Journal of Computational Physics 348 (2017) 694–714

Contents lists available at ScienceDirect

Journal of Computational Physics

www.elsevier.com/locate/jcp

Discovering variable fractional orders of advection–dispersion equations from field data using multi-fidelity Bayesian optimization

Guofei Pang [a], Paris Perdikaris [b], Wei Cai [c], George Em Karniadakis [d,*]

[a] Algorithms Division, Beijing Computational Science Research Center, Beijing 100193, China
[b] Department of Mechanical Engineering, Massachusetts Institute of Technology, Cambridge, MA 02139, USA
[c] Department of Mathematics, Southern Methodist University, Dallas, TX 75275, USA
[d] Division of Applied Mathematics, Brown University, Providence, RI 02912, USA

ARTICLE INFO

Article history:
Received 7 May 2017
Received in revised form 1 July 2017
Accepted 27 July 2017
Available online 2 August 2017

Keywords:
Fractional modeling
Porous media
Machine learning
Inverse problem
Gaussian process regression
Uncertainty quantification
Model uncertainty

ABSTRACT

The fractional advection–dispersion equation (FADE) can describe accurately the solute transport in groundwater but its fractional order has to be determined *a priori*. Here, we employ multi-fidelity Bayesian optimization to obtain the fractional order under various conditions, and we obtain more accurate results compared to previously published data. Moreover, the present method is very efficient as we use different levels of resolution to construct a stochastic surrogate model and quantify its uncertainty. We consider two different problem set ups. In the first set up, we obtain variable fractional orders of one-dimensional FADE, considering both synthetic and field data. In the second set up, we identify constant fractional orders of two-dimensional FADE using synthetic data. We employ multi-resolution simulations using two-level and three-level Gaussian process regression models to construct the surrogates.

1. Introduction

The fractional advection–dispersion equation (FADE) is an effective mathematical model in describing the solute transport in porous and fractured media, such as contaminant transport in aquifers [1–3]. However, being a phenomenological model, the FADE requires parameter calibration before being applied to predict concentration profiles, namely, the phenomenological parameters, fractional derivative orders, have to be determined by fitting the experimental data. As a result, a parameter identification inverse problem needs to be solved. Due to the high computational cost of solving forward problems using fractional discretization multiple times, sometimes hundreds of times, reaching a converged inverse solution presents a serious computational challenge. Recent efforts for fractional order identification have mainly focused on time-fractional advection-dispersion equations (ADEs) [4–9] and, to a lesser extent, on parameter identification of the space-fractional ADEs but only limited to constant fractional order [10–12]. The FADEs for describing the real-world solute transport, however, are often complex [1] and involve 2-D or 3-D spatial fields as well as variable fractional orders. Thus, the parameter identification of these FADEs is more challenging than those previously considered in the literature.

* Corresponding author.
E-mail address: george_karniadakis@brown.edu (G.E. Karniadakis).

http://dx.doi.org/10.1016/j.jcp.2017.07.052
0021-9991/

To the best of our knowledge, parameter identification of high-dimensional and variable-order FADEs has not yet been investigated using a systematic method. Ref. [1] identifies the fractional orders by trial and error approach, which is feasible for identification of a small number of parameters, but becomes intractable for a large number of parameters as it is extremely time-consuming to transverse the whole parameter space. In this paper, to reduce the computational time, we apply the multi-fidelity model inversion method [13] to identify the fractional orders in such challenging FADEs. Numerical results show that not only the computational overhead is considerably reduced but also the parameters identified by our method lead to more accurate results than those estimated by the trial and error approach in ref. [1]. Specifically, the model inversion method we adopt is based on multi-fidelity Gaussian process regression (multi-fidelity GPR) and Bayesian optimization [13]. The method targets the accurate construction of response surfaces in parameter space, and an efficient pursuit to identify global optima while keeping the number of expensive function evaluations at a minimum. The method has been successfully applied to the calibration of outflow boundary conditions in blood flow simulations, as well as the parameter estimation in elliptic PDEs [13]. In the current paper, we extend the method to parameter identification of FADEs and we also use available field data in addition to synthetic data.

The outline of this paper is as follows. Section 2 introduces the basic ideas behind the multi-fidelity GPR as well as the Bayesian optimization which constitute the model inversion framework. For more details we refer the readers to ref. [13]. Section 3 gives a description of the parameter identification problem to be solved, followed by numerical results on parameter identification of 1-D variable-order and 2-D multiscaling constant-order FADEs. Discussion and conclusion are given in the last two sections. In the first three Appendices we include some more details of the convergence of the model inversion method, and in the last Appendix, we briefly introduce the implementation of the random walk method for solving the forward problem of 2-D FADEs.

2. Methodology

The section introduces the basic building blocks of the model inversion method. The multi-fidelity Gaussian process regression constructs a fast-evaluation response surface for the original expensive system function (or input-output mapping) using limited computational resources, and the Bayesian optimization adaptively searches the resulting response surface with the help of acquisition function and rapidly finds the optimum of the original system function.

2.1. Multi-fidelity GPR

A multi-variate Gaussian random vector $\mathbf{v}_N$ of N components corresponds to the probability density function

$$p(\mathbf{x}) = \frac{1}{(2\pi)^{\frac{N}{2}}|\mathbf{K}|^{\frac{1}{2}}} exp\{-\frac{1}{2}(\mathbf{x}-\mathbf{m})^T\mathbf{K}^{-1}(\mathbf{x}-\mathbf{m})\}, \tag{1}$$

where $\mathbf{m}$ and $\mathbf{K}$ are the mean vector and the covariance matrix, respectively. The index set for the random vector $\mathbf{v}_N = [v_1, v_2, \cdots, v_N]^T$ is a finite number set $\{1, 2, \cdots, N\}$. By extending the finite index set to an infinite continuous set, say $\Omega \subset \mathbf{R}^d$, we have the Gaussian process (GP),

$$f(\mathbf{x}) \sim GP(m(\mathbf{x}), k(\mathbf{x}, \mathbf{x}')), \tag{2}$$

where $\mathbf{x} \in \Omega$ is the index and for the fixed $\mathbf{x}$, $f(\mathbf{x})$ is a Gaussian random variable; $m(\cdot)$ and $k(\cdot,\cdot)$ are the mean and the covariance functions, respectively. On any finite subset of the index set, $\{\mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_N\}$, the corresponding random variables form a multi-variate Gaussian random vector, namely, $\mathbf{f}_N = [f(\mathbf{x}_1), f(\mathbf{x}_2), \cdots, f(\mathbf{x}_N)]^T$.

GP regression constructs the response surface based on the known input-output pairs (also known as training data). Suppose we have obtained N training data points, and the training data set $\mathbf{D}$ is given by

$$\mathbf{D} = \{(\mathbf{x}_1, f(\mathbf{x}_1)), (\mathbf{x}_2, f(\mathbf{x}_2)), \cdots, (\mathbf{x}_N, f(\mathbf{x}_N))\}, \tag{3}$$

where $f(\mathbf{x})$ for the fixed index $\mathbf{x}$ is the output of the system of interest when the system input is the very $\mathbf{x}$. Generally, the system output is contaminated by noise, and the final observation is actually

$$y(\mathbf{x}_i) = f(\mathbf{x}_i) + \epsilon_i, i = 1, 2, \cdots, N. \tag{4}$$

To simplify the problem, Gaussian white noise is usually considered, namely $\boldsymbol{\epsilon}_N \sim \mathcal{N}(\mathbf{0}, \sigma_n^2\mathbf{I})$ where $\boldsymbol{\epsilon}_N$ is the vector of noise on all training data, $\mathcal{N}(\cdot,\cdot)$ is the normal distribution, σ_n^2 is the noise variance, and $\mathbf{I}$ is the identity matrix.

The goal of GPR is to predict the output f at arbitrary test input $\mathbf{x}_t$. Actually, this can be easily done by using the conditional distribution property for a multi-variate Gaussian random vector. We write the joint distribution of the test output $f(\mathbf{x}_t)$ and the noisy training data as a $(N+1)$-variate Gaussian random vector

$$\begin{pmatrix} f(\mathbf{x}_t) \\ y_(\mathbf{x}_1) \\ \vdots \\ y_(\mathbf{x}_N) \end{pmatrix} \sim \mathcal{N}\left(\begin{bmatrix} m(\mathbf{x}_t) \\ m(\mathbf{x}_1) \\ \vdots \\ m(\mathbf{x}_N) \end{bmatrix}, \begin{bmatrix} k(\mathbf{x}_t,\mathbf{x}_t) & k(\mathbf{x}_t,\mathbf{x}_1) & \dots & k(\mathbf{x}_t,\mathbf{x}_N) \\ k(\mathbf{x}_1,\mathbf{x}_t) & k(\mathbf{x}_1,\mathbf{x}_1)+\sigma_n^2 & \dots & k(\mathbf{x}_1,\mathbf{x}_N) \\ \vdots & \vdots & \ddots & \vdots \\ k(\mathbf{x}_N,\mathbf{x}_t) & k(\mathbf{x}_N,\mathbf{x}_1) & \dots & k(\mathbf{x}_N,\mathbf{x}_N)+\sigma_n^2 \end{bmatrix} \right), \tag{5}$$

or in the matrix form,

$$\begin{pmatrix} f(\mathbf{x}_t) \\ \mathbf{y}_N \end{pmatrix} \sim \mathcal{N}\left(\begin{bmatrix} m(\mathbf{x}_t) \\ \mathbf{m}_N \end{bmatrix}, \begin{bmatrix} k(\mathbf{x}_t, \mathbf{x}_t) & \mathbf{k}^T \\ \mathbf{k} & \mathbf{K} + \sigma_n^2 \mathbf{I} \end{bmatrix} \right). \tag{6}$$

Then, the conditional distribution of $f(\mathbf{x}_t)$ given $\mathbf{y}_N$ is also a Gaussian distribution whose mean and variance are, respectively, (see Eq. (A.6) of [14]),

$$m_*(\mathbf{x}_t) = m(\mathbf{x}_t) + \mathbf{k}^T (\mathbf{K} + \sigma_n^2 \mathbf{I})^{-1} (\mathbf{y}_N - \mathbf{m}_N), \tag{7}$$

$$\sigma_*^2(\mathbf{x}_t) = k(\mathbf{x}_t, \mathbf{x}_t) - \mathbf{k}^T (\mathbf{K} + \sigma_n^2 \mathbf{I})^{-1} \mathbf{k}. \tag{8}$$

The predictive mean $m_*(\mathbf{x}_t)$ gives the response surface for any $\mathbf{x}_t \in \Omega$, and furthermore, the predictive variance reflects the uncertainty of the approximation $m_*(\mathbf{x}_t)$. Before using the predictive formulas (7) and (8), some parameters in these formulas should be first learned by training the GP model. These parameters include the trend coefficients in the prior mean function $m(\mathbf{x})$ (for simplicity, we usually consider a constant coefficient, $m(\mathbf{x}) \equiv \beta$; a general prior mean is given by the basis function expansion $m(\mathbf{x}) = \sum_j \beta_j \phi_j(\mathbf{x})$ where $\phi_j(\cdot)$ is the basis function, say, a polynomial), the signal variance σ^2 in the prescribed covariance function $k(\mathbf{x}, \mathbf{x}') = \sigma^2 r(\mathbf{x}, \mathbf{x}')$ where $r(\cdot, \cdot)$ is the correlation function, the noise variance σ_n^2, and the hyper-parameters $\boldsymbol{\theta}$ in the covariance function $k(\mathbf{x}, \mathbf{x}')$. (The parameters θ_i in Eq. (15) are examples of hyper-parameters.) The trend coefficient and the signal variance can be analytically deduced by using maximal likelihood estimation, while the noise variance and the hyper-parameters have to be optimized by maximizing the marginal likelihood function. We refer the readers to ref. [13] for seeing how these parameters are obtained. In this paper, we use the multi-started conjugate gradient method [15] to optimize the marginal likelihood function.

