Journal of Computational Physics 416 (2020) 109519

Contents lists available at ScienceDirect

Journal of Computational Physics

www.elsevier.com/locate/jcp

ANOVA Gaussian process modeling for high-dimensional stochastic computational models

Chen Chen [a,b], Qifeng Liao [a,*]

[a] School of Information Science and Technology, ShanghaiTech University, Shanghai, 201210, China
[b] Department of Computer Science, University of Maryland, College Park, MD 20742, United States

ARTICLE INFO

Article history:
Received 14 July 2019
Received in revised form 21 April 2020
Accepted 29 April 2020
Available online 11 May 2020

Keywords:
Adaptive ANOVA
Gaussian process
Model reduction
Uncertainty quantification

ABSTRACT

In this paper we present a novel analysis of variance Gaussian process (ANOVA-GP) emulator for models governed by partial differential equations (PDEs) with high-dimensional random inputs. The Gaussian process (GP) is a widely used surrogate modeling strategy, but it can become invalid when the inputs are high-dimensional. In this new ANOVA-GP strategy, high-dimensional inputs are decomposed into unions of local low-dimensional inputs, and principal component analysis (PCA) is applied to provide dimension reduction for each ANOVA term. We then systematically build local GP models for PCA coefficients based on ANOVA decomposition to provide an emulator for the overall high-dimensional problem. We present a general mathematical framework of ANOVA-GP, validate its accuracy and demonstrate its efficiency with numerical experiments.

1. Introduction

During the last few decades there has been a rapid development in surrogate modeling for computational models governed by stochastic partial differential equations (PDEs). This explosion in interest has been driven by practical applications including uncertainty quantification, shape and topological optimizations, and Bayesian inversions. In these applications, repeated simulations for parameterized PDE systems are demanded. High-fidelity numerical schemes, which are also referred to as the simulators, can give accurate predictions for the outputs of these PDE systems, e.g., the finite element methods with a posteriori error bounds [1,2]. However, the simulators are typically computationally expensive, especially when modeling complex science and engineering problems. In order to reduce the costs in these many-query problems of computational models, cheap surrogate models which are also called emulators, are actively developed to replace the simulators. These include Gaussian process (GP) emulators [3–6], (generalized) polynomial chaos surrogates [7–10] and reduced basis methods [11–14].

The original GP emulator is to model the system output by a Gaussian process indexed by input parameters [4], which limits its application to high-dimensional problems. In general, the computational models governed by stochastic PDEs have high-dimensional inputs and outputs. There are always a large number of input parameters, when modeling complex problems, for example, models with inputs described by rough random processes with short correlation lengths. The standard outputs of the PDE systems are the spatial fields, and when a fine resolution representation is required, the outputs need to be high-dimensional to capture detailed local information. This kind of high-dimensional problems currently gains a lot of

* Corresponding author.
E-mail addresses: cchen24@umd.edu (C. Chen), liaoqf@shanghaitech.edu.cn (Q. Liao).

https://doi.org/10.1016/j.jcp.2020.109519
0021-9991/

interests, and new GP emulators are actively developed. These new GP methods usually focus on either a high-dimensional input space or a high-dimensional output space, and propose dimension reduction techniques for the corresponding high-dimensional space. In [15], principal component analysis (PCA) is applied to the output space to result in an efficient GP emulator for models with high-dimensional outputs. In [16,17], novel kernel principal component analysis is developed to perform dimension reduction for the output space. In addition, an active data selection method is developed to build GP surrogates for PCA coefficients in [18]. For problems with high-dimensional inputs, GP models with built-in active subspace dimension reduction are proposed in [19].

We focus on the challenging situation that both inputs and outputs are high-dimensional. A main challenge here is that difficulties caused by high-dimensional inputs and outputs are typically coupled. As discussed in [12], high-dimensional inputs can lead to large ranks in the output space, and direct PCA for the output space can consequently become inefficient. To decouple the difficulties, we propose a novel analysis of variance (ANOVA) based Gaussian process method (ANOVA-GP). In this ANOVA-GP emulator, the high-dimensional parameter space is decomposed into a union of low-dimensional spaces through an adaptive ANOVA procedure. PCA is conducted locally on ANOVA terms associated with these low-dimensional parameter spaces. After that, local GP models are built for PCA coefficients. Since the local inputs are low-dimensional, efficient PCA can be achieved and a small number of training data points are required to result in accurate local GP models. In addition, we note that a Bayesian smoothing spline ANOVA Gaussian process framework is developed for model calibration with categorical parameters [20], but the novelty of our ANOVA-GP lies on adaptive construction procedures for hierarchical GP models for high-dimensional (noncategorical) parameters.

An outline of the rest of the paper is as follows. Section 2 sets the problem, and section 3 gives a detailed discussion of the ANOVA decomposition. In section 4, we first discuss PCA for each ANOVA term and active training for each local GP model, and next present our novel overall ANOVA-GP emulator. Numerical results are discussed in section 5. Section 6 concludes the paper.

2. Problem setting

Let D denote a physical domain (in $\mathbb{R}^2$ or $\mathbb{R}^3$) which is bounded, connected and with a polygonal boundary ∂D. Suppose $\xi = [\xi_1, \ldots, \xi_m]^T$ is a m-dimensional vector which collects a finite number of independent random variables and the probability density function of ξ is denoted by $\pi(\xi)$. Without loss of generality, we further assume that ξ has a bounded and connected support I^m, where I is a real closed interval. In this paper, we consider physical problems governed by PDEs over the physical domain D and boundary conditions on the boundary ∂D, which can be stated as: find a stochastic function $u_{sol}(x, \xi) : D \times I^m \to \mathbb{R}$, such that

$$\mathcal{L}(x, \xi;\ u_{sol}(x, \xi)) = f(x, \xi)\ \ \forall (x, \xi) \in D \times I^m, \tag{1}$$

$$\mathcal{B}(x, \xi;\ u_{sol}(x, \xi)) = g(x, \xi)\ \forall (x, \xi) \in \partial D \times I^m, \tag{2}$$

where $\mathcal{L}$ is a partial differential operator and $\mathcal{B}$ is a boundary operator, both of which can have random coefficients. Given a realization of ξ which is denoted by $\xi^{(j)}$ ($j \in \mathbb{N}$), a simulator (e.g., the finite element method [2]) can provide approximate values of $u_{sol}(x, \xi)$ on given physical grid points, which result in a high-dimensional output. We denote this output as

$$y^{(j)} := u\left(\xi^{(j)}\right) := \left[u_{sol}\left(x^{(1)}, \xi^{(j)}\right), \ldots, u_{sol}\left(x^{(d)}, \xi^{(j)}\right)\right]^T \in \mathbb{R}^d, \tag{3}$$

where d is the number of grid points (or the finite element degrees of freedom) and $x^{(k)}, k = 1, \ldots, d$ are the locations of the grid points. Letting $\mathcal{O} \subset \mathbb{R}^d$ denote the manifold consisting of $u(\xi)$ associated with all realizations of ξ, a simulator can be viewed as a mapping $\chi : I^m \to \mathcal{O}$. The inputs and the outputs of χ are both high-dimensional in this general setting, which causes difficulties for applying traditional GP methods. For this purpose, we in this work provide a novel ANOVA-GP surrogate for χ, where ANOVA decomposition is conducted to decompose the high-dimensional inputs into a union of low-dimensional local inputs. For each local input, PCA is applied to result in a reduced dimensional representation of the corresponding local output. After that, local GP models are built for the PCA coefficients. The next section is to review the ANOVA decomposition following the presentation in [21–27], while PCA for the outputs and our overall ANOVA-GP strategy are presented in section 4.

3. ANOVA decomposition

Let $\mathcal{P}$ be the set consisting of coordinate indices $\{1, 2, \ldots, m\}$. Any non-empty subset $t \subseteq \mathcal{P}$ is referred to as an ANOVA index, and the elements of t are sorted in ascending order, while its cardinality is denoted by $|t|$. For a given t, let ξ_t denote a $|t|$-dimensional vector that includes components of the vector $\xi \in I^m$ indexed by t. For example, if $t = \{1, 3, 4\}$, then $|t| = 3$ and $\xi_t = [\xi_1, \xi_3, \xi_4]^T \in I^3$. Letting $\mathrm{d}\mu$ denote a given probability measure on I^m, the ANOVA decomposition of the simulator output $u(\xi)$ of the problem (1)–(2) can be expressed as

$$u(\xi) = \sum_{t \subseteq \mathcal{P}} u_t(\xi_t). \tag{4}$$

In (4), each term on the right hand side is defined recursively through

$$u_t(\xi_t) = \int_{I^{m-|t|}} u(\xi)\, \mathrm{d}\mu\left(\xi_{\mathcal{P}\setminus t}\right) - \sum_{w\subset t} u_w(\xi_w), \tag{5}$$

starting with

$$u_\emptyset = \int_{I^m} u(\xi)\, \mathrm{d}\mu(\xi), \tag{6}$$

where $\mathrm{d}\mu\left(\xi_{\mathcal{P}\setminus t}\right) := \prod_{i\in\mathcal{P}\setminus t} \mathrm{d}\mu(\xi_i)$, since $\{\xi_i\}_{i=1}^m$ are assumed to be independent. Note that $u(\xi)$ is a vector and integrals involving them (e.g., (5) and (6)) are defined componentwise. In this paper, we call $u_t(\xi_t)$ a $|t|$-th order ANOVA term and t a $|t|$-th order index.

When the ordinary Lebesgue measure is used in (5)–(6), (4) is referred to as the classic ANOVA decomposition, and each expansion term is

$$u_t(\xi_t) = \int_{I^{m-|t|}} u(\xi) \prod_{i\in\mathcal{P}\setminus t} \mathrm{d}\xi_i - \sum_{w\subset t} u_w(\xi_w), \tag{7}$$

and

$$u_\emptyset = \int_{I^m} u(\xi)\, \mathrm{d}\xi. \tag{8}$$

Computing each term (7) in the classic ANOVA decomposition requires computing integrals over $I^{m-|t|}$. When $|t|$ is small, $I^{m-|t|}$ has a high dimensionality, and computing integrals over it is expensive. To alleviate this difficulty, anchored ANOVA methods [28] are developed, and are reviewed as follows.

3.1. Anchored ANOVA decomposition

As discussed in [28,24,26], the idea of anchored ANOVA decomposition is to replace the Lebesgue measure used in (7)–(8) by the Dirac measure

$$\mathrm{d}\mu(\xi) := \delta(\xi - c)\, \mathrm{d}\xi = \prod_{i=1}^m \delta(\xi_i - c_i)\, \mathrm{d}\xi_i, \tag{9}$$

where $c = [c_1, c_2, \ldots, c_m]^T \in I^m$ is a given anchor point. With the Dirac measure, each term in (5) is

$$u_t(\xi_t) = u\left(\xi^{c,t}\right) - \sum_{w\subset t} u_w(\xi_w), \tag{10}$$

where the initial term is set to $u_\emptyset = u(c)$ and $\xi^{c,t} := [\xi_1^{c,t}, \ldots, \xi_m^{c,t}]^T \in I^m$ is defined through

$$\xi_i^{c,t} := \begin{cases} c_i & \text{for } i \in \{1,\ldots,m\}\setminus t \\ \xi_i & \text{for } i \in t \end{cases}. \tag{11}$$

The anchored ANOVA decomposition expresses the simulator output $u(\xi)$ by the knowledge of its values on lines, planes and hyper-planes passing through the anchor point c [21]. Here comes a natural question that how to choose the anchor point. Generally, the anchor point can be chosen arbitrarily since the ANOVA decomposition (4) is always exact. However, an appropriately chosen anchor point enables the decomposition to give an accurate approximation with a small number of expansion terms [28,29], which give computational efficiency (the selection procedure of ANOVA terms is discussed in the next section). In [28,29], it is shown that a good choice is the input sample point where the corresponding output sample equals or is close to the mean of the output. However, the mean of the output is not given a priori in our setting, and it is not trivial to find the input sample point which gives an output sample close to the mean of the output. As shown in [30,23], an optimal choice is the mean of the input, and we use this choice of the anchor point for all numerical studies in this paper.

