Machine learning subsurface flow equations from data

Haibin Chang¹ · Dongxiao Zhang¹

Received: 13 December 2018 / Accepted: 14 June 2019 / Published online: 12 July 2019
© The Author(s) 2019

Abstract

Governing equations of physical problems are traditionally derived from conservation laws or physical principles. However, some complex problems still exist for which these first-principle derivations cannot be implemented. As data acquisition and storage ability have increased, data-driven methods have attracted great attention. In recent years, several works have addressed how to learn dynamical systems and partial differential equations using data-driven methods. Along this line, in this work, we investigate how to discover subsurface flow equations from data via a machine learning technique, the least absolute shrinkage and selection operator (LASSO). The learning of single-phase groundwater flow equation and contaminant transport equation are demonstrated. Considering that the parameters of subsurface formation are usually heterogeneous, we propose a procedure for learning partial differential equations with heterogeneous model parameters for the first time. Derivative calculation from discrete data is required for implementing equation learning, and we discuss how to calculate derivatives from noisy data. For a series of cases, the proposed data-driven method demonstrates satisfactory results for learning subsurface flow equations.

Keywords Machine learning · Data-driven discovery · Governing equations · Noisy data · LASSO

1 Introduction

Governing equations can provide fundamental characterization of physical processes, which is traditionally derived from conservation laws or physical principles. Many well-known governing equations of subsurface flow problems, such as groundwater flow and contaminant transport, were discovered by adopting these first-principle derivations. For example, the governing equation of groundwater flow can be discovered by performing mass conservation analysis of a control volume and incorporating Darcy's law. In order to discover the governing equation, a thorough understanding of the physical process is required. However, some complex subsurface problems still exist for which these first-principle derivations cannot be implemented due to uncertainty about the physical process.

As data acquisition and storage ability have increased, spatial and temporal monitoring of some physical processes have become feasible. In addition, acquired monitoring data can provide important resources for discovering the governing equations of physical processes. In recent years, data-driven methods have gained increasing attention, and several works have addressed how to learn dynamical systems and partial differential equations of physical processes using data-driven methods. For example, Bongard and Lipson [3] and Schmidt and Lipson [23] investigated the automated discovery of natural laws from data. In their works, symbolic regression is utilized to determine both the structure of the governing equations and the parameters simultaneously. Brunton et al. [6] proposed to utilize a sparse regression technique to discover governing equations of nonlinear dynamical systems from data. In their work, a candidate nonlinear function library is constructed first. Then, by assuming that only a few terms in the candidate library form the governing equation of a dynamical system, a sequential thresholded least-squares technique is proposed to obtain the sparse results. Mangan et al. [16] utilized a sparsity-promoting technique to infer biological networks with rational function nonlinearities. In their work, in order

✉ Dongxiao Zhang
dxz@pku.edu.cn

¹ ERE and BIC-ESAT, College of Engineering, Peking University, Beijing 100871, China

to deal with rational function nonlinearities, the candidate function library is constructed containing both functions of the state variable and functions of the derivative of the state variable with respect to time. An alternating-directions method is employed to obtain the sparse results, and Pareto front is calculated to select the most parsimonious model from different weights of the sparsity constraint. Mangan et al. [17] utilized Akaike information criteria (AIC) to further select the most informative model from the initial sub-selection models from the sparse identification of nonlinear dynamical systems. Rudy et al. [21] investigated the discovery of partial differential equations from data. In their work, the candidate function library is constructed containing the terms of the state variables and the partial derivatives of the state variable with respect to spatial dimension, and a sequential threshold ridge regression technique is proposed to obtain the sparse results. Schaeffer [22] employed L^1 regularized least-squares minimization to learn the partial differential equations from data.

Along this line, we investigate how to discover subsurface flow equations from data using a machine learning technique, the least absolute shrinkage and selection operator (LASSO). The learning of single-phase groundwater flow equation and contaminant transport equation are demonstrated. For the single-phase groundwater flow problem, a 1-D flow without source term and a 2-D flow with source term are considered. For the contaminant transport problem, both 1-D and 2-D flows with instantaneous point source released at the origin initially are considered. The influence of the data size and the finite difference scheme for calculating derivatives are investigated. Considering that the parameters of subsurface formation are usually heterogeneous, we proposed a procedure for learning the partial differential equation with heterogeneous model parameters. To the best of our knowledge, the learning of partial differential equation with heterogeneous model parameters has not yet been studied. Derivative calculation from discrete data is required for implementing equation learning, and we investigate how to calculate the derivative from noisy data using a polynomial technique. For a series of cases, the proposed data-driven method demonstrates satisfactory results for learning subsurface flow equations.

2 Methods

For a complex physical problem, if the first-principle derivation of the governing equation is difficult or impractical, discovering it from monitoring data using a data-driven method constitutes another option. In this work, we investigate the data-driven discovery of governing equations. And, we focus on physical problems that are governed by partial differential equations (PDE).

2.1 General form of PDE

In order to learn the PDE from data, we first need to establish a general form of PDE. To achieve this, let us first recall some PDEs. Two PDEs from subsurface flow problems are given as follows:

$$\frac{\partial u}{\partial t} = S_s^{-1} K \frac{\partial^2 u}{\partial x^2} + S_s^{-1} Q, \quad (1)$$

$$\frac{\partial u}{\partial t} = -v \frac{\partial u}{\partial x} + D \frac{\partial^2 u}{\partial x^2}. \quad (2)$$

Equation 1 is the governing equation of groundwater flow in a saturated confined aquifer, where u denotes hydraulic head S_s denotes specific storage K denotes hydraulic conductivity, and Q denotes source or sink term. Equation 2 is the governing equation of the contaminant transport process, where u denotes concentration, v denotes pore water velocity, and D denotes dispersion coefficient. We may recall some additional PDEs from other fields, such as the Burgers equation with the form $\partial u / \partial t = -u \partial u / \partial x + \varepsilon \partial^2 u / \partial x^2$ and the Kuramoto-Sivashinsky equation with the form $\partial u / \partial t = -u \partial u / \partial x - \partial^2 u / \partial x^2 - \partial^4 u / \partial x^4$. By observing these equations, the general form of this kind of PDEs can be expressed as

$$\frac{\partial u}{\partial t} = \Phi(u, Q)\alpha, \quad (3)$$

with

$$\begin{aligned} & \Phi(u, Q) \\ &= \left[1, u, Q, \dots, \frac{\partial u}{\partial x}, \frac{\partial^2 u}{\partial x^2}, \frac{\partial^3 u}{\partial x^3}, \dots, u \frac{\partial u}{\partial x}, u \frac{\partial^2 u}{\partial x^2}, u \frac{\partial^3 u}{\partial x^3}, \dots \right], \end{aligned} \quad (4)$$

where $\Phi(u, Q)$ is the library of candidate terms and α is the coefficient. Here, note that for a given PDE, α is a sparse vector, i.e., only a small number of entries of α are nonzero. For example, there are only two terms of the candidate library that have nonzero coefficients for Eqs. 1 and 2.

When monitoring data are available, the target is to learn the coefficient using a data-driven method. Here, we suppose that the monitoring data are taken at m spatial locations, $x_1, x_2, \dots, x_m$, and for each monitoring location, they are recorded at n time steps, $t_1, t_2, \dots, t_n$. Let N_d denote the number of data, and then we have $N_d = m \times n$. Considering the fact that the PDE holds for each spatial location at each time, Eq. 3 can be written as

$$\frac{\partial U}{\partial t} = \Phi(U, Q)\alpha, \quad (5)$$

which is

$$\begin{bmatrix} \frac{\partial u}{\partial t}(x_1, t_1) \\ \frac{\partial u}{\partial t}(x_2, t_1) \\ \vdots \\ \frac{\partial u}{\partial t}(x_m, t_1) \\ \vdots \\ \frac{\partial u}{\partial t}(x_1, t_n) \\ \frac{\partial u}{\partial t}(x_2, t_n) \\ \vdots \\ \frac{\partial u}{\partial t}(x_m, t_n) \end{bmatrix} = \begin{bmatrix} 1 & u(x_1, t_1) & \cdots & \frac{\partial u}{\partial x}(x_1, t_1) & \cdots \\ 1 & u(x_2, t_1) & \cdots & \frac{\partial u}{\partial x}(x_2, t_1) & \cdots \\ \vdots & \vdots & \ddots & \vdots & \ddots \\ 1 & u(x_m, t_1) & \cdots & \frac{\partial u}{\partial x}(x_m, t_1) & \cdots \\ \vdots & \vdots & \ddots & \vdots & \ddots \\ 1 & u(x_1, t_n) & \cdots & \frac{\partial u}{\partial x}(x_1, t_n) & \cdots \\ 1 & u(x_2, t_n) & \cdots & \frac{\partial u}{\partial x}(x_2, t_n) & \cdots \\ \vdots & \vdots & \ddots & \vdots & \ddots \\ 1 & u(x_m, t_n) & \cdots & \frac{\partial u}{\partial x}(x_m, t_n) & \cdots \end{bmatrix} \begin{bmatrix} \alpha_1 \\ \alpha_2 \\ \vdots \\ \vdots \\ \vdots \\ \vdots \\ \vdots \\ \vdots \end{bmatrix} \quad (6)$$

Overall, learning PDE from data is to learn the coefficient α , a zero value of which indicates the non-existence of the particular term in Eq. 6. In order to implement PDE learning, we need to calculate the partial derivatives of the data with respect to time and spatial dimensions, which will be discussed in the case studies.

