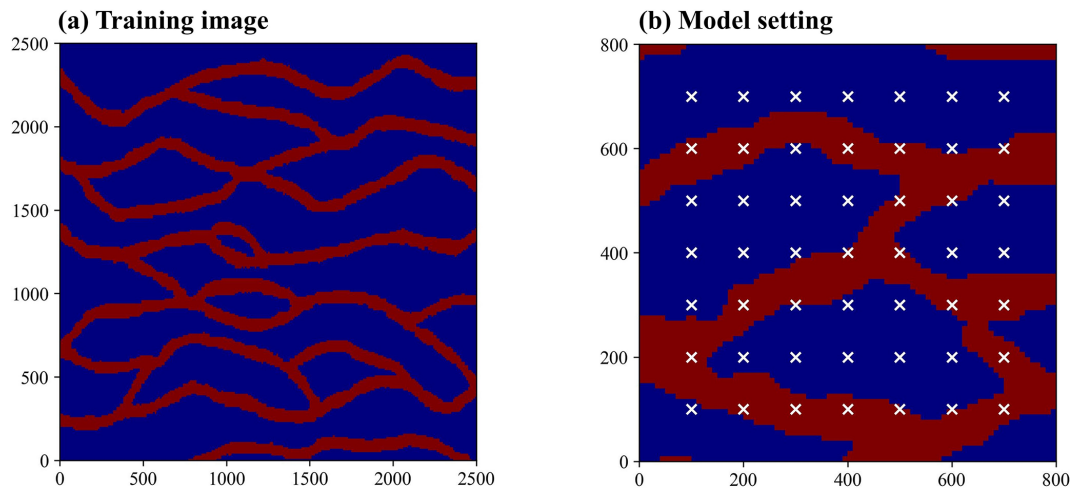


Figure 4. (a) The training image used to generate random realizations of the permeability field. (b) The reference field of the synthetic case (white symbols indicate observation locations).

indirectly reconstructed by estimating the 100-dimensional latent vector.

4 Results and discussion

4.1 Surrogate model evaluations

Surrogate models were first compared using case 1, with a low-dimensional parameter. For this scenario, the input parameters for the surrogate models consist of a nine-dimensional vector, including eight permeability parameters and the contaminant source release concentration. The output consists of the simulated hydraulic heads and solute concentrations at 25 observation points. Four training datasets $\mathbf{D}_{\text{train}} = \{\mathbf{M}_{\text{train}}, \mathbf{Y}_{\text{train}}\}$ with 200, 500, 1000, and 2000 samples (represented as $\mathbf{D}_{\text{train-200}}$, $\mathbf{D}_{\text{train-500}}$, $\mathbf{D}_{\text{train-1000}}$, and $\mathbf{D}_{\text{train-2000}}$, respectively) and a testing dataset $\mathbf{D}_{\text{test}} = \{\mathbf{M}_{\text{test}}, \mathbf{Y}_{\text{test}}\}$ with 100 samples (represented as $\mathbf{D}_{\text{test-100}}$) are prepared. These datasets were generated using Latin hypercube sampling (LHS) and numerical simulations. The predictive accuracy of the surrogate models was quantitatively evaluated using root mean square error (RMSE) and determination coefficient (R^2) metrics (Chen et al., 2022).

For solute transport inverse modeling problems, it is crucial to consider the observations of both hydraulic heads and solute concentrations simultaneously. Therefore, the surrogate model within an inversion framework should have accurate predictive capabilities for hydraulic heads and solute concentrations. This study calculates the RMSE and R^2 values separately for hydraulic heads, solute concentrations, and all model response data, resulting in the following evaluation criteria: RMSE_{ALL} and R^2_{ALL} for overall data, RMSE_H and R^2_H for hydraulic heads, and RMSE_C and R^2_C for solute concentrations. Additionally, it should be noted that the above

RMSE and R^2 metrics are computed based on the normalized hydraulic head and solute concentration data.

Figures 5 and 6 display the RMSE and R^2 values of each surrogate model, and Figs. S3–S6 in the Supplement present the pairwise comparison results. The optimal values for C , σ , and ε in the MSVR method are provided in Table S1 in the Supplement. For the FC-DNN, the optimal number of hidden layers was separately determined for each of the four datasets. The candidate range for the number was set from 1 to 7. According to the RMSE_{ALL} and R^2_{ALL} values in Tables S2 and S3 in the Supplement, the optimal number of hidden layers in the FC-DNN for $\mathbf{D}_{\text{train-200}}$, $\mathbf{D}_{\text{train-500}}$, $\mathbf{D}_{\text{train-1000}}$, and $\mathbf{D}_{\text{train-2000}}$ are two, four, three, and three, respectively. When training the FC-DNN, LeNet, and ResNet for case 1, the hyperparameters for batch size and learning rate were consistently set to 50 and 1×10^{-4} , respectively. The weight decay values for LeNet and ResNet were both set to 1×10^{-5} , while FC-DNN used a weight decay of 0. The number of training epochs was uniformly set to 500 for all three models.

According to the performance criteria in Figs. 5 and 6, the prediction accuracy of each surrogate model significantly improves with an increasing number of training samples. Based on the RMSE_{ALL} and R^2_{ALL} values, their performance ranks as follows: ResNet, LeNet, FC-DNN, and MSVR. The MSVR method accurately predicts hydraulic heads but performs the worst in predicting solute concentration. Training MSVR with the four prepared datasets, the RMSE_H values are below 0.02, and the R^2_H values are near 1. Notably, with a training sample size of 200, the prediction accuracy of MSVR for hydraulic heads is higher than that of FC-DNN and LeNet, as indicated by their RMSE_H and R^2_H values, closely matching that of ResNet. However, when using 200 training samples, the RMSE_C value for MSVR exceeds 0.08, and the R^2_C value falls below 0.85. Even with a dataset size of 2000, the enhancement in the MSVR-based surrogate model is limited, as

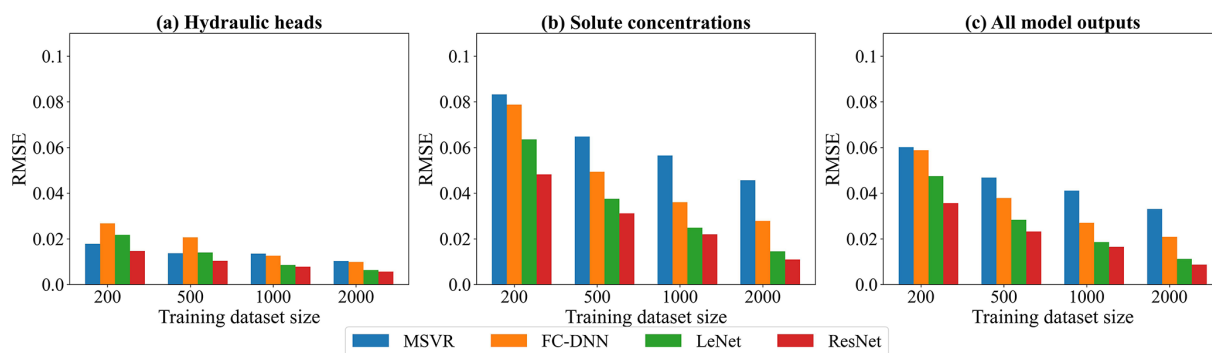


Figure 5. The RMSE results of surrogate model predictions. Panels (a)–(c) show respectively the RMSE values of hydraulic heads, solute concentrations, and all model outputs.

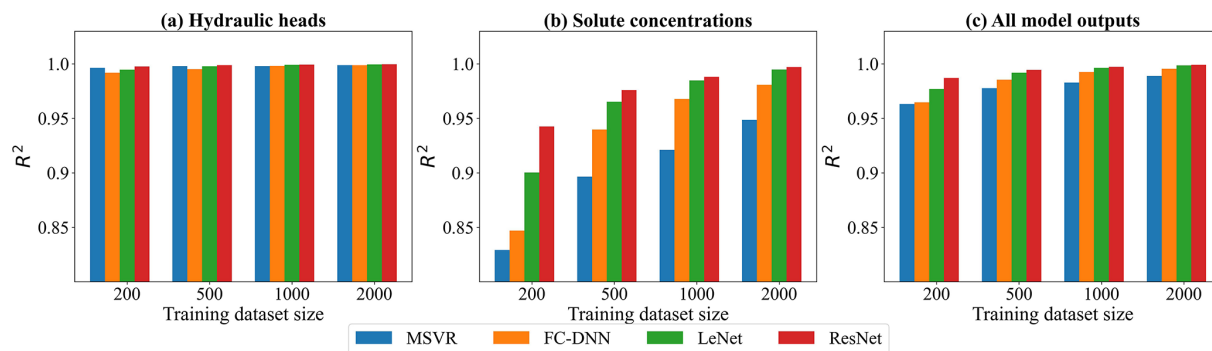


Figure 6. The R^2 results of surrogate model predictions. Panels (a)–(c) show respectively the R^2 values of hydraulic heads, solute concentrations, and all model outputs.

the RMSE_C value remains at around 0.05, and the R_C^2 value stays below 0.95. FC-DNN demonstrates a significant advantage over MSVR in predicting solute concentration, particularly with larger training sample sizes of 1000 or 2000. However, there are still some obvious biases between some surrogate modeling results and their numerical modeling results (see Fig. S2d). When adopting CNN-based surrogate models (LeNet and ResNet), the prediction accuracy for solute concentrations significantly improves (see Figs. 5b and 6b). With training datasets of 2000 samples, LeNet and ResNet achieve RMSE_C values below 0.02 and R_C^2 values close to 1. It is worth noting that ResNet performs well even with smaller sample sizes. For example, with 200 training samples, the RMSE_C and R_C^2 values for LeNet are around 0.06 and 0.9, respectively, while these criteria values for ResNet are around 0.04 and 0.95 (see Figs. 5b and 6b). As the number of training samples increases, the advantages of ResNet become more apparent. According to Fig. S4d, when the training sample size reaches 2000, the prediction results of ResNet are closely consistent with the numerical simulation results for both hydraulic heads and solute concentrations.

The comparison results of the surrogate models reflect a trend of enhanced robustness attributable to advancements in machine learning methodologies. Different machine learn-

ing approaches employ distinct strategies for achieving non-linear mappings in developing surrogate models. Generally, deeper or larger models contain more trainable parameters, resulting in higher degrees of freedom to capture more robust non-linear relationships. The essence of machine learning development lies in addressing the challenge of training these complex DNNs. Current state-of-the-art machine learning techniques have demonstrated proficiency in training each of the four selected surrogate modeling methods. With sufficient training samples, a surrogate model of greater complexity exhibits enhanced capability in representing higher levels of non-linearity (LeCun et al., 2015; He et al., 2016). This also explains why, despite having a sufficient number of training samples, the improvement in prediction accuracy of the MSVR for solute concentration is limited. In CNNs, sparse connections and weight sharing in convolutional layers reduce redundant weight parameters in DNNs, enhancing the feature extraction of hidden layers. Consequently, LeNet demonstrates better performance than FC-DNN. ResNet, which employs residual blocks in conjunction with convolutional layers, effectively addresses the issues of gradient vanishing and exploding, making the successful training of deeper CNNs possible.

According to Chen et al. (2021), a more globally accurate surrogate model can enhance the performance of TNNA inversion results. Thus, we selected the ResNet trained with 2000 samples for the subsequent inversion procedure. In the low-dimensional scenario, its RMSE values for hydraulic head and solute concentration data are less than 0.02, with R^2 values greater than 0.99. We further extended ResNet for the surrogate model construction of both Gaussian and non-Gaussian random field scenarios. In the two high-dimensional scenarios, the input parameters for the surrogate models are single-channel matrix data representing the heterogeneous parameter field, while the output consists of a vector formed by flattening the multi-channel matrix data, representing the simulated hydraulic heads and solute concentrations at predefined time steps within the simulation domain. The training and testing datasets for these two case scenarios consist of 2000 and 500 samples, respectively. For ResNet training in case 2 (Gaussian random field), the hyperparameters were set as follows: batch size = 100, learning rate = 1×10^{-4} , and weight decay = 1×10^{-6} . For case 3 (non-Gaussian random field), the corresponding values were batch size = 50, learning rate = 1×10^{-3} , and weight decay = 1×10^{-8} . In both cases, the number of training epochs was also set to 500. The RMSE values for hydraulic head and solute concentration data range from approximately 0.01 to 0.03, and the R^2 values exceed 0.99 (as shown in Table 1). This level of accuracy indicates that the surrogate model meets the predictive accuracy requirements for inversion simulations in both of the designed Gaussian and non-Gaussian random field cases.