It is clear that the whole index set $\{t \mid t \subseteq \mathcal{P}\}$ contains a large number of terms when m is large, and especially, the m-th order index $t = \{1,\ldots,m\}$ is included, which causes challenges to compute the right hand side of (4). However, in practical computation, not all expansion terms in (4) need to be computed—only low order ANOVA terms are typically considered to be active and need to be computed. Denoting a selected index set by $\mathcal{J}$ which is a subset of the whole index set $\{t \mid t \subseteq \mathcal{P}\}$, an approximation of the solution $u(\xi)$ is written as

$$u(\xi) \approx u_{\mathcal{J}}(\xi) := \sum_{t \in \mathcal{J}} u_t(\xi_t), \tag{12}$$

where $u_t(\xi_t)$ is defined in (5). Next, we review the adaptive construction procedure for the index set $\mathcal{J}$ following [24].

3.2. Adaptive index construction

For each $i = 0, \ldots, m$, the set consisting of selected i-th order indices is denoted by $\mathcal{J}_i$, while $\mathcal{J} = \cup_{i=0}^{m} \mathcal{J}_i$. For the zeroth order index, we set $\mathcal{J}_0 = \{\emptyset\}$ and $|\emptyset| = 0$, and $u(c)$ is computed using a given simulator (e.g., the finite element method). Supposing that $\mathcal{J}_i$ is known for a given order $0 \le i \le m-1$, $\mathcal{J}_{i+1}$ is constructed based on $\mathcal{J}_i$ as follows. First, a candidate index set $\widehat{\mathcal{J}}_{i+1}$ is constructed as

$$\widehat{\mathcal{J}}_{i+1} := \left\{ t \;\middle|\; |t| = i+1, \text{and any } s \subset t \text{ with } |s| = i \text{ satisfies } s \in \mathcal{J}_i \right\}. \tag{13}$$

For each $t \in \widehat{\mathcal{J}}_{i+1}$, the contribution weight of $u_t(\xi_t)$ is defined as

$$\gamma_t := \frac{\| \boldsymbol{E}(u_t(\xi_t)) \|_{L_2}}{\left\| \sum_{s \in \mathcal{J}_0 \cup \cdots \cup \mathcal{J}_{|t|-1}} \boldsymbol{E}(u_s(\xi_s)) \right\|_{L_2}}, \tag{14}$$

which measures the relative importance of the index t [24]. In (14), $\|u_t(\xi_t)\|_{L_2}$ is the functional L_2 norm of the approximation function associated with $u_t(\xi_t)$ (e.g., the finite element approximation function with coefficients defined by $u_t(\xi_t)$ [2]), and $\boldsymbol{E}(u_t(\xi_t))$ denotes the mean function of u_t that is defined as

$$\boldsymbol{E}(u_t(\xi_t)) = \int_{I^{|t|}} u_t(\xi_t)\, \pi_t(\xi_t)\, \mathrm{d}\xi_t, \tag{15}$$

where $\pi_t(\xi_t)$ is the marginal probability density function of ξ_t. This mean function can be approximated using the Clenshaw-Curtis tensor quadrature rule [31,32,26], i.e.,

$$\hat{\boldsymbol{E}}(u_t(\xi_t)) := \sum_{\xi_t^{(k)} \in \Xi_t} u_t\left(\xi_t^{(k)}\right) \pi_t\left(\xi_t^{(k)}\right) w\left(\xi_t^{(k)}\right), \; k = 1, 2, \ldots, |\Xi_t|, \tag{16}$$

where Ξ_t contains the Clenshaw-Curtis tensor quadrature points, $\{w(\xi_t^{(k)})\}$ for $k = 1, \ldots, |\Xi_t|$ are the corresponding weights, and $|\Xi_t|$ is the size of Ξ_t. After that, the set $\mathcal{J}_{i+1}$ is formed through the $(i+1)$-th order indices with $\gamma_t \ge tol_{index}$, i.e., $\mathcal{J}_i := \{t \,|\, t \in \widehat{\mathcal{J}}_i \text{ and } \gamma_t \ge tol_{index}\}$, where tol_{index} is a given tolerance. This hierarchical construction procedure stops when $\widehat{\mathcal{J}}_{i+1} = \emptyset$. The above procedure to adaptively select ANOVA terms is introduced in [24] and summarized in Appendix A (Algorithm 4).

4. ANOVA Gaussian process modeling

In this section, our novel ANOVA Gaussian process (ANOVA-GP) modeling strategy is presented. This new strategy is based on building GP models for each ANOVA term. It is clear that, the dimensionality of each ANOVA term in (12) is the same as that of the simulator output, e.g., the finite element degrees of freedom, which is high-dimensional. As discussed in section 1, it is challenging to apply standard GP models for problems with high-dimensional outputs. To result in a reduced dimensional representation of the output, we apply the principal component analysis (PCA) [33,34] for each ANOVA term. After that, based on the data sets obtained in the ANOVA decomposition step (see section 3.2), an active training procedure is developed to construct the GP models for each PCA mode. Our overall ANOVA-GP procedure is summarized at the end of this section.

4.1. Principal component analysis

The principal component analysis [33] is to find subspaces in which observed data can be approximated well. The basis vectors of these subspaces are called the principal components, which are also referred to as proper orthogonal decomposition bases [35,36]. In this work, PCA is applied to obtain reduced dimensional representations for each ANOVA term $u_t(\xi_t)$ in (12). To conduct PCA for $u_t(\xi_t)$, a data set consisting of samples of $u_t(\xi_t)$ is required. In this section, the data set of $u_t(\xi_t)$ is generically denoted by $\vartheta_t := \{y_t^{(j)} \,|\, y_t^{(j)} = u_t(\xi_t^{(j)}) \in \mathbb{R}^d \text{ and } j = 1, \ldots, N\}$, where N denotes the size of $|\vartheta_t|$. Note that the ANOVA decomposition procedure (see section 3.2) gives a data set $\Theta_t := \{y_t^{(k)} = u_t(\xi_t^{(k)}) \,|\, \xi_t^{(k)} \in \Xi_t, \text{ and } k = 1, \ldots, |\Xi_t|\}$ for each $t \in \mathcal{J}$. This data set Θ_t can be used as an initial choice for ϑ_t to conduct PCA, while an active training procedure based on our new selection criterion provides additional sample points, which is discussed in section 4.2.

For each data set ϑ_t for $t \in \mathcal{J}$, the first step of PCA is to normalize the sample mean as follows

1) $\mu_t = \frac{1}{N}\sum_{j=1}^{N} y_t^{(j)}$, for $j = 1, \ldots, N$,

2) $y_t^{(j)} \leftarrow y_t^{(j)} - \mu_t$.

After that, the empirical covariance matrix is assembled

$$\Sigma = \frac{1}{N}\sum_{j=1}^{N} y_t^{(j)} \left(y_t^{(j)}\right)^T.$$

The eigenvalues and the eigenvectors of Σ are denoted by $\lambda_1 \geq \ldots \geq \lambda_N$ and $v_1, \ldots, v_N$ respectively. For a given tolerance tol_{pca}, the first R eigenvectors $\{v_1, \ldots, v_R\}$ satisfying $\sum_{j=1}^{R}\lambda_j / \sum_{j=1}^{N}\lambda_j > 1 - tol_{pca}$ but $\sum_{j=1}^{R-1}\lambda_j / \sum_{j=1}^{N}\lambda_j \leq 1 - tol_{pca}$, are referred to as the principal components. In addition, $V_t := [v_1, \ldots, v_R]$ denotes the matrix collecting the principal components. Details of PCA for each ANOVA term $u_t(\xi)$, $t \in \mathcal{J}$ are summarized in Algorithm 1.

With the principal components, each ANOVA term $u_t(\xi_t) \in \mathbb{R}^d$ for an arbitrary realization of ξ_t can be approximated as:

$$u_t(\xi_t) \approx V_t \tilde{u}_t\left(\xi_t\right) + \mu_t,$$

where

$$\tilde{u}_t\left(\xi_t\right) = [\tilde{u}_{t,1}\left(\xi_t\right), \ldots, \tilde{u}_{t,R}(\xi_t)]^T \in \mathbb{R}^R, \quad \tilde{u}_{t,r}\left(\xi_t\right) := v_r^T\left(u_t(\xi_t) - \mu_t\right) \text{ for } r = 1, \ldots, R. \tag{17}$$

In the following, $\tilde{u}_t(\xi_t)$ is referred to as the principal component representation (PC representation) of $u_t(\xi_t)$.

Algorithm 1 Principal component analysis for each ANOVA term $u_t(\xi_t)$, $t \in \mathcal{J}$.

Input: A data set $\vartheta_t := \left\{ y_t^{(j)} \;\middle|\; y_t^{(j)} = u_t\left(\xi_t^{(j)}\right) \in \mathbb{R}^d \text{ and } j = 1, \ldots, N \right\}$.

1: Compute the sample mean: $\mu_t = \frac{1}{N}\sum_{j=1}^{N} y_t^{(j)}$.

2: Normalize the data: $y_t^{(j)} \leftarrow y_t^{(j)} - \mu_t$ for $j = 1, \ldots, N$.

3: Construct the covariance matrix: $\Sigma = \frac{1}{N}\sum_{j=1}^{N} y_t^{(j)}\left(y_t^{(j)}\right)^T$.

4: Compute eigenpairs (λ_k, v_k) of Σ, where $k = 1, \ldots, N$ and $\lambda_1 \geq \ldots \geq \lambda_N$.

5: Select the first R eigenvectors $\{v_1, \ldots, v_R\}$ such that $\frac{\sum_{j=1}^{R}\lambda_j}{\sum_{j=1}^{N}\lambda_j} > 1 - tol_{pca}$, but $\frac{\sum_{j=1}^{R-1}\lambda_j}{\sum_{j=1}^{N}\lambda_j} \leq 1 - tol_{pca}$.

6: Compute $\tilde{u}_t\left(\xi_t^{(j)}\right) = \left[\tilde{u}_{t,1}\left(\xi_t^{(j)}\right), \ldots, \tilde{u}_{t,R}\left(\xi_t^{(j)}\right)\right]^T \leftarrow \left[v_1^T y_t^{(j)}, \ldots, v_R^T y_t^{(j)}\right]^T$, where $j = 1, \ldots, N$.

7: Construct data vectors for PCA coefficients: $\alpha_{t,r} = \left[\tilde{u}_{t,r}\left(\xi_t^{(1)}\right), \ldots, \tilde{u}_{t,r}\left(\xi_t^{(N)}\right)\right]^T$ for $r = 1, \ldots, R$.

Output: The sample mean μ_t, the principal component matrix $V_t := [v_1, \ldots, v_R]$, eigenvalues λ_r and data vectors $\alpha_{t,r}$ for $r = 1, \ldots, R$.

4.2. Gaussian process regression with active training

In this section for each $t \in \mathcal{J}$, following the active data selection method developed in [18], a Gaussian process modeling strategy with active training is proposed for each PC representation $\tilde{u}_t(\xi_t) = [\tilde{u}_{t,1}(\xi_t), \ldots, \tilde{u}_{t,R}(\xi_t)]^T \in \mathbb{R}^R$ (see (17)). Due to the compression obtained through PCA, the dimension R is typically very small and independent of the dimension of the simulator output (e.g., the finite element degrees of freedom). So, it is computationally feasible to construct GP models for each ANOVA term independently.