The predictive distribution (7) and (8) can also be deduced by using Bayes' rule, as given in Sec. 4.2 of [16]. By further integrating the posterior distribution of trend coefficient β in the predictive distribution, we have the updated predictive distribution (Eqs. (1.20) and (1.21) of [17])

$$m_*(\mathbf{x}_t) = \hat{\beta} + \mathbf{r}^T (\mathbf{R}_{\hat{\theta}} + \hat{\sigma}_n^2 \mathbf{I})(\mathbf{y}_N - \mathbf{1}\hat{\beta}), \tag{9}$$

$$\sigma_*^2(\mathbf{x}_t) = \hat{\sigma}^2 \left[1 - \mathbf{r}^T (\mathbf{R}_{\hat{\theta}} + \hat{\sigma}_n^2 \mathbf{I}) \mathbf{r} + \frac{[1 - \mathbf{r}^T (\mathbf{R}_{\hat{\theta}} + \hat{\sigma}_n^2 \mathbf{I})^{-1} \mathbf{r}]^2}{\mathbf{1}^T (\mathbf{R}_{\hat{\theta}} + \hat{\sigma}_n^2 \mathbf{I})^{-1} \mathbf{1}} \right], \tag{10}$$

where $\mathbf{1}$ is the vector of ones with length N. The "hat" parameters above are estimated from the training data as mentioned in the preceding paragraph. Here, $\mathbf{R}_{\hat{\theta}}$ is the correlation matrix and the subscript $\hat{\boldsymbol{\theta}}$ indicates the dependence on the hyper-parameters. Note that the estimated noise variance is $\hat{\sigma}^2 \hat{\sigma}_n^2$ rather than $\hat{\sigma}_n^2$, and we put $\hat{\sigma}_n^2$ together with $\mathbf{R}_{\hat{\theta}}$ in order to estimate $\hat{\sigma}_n^2$ simultaneously with the hyper-parameters. It should be noted that in calibrating the hyper-parameters and $\hat{\sigma}_n^2$, the matrix inversion $(\mathbf{R}_{\hat{\theta}} + \sigma_n^2 \mathbf{I})^{-1}$ using the Cholesky decomposition requires cubic computational cost $O(N^3)$, which may lead to a serious computational bottleneck for moderately big datasets. Several studies [18,19,14] have been devoted to reducing the training computational cost. In the current paper, we still adopt the standard GPR without any acceleration since the training dataset we consider is not large.

Kennedy and Hagan [20] first proposed the multi-fidelity GPR as an extension of the standard aforementioned GPR. They applied the multi-fidelity GPR to prediction and uncertainty analysis for the outputs of complex computer codes which can be run at different levels of sophistication. However, the corresponding GPR model results in large covariance matrix when the number of fidelity levels is large. To reduce the matrix dimension and thus the computational cost, LeGratiet [21] presented a recursive multi-fidelity GPR, which allowed one to train the coupled GPRs separately. In the paper, we will adopt this improved multi-fidelity GPR model. The auto-regressive scheme of the recursive multi-fidelity GPR reads

$$y_t(\mathbf{x}) = \rho_{t-1} y_{t-1}(\mathbf{x}) + \delta_t(\mathbf{x}), t = 2, \cdots, s, \tag{11}$$

where ρ is the constant scaling factor between two successive GPR models, and $\delta_t(\mathbf{x})$ is another Gaussian process with signal variance σ_t^2 and noise variance $\sigma_{n_t}^2$, which describes the bias between two successive GPR models. It is trivial to show that $y_1(\mathbf{x}) \equiv \delta_1(\mathbf{x})$. $y_s(\mathbf{x})$ represents the output of the most accurate and costly computer code to be surrogated and $y_1(\mathbf{x}), y_2(\mathbf{x}), \cdots, y_{s-1}(\mathbf{x})$ are the cheaper version of it with $y_1(\mathbf{x})$ the cheapest and least accurate one. Analogous to training the single-fidelity GPR model, one can train the GPR models in model (11) successively. Differently, one needs to estimate an extra parameter ρ that couples the two successive GPR models. To guarantee that the recursive formulation is valid, the training input sets in different fidelity levels must be nested, i.e. $\mathbf{D}_s^x \subset \mathbf{D}_{s-1}^x \subset \cdots \subset \mathbf{D}_1^x$ where $\mathbf{D}_t^x := \{\mathbf{x}_1^t, \cdots, \mathbf{x}_{N_t}^t\}$ with N_t being the number of training data in the t-th fidelity level. It should be noted that the Markov property assumption for two successive fidelity levels (Assumption (1) in Kennedy and Hagan's paper [20]), together with the stationarity of $y_t(\mathbf{x})$ over the input space for each t, implies precisely the auto-regressive scheme (11) where $\delta_t(\mathbf{x})$ is assumed to be independent of $y_{t-1}(\cdot), \cdots, y_1(\cdot)$. This is the motivation for adopting the aforementioned auto-regressive scheme.

The predictive distribution of multi-fidelity GPR can be seen in Eqs. (2.5) and (2.6) of [13], and the parameter estimation can also be found in the same reference.

Table 1
Procedure for the model inversion method.

Step 1: Generate the training data sets $\mathbf{D}_i, i = 1, 2, \cdots, s$ and select the covariance kernel $k(\mathbf{x}, \mathbf{x}')$
Step 2: Construct surrogate model by $(m_{*s}(\mathbf{x}), \sigma^2_{*s}(\mathbf{x})) = MGPR(\mathbf{D}_i, k(\mathbf{x}, \mathbf{x}'))$, where m_{*s} and σ^2_{*s} are the predictive mean and variance for the s-th fidelity level, and $MGPR$ is the function constructing the multi-fidelity GP model.
Step 3: Find the next input-output pair $(\mathbf{x}_{new}, y_{new}) = BayesOpt(\mathbf{D}_s, m_{*s}(\mathbf{x}), \sigma^2_{*s}(\mathbf{x}))$ where $\mathbf{x}_{new}$ is the location of the maximum of the acquisition function, $y_{new} = e(\mathbf{x}_{new})$ ($e(.)$ is the true system function defined by Eq. (20)), and $BayesOpt$ is the function implementing Bayesian optimization.
Step 4: Judge whether or not the terminal condition is satisfied. If yes, the identified parameter is $\mathbf{x}_0 = argmin_{\mathbf{x}\in\{\mathbf{x}_1,\cdots,\mathbf{x}_N\}} y(\mathbf{x})$, and the algorithm is terminated; otherwise add $(\mathbf{x}_{new}, y_{new})$ to $\mathbf{D}_i, i = 1, \cdots, s$ and return to **Step 2**.

2.2. Bayesian optimization

The multi-fidelity GPR not only predicts outputs on unknown inputs, but also gives the uncertainty of the predictions because it provides a predictive probability distribution of the test output for each test input. The posterior mean and variance can be delivered to a Bayesian optimizer. It is the acquisition function [22] in the optimizer that accepts the predictive distribution information. Here, we consider the *expected improvement* acquisition function, which is frequently used in literature. Since directly minimizing the predictive mean surface (also known as response surface) could be misleading due to the possible inaccuracy of the response surface, instead, minimizing the acquisition function is a better choice [22]. The acquisition function for the s-th level of fidelity reads

$$EI_s(x) = [\min(\mathbf{y}_N) - m_{*s}(\mathbf{x})]\Phi\left(\frac{\min(\mathbf{y}_N) - m_{*s}(\mathbf{x})}{\sigma_{*s}(\mathbf{x})}\right) + \sigma_{*s}(\mathbf{x})\phi\left(\frac{\min(\mathbf{y}_N) - m_{*s}(\mathbf{x})}{\sigma_{*s}(\mathbf{x})}\right), \tag{12}$$

where $\Phi(\cdot)$ and $\phi(\cdot)$ are the standard normal cumulative distribution and density functions, respectively, and $m_{*s}(\mathbf{x})$ and $\sigma_{*s}(\mathbf{x})$ are the predictive mean and standard deviation, respectively; $\mathbf{y}_N$ is the N training outputs in the s-th fidelity level. Note that we only calculate the acquisition function for the highest fidelity level.

Suppose at present we have N training data in the s-th fidelity level, $\mathbf{D}_s = \{\mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_N, \mathbf{y}_N\}$, and the next data point $(\mathbf{x}_{N+1}, y(\mathbf{x}_{N+1}))$ to be added into the $\mathbf{D}_s$ is determined by the maximum of the acquisition function, namely

$$\mathbf{x}_{N+1} = \arg\max_{\mathbf{x}\in\Omega}\{EI_s(\mathbf{x})\}, \tag{13}$$

which implies that we expect to achieve the largest improvement on $\mathbf{x}_{N+1}$. The first adding term on the right-hand side of Eq. (12) favors the x_{N+1} such that the predictive mean $m_*(x_{N+1})$ is smaller than the current minimum, while the second term favors the x_{N+1} such that the predictive variance $\sigma^2_{*s}(x_{N+1})$ is larger. Thus, the expected improvement function actually represents a trade-off between finding the minimum and reducing the uncertainty. It should be noted that the modified Lipschitzian optimization algorithm (namely the DIRECT algorithm) [23] is employed to maximize the expected improvement acquisition function.

We add consecutively a new training data point in each Bayesian optimization iteration until a terminal condition is satisfied. The terminal condition can be that the minimum among all the training outputs, or that $\min(\mathbf{y}_N)$ is smaller than a tolerance, or that a certain number of iterations has been reached, or that the distance between two consecutive optimal training inputs $\mathbf{x}$ is less than a tolerance, or that the maximum of the acquisition function is less than a tolerance. Note that since the training input sets for all fidelity levels must be nested to achieve computational efficiency, one should add the new training data point simultaneously to all the training data sets. The model inversion method combining the multi-fidelity GPR and the Bayesian optimization is briefly summarized in Table 1.

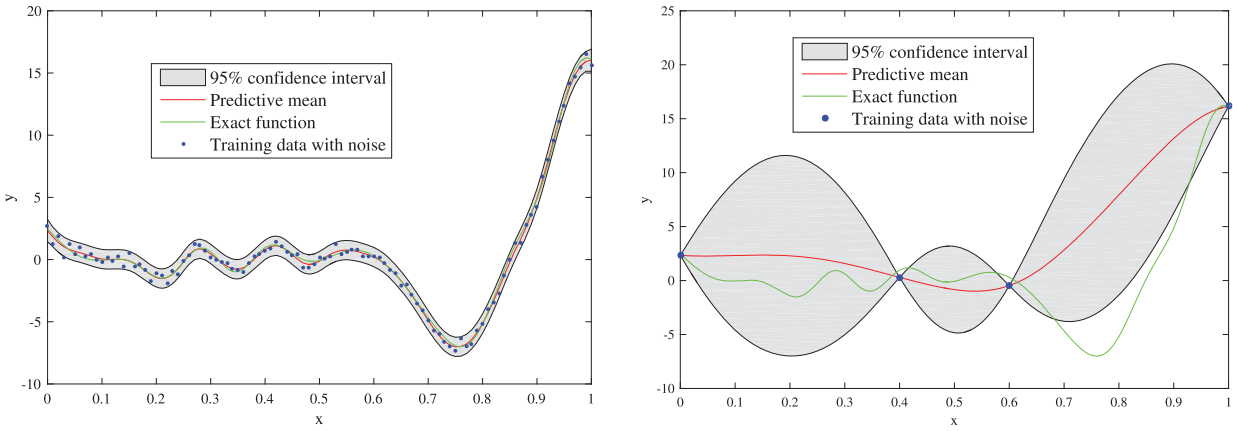
It should be noted that if the noise ϵ is significant, the expected improvement function is not a good choice. To smooth out the noise fluctuation, one can consider its enhanced version, namely the *expected quantile improvement function* [24]. However, for the numerical examples we consider in this paper, the noise can be neglected since the ratio of the learned noise variance to the learned signal variance is smaller than 1%.

2.3. An illustrative example

This subsection introduces how the model inversion method works through considering a pedagogical example. For all the numerical examples in this paper, the covariance function $k(\mathbf{x}, \mathbf{x}')$ is taken as the Matern 5/2-order automatic relevance determination function,

$$k(\mathbf{x}, \mathbf{x}') = \sigma^2\left(1 + h + \frac{h^2}{3}\right)e^{-h}, \tag{14}$$

where



(a) 100 noisy training data. (b) 4 noisy training data.

Fig. 1. Pedagogical example: Single-fidelity GPR response surfaces and uncertainty estimation for high-fidelity function $f_2(x)$. (The upper and lower bounds of the band area are determined by the predictive mean plus two predictive standard derivations.)

$$h = \sqrt{\sum_{i=1}^{d} \frac{5(x_i - x_i')^2}{\theta_i^2}}. \tag{15}$$

Here, d is the dimension of the input vector $\mathbf{x}$, and θ_i is the i-th hyper-parameter, which is also called characteristic length. The selection of the covariance kernel $k(\mathbf{x}, \mathbf{x}')$ of the GP model reflects our prior belief on the structure, regularity and other intrinsic properties of the quantity of interest, say the system function. The kernel considered in this paper, namely the Matern covariance kernel, is frequently used in GPR because of its moderate smoothness. The squared exponential (or Gaussian) kernel is another frequently used kernel due to its simple analytical expression. In some cases, we have to construct new kernels from the existing ones [14] in order to characterize more complicated system functions. Alternatively, the kernel can also be automatically selected, e.g., by learning the covariance matrix via hierarchical Bayes presented in ref. [25].