4.2 Parameter inversion method comparison results

4.2.1 Inversion results of the low-dimensional parameter scenario

For the low-dimensional parameter scenario, the performance of optimization algorithms is thoroughly evaluated across 100 parameter scenarios using the Monte Carlo strategy. The observation data for these scenarios are derived from the testing dataset after adding multiplicative Gaussian random noise $\epsilon \sim N(1, 0.01^2)$. The population sizes of GA, DE, and PSO, along with the chain length in SA, are set in four distinct scenarios: 20, 40, 60, and 80. (These population size or chain length values are represented as N_{PC} in subsequent discussions.) These settings determine the number of forward modeling calls required for each iteration, significantly influencing the convergence rate and computational efficiency of optimization procedures. Maximum iterations for these four metaheuristic algorithms are set to 200. The learning rate, epoch number, and weight decay for the TNNA algorithm are set to (6×10^{-5}), 1000, and 1×10^{-6} , respectively.

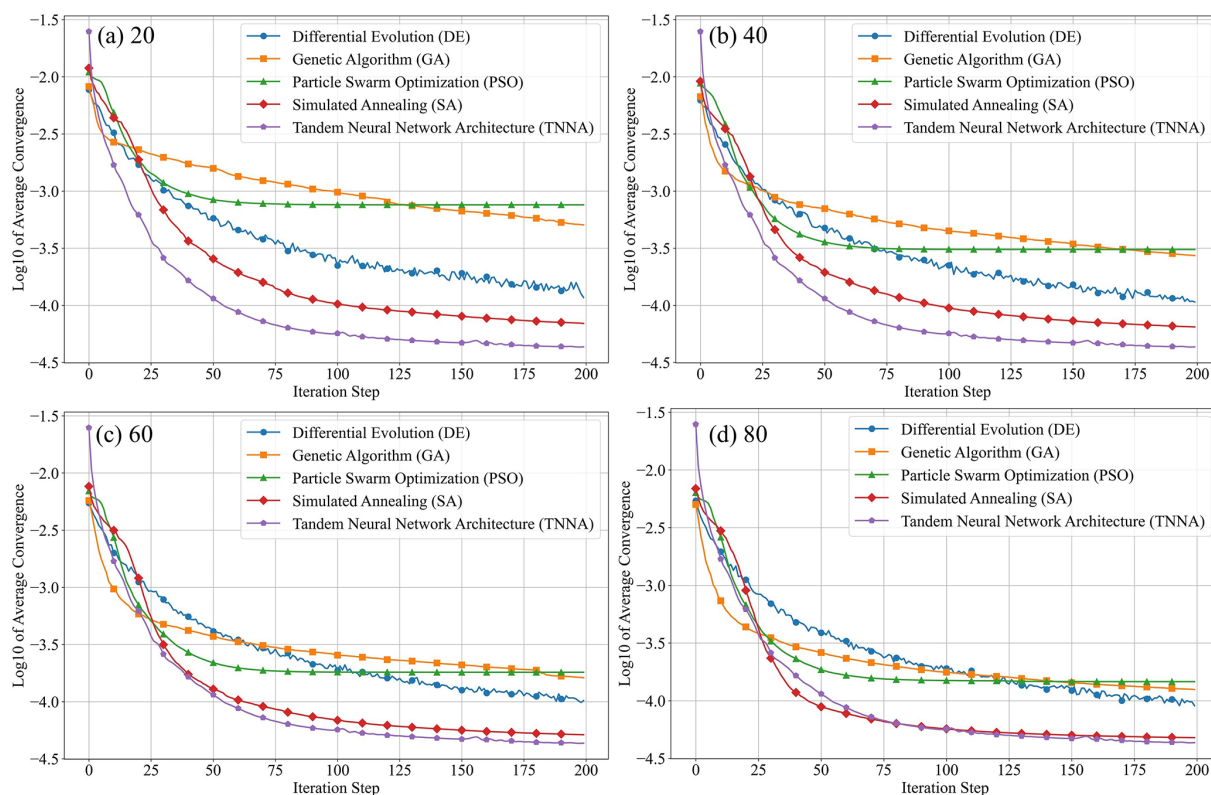
The performance of the five optimization algorithms is evaluated according to three aspects: average convergence

efficiency and accuracy in inversion procedures, predictive accuracy of calibration models for hydraulic heads and solute concentrations, and statistical analysis of the estimated errors for each model parameter. Figure 7 presents the logarithmic average convergence curves (i.e., \log_{10} of the average objective value as a function of the inversion iterations) of four metaheuristic algorithms and the TNNA algorithm throughout 100 parameter scenarios. Specifically, panels (a)–(d) represent the N_{PC} values for metaheuristic algorithms set at 20, 40, 60, and 80, respectively. These figures clearly illustrate the average convergence speed and accuracy of five optimization algorithms. Figure 8 displays the comparison between simulated and observed values across all 100 parameter scenarios for both calibration and spatial predictive evaluation. Panels (a) and (b) illustrate the comparative prediction fit at the 25 observation locations used for model calibration, whereas panels (c) and (d) display the comparative prediction fit at the 24 independent observation locations. In this figure, distinct symbols are used to represent the five optimization algorithms. It should be noted that the N_{PC} values for the four metaheuristic algorithms are uniformly set to 80 during this comparison. Figure 9 illustrates the probability density curves of the estimation errors for nine model parameters across 100 parameter scenarios, with different colors representing the five optimization algorithms.

The results in Fig. 7 demonstrate that the TNNA algorithm achieves the best convergence accuracy, with its convergence logarithmic objective function value (approximately -4.4) being smaller than those of the other four metaheuristic algorithms across these N_{PC} settings. The influence of N_{PC} on the convergence speeds of these four metaheuristic algorithms is not significant, exhibiting a distinct transition from rapid to slower convergence around the 75th iteration. As N_{PC} increased from 20 to 80, each metaheuristic algorithm showed distinct improvements in the accuracy of the final objective function. The DE algorithm showed the least improvement in final convergence accuracy as the N_{PC} value increased from 20 to 80, with the logarithmic value of its objective function dropping from just above -4.0 to slightly below -4.0 . The SA algorithm also showed limited improvement, with its logarithmic average convergence value increasing from around -4.1 at $N_{PC} = 20$ to slightly below -4.3 at $N_{PC} = 80$, close to that of the TNNA algorithm. Among the four metaheuristic algorithms, SA exhibited the highest average convergence accuracy. Contrary to the SA and DE algorithms, the PSO and GA algorithms significantly enhanced average convergence accuracy as N_{PC} increased. Specifically, as N_{PC} increased from 20 to 80, the logarithmic convergence values of PSO and GA decreased by more than 0.5. While increasing N_{PC} values may help metaheuristic algorithms to reduce the gap in average convergence accuracy compared to the TNNA algorithm, larger N_{PC} settings also require additional computational burdens. The above results indicate that the TNNA algorithm has a significant efficiency advantage over the four metaheuristic algorithms in param-

Table 1. The RMSE and R^2 values for surrogate model predictions in five designed high-dimensional scenarios.

	RMSE			R^2		
	RMSE _H	RMSE _C	RMSE _{All}	R^2_{H}	R^2_{C}	R^2_{All}
Gaussian scenario 1	0.0108	0.0174	0.0172	0.9990	0.9980	0.9982
Gaussian scenario 2	0.0102	0.0138	0.0136	0.9995	0.9989	0.9990
Gaussian scenario 3	0.0120	0.0165	0.0163	0.9991	0.9981	0.9983
Gaussian scenario 4	0.0123	0.0161	0.0159	0.9990	0.9984	0.9985
Gaussian scenario 5	0.0137	0.0156	0.0155	0.9989	0.9985	0.9986
Non-Gaussian scenario	0.0181	0.0280	0.0273	0.9952	0.9931	0.9932

**Figure 7.** Comparative convergence trends (\log_{10} of the average objective value) of five optimization algorithms across 100 parameter scenarios. Panels (a)–(d) compare the four metaheuristic algorithms and TNNA under $N_{\text{PC}} = 20, 40, 60$, and 80 , respectively; TNNA was executed only once on the same 100 parameter scenarios, and its curve is identical across all panels. Markers indicate convergence values every 10 iterations.

eter optimization. For instance, when conducting the optimization procedure based on scikit-opt, the DE algorithm requires 32 000 forward model realizations ($80 \times 2 \times 200$) when N_{PC} is set to 80, while the other three metaheuristic algorithms (PSO, GA, and SA) each require 16 000 realizations (80×200). In significant contrast, the TNNA algorithm requires only one forward model realization per iteration, resulting in 200 realizations. These comparisons illustrated that the TNNA method is more effective than the other four metaheuristic algorithms in achieving robust convergence results. It is worth noting that the five optimization al-

gorithms rely on stochastic processes for parameter updates. Therefore, the objective function values are not guaranteed to decrease monotonically with each iteration. According to Fig. 7, the DE algorithm exhibits more noticeable fluctuations compared to other algorithms. Nevertheless, these fluctuations remain within a reasonable range. For example, at $N_{\text{PC}} = 80$, the objective function values after 150 iterations range between 9.05×10^{-5} and 1.32×10^{-4} (corresponding to the logarithmic values between -4.04 and -3.88 in Fig. 7d). Fluctuations between consecutive iterations typically remain

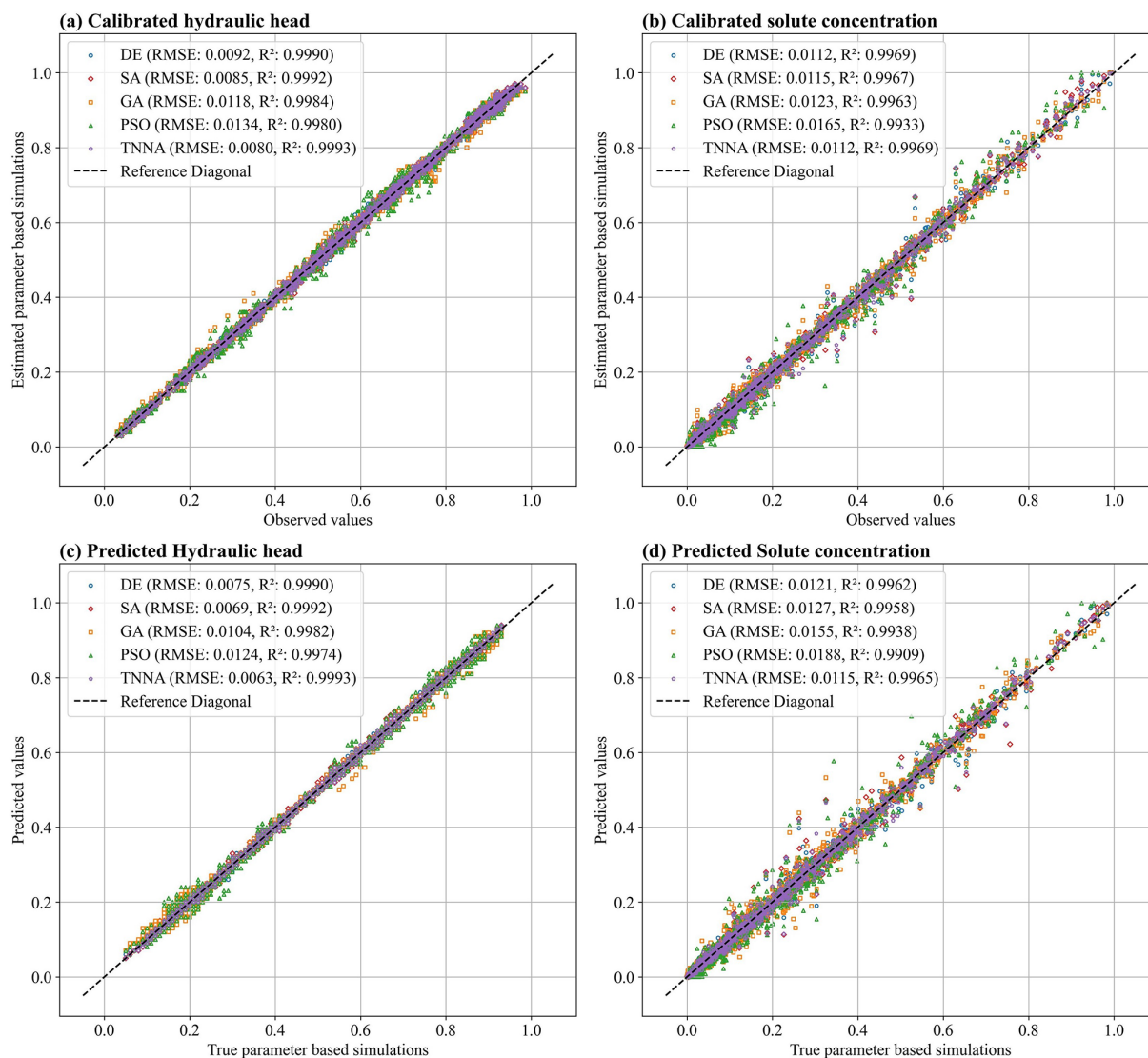


Figure 8. Comparison of predictive accuracy for hydraulic heads and solute concentrations simulated using parameters estimated by the four metaheuristic inversion algorithms (DE, SA, GA, PSO) and the TNNA method. Panels (a) and (b) show predictive comparisons at the 25 observation locations used for model calibration and panels (c) and (d) show predictive comparisons at the other 24 independent observation locations.

within 1×10^{-5} (mostly around 3×10^{-6}), which is considered reasonable for optimization algorithms.