A Gaussian process is a collection of random variables, and any finite combinations of these random variables are joint Gaussian distributions. In our setting, for each realization of ξ_t, $\tilde{u}_{t,r}(\xi_t)$ is considered to be a random variable in a Gaussian process. Following the presentation in [17], each of the prior GP models is denoted by $\tilde{u}_{t,r}(\xi_t) \sim \mathcal{GP}(m(\xi_t), \mathfrak{c}(\xi_t, \xi_t'))$ where $m(\cdot)$ is the mean function and $\mathfrak{c}(\cdot,\cdot)$ is the covariance function of the Gaussian process $\mathcal{GP}$ that needs to be trained. The Gaussian process is specified by its mean function and covariance function [37]. In this work, the mean function is set to $m(\xi_t) = 0$, and the covariance function is set to a noisy squared exponential function

$$\mathfrak{c}\left(\xi_t, \xi_t'\right) = \rho_1^2 \cdot \exp\left(-\left(\xi_t - \xi_t'\right)^T \operatorname{diag}\left(\ell_1, \ldots, \ell_M\right)^{-1}\left(\xi_t - \xi_t'\right)/2\right) + \rho_2^2 \cdot \delta\left(\xi_t, \xi_t'\right). \tag{18}$$

The last term in (18) is called 'jitter' [38], $\delta(\xi_t, \xi_t')$ is a Kronecker delta function which is one if $\xi_t = \xi_t'$ and zero otherwise, and $\operatorname{diag}(\ell_1, \ldots, \ell_M)$ is a diagonal matrix. The hyperparameters $\ell_1, \ldots, \ell_M$ and ρ_1^2, ρ_2^2 are square correlation lengths and signal variances respectively. Denoting $\beta = [\ell_1, \ldots, \ell_M, \rho_1, \rho_2]^T$, the hyperparameters can be determined through minimizing the following negative log marginal likelihood $\mathcal{M}(\beta)$:

$$\mathcal{M}(\beta) = -\log p\left(\alpha_{t,r} | \beta\right) = \frac{1}{2}\log\det\left(C(\beta)\right) + \frac{1}{2}\alpha_{t,r}^T C^{-1}(\beta)\,\alpha_{t,r} + \frac{N}{2}\log(2\pi), \tag{19}$$

where $\alpha_{t,r} = [\tilde{u}_{t,r}(\xi_t^{(1)}), \ldots, \tilde{u}_{t,r}(\xi_t^{(N)})]^T$ is the training target and $C(\beta)$ is the covariance matrix with entries $C(\beta)_{jk} = \mathfrak{c}(\xi_t^{(j)}, \xi_t^{(k)})$ for $j,k = 1, \ldots, N$. Minimizing $\mathcal{M}(\beta)$ is a non-convex optimization problem [39], and we use the MATLAB toolbox [40] to solve it, where conjugate gradient methods are included [41].

Once the hyperparameters are determined, from the joint distribution of $\tilde{u}_{t,r}(\xi_t)$ and $\alpha_{t,r}$, the conditional predictive distribution for any arbitrary realization of ξ_t is:

$$\tilde{u}_{t,r}\left(\xi_t\right) \mid \alpha_{t,r}, \beta \sim U_{t,r}(\xi_t) := \mathcal{GP}\left(m_r'(\xi_t, \beta), v_r'(\xi_t, \beta)\right), \tag{20}$$

where $m_r'(\xi_t, \beta) = c_*^T C(\beta)^{-1} \alpha_{t,r}$, $v_r'(\xi_t, \beta) = \mathfrak{c}(\xi_t, \xi_t) - c_*^T C(\beta)^{-1} c_*$, and $c_* = [\mathfrak{c}(\xi_t, \xi_t^{(i)}), \ldots, \mathfrak{c}(\xi_t, \xi_t^{(N)})]^T$ (see [37]). Collecting the GP models for each PCA mode, the GP model for the overall PC representation for $\tilde{u}_t\left(\xi_t\right)$ (17) is denoted by $U_t(\xi_t) := [U_{t,1}(\xi_t), \ldots, U_{t,R}(\xi_t)]^T$ for each $t \in \mathcal{J}$. For a given realization of ξ_t, the predictive mean of $U_t(\xi_t)$ is

$$\overline{U}_t(\xi_t) := \left[m_1'(\xi_t, \beta), \ldots, m_R'(\xi_t, \beta)\right]^T. \tag{21}$$

With the principal components and the GP models for PC representations, each ANOVA term $u_t(\xi_t)$ can be approximated as the following *local GP model* (the setting of a global GP model to approximate the overall problem (1)–(2) is discussed in section 5),

$$\hat{u}_t(\xi_t) := V_t U_t(\xi_t) + \mu_t, \tag{22}$$

where V_t is the matrix consisting of the principal components and μ_t is the sample mean generated by Algorithm 1. The predictive mean of the local GP model is

$$\overline{u}_t\left(\xi_t\right) := V_t \overline{U}_t(\xi_t) + \mu_t. \tag{23}$$

It is clear that building a local GP model (22) involves two main procedures: PCA and GP regression for each PCA mode. Both of these procedures are determined by the data set ϑ_t (the input of Algorithm 1). Our strategy is to use the data set Θ_t generated by the ANOVA decomposition step (see section 3.2) as an initial input data set to conduct PCA and to build the GP model for each PCA mode, i.e., initially set $\vartheta_t := \Theta_t$. After that, following [18], an active training method is developed to argument the training data set ϑ_t gradually to result in an accurate local GP model for $u_t(\xi_t)$, which proceeds as follows. First, a candidate parameter sample set Ψ is constructed using realizations of ξ_t (different from the quadrature points Ξ_t for ANOVA decomposition in section 3.2). Second, for each sample in $\xi_t \in \Psi$, a variance indicator of the current GP model is computed as

$$\tau\left(\xi_t\right) := \sum_{r=1}^{R} \lambda_r v_r'\left(\xi_t, \beta\right) \Big/ \sum_{r=1}^{R} \lambda_r\,, \tag{24}$$

where $\lambda_1, \ldots, \lambda_R$ are the eigenvalues generated in PCA with the current input data set, and $v_r'(\xi_t, \beta)$ is the variance of the current GP model for each PCA mode (see (20)). Third, the input sample point with the largest variance indicator value $\xi_t^* = \max_{\xi_t \in \Psi} \tau(\xi_t)$ is selected to augment the input data set ϑ_t, and the local GP model is reconstructed with this augmented data set. The second and the third steps are repeated until ϑ_t includes N_{train} data points, where $N_{train} > |\Xi_t|$ is a given number. Details of this active training procedure are shown in Algorithm 2.

Algorithm 2 Local GP modeling with active training for each ANOVA term $u_t\left(\xi_t\right)$, $t \in \mathcal{J}$.

Input: The number of training points N_{train}, and an initial training data set Θ_t.
1: Initialize a candidate set Ψ consisting of realizations of ξ_t, with size $|\Psi| > N_{train}$.
2: Initialize the training data set $\vartheta_t := \Theta_t$.
3: Use Algorithm 1 with ϑ_t to obtain the sample mean μ_t, the principal component matrix $V_t := [v_1, \ldots, v_R]$, the eigenvalues λ_r and the data vectors $\alpha_{t,r}$ for $r = 1, \ldots, R$.
4: Train GP models $U_{t,r} := \mathcal{GP}\left(m_r'(\xi_t, \beta), v_r'(\xi_t, \beta)\right)$ based on the training data vectors $\alpha_{t,r}$ for each PCA mode $r = 1, \ldots, R$ (see (19)–(20)).
5: **if** $|\vartheta_t| < N_{train}$ **then**
6: For each $\xi_t \in \Psi$, compute the variance indicator $\tau\left(\xi_t\right) := \sum_{r=1}^{R} \lambda_r v_r'\left(\xi_t, \beta\right) \Big/ \sum_{r=1}^{R} \lambda_r$.
7: Find $\xi_t^* = \arg\ \max_{\xi_t \in \Psi} \tau\left(\xi_t\right)$.
8: Update the training data set: $\vartheta_t = \vartheta_t \cup \left\{u_t\left(\xi_t^*\right)\right\}$.
9: Remove ξ_t^* from the candidate set: $\Psi = \Psi \setminus \xi_t^*$.
10: Go to line 3.
11: **end if**
12: Construct the GP model for the PC representation: $U_t = [U_{t,1}, \ldots, U_{t,R}]^T \in \mathbb{R}^R$ with $U_{t,r} := \mathcal{GP}\left(m_r'(\xi_t, \beta), v_r'\left(\xi_t, \beta\right)\right)$ for $r = 1, \ldots, R$.
13: Construct the local GP model: $\hat{u}_t(\xi_t) = V_t U_t(\xi_t) + \mu_t$.
Output: The local GP emulator $\hat{u}_t(\xi_t)$.

4.3. Overall ANOVA-GP model

With a given simulator for the problem (1)–(2), our overall ANOVA-GP modeling proceeds as the following three main steps. First, ANOVA decomposition is conducted (see section 3.2), which gives an effective index set $\mathcal{J}$ and initial training data sets $\Theta_t := \{y_t^{(k)} = u_t(\xi_t^{(k)}) \,|\, \xi_t^{(k)} \in \Xi_t, \text{ and } k = 1, \ldots, |\Xi_t|\}$ for $t \in \mathcal{J}$. Denoting the highest ANOVA order by $h := \max_{t\in\mathcal{J}} |t|$, for each i-th order index (i.e., $t \in \mathcal{J}$ and $|t| = i$), the number of training points for the local GP model is set to $N_{agp,i}$ (i.e., $N_{train} = N_{agp,i}$ in Algorithm 2). We next define $N_{agp} := [N_{agp,1}, \ldots, N_{agp,h}]^T$. Considering the intrinsic property of Gaussian process modeling that more training data are needed for problems with higher-dimensional inputs [19], we set $N_{agp,1} < \ldots < N_{agp,h}$. After that, the local GP models $\hat{u}_t(\xi_t)$ for each ANOVA index $t \in \mathcal{J}$ are built using Algorithm 2 with Θ_t and $N_{train} = N_{agp,|t|}$. Finally, the overall ANOVA-GP model is assembled as

$$\hat{u}_{\mathcal{J}}(\xi) := \sum_{t\in\mathcal{J}} \hat{u}_t(\xi_t). \tag{25}$$

For each realization of ξ, the predictive mean of the ANOVA-GP model is,

$$\overline{u}_{\mathcal{J}}(\xi) := \sum_{t\in\mathcal{J}} \overline{u}_t(\xi_t), \tag{26}$$

where $\overline{u}_t(\xi_t)$ is the predictive mean of the local GP model defined in (23). This ANOVA-GP modeling procedure is summarized in Algorithm 3.

Algorithm 3 ANOVA-GP modeling.

Input: A simulator for (1)–(2) and the probability density function of ξ.
1: Conduct ANOVA decomposition (see section 3.2) to obtain an effective index set $\mathcal{J}$ and data sets $\Theta_t := \{y_t^{(k)} = u_t(\xi_t^{(k)}) \,|\, \xi_t^{(k)} \in \Xi_t, \text{ and } k = 1, \ldots, |\Xi_t|\}$ for $t \in \mathcal{J}$.
2: Get the highest ANOVA order $h := \max_{t\in\mathcal{J}} |t|$.
3: Set the number of training points for each ANOVA order $N_{agp} := [N_{agp,1}, \ldots, N_{agp,h}]^T$.
4: **for** $t \in \mathcal{J}$ **do**
5: Build the local GP model $\hat{u}_t(\xi_t)$ using Algorithm 2 with Θ_t and $N_{train} = N_{agp,|t|}$.
6: **end for**
7: Assemble the ANOVA-GP emulator: $\hat{u}_{\mathcal{J}}(\xi) := \sum_{t\in\mathcal{J}} \hat{u}(\xi_t)$.
Output: The ANOVA-GP emulator $\hat{u}_{\mathcal{J}}(\xi)$.

