Enhancing Inverse Modeling in Groundwater Systems through Machine Learning: A Comprehensive Comparative Study

Junjun Chen[1,2], Zhenxue Dai[2,3], Shangxian Yin[4], Mingkun Zhang[5], Mohamad Reza Soltanian[6]

[1] National and Local Joint Engineering Laboratory of Internet Application Technology on Mine, China University of Mining and Technology, Xuzhou, 221008, China
[2] College of Construction Engineering, Jilin University, Changchun, 130026, China
[3] School of Environmental and Municipal Engineering, Qingdao University of Technology, Qingdao, 273400, China
[4] College of Safety Engineering, North China Institute of Science and Technology, Langfang, 065201, China
[5] Shandong Rui Yi technology development Co., Ltd., Jinan, 250000, China
[6] Departments of Geosciences and Environmental Engineering, University of Cincinnati, OH, 45220, USA

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

Abstract. ~~Machine learning has significantly improved inverse modeling for groundwater systems. One promising development is the tandem neural network architecture (TNNA), which integrates surrogate modeling and reverse mapping for efficient forward simulations and data assimilation. Although TNNA has shown success in groundwater inverse modeling, its application scenarios remain limited, and its advantages over conventional methods have not been fully explored. This paper aims to address these gaps by comparing the TNNA method with four conventional metaheuristic algorithms: Particle Swarm Optimization, Genetic Algorithm, Simulated Annealing, and Differential Evolution. Two synthetic solute transport numerical cases are designed, with aquifer parameters characterized by low- and high-dimensional scenarios, respectively. The surrogate model is constructed using a deep residual convolutional neural network (ResNet), selected based on a comparative evaluation against three other popular machine learning models. Inversion performance is evaluated based on the accuracy of calibrated hydraulic heads, solute concentrations, and parameter estimation errors. The results demonstrate that the TNNA algorithm yields more reliable inversion results and significantly reduces computational burden across both low- and high-dimensional cases, effectively balancing exploration and exploitation in global optimization. This study highlights the significant advantages of machine learning in advancing groundwater system inversions.~~

Machine learning has significantly advanced inverse modeling in groundwater systems. The tandem neural network architecture (TNNA) represents a novel approach for estimating uncertain parameters by constructing 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 with 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~~s~~, 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~~s~~. Additionally, we evaluate algorithm performance under both low (1%)~~-~~ and high (10%) ~~-~~noise conditions for ~~each case~~the non-Gaussian random field scenario, which exhibits~~contains~~ the most complex parameter characteristics~~ among the three scenarios.~~. The results demonstrate that both the TNNA algorithm and the metaheuristic algorithms ~~can deliver~~achieve inversion results that satisfy the convergence accuracy within these machine learning- based inversion frameworks. Moreover, under the 10% high-noise condition in the non-Gaussian random field, the inversion results remain robust when sufficient constraints are ~~constructed~~imposed. Compared to metaheuristic approaches, ~~In comparison,~~ the TNNA method yields more reliable inversion results with significantly higher computational efficiency~~.~~, ~~underscores~~highlight~~s~~ing 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 current hydrogeological exploration techniques and budgetary constraints (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 (Zhou et al., 2014; Bandai and Ghezzehei, 2022; Abbas et al., 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). Among available algorithms, deterministic inversion methods are a significant category, where model parameters are estimated through maximizing the posterior distribution probability using optimization techniques (Tsai et al., 2003; 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 nonconvex 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. 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 nonlinearity (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 CNN surrogate model to combine with a hierarchical homogenization method to estimate effective permeability of digital rocks. More related studies can also be found in recent reviews (Yu and Ma, 2021; Luo et al., 2023b; Zhan et al., 2023).

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 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 CO_2 plumes to perform real-time inversion for geologic carbon sequestration. Another strategy is to train a reverse network within the 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 within 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 around 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 ~~for~~ to update parameters in each iteration~~parameter updates,~~. I~~i~~n contrast ~~to~~, conventional metaheuristic algorithms ~~that~~ typically necessitate multiple forward simulations. Despite~~The advantage of the TNNA algorithm is that it requires only one forward simulation~~

per parameter update, whereas conventional metaheuristic algorithms necessitate multiple forward simulations. the innovation of this approach, its applicability in more general groundwater numerical scenarios and its performance compared to conventional metaheuristic algorithms remain uncertain. Despite this approach is innovative, the application of TNNA is primarily limited to low-dimensional parameter settings, leaving its advantages over conventional optimization algorithms uncertain. This study designs three case scenarios with different heterogeneity characteristics to conduct a comparisoncompare the performance study betweenof the TNNA algorithm with four conventional metaheuristic algorithms. In scenario 1, focuswiththe domain is was divided into a finite number of homogeneous zones. The other two scenarios considerfocus on high-dimensional parameter fields based on the spatial variability of the aquifer. These two scenarios are usefulessential for revealing the dynamic behaviors of the groundwater system at the discrete grid scale. BasedDepending on the differences in the spatial variability of the aquifer structure, the two high-dimensional numerical scenarios characterize the heterogeneity of aquifer parameters using a Gaussian random field (i.e., Scenario 2) and a non-Gaussian random field (i.e., Scenario 3), respectively. Specifically, athe Gaussian random field is suitabledfor aquifers with a single lithofacies and relatively uniform physical structures, where the parameter values transition smoothly inacross space. In contrast, athe non-Gaussian random field typically accounts for the existence of a nugget effect in the aquifer structure, such as when the aquiferit contains two or moremultiple lithofacies with differingvarying hydraulic properties (Mariethoz and Caers, 2014). When conductingFor comparative researchstudy of on the above three scenarios, surrogate models will be used to accelerate forward simulation. Additionally, dimensionality reduction techniques are necessary for the two high-dimensional scenarios to reduce the parameter size and mitigate the curse of dimensionalityAdditionally, for the two high-dimensional model scenarios, dimensionality reduction methods are required to reduce inversion parameter size and avoid the curse of dimensionality. Specifically, the Karhunen-Loève expansion (KLE) will be applied for dimensionality reduction in Scenario 2 (Gaussian random field), while a generative machine learning method will be used for Scenario 3 (non-Gaussian random field). and a generative machine learning method will be used for dimensionality reduction in the Gaussian random field of Scenario 2 and the non-Gaussian random field of Scenario 3, respectively. The abovese comparisons are beneficial forwill help clarifying the applicability of the TNNA algorithm in thefor inverse estimation of non-homogeneous parameters with different spatial variability characteristics, as well as highlight its advantages over conventional metaheuristic optimization algorithms in these common model scenarios.This study aims to comprehensively compare the TNNA method with four conventional metaheuristic algorithms across both low-dimensional and high-dimensional parameter settings. Major contributions of this study include (1) systematically improving and applying the TNNA algorithm to high-dimensional heterogeneous aquifer parameter inversion, thereby filling a significant research gap in the field, and (2) quantitatively evaluating the advantages and limitations of the TNNA method in comparison to conventional deterministic inversion methods. The inversion accuracy of the TNNA algorithm depends on the predictive accuracy of the surrogate models. Based on a comparative analysis of four machine learning models, the most accurate DNN for forward simulation will be chosen to build the surrogate model. Additionally, this study also assesses the effectiveness of integrating KLE or generative machine learning with the TNNA algorithm for high-dimensional parameter inversion tasks. With advances advancements in artificial intelligence, the intended

~~research~~ anticipated outcomes ~~are anticipated~~of this study are expected to significantly enhance the development of appropriate frameworks for novel inversion algorithms, ~~providing fresh~~offering new insights for future ~~inversion~~ studies.

This paper is structured as follows: Section 2 introduces the fundamental principles of the methodology involved in this study. Section 3 provides detailed information on numerical models for ~~low- and high-dimensional~~the three case scenarios. Section 4 presents the results and discussions. Finally, Section 5 presents a summary and conclusions drawn from this research, along with recommendations for future investigations.

~~2 Methodology~~

2. Methodology

The inversion framework based on nonlinear optimization theory ~~When conducting inversion simulations based on nonlinear optimization theory, the design of the inversion framework~~ generally ~~involves~~consists of two ~~key~~ aspects: (1) constructing nonlinear 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 nonlinear optimization model in this paper is as follows:

~~The nonlinear inversion optimization model of this study is formulated as follows:~~

$$\begin{cases} \boldsymbol{m}^{*}=\min \sum_{i=1}^{N_{\mathrm{obs}}} \frac{1}{\sigma_i}\left[\mathbf{y}_{\mathrm{obs}}[i]-\hat{\mathbf{y}}[i]\right]^2 \\ \hat{\mathbf{y}}=\boldsymbol{F}_{HF}(\boldsymbol{m}) \approx \boldsymbol{F}_{Forward}(\boldsymbol{m},\boldsymbol{\theta}_{Forward}) \\ \boldsymbol{m}^{L} \leq \boldsymbol{m} \leq \boldsymbol{m}^{U} \end{cases} \tag{1}$$

~~W~~where $\mathbf{y}_{\mathrm{obs}} \in \mathbb{R}^{N_{\mathrm{obs}} \times 1}$ and $\hat{\mathbf{y}} \in \mathbb{R}^{N_{\mathrm{obs}} \times 1}$ represent the observed data vector and the corresponding model simulation output vector, respectively. $\mathbf{y}_{\mathrm{obs}}[i]$ and $\hat{\mathbf{y}}[i]$ refer to the ith element of the observed and simulated vectors, respectively, and σ_i denotes the standard deviation of the ith observed data. $\boldsymbol{m}^{L}$ and $\boldsymbol{m}^{U}$ are the vectors representing the lower and upper limit values of the model parameters, respectively. $\boldsymbol{F}_{HF}(\cdot)$ represent the high-fidelity numerical model.

In this study, three different inversion frameworks are developed ~~for comparing~~to compare the TNNA algorithms ~~and~~with four metaheuristic algorithms. In low-dimensional parameter scenario, a surrogate model $\boldsymbol{F}_{Forward}(\cdot)$ 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:

$$\boldsymbol{m}^{*}=\min \sum_{i=1}^{N_{\mathrm{obs}}} \frac{1}{\sigma_i}\left[\mathbf{y}_{\mathrm{obs}}[i]-\boldsymbol{F}_{Forward}(\boldsymbol{m})[i]\right]^2 \tag{2}$$

In high-dimensional parameter scenarios, in addition to employing surrogate models, dimensionality reduction algorithms are also integrated in inversion frameworks. Let $\boldsymbol{m}=\boldsymbol{G}(\boldsymbol{z})$ represent an operator for parameter dimensionality reduction. Specifically, the Karhunen-Loève Expansion (KLE) and the Octave Convolution Adversarial Autoencoder (OCAAE) are ~~respectively employed~~used for ~~the representation~~representing ~~of~~ Gaussian random fields and non-Gaussian random fields, respectively. The high-dimensional parameter $\boldsymbol{m}$ is optimized indirectly by estimating the low-dimensional vector $\boldsymbol{z}$:

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

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

~~$\boldsymbol{F}_{HF}(\cdot)$ and $\boldsymbol{F}_{Forward}(\cdot)$ represent the high-fidelity numerical model and the surrogate model, respectively. $\boldsymbol{\theta}_{Forward}$ represents the trainable parameters of the surrogate model~~ (Lykkegaard et al., 2021; Luo et al., 2023a)~~.~~

~~The uncertain model parameters $\boldsymbol{m}$ are estimated through optimization algorithms, subject to the constraints defined in the nonlinear optimization model. Specifically, the high fidelity forward model output $\boldsymbol{F}_{HF}(\boldsymbol{m})$ is approximated by the surrogate model $\boldsymbol{F}_{Forward}(\boldsymbol{m},\boldsymbol{\theta}_{Forward})$, ensuring computational efficiency. Detailed information about for surrogate modelling methods and optimization algorithms is provide in Sections 2.1 and 2.2, respectively.~~

2.1 Surrogate modeling methods

As shown in Figure 1, surrogate models are developed using a data-driven strategy. The process begins by sampling model parameters from prior distributions and calculating their responses using high-fidelity numerical models. A training dataset of paired model parameters and responses is then obtained, which is used to construct surrogate models via supervised machine learning. Specifically, four popular machine learning models are evaluated for surrogate modeling: multi-output support vector regression (MSVR), fully connected deep neural network (FC-DNN), convolutional neural network (LeNet), and deep residual convolutional neural network (ResNet). Each model represents a distinct period in the development of machine learning. Despite rapid advancements in artificial intelligence, these four methods remain broadly applicable for constructing surrogate models in most groundwater modeling scenarios.

The detailed principles of MSVR and the three deep learning-based methods are illustrated in the following two sub-sections. The surrogate model for inversion will be constructed using the most accurate among them. Before constructing surrogate models, the training datasets are normalized to ensure that the values for different simulation components fall within the range of [0,1].