Suppose that the functions $f_2(x)$ and $f_1(x)$ represent the exact input-output mapping for the high-fidelity and the low-fidelity levels. They are given by

$$f_1(x) = 0.5((6x-2)^2 \sin(12x-4)) + 10(x-0.5) - 5, \tag{16}$$

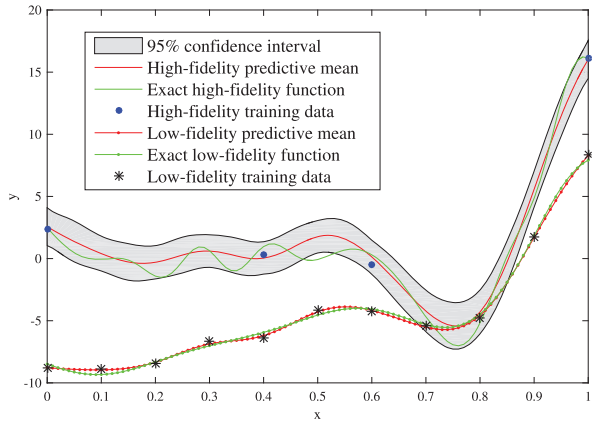
$$f_2(x) = 2f_1(x) - 20x + 20 + \sin(10\cos(5x)), x \in [0,1]. \tag{17}$$

Our goal is to find the minimum of the high-fidelity function $f_2(x)$ based on a limited number of training data. Suppose that the system functions $f_1(x)$ and $f_2(x)$ are both corrupted by a Gaussian white noise with $\sigma_n^2 = 0.16$.

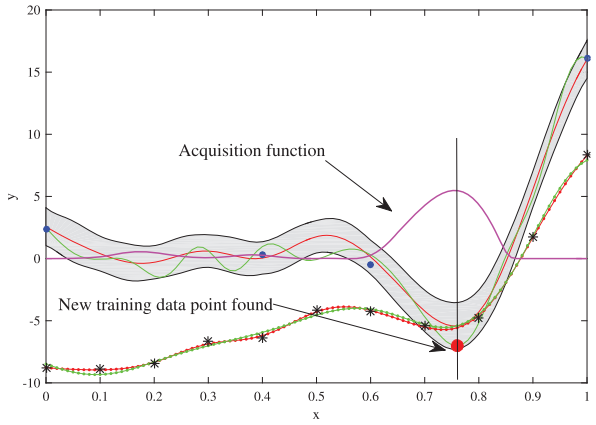
Before proceeding to our goal, we first see whether the method can learn the noise from the training data. Fig. 1(a) shows the predictive mean and variance from the single-fidelity GPR for $f_2(x)$; here 100 noisy training data for $f_2(x)$ are assigned. It is observed that despite the noise, the predictive mean still fits the exact function well, and furthermore the posterior variance, through the 95% confidence interval, can well capture the variation of the training data. Here, the estimated noise variance is 0.12, which is close to the synthetic noise variance 0.16.

Assuming the calculation of $f_2(x)$ is extremely time-consuming, we thus only have 4 training data rather than 100 data. First, we construct the response surface using single-fidelity GPR based on the 4 data. The result is given in Fig. 1(b). The predictive mean deviates a lot from the exact function since few training data are available. The accuracy of the predictive mean can be considerably improved by adding another 11 training data from the low-fidelity level whose system function is $f_1(x)$. Fig. 2(a) shows the surrogate accuracy improvement. Although the low-fidelity function is inaccurate compared to the high-fidelity one, through automatically learning the scale factor and the bias between the two fidelity levels, one can recover the high-fidelity system function accurately. Another observation from Fig. 2(a) is that the uncertainty has been reduced due to the complementary information given by the low-fidelity. Generally, the training data in the lower fidelity levels should be much more than that in the highest level. The information extracted from the lower fidelity levels is much cheaper than that extracted from the highest level since the numerical simulations of lower fidelity levels are much faster.

It should be noted that the term *fidelity* can refer to the accuracy of the computer code or simulator compared to the exact system function it approximates [24]. A large range of fidelity is available by tuning the factors that controls the complexity of the simulator. For instance, adjusting the resolution of a finite element solver yields simulations of different accuracy. Alternatively, *fidelity* can also be characterized as the accuracy of the modeling approaches. For example, in our previous research [13], the following three models are employed to simulate blood flow in an arterial bifurcation: (i) 3-D Navier–Stokes models in rigid domains, (ii) nonlinear 1-D fluid-structure interaction models, and (iii) linearized 1-D fluid-structure interaction models.



(a) 4 noisy training data for high-fidelity level and 11 noisy data for low-fidelity level.



(b) Expected improvement function and the next training data point found after one Bayesian optimization iteration.

Fig. 2. Pedagogical example: Two-fidelity GPR response surfaces, uncertainty estimation, and the acquisition function.

Table 2
Pedagogical example: Parameters learned by training the two-fidelity GPR model with 4 noisy high-fidelity training data and 11 noisy low-fidelity training data. (The synthetic noisy variance is 0.16 for both fidelity levels.)

Types	Parameters	Estimation by training
Low-fidelity signal variance	$\hat{\sigma}_1^2$	59.83
High-fidelity bias signal variance	$\hat{\sigma}_2^2$	0.0101
High-fidelity signal variance	$\hat{\rho}_1^2\hat{\sigma}_1^2 + \hat{\sigma}_2^2$	208.33
Low-fidelity noise variance	$\hat{\sigma}_{n1}^2\hat{\sigma}_1^2$	0.1195
High-fidelity noise variance	$\hat{\rho}_1^2\hat{\sigma}_{n1}^2\hat{\sigma}_1^2 + \hat{\sigma}_{n2}^2\hat{\sigma}_2^2$	0.1778
Low-fidelity trend coefficient	$\hat{\beta}_1$	−1.58
High-fidelity trend coefficient and the scale factor	$(\hat{\beta}_2^1, \hat{\beta}_2^2, \hat{\rho}_1)$	(18.92, −18.36, 1.866)
Low-fidelity hyper-parameter	$\hat{\theta}_1$	0.3141
High-fidelity hyper-parameter	$\hat{\theta}_2$	0.0075

To fulfill our goal of finding the minimum of the high-fidelity function $f_2(x)$, one just needs to compute the acquisition function (12) based on the predictive distribution, and then find its maximum location. Fig. 2(b) displays the acquisition function computed as well as the maximum location found. It is seen that only after one Bayesian optimization iteration, the maximum location we find is already very close to the minimum location of the $f_2(x)$. It should be noted that, however, for a high-dimensional index set Ω (i.e. larger d), due to the curse of dimensionality, it takes generally tens of Bayesian optimization iterations to locate the minimum of the high-fidelity function.

The estimated parameters obtained by training the two-fidelity GPR modes are tabulated in Table 2. Here we assume a linear trend having two coefficients β_2^1, β_2^2 for the high-fidelity level because the exact bias between the two fidelity levels include a linear term $20 - 20x$. Generally, if the exact system function $f_2(x)$ is not known beforehand, a constant trend coefficient β is sufficient to capture the basic trend. It can be learned from Table 2 that the estimated noise variance for the two fidelity levels, 0.1195 and 0.1778, basically agrees with the synthetic variance 0.16. The high-fidelity trend coefficients 18.92 and −18.36 also agree with the exact coefficients 20 and −20. The estimated scale factor 1.866 is close to the exact scale factor 2.

In the following, we will consider some partial different equation (PDE) systems whose system functions are analytically unavailable. These system functions are also expensive to evaluate, indicating that we can obtain only a limited number of training data. The index set (i.e., the input space) will be regarded as the continuous space formed by the equation parameters, and the system function value is defined by the relative error between the PDE numerical solutions and the field or the synthetic data. Our goal will change into finding the optimal parameters that minimize the relative error.

3. Problem definition

In the paper, we consider two types of FADEs. The first equation is the 1-D variable-order FADE, as given by [1]

$$\frac{\partial^{\gamma(x,t)}C(x,t)}{\partial t^{\gamma(x,t)}} = -v\frac{\partial C(x,t)}{\partial x} + D_+\frac{\partial^{\alpha(x,t)}C(x,t)}{\partial x^{\alpha(x,t)}} + D_-\frac{\partial^{\alpha(x,t)}C(x,t)}{\partial(-x)^{\alpha(x,t)}}, x \in \mathcal{R}, \tag{18}$$

where $C(x,t)$ is the solute concentration in groundwater at space-time coordinate (x,t), v is mean-flow velocity, and D_+ and D_- are left-sided and right-sided dispersion coefficients, respectively. The time-fractional order and the space-fractional order can be space-time dependent. In ref. [1], Eq. (18) was employed to describe the tritium transport through a highly heterogeneous medium at the Macrodispersion Experimental (MADE) site at Columbus Air Force Base. Compared to the constant-order model, the variable-order model can better describe the dispersion state transition among normal, sub- and super-dispersion. The finite difference method (FDM) developed in ref. [1] was used to approximate the fundamental solution of Eq. (18). The spatial-temporal domain is restricted to be $t \in [0, 330], x \in [-20, 350]$. A homogeneous Dirichlet boundary condition was assumed at $x = -20$ and $x = 350$ in order to truncate the infinite spatial domain. The instantaneous point source is located at $x = 0$. Ref. [1] considers three cases for time- and space-fractional orders and in this paper we extend them to more generalized ones:

1. **Constant order**: γ and α are assumed to be constants, namely $\gamma = \gamma_0$ and $\alpha = \alpha_0$;
2. **Time-dependent order**: Ref. [1] assumes a constant γ and time-dependent α, namely $\gamma = \gamma_0$ and $\alpha = \alpha_0 + k_2 t$; we further assume a time-dependent time-fractional order, namely $\gamma = \gamma_0 + k_1 t$;
3. **Space-dependent order**: Ref. [1] assumes a constant γ and a space-dependent α, namely $\gamma = \gamma_0$ and $\alpha = \alpha_0 + k_4 x$ for $x \geq 0$ ($\alpha = \alpha_0$ for $x < 0$); we further assume a time-dependent time-fractional order, namely $\gamma = \gamma_0 + k_3 t$.

The coefficients $\gamma_0, \alpha_0, k_1, k_2, k_3, k_4$ are the parameters to be identified.

The second equation is the 2-D multiscaling constant-order FADE [26,1], written as

$$\frac{\partial^{\gamma_0} C(x,y,t)}{\partial t^{\gamma_0}} = -\mathbf{v} \cdot \nabla C(x,y,t) + D \nabla^{\mathbf{H}^{-1}}_{M(\boldsymbol{\theta})} C(x,y,t), (x,y) \in \mathcal{R}^2. \tag{19}$$

This equation can be used to described the contaminant dispersion in a discrete fractured network, as shown in Sec. 3.3 of ref. [1]. $C(x,y,t)$ is the solute concentration at space-time coordinate (x,y,t), $\mathbf{v}$ is flow velocity, and D is constant dispersion coefficient. The time-fractional order γ_0 and the space-fractional orders α_1, α_2 are assumed to be constants. The operator $\nabla^{\mathbf{H}^{-1}}_M$ is the matrix-order fractional derivative [26] whose special case is the fractional Laplacian when the inverse of the scaling matrix $\mathbf{H}^{-1}$ equals a constant space-fractional order times an identity matrix and when the mixing Lévy measure $M(\boldsymbol{\theta})$ is uniform. The eigenvalues and the eigenvectors of the scaling matrix $\mathbf{H}$ correspond to the reciprocal of the space-fractional orders and the directions of the fractures, respectively. Denote by α_1, α_2 the space-fractional orders in two different fracture directions. Then the scaling matrix is dependent on these orders, namely, $\mathbf{H} := \mathbf{H}(\alpha_1, \alpha_2)$. Also, the mixing measure $M(\boldsymbol{\theta})$ determines the shape and the skewness of the particle plume.

The random walk (or particle tracking) method [27,28] is one possible method that can approximate the fundamental solution of general-form Eq. (19). In the paper, we use this method to solve Eq. (19) and calculate the spatial frequency of particles on a rectangular region of size $[-2, 7] \times [-3, 3]$. The implementation of the method is briefly introduced in Appendix D. The instantaneous point source is located at $x = 0, y = 0$.

Based on the above two FADEs, we give below the description of the parameter identification problem. We regard the above FADEs as PDE systems. The system input is the parameters in fractional orders to be identified, denoted by the input vector $\mathbf{x} = \{x_j\}, j = 1, 2, \ldots, d$, where $\mathbf{x}$ is a vector of dimension d and x_j is its j-th component. For example, for the preceding constant order case in 1-D FADE, we have $x_1 = \gamma_0, x_2 = \alpha_0, d = 2$. The system output is the error between the numerical solution and the field or the synthetic data, which is given by

$$y(\mathbf{x}) = \frac{\| \mathbf{G}(\mathbf{x}) - \mathbf{C_0} \|}{\| \mathbf{C_0} \|} \tag{20}$$

where $y(\mathbf{x})$ is the scalar output corresponding to parameter $\mathbf{x}$, $\mathbf{G}(\mathbf{x})$ is the numerical solution of the FADE evaluated on scattered space-time coordinates, and $\mathbf{C}_0$ is the field or the synthetic data on these coordinates. The norm $\| \cdot \|$ is the l_2-norm. The parameter identification problem can be stated as an optimization problem: find an optimal parameter $\mathbf{x}$ that minimizes $y(\mathbf{x})$, namely

$$\mathbf{x} = \mathit{argmin}\ y(\mathbf{x}), x \in \Omega_{\mathbf{x}} \subset \mathcal{R}^d \tag{21}$$

where $\Omega_{\mathbf{x}}$ is a bounded parameter space within d-dimensional Euclidean space.