2.2 Sparse regression method

Utilizing available data, solving the problem described in Eq. 6 constitutes a regression problem, which can be described as

$$\min_{\alpha} \left\| \frac{\partial U}{\partial t} - \Phi(U, Q)\alpha \right\|_2^2. \quad (7)$$

The least-squares method can be utilized to solve the problem described in Eq. 7. However, the results from least squares are usually not sparse [24]. As discussed above, α should be a sparse vector for the problem of learning PDE. In order to obtain sparse results, a straightforward strategy is to add a L^0 regularization term in the objective function of minimization as

$$\min_{\alpha} \left\| \frac{\partial U}{\partial t} - \Phi(U, Q)\alpha \right\|_2^2 + \lambda \|\alpha\|_0, \quad (8)$$

where $\lambda > 0$ is a weight parameter, which regulates the strength of the constraint. Here, the L norm of a vector is defined as the number of nonzero entries of the vector. For the optimization problem with an L regularization term, it is difficult to derive a stable numerical scheme. One way to relax the problem is to use an L^1 regularization term instead of an L regularization term as

$$\min_{\alpha} \left\| \frac{\partial U}{\partial t} - \Phi(U, Q)\alpha \right\|_2^2 + \lambda \|\alpha\|_1, \quad (9)$$

Tibshirani [24] addressed the L^1 regularization problem and proposed the least absolute shrinkage and selection operator (LASSO) to solve this problem. Since then, many further

works followed to investigate the property and solver of the LASSO problem [2, 4, 10–12, 20, 24, 25]. Among the solvers of the LASSO problem, the least angle regression method was shown to be an efficient method, which can provide the entire LASSO solution path [10].

In this work, LASSO is employed to learn the subsurface flow equation from data. An open-source Matlab code is utilized to run LASSO, and the inherent algorithm is a variant of the least angle regression method. The procedure of the adopted solver is given as follows [12]:

1. In preparation, $\partial U/\partial t$ and each column of $\Phi(U, Q)$ are normalized to have zero mean and unit variance. Initially, α is set to be zero and the residual, $\mathbf{r}$, is set to be $\partial U/\partial t$.
2. Find the candidate term Φ_i that is most correlated with $\mathbf{r}$.
3. Move α_i from zero towards its least-squares coefficient until another candidate term Φ_j has as much correlation as Φ_i with the current residual.
4. Move $\{\alpha_i, \alpha_j\}$ along the direction defined by their joint least-squares coefficient of the current residual until another candidate term Φ_k has as much correlation with the current residual.
5. If the coefficient of an added candidate term approaches zero, this term will be dropped. The coefficients of the remaining candidate terms are then recomputed.
6. Continue in this way until all of the candidate terms are added or a stopping criterion is met.

3 Results

In this section, we employ the groundwater flow equation and the contaminant transport equation to test the performance of LASSO for learning PDE.

3.1 Learning groundwater flow equation with homogeneous conductivity

We first investigate the learning of the PDE of groundwater flow in a saturated confined aquifer. For learning PDE, spatial and temporal data are required, which are usually obtained by on-site monitoring or laboratory experiments. Since the aim of this work is proof of concept, we synthetically generate the data through numerical simulation. One dimensional (1-D) flow in a saturated confined aquifer is first considered. The domain length is 1010 m. Initially, the hydraulic head is 1 m at the left boundary and m at other locations. There is no source or sink term. The conductivity is supposed to be homogeneous with a value of 1 m/day. In addition, the specific storage is 10^{-4} 1/m. We monitor the dynamical flow process. There are 101 monitoring

Table 1 LASSO solution path for the 1-D groundwater flow problem

Step	Added	Dropped	Active set size
1	$\partial^2 u / \partial x^2$		1
2	$u \partial^2 u / \partial x^2$		2
3	$u \partial^3 u / \partial x^3$		3
4		$u \partial^2 u / \partial x^2$	2
5	u		3
6	$\partial u / \partial x$		4
7	$\partial^3 u / \partial x^3$		5
8	$u \partial^2 u / \partial x^2$		6
9	$u \partial u / \partial x$		7
10	1		8

locations uniformly distributed in the domain. The monitoring starts at time zero and ends at day 10. Monitoring is taken every 0.1 day. Numerical simulation by MODFLOW software is utilized here to generate the data, and the domain is evenly divided into 101 grid blocks. Here, we assume that the monitoring locations and the grid block centers are coincident. For these data, we have $m = 101$, $n = 100$, and $N_d = 10$, 100.

The library of candidate terms is chosen as

$$\Phi(u, Q) = \left[1, u, \frac{\partial u}{\partial x}, \frac{\partial^2 u}{\partial x^2}, \frac{\partial^3 u}{\partial x^3}, u \frac{\partial u}{\partial x}, u \frac{\partial^2 u}{\partial x^2}, u \frac{\partial^3 u}{\partial x^3} \right]. \quad (10)$$

There are eight terms in this library, including three nonlinear terms, and we consider the derivatives up to order three. In this work, we assume that the candidate library is complete for the unknown PDE, i.e., the terms in the PDE are contained in the candidate library. Furthermore, we need to calculate the derivatives in Eq. 10 from discrete data. Finite difference is employed to calculate the derivatives as shown in Appendix A, and the equation is given in Eq. A.1. Prior to performing equation learning, $\partial U / \partial t$ and each column of $\Phi(U, Q)$ (except for the first column) are normalized to have zero mean and unit variance.

We then employ LASSO to learn the PDE. For the adopted solver, in each step, it adds a term from the candidate library or drops an added term. At the final step, it will select all of the candidate terms, but will assign nearly zero values for these unimportant terms, making the solution sparse. The LASSO solution path is given in Table 1. In this table, we show the added term, the dropped term, and the active set size at each step. Here, the active set size means the number of selected terms. From Table 1, one can see that the term $\partial^2 u / \partial x^2$ is added in the first step. The change of the coefficient and the Bayesian information criterion (BIC) along the solution path are given in Fig. 1. Here, the BIC is calculated using the following formula:

$$BIC = \frac{\left\| \frac{\partial U}{\partial t} - \Phi(U, Q) \alpha \right\|_2^2}{\sigma^2} + N_k \ln(N_d), \quad (11)$$

where N_k denotes the active set size and σ^2 denotes the data error variance. The BIC can measure both the residual error and model complexity. From Fig. 1a, it can be seen that the coefficient of $\partial^2 u / \partial x^2$ is 1.000×10^4 , remaining stable after this term has been added, while the coefficients of other terms are close to zero. From Fig. 1b, one can see that BIC decreases to a small value after $\partial^2 u / \partial x^2$ has been added, and remains stable afterwards. For determining the final learned PDE, we introduce a truncation of the adopted solver, i.e., if the absolute value of the coefficient of a normalized candidate term is less than 0.001, we will assign zero value to that coefficient. At the minimum of BIC, the learned equation takes the form

$$\frac{\partial u}{\partial t} = 1.000 \times 10^4 \frac{\partial^2 u}{\partial x^2}. \quad (12)$$

From Eq. 12, it can be seen that the term for PDE of 1-D groundwater flow in a saturated confined aquifer is correctly selected and the coefficient of the selected term is accurately learned. Here, note that the units of the coefficients of the selected terms can be obtained according to the consistency of the SI unit. In Eq. 12, 1.000×10^4 should have a

Fig. 1 Change of coefficient (a) and BIC (b) along the solution path

Table 2 LASSO solution path for the 2-D groundwater flow problem

Step	Added	Dropped	Active set size
1	u		1
2	$\partial^2 u / \partial y^2$		2
3	$\partial^2 u / \partial x^2$		3
4	$u \partial^2 u / \partial y^2$		4
5		$\partial^2 u / \partial y^2$	3
6	$u \partial^2 u / \partial x^2$		4
7		$\partial^2 u / \partial x^2$	3
8	Q		4
9	$\partial^2 u / \partial x^2$		5
10	$\partial^2 u / \partial y^2$		6
11	$u \partial^2 u / \partial x \partial y$		7
12	$\partial^2 u / \partial x \partial y$		8
13	$\partial u / \partial y$		9
14	$u \partial u / \partial y$		10
15	$\partial u / \partial x$		11
16	$u \partial u / \partial x$		12
17	1		13

unit of m^2/day . For the sake of conciseness, the units of the coefficients in the learned PDE are not given in the following cases.