As the ANOVA decomposition is a popular strategy for resolving high-dimensional problems, many ANOVA based methods have been actively developed in the context of uncertainty quantification, e.g., the ANOVA collocation methods [24,26] and the ANOVA reduced basis methods [42,43], while our ANOVA-GP provides the following benefits. First, ANOVA-GP is a non-intrusive method—only repetitive runs of the given simulator are required to generate the training data sets, so that it is easy to implement and generalize to complex engineering systems. Second, in ANOVA-GP, each ANOVA term is modeled by a local GP model, which is an interpretable Bayesian model and can provide error indicators based on the variance. These error indicators naturally result in the systematic active training procedure for each ANOVA term in section 4.2. In addition, as discussed in [18], the uncertainty in these GP models (and the overall ANOVA-GP model) and the uncertainty in input parameters can be taken into account using the Bayesian theory.

5. Numerical study

In this section, two kinds of model problems are studied to illustrate the effectiveness of our ANOVA-GP strategy: stochastic diffusion problems in section 5.1 and stochastic incompressible flow problems in section 5.2. For comparison, a direct combination of Gaussian process modeling and PCA is considered, which is referred to as the standard Gaussian process (S-GP) in the following. While S-GP is originally developed by [15] for model calibration, we here modify it to build surrogates for the problem (1)–(2). The S-GP emulator for (1)–(2) is denoted by $u_{SGP}(\xi)$, and details of our setting for constructing $u_{SGP}(\xi)$ are summarized in Algorithm 5 in Appendix A. Using the notation in Algorithm 5, the predictive mean of $u_{SGP}(\xi)$ is denoted by

$$\overline{u}_{SGP}(\xi) := V\overline{U} + \mu, \tag{27}$$

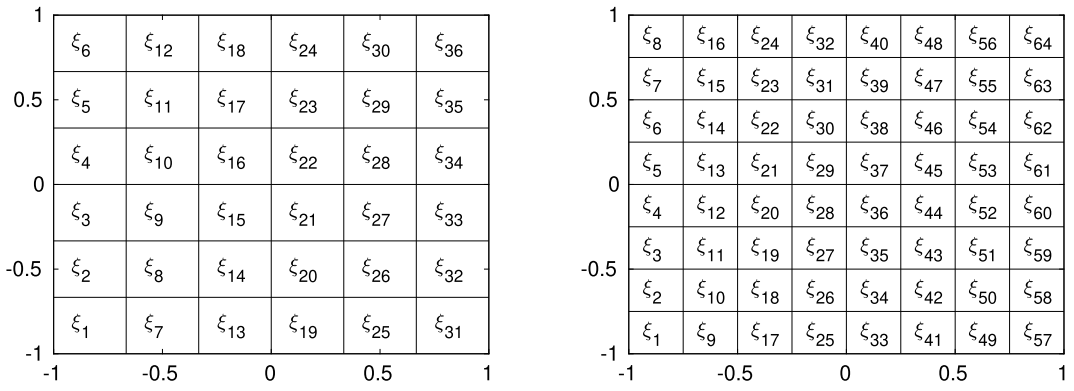
where $\overline{U} := [m'_1, \ldots, m'_R]^T$, $m'_1, \ldots, m'_R$ and μ are defined in Algorithm 5.

5.1. Test problem 1: diffusion problems

We consider the following governing equations posed on the physical domain $D = (-1,\ 1) \times (-1,\ 1)$:

$$-\nabla \cdot [a(x,\xi)\,\nabla u_{sol}(x,\xi)] = 1 \quad \text{in} \quad D \times I^m, \tag{28}$$

$$u_{sol}(x,\xi) = 0 \quad \text{on} \quad \partial D \times I^m. \tag{29}$$



(a) $6 \times 6 = 36$ subdomains (b) $8 \times 8 = 64$ subdomains

Fig. 1. Partitionings of the physical domain with $N_D = 36$ and 64 subdomains, test problem 1.

Table 1
Number of effective ANOVA terms, test problem 1.

N_D	$\lvert\widehat{\mathcal{J}}_1\rvert$	$\lvert\mathcal{J}_1\rvert$	$\lvert\widehat{\mathcal{J}}_2\rvert$	$\lvert\mathcal{J}_2\rvert$	$\lvert\widehat{\mathcal{J}}_3\rvert$	$\lvert\mathcal{J}_3\rvert$
36	36	36	630	100	80	0
64	64	64	2016	172	120	0

Dividing the physical domain D into N_D subdomains, each of which is denoted by D_k for $k = 1, \ldots, N_D$, the permeability coefficient $a(x, \xi)$ is defined to be a piecewise constant function

$$a(x, \xi)\,|_{D_k} = \xi_k,\ k = 1, \ldots, N_D, \tag{30}$$

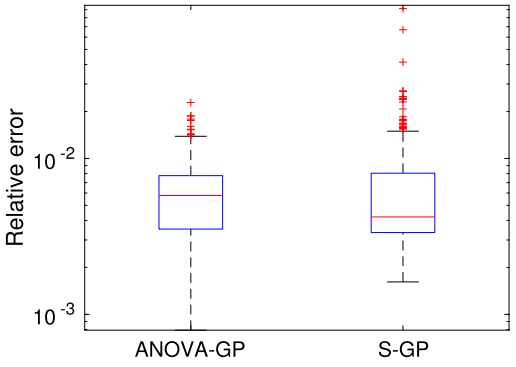
where $\xi_1, \ldots, \xi_{N_D}$ are independently and uniformly distributed in $[0.01, 1]$ and $m = N_D$ for this test problem. Two cases of physical domain partitionings are considered, which are shown in Fig. 1 and include 36 and 64 parameters respectively. For each realization of ξ, the simulator for (28)–(29) is set to the finite element method [44,2], where a bilinear $\boldsymbol{Q}_1$ finite element approximation is used to discretize the physical domain with a 65×65 grid, i.e., the dimension of the simulator output is $d = 4225$.

As discussed in section 4.3, the first step of our ANOVA-GP strategy is to conduct ANOVA decomposition (see section 3.2) for (28)–(29). In the adaptive ANOVA decomposition procedure, the quadrature rule is set to the tensor products of one-dimensional Clenshaw-Curtis quadrature with five quadrature points [45], and the tolerance for selecting effective indices is set to $tol_{index} = 10^{-4}$. The tolerance of PCA (in Algorithm 1) for both ANOVA-GP and S-GP are set to $tol_{pca} = 10^{-2}$. Table 1 shows sizes of the index sets $\widehat{\mathcal{J}}_i$ constructed by (13) and sizes of the selected index sets $\mathcal{J}_i$ at each ANOVA order $i = 1, 2, 3$. For the two cases of physical domain partitionings ($N_D = 36$ and 64), all first order indices and a fraction of second order indices are selected, while there is no third order index selected, which is consistent with the results in [26,42].

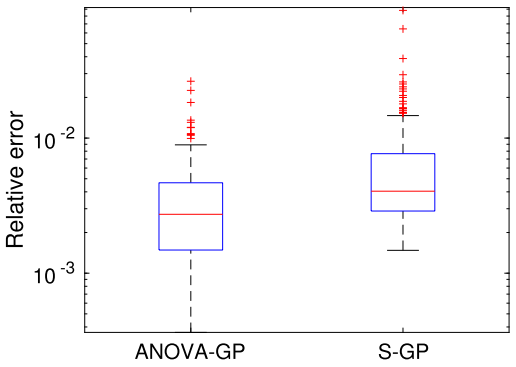
Accuracy of our ANOVA-GP emulator and the standard GP (S-GP) emulator is assessed as follows. First, 200 samples of ξ is generated and denoted by $\{\xi^{(j)}\}_{j=1}^{200}$, and the corresponding simulator output is computed and denoted by $\{y^{(j)} = u(\xi^{(j)})\}_{j=1}^{200}$. Next, for each $j = 1, \ldots, 200$, we consider the relative error used in [17]

$$\text{Relative error} = \frac{\left\| y_p^{(j)} - y^{(j)} \right\|^2}{\left\| y^{(j)} \right\|^2}, \tag{31}$$

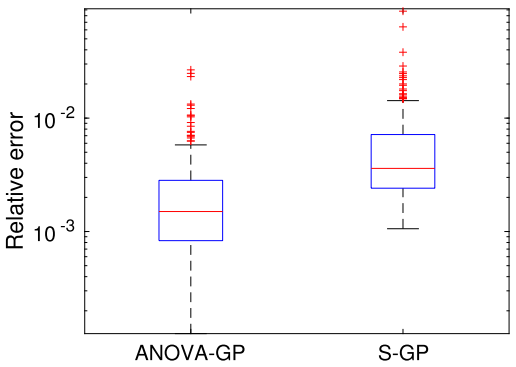
where $\|\cdot\|$ is the standard Euclidean norm. In (31), $y_p^{(j)}$ refers to the predictive mean $\overline{u}_{\mathcal{J}}(\xi^{(j)})$ in (26) when assessing the errors of ANOVA-GP, and it refers to $\overline{u}_{SGP}(\xi^{(j)})$ in (27) when assessing the errors of S-GP. Different numbers of training data points to build the GP models are tested. For ANOVA-GP (Algorithm 3), the following numbers of training points are tested: $N_{agp} = [30, 50]^T$, $[40, 75]^T$ and $[50, 100]^T$, where N_{agp} is defined in section 4.3. For a fair comparison, for S-GP (see Algorithm 5), the number of training points N_{sgp} is set to the number of all training points generated in Algorithm 3, which is $N_{sgp} = N_{agp,1} \times |\mathcal{J}_1| + N_{agp,2} \times |\mathcal{J}_2|$, where $|\mathcal{J}_1|, |\mathcal{J}_2|$ are the numbers of the first and the second order effective ANOVA indices. Fig. 2 shows Tukey box plots of the errors for the test problem with 36 and 64 subdomains. Here, the central line in each box is the median, the lower and the upper edges are the first and the third quartiles respectively, and the crosses are the outliers where the relative errors are large. From Fig. 2, it is clear that as the number of training points increases, our ANOVA-GP has smaller errors than S-GP.



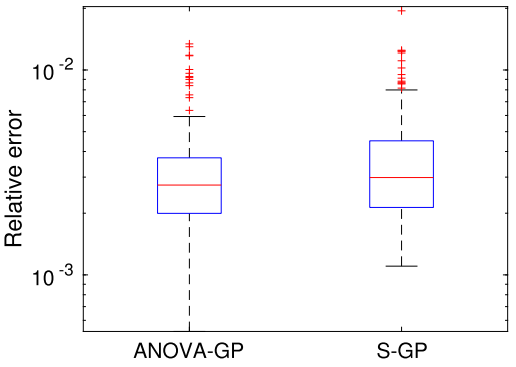
(a) $N_{agp} = [30, 50]^T$, $N_{sgp} = 6080$, for $N_D = 36$ subdomains



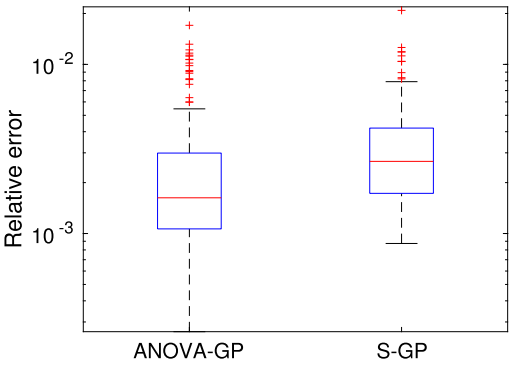
(b) $N_{agp} = [40, 75]^T$, $N_{sgp} = 8940$, for $N_D = 36$ subdomains



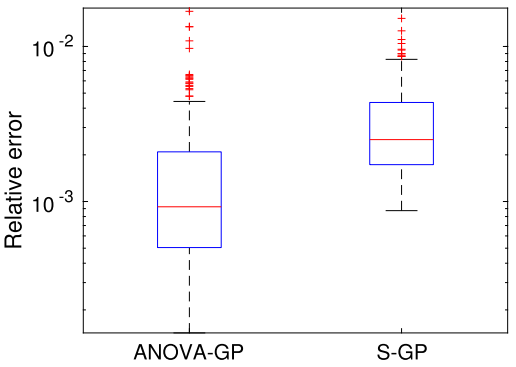
(c) $N_{agp} = [50, 100]^T$, $N_{sgp} = 11800$, for $N_D = 36$ subdomains



(d) $N_{agp} = [30, 50]^T$, $N_{sgp} = 10520$, for $N_D = 64$ subdomains



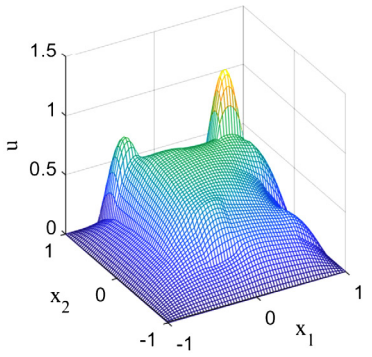
(e) $N_{agp} = [40, 75]^T$, $N_{sgp} = 15460$, for $N_D = 64$ subdomains



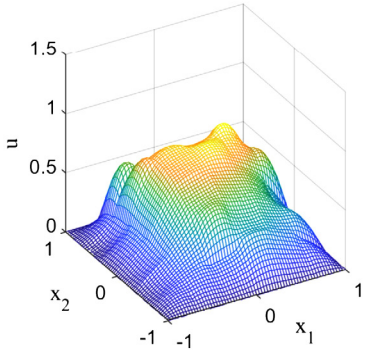
(f) $N_{agp} = [50, 100]^T$, $N_{sgp} = 20400$, for $N_D = 64$ subdomains

Fig. 2. Relative errors of ANOVA-GP and S-GP for 200 test data points, test problem 1.