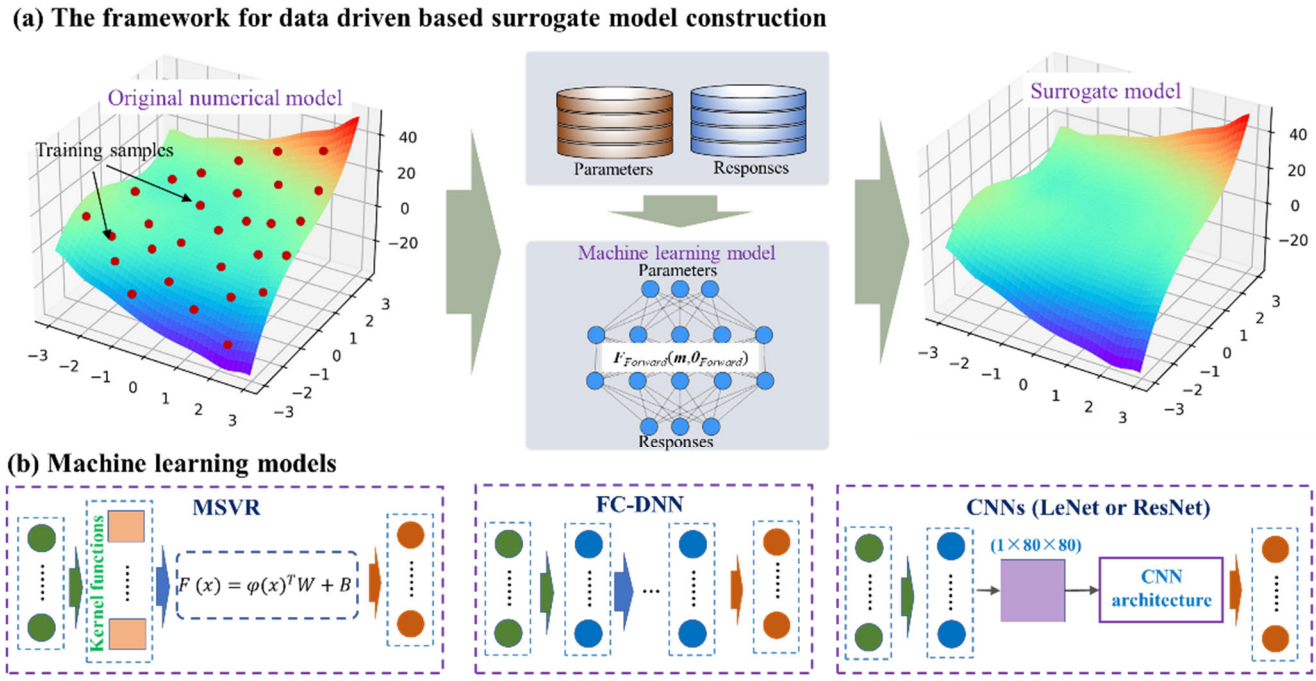

Figure 1. The framework for data-driven based surrogate model construction and the machine learning models employed.

2.1.1 MSVR

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:

$$y = F(x)=\varphi(x)^T W+B \tag{4}$$

where $\varphi(x)$ is a nonlinear regression function. W and B are regression coefficients determined by minimizing the structural risk, as outlined in Eq.(5)~(6):

$$W,B = argminL(W,B) = \frac{1}{2}\sum_{j=1}^{N_{obs}} \left\|w^j\right\|^2 + C\sum_{j=1}^{N_{train}} L(u_i) \tag{5}$$

where N_{train} is the sample size of the training dataset; C is a penalty parameter; and L(u) is a quadratic ε-insensitive loss function, expressed as:

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

where $u_i = \|e_i\| = \sqrt{e_i^T e_i}$; $e_i^T = y_i^T - \varphi^T(x_i)W - B^T$; ε in L(u) is the radius of the insensitive tube. For ε=0, this problem is equivalent to an independent regularized kernel least square 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 W and B are determined

using an iterative reweighted least squares (IRWLS) procedure, employing the quasi-Newton approach. During the IRWLS process, the term $L(u)$ in Eq.(35) 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 W and B are zero. Finally, the optimal values of W and 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: the penalty parameter C, the kernel function parameter 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. 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 $\boldsymbol{X}_{\text{input}} \in \mathbb{R}^{n\times 1}$ is the input vector and $\boldsymbol{X}_{\text{output}} \in \mathbb{R}^{m\times 1}$ is the output vector. The transformation in a fully connected layer is expressed as:

$$X_{\text{output}}=\sigma\left(W\times X_{\text{input}}+B\right) \tag{57}$$

where $\sigma(\cdot)$ is a non-linear active function; $W \in \mathbb{R}^{m\times n}$ is the weight matrix; and $B \in \mathbb{R}^{m\times 1}$ 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):

$$h_{u,v}^{q}(x_{u,v})=\sigma\left(\sum_{i=1}^{k_i^{'}}\sum_{j=1}^{k_j^{'}} w_{i,j}^{q} x_{u+i,v+j}+b\right) \tag{68}$$

where $x_{u,v}$ is the pixel value at position (u, v) of the input matrix; $h_{u,v}^{q}(x_{u,v})$ is the output feature $h_{u,v}^{q}(x_{u,v})$ calculated by employing the qth (q=1,…,N_{out}) convolutional kernel filter $\boldsymbol{w}^{q} \in \mathbb{R}^{k_i^{'}\times k_j^{'}}$. 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 improve the efficiency of model training (Chen et al., 2021).

The residual block is a fundamental component of residual networks (ResNets). It is designed to mitigate the vanishing and exploding gradients commonly encountered in the training of deep neural networks. In a residual block, an intermediate layer is designed to learn a residual mapping, $F(x)=H(x)-x$ (or $H(x)-G(x)$, where $G(x)$ represents another transformation of x).

Here, x is the input to the block. The output of the block is then computed as $F(x)+x$, which is intended to approximate $H(x)$. This design ensures that the output of the module at least replicates the input, thus avoiding overcoming the challenges posed by vanishing gradients. The mathematical formula of a residual block is expressed as follows:

$$y_l = F(x_l, W_l) + x_l \quad (7\text{~~9~~})$$

$$x_{l+1} = f(y_l) \quad (8\text{~~10~~})$$

where x_l and W_l are the input data and the connection weight matrix for the l-th residual block, respectively. $F(\bullet)$ is the residual function. Within this framework, the function $f(\cdot)$ is configured as an identity map, such that $x_{l+1}=y_l$. Then, the relationship between the L-th residual block in a deeper layer and the l-th residual block is expresses as follows (He et al., 2016):

$$x_L = x_l + \sum_{i=1}^{L-1} F(x_i, W_i) \quad (9\text{~~11~~})$$

According to the chain rule in derivatives, the gradient of the loss function ε with respect to x_l can be expressed as:

$$\frac{\partial \varepsilon}{\partial x_l} = \frac{\partial \varepsilon}{\partial x_L}\frac{\partial x_L}{\partial x_l} = \frac{\partial \varepsilon}{\partial x_L}\left(1 + \frac{\partial}{\partial x_l}\sum_{i=1}^{L-1} F(x_i, \omega_i)\right) \quad (10\text{~~12~~})$$

This formulation highlights two key properties of the residual network. First, the gradient does not vanish during network training processes because the term $\frac{\partial}{\partial x_l}\sum_{i=1}^{L-1} F(x_i, \omega_i)$ is never equal to -1. Second, the gradient of the deepest residual block $\frac{\partial \varepsilon}{\partial x_L}$ can directly affect all preceding layers, ensuring effectively transmission of gradients throughout the network (Chang et al., 2022).

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 Sigmoid, and the other hidden layers use Swish. The number of hidden layers n is determined by comparing the model prediction accuracy with different configurations, where n varies from 1 to 7. For the LeNet and ResNet models, the initial processing maps the input vector to a fixed matrix shaped 1×80×80 using a combination of a fully connected layer and a reshaped layer, as shown in Figure 1(b). 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 Figure S1 and Figure S2, respectively.

(2) DNN model training

The purpose of a surrogate model is to minimize the difference between the predicted outputs $\hat{y}_i = f_{DNN}(m_i, \theta_{DNN})$ and the numerical modeling outputs y_i. Consequently, the loss function is formulated with L1 norm constraints:

$$\theta_{DNN}=\text{argmin}\ \frac{1}{N}\sum_{i=1}^{N}|F_{DNN}(m_i,\theta_{DNN})-y_i| + \frac{w_d}{2}\theta_{DNN}^T\theta_{DNN} \quad (\text{~~11~~}13)$$

where w_d is the weight decay to avoid overfitting, referred to as the regularization coefficient. This study implemented the DNN models using PyTorch (https://pytorch.org/), a widely used machine learning framework. 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.

2.2 Dimensionality reduction methods

2.2.1 Karhunen-Loève Expansion for Gaussian random field

Let $G(\boldsymbol{s})\sim N(\boldsymbol{m}, \boldsymbol{C}(\cdot,\cdot))$ represent a Gaussian random field, where $\boldsymbol{m}$ denotes the mean of the random field, and $\boldsymbol{C}(\cdot,\cdot)$ represents the ~~covariance~~ exponential covariance function between two arbitrary ~~two~~ spatial points $\boldsymbol{s}=(s_x,s_y)$ and $\boldsymbol{s}'=(s_x',s_y')$. ~~When using L_2 based covariance exponential function,~~ The covariance function for these two spatial locations is given by:

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

where σ_G^2 is the variance, 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 ~~using~~through the ~~the Gaussian random field can be expressed using the~~ Karhunen-Loève Expansion (KLE) as follows:

$$\boldsymbol{G}(\boldsymbol{s})=\boldsymbol{m}(\boldsymbol{s})+\sum_{i=1}^{\infty}\xi_i\sqrt{\lambda_i}\phi_i(\boldsymbol{s}) \quad (\text{~~12~~}15)$$

where ξ_i represents a random variable following a Gaussian distribution of $\xi_i\sim N(0,1)$, also known as a KL term; $\phi_i(\boldsymbol{s})$ and λ_i denote the eigenfunction and eigenvalue, respectively. For discretized numerical models, the index i takes values from 1 to n, which represents the number of discrete grid points (i.e., in Eq.(15), ∞ is replaced by n). Dimensionality reduction via KLE is achieved through a truncated expansion. For example, if the dimensionality of the reduced parameter space is ~~n~~ n', the first ~~n~~ n' KL terms corresponding to the largest λ_i are used to represent the reduced-dimensional parameters (Zhang and Lu, 2004).

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,~~using~~ leveraging~~utilizing~~ Octave Convolution Neural Networks (Zhan et al., 2021). ~~The OCAAE~~It consists of three main components: an encoder, a decoder, and a discriminator.

The encoder maps high-dimensional parameter fields X to a low-dimensional latent vector space $z \sim q(z)$ ~~z~~. Specifically, the encoder outputs two low-dimensional vectors: the mean vector $\boldsymbol{\mu}$ and the log-variance vector $\ln(\boldsymbol{\sigma}^2)$ of the latent vector $\boldsymbol{z}$. Then, a vector $\boldsymbol{z}'$ is randomly drawn from a standard normal distribution $\boldsymbol{N}(\boldsymbol{0}, \boldsymbol{I})$, and the latent vector is produced as $\boldsymbol{z} = \boldsymbol{\mu} + \boldsymbol{\sigma} \times \boldsymbol{z}'$. The decoder reconstructs the high-dimensional parameter field $\tilde{\boldsymbol{X}}$ by taking the latent vector $\boldsymbol{z}$ as input. The discriminator enforces adversarial training, ensuring that the encoded latent vector distribution $z \sim q(z)$ approximates a prior Gaussian distribution $z \sim p(z)$. It receives input from the latent vectors generated by the encoder $z \sim q(z)$ or from the prior distribution $z \sim p(z)$, and discriminates which distribution the input latent vector originates from~~(Mo et al., 2020)*Zhan, 2021 #2236$~~.

~~between which distribution the input latent vector originates from.~~

~~,where d 表示降维后的维度。The decoder reconstructs the high-dimensional parameter field $\tilde{X}$ from the latent vector z:.The discriminator is designed for adversarial training, enforcing the distribution of the encoded latent vector $z \sim q(z)$ to approximate a prior Gaussian distribution $z \sim \ln \boldsymbol{N}(\boldsymbol{0}, \boldsymbol{I})$.~~ This adversarial framework ~~improves~~enhances the generative ~~capabilities~~capability and ensures smooth ~~interpolation~~transitions between different field realizations ~~of the channel field~~.

In the adversarial autoencoder method, the encoder $G(\cdot)$ (which also acts as the generator of the adversarial network), decoder, and discriminator $D(\cdot)$ are trained jointly in two phases ~~for~~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:

$$\mathcal{L}_{ED} = \frac{1}{N}\sum_{i=1}^{N}\left\|X_i - \tilde{X}_i\right\|_1 - w\left(\frac{1}{N}\sum_{i=1}^{N}\log\{D[G(X_i)]\}\right)\left\{\frac{1}{N}\sum_{i=1}^{N}\log\{D[G(X_i)]\}\right\} \quad (~~13~~16)$$

~~Here,~~where w is a weight ~~factor~~ balancing the reconstruction and adversarial losses (set to 0.01 in this study); ~~two losses and a value of $w = 0.01$ is used;~~ $\tilde{X}_i$ is the reconstruct~~ion~~ed ~~of~~ sample of X_i; and N is the number of training samples.

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

$$\mathcal{L}_{~~E~~D} = -\frac{1}{N}\sum_{i=1}^{N}\{\log[D(z_i)] + \log[1 - D[G(X_i)]\}~~\log~~ \quad (~~14~~17)$$

This loss function helps the discriminator distinguish between the latent vector z_i (from the true distribution $p(z)$) and the fake latent vector produced by the encoder $G(X_i)$.

The constraint loss functions in the adversarial autoencoder framework ensure that the reconstructed high-dimensional parameter field $\tilde{\boldsymbol{X}}$ closely matches the original field $\boldsymbol{X}$, while also making sure that the distribution of the low-dimensional latent vector $\boldsymbol{z}$ approximates a predefined standard normal distribution $p(z)$. After finishing the training process, it is possible to sample from the low-dimensional space of $p(z)$ and 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). ~~to distinguish the real sample from $p(z)$ (i.e. z_*) from the fake sample $G(z)$ (xi) produced by the generator.~~

2.~~2~~ 3 Optimization algorithms

2.~~2~~3.1 Metaheuristic algorithms

(1) Particle swarm optimization algorithm

Particle swarm optimization (PSO) is a population-based intelligent optimization algorithm inspired by the foraging behavior of birds (Eberhart and Kennedy, 1995). It is realized through the following steps:

Step 1: Initialize a population with n particles of a m-dimensional space $X=(X_1,X_2,\ldots,X_n)$. For an arbitrary particle (i), denote its position, velocity and best position at the kth iteration as $X_i^k=(x_{i1}^k,\ldots,x_{im}^k)$, $V_i^k=(v_{i1}^k,\ldots,v_{im}^k)$, and $P_i^k=(p_{i1}^k,\ldots,p_{im}^k)$, respectively.