We then continue to investigate the learning of the PDE of 2-D groundwater flow in a saturated confined aquifer. The data are also synthetically generated by MODFLOW software. A square domain is considered, and the side length is 1020 m, which is evenly divided into 51 grid blocks. The two lateral boundaries of the domain are treated as no-flow boundaries, while the left and the right boundaries are treated as Dirichlet boundaries with the prescribed hydraulic head being of 202 m and 200 m, respectively. A pumping well is located at the center of the domain with a pumping rate of $200 \text{ m}^3/\text{day}$. Conductivity and specific storage take the same values as that of the 1-D case. Monitoring is taken at all of the grid block centers. Monitoring starts at time zero

and ends on day 5. Monitoring is taken every 0.1 day. For these data, we have $m = 2601$, $n = 50$, and $N_d = 130,050$.

The candidate library is chosen as

$$\Phi(u, Q) = \left[1, u, Q, \frac{\partial u}{\partial x}, \frac{\partial u}{\partial y}, \frac{\partial^2 u}{\partial x^2}, \frac{\partial^2 u}{\partial y^2}, \frac{\partial^2 u}{\partial x \partial y}, u \frac{\partial u}{\partial x}, u \frac{\partial u}{\partial y}, u \frac{\partial^2 u}{\partial x^2}, u \frac{\partial^2 u}{\partial y^2}, u \frac{\partial^2 u}{\partial x \partial y} \right]. \quad (13)$$

There are 13 terms in this library, and we consider the derivatives up to order two. Here, for the source or sink term vector, Q , its entry is set to be 1 for the well location and 0 for the other locations. Finite difference is utilized to calculate the derivatives, and the equation is given in Eq. A.2

We then employ LASSO to learn the PDE. The LASSO solution path is given in Table 2. From Table 2, one can see that the terms $\partial^2 u / \partial y^2$ and $\partial^2 u / \partial x^2$ are added at step 2 and step 3, respectively, but are dropped at step 5 and step 7, respectively. At step 8, the source term Q is added, and then the terms $\partial^2 u / \partial y^2$ and $\partial^2 u / \partial x^2$ are added back in the next two steps. The change of the coefficient and the BIC along the solution path are given in Fig. 2. From Fig. 2a, it can be seen that, before the term Q is added, the coefficients of the added terms are nearly zero; after Q is added, the coefficients of terms $\partial^2 u / \partial y^2$ and $\partial^2 u / \partial x^2$ are learned to be the same as 1.000×10^4 , and the coefficient of the term Q is learned to be -0.500×10^4 . From Fig. 2b, one can see that before the term Q is added, there is no noticeable decrease of BIC; whereas, after the terms Q , $\partial^2 u / \partial y^2$ and $\partial^2 u / \partial x^2$ are all added, BIC has a large decrease at step 10 and has a slight improvement in the following steps. From these results, it can be seen that, for a physical problem with a source or sink, identification of the source or sink term is important for learning the inherent PDE. It is easy to understand this phenomenon by considering that the source or sink can largely impact the profile of the response. The identification of the source or sink term is difficult since the monitoring data reflecting this term are usually limited. For

Fig. 2 Change of coefficient (a) and BIC (b) along the solution path

this problem, at the minimum of BIC, the learned equation takes the form

$$\frac{\partial u}{\partial t} = 1.000 \times 10^4 \frac{\partial^2 u}{\partial x^2} + 1.000 \times 10^4 \frac{\partial^2 u}{\partial y^2} - 0.500 \times 10^4 Q. \quad (14)$$

In Eq. 14, $0.5 = 200/(20 \times 20) = Q/(\Delta x \times \Delta y)$, which is the volumetric pumping rate over a grid block. Then, one can see that the PDE of the 2-D groundwater flow in a saturated confined aquifer is successfully learned.

For the above-discussed 2-D problem, the data size is large since we have monitoring from 51×51 spatial locations. Here, we want to examine the influence of data size for LASSO. The same candidate library is used, and the derivatives are still calculated from the 51×51 grid system. However, the data from part of the spatial monitoring locations are utilized for performing LASSO. Extracting from the 51×51 spatial locations, the data from 11×11 spatial locations, including the well location, are employed to perform LASSO. The learned equation is the same as Eq. 14. Further, the data from 3×3 spatial locations, including the well location, are employed to perform LASSO, and the learned equation is also the same as Eq. 14. From the results, it can be seen that, when the derivatives in the candidate library can be calculated with high accuracy, the data size does not show a noticeable influence on the final learned equation. This suggests that, for learning the PDE of a physical problem, a sparse spatial monitoring system may work under the condition that for each monitoring location, some measurements at a nearby location exist to provide the data for calculating the derivative of the response.

3.2 Learning groundwater flow equation with heterogeneous conductivity

Considering the fact that the parameter field of subsurface formation is usually heterogeneous, here we want to investigate the learning of the PDE of groundwater flow with heterogeneous conductivity, which has not yet been studied. In order to learn the general form of the PDE of groundwater flow with heterogeneous conductivity, we need to know the detailed values of the conductivity, which is difficult for conductivity with large heterogeneity. Thus, we work with a zonally homogeneous conductivity field and suppose that the detailed values of the conductivity are known. Numerical simulation is still employed for generating the data. The conductivity field shown in Fig. 3 is considered, and we suppose that the zone distribution and the conductivity value inside of each zone are known. For this field, K equals 0.5 m/day for $1 \leq j \leq 17$, K equals 1.5 m/day for $18 \leq j \leq 34$, and K equals

0.25 m/day for $35 \leq j \leq 51$. Here note that as the zonally homogeneous porous media may be synthetically made in a lab, the zone distribution may be known for controlled experiments. And the corresponding data may be obtained from lab experiment. In order to discern the specific form of conductivity in a PDE, we add K , $\partial K/\partial y$, and some related terms into the candidate library, which takes the form:

$$\Phi(u, K, Q) = \left[1, u, K, \frac{\partial K}{\partial y}, Q, \frac{\partial u}{\partial x}, \frac{\partial u}{\partial y}, \frac{\partial^2 u}{\partial x^2}, \frac{\partial^2 u}{\partial y^2}, \frac{\partial^2 u}{\partial x \partial y}, K u, \frac{\partial K}{\partial y} u, u \frac{\partial u}{\partial x}, K \frac{\partial u}{\partial x}, \frac{\partial K}{\partial y} \frac{\partial u}{\partial x}, u \frac{\partial u}{\partial y}, K \frac{\partial u}{\partial y}, \frac{\partial K}{\partial y} \frac{\partial u}{\partial y}, u \frac{\partial^2 u}{\partial x^2}, K \frac{\partial^2 u}{\partial x^2}, \frac{\partial K}{\partial y} \frac{\partial^2 u}{\partial x^2}, u \frac{\partial^2 u}{\partial y^2}, K \frac{\partial^2 u}{\partial y^2}, \frac{\partial K}{\partial y} \frac{\partial^2 u}{\partial y^2}, u \frac{\partial^2 u}{\partial x \partial y}, K \frac{\partial^2 u}{\partial x \partial y}, \frac{\partial K}{\partial y} \frac{\partial^2 u}{\partial x \partial y} \right]. \quad (15)$$

There are 27 terms in this library. The finite difference scheme for computing $\partial K/\partial y$ is given in Eq. A.3. The other derivatives are calculated using Eq. A.2

We then employ LASSO to learn the PDE. At the minimum of BIC, the learned equation takes the form

$$\frac{\partial u}{\partial t} = \left[-0.047 \frac{\partial K}{\partial y} + 0.002 \frac{\partial K}{\partial y} \frac{\partial u}{\partial x} + 1.184 \frac{\partial K}{\partial y} \frac{\partial u}{\partial y} + 0.130 \frac{\partial^2 u}{\partial x^2} + 0.851 K \frac{\partial^2 u}{\partial x^2} - 1.548 \frac{\partial K}{\partial y} \frac{\partial^2 u}{\partial x^2} + 0.176 \frac{\partial^2 u}{\partial y^2} + 0.829 K \frac{\partial^2 u}{\partial y^2} + 5.105 \frac{\partial K}{\partial y} \frac{\partial^2 u}{\partial y^2} + 0.088 \frac{\partial K}{\partial y} \frac{\partial^2 u}{\partial x \partial y} - 0.470 Q \right] \times 10^4. \quad (16)$$

From Eq. 16, one can see that LASSO failed to learn the correct PDE. For the chosen candidate library, the PDE of this problem should only contain the terms $\partial K/\partial y \partial u/\partial y$, $K \partial^2 u/\partial x^2$, $K \partial^2 u/\partial y^2$, and Q . These four terms are identified in Eq. 16 with large coefficients, but there are numerous misidentified terms. Using a zonally homogeneous K , the term $\partial K/\partial y$ takes zero values inside of each zone, and takes non-zero values near the interface of the zones only. Thus, it will be difficult to calculate the correlation between this term with $\partial u/\partial t$. Then, it is hard to identify terms containing $\partial K/\partial y$. This may be the reason that there are some misidentified terms containing $\partial K/\partial y$. In addition, for zonally homogeneous K , the terms $\partial^2 u/\partial x^2$ and $K \partial^2 u/\partial x^2$ may have similar correlations with $\partial u/\partial t$. It is the same for $\partial^2 u/\partial y^2$ and $K \partial^2 u/\partial y^2$. This may be the reason for misidentifying the terms $\partial^2 u/\partial x^2$ and $\partial^2 u/\partial y^2$. The misidentification of the terms containing $\partial^2 u/\partial x \partial y$ may be caused by the fact that this term has small-scale values.