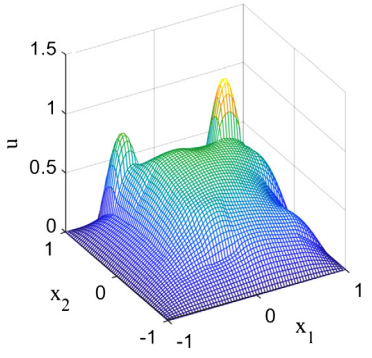
Fig. 3 shows the simulator output and the emulator predictive means corresponding to a given realization of ξ. It is clear that the predictive means of ANOVA-GP are much more accurate than those of S-GP. For example, looking at the simulator output for the case $N_D = 36$ in Fig. 3(a), there is a bump near the top right corner $(1, 1)$. The predictive mean of S-GP in Fig. 3(b) is too smooth and can not show the bump, while our ANOVA-GP output in Fig. 3(c) can capture all details of the simulator output. For the case $N_D = 64$, the predictive mean of ANOVA-GP in Fig. 3(f) is very close to the simulator output (shown in Fig. 3(d)), while the predictive mean of S-GP (Fig. 3(e)) can not capture the bump near $(1, -1)$ which can clearly be seen in Fig. 3(d) and Fig. 3(f).



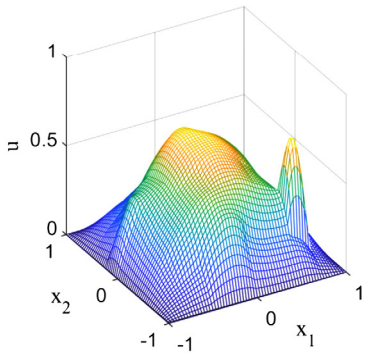
(a) Simulator output, $N_D = 36$



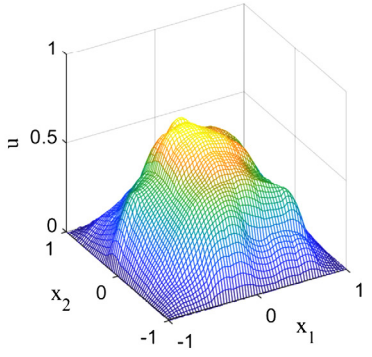
(b) Predictive mean of S-GP, $N_{sgp} = 11800$, $N_D = 36$



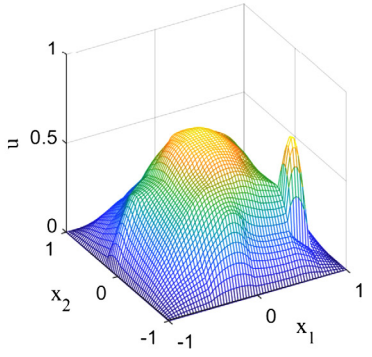
(c) Predictive mean of ANOVA-GP, $N_{agp} = [50, 100]^T$, $N_D = 36$



(d) Simulator output, $N_D = 64$



(e) Predictive mean of S-GP, $N_{sgp} = 20400$, $N_D = 64$

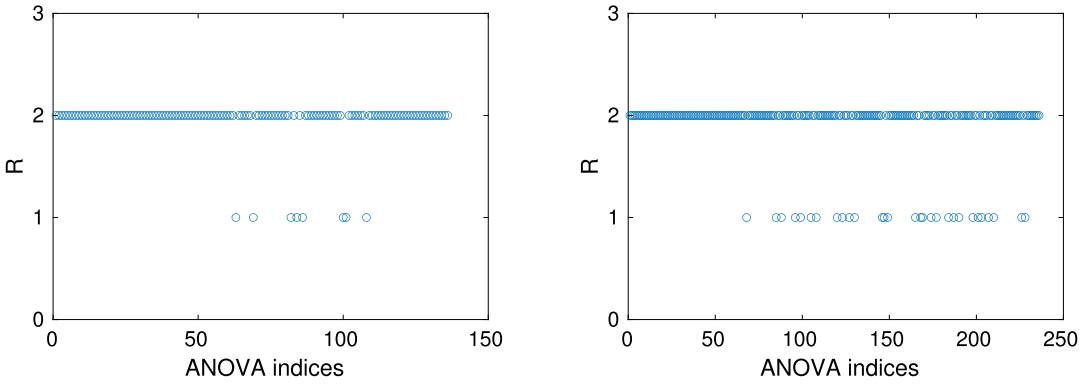


(f) Predictive mean of ANOVA-GP, $N_{agp} = [50, 100]^T$, $N_D = 64$

Fig. 3. Examples of predictions made by S-GP and ANOVA-GP, and the simulator outputs for both $N_D = 36$ and 64 subdomains, test problem 1.

For both ANOVA-GP and S-GP, PCA is conducted to result in a reduced dimensional representation of the outputs. Here, we show results of the case $N_D = 36$ with $N_{agp} = [50, 100]^T$ for ANOVA-GP and $N_{sgp} = 11800$ for S-GP, and the case $N_D = 64$ with $N_{agp} = [50, 100]^T$ and $N_{sgp} = 20400$. For S-GP, the number of PCA modes retained is 60 for the case $N_D = 36$, and that is 100 for $N_D = 64$. Fig. 4 shows the number of PCA modes retained for each ANOVA term in ANOVA-GP. It is clear that, the numbers are very small—there are at most two PCA modes retained for both cases ($N_D = 36$ and $N_D = 64$). In Fig. 4 the ANOVA indices are ordered alphabetically as: for any two different indices $t^{(j)}$ and $t^{(k)}$ belonging $\mathcal{J}$, $t^{(j)}$ is ordered before $t^{(k)}$ (i.e., $j < k$), if one of the following two cases is true: (a) $|t^{(j)}| < |t^{(k)}|$; (b) $|t^{(j)}| = |t^{(k)}|$ and for the smallest number $n \in \{1, \ldots, |t^{(j)}|\}$ such that $t_n^{(j)} \neq t_n^{(k)}$, we have $t_n^{(j)} < t_n^{(k)}$ (where $t_n^{(j)}$ and $t_n^{(k)}$ are the n-th components of $t^{(j)}$ and $t^{(k)}$). So, each local GP model (see line 4 of Algorithm 2) in ANOVA-GP only involves a very small number of independent GP models, so that training local GP models and using them to conduct predictions are both cheap.

Note that the anchored ANOVA decomposition in general loses the orthogonality between expansion terms when the Lebesgue measure is employed [46]. To assess how ANOVA-GP keeps the properties of the anchored ANOVA decomposition,



(a) $N_D = 36$ (b) $N_D = 64$

Fig. 4. Number of PCA modes retained, test problem 1.

we compute the errors in covariance (including variances) estimates between the effective ANOVA terms as follows. For any two indices $t, s \in \mathcal{J}$, the covariance between ANOVA terms u_t and u_s is computed through

$$\mathrm{Cov}\,(u_t, u_s) \approx \widehat{\mathrm{Cov}}\,(u_t, u_s) := \hat{\boldsymbol{E}}\left(\left(u_t - \hat{\boldsymbol{E}}(u_t)\right)\left(u_s - \hat{\boldsymbol{E}}(u_s)\right)\right), \tag{32}$$

where the quadrature rule for $\hat{\boldsymbol{E}}$ (see (16)) is set to the tensor products of one-dimensional Clenshaw-Curtis quadrature with nine quadrature points. The errors in covariance estimates are then defined as $\|\widehat{\mathrm{Cov}}(u_t, u_s) - \widehat{\mathrm{Cov}}(\overline{u}_t, \overline{u}_s)\|$, where u_t and u_s are generated through the simulator, and $\overline{u}_t$ and $\overline{u}_s$ are the predictive means of the local GP models. Fig. 5 shows Tukey box plots of the errors in the covariance estimates for all pairs of the ANOVA terms, where the central line in each box is the median, the lower and the upper edges are the first and the third quartiles respectively, the crosses are the outliers, and Setting 1, Setting 2 and Setting 3 refer to the cases $N_{agp} = [30, 50]^T$, $[40, 75]^T$ and $[50, 100]^T$ respectively. From Fig. 5, it is clear that the majority of the errors are small and decrease as the training data are augmented. Looking more closely, Fig. 6 shows the 200 largest Euclidean norms of the covariance estimates $\|\widehat{\mathrm{Cov}}(u_t, u_s)\|$ and the corresponding $\|\widehat{\mathrm{Cov}}(\overline{u}_t, \overline{u}_s)\|$. It can be seen that for these 200 pairs of ANOVA terms, as the training data are augmented, the Euclidean norms of the covariance estimates obtained through the simulator and the local GP models become closer.

In addition, following [46,42], we compute the relative standard deviation for this test problem through $\kappa := \|(\sum_{t\in\mathcal{J}}\sum_{s\in\mathcal{J}}\widehat{\mathrm{Cov}}(u_t, u_s))^{1/2}\|/\|\sum_{t\in\mathcal{J}}\hat{\boldsymbol{E}}(u_t)\|$. The values of κ are 0.3053 and 0.2044 for the cases $N_D = 36$ and $N_D = 64$ respectively. The relative standard deviation for each ANOVA term is computed as $\kappa_t := \|(\widehat{\mathrm{Cov}}(u_t, u_t))^{1/2}\|/\|\hat{\boldsymbol{E}}(u_t)\|$. The values of κ_t (for $t \in \mathcal{J}$) ranges from 4 to 11 for both cases ($N_D = 36$ and $N_D = 64$). As the norm of the mean for each ANOVA term is smaller than that for the whole ANOVA expansion for this test problem, it is not surprising that the values of κ_t are larger than the values of κ, while neither κ nor κ_t has small values. To this end, it can be seen that a small number of training data points (e.g., $N_{agp} = [50, 100]^T$) can result in efficient local GP models and the overall ANOVA-GP emulator. As discussed in [47,19], the input-space correlations of standard GP emulators are defined based on the Euclidean distance. So, to achieve an accurate approximation, the number of required training data points dramatically increases as the dimensionality of the input space increases. This is consistent with our numerical results above—ANOVA-GP has smaller errors than S-GP for the same number of training data points. Unlike S-GP which takes the high-dimensional inputs as a whole, the ANOVA terms in ANOVA-GP only have low-dimensional inputs, which are at most two-dimensional for this test problem, and therefore a small number of training points can lead to accurate local GP models.