The results presented in Fig. 8 indicate that, among the five optimization algorithms, the TNNA algorithm achieves the smallest RMSE and R^2 values closest to 1.0 for both hydraulic heads and solute concentration during model calibration and spatial predictive evaluation. Furthermore, the distribution of comparison points demonstrates that the modeling results obtained from both calibration and independent prediction using the TNNA algorithm match the observed values more accurately than those of the other four metaheuristic algorithms, particularly for solute concentrations. Among the four metaheuristic algorithms, SA and DE outperform GA and PSO regarding RMSE and R^2 values. During model cal-

ibration and predictive evaluation, PSO exhibits the worst predictive accuracy, recording the highest RMSE and R^2 values for both hydraulic heads and solute concentrations. It is noteworthy that the RMSE and R^2 values for SA during hydraulic head calibration are 0.0085 and 0.9992, respectively, while those for DE during solute concentration calibration are 0.0112 and 0.9969. These values are almost equal to those of the TNNA algorithm. The robustness of an inversion algorithm is determined by its accuracy in both calibration and predictive evaluation for hydraulic heads and solute concentrations. However, DE and SA demonstrate appropriate calibration accuracy for only one of the two simulation components. Overall, the TNNA algorithm provides more robust

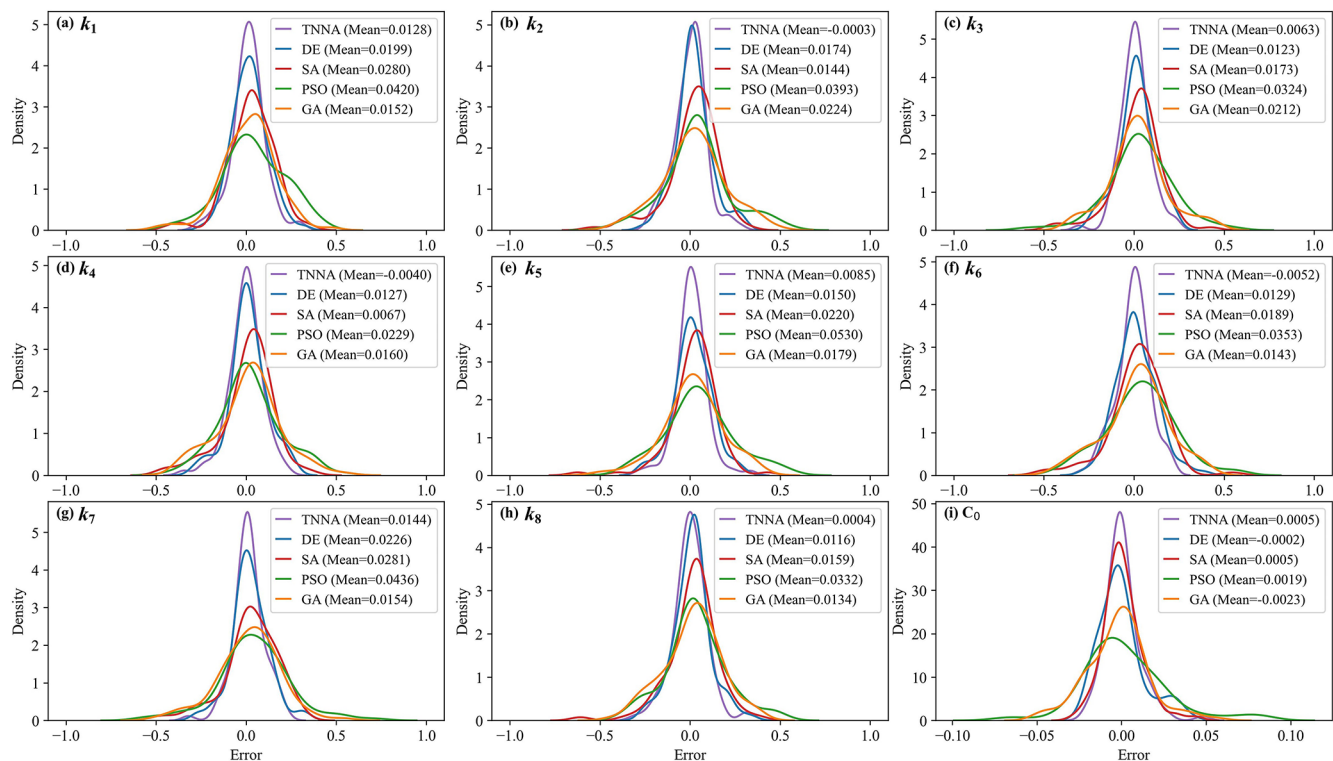


Figure 9. Probability density curves of estimation errors for nine model parameters using five optimization methods. Each curve represents the distribution of estimation errors across 100 parameter scenarios, with their mean error values indicated in the legends.

model calibration and predictive evaluation results than the other four metaheuristic algorithms.

Figure 9 indicates that the estimated error distributions for the nine model parameters derived from the TNNA algorithm are more concentrated than those obtained from the four metaheuristic algorithms. The mean estimated error values for the nine numerical model parameters using the TNNA algorithm are also the lowest. These results highlight the high accuracy and reliability of the TNNA inversion algorithm. Among the four metaheuristic algorithms, DE and SA outperform GA and PSO. This is because the probability density curves of estimation errors for the nine parameters using DE and SA are more concentrated around zero, with mean values lower than those of GA and PSO. The DE algorithm shows a more concentrated distribution of around zero for the overall estimation errors of parameters k_1 to k_8 . In contrast, the SA reveals reduced estimation errors for the C_0 parameter in most cases, ranking just behind the TNNA algorithm. GA outperforms PSO in estimation accuracy for seven of the nine model parameters, with PSO matching its probability density curves to that of GA only for parameters k_2 and k_4 . As a whole, the statistical results of the estimated model parameter errors illustrate that the machine-learning-based TNNA algorithm exhibits enhanced inversion performance compared to the four metaheuristic optimization algorithms. However, the findings also reveal that none of the five algo-

gorithms consistently offers completely reliable inversion solutions across all scenarios. For example, the TNNA algorithm, despite its generally better performance, demonstrates estimation errors as high as 0.4 for parameters k_4 and k_6 in some scenarios. Such results are likely because the provided observational data cannot ensure equifinality in some scenarios. In these cases, it is essential to introduce additional regularization constraints to attenuate the equifinality (Wang and Chen, 2013; Arsenault and Brissette, 2014). These findings emphasize the importance of employing the Monte Carlo method in comparative studies of inversion algorithms to ensure comprehensive evaluations and to avoid misleading conclusions.

The above comparison results indicate that the machine-learning-based TNNA algorithm outperforms the other four metaheuristic algorithms in both inversion accuracy and computational efficiency. The primary advantage of the TNNA algorithm over the four metaheuristic algorithms is its well-defined updating direction of model parameters, guided by the loss function, which serves as the objective function for inverse modeling. Research on machine learning applications indicates that DNNs can approximate continuous functions by adjusting weights and biases (LeCun et al., 2015; Goodfellow et al., 2016). The TNNA algorithm leverages this capability by transforming the model parameter inversion issue into the training of a reverse network to achieve reverse mappings. By establishing a loss function based on

inversion constraints from the Bayesian theorem, the TNNA algorithm ensures that training the reverse network brings each parameter update closer to the optimal solution during each epoch, thereby improving accuracy and convergence speed. In contrast, the four metaheuristic algorithms require numerous forward simulations for each parameter update. The optimization direction for model parameters is determined by evaluating the objective function. This process is governed by the exploration and exploitation strategies inherent in metaheuristic algorithms. However, these approaches introduce randomness in the direction of model parameter updates, making it challenging to ensure that updates move towards the direction of fastest convergence under specific hyperparameter settings. This also explains why the TNNA algorithm can update model parameters more efficiently and achieve higher convergence accuracy despite requiring only one forward realization in each training epoch.

4.2.2 Inversion results of the high-dimensional Gaussian scenario

For estimating the permeability field under five designed observational scenarios, the iteration number for the four metaheuristic algorithms was set at 200, with N_{PC} values of 100, 500, and 1000. The learning rate and weight decay for training reverse networks within the TNNA framework were set to 1×10^{-3} and 1×10^{-4} , respectively.

Figures 10 and 11 illustrate the log-permeability field estimation results and error distributions for the four metaheuristic algorithms and the TNNA algorithm under the most densely observed scenario (i.e., scenario 5). The corresponding results for scenarios 1–4 are presented in Figs. S7–S14 in the Supplement. Figure 12 compares the RMSE values for the log-permeability fields estimated by the four metaheuristic algorithms and the TNNA algorithm across all five scenarios. These detailed RMSE values can be found in Table 2 (scenario 5) and Table S4 in the Supplement (scenarios 1–4). For scenario 5, the accuracy of permeability estimations by each metaheuristic algorithm improves as the N_{PC} value increases (see Fig. 10 and Table 2). Notably, the GA achieves the best results with an N_{PC} of 1000, recording an RMSE of 0.1057. The DE and SA algorithms yield their most accurate permeability estimations with RMSE values of 0.1597 ($N_{PC} = 100$) and 0.1549 ($N_{PC} = 1000$), respectively. The PSO method is the least effective, achieving an RMSE of 0.3334 at $N_{PC} = 1000$. As shown in Fig. 11 and Table 2, the TNNA algorithm provides inversion results with an RMSE of 0.1063 after training the reverse network for 200 epochs. This suggests that the TNNA algorithm can estimate high-dimensional permeability fields with accuracy comparable to that of the GA method ($N_{PC} = 1000$), with significantly fewer forward model realizations (200 compared to 200 000), reducing the computational burden by 99.9 % and improving inversion efficiency by a factor of 1000. Increasing the training epochs of the reverse network to 1000 further reduces

the RMSE of the TNNA method to 0.0595, demonstrating its advantages over the four metaheuristic algorithms in this scenario. Across all scenarios, the accuracy of the estimated permeability fields correlates positively with the density of observation wells, and estimation errors are generally higher in areas not covered by monitoring wells (see Figs. S7–S14). Figure 12 further demonstrates that the RMSE values for permeability estimation using the TNNA algorithm are consistently lower than those of the four metaheuristic algorithms across scenarios 1–4, indicating that the TNNA algorithm exhibits greater robustness compared to the metaheuristic algorithms in all five scenarios.

To evaluate the predictive performance of the numerical model calibrated by various inversion methods, simulations of hydraulic heads and solute concentrations were conducted over 60 d, starting on the second day with recordings every 2 d, using the permeability fields with the lowest RMSE values identified by each inversion method. Observation data from the second day to the 40th day were used for model calibration, while additional data from the 42nd to the 60th day were employed to evaluate the future predictions of the calibrated numerical models. The RMSE values for the calibrated hydraulic heads and time series solute concentrations are presented in Table 3 and Fig. 13. Figure 14 displays the spatial distribution of the calibrated numerical simulation results and errors for hydraulic heads and solute concentration simulation results at three specific times ($t = 4$ th, 20th, and 52nd days). Results for the entire 60 d period are presented in Figs. S15–S44 in the Supplement. Note that in Fig. 14, \hat{y}_H and \hat{y}_C represent the simulated spatial distributions of hydraulic heads and solute concentrations based on the estimated permeability fields through inverse modeling, while y_H and y_C represent the spatial distributions simulated using the true permeability field.