4. Results

The section demonstrates the effectiveness of the proposed inversion method through two benchmark problems, which are described in Sec. 3. In the first example, we shall compare the estimated concentration profile with the field data at MADE site. To illustrate the convergence of the proposed method, we also consider the cases with synthetic data whose results are attached in Appendix A. In the second example, we shall compare the estimated concentration profile with the synthetic data.

Assuming that the parameters are uniformly distributed in the parameter space, the initial training inputs are sampled as a quasi-random point set, namely the Sobol sequence [29]. Other sampling strategies, such as Latin hypercube sampling, are also suitable. Generally, we hope that the initial training inputs can cover the whole parameter space as much as possible.

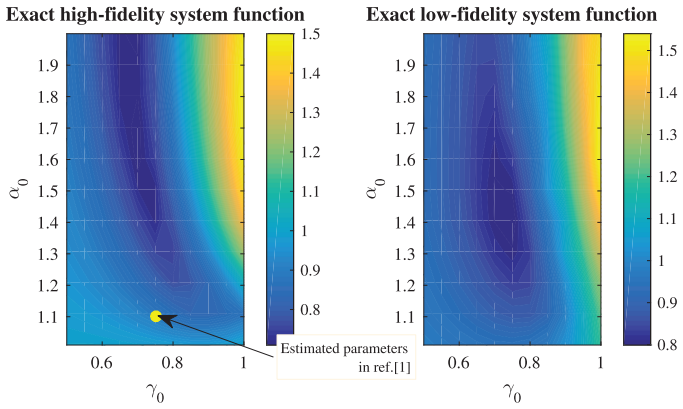


Fig. 3. 1-D constant-order FADE using field data: Comparison of exact system functions for low-fidelity and high-fidelity levels. (For interpretation of the references to color in this figure, the reader is referred to the web version of this article.)

The setup of the multi-fidelity levels could follow certain rules of thumb, if one does not adopt special techniques such as the nonlinear auto-regressive GPR in ref. [30] to safeguard against the lower fidelity levels having poor surrogate accuracy. Two rules are given below. (1) The total time for computing the training outputs in the low fidelity level should not considerably exceed that in the high fidelity level. For example, if 15 seconds is needed for computing one training output in the low level and 6 minutes in the high level, supposing that we have 10 training data in the high level, then the number of training data in the low level should be roughly $360 * 10/15 = 240$. This rule suggests the number of training data in each level after the fidelity levels have been prescribed. (2) The scale factor ρ between the lowest and the highest fidelity levels should fall into $[0.5, 1.5]$. The scale factor represents the cross-correlation between two fidelity levels. A stronger cross-correlation is achieved for a scale factor closer to 1, implying higher surrogate accuracy for the lowest fidelity level. The rule actually requires the surrogate accuracy of the lowest fidelity level not to be extremely low.

4.1. One-dimensional variable-order time-space FADE

4.1.1. Constant fractional order

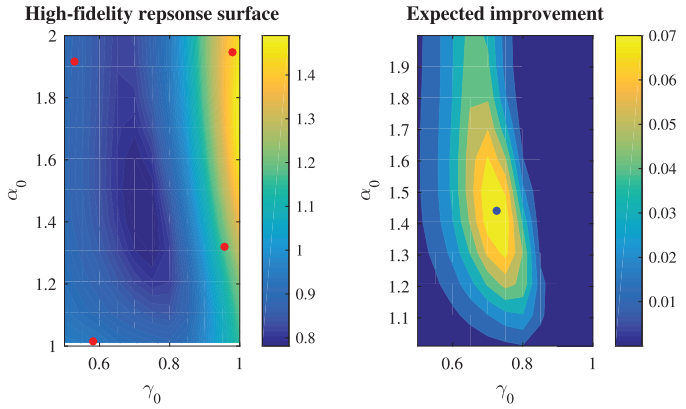
The parameter space is taken as $(\gamma_0, \alpha_0) \in [0.5, 1] \times [1.01, 2]$. The mean-flow velocity v and the dispersion coefficients D in Eq. (18) take the same values as ref. [1], namely $v = 0.14, D = 0.14$. These two parameters are also estimated by trial and error [31]. Their identification using the proposed method is beyond the scope of the paper since the fractional order identification is of our only concern. Also, the identification of the source location and release history [32] is not our goal here but both of these can be also learned given more data points using a similar method.

From the field data, ref. [1] estimates the fractional orders by trial and error which are $\gamma_0 = 0.75, \alpha_0 = 1.1$. The estimation includes some expert knowledge: α_0 is expected to be small, since a smaller α_0 implies a faster dispersion and ensures a better data-fitting at the heavy leading contaminant plume edge. Differently, the paper bases the parameter identification on data themselves, and actually ignores the details of the PDE system and therefore the expert knowledge that the PDE implies. This is one feature of machine learning algorithms, which regards the system as a black-box. The proposed method is completely data-driven and more flexible for the system that is too complex to access the expert knowledge. We put the estimated parameters by trial and error from ref. [1] and the identified parameters by the proposed method in the same error estimation (20) and find that our results produce a smaller error.

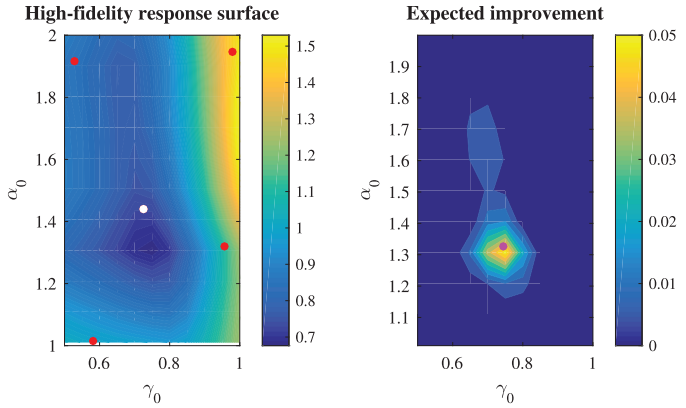
We solve Eq. (18) within the time interval $[0, 224]$ and evaluate the numerical solutions at the space-time coordinates that are the same as those of field data.

To realize the multi-fidelity formulation, we consider two-fidelity levels: (1) Low fidelity — the temporal and the spatial steps are set to be $\Delta t = 0.5$ and $\Delta x = 5$ in the FDM solver; (2) High fidelity — the temporal and the spatial steps are set to be $\Delta t = 0.5$ and $\Delta x = 1$ in the FDM solver. It takes 15 seconds and 6 minutes (on the Intel i7 laptop) for running once the FDM solver of the two fidelity levels. We sample 100 and 4 initial training data for the low and the high levels, respectively, which indicates that we have to run the low-fidelity and the high-fidelity FDM solvers for 100 and 4 times in order to obtain the training outputs. If we simply use the training data, i.e. 4 data, in the high-fidelity level, since the data is a few due to the limited computational resources, the surrogate accuracy of the GPR could be poor. Through adding much more training data, i.e. 100 data, from the low-fidelity level, we might infer an approximate variation trend of the exact high-fidelity system function and obtain a higher surrogate accuracy for the high-fidelity level.

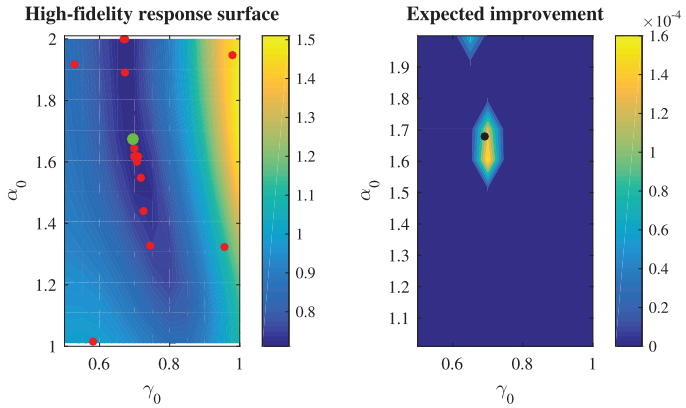
Fig. 3 compares the exact high-fidelity and low-fidelity system functions evaluated on a 11×11 regular grid. We see that the low-fidelity system function values deviate somewhat from the high-fidelity function values. Also, the parameter estimated by trial and error in ref. [1] is denoted by the yellow disk in Fig. 3(left), and we see that the yellow disk is definitely far from the dark blue region where the minimum resides. Using our method, one can gradually enter the region as shown by Fig. 4. Starting from the 4 high-fidelity initial training data and 100 low-fidelity initial training data, we add



(a) Before Bayesian optimization iteration



(b) After one Bayesian optimization iteration



(c) After 20 Bayesian optimization iterations

Fig. 4. 1-D constant-order FADE using field data: Response surfaces of two-fidelity GPR (left) and expected improvement acquisition function (right) for different number of iterations.

consecutively training data points into the current training data set until we find that the added training inputs have been clustered. The maximum location of the expected improvement acquisition function in the current iteration determines the extra training input in the next iteration. For instance, the white disks in Fig. 4(a, right) and Fig. 4(b, left) have the same location. It can also be seen that the expected improvement basically decreases as the iteration proceeds. This trend can be used as the terminal condition for Bayesian optimization iteration. We terminate the iteration when the maximum expected improvement is smaller than a tolerance.

In Appendix A, we also consider the problem with synthetic data. In the synthetic data case, the system function value, namely the error between numerical solution and the data, can reach zero, since the data also comes from the same 1-D FADE. In contrast, the system output in the field data case is dominated by the mathematical modeling error ("model

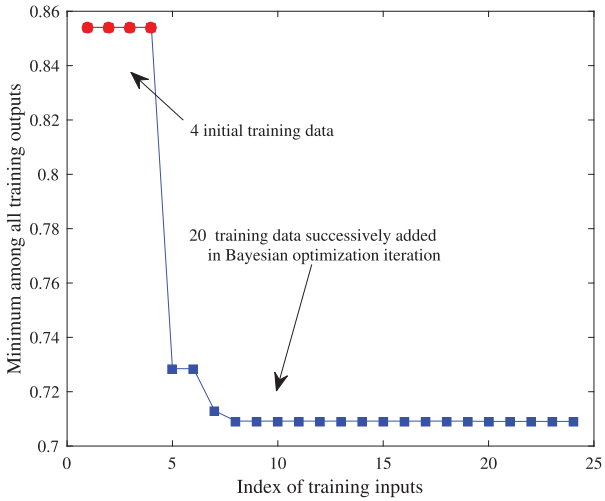


Fig. 5. 1-D constant-order FADE using field data: Convergence of the Bayesian optimization.

Table 3
1-D constant-order FADE using field data: Comparison of identified parameters by our method and that in ref. [1].

Approaches	Identified parameter **p**	Output for **x**, i.e. $y(\mathbf{x})$
Our method	$\gamma_0 = 0.698$, $\alpha_0 = 1.68$	0.709
Trial and error in ref. [1]	$\gamma_0 = 0.75$, $\alpha_0 = 1.1$	0.897

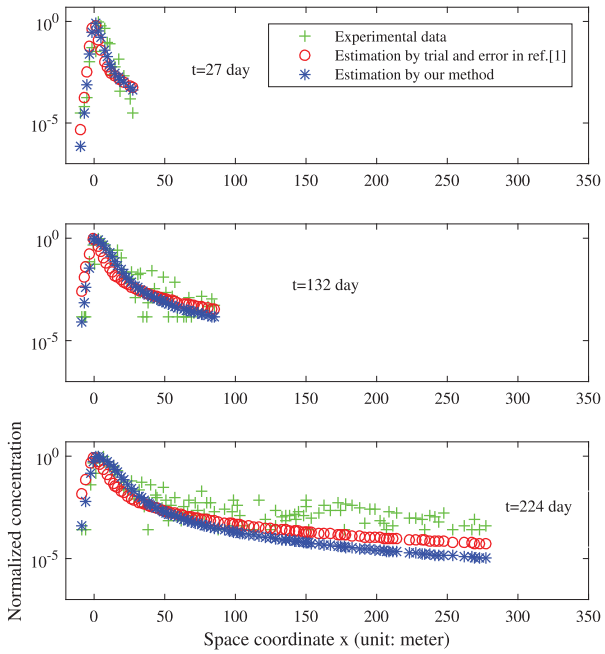


Fig. 6. 1D constant-order FADE using field data: Estimated concentrations by our method and by trial and error in comparison with the experimental data for $t = 27$ (top), 132 (middle), and 224 (bottom).

uncertainty"), and thus the output cannot be arbitrarily close to zero. We can see from Fig. 3(left) that the minimum of the exact system function for the field data case is above 60%, whereas the minimum error in the synthetic data case is close to zero as shown in Fig. A1. Comparison of Fig. 5 and Fig. A2(a) also shows that the minimum of the system function in the field data case is much larger than that in the synthetic data case.