Another test is further implemented. For the candidate library shown in Eq. 15, $\partial(K \partial u/\partial y)/\partial y$ is introduced to replace $\partial K/\partial y \partial u/\partial y$ and $K \partial^2 u/\partial y^2$. The finite difference scheme shown in Eq. A.4 is utilized to calculate this newly introduced term. The learned equation takes the form

$$\frac{\partial u}{\partial t} = 1.000 \times 10^4 K \frac{\partial^2 u}{\partial x^2} + 1.000 \times 10^4 \frac{\partial}{\partial y} \left(K \frac{\partial u}{\partial y} \right) - 0.500 \times 10^4 Q. \quad (17)$$

It can be seen that, for the new candidate library, the PDE can be successfully learned. The reason for the good result may be that the term $\partial(K \partial u / \partial y) / \partial y$ has a larger correlation with $\partial u / \partial t$ than $\partial K / \partial y \partial u / \partial y$ and $K \partial^2 u / \partial y^2$. When the three terms on the right-hand side of Eq. 17 are identified, the residual will have a nearly zero value, and thus it will assign nearly zero values for other coefficients.

From the above cases, one can see that learning the PDE of a physical problem with heterogeneous parameters is difficult. Constructing a proper candidate library may benefit the learning results, but it is usually unknown beforehand. In the following, we propose a different procedure to solve this problem. Considering that, for a zonally homogeneous parameter field, the derivative of the parameter field with respect to spatial dimension has non-zero values near the interface of the zones only, we may learn the PDE for different parts of the domain separately. In our proposed procedure, the PDE inside of each zone is first learned. Next, we learn the PDE near the interface of zones. Finally, the general form of the PDE in the whole domain is summarized from the obtained results.

The candidate library shown in Eq. 13 is utilized to learn the PDE for the middle zone. In order to learn the PDE for the upper and bottom zones, and near the interface of zones, the source term is deleted from this candidate library.

Inside of each zone, the learned PDE takes the form of

$$\frac{\partial u}{\partial t} = 0.499 \times 10^4 \frac{\partial^2 u}{\partial x^2} + 0.499 \times 10^4 \frac{\partial^2 u}{\partial y^2}, \quad \text{for } 2 \leq j \leq 16, \quad (18)$$

$$\frac{\partial u}{\partial t} = 1.500 \times 10^4 \frac{\partial^2 u}{\partial x^2} + 1.500 \times 10^4 \frac{\partial^2 u}{\partial y^2} - 0.500 \times 10^4 Q, \quad \text{for } 19 \leq j \leq 33, \quad (19)$$

and

$$\frac{\partial u}{\partial t} = 0.250 \times 10^4 \frac{\partial^2 u}{\partial x^2} + 0.250 \times 10^4 \frac{\partial^2 u}{\partial y^2}, \quad \text{for } 36 \leq j \leq 50. \quad (20)$$

By considering the values of K inside of each zone and the value of S_s , the general form of the PDE inside of each zone takes the following form:

$$\frac{\partial u}{\partial t} = K \times S_s^{-1} \frac{\partial^2 u}{\partial x^2} + K \times S_s^{-1} \frac{\partial^2 u}{\partial y^2} + S_s^{-1} Q', \quad (21)$$

Here, the entry of Q' takes a value of the volumetric pumping rate at the well location and takes a value of zero at other locations. Thus, Q' is a zero vector for the up and bottom zone.

Near the interface of zones, the learned PDE takes the form of

$$\frac{\partial u}{\partial t} = 0.498 \times 10^4 \frac{\partial^2 u}{\partial x^2} + 0.621 \times 10^4 \frac{\partial^2 u}{\partial y^2} + 0.0124 \times 10^4 \frac{\partial u}{\partial y}, \quad \text{for } j = 17, \quad (22)$$

$$\frac{\partial u}{\partial t} = 1.483 \times 10^4 \frac{\partial^2 u}{\partial x^2} + 1.110 \times 10^4 \frac{\partial^2 u}{\partial y^2} + 0.0370 \times 10^4 \frac{\partial u}{\partial y}, \quad \text{for } j = 18, \quad (23)$$

$$\frac{\partial u}{\partial t} = 1.498 \times 10^4 \frac{\partial^2 u}{\partial x^2} + 0.963 \times 10^4 \frac{\partial^2 u}{\partial y^2} - 0.0535 \times 10^4 \frac{\partial u}{\partial y}, \quad \text{for } j = 34, \quad (24)$$

and

$$\frac{\partial u}{\partial t} = 0.250 \times 10^4 \frac{\partial^2 u}{\partial x^2} + 0.341 \times 10^4 \frac{\partial^2 u}{\partial y^2} - 0.0090 \times 10^4 \frac{\partial u}{\partial y}, \quad \text{for } j = 35. \quad (25)$$

From Eqs. 22–25, we can find that the coefficient of $\partial^2 u / \partial x^2$ is approximately $K S_s^{-1}$. We need to identify the meaning of the coefficients of $\partial^2 u / \partial y^2$ and $\partial u / \partial y$. To determine the coefficient of $\partial^2 u / \partial y^2$, we calculate an average of $K(x_{i,j})$ in the y direction using the following formula:

$$\begin{aligned} K(x_{i,j}) &= \frac{1}{2} (K(x_{i,j+1/2}) + K(x_{i,j-1/2})) \\ &= \frac{1}{2} \left(\frac{2}{1/K(x_{i,j+1}) + 1/K(x_{i,j})} + \frac{2}{1/K(x_{i,j}) + 1/K(x_{i,j-1})} \right), \end{aligned} \quad (26)$$

which gives values of 0.625, 1.125, 0.964, and 0.339 for $j = 17$, $j = 18$, $j = 34$, and $j = 35$, respectively. Thus, it can be seen that the coefficient of $\partial^2 u / \partial y^2$ is approximately $K S_s^{-1}$, with K being calculated using Eq. 26. Here, it is worth noting that the K in the coefficient of $\partial^2 u / \partial x^2$ can also be calculated using Eq. 26 by considering the fact that $K(x_{i+1/2,j})$ and $K(x_{i-1/2,j})$ take the same value. To discern the coefficient of $\partial u / \partial y$, we calculate $\partial K(x_{i,j}) / \partial y$ using Eq. A.3, which gives values of 0.0125, 0.0375, -0.0536 , and -0.0089 for $j = 17$, $j = 18$, $j = 34$, and $j = 35$, respectively. Thus, one can see that the coefficient of $\partial u / \partial y$ is approximately $\partial K / \partial y S_s^{-1}$.

Thus we conjecture that the general form of the PDE near the interfaces of zones takes the following form:

$$\frac{\partial u}{\partial t} = K \times S_s^{-1} \frac{\partial^2 u}{\partial x^2} + K \times S_s^{-1} \frac{\partial^2 u}{\partial y^2} + \frac{\partial K}{\partial y} S_s^{-1} \frac{\partial u}{\partial y}, \quad (27)$$

From Eqs. 21 and 27, we suppose that the general form of the PDE with heterogeneous K takes the form of

$$\frac{\partial u}{\partial t} = K \times S_s^{-1} \frac{\partial^2 u}{\partial x^2} + K \times S_s^{-1} \frac{\partial^2 u}{\partial y^2} + \frac{\partial K}{\partial y} S_s^{-1} \frac{\partial u}{\partial y} + S_s^{-1} Q'. \quad (28)$$

Fig. 3 Heterogeneous conductivity field with three layers. In this figure, i and j denote the grid block number in x and y direction, respectively

Here we use the fact that $\partial K/\partial y$ is a zero vector inside of each zone and Q' is a zero vector near the interface of the zones.