Step 2: Calculate the best solution for each particle ($X_{pbest_i^k}$) according to Eq.(~~12~~18):

$$X_{pbest_i^k}=\begin{cases} X_{pbest_i^{k-1}}, & f(X_i^k)\geq f(X_{pbest_i^{k-1}}) \\ X_i^k, & f(X_i^k)<f(X_{pbest_i^{k-1}}) \end{cases} \quad (1\text{~~2~~}8)$$

where $f(\cdot)$ is the objective function, also known as the fitness function.

Step 3: Calculate the best position of the population ($X_{gbest_i^k}$) according to Eq.(~~13~~19).

$$X_{gbest_i^k}=\min\{f(X_1^k),\ldots,f(X_n^k)\} \quad (\text{~~13~~}19)$$

Step 4: Updated the velocity and position for each particle (i) according to Eq.(~~14~~20) and Eq.(~~15~~21):

$$V_i^{k+1}=w_iV_i^k+r_1c_1(X_{pbest_i^k}-X_i^k)+r_2c_2(X_{gbest_i^k}-X_i^k) \quad (\text{~~14~~}20)$$

$$X_i^{k+1}=X_i^k+V_i^{k+1} \quad (\text{~~15~~}21)$$

where c_1 and c_2 are learning parameters, generally taken as two equal non-negative constants and are set to 0.5 and 0.1 here; r_1 and r_2 are two random values within the range of [0, 1]; w_i is the inertia weight and set to 0.8 for this study.

(2) Genetic algorithm

Genetic algorithm (GA) is initially introduced by Holland John (1975). It draws inspiration from natural evolution and genetics, where individuals within a population are selected or eliminated based on their adaptability to the environment. The GA is realized through the following steps:

Step 1: Generate an initial population $X=(X_1,X_2,\ldots,X_n)$ randomly.

Step 2: Perform binary encoding on all individuals in the population X to obtain their respective binary symbol strings. These binary symbol strings are called chromosomes, and each value (“0” or “1”) on a symbol string is called a gene.

Step 3: Crossover: Perform crossover operations on randomly paired combinations of individuals in X. The essence of crossover is to exchange some values in the symbol strings of a pair of individuals.

Step 4: Mutation: Perform mutation operations on some random individuals in X by changing some values of their symbol strings.

Step 5: Selection: Perform selection operations based on the fitness values of each individual (X_i) to generate the next generation population. This step is realized through the roulette wheel selection method, where individuals with higher fitness values are more likely to be selected.

Step 6: Determine whether the current results satisfy the iteration termination condition. If not, return to step (2); otherwise, output the optimal individual in the current population as the final result.

(3) Simulated Annealing

The SA method is a Monte Carlo-based stochastic optimization algorithm proposed by Metropolis et al. (1953) and initially applied to combinational optimization problems by Kirkpatrick et al. (1983). The realization steps for SA method are as follows:

Step 1: Set the starting temperature as T_0 and draw an initial optimal solution as X_i.

Step 2: Generate a new solution X_j from the neighborhood of the current solution X_i.

Step 3: Calculate the objective function values $f(X_i)$ and $f(X_j)$. If $f(X_i) \geq f(X_j)$, then X_j becomes the current solution X_i; otherwise, X_j becomes the current solution X_i with a probability calculated as:

$$P(X_i \rightarrow X_j) = \exp\left(\frac{f(X_i)-f(X_j)}{a^t T_0}\right) \quad (1622)$$

where t is the current time and a is the temperature decay constant.

Step 4: Under the current temperature conditions, repeat steps (2) and (3) until reaching the predetermined number of internal iterations. Then, update the temperature and time as follows: set $t=t+1$ and $T_t=a_t T_0$, then proceed to the next step.

Step 5: Return to step (2) and continue the iteration according to the new temperature (T_t) and time (t) until the termination conditions are met. The iterations in this step can be considered outer iterations, distinguished from step (4).

(4) Differential evolution

DE is another evolutionary algorithm proposed by Storn and Price (1997). Similar to GA, DE also employs mutation, crossover and selection operators, but they update uncertain model parameters in different ways (Tran et al., 2022). The detailed steps for realizing DE are as follows:

Step 1: Generate the initial population $X=(X_1,X_2,\ldots,X_n)$ randomly.

Step 2: Perform encoding for each individual in X. The encoding method used in DE is floating-point real encoding, rather than binary encoding used in GA.

Step 3: Mutation**:** After completing individual encoding, DE performs mutation operations to generate new individuals according to Eq. (1723):

$$x_{perturbed}(g+1)=x_{rand_1}(g)+F_{DE}\times(x_{rand_2}(g)-x_{rand_3}(g)) \quad (1723)$$

where x_{rand_1}, x_{rand_2} and x_{rand_3} are randomly selected individuals among the candidate solutions of the current population and must be different from each other. F_{DE} is a scaling parameter within the range of [0,1], controlling differential variations. g represents the sequence number of iterations.

Step 4: Crossover: Perform crossover operations to generate the trial vector by combining the mutant and target vectors. The formula for this step is as follows:

$$u_j(g+1)=\begin{cases} x^j_{perturbed}(g+1) & \text{if } P_j \le CR \\ x_j(g) & \text{if } P_j > CR \end{cases} \quad (18\,24)$$

where P_j is a random number in the range of [0,1], CR is the crossover rate. If some variables of the trial vector have the same values, keep one of them and reset the others with random integer numbers in the range [1, D].

Step 5: Selection: Perform selection operations to determine whether the new generated trial vector $u_j(g+1)$ can survive the next generation, $x_j(g+1)$. Therefore, a candidate solution replaces the parent only if it has better objective function value.

Step 6: Return to step 3 until the convergence criteria are met.

2.2.3.2 TNNA algorithm

The TNNA algorithm aims to obtain a reverse network that maps the observation vector to model parameters, as shown in Eq. (19 25).

$$m=F_{Reverse}(\tilde{y}_{obs},\theta_{Reverse}) \quad (19\,25)$$

where $\theta_{Reverse}$ are the trainable parameters of $\boldsymbol{F}_{\text{Reverse}}$. The training of the reverse network is guided by the constraints of the nonlinear optimization model defined in Eq. (1). The loss function for training is expressed as follows:

$$\theta_{Reverse} = argmin \sum_{i=1}^{N_{obs}} \frac{1}{\sigma_i}\left[\hat{y}_{obs}[i] - F^i_{Forward}(F_{Reverse}(\hat{y}_{obs},\theta_{Reverse}),\theta_{Forward})\right]^2 \quad (20\,26)$$

The $\boldsymbol{F}_{\text{Reverse}}$ is also trained within the pytorch framework. 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 reverse network training processes, each iteration of updating the trainable parameters $\theta_{Forward}$ involves two steps: First, the vector $\hat{y}_{obs}$ is input into the reverse network F_{Reverse} to obtain the parameter prediction $\tilde{m}$. This predicted parameter $\tilde{m}$ is then input into the forward network F_{Forward} to generate the corresponding forward prediction results. Subsequently, the trainable parameters $\theta_{Reverse}$ of the reverse network are updated based on the error feedback from the loss function in Eq. (20 26) through DNN model training. This process demonstrates that F_{Reverse} and F_{Forward} are connected in a TNNA, wherein the forward simulation realization is executed once during each epoch to update the trainable parameters of $\theta_{Reverse}$. This is a marked difference from the four selected metaheuristic algorithms, which require numerous forward simulations for each update of estimated model parameters. Upon completion of F_{Reverse} training, the final optimal parameters are predicted by inputting observation data into F_{Reverse}. Further details on TNNA can be found in (Chen et al., 2021).

~~3. Case Study~~

3. Case Study

This study designed three synthetic cases based on previous research, covering different model scales 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 ~~both~~ small-scale scenarios, with simulation time measured in days. 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 at a sub-regional scale (approximately 1 km), with simulation time measured in years. It uses a hydraulic gradient of 0.00625, representing a smaller natural gradient typically found in plain 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 via the Karhunen-Loève expansion (KLE); and Case 3 uses a high-dimensional non-Gaussian binary random permeability field parameterized by a decoder trained ~~by~~with OCAAE.

After developing ~~the~~ numerical models for the three scenarios, we ~~will~~ first evaluate four surrogate models in Case 1, and the optimal surrogate model will be integrated into the inversion framework. Subsequently, hypothetical observation scenarios ~~will be~~are used to systematically compare the inversion accuracy of TNNA ~~with~~against four metaheuristic algorithms across the three cases. The observation data for model parameter inversion are generated by adding Gaussian noise perturbations to the numerical model simulation results. Specifically, observational noise is introduced by multiplying the simulated data by a random noise factor $\varepsilon \sim N(1, \sigma^2)$, 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 σ=0.01. Additionally, our previous study (Chen et al., 2021) examined the effects of higher observational noise levels (σ=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% (σ=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. ~~(Chen et al., 2021)~~The details of these three cases are provided in Sections 3.1~3.3.

~~We introduce two numerical cases to compare the TNNA algorithm with conventional metaheuristic algorithms: one with low-dimensional parameters and the other with high-dimensional parameters. Initially, four surrogate models will be assessed using the low-dimensional parameter case, and the model with the highest accuracy will be integrated into the inversion~~

framework. Based on hypothetical observation scenarios, we will compare the inversion performance of the TNNA method and the four metaheuristic algorithms in low- and high-dimensional cases.

3.1 Case 1: Low-dimensional zoned permeability field scenario

As shown in Figure 2, the numerical model for the low-dimensional scenario focuses on conservative solute transport within a zoned permeability field. The model domain is a two-dimensional rectangular area measuring 10m×20m. The left and right boundaries are 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 9.9×10^{-12} 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. The simulation area is uniformly discretized into 3,200 (40×80) meshes, and the simulation time is set to 20 days.

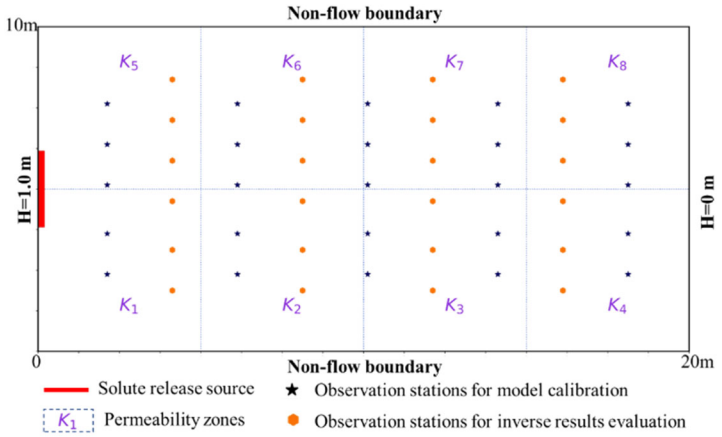

Figure 2. Flow domain of the solute transport model for the low-dimensional scenario.

According to these model conditions, there are nine uncertain model parameters to be estimated: eight permeability parameters (k_1 to k_8) and the source release concentration. As shown in Figure 2, these parameters will be estimated using the observation data of hydraulic heads and solute concentrations collected from 25 locations, denoted by black pentagrams. Additionally, observation data from another 24 locations, denoted by orange hexagons, will be used to validate 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 10m×10m, with impervious upper and lower boundaries and constant head boundaries at the left (1m) and right (0m) sides. The domain is discretized into 4,096 (64×64) grids. The log-permeability field follows a Gaussian distribution, and the permeability value of the i-th mesh is defined as follows:

$$k_i=\alpha_i k_{ref} \quad (2\sout{7})$$

where k_{ref} is the reference permeability, set to 2×10^{-13}m^2. The heterogeneity of k_i is controlled by the modifier α_i. The geostatistical parameters for this Gaussian field are as: $m = 0$, $\sigma_G^2 = 2$, and $\lambda_x = \lambda_y = 2.5$ m. Under this heterogeneous condition, ~~The modifier α for the logarithmic Gaussian random field satisfies the following formula:~~

~~$$\alpha(\boldsymbol{s}) = \exp\big(G(\boldsymbol{s})\big), G(\cdot) \sim N\big(m, C(\cdot,\cdot)\big), \quad (27)$$~~

~~where $m = 0$ is the constant mean and L_2-exponentiated quadratic covariance function for two arbitrary spatial locations, $\boldsymbol{s} = (s_x, s_y)$ and $\boldsymbol{s'} = (s_x', s_x')$:~~

~~$$C\left(\boldsymbol{s}, \boldsymbol{s}'\right) = \sigma_G^2 \exp\left(-\sqrt{\left(\frac{s_x - s_x'}{\lambda_x}\right)^2 - \left(\frac{s_y - s_y'}{\lambda_y}\right)^2}\right), \quad (28)$$~~

~~where $\sigma_G^2 = 2$ is the variance and $\lambda_x = \lambda_y = 2.5$ m are the correlation lengths along the x and y directions, respectively.~~

~~The Karhunen-Loève expansion (KLE) is utilized to parameterize the permeability field~~ (Zhang and Lu, 2004)~~. In this case,~~ 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 stable flow field and solute concentrations measured every two days over 40 days, starting from the 2nd day to the 40th day (day: $t = 2i$, i=1,…,20). To mitigate inversion errors arising from equifinality, actual permeability values at observed locations are included as regularization constraints. ~~The standard deviation of Gaussian noise for the normalized observations is set to 0.01.~~

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.