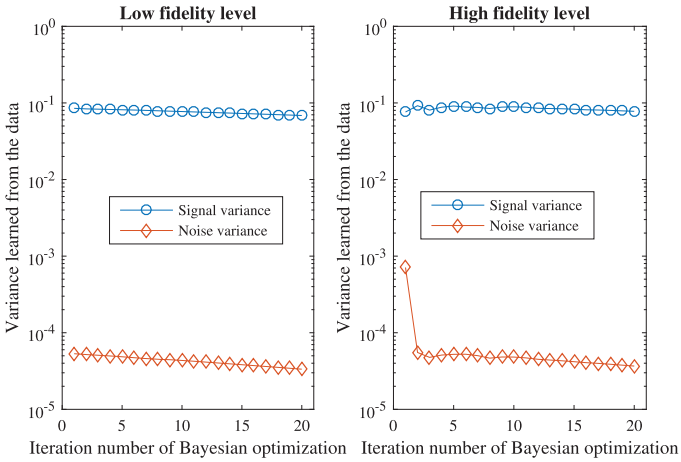


Fig. 7. 1D constant-order FADE: Comparison of the signal and the noise variances for two fidelity levels.

Table 3 compares the identified parameters from the field data with those given in ref. [1]; our results have smaller average relative error. Fig. 6 further shows the estimated concentration profile by our method and the trial and error, in comparison with the field data. At first glance, the two estimations can both fit the field data to a certain extent, but in the sense of the error (20), our method has smaller error and thus performs better.

It should be also noted that the computational cost has been considerably reduced in comparison with the trial and error approach. In the aforementioned example, we solved the forward problem with high-fidelity level $4+20$ times and the problem with low-fidelity level $100+20$ times in order to obtain the training data. For small training data set ($N < 500$) and low-dimensional system (small d), the training (maximizing likelihood function) and the global optimization (maximizing acquisition function) costs are small. It takes $24 * 6 + 120 * 0.25 = 174$ minutes (on the Intel i7 laptop) to compute the training outputs, and it also takes extra $3 * 20 = 60$ minutes to train the multi-fidelity GP model and optimize the acquisition function. Totally, the CPU time is around 4 hours. However, for the trial and error approach, even though considering a coarse regular grid in the parameter space with a discretization step 0.05, one has to solve the forward problem with the high-fidelity level $11 * 21 = 231$ times, which takes $231 * 6 = 1386$ minutes (roughly 23 hours).

The influence of noise on the results can be neglected because the noise variance is much smaller than the signal variance. We take the noise influence of the constant-order case for an example. For the remaining cases, the situation is the same. Fig. 7 shows the signal and the noise variances learned from the data in the two fidelity levels. It is observed that the ratio of the noise variance to the signal variance does not exceed 0.001. From the preceding introduction to Bayesian optimization (see Sec. 2.2), we see that the expected improvement acquisition function (12) cannot take the data noise into account. To show the influence of noise, we replace the expected improvement (EI) function with the expected quantile improvement (EQI) function [24] that considers the data noise, and compare the Bayesian optimization results produced by these two acquisition functions. Fig. 8 gives the comparison of the two acquisition functions. The iteration results are almost the same, which implies that taking the noise into account leads to no improvement. On the contrary, considering noise could lead to worse results. In fact, the minimum of the system output from EQI function, 0.7097, is slightly larger than that from the EI function, 0.7091. In the following examples, we still adopt the EI acquisition function for a slightly faster computation.

4.1.2. Time-dependent fractional order

Compared to constant fractional order, the time-dependent fractional order case can capture better the concentration distribution around the plume peak [1]. In this case, the parameter space is taken as

$$(\gamma_0, k_1, \alpha_0, k_2) \in [0.5, 1] \times [-1/660, 1/660] \times [1.01, 2] \times [-1/330, 1/330],$$
$$\gamma_0 + k_1 t \in [0.5, 1], \alpha_0 + k_2 t \in [1.01, 2], t \in [0, 330]$$

If $\gamma_0 + k_1 t$ or $\alpha_0 + k_2 t$ goes outside the prescribed interval, it will take the value of the nearest endpoint. The mean-flow velocity v and the dispersion coefficients D in Eq. (18) still take $v = 0.14$, $D = 0.14$.

We consider three-fidelity levels: (1) Low fidelity — the temporal and the spatial steps are set to be $\Delta t = 0.5$ and $\Delta x = 5$ in the FDM solver; (2) Intermediate fidelity — the temporal and the spatial steps are set to be $\Delta t = 1$ and $\delta x = 1.54$ in the FDM solver; (3) High fidelity — the temporal and the spatial steps are set to be $\Delta t = 0.5$ and $\Delta x = 1$ in the FDM solver. It takes 15 seconds, 1 min, and 6 minutes for running once the FDM solver of the three levels. We sample 300, 100 and 20 training data for the low, the intermediate, and the high levels. Fig. 9 compares the training outputs of the successive fidelity levels. We see that the intermediate and the high fidelity levels agree with each other better than the low and the intermediate levels do. This is the reason why we have added the intermediate level.

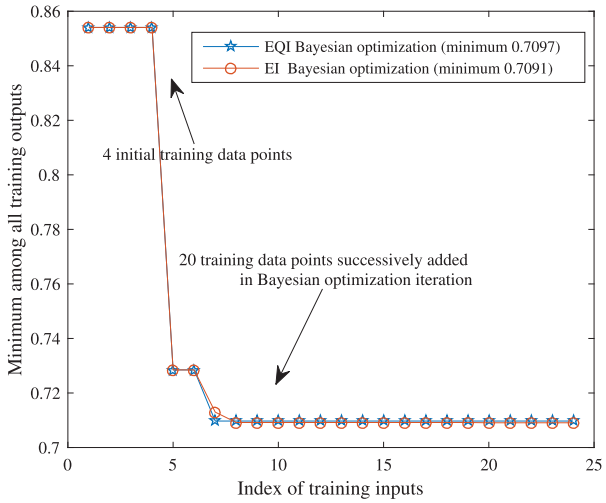


Fig. 8. 1D constant-order FADE: Comparison of Bayesian optimization results obtained by using the expected improvement (EI) and the expected quantile improvement (EQI) acquisition functions.

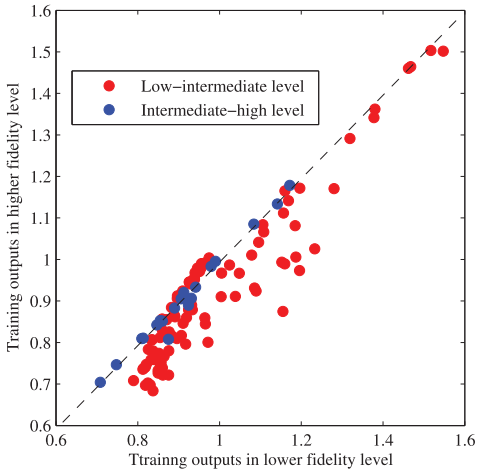


Fig. 9. 1D time-dependent-order FADE: Comparison of training outputs for two fidelity levels.

Table 4
1-D time-dependent-order FADE using field data: Comparison of identified parameters by our method and that in ref. [1].

Approaches	Identified parameter $\mathbf{x}$	Output for $\mathbf{x}$, i.e. $y(\mathbf{x})$
Our method	$\gamma_0 = 0.814$, $k_1 = -9.03\text{e}{-4}$, $\alpha_0 = 1.99$, $k_2 = -2.77\text{e}{-3}$	0.679
Trial and error in ref. [1]	$\gamma_0 = 0.75$, $k_1 = 0$, $\alpha_0 = 2$, $k_2 = -3\text{e}{-3}$	0.728

Fig. 10 shows the Bayesian optimization iteration results based on a total of 20 iterations. Similar to the previous case, due to the mathematical modeling error, the system function minimum is far from zero. In contrast, in Appendix B we consider the synthetic data case as shown in Fig. A2(b). We find that the minimum can approach zero when the mathematical modeling error is removed.

Table 4 compares the identified parameters with those given in ref. [1]. Similar to the previous case, our results still have smaller average relative error. Fig. 11 shows the estimated concentration profile by our method and the trial and error compared to the field data. We see that the estimations by the two methods are very close for $t = 27$ and $t = 132$, but, for $t = 224$, our estimation fits the concentration around the peak better, because the l_2-norm error in Eq. (20) for our method is smaller.

4.1.3. Space-dependent fractional order

Unlike the time-dependent fractional order case, the space-dependent fractional order case can capture better the heavy leading plume edge [1]. In this case, the parameter space is taken as

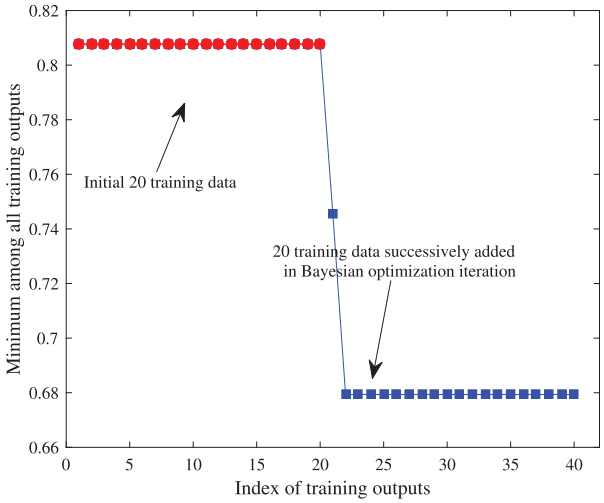


Fig. 10. 1-D time-dependent-order FADE using field data: Variation of minimum of training outputs against the Bayesian optimization iterations.

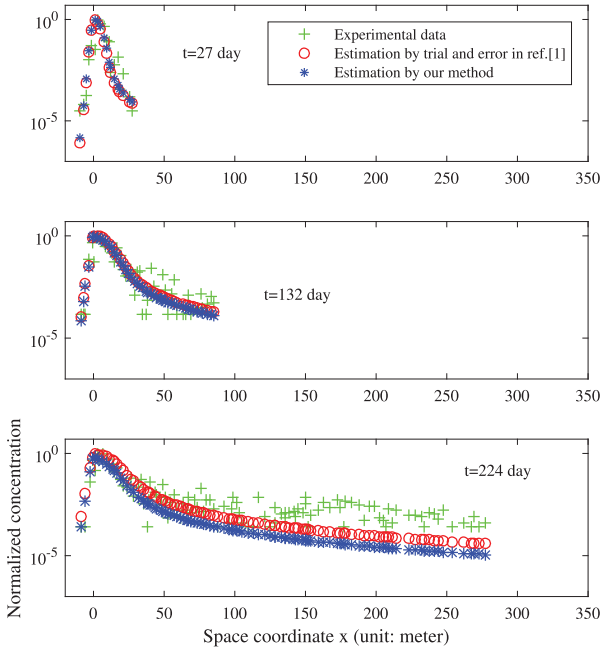


Fig. 11. 1-D time-dependent-order FADE using field data: Estimated concentrations by our method and trial and error in comparison with the experimental data for $t = 27$ (top), 132 (middle), and 224 (bottom).

$$(\gamma_0, k_3, \alpha_0, k_4) \in [0.5, 1] \times [-1/660, 1/660] \times [1.01, 2] \times [-1/350, 0],$$
$$\gamma_0 + k_3 t \in [0.5, 1], t \in [0, 330], \alpha_0 + k_4 x \in [1.01, 2], x \in [0, 350]$$

If $\gamma_0 + k_3 t$ or $\alpha_0 + k_4 t$ goes outside the prescribed interval endpoints, it will take the value of the nearest endpoint. The mean-flow velocity v and the dispersion coefficients D in Eq. (18) still take the values $v = 0.14$, $D = 0.14$.

The multi-fidelity setup and the number of training data are the same as those in the preceding time-dependent case. Here, we use a different definition of system function, namely

$$y_1(\mathbf{x}) = \frac{\| ln(\mathbf{G}(\mathbf{x})) - ln(\mathbf{C_0}) \|_2}{\| ln(\mathbf{C_0}) \|_2} \tag{22}$$

The advantage of the space-dependent fractional order case over the previous two cases is reflected on the more accurate fitting to the data near the edge. However, in the previous system function (l_2 norm without ln transform), the data near the edge is almost omitted since the data spreads over several orders of magnitude. To make the data near the edge really

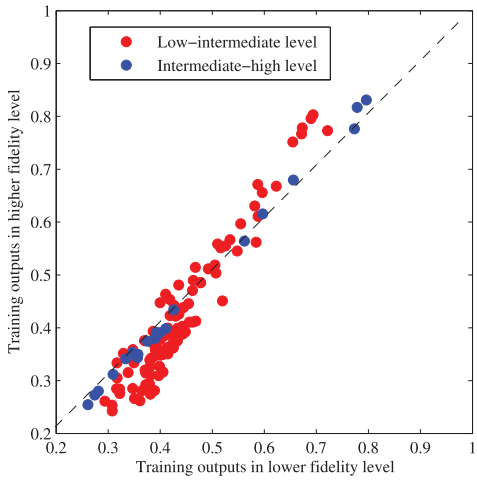


Fig. 12. 1-D space-dependent-order FADE: Comparison of training outputs for two fidelity levels.