For the conductivity shown in Fig. 3, $\partial K/\partial x$ is a zero vector. Thus, the PDE shown in Eq. 28 may contain terms with $\partial K/\partial x$. If we further design a different conductivity field with layers extending along the y direction, we can discern that the PDE of groundwater flow in a confined aquifer with heterogeneous K should take the form

$$\begin{aligned} \frac{\partial u}{\partial t} &= K \times S_s^{-1} \frac{\partial^2 u}{\partial x^2} + K \times S_s^{-1} \frac{\partial^2 u}{\partial y^2} + \frac{\partial K}{\partial x} S_s^{-1} \frac{\partial u}{\partial x} \\ &\quad + \frac{\partial K}{\partial y} S_s^{-1} \frac{\partial u}{\partial y} + S_s^{-1} Q', \\ &= S_s^{-1} \frac{\partial}{\partial x} \left(K \frac{\partial u}{\partial x} \right) + S_s^{-1} \frac{\partial}{\partial y} \left(K \frac{\partial u}{\partial y} \right) + S_s^{-1} Q'. \end{aligned} \quad (29)$$

Thus it can be seen that, by using the proposed procedure, we can learn the PDE with heterogeneous parameters.

The reason that the coefficients K and $\partial K/\partial y$ need to be calculated using Eq. 26 and Eq. A.3, respectively, is discussed in Appendix B.

In this subsection, we assume that the detailed values of the heterogeneous conductivity are known for determining the general form of PDE with heterogeneous parameters. In our further work, we intend to develop a strategy for directly learning the general form of PDE without knowing the heterogeneous parameter values via combined data-driven and data-assimilation methods as recently proposed by the authors [7].

3.3 Learning contaminant transport equation

In this subsection, we investigate the learning of the PDE of the contaminant transport process. A 1-D problem with

instantaneous injection of contaminant at the origin is first investigated. An analytical solution for this problem exists, which is [1]

$$u(x, t) = \frac{u_0}{2\phi\sqrt{\pi D_L t}} \exp\left(-\left[\frac{(x - vt)^2}{4D_L t}\right]\right), \quad (30)$$

where u denotes concentration, u_0 denotes initial concentration at the origin, ϕ denotes porosity, D_L denotes longitudinal dispersion coefficient, and v denotes pore water velocity. The monitoring data are taken from the analytical solution. We set $u_0 = 1 \text{ g/m}^3$, $\phi = 0.15$, $D_L = 0.25$, and $v = 1 \text{ m/day}$. There are 300 monitoring locations uniformly distributed in the domain, and the nearby monitoring locations are separated by 0.1 m. The monitoring starts on day 2 and ends on day 15. Monitoring is taken every 0.1 d. For these data, we have $m = 300$, $n = 130$, and $N_d = 39,000$.

The candidate library is the same as that in Eq. 10. The finite difference scheme shown in Eq. A.5 is employed to compute the derivatives. In addition, L , which denotes the number of data points in each side of the considered location, is chosen to be two for calculating the derivatives with respect to x up to order three and the derivative with respect to t . We then utilize LASSO to learn the PDE. The change of the coefficient and the BIC along the solution path are given in Fig. 4. From Fig. 4a, one can see that the term $\partial u/\partial x$ is added at the first step, the term $\partial^2 u/\partial x^2$ is added at the second step, and the coefficients of these two terms remain stable in the following steps. At the minimum of BIC, the learned equation takes the form

$$\frac{\partial u}{\partial t} = -1.000 \frac{\partial u}{\partial x} + 0.250 \frac{\partial^2 u}{\partial x^2}. \quad (31)$$

From Eq. 31, it can be seen that the advection term and the dispersion term for the PDE of the 1-D contaminant transport process are correctly selected; the pore water velocity and the longitudinal dispersion coefficient are accurately learned. Thus, the PDE of the considered physical process is successfully learned.

In order to further test the performance of LASSO, the number of monitoring locations is reduced to 60. The monitoring locations are also evenly distributed, with the separation distance being 0.5 m. Different from the case discussed in Section 3.1, here, we assume that there is no more monitoring near each monitoring location, and the derivatives need to be calculated using the sparse monitoring network. L is also chosen to be two for calculating the derivatives. The learned equation takes the following form:

$$\begin{aligned} \frac{\partial u}{\partial t} &= -0.999 \frac{\partial u}{\partial x} + 0.003u \frac{\partial u}{\partial x} + 0.248 \frac{\partial^2 u}{\partial x^2} \\ &\quad + 0.001u \frac{\partial^2 u}{\partial x^2} + 0.002 \frac{\partial^3 u}{\partial x^3} + 0.005u \frac{\partial^3 u}{\partial x^3}. \end{aligned} \quad (32)$$

Fig. 4 Change of coefficient (a) and BIC (b) along the solution path

From Eq. 32, one can see that there are four misidentified terms with small coefficients, and there are small errors of the coefficients of $\partial u/\partial x$ and $\partial^2 u/\partial x^2$. The error of the learned equation may come from the inaccurate calculated derivatives that are caused by the large separation distance between monitoring locations. In order to improve the accuracy of derivative calculation, we may increase the number of data points for calculating the derivatives. To achieve this, L is set to be three, and the learned equation takes the form of

$$\frac{\partial u}{\partial t} = -1.000 \frac{\partial u}{\partial x} + 0.249 \frac{\partial^2 u}{\partial x^2} + 0.001 u \frac{\partial^3 u}{\partial x^3}. \quad (33)$$

From Eq. 33, it can be seen that there is one misidentified term with a small coefficient, and there is a small error of the coefficient of $\partial^2 u/\partial x^2$. Further, L is set to be four, and the learned equation takes the following form:

$$\frac{\partial u}{\partial t} = -1.000 \frac{\partial u}{\partial x} + 0.250 \frac{\partial^2 u}{\partial x^2}. \quad (34)$$

Thus, the equation can be successfully learned.

From the above discussion, one can see that, for a sparse monitoring network, using more data points for calculating derivatives can improve accuracy and can then lead to more accurate learned PDE.

Furthermore, a 2-D problem with instantaneous injection of contaminant at the origin initially is investigated. An analytical solution for this problem is [1]

$$u(x, t) = \frac{u_0}{4\pi\phi\sqrt{D_L D_T t}} \exp\left(-\left[\frac{(x-vt)^2}{4D_L t} + \frac{y^2}{4D_T t}\right]\right), \quad (35)$$

where D_T denotes the transverse dispersion coefficient. Monitoring is taken from the analytical solution. We set $u_0 = 1 \text{ g/m}^3$, $\phi = 0.15$, $D_L = 0.5$, $D_T = 0.05$, and $v = 1 \text{ m/day}$. There are 120×12 monitoring locations uniformly distributed in the domain, and the nearby monitoring locations are separated by 0.1 m in both the x and y directions. Monitoring starts at day 2 and ends

at day 15. Monitoring is taken every 0.1 day. For these data, we have $m = 1,440$, $n = 130$, and $N_d = 187,200$.

The candidate library is chosen as

$$\Phi(u) = \left[1, u, \frac{\partial u}{\partial x}, \frac{\partial u}{\partial y}, \frac{\partial^2 u}{\partial x^2}, \frac{\partial^2 u}{\partial y^2}, \frac{\partial^2 u}{\partial x \partial y}, u \frac{\partial u}{\partial x}, u \frac{\partial u}{\partial y}, u \frac{\partial^2 u}{\partial x^2}, u \frac{\partial^2 u}{\partial y^2}, u \frac{\partial^2 u}{\partial x \partial y}\right]. \quad (36)$$

The finite difference scheme shown in Eq. A.5 is utilized to calculate the derivatives, and L is chosen to be four. Then we utilize LASSO to learn the PDE. The change of the coefficient and the BIC along the solution path is given in Fig. 5. At the minimum of BIC, the learned equation takes the form

$$\frac{\partial u}{\partial t} = -1.000 \frac{\partial u}{\partial x} + 0.500 \frac{\partial^2 u}{\partial x^2} + 0.050 \frac{\partial^2 u}{\partial y^2}. \quad (37)$$

From Eq. 37, it can be seen that the PDE of the 2-D contaminant transport process is successfully learned.

3.4 Learning PDE from noisy data

In previous cases, the data utilized for learning PDE are clean, i.e., there is no monitoring error. However, in reality, monitoring a physical process may be associated with some noise, which may deteriorate data quality. In this subsection, we will investigate how to learn PDE from noisy data.