5.2. Test problem 2: the Stokes problems

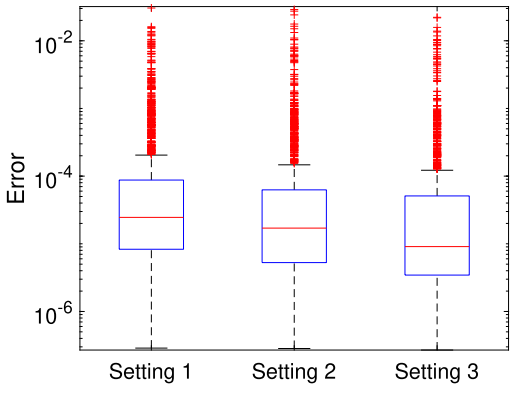
The Stokes equations for this test problem are

$$\nabla \cdot [a\,(x, \xi)\,\nabla u_{sol}\,(x, \xi)] + \nabla p_{sol}\,(x, \xi) = 0 \quad \text{in} \quad D \times I^m, \tag{33}$$

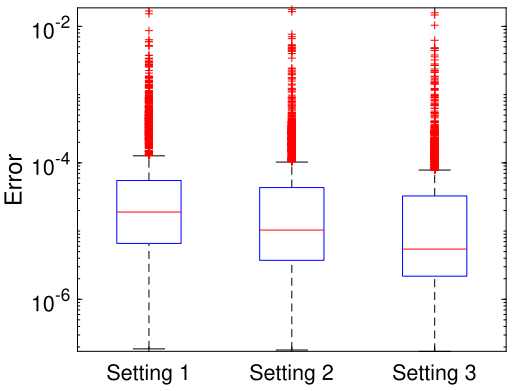
$$\nabla \cdot u_{sol}\,(x, \xi) = 0 \quad \text{in} \quad D \times I^m, \tag{34}$$

$$u_{sol}\,(x, \xi) = g \quad \text{on} \quad \partial D \times I^m, \tag{35}$$

where $D \in \mathbb{R}^2$, and $u_{sol}(x, \xi) = [u_{sol,1}(x, \xi), u_{sol,2}(x, \xi)]^T$ and $p_{sol}(x, \xi)$ are the flow velocity and the scalar pressure respectively. In (33), we focus on the situation that there exists uncertainties in the flow viscosity $a(x, \xi)$, which is assumed to be a random field with mean function $a_0(x) = 1$, standard deviation $\sigma = 0.5$ and covariance function $Cov(x, x')$

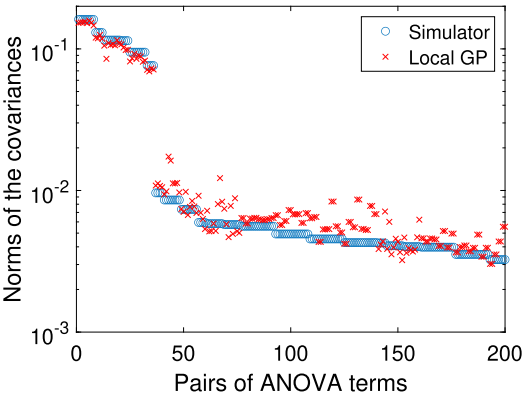


(a) $N_D = 36$

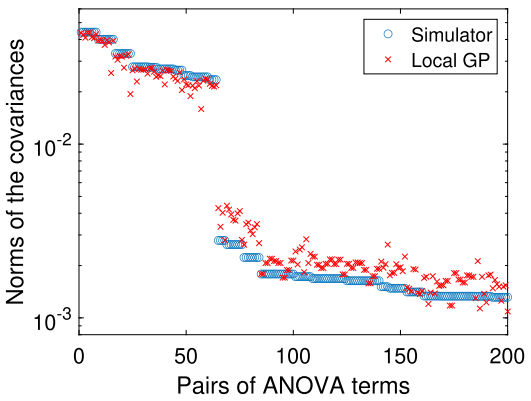


(b) $N_D = 64$

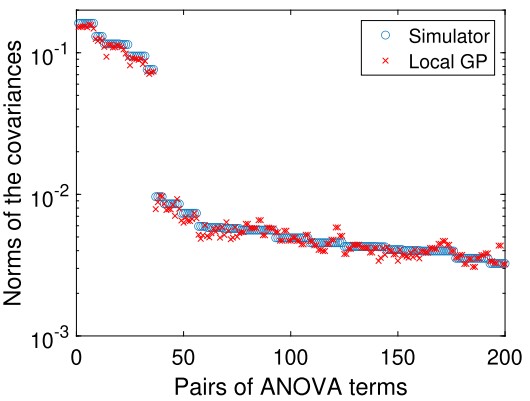
Fig. 5. Errors in covariance estimates, test problem 1.



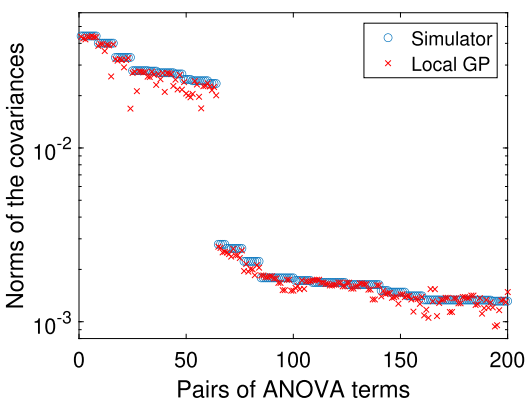
(a) $N_D = 36$, $N_{agp} = [30, 50]^T$ (Setting 1)



(c) $N_D = 64$, $N_{agp} = [30, 50]^T$ (Setting 1)



(b) $N_D = 36$, $N_{agp} = [50, 100]^T$ (Setting 3)



(d) $N_D = 64$, $N_{agp} = [50, 100]^T$ (Setting 3)

Fig. 6. The 200 largest Euclidean norms of the covariance estimates (Simulator refers to $\|\widehat{\mathrm{Cov}}(u_t, u_s)\|$, and Local GP refers to $\|\widehat{\mathrm{Cov}}(\overline{u}_t, \overline{u}_s)\|$), test problem 1.

$$Cov\left(x, x'\right) = \sigma^2 \exp\left(-\frac{|x_1 - x_1'|}{l_c} - \frac{|x_2 - x_2'|}{l_c}\right). \tag{36}$$

In (36), $x = [x_1, x_2]^T, x' = [x_1', x_2']^T \in D$, and the correlation length is set to $l_c = 0.5$. To approximate the random field $a(x, \xi)$, the truncated KL expansion can be applied [48,7,49]

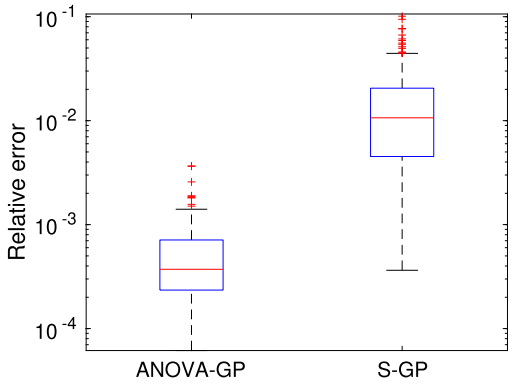


Fig. 7. Relative errors of ANOVA-GP and S-GP for 200 test data points, test problem 2.

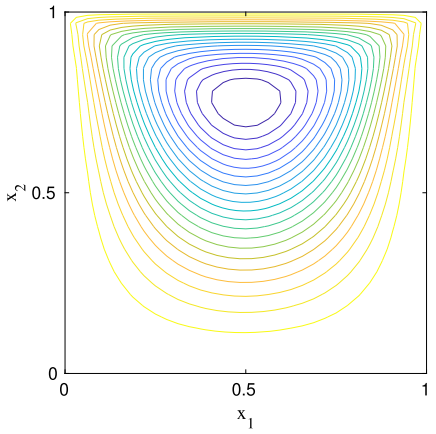
$$a(x,\xi) \approx a_0(x) + \sum_{i=1}^{m} \sqrt{\zeta_i} a_i(x)\,\xi_i,$$

where $\zeta_1 \geq, \ldots, \geq \zeta_m$ and $a_1(x), \ldots, a_m(x)$ are the eigenvalues and eigenfunctions of the covariance function (36), m is the number of KL modes retained, and $\xi = [\xi_1, \ldots, \xi_m]^T$ are uncorrelated random variables. In this test problem, $\xi_1, \ldots, \xi_m$ are assumed to be independent uniform distributions in $[-1, 1]$. We consider the driven cavity flow problem posed on the physical domain $D = (0,1) \times (0,1)$. The velocity profile $u = [1, 0]^T$ is imposed on the top boundary $\{[x_1, x_2]^T \mid x_1 \in (0,1), x_2 = 1\}$, and the no-slip and no-penetration condition $u = [0, 0]^T$ is applied on all other boundaries. The error of the truncated KL expansion depends on the amount of total variance captured, and we set $m = 109$ to capture 95% of the total variance, i.e., $\sum_{j=1}^{m} \zeta_j / (|D|\sigma^2) > 0.95$, where $|D|$ refers to the area of D [7,50].

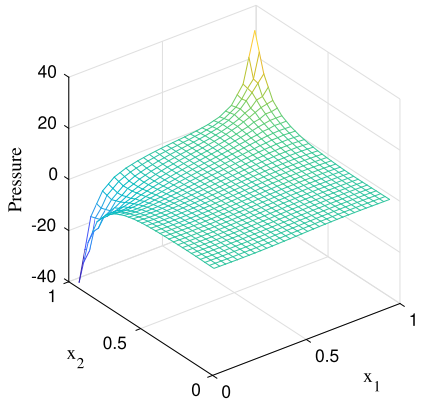
The simulator for this test problem is set to the $\boldsymbol{Q}_2 - \boldsymbol{P}_{-1}$ mixed finite element method (biquadratic velocity–linear discontinuous pressure) implemented in IFISS [51,2], with the physical domain discretized on a uniform 33×33 grid, which gives the velocity degrees of freedom $N_u = 2178$ and the pressure degrees of freedom $N_p = 768$. For each realization of ξ, the simulator output y is defined to be the vector collecting the coefficients of the $\boldsymbol{Q}_2 - \boldsymbol{P}_{-1}$ approximation solution for (33)–(35), and the dimension of the simulator output is 2946.

For this test problem, since the simulator output for the Stokes problem involves velocity and pressure approximations, the relative mean (14) is defined to be the sum of the functional L_2 norms of the approximation functions associated with them, i.e., $\gamma_t := (\|\hat{\boldsymbol{E}}(u_t)\|_{L_2} + \|\hat{\boldsymbol{E}}(p_t)\|_{L_2}) / (\|\sum_{s \in \mathcal{J}_0 \cup \cdots \cup \mathcal{J}_{|t|-1}} \hat{\boldsymbol{E}}(u_s)\|_{L_2} + \|\sum_{s \in \mathcal{J}_0 \cup \cdots \cup \mathcal{J}_{|t|-1}} \hat{\boldsymbol{E}}(p_s)\|_{L_2})$, where u_t and p_t denote ANOVA terms for velocity and pressure respectively (see (10)). The tolerance for selecting ANOVA terms (see section 3.2) is set to $tol_{index} = 10^{-5}$, and the quadrature rule is set to the tensor products of one-dimensional Clenshaw-Curtis quadrature with five quadrature points, while the tolerance for PCA is set to $tol_{pca} = 10^{-3}$ in Algorithm 1. In this setting, the index set $\mathcal{J}$ constructed through the adaptive ANOVA decomposition procedure only contains the zeroth order index and 32 first order indices, i.e., $|\mathcal{J}_1| = 32$ and $|\mathcal{J}| = 33$. The number of training points for ANOVA-GP is set to $N_{agp} = N_{agp,1} = 50$ (as only first order ANOVA terms exist) and that for S-GP is set to $N_{sgp} := (|\mathcal{J}| - 1) \times N_{agp,1} = 1600$ (as the input of Algorithm 5) for a fair comparison. Again, 200 samples of ξ are generated and the corresponding simulator outputs are computed. The errors of ANOVA-GP and S-GP are assessed through the relative error defined in (31). Fig. 7 shows Tukey box plots of the errors for both ANOVA-GP and S-GP, where the central line in each box is the median, the lower and the upper edges are the first and the third quartiles respectively, and the crosses are the outliers where the relative errors are large. It is clear that the errors of ANOVA-GP are one order of magnitude smaller than the errors of S-GP. In addition, the number of principal components retained for S-GP is 30 and that for each ANOVA term in ANOVA-GP is one, which indicates that the ANOVA terms (10) have very small ranks.