According to Fig. 14a, the calibrated simulation errors for hydraulic heads did not exceed 0.02 m for the TNNA method and three of the four considered metaheuristic algorithms, except the PSO method, which exhibited hydraulic head errors larger than 0.06 m in certain areas. Among the four metaheuristic algorithms, the GA method achieved the lowest RMSE in hydraulic head simulations, with a value of 7.4837×10^{-4} . For solute concentrations, the GA algorithm consistently has the highest prediction accuracy among the metaheuristic algorithms, with RMSE values generally around 0.005 (Fig. 13). The TNNA algorithm achieved a similar level of accuracy to GA in the calibrated numerical model predictions. Specifically, during the initial 10 d and from the 41st day to the 60th day, the TNNA algorithm showed slightly higher prediction accuracy than the GA-calibrated model. However, during the intermediate period from the 10th day to the 40th day, the GA-calibrated model had a slight advantage over the TNNA algorithm. The normalized absolute errors in the solute transport simulation results obtained using the TNNA algorithm remained consistently below 0.02 throughout the simulation period (Fig. 14b and c).

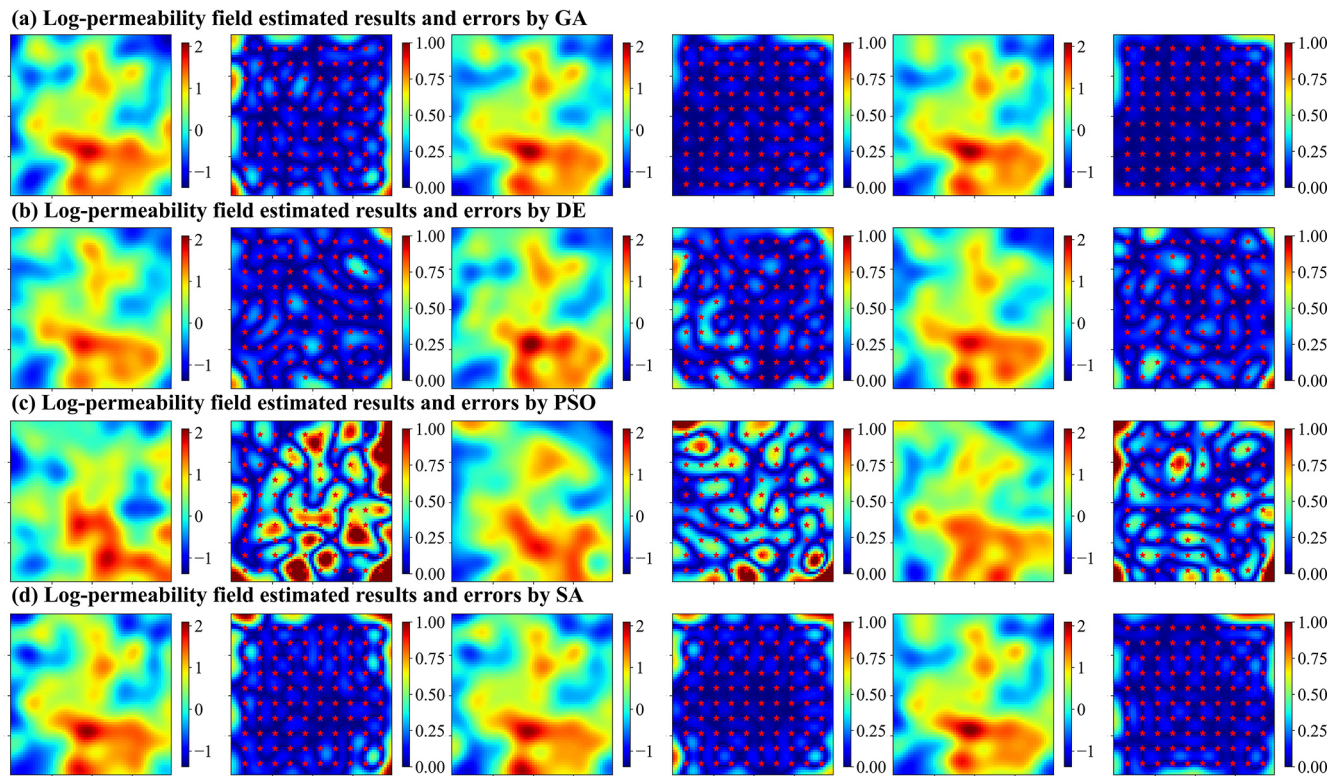


Figure 10. Spatial distribution of log-permeability field estimation results (row 1, 3, and 5 for $N_{PC} = 100, 500$, and 1000 , respectively) and absolute errors (row 2, 4, and 6 for $N_{PC} = 100, 500$, and 1000 , respectively) for scenario 5, achieved by four metaheuristic algorithms (panels a–d correspond to GA, DE, PSO, and SA, respectively).

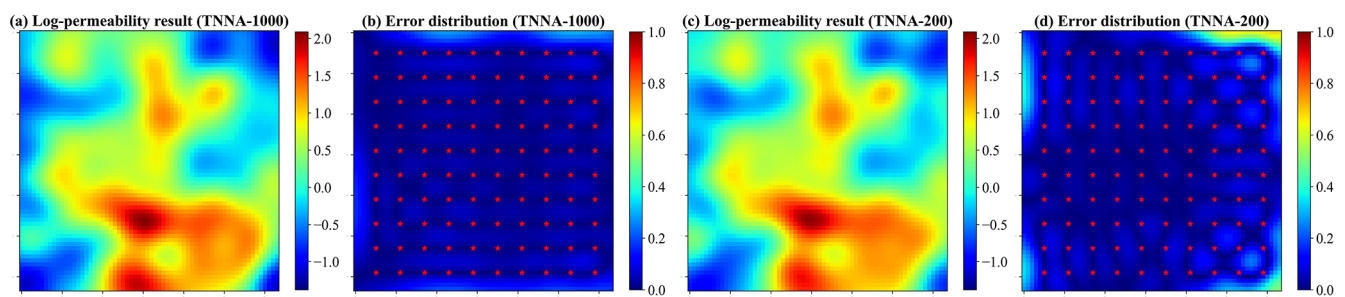


Figure 11. Spatial distribution of log-permeability field estimation results and absolute errors for scenario 5, achieved by TNNA. Panels (a) and (c) show the log-permeability fields estimated using 1000 (TNNA-1000) and 200 (TNNA-200) training samples, respectively; panels (b) and (d) present the corresponding absolute error distributions.

These results indicate that in high-dimensional settings, the TNNA algorithm provides inversion outcomes that enable the calibrated model to deliver simulation results comparable to those of the best-performing metaheuristic algorithm. Overall, the TNNA method also demonstrates advantages over the four metaheuristic optimization algorithms in the designed high-dimensional scenarios, excelling in both inversion efficiency and accuracy.

4.2.3 Inversion results of the high-dimensional non-Gaussian scenario

In this scenario, the iteration number for the four metaheuristic algorithms was set at 200, with N_{PC} values of 1000. For the TNNA method, the reverse network is trained for 1000 epochs. Thus, each metaheuristic algorithm had 100 times more forward model evaluations than the TNNA algorithm. Figures 14 and 15 show the permeability fields estimated by the five optimization algorithms and their error distributions compared to the true field (i.e., the error fields). Figures 16a

Table 2. RMSE values of estimated log-permeability fields for the four metaheuristic algorithms and the TNNA algorithm under scenario 5.

	Metaheuristic algorithms				TNNA	
	GA	DE	PSO	SA		
$N_{PC} = 100$	0.1940	0.1597	0.5399	0.2071	epoch = 200	0.1063
$N_{PC} = 500$	0.1142	0.1904	0.3810	0.1781	epoch = 1000	0.0595
$N_{PC} = 1000$	0.1057	0.1748	0.3334	0.1549		

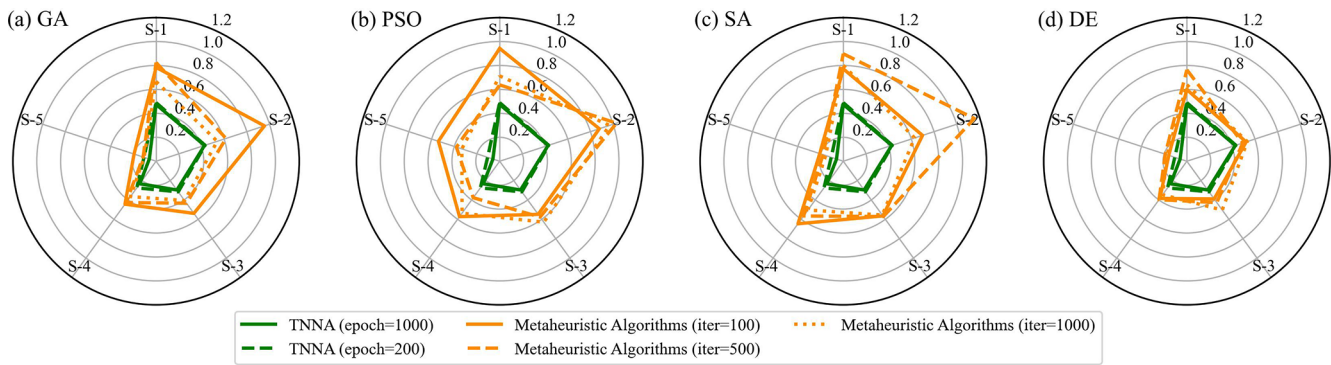


Figure 12. Comparison of RMSE in estimating log-permeability fields using four metaheuristic algorithms and the TNNA algorithm across the five scenarios (S-1 to S-5).

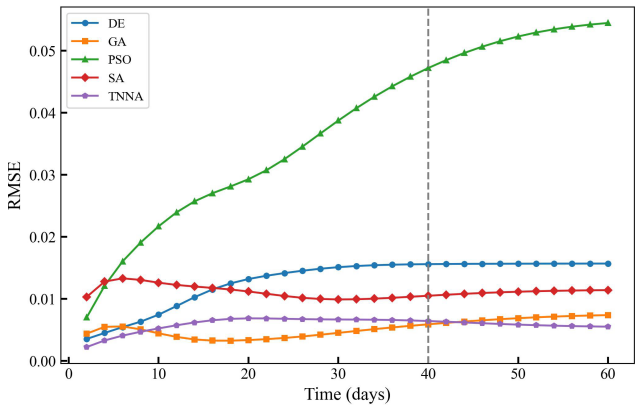


Figure 13. RMSE values of calibrated solute concentrations over 60 d for the four metaheuristic algorithms and the TNNA algorithm.

and 17a present the comparison between calibrated simulations and hydraulic head observations, as well as solute concentration observations. Figures 16b and 17b compare the solute concentration simulations for the 26th, 28th, and 30th years based on the estimated parameter field and the designed true field.

According to Figs. 15 and 16, the binary channel fields reconstructed by each inversion algorithm are highly consistent with their corresponding true fields, with the estimated errors primarily concentrated at the interfaces between high-permeability channels and low-permeability regions. It is found that increasing the observation noise level from 1 %

to 10 % does not lead to a noticeable increase in the number of grid cells exhibiting differences between the estimated parameter fields and the true field. One potential reason for this is that the least squares objective function used in the inversion framework of this study is based on the assumption that the observation noise follows a zero-mean Gaussian distribution. With adequate regularization constraints, such as the dense monitoring network design used in this study, the model responses corresponding to the optimal parameter estimates obtained through global optimization algorithms statistically converge to the mean of the observed data. It can also be evaluated by the calibration simulations. Specifically, the pairwise scatter plots in Figs. 17a and 18a indicate that the calibrated simulation results from different methods are closely distributed around the reference diagonal. This suggests that even with increased observational noise, the inversion-derived calibration results do not exhibit noticeable bias. Furthermore, the predictions based on inversion results remain highly consistent with those of the true permeability field (Figs. 17b and 18b). The $RMSE_{All}$ and R^2_{All} values for the predictions beyond the observational period range from 0.018 to 0.044 and from 0.962 to 0.994, respectively. This indicates that even under relatively high Gaussian noise conditions, the non-linear inversion framework used in this study can reliably reconstruct the non-Gaussian permeability field, ensuring high predictive accuracy. Nevertheless, it is important to note that while the inversion accuracy at a 10 % noise level remains comparable to that in the 1 % noise scenario, increasing the observational noise inevitably raises the con-

Table 3. RMSE values of calibrated hydraulic heads for the four metaheuristic algorithms and the TNNA algorithm.

	TNNA	DE	GA	PSO	SA
RMSE	6.8537×10^{-4}	1.2181×10^{-3}	7.4837×10^{-4}	2.1683×10^{-3}	1.0316×10^{-3}

vergence value of the least squares loss function. This trend is evident from the RMSE values in Figs. 17a and 18a. Moreover, since the observational noise here is assumed to follow a Gaussian distribution, real-world scenarios with more complex noise characteristics may further exacerbate equifinality in the inversion results. In such cases, incorporating additional system information such as regularization constraints is essential to enhance the robustness of the objective function and to mitigate ill-posedness.