In this work, noise is synthetically added to the data at each monitoring location as

$$u(x, t) = u(x, t) + u_{\text{diff}}(x) \times a\% \times \varepsilon, \quad (38)$$

where $u_{\text{diff}}(x)$ denotes the maximum of the response difference at monitoring location x during the monitoring process, a denotes percentage, and ε denotes the uniform random variable taking values from -1 to 1 . Utilizing the 1-D groundwater flow problem as an illustration, Fig. 6 shows the hydraulic head at $x = 505 \text{ m}$ without noise (Fig. 6a) and with 5% noise (Fig. 6b).

Fig. 5 Change of coefficient (a) and BIC (b) along the solution path

The existence of noise may seriously deteriorate the accuracy of the calculated derivatives from finite difference. Figure 7 shows the first derivative with respect to t at $x = 505$ m (first column), the first derivative with respect to x at $t = 5$ days (second column), and the second derivative with respect to x at $t = 5$ day (third column) from the data with 0% noise (first line), 1% noise (second line), and 5% noise (third line). In these figures, for the same quantity, the scale of the vertical coordinate is different due to the large difference of quantity scale. From this figure, one can see that, for 1% noise, the results are seriously deteriorated, especially for the first derivative with respect to t and the second derivative with respect to x . In addition, the results become worse for larger noise.

The accuracy of the calculated derivatives is critical for the investigated PDE learning method. Thus, it is necessary to investigate how to obtain accurate derivatives from noisy data. Several techniques exist for derivative calculation from noisy data, such as least-squares regression, Kernel smoothing [15], Tikhonov regularization [8, 9, 19], polynomial derivative [5], and Fourier transform [14]. In this work, we used the polynomial technique for obtaining stable derivatives from noisy data, and the idea is to smooth the data first and then smooth the calculated derivatives. The procedure is given below:

1. For each monitoring location x_0 , smooth the data along t according to the following procedures:
 - (a) For each t_k , $k = 1, \dots, n$, design an interval $[t_k - n^{\text{CH}} \Delta k, t_k + n^{\text{CH}} \Delta k]$. Generate $1 + N^{\text{CH}}$ Chebyshev interpolation points t_i^{CH} , $i = 1, \dots, 1 + N^{\text{CH}}$.
 - (b) For each t_i^{CH} , $i = 1, \dots, 1 + N^{\text{CH}}$, design an interval $[t_i^{\text{CH}} - n^{\text{LS}} \Delta k, t_i^{\text{CH}} + n^{\text{LS}} \Delta k]$. Calculate the smoothed value at the Chebyshev interpolation point, $u^{\text{LS}}(x_0, t_i^{\text{CH}})$, by performing a least-squares regression with polynomial up to order N^{LS} using the data inside of the designed interval.
 - (c) Calculate the smoothed value $u^{\text{CH}}(x_0, t_k)$ by performing Chebyshev interpolation using the values $u^{\text{LS}}(x_0, t_i^{\text{CH}})$, $i = 1, \dots, 1 + N^{\text{CH}}$.
2. For each monitoring step t_0 , smooth the data along x using the procedures described in step 1.
3. Calculate the derivatives using finite difference.
4. Smooth the derivative with respect to t using the procedures described in step 1, and smooth the derivatives with respect to x using the procedures described in step 2.

For the investigated problem, we set $N^{\text{CH}} = 3$, $N^{\text{LS}} = 2$, $n^{\text{CH}} = 10$, and $n^{\text{LS}} = 10$. Figure 8 shows the smoothed

Fig. 6 Hydraulic head at $x = 505$ m with no noise (a) and with 5% noise (b)

Fig. 7 Derivatives with 0% noise: $\partial u/\partial t$ (a), $\partial u/\partial x$ (b), and $\partial^2 u/\partial x^2$ (c); derivatives with 1% noise: $\partial u/\partial t$ (d), $\partial u/\partial x$ (e), and $\partial^2 u/\partial x^2$ (f); and derivatives with 5% noise: $\partial u/\partial t$ (g), $\partial u/\partial x$ (h), and $\partial^2 u/\partial x^2$ (i)

derivatives with 1% noise (first line) and 5% noise (second line). Here, due to the fact that there is no sufficient data for smoothing near the boundaries in time and space, the derivatives near the boundary (within 10 data points) are not calculated and are not employed for learning PDE. From Fig. 8, it can be seen that the calculated derivatives after smoothing are close to the results with clean data. In addition, for larger noise, the results become worse.

We then employ LASSO to learn the PDE. The change of coefficient and BIC along the solution path are given in Fig. 9. The minimum of BIC occurs at step 11, and the corresponding learned equation takes the form

$$\frac{\partial u}{\partial t} = -0.001 \times 10^4 \frac{\partial u}{\partial x} + 0.898 \times 10^4 \frac{\partial^2 u}{\partial x^2} - 0.284 \times 10^4 u \frac{\partial^2 u}{\partial x^2}. \quad (39)$$

From Eq. 39, one can see that there are two misidentified terms, $\partial u/\partial x$ and $u \partial^2 u/\partial x^2$. The coefficient of $\partial u/\partial x$

is small, which cannot be discerned from Fig. 9a. The coefficient of $u \partial^2 u/\partial x^2$ is identified to have large values in the last four steps, while there is no obvious change of BIC values in the last four steps. By observing this, one speculation is that, for this problem, the learned PDE at the minimum of BIC may not constitute the optimal choice, since it may not be a parsimonious model, which is preferred for a PDE model. In order to choose the optimal learned PDE, cross validation is adopted here, as was done in Meng and Li [18]. For performing cross validation, the first 70% data in the temporal dimension are used as training data for learning PDE, and the remaining 30% data are used as testing data for determining the performance of the learned PDE. The average relative error of the testing data is defined as

$$\varepsilon = \frac{1}{n_{\text{test}}} \sum_{i=1}^{n_{\text{test}}} \left| \frac{u_i^p - u_i}{u_i} \right|, \quad (40)$$

Fig. 8 Smoothed derivatives from the data with 1% noise: $\partial u/\partial t$ (a), $\partial u/\partial x$ (b), and $\partial^2 u/\partial x^2$ (c); and smoothed derivatives from the data with 5% noise: $\partial u/\partial t$ (d), $\partial u/\partial x$ (e), and $\partial^2 u/\partial x^2$ (f)

where n_{test} denotes the number of testing data u_i denotes the i th testing data and u_i^p denotes the i th predicted data from the learned equation from training data. For the testing data of the investigated problem, we only use hydraulic head data that are larger than 0.05 m for calculating the relative error. Figure 10 shows the change of the average relative error along the LASSO solution path. From this figure, it can be seen that the minimum of the average relative error occurs at step 6, and the error increases after that step. At the minimum of the average relative error, the learned equation takes the form of

$$\frac{\partial u}{\partial t} = 0.985 \times 10^4 \frac{\partial^2 u}{\partial x^2}. \quad (41)$$

By comparing with Eq. 39, Eq. 41 is more parsimonious. The learned PDE shown in Eq. 41 identified the correct

terms of the inherent PDE, but there is a small error of the coefficient due to data noise. Increasing the noise level to 5%, at the minimum of the average relative error, the learned equation takes the following form:

$$\frac{\partial u}{\partial t} = 0.957 \times 10^4 \frac{\partial^2 u}{\partial x^2}. \quad (42)$$

From Eq. 42, one can see that the error of the coefficient becomes larger. Increasing the noise level to 10%, at the minimum of the average relative error, the learned equation takes the form of

$$\frac{\partial u}{\partial t} = 0.736 \times 10^4 \frac{\partial^2 u}{\partial x^2} - 1.129 \times 10^4 u \frac{\partial^2 u}{\partial x^2}. \quad (43)$$

This time, a misidentified term occurs with a larger coefficient.

Fig. 9 Change of coefficient (a) and BIC (b) along the solution path

Fig. 10 Change of the average relative error along the solution path

Using the same procedure, we continue to investigate the 1-D contaminant transport problem with noisy data. With 1%, 5%, and 10% noise, the learned PDE at the minimum of the average relative error takes the following form:

$$\frac{\partial u}{\partial t} = -1.001 \frac{\partial u}{\partial x} + 0.243 \frac{\partial^2 u}{\partial x^2}, \quad (44)$$

$$\frac{\partial u}{\partial t} = -0.977 \frac{\partial u}{\partial x} + 0.238 \frac{\partial^2 u}{\partial x^2} + 0.033u \frac{\partial^2 u}{\partial x^2} + 0.007, \quad (45)$$

and

$$\begin{aligned} \frac{\partial u}{\partial t} = & -0.018u - 0.997 \frac{\partial u}{\partial x} + 0.207 \frac{\partial^2 u}{\partial x^2} + 0.038u \frac{\partial^2 u}{\partial x^2} \\ & - 0.004 \frac{\partial^3 u}{\partial x^3} + 0.014, \end{aligned} \quad (46)$$

respectively. Similar phenomena can be found as that with 1% noise, the correct terms of the inherent PDE can be identified, but there are small errors of the coefficients. Specifically, with 5% noise, two misidentified terms appear, and the results become worse as the noise level increases.