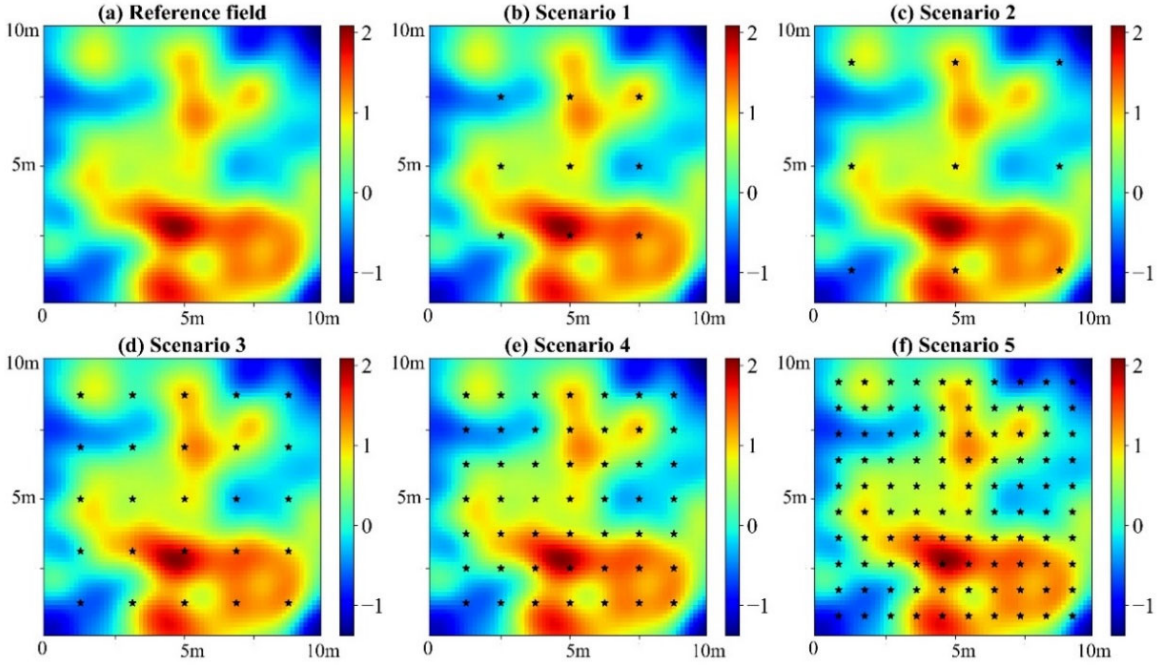

Figure 3. The reference log-permeability field and locations of observation stations for five scenarios. The observation stations are represented by black pentagrams.

3.3 Case 3: High-dimensional non-gaussian random permeability field scenario

~~(Bao et al., 2020)The numerical model for the high-dimensional non-gaussian scenario features a domain size of 800m×800m, with impervious upper and lower boundaries and constant head boundaries at the left (5m) and right (0m) sides. The domain is discretized into 6400 (80×80) grids. The permeability field is a channelized random field composed of two lithofacies, with permeability values of 5.9×10−13m2 and 2.71×10−12m2 for the two media, respectively. Dimensionality reduction of binary face permeability fields is achieved using an Octave Convolution based Adversarial Autoencoder (OCAAE). This generative machine learning method establishes a bidirectional mapping between the high-dimensional permeability field and a low-dimensional standard latent vector. Thus, the estimation of high-dimensional permeability field can be indirectly realized by constructing an inversion framework for the low-dimensional latent vector. In this case scenario, the dataset used for OCAAE training consists of 15000 samples generated based on multi-point geostatistical methods. The training image for geostatistical modeling is shown as Figure 1.~~

This case focuses on an estimation of a binary non-Gaussian permeability field. The numerical model features a domain size of 800m×800m, with impervious upper and lower boundaries and constant head boundaries at the left (5m) and right (0m) sides. The domain is discretized into 6400 (80×80) grids. The permeability field is a channelized random field composed of two lithofacies, with permeability values of $1.0\times10^{-13}m^2$ and $5.46\times10^{-12}m^2$ for the two media, respectively. The

reference field (Figure 4b) is generated from a training image (Figure 4a) using the direct sampling (DS) method proposed by Mariethoz et al. (2010)~~Mariethoz et al. (2010)~~. The contaminant release source is located at the ~~whole~~entire left boundary, with a concentration of 1 mol/L. 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 (t=2~24 years, with 2-year intervals). ~~Two different observational noise levels are considered in this case, with standardized noise standard deviations of 0.01 and 0.1, respectively.~~

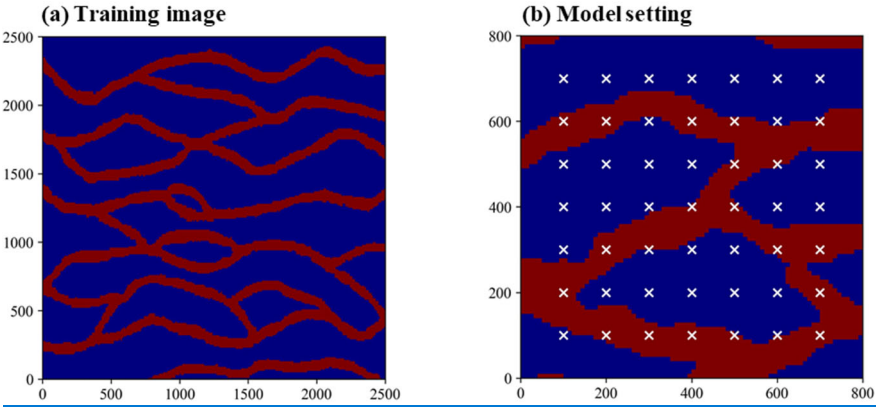

Figure 4. (a) The training image used to generate random realizations of permeability field; (b) The reference field of the synthetic case (white symbols indicate observation locations).

To achieve low-dimensional representation of permeability fields, a training dataset~~s~~ comprising 2000 stochastic realizations ~~are~~ is generated using multi-point statistics (MPS). Then, an Octave convolution-based Adversarial Autoencoder (OCAAE) is developed, where the decoder network learns a nonlinear mapping from 100-dimensional Gaussian latent vectors to 6400-dimensional binary non-Gaussian permeability fields. Thus, the non-Gaussian permeability field is indirectly reconstructed by estimating the 100-dimensional latent vector.

~~The contaminant release source is located at the whole left boundary, with a concentration of 1 mol/L. The simulation duration is 12 years, with observation data collected at six time points (t = 2, 4, 6, 8, 10, and 12 years).~~

4. Results and discussion

4.1 Surrogate model evaluations

Surrogate models were first compared using the low-dimensional parameter case. Four training datasets $\boldsymbol{D}_{train}=\{\boldsymbol{M}_{train},\boldsymbol{Y}_{train}\}$ with 200, 500, 1000, and 2000 samples (represented as $\boldsymbol{D}_{train\text{-}200}$, $\boldsymbol{D}_{train\text{-}500}$, $\boldsymbol{D}_{train\text{-}1000}$ and $\boldsymbol{D}_{train\text{-}2000}$, respectively) and a testing dataset $\boldsymbol{D}_{test}=\{\boldsymbol{M}_{test},\boldsymbol{Y}_{test}\}$ with 100 samples (represented as $\boldsymbol{D}_{test\text{-}100}$) are prepared. These datasets were generated using Latin hypercube sampling (LHS) and numerical simulations. The predictive accuracy of 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 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 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.

Figure ~~4~~ 5 and Figure ~~5~~ 6 display the RMSE and R^2 values of each surrogate model, and Figure S3~Figure S6 present the pairwise comparison results. The optimal values for C, σ, and ε in the MSVR method are provided in Table S1. Moreover, the optimal number of hidden layers in the FC-DNN for $\boldsymbol{D}_{train\text{-}200}$, $\boldsymbol{D}_{train\text{-}500}$, $\boldsymbol{D}_{train\text{-}1000}$ and $\boldsymbol{D}_{train\text{-}2000}$ are 2, 4, 3, and 3, respectively, as determined by the corresponding $RMSE_{\mathrm{All}}$ and R^2_{All} values in Table S2 and Table S3.

According to the performance criteria in Figure ~~4~~ 5 and Figure ~~5~~6, the prediction accuracy of each surrogate model significantly improves with an increasing number of training samples. Based on $RMSE_{\mathrm{All}}$ 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 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 the $RMSE_C$ value remains around 0.05, and the R^2_C 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 Figure S2(d)). When adopting CNN-based surrogate models (LeNet and ResNet), the prediction accuracy for solute concentrations significantly improves (see Figure ~~4~~5(b) and Figure ~~5~~6(b)). With training datasets of 2000 samples, LeNet and ResNet achieve RMSE values below 0.02 and R^2 values close to 1. It is worth noting that the ResNet performs well even with smaller sample sizes. For example, with 200 training samples, the $RMSE_C$ and R^2_C 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 Figure 45(b) and Figure 56(b)). As the number of training samples increases, the advantages of ResNet become more apparent. According to Figure S4(d), 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.

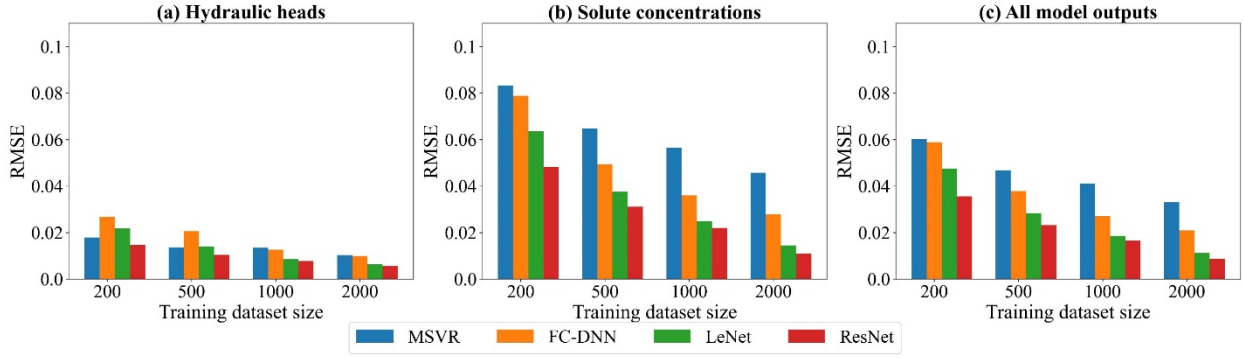

Figure 45. The RMSE results of surrogate model predictions. (a)~(c) are respectively the RMSE values of hydraulic heads, solute concentrations and all model outputs.

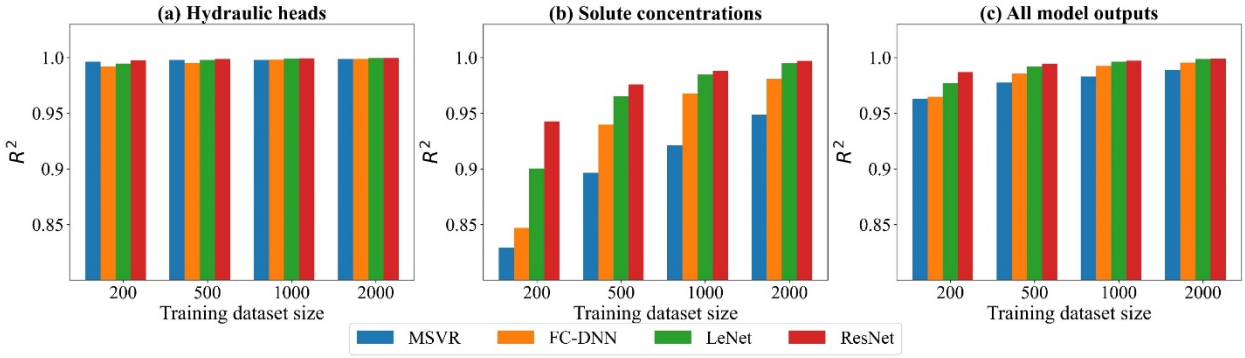

Figure 56. The R^2 results of surrogate model predictions. (a)~(c) are respectively the R^2 values of hydraulic heads, solute concentrations and all model outputs.

The comparison results of the surrogate models reflect a trend of enhanced robustness attributable to advancements in machine learning methodologies. Different machine learning approaches employ distinct strategies for achieving nonlinear mappings in developing surrogate models. Generally, deeper or larger models contain more trainable parameters, resulting in higher degrees of freedom to capture more robust nonlinear 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. The 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 the ResNet for the surrogate model construction of both Gaussian and non-Gaussian random field scenarios, with training and testing datasets consisting of 2000 and 500 samples, respectively. 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.

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

	RMSE			**R^2**		
	$RMSE_H$	$RMSE_C$	$RMSE_{All}$	R_H^2	R_C^2	R_{All}^2
~~Scenario~~ Gaussian Scenario-1	0.0108	0.0174	0.0172	0.9990	0.9980	0.9982
Gaussian Scenario-~~Scenario~~ 2	0.0102	0.0138	0.0136	0.9995	0.9989	0.9990
Gaussian Scenario-~~Scenario~~ 3	0.0120	0.0165	0.0163	0.9991	0.9981	0.9983
Gaussian Scenario-~~Scenario~~ 4	0.0123	0.0161	0.0159	0.9990	0.9984	0.9985
Gaussian Scenario-~~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

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 Gaussian random noise $\varepsilon \sim N(0,0.01)$. 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 ~~6~~ 7 presents the logarithmic average convergence curves of four metaheuristic algorithms and the TNNA algorithm throughout 100 parameter scenarios. Specifically, sub-figures (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 ~~7~~ 8 displays the comparison of calibration and validation between the simulation results and the observed values across all 100 parameter scenarios. Sub-figures (a) and (b) illustrate the comparative prediction fit at the 25 observation locations used for model calibration, whereas sub-figures (c) and (d) display the comparative prediction fit at the 24 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 ~~8~~ 9 illustrates the probability density curves of the estimation errors for nine model parameters across 100 parameter scenarios, with different colours representing the five optimization algorithms.

The results in Figure ~~6~~ 7 demonstrate that the TNNA algorithm achieves the best convergence accuracy, with its convergence logarithmic objective function value (i.e., 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 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 parameter optimization. For instance, the DE algorithm requires 32,000 forward model realizations (80×2×200) 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.