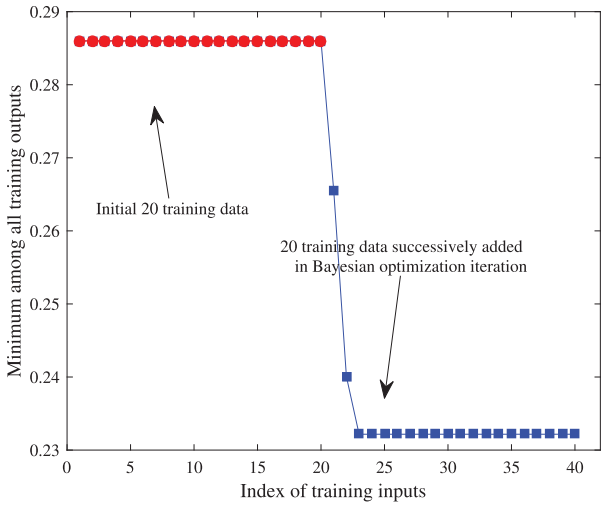


Fig. 13. 1-D space-dependent-order FADE using field data: Variation of minimum number of training outputs against the Bayesian optimization iteration.

matter, we take the *ln* transform on the l_2 norm. Fig. 12 compares the training outputs of three fidelity levels. It is seen that the intermediate and the high fidelity levels agree with each other better than the low and the intermediate levels do.

Fig. 13 shows the Bayesian optimization iteration results based on 20 iterations. In Appendix C, we also consider the synthetic data case in order to see the effectiveness of the proposed method when the mathematical modeling error is removed. Similarly, we see from Fig. A2(c) that Bayesian optimization performs much better for the case with no model uncertainty.

Table 5 compares the identified parameters with those given in ref. [1]. Similar to the previous two cases, our results still have smaller average relative error. The estimated concentration in Fig. 14 shows that our results capture the leading plume edge better than that in ref. [1].

It is worth noting that up to now we have assumed that the variable fractional orders vary linearly with the time and/or space coordinates. Generally, we can assume that the orders vary with the time-space coordinates arbitrarily, by letting

$$\mu(x,t) = \sum_i \lambda_{ij}\varphi_i(x)\varphi_j(t), \tag{23}$$

where $\mu(\cdot,\cdot)$ is the time or space fractional order varying with time-space coordinates, λ_{ij} are the parameters to be identified, and $\varphi_i(\cdot)$ can be taken as the orthogonal polynomials. Alternatively, the variable fractional orders can also be written as the radial basis function expansion or other nonlinear expressions. Whenever the variable order is a nonlinear function we have to identify much more parameters compared to the linear case. The nonlinear case is beyond the scope of the present paper, and we will report the results for this case in the future.

Table 5
1-D space-dependent-order FADE using field data: Comparison of identified parameters by our method and that in ref. [1].

Approaches	Identified parameter $\mathbf{x}$	Output for $\mathbf{x}$, i.e. $y_1(\mathbf{x})$
Our method	$\gamma_0 = 0.730, k_3 = 5.27\text{e}{-4}, \alpha_0 = 1.87, k_4 = -2.86\text{e}{-3}$	0.231
Trial and error in ref. [1]	$\gamma_0 = 0.750, k_3 = 0, \alpha_0 = 2, k_4 = -2.80\text{e}{-3}$	0.286

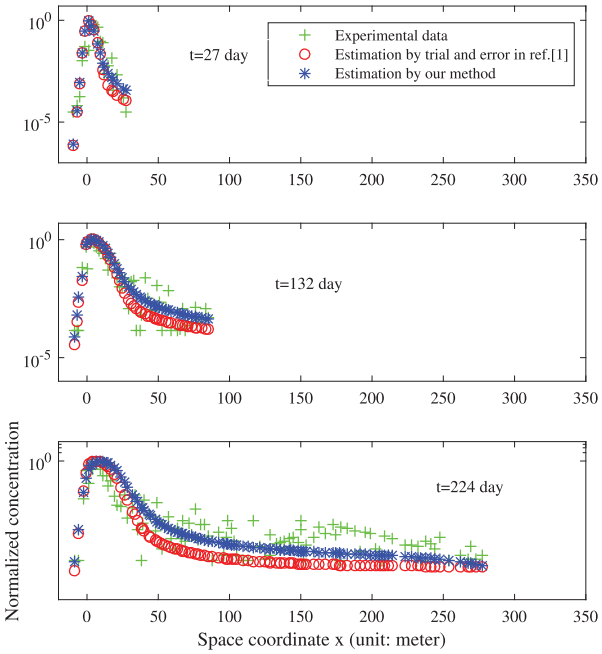


Fig. 14. 1-D space-dependent-order FADE using field data: Estimated concentrations by our method and trial and error in comparison with the experimental data for $t = 27$ (top), 132 (middle), and 224 (bottom).

4.2. Two-dimensional multiscaling constant-order time-space FADE

This example is based on synthetic data as we did not have field data available, and it is used to show the effectiveness of the proposed model inversion method for 2-D FADE. We let the flow velocity be $\mathbf{v} = [1, 0]$ and the dispersion coefficient be $D = 0.01$. The discrete fracture directions are set to be $\theta_1 = \pi/6, \theta_2 = -7\pi/32$. The parameter space is $(\gamma_0, \alpha_1, \alpha_2) \in [0.4, 1] \times [1.01, 2] \times [1.01, 2]$.

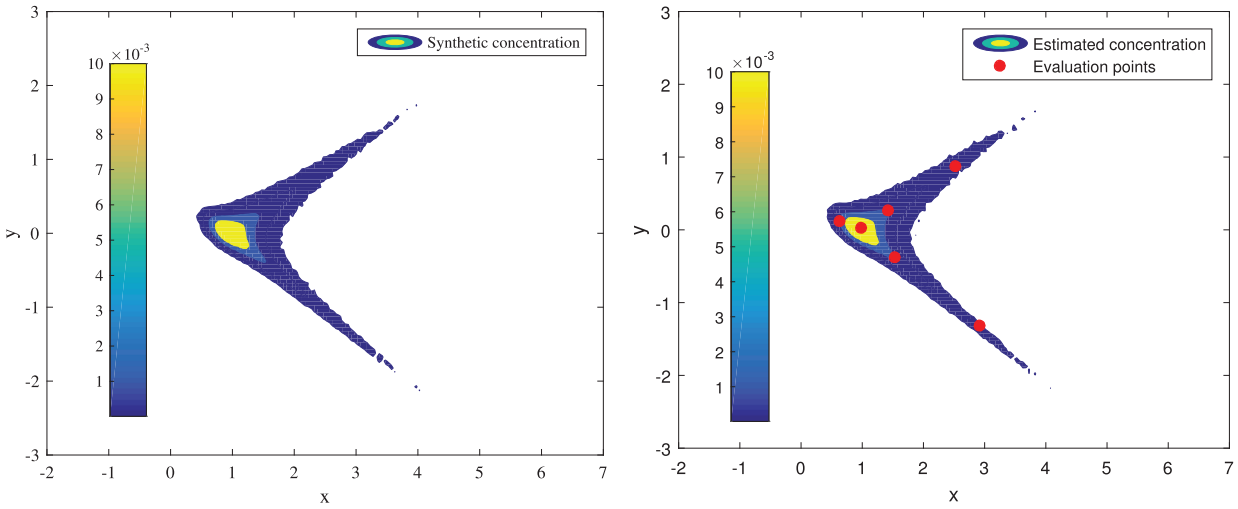
We solve Eq. (19) within the time interval $[0, 1]$ and evaluate the numerical solutions at certain space-time coordinates. In this case, there is no field data and hence we cannot select the space-time coordinates of the field data as evaluation points. Care needs to be taken on selecting these coordinates. We investigated two groups of evaluation points: the sparse and the dense evaluation points. Theoretically, it is better to have dense evaluation points, for this ensures an accurate system output and could guarantee the uniqueness of inverse problem solution. Practically, however, we cannot build many monitor wells within the regional-scale aquifer and thus the evaluation points must be sparse.

We consider two levels of fidelity. (1) Low fidelity: 10^5 particles are released, which requires 6 seconds to run the random walk solver. (2) High fidelity: 10^7 particles are released, which requires 10 minutes to run the random walk solver. We take 100 and 15 training data for the low- and high-fidelity levels, respectively.

4.2.1. Dense evaluation points

Since the l_2-norm error in Eq. (20) inherently ignores the contribution of the small concentration data, the evaluation points should be selected at the locations where the concentration is large. For the synthetic concentration data computed by using exact parameters $(\gamma_0, \alpha_1, \alpha_2) = (0.5, 1.6, 1.8)$, we choose the locations near the concentration peak as the evaluation points. Actually, we select the evaluation points where the normalized concentration is larger than 10^{-4}. Totally 323 evaluation points are selected. Figs. 15(a) and 16 show the synthetic concentration data and the evaluation points, respectively.

The computed concentration using the random walk method is a stochastic field. Each run of the random walk solver produces a slightly different concentration profile. The randomness can be reduced if a sufficiently large number of particles are released. Furthermore, the randomness of the computed concentration leads to the randomness of the Bayesian opti-



(a) Synthetic concentration data (b) Predited concentration data

Fig. 15. 2-D constant-order FADE using synthetic data: Comparison of synthetic and predicted concentration profiles (normalized).

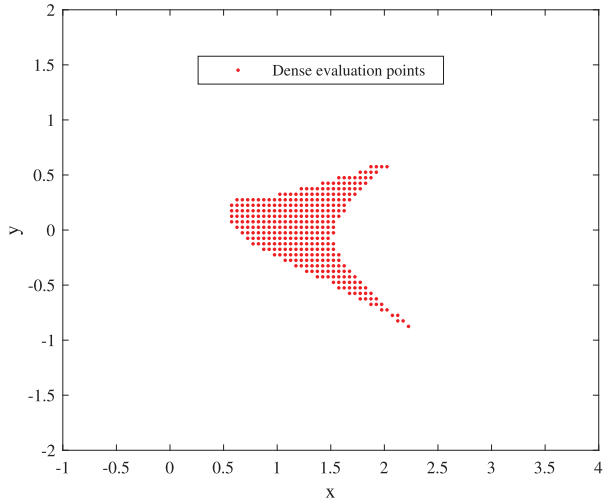


Fig. 16. 2-D constant-order FADE using synthetic data: Dense evaluation points around the peak, where the normalized concentration exceeds the value 1e-4.

mization results and therefore of the identified parameters. Table 6 shows the identified parameters corresponding to five runs of the multi-fidelity GPR and Bayesian optimization code. We see that the identified parameters are perturbed around the exact ones by a minor random quantity. Fig. 17 further shows the randomness of Bayesian optimization results.

To partially reduce the influence of the randomness of the forward problem solution, we solve the forward problem corresponding to exact parameters for ten times, and regard the expectation of these ten groups of solutions as the synthetic data. Furthermore, we select from the five groups of parameters in Table 6 the parameters producing the smallest system output $e(\mathbf{p})$ as the final identified parameters.

4.2.2. Sparse evaluation points

We let the evaluation points be sparse and select 6 scattered evaluation points distributed as Fig. 15(b) shows. We place these points at the locations that characterize the shape of the concentration profile. The estimated concentration is also shown in Fig. 15(b) and the estimated parameters are $(\gamma_0, \alpha_1, \alpha_2) = (0.4946, 1.598, 1.799)$, which we actually select from five groups of random identified parameters.

We have seen that for 2-D FADE (19), the dense and sparse evaluation points are both acceptable. The validity of using the sparse evaluation points enables one to identify the parameters from the field data when the monitor wells in the aquifer are few. The strategy to select the sparse evaluation points is that the concentration on some of these points should be large and also some other points should get close to the boundary of the concentration profile.