Compared to the four metaheuristic algorithms, TNNA demonstrates advantages in computational efficiency and accuracy for non-Gaussian random field inversion. In the low-noise scenario, TNNA achieves an inversion convergence accuracy with an RMSE_{All} of 0.021 and an R^2_{All} of 0.996 (Fig. 17a). In contrast, the two best-performing metaheuristic methods, GA and SA, yield RMSE_{All} values of 0.027 and 0.029, with R^2_{All} values of 0.994 and 0.993, respectively (Fig. 17a). Moreover, TNNA achieves the highest fitting accuracy for predictive results among the five optimization algorithms, with an RMSE of 0.018 and an R^2 of 0.994 (Fig. 17b). Even in high-noise scenarios, TNNA continues to exhibit an advantage over the four metaheuristic algorithms in both inversion convergence accuracy (Fig. 18a) and predictive accuracy (Fig. 18b). Additionally, considering the number of forward simulation calls required by each inversion algorithm, TNNA proves to be a more efficient approach in this case study.

4.3 Parameter inversion method comparison results

This study evaluates the computational efficiency and inversion reliability of the TNNA algorithm under three different heterogeneous conditions. In optimization-based inversion studies, the primary challenge is to establish non-linear inversion constraints and design efficient algorithms to find optimal parameter solutions. The main difference between cases lies in how the constraint conditions are formulated, while the optimization algorithm itself remains generally applicable across different optimization tasks if these conditions are properly defined. Therefore, the fundamental challenge in applying well-performing inversion methods to real-world cases lies in whether robust non-linear optimization constraints can be effectively established for inversion tasks. Given the complexities of subsurface systems, three key aspects should be considered to extend the TNNA method to real-world applications: (1) representing complex heterogeneous model parameter fields, (2) maximizing the effective observational information while optimizing monitoring

costs, and (3) integrating multi-source data and accounting for uncertainties in the model process to better address complex observational noise scenarios and uncertainties in physical mechanisms. Detailed considerations for these issues are as follows.

- *Heterogeneity in aquifer parameter structures.* This study developed a dimensionality-reduction framework using OCAAE for high-dimensional parameter field inversion. Generative machine learning methods, including state-of-the-art variants, have the potential to characterize complex non-Gaussian fields. However, obtaining representative parameter field datasets that accurately capture the spatial variability and heterogeneous geostatistical characteristics of the target aquifer remains challenging in practical research. For instance, spatial variations in non-stationary stochastic aquifer systems may result in significant discrepancies in geostatistical parameters across sampling windows (Mariethoz and Caers, 2014). Therefore, developing appropriate generator-training strategies is essential for these practical scenarios.
- *Monitoring network optimization.* The inversion performance of the TNNA and four metaheuristic algorithms is evaluated based on a non-linear optimization model with dense distributed monitoring networks. This monitoring strategy is commonly employed in the evaluation of inversion algorithms to ensure sufficient observational information, thereby reducing non-uniqueness in parameter inversion results (Bao et al., 2020; Mo et al., 2020; Zhang et al., 2024). Such monitoring strategies for comparing inversion methods also aim to minimize external interferences, ensuring that differences in performance are primarily determined by inversion algorithms themselves. However, the number and locations of monitoring stations are constrained by financial budgets. Thus, optimizing the monitoring network design to minimize monitoring costs without compromising constraint information quality is indispensable for practical applications (Keum et al., 2018; Chen et al., 2022; Cao et al., 2025).
- *Considering multi-source data and uncertainties in model processes.* This study considers only hydraulic head and solute concentration data, assuming ideal white Gaussian noises. However, in real-world scenarios, observational noise is often more complex and may exhibit non-Gaussian characteristics. For instance,

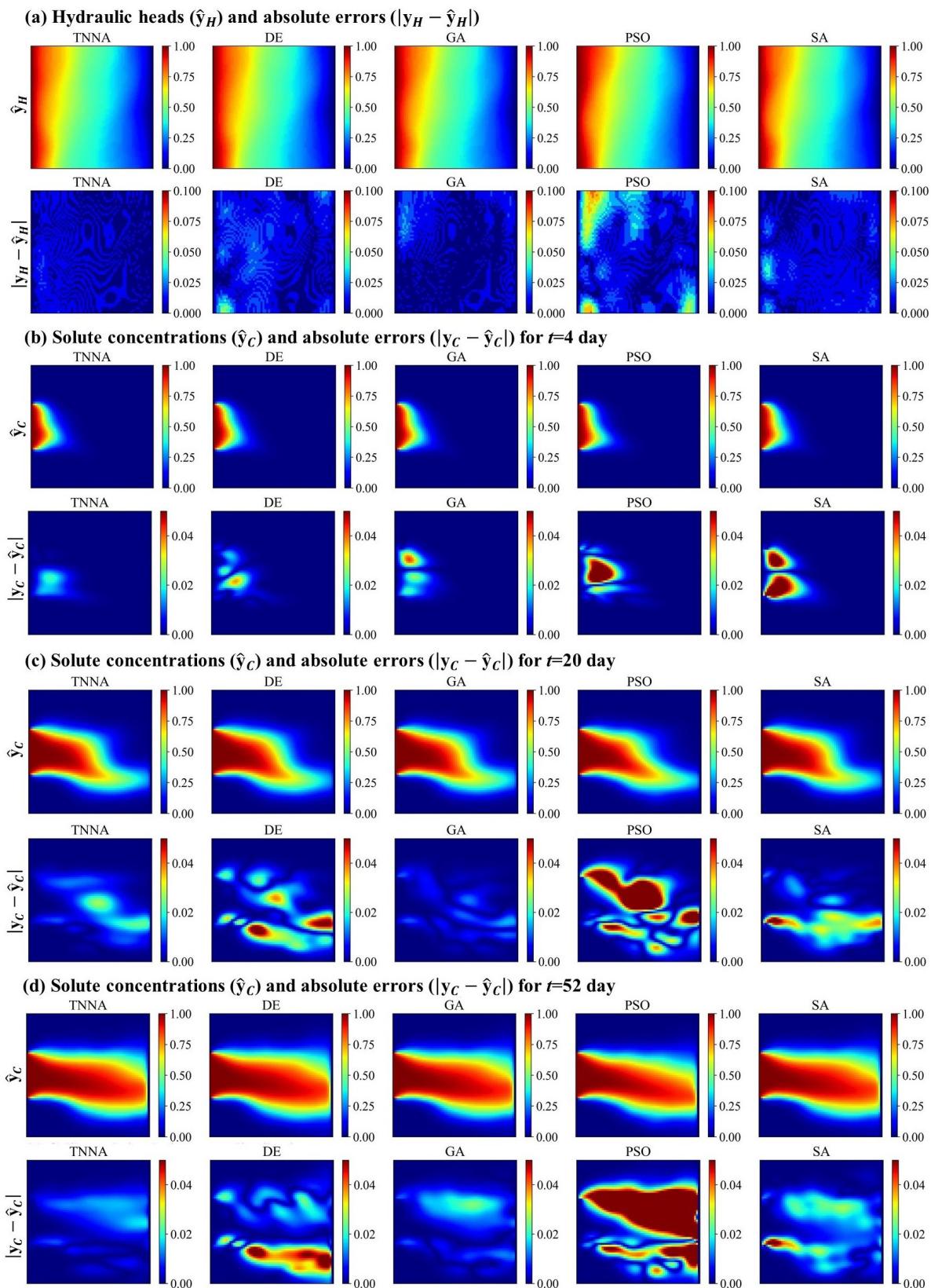


Figure 14. Spatial distribution of calibrated numerical simulation results and absolute errors for hydraulic heads and solute concentrations at three dynamic times ($t = 4, 20$, and 50 d) using the TNNA algorithm and four metaheuristic algorithms.

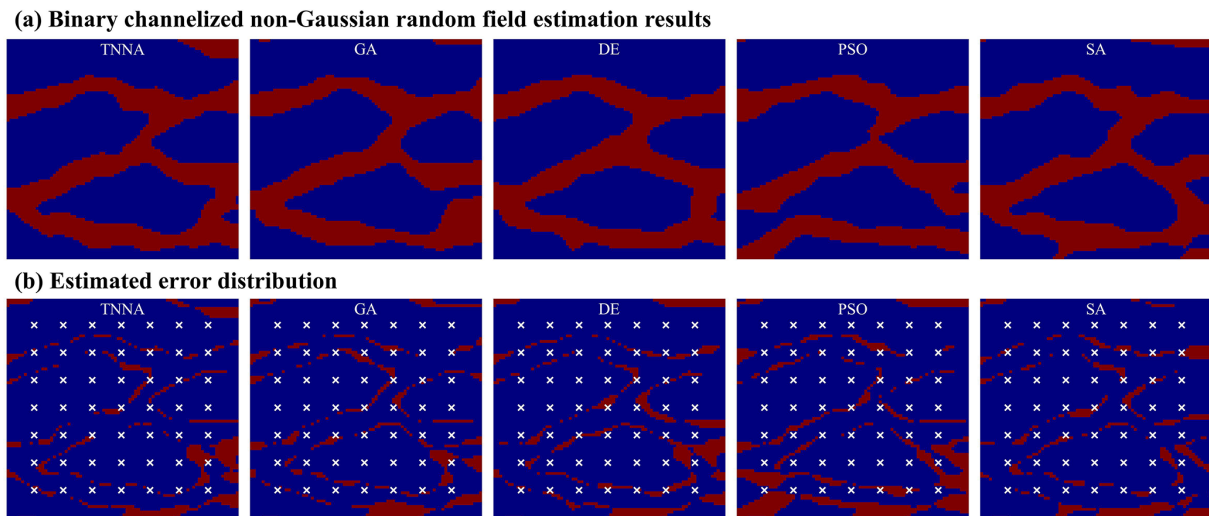


Figure 15. Reconstructed non-Gaussian binary channelized fields and their error distributions (1 % observation noise).

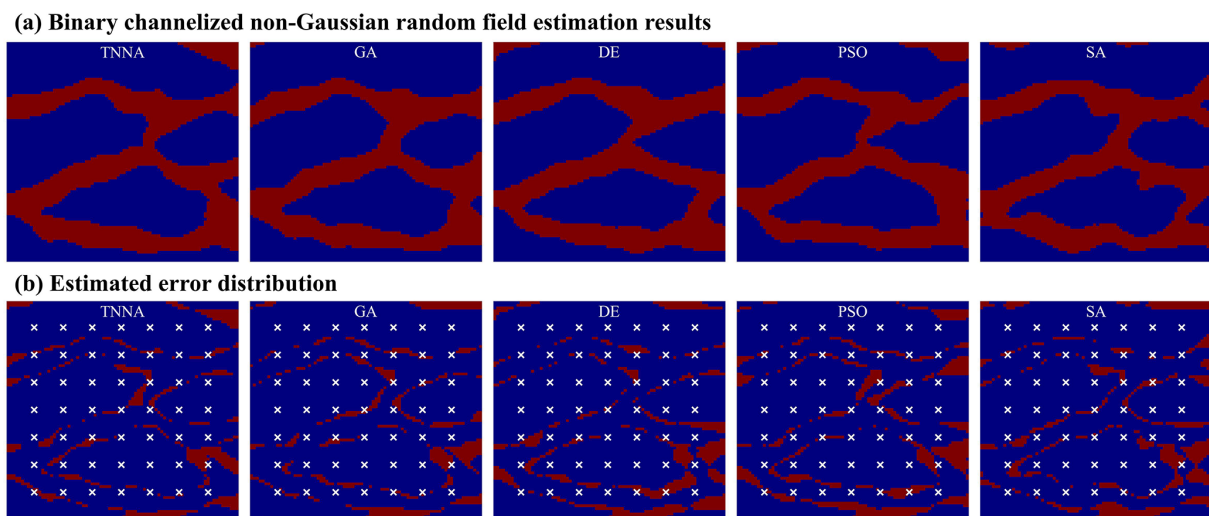


Figure 16. Reconstructed non-Gaussian binary channelized fields and their error distributions (10 % observation noise).

some solute concentrations cannot be measured in situ, and unavoidable perturbations may be included during sample collection and laboratory analysis. Similarly, hydraulic head measurements may be influenced by other factors, including meteorological conditions, human groundwater extraction, and engineering disturbances, among others. Moreover, all observational data in this study are constrained by a single predetermined process model. However, if significant uncertainties exist in the actual aquifer model processes or if the conceptual model deviates substantially from real-world conditions, even an advanced optimization algorithm may produce incorrect inversion results. Therefore, it is crucial to integrate multi-source data (e.g., geophysical measurements or isotope data) and to develop multi-

process coupled models to establish more robust inversion frameworks (Dai and Samper, 2006; Botto et al., 2018; Chang and Zhang, 2019). Specifically, parameterizing model process uncertainties to enable the simultaneous identification of both model processes and unknown parameters could be a promising direction for real-world studies.