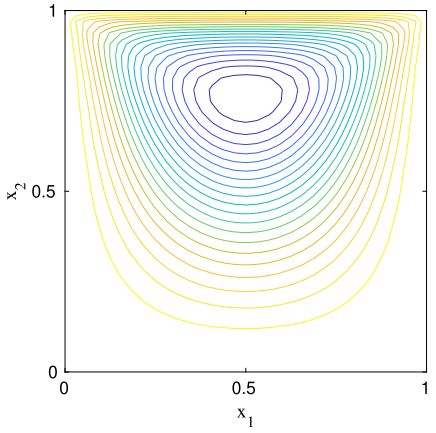
Fig. 8 shows the simulator output and the ANOVA-GP and the S-GP predictive means corresponding to a given realization of ξ. From Fig. 8(a), Fig. 8(b) and Fig. 8(c), it can be seen that the velocity streamlines obtained from the simulator output and those from ANOVA-GP and S-GP emulators are visually indistinguishable. However, from 8(d), Fig. 8(e) and Fig. 8(f), the pressure obtained from S-GP is clearly larger than that of the simulator near the upper right corner (1,1), while the pressure fields obtained from ANOVA-GP and the simulator are visually indistinguishable. To look more closely, we compute the errors of the emulator predictive means as follows. For a physical grid point, let $u = [u_1, u_2]^T$ and p denote the velocity and the pressure obtained through the simulator at this grid point, and $\overline{u} = [\overline{u}_1, \overline{u}_2]$ and $\overline{p}$ denote the velocity and the pressure obtained through the emulators (predictive means of ANOVA-GP and S-GP). The errors of velocity and pressure at this grid point are defined as $error_u := \sqrt{(u_1 - \overline{u}_1)^2 + (u_2 - \overline{u}_2)^2}$ and $error_p = |p - \overline{p}|$ respectively. Fig. 9 shows these errors. From Fig. 9(a) and Fig. 9(b), it can be seen that the maximum velocity error of ANOVA-GP is less than half of the maximum error of S-GP. From Fig. 9(c) and Fig. 9(d), the maximum pressure error of ANOVA-GP is less than ten percent of the maximum error of S-GP.



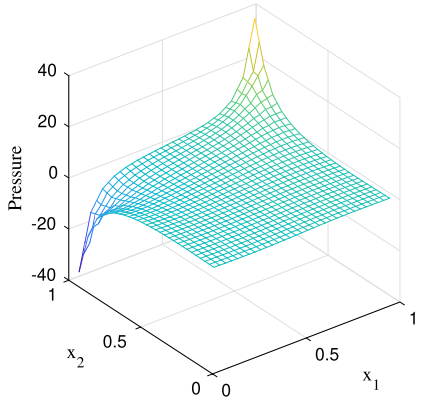
(a) Streamline, simulator



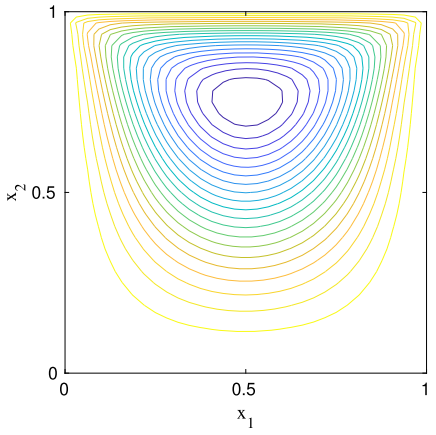
(d) Pressure, simulator



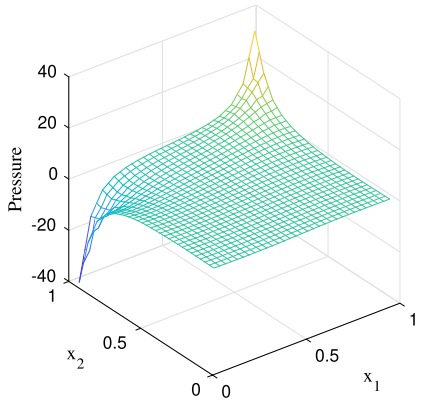
(b) Streamline, S-GP



(e) Pressure, S-GP



(c) Streamline, ANOVA-GP

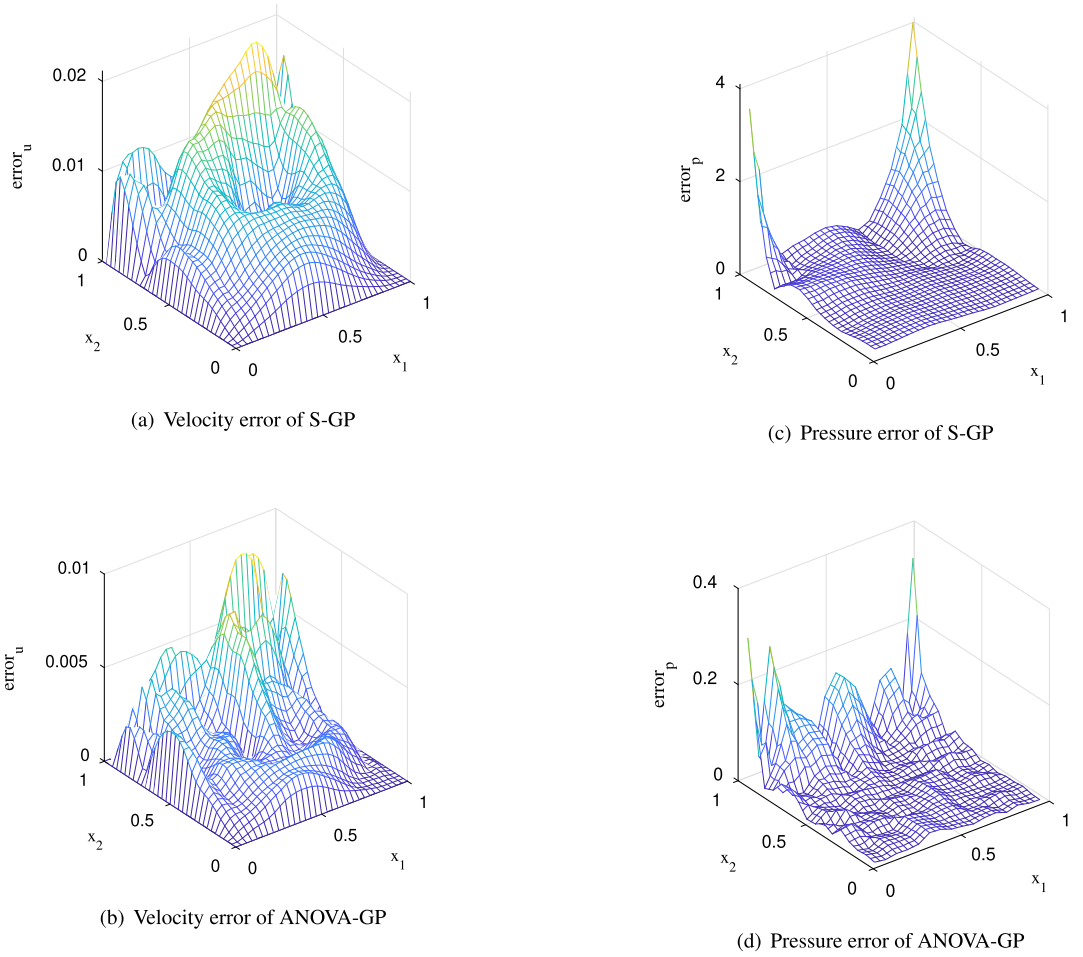


(f) Pressure, ANOVA-GP

Fig. 8. Examples of predictions made by S-GP and ANOVA-GP, and the simulator outputs, test problem 2.

6. Concluding remarks

Conducting dimension reduction is one of the fundamental concepts to develop efficient GP emulators for complex computational models with high-dimensional inputs and outputs. With a focus on adaptive ANOVA decomposition, this



(a) Velocity error of S-GP

(c) Pressure error of S-GP

(b) Velocity error of ANOVA-GP

(d) Pressure error of ANOVA-GP

Fig. 9. Errors of S-GP and ANOVA-GP predictions, test problem 2.

paper proposes a novel ANOVA-GP strategy. In ANOVA-GP, the high-dimensional inputs are decomposed into a combination of low-dimensional local inputs through adaptive ANOVA decomposition, and PCA is applied on each ANOVA term (12) to result in a reduced dimensional representation of the outputs. Local GP models are built through active training with initial data obtained in the ANOVA decomposition procedure. Since each local input is low-dimensional and the resulting term in the ANOVA expansion has a small rank, GP emulation for each ANOVA term becomes less challenging compared with that for the overall problem (1)–(2). From numerical studies, it can be seen that a very small number of data points are required to build local GP models for each ANOVA term. It is also clear that for a given number of training data points, prediction errors of ANOVA-GP are smaller than the errors of the standard GP method. In addition, the cost of ANOVA-GP for conducting predictions is cheaper than that of standard GP. From Algorithm 3, it can be seen that the total number of training data points to generate the ANOVA-GP model is $\sum_{i=1}^{h} |\mathcal{J}_i| N_{agp,i}$. The cost of using GP models to make a single prediction is dominated by the cost of computing the inverse of the covariance matrix (see (20))—the main cost of ANOVA-GP is then $O(\sum_{i=1}^{h} |\mathcal{J}_i| N_{agp,i}^3)$, and the main cost of the standard GP method with $\sum_{i=1}^{h} |\mathcal{J}_i| N_{agp,i}$ training data points is $O((\sum_{i=1}^{h} |\mathcal{J}_i| N_{agp,i})^3)$ (which is larger than that of ANOVA-GP). As in our ANOVA-GP setting, PCA is applied to conduct dimension reduction for the output space, and the number of training data points for local GP models with ANOVA order i are all set to the same number $N_{agp,i}$ (see Algorithm 3), which may not be optimal when the underlying problem has highly nonlinear structures. A possible solution is to apply nonlinear model reduction methods and adaptive training procedures to result in different numbers of training data points for each ANOVA term, which will be the focus of our future work.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgements

This work is supported by the National Natural Science Foundation of China (No. 11601329).

Appendix A. Algorithms for the adaptive ANOVA decomposition and the standard Gaussian process modeling

Algorithm 4 The adaptive anchored ANOVA decomposition.

Input: A simulator for (1)–(2) and the probability density function of ξ.