Learning PDE from noisy data constitutes a challenge. Using the polynomial technique to obtain derivatives and performing cross validation to choose the optimal PDE model can obtain satisfactory results for small noise levels. However, it does not work well for large levels of noise. In this respect, further research is requisite.

4 Conclusions and discussion

In this work, the machine learning technique, LASSO, is employed for learning subsurface flow equations. The groundwater flow equation and the contaminant transport equation are utilized for testing the performance of the proposed method. For a series of testing cases, LASSO demonstrates satisfactory results for learning PDE from data.

For a subsurface formation, the parameter fields are usually heterogeneous. In this work, a procedure for learning the general form of the PDE with heterogeneous parameters is proposed for the first time.

Derivative calculation is needed for performing the proposed method for learning PDE. For clean data, finite difference is an appropriate choice and is simple to implement. Moreover, if the derivative can be accurately calculated, data size does not have a noticeable influence on the results. If the derivatives need to be calculated from a sparse monitoring network, a high-order finite difference scheme may assist for obtaining accurate derivatives. Calculating derivatives from noisy data constitutes a challenge. For example, a small noise level to the data can cause an enormous influence of the calculated derivative from finite difference. In this work, a procedure for calculating derivatives from noisy data by using a polynomial technique is proposed. For small noise levels, the results are acceptable, but it does not work well for large noise levels. Further work is required in this aspect.

For a learned PDE model, a balance between parsimony and accuracy should be achieved. In order to attain this, besides information criteria, such as AIC and BIC, cross validation may assist in choosing the optimal model.

This work is only at an early stage, and many further investigations are needed, such as examining the performance of the proposed method with an incomplete candidate library, studying the influence of misidentified coefficients and terms, and developing a strategy for dealing with observations of biased errors.

In this work, we also only use well-known PDE for proof of concept. In the future, we will work on complex problems without knowing their inherent PDEs, utilizing the investigated method combined with experimental work.

Acknowledgments This work is partially funded by the National Natural Science Foundation of China (Grant No. U1663208 and 51520105005) and the National Science and Technology Major Project of China (Grant No. 2017ZX05009-005 and 2016ZX05037-003). The link for the open-source Matlab code is provided in Hesterberg et al. [13]. The other computer codes and data used are available upon request from the corresponding author.

Compliance with Ethical Standards

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

Open Access This article is distributed under the terms of the Creative Commons Attribution 4.0 International License (<http://creativecommons.org/licenses/by/4.0/>), which permits unrestricted use, distribution, and reproduction in any medium, provided you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons license, and indicate if changes were made.

Appendix: A

In this appendix, the finite difference scheme for calculating the derivatives in the candidate library is provided.

The finite difference schemes for calculating the derivatives in Eq. 10 are

$$\begin{aligned}\frac{\partial u}{\partial t}(x, t_k) &= (u(x, t_k) - u(x, t_{k-1}))/\Delta t, \\ \frac{\partial u}{\partial x}(x_i, t) &= (0.5u(x_{i+1}, t) - 0.5u(x_{i-1}, t))/\Delta x, \\ \frac{\partial^2 u}{\partial x^2}(x_i, t) &= (u(x_{i+1}, t) - 2u(x_i, t) + u(x_{i-1}, t))/(\Delta x)^2, \\ \frac{\partial^3 u}{\partial x^3}(x_i, t) &= (0.5u(x_{i+2}, t) - u(x_{i+1}, t) + u(x_{i-1}, t) \\ &\quad - 0.5u(x_{i-2}, t))/(\Delta x)^3.\end{aligned}\quad (\text{A.1})$$

where Δt and Δx are the step sizes for time and space, respectively. Here, note that, to calculate the derivatives, we need the spatially or temporally nearby data. The data near the boundaries without sufficient nearby data for calculating the derivatives are not utilized for learning PDE.

The finite difference schemes for computing the derivatives in Eq. 13 are

$$\begin{aligned}\frac{\partial u}{\partial t}(x, t_k) &= (u(x, t_k) - u(x, t_{k-1}))/\Delta t, \\ \frac{\partial u}{\partial x}(x_{i,j}, t) &= (0.5u(x_{i+1,j}, t) - 0.5u(x_{i-1,j}, t))/\Delta x, \\ \frac{\partial u}{\partial y}(x_{i,j}, t) &= (0.5u(x_{i,j+1}, t) - 0.5u(x_{i,j-1}, t))/\Delta y, \\ \frac{\partial^2 u}{\partial x^2}(x_{i,j}, t) &= (u(x_{i+1,j}, t) - 2u(x_{i,j}, t) + u(x_{i-1,j}, t))/(\Delta x)^2, \\ \frac{\partial^2 u}{\partial y^2}(x_{i,j}, t) &= (u(x_{i,j+1}, t) - 2u(x_{i,j}, t) + u(x_{i,j-1}, t))/(\Delta y)^2, \\ \frac{\partial^2 u}{\partial x \partial y}(x_{i,j}, t) &= (0.5 \times (0.5u(x_{i+1,j+1}, t) - 0.5u(x_{i+1,j-1}, t)) \\ &\quad - 0.5 \times (0.5u(x_{i-1,j+1}, t) - 0.5u(x_{i-1,j-1}, t)))/\Delta x \Delta y.\end{aligned}\quad (\text{A.2})$$

where Δx and Δy are the step sizes in x and y dimension in space, respectively.

Table 3 Coefficient values for calculating the first-order derivative ($d = 1$)

	a_0	a_1	a_2	a_3	a_4
$L = 1$	0	0.5000			
$L = 2$	0	0.6667	-0.0833		
$L = 3$	0	0.7500	-0.1500	0.0167	
$L = 4$	0	0.8000	-0.2000	0.0381	-0.0036

Table 4 Coefficient values for calculating the second-order derivative ($d = 2$)

	a_0	a_1	a_2	a_3	a_4
$L = 1$	-2.0000	1.0000			
$L = 2$	-2.5000	1.3333	-0.0833		
$L = 3$	-2.7222	1.5000	-0.1500	0.0111	
$L = 4$	-2.8472	1.6000	-0.2000	0.0254	-0.0018

The finite difference scheme for calculating $\partial K/\partial y$ in Eq. 15 is

$$\begin{aligned}\frac{\partial K}{\partial y}(x_{i,j}) &= \left(2/\left(\frac{1}{K(x_{i,j+1})} + \frac{1}{K(x_{i,j})} \right) \right. \\ &\quad \left. - 2/\left(\frac{1}{K(x_{i,j})} + \frac{1}{K(x_{i,j-1})} \right) \right) / \Delta y.\end{aligned}\quad (\text{A.3})$$

The finite difference scheme for calculating $\partial(K \partial h/\partial y)/\partial y/\partial y$ is

$$\begin{aligned}&\frac{\partial}{\partial y} \left(K \frac{\partial u}{\partial y} \right) (x_{i,j}, t) \\ &= \left(\left(2/\left(\frac{1}{K(x_{i,j+1})} + \frac{1}{K(x_{i,j})} \right) \right) (u(x_{i,j+1}, t) - u(x_{i,j}, t)) \right. \\ &\quad \left. - 2/\left(\frac{1}{K(x_{i,j})} + \frac{1}{K(x_{i,j-1})} \right) (u(x_{i,j}, t) - u(x_{i,j-1}, t)) \right) / (\Delta y)^2.\end{aligned}\quad (\text{A.4})$$

For the contaminant transport problem, the finite difference scheme for calculating the d th derivative is

$$\frac{\partial^d u(x)}{\partial x^d} = \sum_{l=-L}^L \frac{a_l u(x + l \Delta x)}{(\Delta x)^d}, \quad (\text{A.5})$$

where $d > 0$ denotes the order of the derivative, L denotes the number of data points in each side of the considered location, and a_l denotes the coefficient. When d is odd, $a_{-l} = -a_l$, while when d is even, $a_{-l} = a_l$. Tables 3, 4, and 5 show the coefficient values for the first three orders of derivatives with different choice of L .

Here, note that the derivative of u with respect to t is also calculated using Eq. A.5 by replacing x with t .

Table 5 Coefficient values for calculating the third-order derivative ($d = 3$)

	a_0	a_1	a_2	a_3	a_4
$L = 2$	0	-1.0000	0.5000		
$L = 3$	0	-1.6250	1.0000	-0.1250	
$L = 4$	0	-2.0333	1.4038	-0.3000	0.0292

Appendix: B

In this appendix, we will discuss the reason that the coefficient K and $\partial K/\partial y$ in the PDE shown in Eq. 29 needs to be calculated using Eqs. 26 and A.3, respectively.