5 Summary and conclusions

This study systematically evaluates the performance of tandem neural network architecture (TNNA) in comparison to four widely used metaheuristic algorithms (GA, PSO, DE, and SA) across three inversion frameworks designed for different heterogeneous groundwater conditions. The re-

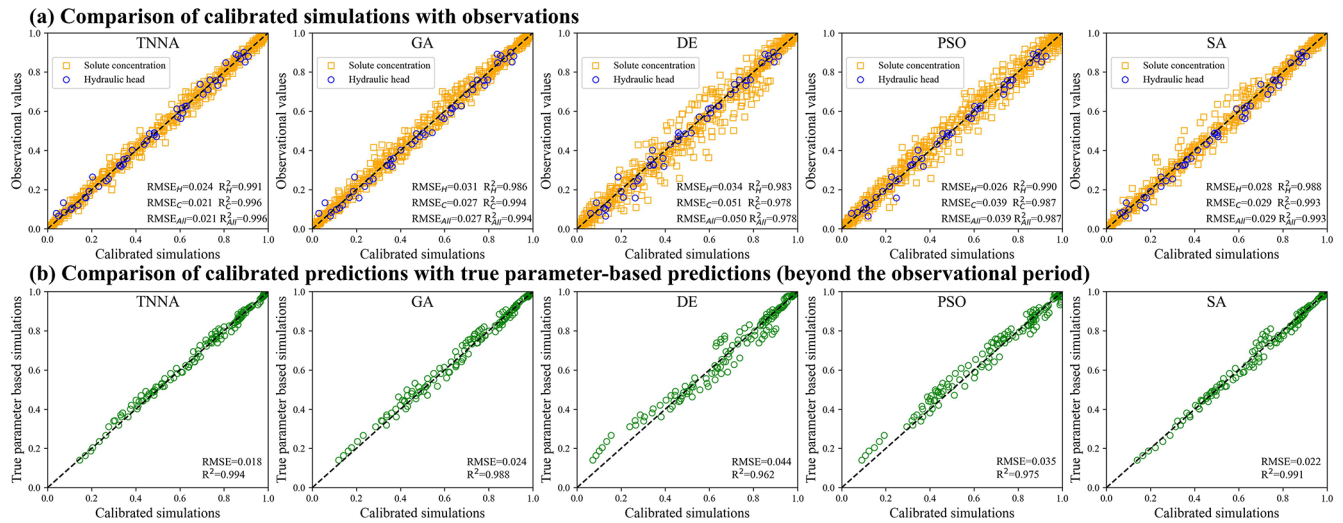


Figure 17. Pair-wise comparison between the calibrated simulation results and the observational data (a); and the true parameter-based predictions (1 % observation noise).

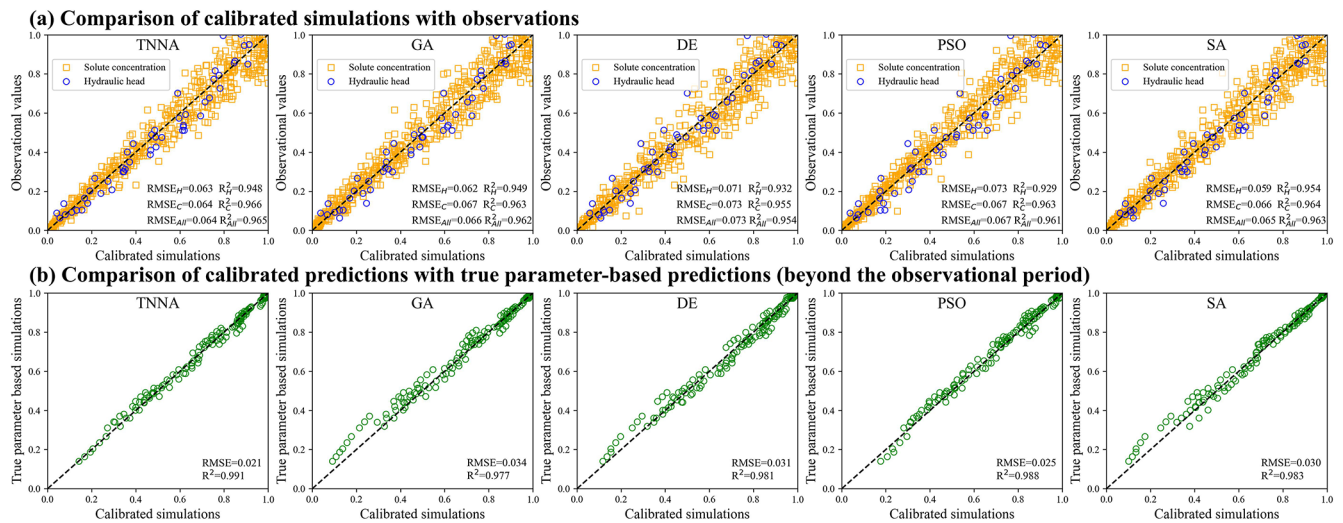


Figure 18. Pair-wise comparison between the calibrated simulation results and the observational data (a); and the true parameter-based predictions (10 % observation noise).

sults demonstrate that TNNA consistently outperforms the four conventional metaheuristic algorithms across the designed scenarios, covering both low-dimensional and high-dimensional cases. It provides more accurate inversion results while significantly reducing computational costs. Moreover, it has been verified that the TNNA algorithm consistently delivers reliable inversion results with just a single forward simulation per iteration step in scenarios featuring various complex and uncertain model parameters. This characteristic offers a practical approach to balancing exploration and exploitation with a reduced computational burden, contrasting with conventional metaheuristic algorithms that require increasing forward simulations as the inversion problem grows more complex. Furthermore, this study intro-

duces a novel framework that integrates TNNA, along with optimization algorithms, with generative machine-learning-based parameterization methods for dimensionality reduction in complex heterogeneous parameter fields.

In summary, training reverse network through the TNNA method provides significant advantages over conventional metaheuristic algorithms. The proposed integrated framework, which combines the TNNA method with dimensionality reduction techniques, further enhances its applicability and demonstrates strong potential for high-dimensional inversion problems. Developing specialized inversion algorithm frameworks based on state-of-the-art machine learning methods tailored to specific problem scenarios represents a promising research direction. Furthermore, hyperparam-

ters can significantly influence neural network performance in certain scenarios. It is necessary for future research to explore hyperparameter optimization and sensitivity analysis to identify the optimal neural network structures and training strategies, ultimately enhancing model performance across diverse hydrological conditions.

Data availability. The data and codes for four surrogate models and five optimization algorithms are available online at <https://doi.org/10.5281/zenodo.10499582> (Chen et al., 2024).

Supplement. The supplement related to this article is available online at <https://doi.org/10.5194/hess-29-4251-2025-supplement>.

Author contributions. JC: conceptualization, methodology, writing (original draft), formal analysis, and funding acquisition. ZD: supervision, funding acquisition, and writing (review and editing). SY: writing (review and editing). MZ: writing (review and editing). MRS: writing (review and editing).

Competing interests. The contact author has declared that none of the authors has any competing interests.

Disclaimer. Publisher's note: Copernicus Publications remains neutral with regard to jurisdictional claims made in the text, published maps, institutional affiliations, or any other geographical representation in this paper. While Copernicus Publications makes every effort to include appropriate place names, the final responsibility lies with the authors.

Acknowledgements. This work is supported by the National Natural Science Foundation of China (NSFC: 42402241, U2267217, 42141011, and 42002254) and the Fundamental Research Funds for the Central Universities (2024QN11066). The authors would like to thank the editor and reviewers for their constructive comments and suggestions, and they also thank Chuanjun Zhan from Qingdao University of Technology for providing the source code for OCAAE.

Financial support. This research has been supported by the National Natural Science Foundation of China (grant nos. 42402241, U2267217, 42141011, and 42002254) and the Fundamental Research Funds for the Central Universities (grant no. 2024QN11066).

Review statement. This paper was edited by Mauro Giudici and reviewed by two anonymous referees.

References

- Abbas, S. A., Bailey, R. T., White, J. T., Arnold, J. G., White, M. J., Čerkasova, N., and Gao, J.: A framework for parameter estimation, sensitivity analysis, and uncertainty analysis for holistic hydrologic modeling using SWAT+, *Hydrol. Earth Syst. Sci.*, 28, 21–48, <https://doi.org/10.5194/hess-28-21-2024>, 2024.
- Adler, J. and Öktem, O.: Solving ill-posed inverse problems using iterative deep neural networks, *Inverse Probl.*, 33, 124007, <https://doi.org/10.1088/1361-6420/aa9581>, 2017.
- Arsenault, R. and Brissette, F. P.: Continuous streamflow prediction in ungauged basins: The effects of equifinality and parameter set selection on uncertainty in regionalization approaches, *Water Resour. Res.*, 50, 6135–6153, <https://doi.org/10.1002/2013wr014898>, 2014.
- Bandai, T. and Ghezzehei, T. A.: Forward and inverse modeling of water flow in unsaturated soils with discontinuous hydraulic conductivities using physics-informed neural networks with domain decomposition, *Hydrol. Earth Syst. Sci.*, 26, 4469–4495, <https://doi.org/10.5194/hess-26-4469-2022>, 2022.
- Bao, J., Li, L., and Redoloz, F.: Coupling ensemble smoother and deep learning with generative adversarial networks to deal with non-Gaussianity in flow and transport data assimilation, *J. Hydrol.*, 590, 125443, <https://doi.org/10.1016/j.jhydrol.2020.125443>, 2020.
- Bentivoglio, R., Iusfi, E., Jonkman, S. N., and Taormina, R.: Deep learning methods for flood mapping: a review of existing applications and future research directions, *Hydrol. Earth Syst. Sci.*, 26, 4345–4378, <https://doi.org/10.5194/hess-26-4345-2022>, 2022.
- Beven, K. and Binley, A.: The future of distributed models: Model calibration and uncertainty prediction, *Hydrol. Process.*, 6, 279–298, <https://doi.org/10.1002/hyp.3360060305>, 1992.
- Blasone, R.-S., Madsen, H., and Rosbjerg, D.: Parameter estimation in distributed hydrological modelling: comparison of global and local optimisation techniques, *Hydrol. Res.*, 38, 451–476, <https://doi.org/10.2166/nh.2007.024>, 2007.
- Botto, A., Belluco, E., and Camporese, M.: Multi-source data assimilation for physically based hydrological modeling of an experimental hillslope, *Hydrol. Earth Syst. Sci.*, 22, 4251–4266, <https://doi.org/10.5194/hess-22-4251-2018>, 2018.
- Cao, M., Dai, Z., Chen, J., Yin, H., Zhang, X., Wu, J., Thanh, H. V., and Soltanian, M. R.: An integrated framework of deep learning and entropy theory for enhanced high-dimensional permeability field identification in heterogeneous aquifers, *Water Res.*, 268, 122706, <https://doi.org/10.1016/j.watres.2024.122706>, 2025.
- Carrera, J. and Glorioso, L.: On geostatistical formulations of the groundwater flow inverse problem, *Adv. Water Resour.*, 14, 273–283, [https://doi.org/10.1016/0309-1708\(91\)90039-Q](https://doi.org/10.1016/0309-1708(91)90039-Q), 1991.
- Castaings, W., Dartus, D., Le Dimet, F.-X., and Saulnier, G.-M.: Sensitivity analysis and parameter estimation for distributed hydrological modeling: potential of variational methods, *Hydrol. Earth Syst. Sci.*, 13, 503–517, <https://doi.org/10.5194/hess-13-503-2009>, 2009.
- Chang, H. and Zhang, D.: Identification of physical processes via combined data-driven and data-assimilation methods, *J. Comput. Phys.*, 393, 337–350, <https://doi.org/10.1016/j.jcp.2019.05.008>, 2019.
- Chang, Z., Lu, W., and Wang, Z.: Study on source identification and source-sink relationship of LNAPLs pollution in groundwater by the adaptive cyclic improved iterative process and