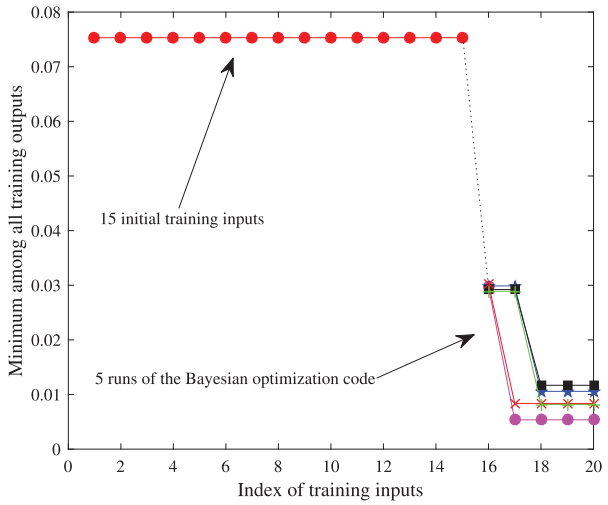


Fig. 17. 2-D constant FADE using synthetic data: Iteration results for 5 runs of the Bayesian optimization solver. (The five different colors on the right part of the figure correspond to 5 runs of the code.)

Table 6
2-D constant-order FADE using synthetic data: Parameters from 5 runs of Bayesian optimization solver. The final identified parameters are the Run 1 results since the corresponding output is minimum among the five groups of results.

	γ_0	α_1	α_2	$\min(y(\mathbf{x}))$
Exact parameters	$\gamma_0 = 0.500$	$\alpha_1 = 1.600$	$\alpha_2 = 1.800$	0.00000
Run 1	$\gamma_0 = 0.484$	$\alpha_1 = 1.600$	$\alpha_2 = 1.801$	0.00536
Run 2	$\gamma_0 = 0.528$	$\alpha_1 = 1.596$	$\alpha_2 = 1.794$	0.00972
Run 3	$\gamma_0 = 0.531$	$\alpha_1 = 1.590$	$\alpha_2 = 1.795$	0.01169
Run 4	$\gamma_0 = 0.490$	$\alpha_1 = 1.597$	$\alpha_2 = 1.806$	0.00836
Run 5	$\gamma_0 = 0.482$	$\alpha_1 = 1.596$	$\alpha_2 = 1.802$	0.00818

5. Discussion

We have extended the multi-fidelity Bayesian optimization method, one of the probabilistic machine learning algorithms [33], to identification of the fractional orders in advection–dispersion equations of complex form. Besides the fractional inverse problem, the method could also be applicable for inverse problems in a variety of systems having a certain number of accessible input-output pairs.

There are very few studies on the theoretical analysis for solution existence, solution uniqueness, and well-posedness of the inverse problem we considered. Although ref. [34] theoretically analyzes the well-posedness of fractional elliptic inverse problems, yet the fractional diffusion operator therein differs completely from the one considered in our paper. Well-posedness of the inverse problem implies that a small change in the training outputs leads to a small change in the identified parameters, namely, the mapping from system output to input is continuous or the identified parameters are robust to training data noise. By imposing additional constraints on solutions, regularization is a common approach to guarantee the well-posedness of the inverse problem. For GPR, the reproducing kernel Hilbert space regularization term (see Eq. (5.48) of ref. [35]) inherently avoids the over-fitting and guarantees a smooth solution, which implies a resilience to the data noise. With regards to solution existence and uniqueness, numerically speaking, our numerical examples for synthetic data show the uniqueness solutions of the inverse problem for the parameter intervals that we considered. However, it may be that not all the parameter identification problems of FADE produce unique solutions. For example, if we include more parameters besides fractional orders, say, the flow velocity, dispersion coefficients, source location, and the parameters in source history function, we do not quite know whether or not the inverse problem in the case has a unique solution. We will certainly need much more data in this case! We plan to apply the proposed model inversion method to these more complicated inverse problems in the future, expecting to get some insights into the nature of the solutions we obtain.

Another future work includes learning the fractional orders directly by solving only the forward problem by resorting to multi-fidelity GPR. This can be done by absorbing the fractional orders into the hyper-parameters $\boldsymbol{\theta}$ of the multi-fidelity GP models, and therefore one can obtain the identified fractional orders by simply training the GP models [36]. The challenge is that the 2-D space-fractional derivative in Eq. (19) is too complicated to derive the explicit expression of the space-fractional derivative of the covariance function. We will work along this line to resolve the problem.

Lastly, we compare our model inversion method to the well-known Bayesian approach for inverse problems [37,34]. The basic properties of the two methods are shown in Table 7. Generally speaking, if one expects to infer fast point-estimates

Table 7
Comparison of the model inversion method and the Bayesian approach for inverse problems.

Items	Our model inversion method	Bayesian approach for inverse problems
System input	$\mathbf{x} \in \mathcal{R}^d$	$\mathbf{x} \in \mathcal{R}^d$
System output	$f(\mathbf{x}) \in \mathcal{R}^1$	$\mathbf{G}(\mathbf{x}) \in \mathcal{R}^n$
Observation/data	$y(\mathbf{x}) = \frac{\|\mathbf{G}(\mathbf{x})-\mathbf{C}_0\|}{\|\mathbf{C}_0\|} = f(\mathbf{x}) + \epsilon$	$\mathbf{C}_0 = \mathbf{G}(\mathbf{x}) + \boldsymbol{\eta}$
Prior	$\mathbf{f}_N \sim \mathcal{N}(\mathbf{m}, \mathbf{K})(\in \mathcal{R}^N)$	$\mathbf{x} \sim \mathcal{P}(\mathbf{m}_0, \boldsymbol{\Sigma}_0)$
Likelihood	$p(\mathbf{y}_N\|\mathbf{f}_N) = p(\boldsymbol{\epsilon}_N), \mathbf{y}_N, \boldsymbol{\epsilon}_N \in \mathcal{R}^N$	$p(\mathbf{C}_0\|\mathbf{G}(\mathbf{x})) = p(\boldsymbol{\eta}), \mathbf{z}, \boldsymbol{\eta} \in \mathcal{R}^n$
Posterior	$p(\mathbf{f}_N\|\mathbf{y}_N)$	$p(\mathbf{x}\|\mathbf{C}_0)$
Optimal input	$\mathbf{x}_{opt} = \arg\min_{\mathbf{x}}\{f(\mathbf{x})\}$	$\mathbf{x}_{opt} = \arg\max_{\mathbf{x}}\{p(\mathbf{x}\|\mathbf{C}_0)\}$
Surrogate for $f(\mathbf{x})$	Yes	No
Surrogate for $\mathbf{G}(\mathbf{x})$	No	Yes [9]

of the fractional orders but does not care about uncertainty in the identified fractional orders, then one can consider the Bayesian optimization; otherwise, one can adopt a Bayesian approach which gives the posterior distribution of the identified fractional orders, namely $p(\mathbf{x}|\mathbf{C}_0)$. Although these two methods both take advantage of Bayes' rule, yet they prescribe the prior distributions on different quantities of the system. GPR gives the Gaussian prior on the system output, whereas the Bayesian inversion method assigns the certain prior $\mathcal{P}$ to the system input. Despite different priors, the outputs of the two methods can be related. The output of our method is actually the norm of the noise term of the Bayesian approach output up to a constant multiplier. For the nonlinear implicit expensive mapping $\mathbf{G}(\cdot)$ from the equation parameters $\mathbf{x}$ to the equation solutions $\mathbf{G}(\mathbf{x})$, it is sensible to construct its surrogate model. Our method simulates the composite mapping $f := O \circ G$ using a multi-fidelity GPR model; similarly in the Bayesian approach, one can surrogate $\mathbf{G}$ directly. For instance, ref. [9] surrogates the input-output mapping for time-fractional diffusion equation using the stochastic collocation method. By comparison, our method further takes advantage of the uncertainty information of the surrogate surface and adaptively searches the optimal input aiming at keeping the number of expensive function evaluations at a minimum. On the other hand, the prior information in the Bayesian approach, e.g. the form of the prior distribution $\mathcal{P}$ and the prior parameters $\mathbf{m}_0, \boldsymbol{\Sigma}_0$, should be prescribed beforehand, bearing in mind that unreasonable parameters could lead to erroneous solutions. In contrast, for our method, the prior form is fixed, i.e., Gaussian, and the prior distribution for signal variances and trend coefficients is taken as the non-informative Jeffreys distribution having no extra parameters to be tuned [17]. Moreover, the hyper-parameters in the output prior can be automatically learned by maximizing the marginal likelihood.

6. Summary

In this paper, we extend the model inversion method comprising multi-fidelity GPR and Bayesian optimization to identification of the fractional orders in 1-D variable-order FADE and 2-D multi-scaling constant-order FADE. Using our systematic method, the identified parameters are more accurate than those estimated by trial and error in ref. [1], and most importantly, the computational cost is considerably reduced. It should be noted that our method and the trial and error method are the only two methods up to now having been employed to solve such advection dispersion problems.

Acknowledgements

This work is supported by the Beijing Computational Science Research Center (CSRC), Beijing, China, and also by the MURI/ARO at Brown University on "Fractional PDEs for Conservation Laws and Beyond: Theory, Numerics and Applications" (W911NF-15-1-0562). The work of W. Cai is supported by US Army Research Office (W911NF-14-1-0297) and US NSF (DMS-1719303). The work of P. Perdikaris is supported by DARPA EQUiPS (N66001-15-2-4055).

Appendix A. One-dimensional constant-order FADE using synthetic data

This Appendix provides the numerical results of identifying the constant orders in 1-D FADE using synthetic data, in comparison with the field data case in Sec. 4.1.1, we simply replace the field data with the synthetic data while keeping other setups of computation unchanged. We see from Fig. A2(a) that the minimum of the training outputs gradually decreases to zero, which shows the convergence of the multi-fidelity Bayesian optimization. Table A1 shows that the estimated parameters are very close to the exact ones and the corresponding output also approaches zero. Actually, the terminal condition for Bayesian optimization in this case is that the output $y(\mathbf{x})$ is smaller than 1%.

Table A1
1-D constant-order FADE using synthetic data: Identified parameters by the proposed method.

Approaches	Identified parameter $\mathbf{x}$	Output for $\mathbf{x}$, i.e. $y(\mathbf{x})$
Our method	$\gamma_0 = 0.797, \alpha_0 = 1.611$	3.7e−3
Exact parameters	$\gamma_0 = 0.8, \alpha_0 = 1.6$	0

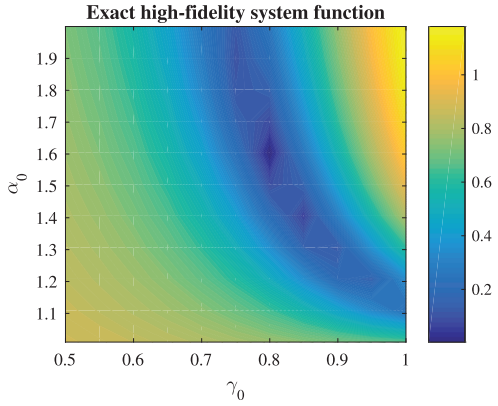
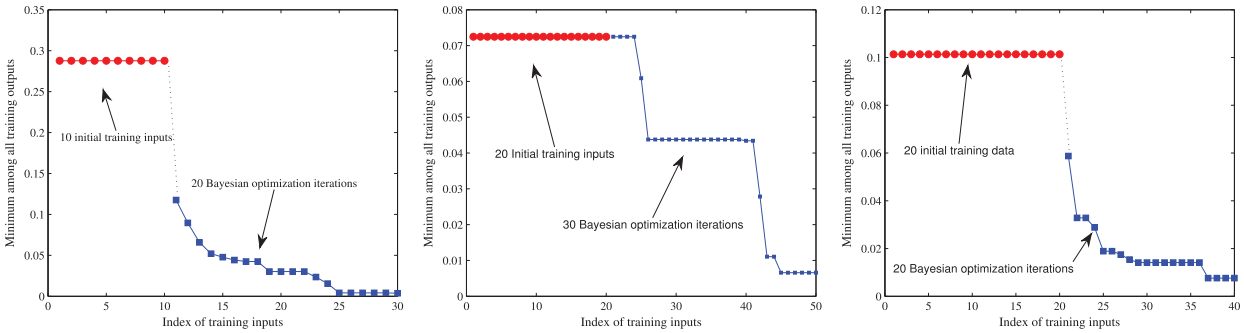


Fig. A1. 1-D constant-order FADE using synthetic data: Exact high-fidelity system function with exact parameters $\gamma_0 = 0.8, \alpha_0 = 1.6$.



(a) Constant order (b) Time-dependent order (c) Space-dependent order

Fig. A2. 1-D FADE using synthetic data: Bayesian optimization results for (a) constant, (b) time-dependent, and (c) space-dependent orders.

Table A2
1D time-dependent-order FADE using synthetic data: Identified parameters by the proposed method.

Approaches	Identified parameter $\mathbf{x}$	Output for $\mathbf{x}$, i.e. $y(\mathbf{x})$
Our method	$\gamma_0 = 0.797, k_1 = -8.00\text{e}{-5}, \alpha_0 = 1.574, k_2 = 3.80\text{e}{-4}$	6.6e−3
Exact parameters	$\gamma_0 = 0.8, k_1 = 0, \alpha_0 = 1.6, k_2 = 0$	0

Table A3
1-D space-dependent-order FADE using synthetic data: Identified parameter by the proposed method.