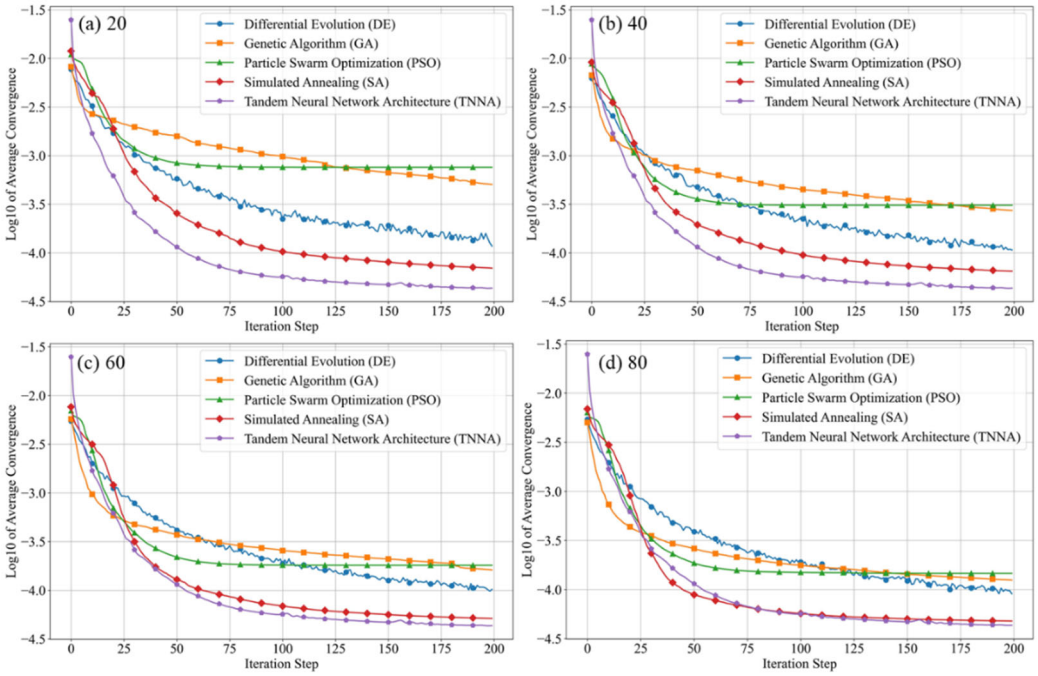

Figure ~~6~~7. Comparative convergence trends of five optimization algorithms (Markers indicate convergence values at every 10 steps to indicate convergence values; for TNNA, only the first 200 out of 1000 iterations are presented).

The results presented in Figure ~~7~~ 8 indicate that, among the five optimization algorithms, the TNNA algorithm achieves the smallest RMSE values and R^2 values closest to 1.0 for both hydraulic heads and solute concentration during model calibration and validation. Furthermore, the distribution of comparison points demonstrates that the calibrated and validated modeling results of the TNNA algorithm are more accurately matched with their actual values than 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 calibration and validation, 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 validation for hydraulic heads and solute concentrations. However, DE and SA demonstrate appropriate calibration accuracy only for one of the two simulation components. Overall, the TNNA algorithm provides more robust model calibration and validation results than the other four metaheuristic algorithms.

Figure ~~8~~ 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 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 algorithms 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 K4 and K6 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 avoid misleading conclusions.

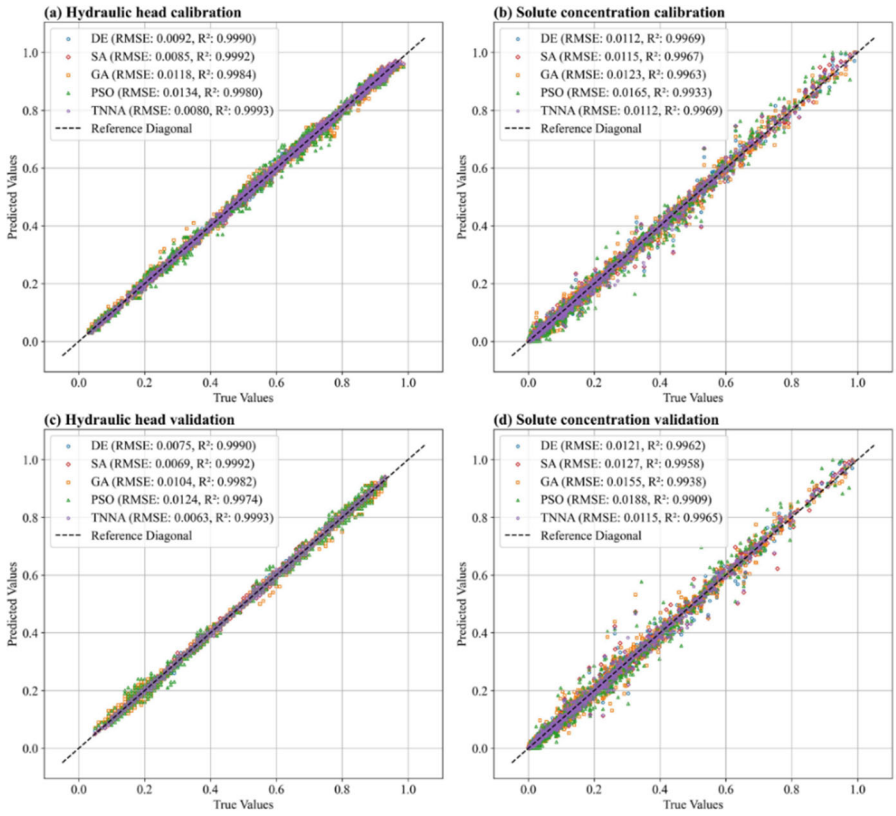

Figure ~~7~~8. Comparison of calibrated and validated model predictive accuracy for hydraulic heads and solute concentrations by the four metaheuristic algorithms and the TNNA method.

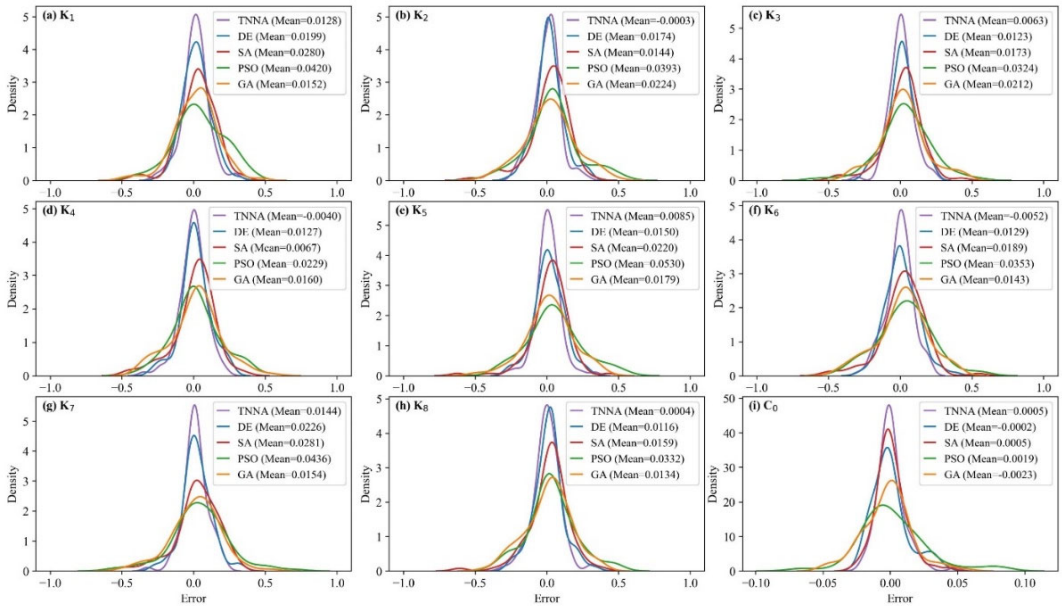

Figure 89. Probability density curves of estimation errors for nine model parameters using five optimization methods.

The above comparison results indicated 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 highly deterministic 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 ~~parameter~~ 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.

Figure ~~9~~ 10 and Figure ~~10~~ 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 Figure S7-S14. Figure ~~11~~ 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 (Scenarios 1-4). For Scenario 5, the accuracy of permeability estimations by each metaheuristic algorithm improves as the N_{PC} value increases (see Figure ~~9~~ 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 Figure ~~10~~ 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 Figure S7-S14). Figure ~~11~~ 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.

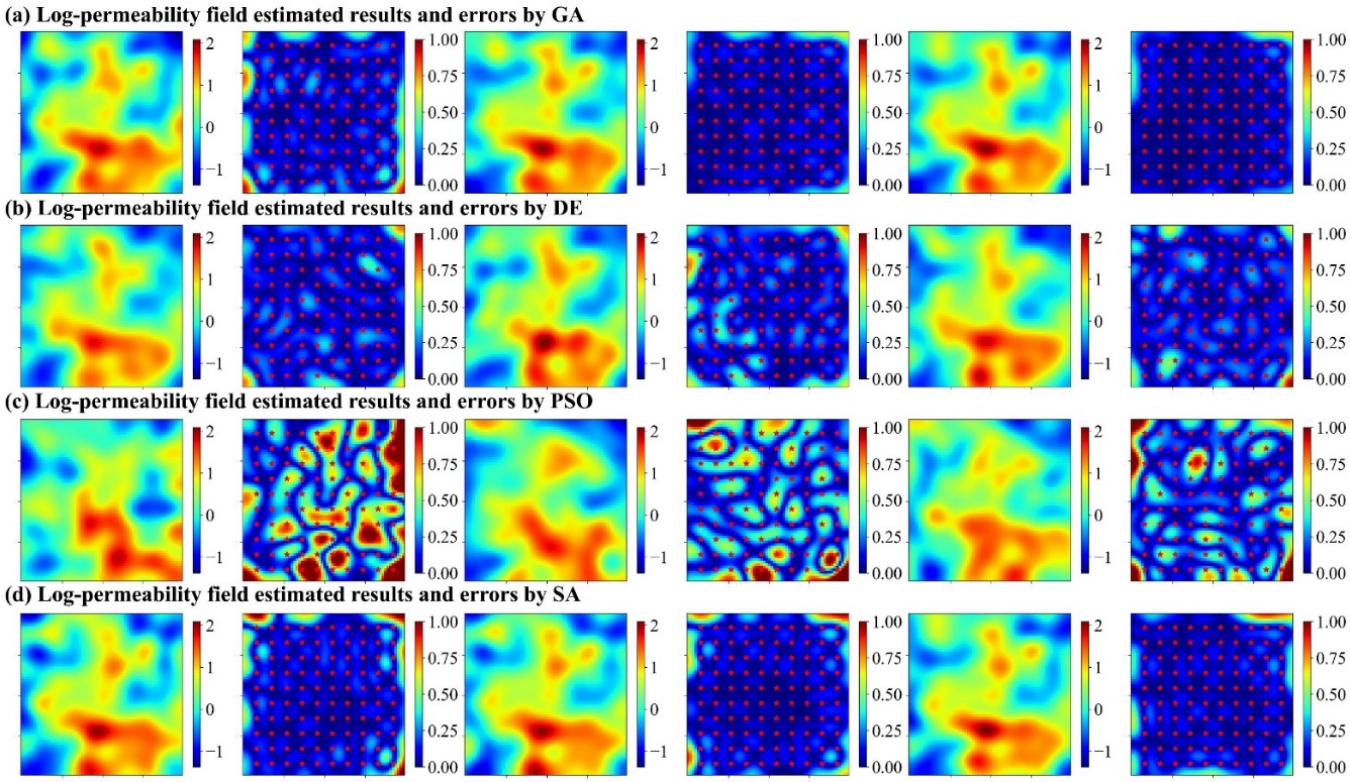

Figure ~~9~~10. Spatial distributions of log-permeability field estimation results (row 1, 3, and 5 for N_{PC}=100, 500, and 1000, respectively) and absolute errors (row 2, 4, and 6 for N_{PC}=100, 500, and 1000, respectively) for Scenario 5, achieved by four metaheuristic algorithms.

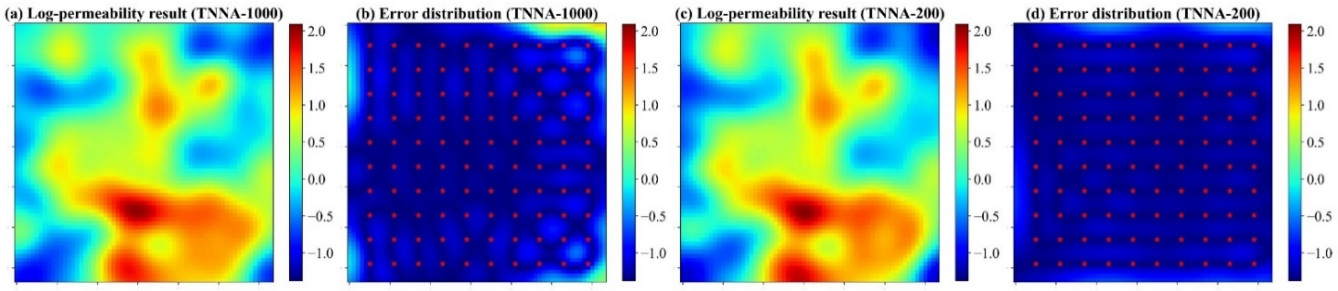

Figure ~~10~~11. Spatial distributions log-permeability field estimation results and absolute errors for Scenario 5, achieved by the TNNA.