We take the term $\partial(K\partial h/\partial y)/\partial y$ for analysis. According to the equation in MODFLOW, $\partial(K\partial h/\partial y)/\partial y$ is calculated using the following formula:

$$\begin{aligned} & \frac{\partial}{\partial y} \left(K \frac{\partial u}{\partial y} \right) (x_{i,j}, t) \\ &= \left(K(x_{i,j+1/2}) \frac{\partial u}{\partial y}(x_{i,j+1/2}, t) - K(x_{i,j+1/2}) \frac{\partial u}{\partial y}(x_{i,j}, t) + K(x_{i,j+1/2}) \frac{\partial u}{\partial y}(x_{i,j}, t) \right. \\ & \quad \left. - K(x_{i,j-1/2}) \frac{\partial u}{\partial y}(x_{i,j}, t) + K(x_{i,j-1/2}) \frac{\partial u}{\partial y}(x_{i,j}, t) - K(x_{i,j-1/2}) \frac{\partial u}{\partial y}(x_{i,j-1/2}, t) \right) / \Delta y \\ &= \frac{1}{2} K(x_{i,j+1/2}) \frac{\partial^2 u}{\partial y^2}(x_{i,j+1/4}, t) + \frac{\partial K}{\partial y}(x_{i,j}) \frac{\partial u}{\partial y}(x_{i,j}, t) + \frac{1}{2} K(x_{i,j-1/2}) \frac{\partial^2 u}{\partial y^2}(x_{i,j-1/4}, t). \end{aligned} \quad (\text{B.2})$$

If $\partial^2 u/\partial y^2(x_{i,j+1/4}, t)$ and $\partial^2 u/\partial y^2(x_{i,j-1/4}, t)$ are close to $\partial^2 u/\partial y^2(x_{i,j}, t)$, we have

$$\begin{aligned} & \frac{\partial}{\partial y} \left(K \frac{\partial u}{\partial y} \right) (x_{i,j}, t) \\ &\approx \frac{1}{2} (K(x_{i,j+1/2}) + K(x_{i,j-1/2})) \frac{\partial^2 u}{\partial y^2}(x_{i,j}, t) \\ & \quad + \frac{\partial K}{\partial y}(x_{i,j}) \frac{\partial u}{\partial y}(x_{i,j}, t), \end{aligned} \quad (\text{B.3})$$

where

$$\frac{\partial K}{\partial y}(x_{i,j}) = (K(x_{i,j+1/2}) - K(x_{i,j-1/2}))/\Delta y. \quad (\text{B.4})$$

Conductivity at the interface of nearby grid blocks is calculated using the harmonic mean in MODFLOW to ensure flux continuity.

References

- Bear, J.: Dynamics of Fluids in Porous Media. New York: Environmental Science Series (1972)
- Beck, A., Teboulle, M.: A fast iterative shrinkage-thresholding algorithm for linear inverse problems. *SIAM J. Imag. Sci.* **2**(1), 183–202 (2009). <https://doi.org/10.1137/080716542>
- Bongard, J., Lipson, H.: Automated reverse engineering of nonlinear dynamical systems. *Proc. Natl. Acad. Sci. USA* **104**(24), 9943–9948 (2007). <https://doi.org/10.1073/pnas.0609476104>
- Boyd, S., Parikh, N., Chu, E., Peleato, B., Eckstein, J.: Distributed optimization and statistical learning via the alternating direction method of multipliers. *Foundations and Trends in Machine Learning* **3**(1), 1–122 (2010). <https://doi.org/10.1561/22000000016>
- Bruno, O., Hoch, D.: Numerical differentiation of approximated functions with limited order-of-accuracy deterioration. *SIAM J. Numer. Anal.* **50**(3), 1581–1603 (2012). <https://doi.org/10.1137/100805807>
- Brunton, S.L., Proctor, J.L., Kutz, J.N.: Discovering governing equations from data by sparse identification of nonlinear dynamical systems. *Proc. Natl. Acad. Sci. USA* **113**(15), 3932–3937 (2016). <https://doi.org/10.1073/pnas.1517384113>
- Chang, H., Zhang, D.: Identification of physical processes via combined data-driven and data-assimilation methods. *J. Comp. Phys.* **393**, 337–350 (2019). <https://doi.org/10.1016/j.jcp.2019.05.008>
- Chartrand, R.: Numerical differentiation of noisy, nonsmooth data. *ISRN Applied Mathematics* **2011**, 1–11 (2011). <https://doi.org/10.5402/2011/164564>
- Cullum, J.: Numerical differentiation and regularization. *SIAM J. Numer. Anal.* **8**(2), 254–265 (1971). <https://doi.org/10.1137/0708026>
- Efron, B., Hastie, T., Johnstone, I., Tibshirani, R.J.: Least angle regression. *Ann. Stat.* **32**(2), 407–451 (2004). <https://doi.org/10.1214/009053604000000067>
- Figueiredo, M.A.T., Nowak, R.D., Wright, S.J.: Gradient projection for sparse reconstruction: application to compressed sensing and other inverse problems. *IEEE J. Sel. Top. Sign. Proces.* **1**(4), 586–597 (2007). <https://doi.org/10.1109/JSTSP.2007.910281>
- Hastie, T., Tibshirani, R.J., Friedman, J.H.: The elements of statistical learning: data mining, inference, and prediction. New York: Springer series in statistics. <https://doi.org/10.1007/978-0-387-84858-7> (2009)
- Hesterberg, T., Choi, N.H., Meier, L., Fraley, C.: Least angle and l1 penalized regression: a review. *Statistics Surveys* **2**, 61–93 (2008). <https://doi.org/10.1214/08-SS035>
- Jauberteau, F., Jauberteau, J.L.: Numerical differentiation with noisy signal. *Appl. Math. Comput.* **215**(6), 2283–2297 (2009). <https://doi.org/10.1016/j.amc.2009.08.042>
- Knowles, I., Le, T., Yan, A.: On the recovery of multiple flow parameters from transient head data. *J. Comput. Appl. Math.* **169**(1), 1–15 (2004). <https://doi.org/10.1016/j.cam.2003.10.013>
- Mangan, N.M., Brunton, S.L., Proctor, J.L., Kutz, J.N.: Inferring biological networks by sparse identification of nonlinear dynamics. *IEEE Transactions on Molecular Biological and Multi-Scale Communications* **2**(1), 52–63 (2016). <https://doi.org/10.1109/TMBMC.2016.2633265>
- Mangan, N.M., Kutz, J.N., Brunton, S.L., Proctor, J.L.: Model selection for dynamical systems via sparse regression and information criteria. *Proceedings of the Royal Society A-Mathematical Physical and Engineering Sciences* **473**(2204), 16 (2017). <https://doi.org/10.1098/rspa.2017.0009>

18. Meng, J., Li, H.: An efficient stochastic approach for flow in porous media via sparse polynomial chaos expansion constructed by feature selection. *Adv. Water Resour.* **105**, 13–28 (2017). <https://doi.org/10.1016/j.advwatres.2017.04.019>
19. Ramos, G., Carrera, J., Gómez, S., Minutti, C., Camacho, R.: A stable computation of log-derivatives from noisy drawdown data. *Water Resour. Res.* **53**(9), 7904–7916 (2017). <https://doi.org/10.1002/2017WR020811>
20. Rosset, S., Zhu, J.: Piecewise linear regularized solution paths. *Ann. Stat.* **35**(3), 1012–1030 (2007). <https://doi.org/10.1214/0090536060000001370>
21. Rudy, S.H., Brunton, S.L., Proctor, J.L., Kutz, J.N.: Data-driven discovery of partial differential equations. *Sci. Adv.* **3**(4), e1602614 (2017). <https://doi.org/10.1126/sciadv.1602614>
22. Schaeffer, H.: Learning partial differential equation via data discovery and sparse optimisation. *Proceedings of the Royal Society A-Mathematical Physical and Engineering Sciences* **473**(2197), 20160446 (2017). <https://doi.org/10.1098/rspa.2016.0446>
23. Schmidt, M., Lipson, H.: Distilling free-form natural laws from experimental data. *Science* **324**(5923), 81–85 (2009). <https://doi.org/10.1126/science.1165893>
24. Tibshirani, R.J.: The lasso problem and uniqueness. *Electronic Journal of Statistics* **7**(1), 1456–1490 (2013). <https://doi.org/10.1214/13-EJS815>
25. Zou, H.: The adaptive Lasso and its oracle properties. *J. Am. Stat. Assoc.* **101**(476), 1418–1429 (2006). <https://doi.org/10.1198/016214506000000735>

Publisher's note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.