Initialize: $\mathcal{J} = \{\emptyset\}$, $\widehat{\mathcal{J}}_1 = \{1, 2 \ldots, M\}$ and $i = 1$.
Compute $u(c)$ through the given simulator, where c is the anchor point.
while $\widehat{\mathcal{J}}_i \neq \emptyset$ **do**
  Set $\mathcal{J}_i = \emptyset$.
  for $t \in \widehat{\mathcal{J}}_i$ **do**
    Setup the Clenshaw-Curtis tensor quadrature points $\Xi_t = \left\{\xi_t^{(1)}, \ldots, \xi_t^{(|\Xi_t|)}\right\}$ and weights $\left\{w\left(\xi_t^{(k)}\right)\right\}$ for $k = 1, \ldots, |\Xi_t|$.
    Compute $\gamma_t := \dfrac{\left\|\hat{\boldsymbol{E}}(u_t(\xi_t))\right\|_{L_2}}{\left\|\sum_{s\in\mathcal{J}} \hat{\boldsymbol{E}}(u_s(\xi_s))\right\|_{L_2}}$, where $\hat{\boldsymbol{E}}$ is defined in (16) and $u_t(\xi_t)$ is computed using (10) with the simulator.
    if $\gamma_t > tol_{index}$ **then**
      Update $\mathcal{J}_i = \mathcal{J}_i \cup \{t\}$.
    end if
  end for
  Update $\mathcal{J} = \mathcal{J} \cup \mathcal{J}_i$.
  Construct $\widehat{\mathcal{J}}_{i+1} = \left\{t \mid |t| = i+1, \text{and any } s \subset t \text{ with } |s| = i \text{ satisfies } s \in \mathcal{J}_i\right\}$.
  Update the ANOVA order: $i = i + 1$.
end while

Output: An effective index set $\mathcal{J}$ and data sets $\Theta_t := \left\{y_t^{(k)} = u_t\left(\xi_t^{(k)}\right) \,\middle|\, \xi_t^{(k)} \in \Xi_t, \text{ and } k = 1, \ldots, |\Xi_t|\right\}$ for $t \in \mathcal{J}$.

Algorithm 5 Standard Gaussian process (S-GP) modeling.

Input: A simulator for (1)–(2), the probability density function of ξ and the number of training points N_{sgp}.

Generate N_{sgp} samples of ξ: $\left\{\xi^{(j)}\right\}_{j=1}^{N_{sgp}}$.
Compute the simulator outputs: $\left\{y^{(j)} = u\left(\xi^{(j)}\right)\right\}_{j=1}^{N_{sgp}}$.
Conduct PCA using Algorithm 1 with the input data set $\left\{y^{(j)} = u\left(\xi^{(j)}\right)\right\}_{j=1}^{N_{sgp}}$, to obtain the sample mean μ, the principal component matrix $V := [v_1, \ldots, v_R]$, and the data vectors α_r for $r = 1, \ldots, R$.
Construct GP models for PCA modes: $U = [U_1, \ldots, U_R]^T \in \mathbb{R}^R$ with $U_r := \mathcal{GP}\left(m_r'(\xi, \beta), v_r'(\xi, \beta)\right)$ for $r = 1, \ldots, R$ (see (19)–(20)).
Construct the S-GP emulator: $u_{SGP}(\xi) = VU(\xi) + \mu$.

Output: The S-GP emulator $u_{SGP}(\xi)$.

References

[1] M. Ainsworth, J. Oden, A Posteriori Error Estimation in Finite Element Analysis, Wiley, New York, 2000.
[2] H. Elman, D. Silvester, A. Wathen, Finite Elements and Fast Iterative Solvers: with Applications in Incompressible Fluid Dynamics, Oxford University Press (UK), 2014.
[3] M.C. Kennedy, A. O'Hagan, Bayesian calibration of computer models, J. R. Stat. Soc., Ser. B, Stat. Methodol. 63 (3) (2001) 425–464.
[4] J. Oakley, A. O'Hagan, Bayesian inference for the uncertainty distribution of computer model outputs, Biometrika 89 (2002) 769–784.
[5] M. Kennedy, C. Anderson, S. Conti, A. O'Hagan, Case studies in Gaussian process modelling of computer codes, Reliab. Eng. Syst. Saf. 91 (2006) 1301–1309.
[6] P.M. Tagade, B.-M. Jeong, H.-L. Choi, A Gaussian process emulator approach for rapid contaminant characterization with an integrated multizone-CFD model, Build. Environ. 70 (2013) 232–244.
[7] R.G. Ghanem, P.D. Spanos, Stochastic Finite Elements: A Spectral Approach, Courier Corporation, 2003.
[8] D. Xiu, G.E. Karniadakis, The Wiener–Askey polynomial chaos for stochastic differential equations, SIAM J. Sci. Comput. 24 (2) (2002) 619–644.
[9] D. Xiu, J. Hesthaven, High-order collocation methods for differential equations with random inputs, SIAM J. Sci. Comput. 27 (2005) 1118–1139.
[10] D. Xiu, Numerical Methods for Stochastic Computations: A Spectral Method Approach, Princeton University Press, Princeton, 2010.
[11] S. Boyaval, C.L. Bris, T. Lelièvre, Y. Maday, N. Nguyen, A. Patera, Reduced basis techniques for stochastic problems, Arch. Comput. Methods Eng. 17 (2010) 1–20.
[12] H. Elman, Q. Liao, Reduced basis collocation methods for partial differential equations with random coefficients, SIAM/ASA J. Uncertain. Quantificat. 1 (2013) 192–217.
[13] P. Chen, A. Quarteroni, G. Rozza, Comparison between reduced basis and stochastic collocation methods for elliptic problems, J. Sci. Comput. 59 (2014) 187–216.
[14] J. Jiang, Y. Chen, A. Narayan, A goal-oriented reduced basis methods-accelerated generalized polynomial chaos algorithm, SIAM/ASA J. Uncertain. Quantificat. 4 (2016) 1398–1420.
[15] D. Higdon, J. Gattiker, B. Williams, M. Rightley, Computer model calibration using high-dimensional output, J. Am. Stat. Assoc. 103 (482) (2008) 570–583.
[16] X. Ma, N. Zabaras, Kernel principal component analysis for stochastic input model generation, J. Comput. Phys. 230 (2011) 7311–7331.
[17] W. Xing, V. Triantafyllidis, A. Shah, P. Nair, N. Zabaras, Manifold learning for the emulation of spatial fields from computational models, J. Comput. Phys. 326 (2016) 666–690.

[18] M. Guo, J.S. Hesthaven, Reduced order modeling for nonlinear structural analysis using Gaussian process regression, Comput. Methods Appl. Mech. Eng. 341 (2018) 807–826.
[19] R. Tripathy, I. Bilionis, M. Gonzalez, Gaussian processes with built-in dimensionality reduction: applications to high-dimensional uncertainty propagation, J. Comput. Phys. 321 (2016) 191–223.
[20] C.B. Storlie, W.A. Lane, E.M. Ryan, J.R. Gattiker, D.M. Higdon, Calibration of computational models with categorical parameters and correlated outputs via Bayesian smoothing spline ANOVA, J. Am. Stat. Assoc. 110 (509) (2015) 68–82.
[21] H. Rabitz, Ö.F. Aliş, J. Shorter, K. Shim, Efficient input–output model representations, Comput. Phys. Commun. 117 (1–2) (1999) 11–20.
[22] H. Rabitz, Ö.F. Aliş, General foundations of high-dimensional model representations, J. Math. Chem. 25 (2–3) (1999) 197–233.
[23] Z. Gao, J.S. Hesthaven, On ANOVA expansions and strategies for choosing the anchor point, Appl. Math. Comput. 217 (7) (2010) 3274–3285.
[24] X. Ma, N. Zabaras, An adaptive high-dimensional stochastic model representation technique for the solution of stochastic partial differential equations, J. Comput. Phys. 229 (10) (2010) 3884–3915.
[25] Z. Zhang, M. Choi, G. Karniadakis, Error estimates for the ANOVA method with polynomial chaos interpolation: tensor product functions, SIAM J. Sci. Comput. 34 (2) (2012) A1165–A1186.
[26] X. Yang, M. Choi, G. Lin, G.E. Karniadakis, Adaptive ANOVA decomposition of stochastic incompressible and compressible flows, J. Comput. Phys. 231 (4) (2012) 1587–1614.
[27] J.S. Hesthaven, S. Zhang, On the use of ANOVA expansions in reduced basis methods for parametric partial differential equations, J. Sci. Comput. 69 (1) (2016) 292–313.
[28] I.M. Sobol, Theorems and examples on high dimensional model representation, Reliab. Eng. Syst. Saf. 79 (2) (2003) 187–193.
[29] X. Wang, On the approximation error in high dimensional model representation, in: Simulation Conference, 2008. WSC 2008, Winter, IEEE, 2008, pp. 453–462.
[30] H. Xu, S. Rahman, A generalized dimension-reduction method for multidimensional integration in stochastic mechanics, Int. J. Numer. Methods Eng. 61 (12) (2004) 1992–2019.
[31] E. Novak, K. Ritter, High dimensional integration of smooth functions over cubes, Numer. Math. 75 (1) (1996) 79–97.
[32] L.N. Trefethen, Is Gauss quadrature better than clenshaw–curtis?, SIAM Rev. 50 (1) (2008) 67–87.
[33] I. Jolliffe, Principal component analysis, in: International Encyclopedia of Statistical Science, Springer, 2011, pp. 1094–1096.
[34] R. Vidal, Y. Ma, S.S. Sastry, Generalized Principal Component Analysis, vol. 5, Springer, 2016.
[35] P. Holmes, J.L. Lumley, G. Berkooz, Turbulence, Coherent Structures, Dynamical Systems and Symmetry, 1996, Cambridge, New York.
[36] P. Benner, S. Gugercin, K. Willcox, A survey of model reduction methods for parametric systems, SIAM Rev. 57 (2015) 483–531.
[37] C.E. Rasmussen, Gaussian processes in machine learning, in: Advanced Lectures on Machine Learning, Springer, 2004, pp. 63–71.
[38] I. Andrianakis, P.G. Challenor, The effect of the nugget on Gaussian process emulators of computer models, Comput. Stat. Data Anal. 56 (12) (2012) 4215–4228.
[39] E.L. Snelson, Flexible and efficient Gaussian process models for machine learning, PhD thesis, UCL (University College London), 2007.
[40] C.E. Rasmussen, H. Nickisch, Gaussian processes for machine learning (gpml) toolbox, J. Mach. Learn. Res. 11 (Nov. 2010) 3011–3015.
[41] M.F. Møller, A scaled conjugate gradient algorithm for fast supervised learning, Neural Netw. 6 (4) (1993) 525–533.
[42] Q. Liao, G. Lin, Reduced basis ANOVA methods for partial differential equations with high-dimensional random inputs, J. Comput. Phys. 317 (2016) 148–164.
[43] H. Cho, H.C. Elman, An adaptive reduced basis collocation method based on PCM ANOVA decomposition for anisotropic stochastic PDEs, Int. J. Uncertain. Quantificat. 8 (2018) 193–210.
[44] D. Braess, Finite Elements: Theory, Fast Solvers, and Applications in Solid Mechanics, Cambridge University Press, 2007.
[45] A. Klimke, Sparse Grid Interpolation Toolbox – user's guide, Tech. Rep. IANS report 2007/017, University of Stuttgart, 2007.
[46] K. Tang, P.M. Congedo, R. Abgrall, Sensitivity analysis using anchored ANOVA expansion and high-order moments computation, Int. J. Numer. Methods Eng. 102 (2015) 1554–1584.
[47] Y. Bengio, O. Delalleau, N.L. Roux, The curse of highly variable functions for local kernel machines, in: Y. Weiss, B. Schölkopf, J.C. Platt (Eds.), Advances in Neural Information Processing Systems, vol. 18, MIT Press, 2006, pp. 107–114.
[48] J. Brown Jr, Mean square truncation error in series expansions of random functions, J. Soc. Ind. Appl. Math. 8 (1) (1960) 28–32.
[49] C. Schwab, R.A. Todor, Karhunen–Loève approximation of random fields by generalized fast multipole methods, J. Comput. Phys. 217 (1) (2006) 100–122.
[50] C. Powell, H. Elman, Block-diagonal preconditioning for spectral stochastic finite-element systems, IMA J. Numer. Anal. 29 (2009) 350–375.
[51] H.C. Elman, A. Ramage, D.J. Silvester, IFISS: A computational laboratory for investigating incompressible flow problems, SIAM Rev. 56 (2) (2014) 261–273.