Approaches	Identified parameter $\mathbf{x}$	Output for $\mathbf{x}$, i.e. $y_1(\mathbf{x})$
Our method	$\gamma_0 = 0.791, k_3 = -6.80\text{e}{-5}, \alpha_0 = 1.586, k_4 = 2.4\text{e}{-8}$	7.6e−3
Exact parameters	$\gamma_0 = 0.8, k_3 = 0, \alpha_0 = 1.6, k_4 = 0$	0

Appendix B. One-dimensional time-dependent-order FADE using synthetic data

This Appendix provides the numerical results of identifying the time-dependent fractional orders in 1-D FADE using the synthetic data, in order to contrast the results with the field data case in Sec. 4.1.2. We simply replace the field data with the synthetic data while keeping the rest of the setup of computation unchanged. We see from Fig. A2(b) that the minimum of the training outputs is reduced considerably after 30 Bayesian iterations, which shows the convergence of the multi-fidelity Bayesian optimization for this case. The straight line within the iteration process indicates that the updating training output is not smaller than the current smallest training output, and thus the current minimum is always preserved until a smaller one appears. Table A2 shows that the estimated parameters are close to the exact ones and the corresponding output also approaches zero. The terminal condition for Bayesian optimization is that the output $y(\mathbf{x})$ is smaller than 1%.

Appendix C. One-dimensional space-dependent-order FADE using synthetic data

This Appendix provides the numerical results of identifying the space-dependent fractional orders in 1-D FADE using the synthetic data; see the field data case in Sec. 4.1.3. We see from Fig. A2(c) that the minimum of the training outputs

decreases gradually to zero, which shows the convergence of the multi-fidelity Bayesian optimization. Table A3 shows the estimated parameters that approach the exact ones. The terminal condition for Bayesian optimization is that the output $y_1(\mathbf{x})$ is smaller than 1%.

Appendix D. Random walk method

This Appendix describes the implementation of the random walk method [27,28] for solving 2-D FADE (19). The particle location at time t, $\mathbf{X}(t)$, is the sum of random jumps, each with a random jump time [26],

$$\mathbf{X}(t) = \sum_{n=1}^{N(t)} R_n^{\mathbf{H}} \cdot \boldsymbol{\theta}_n. \tag{A.1}$$

The jump size matrix $R^{\mathbf{H}}$ related to the space-fractional orders α_1, α_2 is given by

$$R^{\mathbf{H}} = \begin{bmatrix} \mathbf{v}_1 & \mathbf{v}_2 \end{bmatrix} \begin{bmatrix} R_n^{1/\alpha_1} & 0 \\ 0 & R_n^{1/\alpha_2} \end{bmatrix} \begin{bmatrix} \mathbf{v}_1 & \mathbf{v}_2 \end{bmatrix}^{-1}, \tag{A.2}$$

where $\mathbf{v}_i, i = 1, 2$ and $1/\alpha_i$ are the eigenvectors and the eigenvalues of the scaling matrix $\mathbf{H}$, respectively. The jump size R^{1/α_i} is given by the Lévy noise [1,27]

$$R_n^{1/\alpha_i} = \left| cos\frac{\pi\alpha_i}{2} \cdot D \cdot d\tau \right|^{1/\alpha_i} S_{\alpha_i}(\beta = 1, \sigma = 1, \mu = 0). \tag{A.3}$$

$S_{\alpha_i}(\beta, \sigma, \mu)$ is the random variable drawn from the Lévy stable distribution with index α_i, maximum skewness β, unit scale σ, and zero shift μ. D is the constant dispersion coefficient, and $d\tau$ is the operational time increment [28]. (In the paper, we let $d\tau = 0.1$.) The random jump direction $\boldsymbol{\theta}_n$ is drawn from the mixing Lévy measure $M(d\theta)$. In the example we consider, we choose the discrete measure, $p(\boldsymbol{\theta}_n = \boldsymbol{\theta}_1) = 0.4$ and $p(\boldsymbol{\theta}_n = \boldsymbol{\theta}_2) = 0.6$, where the jump directions $\boldsymbol{\theta}_1 = [\cos(\pi/6), \sin(\pi/6)]^T$ and $\boldsymbol{\theta}_2 = [\cos(-7\pi/32), \sin(-7\pi/32)]^T$ coincide with the aquifer fracture directions. Further, if letting the eigenvectors $\mathbf{v}_1, \mathbf{v}_2$ also coincide with the fracture directions, the particle location can be simplified into

$$\mathbf{X}(t) = \sum_{n=1}^{N(t)} R_n^{1/\alpha_i} \boldsymbol{\theta}_i, \quad p(i = 1) = 0.4, \quad p(i = 2) = 0.6. \tag{A.4}$$

For general mixing probability measure $M(d\theta)$ and variable dispersion coefficients and flow velocity, we refer the readers to ref. [27].

To determine the number of jump steps $N(t)$, we resort to another Lévy noise increment dt that is related to the time-fractional order γ [28],

$$dt = \left| \cos\frac{\pi\gamma}{2} \cdot d\tau \right|^{1/\gamma} S_\gamma(\beta = 1, \sigma = 1, \mu = 0). \tag{A.5}$$

Assuming that the observation time is T (we choose $T = 1$ in the example), the number of jump steps is determined by

$$N(t) = \min_m \{t_m = \sum_{n=1}^{m} (dt)_n \geq T\}. \tag{A.6}$$

Generally, $t_{N(t)}$ will exceed T, and we determine the particle location at T using linear interpolation between the locations $\mathbf{X}(t_{N(t)-1})$ and $\mathbf{X}(t_{N(t)})$.

References

[1] HongGuang Sun, Yong Zhang, Wen Chen, Donald M. Reeves, Use of a variable-index fractional-derivative model to capture transient dispersion in heterogeneous media, J. Contam. Hydrol. 157 (2014) 47–58.
[2] David A. Benson, Stephen W. Wheatcraft, Mark M. Meerschaert, Application of a fractional advection–dispersion equation, Water Resour. Res. 36 (6) (2000) 1403–1412.
[3] Mark M. Meerschaert, David A. Benson, Boris Bäumer, Multidimensional advection and fractional dispersion, Phys. Rev. E 59 (5) (1999) 5026.
[4] Jin Cheng, Junichi Nakagawa, Masahiro Yamamoto, Tomohiro Yamazaki, Uniqueness in an inverse problem for a one-dimensional fractional diffusion equation, Inverse Probl. 25 (11) (2009) 115002.
[5] Vu Tuan, Inverse problem for fractional diffusion equation, Fract. Calc. Appl. Anal. 14 (1) (2011) 31–55.
[6] Miller Luc, Masahiro Yamamoto, Coefficient inverse problem for a fractional diffusion equation, Inverse Probl. 29 (7) (2013) 075013.
[7] Ting Wei, Jungang Wang, A modified quasi-boundary value method for an inverse source problem of the time-fractional diffusion equation, Appl. Numer. Math. 78 (2014) 95–111.
[8] Ying Zhang, Xiang Xu, Inverse source problem for a fractional diffusion equation, Inverse Probl. 27 (3) (2011) 035010.
[9] Liang Yan, Ling Guo, Stochastic collocation algorithms using l_1-minimization for Bayesian solution of inverse problems, SIAM J. Sci. Comput. 37 (3) (2015) A1410–A1435.

[10] Hui Wei, Wen Chen, Hongguang Sun, Xicheng Li, A coupled method for inverse source problem of spatial fractional anomalous diffusion equations, Inverse Probl. Sci. Eng. 18 (7) (2010) 945–956, Formerly Inverse Probl. Eng..
[11] Yong Zhang, Mark M. Meerschaert, Roseanna M. Neupauer, Backward fractional advection dispersion model for contaminant source prediction, Water Resour. Res. 52 (4) (2016) 2462–2473.
[12] Bangti Jin, William Rundell, A tutorial on inverse problems for anomalous diffusion processes, Inverse Probl. 31 (3) (2015) 035003.
[13] Paris Perdikaris, George Em Karniadakis, Model inversion via multi-fidelity Bayesian optimization: a new paradigm for parameter estimation in haemodynamics, and beyond, J. R. Soc. Interface 13 (118) (2016) 20151107.
[14] Carl Edward Rasmussen, Christopher K.I. Williams, Gaussian Processes for Machine Learning, The MIT Press, 2006.
[15] Carl Edward Rasmussen, Hannes Nickisch, Gaussian processes for machine learning (GPML) toolbox, J. Mach. Learn. Res. 11 (2010) 3011–3015.
[16] Carl Edward Rasmussen, Evaluation of Gaussian Processes and Other Methods for Non-Linear Regression, PhD thesis, CiteSeer, 1996.
[17] Loic Le Gratiet, Multi-Fidelity Gaussian Process Regression for Computer Experiments, PhD thesis, Université Paris-Diderot-Paris VII, 2013.
[18] James Hensman, Nicolo Fusi, Neil D. Lawrence, Gaussian processes for big data, arXiv preprint, arXiv:1309.6835, 2013.
[19] Paris Perdikaris, Daniele Venturi, George Em Karniadakis, Multifidelity information fusion algorithms for high-dimensional systems and massive data sets, SIAM J. Sci. Comput. 38 (4) (2016) B521–B538.
[20] Marc C. Kennedy, Anthony O'Hagan, Predicting the output from a complex computer code when fast approximations are available, Biometrika 87 (1) (2000) 1–13.
[21] Loic Le Gratiet, Josselin Garnier, Recursive co-kriging model for design of computer experiments with multiple levels of fidelity, Int. J. Uncertain. Quantificat. 4 (5) (2014).
[22] Eric Brochu, Vlad M. Cora, Nando De Freitas, A tutorial on Bayesian optimization of expensive cost functions, with application to active user modeling and hierarchical reinforcement learning, arXiv preprint, arXiv:1012.2599, 2010.
[23] Donald R. Jones, Cary D. Perttunen, Bruce E. Stuckman, Lipschitzian optimization without the Lipschitz constant, Int. J. Optim.: Theory Methods Appl. 79 (1) (1993) 157–181.
[24] Victor Picheny, David Ginsbourger, Yann Richet, Gregory Caplin, Quantile-based optimization of noisy computer experiments with tunable precision, Technometrics 55 (1) (2013) 2–13.
[25] Anton Schwaighofer, Volker Tresp, Kai Yu, Learning Gaussian process kernels via hierarchical Bayes, in: NIPS, 2004, pp. 1209–1216.
[26] Rina Schumer, David A. Benson, Mark M. Meerschaert, Boris Baeumer, Multiscaling fractional advection–dispersion equations and their solutions, Water Resour. Res. 39 (1) (2003).
[27] Yong Zhang, David A. Benson, Mark M. Meerschaert, Eric M. LaBolle, Hans-Peter Scheffler, Random walk approximation of fractional-order multiscaling anomalous diffusion, Phys. Rev. E 74 (2) (2006) 026706.
[28] Yong Zhang, Mark M. Meerschaert, Boris Baeumer, Particle tracking for time-fractional diffusion, Phys. Rev. E 78 (3) (2008) 036705.
[29] Il'ya Meerovich Sobol', On the distribution of points in a cube and the approximate evaluation of integrals, Zh. Vychisl. Mat. Mat. Fiz. 7 (4) (1967) 784–802.
[30] Paris Perdikaris, Maziar Raissi, Andreas Damianou, Neil Lawrence, George Em Karniadakis, Nonlinear information fusion algorithms for data-efficient multi-fidelity modelling, Proc. R. Soc. Lond., Ser. A, Math. Phys. Eng. Sci. 473 (2198) (2017).
[31] Yong Zhang, David A. Benson, Donald M. Reeves, Time and space nonlocalities underlying fractional-derivative models: distinction and literature review of field applications, Adv. Water Resour. 32 (4) (2009) 561–581.
[32] Ilaria Butera, Maria Giovanna Tanda, Andrea Zanini, Simultaneous identification of the pollutant release history and the source location in groundwater by means of a geostatistical approach, Stoch. Environ. Res. Risk Assess. 27 (5) (2013) 1269–1280.
[33] Zoubin Ghahramani, Probabilistic machine learning and artificial intelligence, Nature 521 (7553) (2015) 452–459.
[34] Nicolas Garcia Trillos, Daniel Sanz-Alonso, The bayesian formulation and well-posedness of fractional elliptic inverse problems, arXiv preprint, arXiv:1611.05475, 2016.
[35] Jerome Friedman, Trevor Hastie, Robert Tibshirani, The Elements of Statistical Learning, Springer Ser. Stat., vol. 1, Springer, Berlin, 2001.
[36] Maziar Raissi, George Em Karniadakis, Machine learning of linear differential equations using Gaussian processes, arXiv preprint, arXiv:1701.02440, 2017.
[37] Andrew M. Stuart, Inverse problems: a Bayesian perspective, Acta Numer. 19 (2010) 451–559.