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		

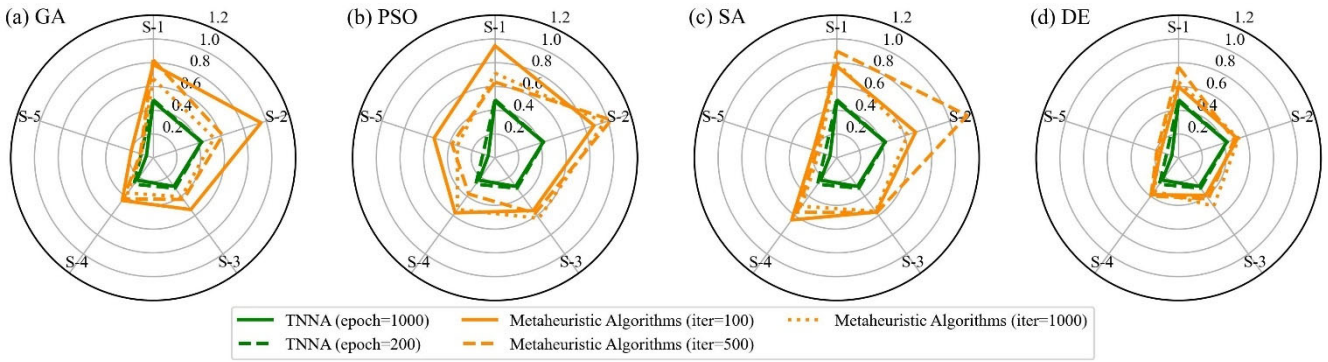

Figure ~~11~~12. Comparison of RMSE in estimating log-permeability fields using four metaheuristic algorithms and the TNNA algorithm across five scenarios (S-1 to S-5).

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 days, starting on the 2nd day with bi-daily recordings, using the permeability fields with the lowest RMSE values identified by each inversion method. Observation data from the 2nd 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 Figure ~~12~~13. Figure ~~13~~ 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=4th, 20th, and 52nd days). Results for the entire 60-day period are presented in Figure S15-S44.

According to Figure ~~13~~14(a), the calibrated simulation errors for hydraulic heads did not exceed 0.02 meters for the TNNA method and three of the four considered metaheuristic algorithms, except PSO method, which exhibited hydraulic head errors larger than 0.06 meters 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 (Figure ~~12~~13). The TNNA algorithm achieved a similar level of accuracy to GA in the calibrated numerical model predictions. Specifically, during the initial 10 days 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 (Figure ~~13~~14(b~c)). 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.

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}

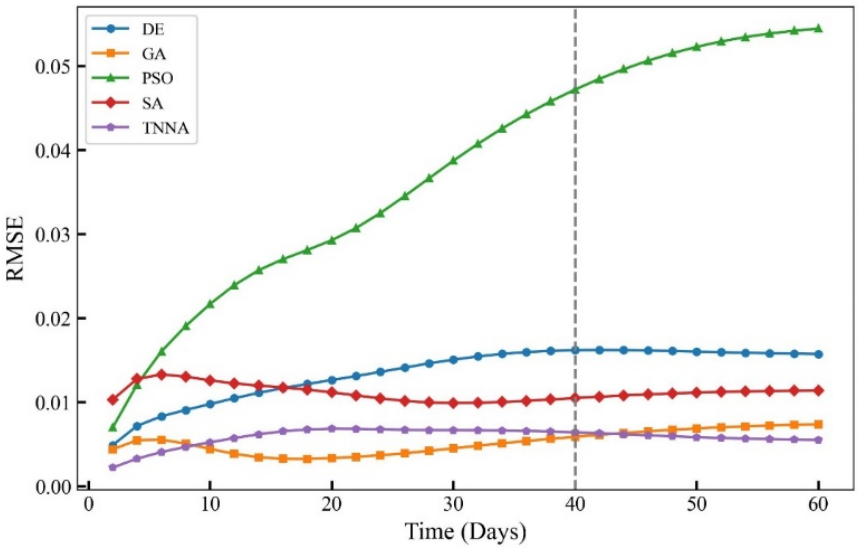

Figure ~~12~~13. RMSE values of calibrated solute concentrations over 60 days for the four metaheuristic algorithms and the TNNA algorithm.

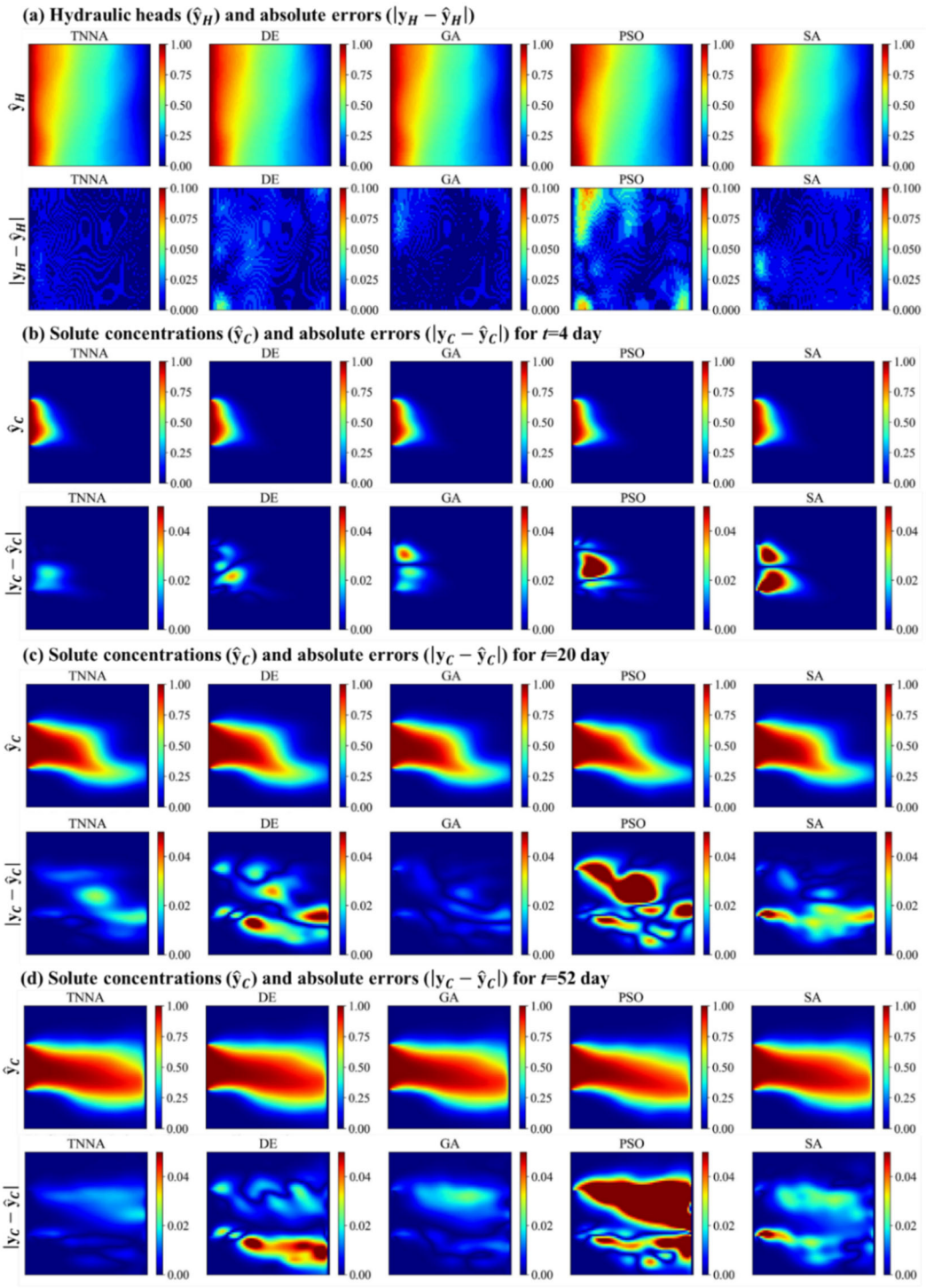

Figure ~~13~~14. Spatial distributions of calibrated numerical simulation results and absolute errors for hydraulic heads and solute concentrations at three dynamic times (t=4, 20, and 50 day) using the TNNA algorithm and four metaheuristic algorithms.

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 spendedt 100 times more forward model evaluations than the TNNA algorithm. Figure 14 and Figure 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). Figure 16(a) and Figure 17(a) present the comparison between calibrated simulations and hydraulic head observations, as well as solute concentration observations. Figure 16(b) and Figure 17(b) 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 Figure 1415-1516, 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 the increasing the observation noise level from 1% to 10% does not lead to 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. The iteration number and N_{PC} values for the four metaheuristic algorithms 」j

It can also be validated by the calibration simulations. Specifically, the pairwise scatter plots in Figure 1667(a) and Figure 1778(a) 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, Figures 16(b) and 17(b) demonstrate that the inversion based predictions remain highly consistent with the true permeability field. Furthermore, the predictions based on inversion results remain highly consistent with those of the true permeability field (Figure 17(b) and Figure 18(b)). The $RMSE_{All}$ and R^2_{All} values for the predictions beyond the observational period range from 0.018 to 0.044 and 0.962 to 0.994, respectively. This indicates that, even under relatively high Gaussian noise conditions, the nonlinear inversion framework used in this study can reliably reconstruct thisc non-Gauissian permeability field, ensuring high predictive accuracy. Nevertheless, it is important to note that althoughwhile the inversion accuracy under a 10% noise level remains comparable to that in the 1% noise scenarios, increasing observational noise inevitably raises the convergence value of the least-squares loss function. This trend is evident from the RMSE values in Figures 1667(a) and 1778(a). Moreover, since the observational noise here is assumed to follow a Gaussian distribution, more complex noise in real--world scenarios with more complex noise characteristics may further exacerbateincrease equifinality in the inversion results. In such cases, incorporating additional system information as regularization constraints is essential to enhance the robustness of the objective function and 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 level scenario, TNNA achieves an inversion convergence accuracy with an $RMSE_{All}$ of 0.021 and an R^2_{All} of 0.996 (Figure 17(a)). 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 (Figure 16~~7~~(a)). 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 (Figure 16~~7~~(b)). Even in high-noise scenarios, TNNA continues to exhibit an advantage over the four metaheuristic algorithms in both inversion convergence accuracy (Figure 18(a)) and predictive accuracy (Figure 18(b)). 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.

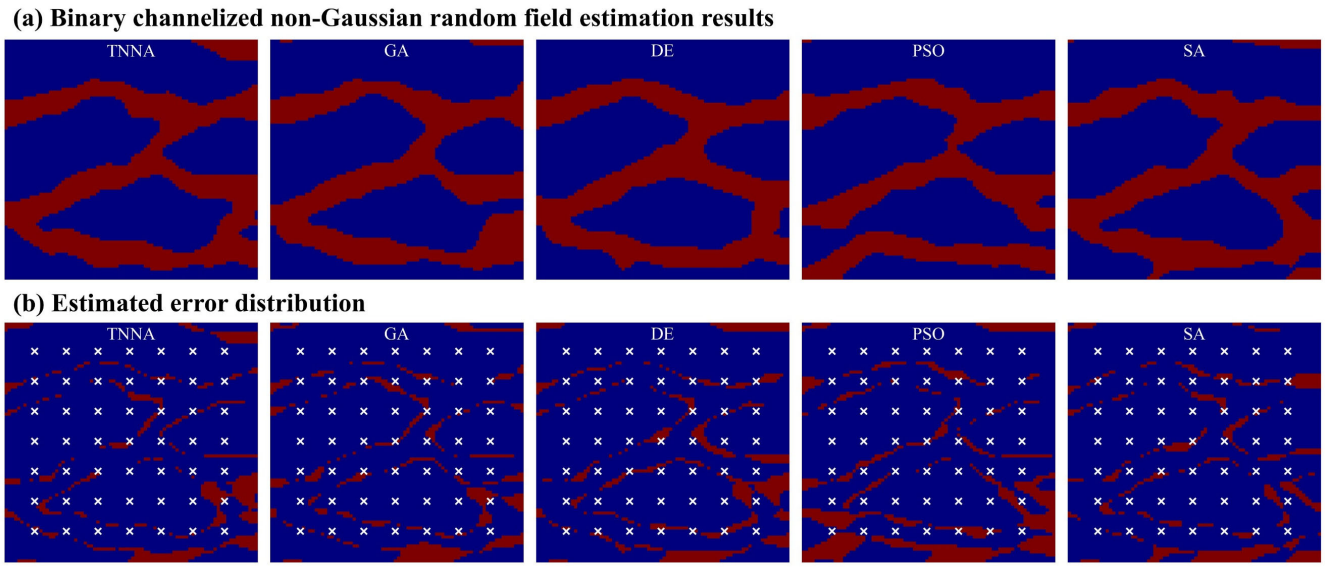

Figure 15. Reconstructed non-Gaussian binary channelized fields and their error distributions (1% observation noise)

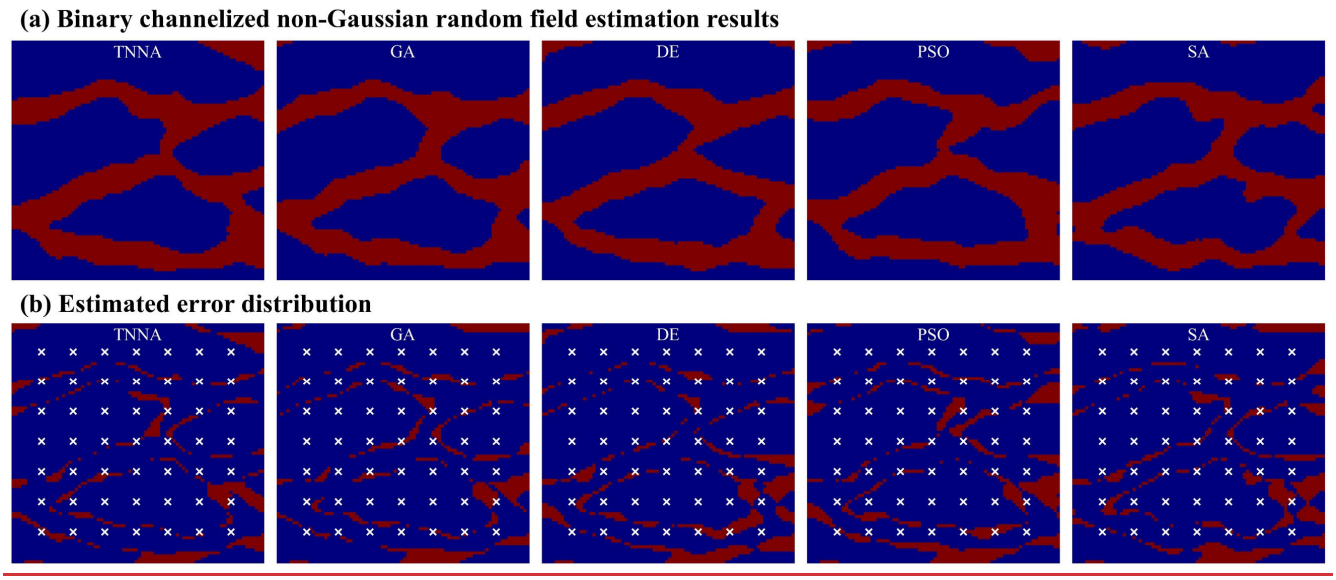

Figure 16. Reconstructed non-Gaussian binary channelized fields and their error distributions (10% observation noise)