- Monte Carlo stochastic simulation, *J. Hydrol.*, 612, 128109, <https://doi.org/10.1016/j.jhydrol.2022.128109>, 2022.
- Chen, J., Dai, Z., Yang, Z., Pan, Y., Zhang, X., Wu, J., and Reza Soltanian, M.: An improved tandem neural network architecture for inverse modeling of multicomponent reactive transport in porous media, *Water Resour. Res.*, 57, 2021WR030595, <https://doi.org/10.1029/2021wr030595>, 2021.
- Chen, J., Dai, Z., Dong, S., Zhang, X., Sun, G., Wu, J., Ershadnia, R., Yin, S., and Soltanian, M. R.: Integration of deep learning and information theory for designing monitoring networks in heterogeneous aquifer systems, *Water Resour. Res.*, 58, 2022WR032429, <https://doi.org/10.1029/2022wr032429>, 2022.
- Chen, J., Dai, Z., Yin, S., Zhang, M., and Soltanian, M. R.: Enhancing Inverse Modeling in Groundwater Systems through Machine Learning: A Comprehensive Comparative Study, Zenodo [dataset], <https://doi.org/10.5281/zenodo.10499582>, 2024.
- Chen, X., Hammond, G. E., Murray, C. J., Rockhold, M. L., Vermeul, V. R., and Zachara, J. M.: Application of ensemble-based data assimilation techniques for aquifer characterization using tracer data at Hanford 300 area, *Water Resour. Res.*, 49, 7064–7076, <https://doi.org/10.1002/2012wr013285>, 2013.
- Dai, Z. and Samper, J.: Inverse problem of multicomponent reactive chemical transport in porous media: Formulation and applications, *Water Resour. Res.*, 40, W07407, <https://doi.org/10.1029/2004wr003248>, 2004.
- Dai, Z. and Samper, J.: Inverse modeling of water flow and multicomponent reactive transport in coastal aquifer systems, *J. Hydrol.*, 447–461, <https://doi.org/10.1016/j.jhydrol.2005.11.052>, 2006.
- Dragonetti, G., Comegna, A., Ajeel, A., Deidda, G. P., Lamadalena, N., Rodriguez, G., Vignoli, G., and Coppola, A.: Calibrating electromagnetic induction conductivities with time-domain reflectometry measurements, *Hydrol. Earth Syst. Sci.*, 22, 1509–1523, <https://doi.org/10.5194/hess-22-1509-2018>, 2018.
- Eberhart, R. and Kennedy, J.: Particle swarm optimization, in: Proceedings of the IEEE international conference on neural networks, Perth, Western Australia, Australia, 27 November 1995, 1942–1948, <https://doi.org/10.1109/ICNN.1995.488968>, 1995.
- Elfving, S., Uchibe, E., and Doya, K.: Sigmoid-weighted linear units for neural network function approximation in reinforcement learning, *Neural Networks*, 107, 3–11, <https://doi.org/10.1016/j.neunet.2017.12.012>, 2018.
- Epp, R., Schmid, F., and Jenny, P.: Fast convergence strategy for ambiguous inverse problems based on hierarchical regularization, *J. Comput. Phys.*, 489, 112264, <https://doi.org/10.1016/j.jcp.2023.112264>, 2023.
- Ghelichkhan, S., Gibson, A., Davies, D. R., Kramer, S. C., and Ham, D. A.: Automatic adjoint-based inversion schemes for geodynamics: reconstructing the evolution of Earth's mantle in space and time, *Geosci. Model Dev.*, 17, 5057–5086, <https://doi.org/10.5194/gmd-17-5057-2024>, 2024.
- Ginn, T. R. and Cushman, J. H.: Inverse methods for subsurface flow: A critical review of stochastic techniques, *Stoch. Hydrol. Hydraul.*, 4, 1–26, <https://doi.org/10.1007/BF01547729>, 1990.
- Giudici, M.: Some Remarks About Forward and Inverse Modelling in Hydrology, Within a General Conceptual Framework, *Hydrol.*, 11, 189, <https://doi.org/10.3390/hydrology11110189>, 2024.
- Goodfellow, I., Bengio, Y., and Courville, A.: Deep learning, MIT press, Cambridge, MA, USA, 800 pp., ISBN 9780262337373, 2016.
- Guo, Q., Liu, M., and Luo, J.: Predictive Deep Learning for High-Dimensional Inverse Modeling of Hydraulic Tomography in Gaussian and Non-Gaussian Fields, *Water Resour. Res.*, 59, 2023WR035408, <https://doi.org/10.1029/2023wr035408>, 2023.
- He, K., Zhang, X., Ren, S., and Sun, J.: Deep residual learning for image recognition, in: Proceedings of the IEEE conference on computer vision and pattern recognition, Las Vegas, NV, USA, 27–30 June 2016, 770–778, <https://doi.org/10.1109/CVPR.2016.90>, 2016.
- Hinton, G. E. and Salakhutdinov, R. R.: Reducing the dimensionality of data with neural networks, *Science*, 313, 504–507, <https://doi.org/10.1126/science.1127647>, 2006.
- Holland John, H.: Adaptation in natural and artificial systems, University of Michigan Press, Ann Arbor, ISBN 10 0472084607, 13 978-0472084609, 1975.
- Hopmans, J. W., Šimůnek, J., Romano, N., and Durner, W.: 3.6.2. Inverse Methods, in: Methods of Soil Analysis, edited by: Dane, J. H. and Clarke Topp, G., Wiley, 963–1008, <https://doi.org/10.2136/sssabookser5.4.c40>, 2002.
- Ines, A. V. M. and Droogers, P.: Inverse modelling in estimating soil hydraulic functions: a Genetic Algorithm approach, *Hydrol. Earth Syst. Sci.*, 6, 49–66, <https://doi.org/10.5194/hess-6-49-2002>, 2002.
- Jardani, A., Vu, T. M., and Fischer, P.: Use of convolutional neural networks with encoder-decoder structure for predicting the inverse operator in hydraulic tomography, *J. Hydrol.*, 604, 127233, <https://doi.org/10.1016/j.jhydrol.2021.127233>, 2022.
- Jaumann, S. and Roth, K.: Soil hydraulic material properties and layered architecture from time-lapse GPR, *Hydrol. Earth Syst. Sci.*, 22, 2551–2573, <https://doi.org/10.5194/hess-22-2551-2018>, 2018.
- Jose, S. C., Rahman, M. A., and Cirpka, O. A.: Large-scale sandbox experiment on longitudinal effective dispersion in heterogeneous porous media, *Water Resour. Res.*, 40, W12415, <https://doi.org/10.1029/2004wr003363>, 2004.
- Kabala, Z. J. and Skaggs, T. H.: Comment on “Minimum relative entropy inversion: Theory and application to recovering the release history of a groundwater contaminant” by Allan, D. Woodbury and Tadeusz, J. Ulrych, *Water Resour. Res.*, 34, 2077–2079, <https://doi.org/10.1029/98WR01337>, 1998.
- Kapsoulis, D., Tsiakas, K., Trompoukis, X., Asouti, V., and Giannakoglou, K.: A PCA-assisted hybrid algorithm combining EAs and adjoint methods for CFD-based optimization, *Appl. Soft Comput.*, 73, 520–529, <https://doi.org/10.1016/j.asoc.2018.09.002>, 2018.
- Keum, J., Coulibaly, P., Razavi, T., Tapsoba, D., Gobena, A., Weber, F., and Pietroniro, A.: Application of SNODAS and hydrologic models to enhance entropy-based snow monitoring network design, *J. Hydrol.*, 561, 688–701, <https://doi.org/10.1016/j.jhydrol.2018.04.037>, 2018.
- Kirkpatrick, S., Gelatt, C. D., and Vecchi, M. P.: Optimization by Simulated Annealing, *Science*, 220, 671–680, <https://doi.org/10.1126/science.220.4598.671>, 1983.
- Kool, J. B., Parker, J. C., and van Genuchten, M. T.: Parameter estimation for unsaturated flow and transport models-A

- review, *J. Hydrol.*, 91, 255–293, [https://doi.org/10.1016/0022-1694\(87\)90207-1](https://doi.org/10.1016/0022-1694(87)90207-1), 1987.
- Kuang, W., Yuan, C., and Zhang, J.: Real-time determination of earthquake focal mechanism via deep learning, *Nat. Commun.*, 12, 1432, <https://doi.org/10.1038/s41467-021-21670-x>, 2021.
- LeCun, Y., Bengio, Y., and Hinton, G.: Deep learning, *Nature*, 521, 436–444, <https://doi.org/10.1038/nature14539>, 2015.
- Li, E.: An adaptive surrogate assisted differential evolutionary algorithm for high dimensional constrained problems, *Appl. Soft Comput.*, 85, 105752, <https://doi.org/10.1016/j.asoc.2019.105752>, 2019.
- Lindsay, A., McCloskey, J., and Bhloscaidh, M. N.: Using a genetic algorithm to estimate the details of earthquake slip distributions from point surface displacements, *J. Geophys. Res.-Sol. Ea.*, 121, 1796–1820, <https://doi.org/10.1002/2015jb012181>, 2016.
- Liu, D., Tan, Y., Khoram, E., and Yu, Z.: Training deep neural networks for the inverse design of nanophotonic structures, *ACS Photonics*, 5, 1365–1369, <https://doi.org/10.1021/acsp Photonics.7b01377>, 2018.
- Liu, M., Ahmad, R., Cai, W., and Mukerji, T.: Hierarchical Homogenization With Deep-Learning-Based Surrogate Model for Rapid Estimation of Effective Permeability From Digital Rocks, *J. Geophys. Res.-Sol. Ea.*, 128, e2022JB025378, <https://doi.org/10.1029/2022jb025378>, 2023.
- Loève, M.: Probability Theory, Van Nostrand, New York, 1955.
- Long, Y., Ren, J., Li, Y., and Chen, H.: Inverse design of photonic topological state via machine learning, *Appl. Phys. Lett.*, 114, 181105, <https://doi.org/10.1063/1.5094838>, 2019.
- Luo, J., Ma, X., Ji, Y., Li, X., Song, Z., and Lu, W.: Review of machine learning-based surrogate models of groundwater contaminant modeling, *Environ. Res.*, 238, 117268, <https://doi.org/10.1016/j.envres.2023.117268>, 2023.
- Ma, J., Xia, D., Guo, H., Wang, Y., Niu, X., Liu, Z., and Jiang, S.: Metaheuristic-based support vector regression for landslide displacement prediction: a comparative study, *Landslides*, 19, 2489–2511, <https://doi.org/10.1007/s10346-022-01923-6>, 2022.
- Makhzani, A., Shlens, J., Jaitly, N., Goodfellow, I., and Frey, B.: Adversarial autoencoders, *arXiv [preprint]*, <https://doi.org/10.48550/arXiv.1511.05644>, 2015.
- Mariethoz, G. and Caers, J.: Multiple-point geostatistics: stochastic modeling with training images, Wiley Blackwell, 364 pp., <https://doi.org/10.1002/9781118662953>, 2014.
- Mariethoz, G., Renard, P., and Straubhaar, J.: The Direct Sampling method to perform multiple-point geostatistical simulations, *Water Resour. Res.*, 46, W11536, <https://doi.org/10.1029/2008WR007621>, 2010.
- McLaughlin, D. and Townley, L. R.: A Reassessment of the Groundwater Inverse Problem, 32, 1131–1161, <https://doi.org/10.1029/96WR00160>, 1996.
- Metropolis, N., Rosenbluth, A. W., Rosenbluth, M. N., Teller, A. H., and Teller, E.: Equation of state calculations by fast computing machines, *J. Chem. Phys.*, 21, 1087–1092, <https://doi.org/10.1063/1.1699114>, 1953.
- Mo, S., Zabarar, N., Shi, X., and Wu, J.: Deep autoregressive neural networks for high-dimensional inverse problems in groundwater contaminant source identification, *Water Resour. Res.*, 55, 3856–3881, <https://doi.org/10.1029/2018wr024638>, 2019.
- Mo, S., Zabarar, N., Shi, X., and Wu, J.: Integration of adversarial autoencoders with residual dense convolutional networks for estimation of non-Gaussian hydraulic conductivities, *Water Resour. Res.*, 56, 2019WR026082, <https://doi.org/10.1029/2019WR026082>, 2020.
- Neupauer, R. M., Borchers, B., and Wilson, J. L.: Comparison of inverse methods for reconstructing the release history of a groundwater contamination source, *Water Resour. Res.*, 36, 2469–2475, <https://doi.org/10.1029/2000WR900176>, 2000.
- Nhu, V. H.: Levenberg–Marquardt method for ill-posed inverse problems with possibly non-smooth forward mappings between Banach spaces, *Inverse Probl.*, 38, 015007, <https://doi.org/10.1088/1361-6420/ac38b7>, 2022.
- Pérez-Cruz, F., Camps-Valls, G., Soria-Olivas, E., Pérez-Ruixo, J. J., Figueiras-Vidal, A. R., and Artés-Rodríguez, A.: Multi-dimensional Function Approximation and Regression Estimation, Artificial Neural Networks – ICANN 2002, Berlin, Heidelberg, 757–762, https://doi.org/10.1007/3-540-46084-5_123, 2002.
- Plessix, R.: A review of the adjoint-state method for computing the gradient of a functional with geophysical applications, *Geophys. J. Int.*, 167, 495–503, <https://doi.org/10.1111/j.1365-246X.2006.02978.x>, 2006.
- Qin, Y., Kavetski, D., Kuczera, G., McInerney, D., Yang, T., and Guo, Y.: Can Gauss-Newton Algorithms Outperform Stochastic Optimization Algorithms When Calibrating a Highly Parameterized Hydrological Model? A Case Study Using SWAT, *Water Resour. Res.*, 58, e2021WR031532, <https://doi.org/10.1029/2021wr031532>, 2022.
- Rafei, V., Nejadhashemi, A. P., Mushtaq, S., Bailey, R. T., and An-Vo, D.-A.: An improved calibration technique to address high dimensionality and non-linearity in integrated groundwater and surface water models, *Environ. Modell. Softw.*, 149, 105312, <https://doi.org/10.1016/j.envsoft.2022.105312>, 2022.
- Razavi, S., Tolson, B. A., and Burn, D. H.: Review of surrogate modeling in water resources, *Water Resour. Res.*, 48, W07401, <https://doi.org/10.1029/2011wr011527>, 2012.
- Sanchez-Fernandez, M., de-Prado-Cumplido, M., Arenas-Garcia, J., and Perez-Cruz, F.: SVM multiregression for nonlinear channel estimation in multiple-input multiple-output systems, *IEEE T. Signal Proces.*, 52, 2298–2307, <https://doi.org/10.1109/tsp.2004.831028>, 2004.
- Sanchez-Vila, X., Donado, L. D., Guadagnini, A., and Carrera, J.: A solution for multicomponent reactive transport under equilibrium and kinetic reactions, *Water Resour. Res.*, 46, W07539, <https://doi.org/10.1029/2009wr008439>, 2010.
- Scharnagl, B., Vrugt, J. A., Vereecken, H., and Herbst, M.: Inverse modelling of in situ soil water dynamics: investigating the effect of different prior distributions of the soil hydraulic parameters, *Hydrol. Earth Syst. Sci.*, 15, 3043–3059, <https://doi.org/10.5194/hess-15-3043-2011>, 2011.
- Schneider-Zapp, K., Ippisch, O., and Roth, K.: Numerical study of the evaporation process and parameter estimation analysis of an evaporation experiment, *Hydrol. Earth Syst. Sci.*, 14, 765–781, <https://doi.org/10.5194/hess-14-765-2010>, 2010.
- Shen, C., Appling, A. P., Gentine, P., Bandai, T., Gupta, H., Tarkovsky, A., Baity-Jesi, M., Fenicia, F., Kifer, D., Li, L., Liu, X., Ren, W., Zheng, Y., Harman, C. J., Clark, M., Farthing, M., Feng, D., Kumar, P., Aboelyazeed, D., Rahmani, F., Song, Y., Beck, H. E., Bindas, T., Dwivedi, D., Fang, K., Höge, M., Rackauckas, C., Mohanty, B., Roy, T., Xu, C., and Lawson, K.: Differ-