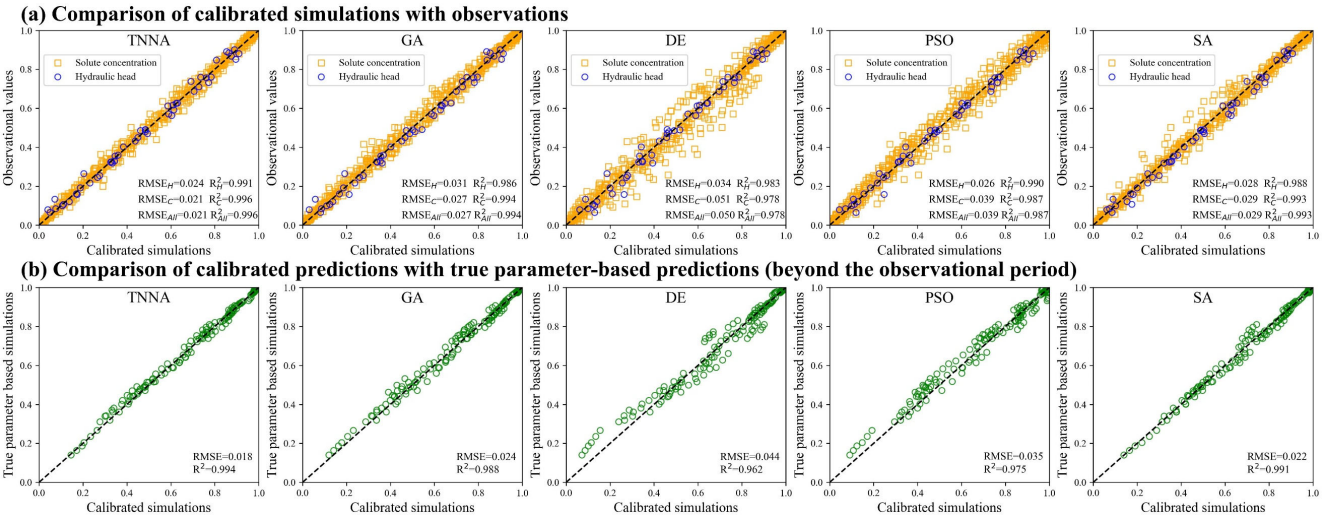

Figure 17. pair-wise comparison between the calibrated simulation results with the observational data (a); and the true parameter based predictions (1% observation noise).

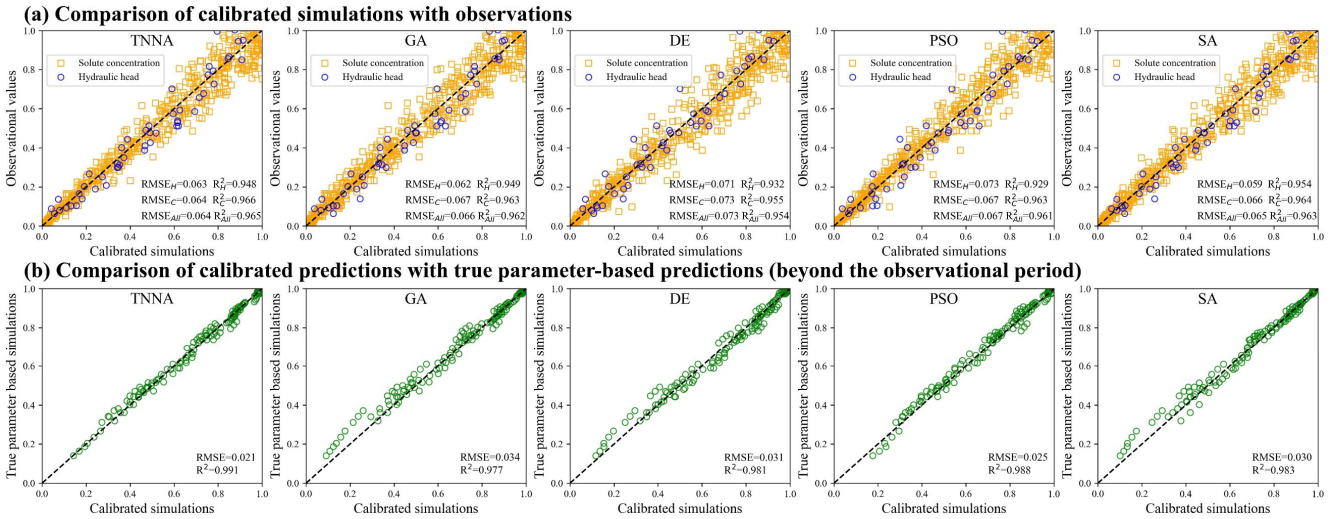

Figure 18. pair-wise comparison between the calibrated simulation results with the observational data (a); and the true parameter based predictions (10% observation noise).

4.3 ~~Insights from synthetic cases for practical research~~Parameter inversion method comparison results

This study validates 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 nonlinear 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 nonlinear optimization constraints can be effectively established for inversion tasks. Considering the complexities of real-world groundwater systems, three key aspects should be considered to extended for real-world applications: 1) Representation of 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 model process to better address complex observational noise scenarios and uncertainties in physical mechanisms. Detailed considerations for these issues are as follows:

~~This study extends the application scenarios of the TNNA algorithm and verifies its advantages in computational efficiency and inversion reliability under multiple heterogeneous conditions. However, the research primarily remains at the~~

~~algorithmic theoretical level. To extend the TNNA algorithm to practical applications, the following issues warrant further attention:~~

- Heterogeneity in aquifer parameter structures: This study developed a dimensionality-reduction framework using the OCAAE for high-dimensional parameter field inversion. While generative machine learning methods (including state-of-the-art variants) show potential for extending to complex non-Gaussian fields in practical scenarios, a critical challenge arises when heterogeneity exhibits ambiguous statistical features. For instance, in non-stationary stochastic aquifer systems, spatial variations across sampling windows may induce substantial discrepancies in geostatistical parameters~~. Therefore, the critical challenge in practical research is creating training datasets that fully represent parameter distributions using similar geological cases, or designing generator training rules based on geological principles to ensure consistent parameter field outputs~~ (Mariethoz and Caers, 2014). Therefore, the critical challenge in practical research is creating training datasets that fully represent parameter distributions using similar geological cases, or designing generator training rules based on geological principles to ensure consistent parameter field outputs.
- Monitoring network optimization: The inversion performance of the TNNA and four metaheuristic algorithms is evaluated based on a nonlinear 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 designs are also to eliminate other interferences affecting inversion results, ensuring that differences in inversion accuracy are primarily determined by the performance of the inversion algorithms themselves. However, the number and locations of monitoring stations are constrained by financial budgets. Thus, optimizing 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).
- ~~Multi-source data constraints~~Considering multi-source data and uncertainties in model processes:~~ based inverse modeling:~~ 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, some solute concentrations cannot be measured in situ, and unavoidable perturbations may be included during sample collection and laboratory analysis. Similarly, hydraulic head data~~ can also~~ may be ~~affected~~influenced by meteorological factors. Moreover, all observational data in this study are constrained by a ~~-~~single predetermined process model. However, i~~I~~f 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.~~, inversion simulations based on a single model may lead to substantial errors. Therefore, future work should integrate multi-source data (e.g., geophysical measurements or isotope data) and develop multi-process coupled models to establish more robust inversion frameworks.~~ Therefore, it is crucial to integrate multi-source data (e.g., geophysical measurements or isotope data) and 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.

~~This approach would mitigate the impact of errors from single data sources on final inversion results.~~

~~The model conditions of the two synthetic cases in this study are primarily based on previous studies, as well as large-scale sandbox~~ (Jose et al., 2004; Zhang et al., 2018; Mo et al., 2019)~~. It is worth noting that the domain sizes for field-scale groundwater models are commonly on the kilometre scale, which is significantly larger than 10m or 20m sizes used in this study. Moreover, the hydraulic gradient of groundwater in natural alluvial aquifers is generally below 0.01, while hydraulic gradients of 0.1 and 0.05 may occur when the underlying aquitard of the aquifer has a natural slope (Chai et al., 2024). Additionally, the monitoring stations in the synthetic cases of this study are densely distributed, whereas in practical studies, the number and locations of monitoring stations are constrained by financial budgets. They are often designed based on multi-objective optimization criteria, such as maximizing information and minimizing redundancy~ (Keum et al., 2018; Chen et al., 2022; Cao et al., 2025)~~. Therefore, a common concern that may arise is whether the results of this study can guide the selection of appropriate algorithms in practical research scenarios.~~

~~As summarized in the Introduction, the performance of an inversion algorithm depends on the degree of model nonlinearity and the complexity of parameter space. The mathematical model of this study consists of the groundwater flow continuity equation and the advection-dispersion equation. Within a fixed system mathematical model, variations in the average hydraulic gradient by an order of magnitude primarily affect flow velocity and solute transport rates, with little impact on the degree of nonlinearity in the system. Therefore, the inversion algorithms are expected to perform similarly in these real-world scenarios, provided that the system variables at field sites adhere to the same mathematical models and exhibit comparable parameter heterogeneity to that in this study. Additionally, monitoring strategies with densely distributed stations are 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)~~. Although such monitoring networks are unlikely to be used in field-scale groundwater studies, the effectiveness of the inversion results ultimately depends on the amount of effective information in the observational data. Furthermore, as optimized monitoring networks exclude redundant monitoring stations, dynamic responses from optimal monitoring stations typically exhibit higher sensitivity to model parameters, making it easier to achieve data assimilation goals for inversion algorithms. As a whole, the findings of this study are also applicable to research conducted at field scales with similar mathematical model conditions.~~

5. Summary and conclusions

This study systematically evaluates the performance of the 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 results demonstrate that TNNA consistently ~~Recent advancements in machine learning have significantly contributed to the development of inverse modeling. This study aims to compare the universality~~

~~and advantages of the novel TNNA algorithm with four popular metaheuristic algorithms (GA, PSO, DE, and SA) across various parameter dimensions in solute transport models. Surrogate models for these inversion methods were constructed using ResNet, which achieved the highest predictive accuracy for hydraulic heads and solute concentrations among four evaluated surrogate modeling methods (MSVR, FC-DNN, LeNet, and ResNet).~~

~~The inversion results indicate that the TNNA algorithm~~ outperforms the four conventional metaheuristic algorithms across the designed scenarios, covering both low-dimensional and high-dimensional cases. ~~in both high-dimensional and low-dimensional scenarios~~ It provides~~, providing~~ more accurate inversion results while significantly ~~reduces~~ reducing ~~the~~ computational ~~burden~~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 introduces 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.

This study demonstrates that achievements in machine learning can significantly enhance inversion results compared to conventional methods. Given that nonlinearity and ill-posedness are two common challenges in inversion problems across various disciplines, establishing constraints on nonlinear relationships and applying appropriate machine learning techniques can be treated as vital approaches for future research. Key focus areas include heterogeneous structure parameterization, monitoring network design, and multi-source data assimilation. ~~Meanwhile, the equifinality induced by ill-posedness can be attenuated through monitoring network optimizations. The model processes in this study are clearly defined, but in real-world scenarios, model response measurements often involve significant uncertainty, especially in groundwater systems with multi-component reactive transport. In such scenarios, it is crucial to consider the inversion of model parameters and integrate the identification of key processes within the groundwater system models. These optimization challenges may also benefit from the capabilities of machine learning, since they often involve uncertain quantifications.~~ Furthermore, it is essential to continuously follow the latest developments of machine learning and consider integrating more advanced DNN models to address increasingly complex groundwater system inversion problems. For example, the emergence of large language models offers opportunities for complex system modeling and inversion studies across various scientific and engineering disciplines (Birhane et al., 2023; Buehler, 2023). The latent capacity of large language models has yet to be fully explored. Significant achievements have primarily focus on protein designs (Jumper et al., 2021; Ferruz et al., 2022; Lin et al., 2023), drug development (Peng et al., 2023; Duffy et al., 2024), and molecular discovery (Flam-Shepherd et al., 2022; Li et al., 2023). Developing groundwater system inversion frameworks based on large language models is of great significance for the advancement of hydrology and earth science (Deng et al., 2023; Foroumandi et al., 2023).

Competing interests

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

Acknowledgments

This work is supported by the Fundamental Research Funds for the Central Universities (XJ2023005201), the National Natural Science Foundation of China (NSFC: 42402241, U2267217, 42141011, and 42002254).

Data Availability Statement

The data and codes for four surrogate models and five optimization algorithms are available on: https://doi.org/10.5281/zenodo.10499582

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+, Hydrology and Earth System Sciences, 28, 21-48, 10.5194/hess-28-21-2024, 2024.

Adler, J. and Öktem, O.: Solving ill-posed inverse problems using iterative deep neural networks, Inverse Problems, 33, 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 Resources Research, 50, 6135-6153, 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, Hydrology and Earth System Sciences, 26, 4469-4495, 10.5194/hess-26-4469-2022, 2022.

Bao, J., Li, L., and Redoloza, F.: Coupling ensemble smoother and deep learning with generative adversarial networks to deal with non-Gaussianity in flow and transport data assimilation, Journal of Hydrology, 590, 10.1016/j.jhydrol.2020.125443, 2020.

Bentivoglio, R., Isufi, 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, 10.5194/hess-26-4345-2022, 2022.

Birhane, A., Kasirzadeh, A., Leslie, D., and Wachter, S.: Science in the age of large language models, Nat Rev Phys, 5, 277-280, 10.1038/s42254-023-00581-4, 2023.

Botto, A., Belluco, E., and Camporese, M.: Multi-source data assimilation for physically based hydrological modeling of an experimental hillslope, Hydrology and Earth System Sciences, 22, 4251-4266, 10.5194/hess-22-4251-2018, 2018.

Buehler, M. J.: MeLM, a generative pretrained language modeling framework that solves forward and inverse mechanics problems, J Mech Phys Solids, 181, 10.1016/j.jmps.2023.105454, 2023.

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 Research, 268, 122706, https://doi.org/10.1016/j.watres.2024.122706, 2025.

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, 10.5194/hess-13-503-2009, 2009.

Chai, J., Zhang, W., Zhao, K., Li, S., Baloch, M. Y. J., Wang, Z., Zhang, D., and Yang, Y.: Multi-biological risk in groundwater-surface water system under landfill stress: Driven by bacterial size and biological toxicity, Journal of Hydrology, 636, 10.1016/j.jhydrol.2024.131282, 2024.

Chang, H. and Zhang, D.: Identification of physical processes via combined data-driven and data-assimilation methods, Journal of Computational Physics, 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, Journal of Hydrology, 612, 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 Resources Research, 57, 2021WR030595, 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 Resources Research, 58, 2022WR032429, 10.1029/2022wr032429, 2022.

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 Resources Research, 49, 7064-7076, 10.1002/2012wr013285, 2013.

Dai, Z. and Samper, J.: Inverse problem of multicomponent reactive chemical transport in porous media: Formulation and applications, Water Resources Research, 40, W07407, 10.1029/2004wr003248, 2004.

Dai, Z. and Samper, J.: Inverse modeling of water flow and multicomponent reactive transport in coastal aquifer systems, Journal of Hydrology, 447-461, 10.1016/j.jhydrol.2005.11.052, 2006.

Deng, C., Zhang, T., He, Z., Xu, Y., Chen, Q., Shi, Y., Fu, L., Zhang, W., Wang, X., Zhou, C., Lin, Z., and He, J.: K2: A Foundation Language Model for Geoscience Knowledge Understanding and Utilization, arXiv:2306.05064v2 2023.

Dragonetti, G., Comegna, A., Ajeel, A., Deidda, G. P., Lamaddalena, N., Rodriguez, G., Vignoli, G., and Coppola, A.: Calibrating electromagnetic induction conductivities with time-domain reflectometry measurements, Hydrol. Earth Syst. Sci., 22, 1509-1523, 10.5194/hess-22-1509-2018, 2018.

Duffy, Á., Petrazzini, B. O., Stein, D., Park, J. K., Forrest, I. S., Gibson, K., Vy, H. M., Chen, R., Márquez-Luna, C., Mort, M., Verbanck, M., Schlessinger, A., Itan, Y., Cooper, D. N., Rocheleau, G., Jordan, D. M., and Do, R.: Development of a human genetics-guided priority score for 19,365 genes and 399 drug indications, Nature Genetics, 10.1038/s41588-023-01609-2, 2024.

Eberhart, R. and Kennedy, J.: Particle swarm optimization, Proceedings of the IEEE international conference on neural networks, 1942-1948,

Ferruz, N., Schmidt, S., and Hocker, B.: ProtGPT2 is a deep unsupervised language model for protein design, Nature Communications, 13, 4348, 10.1038/s41467-022-32007-7, 2022.

Flam-Shepherd, D., Zhu, K., and Aspuru-Guzik, A.: Language models can learn complex molecular distributions, Nature Communications, 13, 10.1038/s41467-022-30839-x, 2022.

Foroumandi, E., Moradkhani, H., Sanchez-Vila, X., Singha, K., Castelletti, A., and Destouni, G.: ChatGPT in Hydrology and Earth Sciences: Opportunities, Prospects, and Concerns, Water Resources Research, 59, 10.1029/2023wr036288, 2023.

Goodfellow, I., Bengio, Y., Courville, A., and Bengio, Y.: Deep learning, 2, MIT press Cambridge2016.

He, K., Zhang, X., Ren, S., and Sun, J.: Deep residual learning for image recognition, Proceedings of the IEEE conference on computer vision and pattern recognition, 770-778,

Hinton, G. E. and Salakhutdinov, R. R.: Reducing the dimensionality of data with neural networks, Science, 313, 504-507, 2006.

Holland John, H.: Adaptation in natural and artificial systems, University of Michigan Press, Ann Arbor1975.

Ines, A. V. M. and Droogers, P.: Inverse modelling in estimating soil hydraulic functions: a Genetic Algorithm approach, Hydrology and Earth System Sciences, 6, 49-66, 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, Journal of Hydrology, 604, 10.1016/j.jhydrol.2021.127233, 2022.

Jaumann, S. and Roth, K.: Soil hydraulic material properties and layered architecture from time-lapse GPR, Hydrology and Earth System Sciences, 22, 2551-2573, 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 Resources Research, 40, 10.1029/2004wr003363, 2004.

Jumper, J., Evans, R., Pritzel, A., Green, T., Figurnov, M., Ronneberger, O., Tunyasuvunakool, K., Bates, R., Žídek, A., Potapenko, A., Bridgland, A., Meyer, C., Kohl, S. A. A., Ballard, A. J., Cowie, A., Romera-Paredes, B., Nikolov, S., Jain, R., Adler, J., Back, T., Petersen, S., Reiman, D., Clancy, E., Zielinski, M., Steinegger, M., Pacholska, M., Berghammer, T., Bodenstein, S., Silver, D., Vinyals, O., Senior, A. W., Kavukcuoglu, K., Kohli, P., and Hassabis, D.: Highly accurate protein structure prediction with AlphaFold, Nature, 596, 583-589, 10.1038/s41586-021-03819-2, 2021.

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, Journal of Hydrology, 561, 688-701, 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, doi:10.1126/science.220.4598.671, 1983.

Kuang, W., Yuan, C., and Zhang, J.: Real-time determination of earthquake focal mechanism via deep learning, Nature Communications, 12, 1432, 10.1038/s41467-021-21670-x, 2021.

LeCun, Y., Bengio, Y., and Hinton, G.: Deep learning, Nature, 521, 436-444, 10.1038/nature14539, 2015.

Li, E.: An adaptive surrogate assisted differential evolutionary algorithm for high dimensional constrained problems, Applied Soft Computing, 85, 10.1016/j.asoc.2019.105752, 2019.

Li, J., Liu, Y., Fan, W., Wei, X.-Y., Liu, H., Tang, J., and Li, Q.: Empowering Molecule Discovery for Molecule-Caption Translation with Large Language Models: A ChatGPT Perspective, arXiv preprint arXiv:.06615, arxiv.org/abs/2306.05064, 2023.

Lin, P., Tao, H., Li, H., and Huang, S.-Y.: Protein–protein contact prediction by geometric triangle-aware protein language models, Nature Machine Intelligence, 5, 1275-1284, 10.1038/s42256-023-00741-2, 2023.

Lindsay, A., McCloskey, J., and Bhloscaidh, M. N.: Using a genetic algorithm to estimate the details of earthquake slip distributions from point surface displacements, Journal of Geophysical Research-Solid Earth, 121, 1796-1820, 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, 10.1021/acsphotonics.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, Journal of Geophysical Research: Solid Earth, 128, 10.1029/2022jb025378, 2023.

Long, Y., Ren, J., Li, Y., and Chen, H.: Inverse design of photonic topological state via machine learning, Applied Physics Letters, 114, 181105, 10.1063/1.5094838, 2019.

Luo, J., Li, X., Xiong, Y., and Liu, Y.: Groundwater pollution source identification using Metropolis-Hasting algorithm combined with Kalman filter algorithm, Journal of Hydrology, 626, 10.1016/j.jhydrol.2023.130258, 2023a.

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, 10.1016/j.envres.2023.117268, 2023b.

Lykkegaard, M. B., Dodwell, T. J., and Moxey, D.: Accelerating uncertainty quantification of groundwater flow modelling using a deep neural network proxy, Computer Methods in Applied Mechanics and Engineering, 383, 10.1016/j.cma.2021.113895, 2021.

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, 10.1007/s10346-022-01923-6, 2022.

Makhzani, A., Shlens, J., Jaitly, N., Goodfellow, I., and Frey, B.: Adversarial autoencoders, arXiv preprint arXiv:.05644, 2015.

Mariethoz, G. and Caers, J.: Multiple-point geostatistics: stochastic modeling with training images, John Wiley & Sons2014.

Mariethoz, G., Renard, P., and Straubhaar, J.: The Direct Sampling method to perform multiple-point geostatistical simulations, Water Resources Research, 46, W11536, https://doi.org/10.1029/2008WR007621, 2010.

Metropolis, N., Rosenbluth, A. W., Rosenbluth, M. N., Teller, A. H., and Teller, E.: Equation of state calculations by fast computing machines, The journal of chemical physics, 21, 1087-1092, 1953.

Mo, S., Zabaras, N., Shi, X., and Wu, J.: Deep autoregressive neural networks for high-dimensional inverse problems in groundwater contaminant source identification, Water Resources Research, 55, 3856-3881, 10.1029/2018wr024638, 2019.

Mo, S., Zabaras, N., Shi, X., and Wu, J.: Integration of adversarial autoencoders with residual dense convolutional networks for estimation of non-Gaussian hydraulic conductivities, Water Resources Research, 56, 2019WR026082, 2020.

Nhu, V. H.: Levenberg-Marquardt method for ill-posed inverse problems with possibly non-smooth forward mappings between Banach spaces, Inverse Problems, 38, ARTN 015007
10.1088/1361-6420/ac38b7, 2022.

Peng, C., Yang, X., Chen, A., Smith, K. E., PourNejatian, N., Costa, A. B., Martin, C., Flores, M. G., Zhang, Y., Magoc, T., Lipori, G., Mitchell, D. A., Ospina, N. S., Ahmed, M. M., Hogan, W. R., Shenkman, E. A., Guo, Y., Bian, J., and Wu, Y.: A study of generative large language model for medical research and healthcare, npj Digital Medicine, 6, 210, 10.1038/s41746-023-00958-w, 2023.

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, 757-762,

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 Resources Research, 58, 10.1029/2021wr031532, 2022.

Rafiei, 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, Environmental Modelling & Software, 149, 10.1016/j.envsoft.2022.105312, 2022.

Razavi, S., Tolson, B. A., and Burn, D. H.: Review of surrogate modeling in water resources, Water Resources Research, 48, 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 Transactions on Signal Processing, 52, 2298-2307, 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 Resources Research, 46, 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, Hydrology and Earth System Sciences, 15, 3043-3059, 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, Hydrology and Earth System Sciences, 14, 765-781, 10.5194/hess-14-765-2010, 2010.

Shen, C., Appling, A. P., Gentine, P., Bandai, T., Gupta, H., Tartakovsky, 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.: Differentiable modelling to unify machine learning and physical models for geosciences, Nature Reviews Earth & Environment, 4, 552-567, 10.1038/s43017-023-00450-9, 2023.

Steefel, C., Depaolo, D., and Lichtner, P.: Reactive transport modeling: An essential tool and a new research approach for the earth sciences, Earth and Planetary Science Letters, 240, 539-558, 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, Hydrology and Earth System Sciences, 25, 1483-1508, 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, Journal of Global Optimization, 11, 341-359, 10.1023/A:1008202821328, 1997.

Sun, A. Y.: Discovering state-parameter mappings in subsurface models using generative adversarial networks, Geophysical Research Letters, 45, 11137-11146, 10.1029/2018gl080404, 2018.

Sun, N.: Inverse problems in groundwater modeling, Springer Science & Business Media2013.

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, Hydrology and Earth System Sciences, 27, 1343-1359, 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 Resources Research, 39, 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 Geoscience and Remote Sensing Letters, 8, 804-808, 10.1109/lgrs.2011.2109934, 2011.

Vrugt, J. A.: Markov chain Monte Carlo simulation using the DREAM software package: Theory, concepts, and MATLAB implementation, Environmental Modelling & Software, 75, 273-316, 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, Ecological Modelling, 253, 107-116, 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, Journal of Geophysical Research: Solid Earth, 126, 2020JB020549, 10.1029/2020jb020549, 2021.

Wang, Y., Fang, Z., and Hong, H.: Comparison of convolutional neural networks for landslide susceptibility mapping in Yanshan County, China, Science of The Total Environment, 666, 975-993, 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, Hydrology and Earth System Sciences, 25, 1689-1709, 10.5194/hess-25-1689-2021, 2021.

Xu, Z. X., 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, Hydrology and Earth System Sciences, 26, 755-773, 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, Journal of Geophysical Research: Solid Earth, 128, 10.1029/2023jb027157, 2023.

Yang, X., Chen, X., and Smith, M. M.: Deep learning inversion of gravity data for detection of CO_2 plumes in overlying aquifers, Journal of Applied Geophysics, 196, 104507, 10.1016/j.jappgeo.2021.104507, 2022.

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, 10.1515/nanoph-2020-0549, 2021.

Yu, S. and Ma, J.: Deep Learning for Geophysics: Current and Future Trends, Reviews of Geophysics, 59, 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, Geophysical Research Letters, 49, 2021GL095823, 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-Science Reviews, 239, 104370, 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-Loeve and polynomial expansions, Journal of Computational Physics, 194, 773-794, 10.1016/j.jcp.2003.09.015, 2004.

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 Resources Research, 54, 1716-1733, 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 Resources Research, 60, e2023WR035389, https://doi.org/10.1029/2023WR035389, 2024.

Zhou, H., Gómez-Hernández, J. J., and Li, L.: Inverse methods in hydrogeology: Evolution and recent trends, Advances in Water Resources, 63, 22-37, 10.1016/j.advwatres.2013.10.014, 2014.