- entiable modelling to unify machine learning and physical models for geosciences, *Nature Reviews Earth and Environment*, 4, 552–567, <https://doi.org/10.1038/s43017-023-00450-9>, 2023.
- Sorensen, J. P. R. and Butcher, A. S.: Water Level Monitoring Pressure Transducers – A Need for Industry-Wide Standards, *Groundwater Monitoring and Remediation*, 31, 56–62, <https://doi.org/10.1111/j.1745-6592.2011.01346.x>, 2011.
- Steeffel, C., Depaolo, D., and Lichtner, P.: Reactive transport modeling: An essential tool and a new research approach for the earth sciences, *Earth Planet. Sc. Lett.*, 240, 539–558, <https://doi.org/10.1016/j.epsl.2005.09.017>, 2005.
- Sternagel, A., Loritz, R., Klaus, J., Berkowitz, B., and Zehe, E.: Simulation of reactive solute transport in the critical zone: a Lagrangian model for transient flow and preferential transport, *Hydrol. Earth Syst. Sci.*, 25, 1483–1508, <https://doi.org/10.5194/hess-25-1483-2021>, 2021.
- Storn, R. and Price, K.: Differential Evolution – A Simple and Efficient Heuristic for global Optimization over Continuous Spaces, *J. Global Optim.*, 11, 341–359, <https://doi.org/10.1023/A:1008202821328>, 1997.
- Sun, A. Y.: Discovering state-parameter mappings in subsurface models using generative adversarial networks, *Geophys. Res. Lett.*, 45, 11137–11146, <https://doi.org/10.1029/2018gl080404>, 2018.
- Sun, N.: Inverse problems in groundwater modeling, *Springer Science and Business Media*, 338 pp., <https://doi.org/10.1007/978-94-017-1970-4>, 2013.
- Tran, H.-N., Phan, G. T. T., Do, Q. B., and Tran, V.-P.: Comparative evaluation of the performance of improved genetic algorithms and differential evolution for in-core fuel management of a research reactor, *Nucl. Eng. Des.*, 398, 111953, <https://doi.org/10.1016/j.nucengdes.2022.111953>, 2022.
- Travaš, V., Zaharija, L., Stipanić, D., and Družeta, S.: Estimation of hydraulic conductivity functions in karst regions by particle swarm optimization with application to Lake Vrana, Croatia, *Hydrol. Earth Syst. Sci.*, 27, 1343–1359, <https://doi.org/10.5194/hess-27-1343-2023>, 2023.
- Tsai, F. T. C., Sun, N., and Yeh, W. W. G.: Global-local optimization for parameter structure identification in three-dimensional groundwater modeling, *Water Resour. Res.*, 39, 1043, <https://doi.org/10.1029/2001wr001135>, 2003.
- Tuia, D., Verrelst, J., Alonso, L., Perez-Cruz, F., and Camps-Valls, G.: Multioutput support vector regression for remote sensing biophysical parameter estimation, *IEEE Geosci. Remote S.*, 8, 804–808, <https://doi.org/10.1109/lgrs.2011.2109934>, 2011.
- Vrugt, J. A.: Markov chain Monte Carlo simulation using the DREAM software package: Theory, concepts, and MATLAB implementation, *Environ. Modell. Softw.*, 75, 273–316, <https://doi.org/10.1016/j.envsoft.2015.08.013>, 2016.
- Wang, G. S. and Chen, S. L.: Evaluation of a soil greenhouse gas emission model based on Bayesian inference and MCMC: Parameter identifiability and equifinality, *Ecol. Model.*, 253, 107–116, <https://doi.org/10.1016/j.ecolmodel.2012.09.011>, 2013.
- Wang, N., Chang, H., and Zhang, D.: Deep-learning-based inverse modeling approaches: A subsurface flow example, *J. Geophys. Res.-Sol. Ea.*, 126, 2020JB020549, <https://doi.org/10.1029/2020jb020549>, 2021.
- Wang, Y., Fang, Z., and Hong, H.: Comparison of convolutional neural networks for landslide susceptibility mapping in Yanshan County, China, *Sci. Total Environ.*, 666, 975–993, <https://doi.org/10.1016/j.scitotenv.2019.02.263>, 2019.
- Xia, C.-A., Luo, X., Hu, B. X., Riva, M., and Guadagnini, A.: Data assimilation with multiple types of observation boreholes via the ensemble Kalman filter embedded within stochastic moment equations, *Hydrol. Earth Syst. Sci.*, 25, 1689–1709, <https://doi.org/10.5194/hess-25-1689-2021>, 2021.
- Xiao, C., Deng, Y., and Wang, G.: Deep-Learning-Based Adjoint State Method: Methodology and Preliminary Application to Inverse Modeling, *Water Resour. Res.*, 57, 2020WR027400, <https://doi.org/10.1029/2020wr027400>, 2021.
- Xu, T., Spycher, N., Sonnenthal, E., Zhang, G., Zheng, L., and Pruess, K.: TOUGHREACT Version 2.0: A simulator for subsurface reactive transport under non-isothermal multiphase flow conditions, *Comput. Geosci.*, 37, 763–774, <https://doi.org/10.1016/j.cageo.2010.10.007>, 2011.
- Xu, Z., Serata, R., Wainwright, H., Denham, M., Molins, S., Gonzalez-Raymat, H., Lipnikov, K., Moulton, J. D., and Eddy-Dilek, C.: Reactive transport modeling for supporting climate resilience at groundwater contamination sites, *Hydrol. Earth Syst. Sci.*, 26, 755–773, <https://doi.org/10.5194/hess-26-755-2022>, 2022.
- Yan, Z., Ran, J., Xiao, Y., Xu, Z., Wu, H., Deng, X. L., Du, L., and Zhong, M.: The Temporal Improvement of Earth's Mass Transport Estimated by Coupling GRACE-FO With a Chinese Polar Gravity Satellite Mission, *J. Geophys. Res.-Sol. Ea.*, 128, e2023JB027157, <https://doi.org/10.1029/2023jb027157>, 2023.
- Yang, X., Chen, X., and Smith, M. M.: Deep learning inversion of gravity data for detection of CO₂ plumes in overlying aquifers, *J. Appl. Geophys.*, 196, 104507, <https://doi.org/10.1016/j.jappgeo.2021.104507>, 2022.
- Yeh, W. W.-G.: Review of Parameter Identification Procedures in Groundwater Hydrology: The Inverse Problem, *Water Resour. Res.*, 22, 95–108, <https://doi.org/10.1029/WR022i002p00095>, 1986.
- Yeung, C., Tsai, J.-M., King, B., Pham, B., Ho, D., Liang, J., Knight, M. W., and Raman, A. P.: Multiplexed supercell metasurface design and optimization with tandem residual networks, *Nanophotonics*, 10, 1133–1143, <https://doi.org/10.1515/nanoph-2020-0549>, 2021.
- Yu, S. and Ma, J.: Deep Learning for Geophysics: Current and Future Trends, *Rev. Geophys.*, 59, e2021RG000742, <https://doi.org/10.1029/2021rg000742>, 2021.
- Zhan, C., Dai, Z., Soltanian, M. R., and Zhang, X.: Stage-wise stochastic deep learning inversion framework for subsurface sedimentary structure identification, *Geophys. Res. Lett.*, 49, 2021GL095823, <https://doi.org/10.1029/2021gl095823>, 2021.
- Zhan, C., Dai, Z., Yang, Z., Zhang, X., Ma, Z., Thanh, H. V., and Soltanian, M. R.: Subsurface sedimentary structure identification using deep learning: A review, *Earth-Sci. Rev.*, 239, 104370, <https://doi.org/10.1016/j.earscirev.2023.104370>, 2023.
- Zhang, D. X. and Lu, Z. M.: An efficient, high-order perturbation approach for flow in random porous media via Karhunen–Loève and polynomial expansions, *J. Comput. Phys.*, 194, 773–794, <https://doi.org/10.1016/j.jcp.2003.09.015>, 2004.
- Zhang, J., Zeng, L., Chen, C., Chen, D., and Wu, L.: Efficient Bayesian experimental design for contaminant source identification, *Water Resour. Res.*, 51, 576–598, <https://doi.org/10.1002/2014wr015740>, 2015.

- Zhang, J., Lin, G., Li, W., Wu, L., and Zeng, L.: An iterative local updating ensemble smoother for estimation and uncertainty assessment of hydrologic model parameters with multimodal distributions, *Water Resour. Res.*, 54, 1716–1733, <https://doi.org/10.1002/2017wr020906>, 2018.
- Zhang, J., Cao, C., Nan, T., Ju, L., Zhou, H., and Zeng, L.: A Novel Deep Learning Approach for Data Assimilation of Complex Hydrological Systems, *Water Resour. Res.*, 60, e2023WR035389, <https://doi.org/10.1029/2023WR035389>, 2024.
- Zheng, L. and Samper, J.: Formulation of the inverse problem of non-isothermal multiphase flow and reactive transport in porous media, in: *Developments in Water Science*, edited by: Miller, C. T. and Pinder, G. F., Elsevier, 1317–1327, [https://doi.org/10.1016/S0167-5648\(04\)80146-1](https://doi.org/10.1016/S0167-5648(04)80146-1), 2004.
- Zhou, H., Gómez-Hernández, J. J., and Li, L.: Inverse methods in hydrogeology: Evolution and recent trends, *Adv. Water Resour.*, 63, 22–37, <https://doi.org/10.1016/j.advwatres.2013.10.014>, 2014.