MULTISCALE MODEL. SIMUL.
Vol. 13, No. 1, pp. 173–204

A MULTISCALE DATA-DRIVEN STOCHASTIC METHOD FOR ELLIPTIC PDEs WITH RANDOM COEFFICIENTS[*]

ZHIWEN ZHANG[†], MAOLIN CI[†], AND THOMAS Y. HOU[‡]

Abstract. In this paper, we propose a multiscale data-driven stochastic method (MsDSM) to study stochastic partial differential equations (SPDEs) in the multiquery setting. This method combines the advantages of the recently developed multiscale model reduction method [M. L. Ci, T. Y. Hou, and Z. Shi, *ESAIM Math. Model. Numer. Anal.,* 48 (2014), pp. 449–474] and the data-driven stochastic method (DSM) [M. L. Cheng et al., *SIAM/ASA J. Uncertain. Quantif.*, 1 (2013), pp. 452–493]. Our method consists of offline and online stages. In the offline stage, we decompose the harmonic coordinate into a smooth part and a highly oscillatory part so that the smooth part is invertible and the highly oscillatory part is small. Based on the Karhunen–Loève (KL) expansion of the smooth parts and oscillatory parts of the harmonic coordinates, we can derive an effective stochastic equation that can be well-resolved on a coarse grid. We then apply the DSM to the effective stochastic equation to construct a data-driven stochastic basis under which the stochastic solutions enjoy a compact representation for a broad range of forcing functions. In the online stage, we expand the SPDE solution using the data-driven stochastic basis and solve a small number of coupled deterministic partial differential equations (PDEs) to obtain the expansion coefficients. The MsDSM reduces both the stochastic and the physical dimensions of the solution. We have performed complexity analysis which shows that the MsDSM offers considerable savings over not only traditional methods but also DSM in solving multiscale SPDEs. Numerical results are presented to demonstrate the accuracy and efficiency of the proposed method for several multiscale stochastic problems without scale separation.

Key words. stochastic partial differential equations, multiscale problems, data-driven methods, Karhunen–Loève expansion, uncertainty quantification, model reduction

AMS subject classifications. 35R60, 60H15, 60H35, 65C30

DOI. 10.1137/130948136

1. Introduction.

In recent years, there has been an increased interest in the simulation of systems with uncertainties. Many physical and engineering applications involving uncertainty quantification can be described by stochastic partial differential equations (SPDEs). Several numerical methods have been developed in the literature to solve SPDEs, such as the stochastic finite element method [24], Wiener chaos expansion or generalized polynomial chaos (gPC) method [31, 25, 40, 41, 42, 39, 33, 32], and stochastic collocation method [43, 5, 30, 6]. These methods can effectively solve the SPDEs when the dimension of stochastic input variables is low. However, their performance deteriorates dramatically when the dimension of stochastic input variables is high. This so-called curse of dimensionality is one of the essential challenges in uncertainty quantification. Recently, some progress has been made to alleviate this difficulty by exploring the sparse structure of the solutions and constructing a problem-dependent stochastic basis to solve these SPDEs; see, e.g., the data-driven stochastic method [11, 44] and dynamically biorthogonal method [9, 10].

[*]Received by the editors December 9, 2013; accepted for publication (in revised form) October 28, 2014; published electronically January 20, 2015. This work was supported in part by AFOSR MURI grant FA9550-09-1-0613, DOE grant DE-FG02-06ER25727, and NSF grants DMS-1159138 and DMS-1318377.

http://www.siam.org/journals/mms/13-1/94813.html

[†]Computing and Mathematical Sciences, California Institute of Technology, Pasadena, CA 91125 (zhangzw@caltech.edu, cimaolin@gmail.com).

[‡]Corresponding author. Computing and Mathematical Sciences, California Institute of Technology, Pasadena, CA 91125 (hou@cms.caltech.edu).

In this paper, we consider another challenge in uncertainty quantification, i.e., solving SPDEs involving multiple scales. Due to the large range of scales in these solutions, it is extremely challenging to solve for the small scales of the solution. It requires tremendous computational resources. Thus, finding an effective (upscaled) equation that governs the large-scale solution is very important. When the solution has scale separation and a periodic structure, the classical homogenization theory provides a powerful tool for deriving an effective equation. However, in many applications, the solutions usually do not satisfy the scale separation assumption or may not have periodic structures. In this case, it is very difficult to derive an effective equation. In the past three decades, there have been a number of multiscale methods for deterministic PDEs in the literature; see [2, 3, 4, 17, 8, 14, 15, 16, 18, 19, 20, 27, 28, 35]. The SPDEs involving multiple scales become more complicated because we cannot directly apply these well-developed upscaling techniques for each realization of the stochastic parameters. Recently, Zabaras and coworkers proposed a stochastic variational multiscale method for diffusion in heterogeneous random media [38, 23]. They combined the gPC method with the variational multiscale method to do model reduction. However, when the dimension in stochastic direction is large, this method is inefficient due to the exponential growth of the number of the gPC basis elements. We also point out that in [1], Arnst and Ghanem considered the probabilistic equivalence and stochastic model reduction in multiscale analysis. In [27, 37], Kevrekidis et al. applied the equation-free idea to study stochastic incompressible flows.

To address this issue, we propose a multiscale data-driven stochastic method (MsDSM) to systematically perform model reduction in both the stochastic and physical dimensions. Our new method consists of offline and online stages. In the offline stage, we perform model reduction in both the stochastic and physical dimensions, respectively. We choose the stochastic collocation method to approximate the stochastic space since it can exploit the possible regularity of the solution with respect to the stochastic parameters to achieve faster convergence, and it leads to the solution of uncoupled deterministic problems, just as in the Monte Carlo method. We utilize the *Clenshaw–Curtis* rule to generate the sparse grids [5, 43]. On each collocation point, we solve a deterministic problem. We first derive an effective (upscaled) equation that can be resolved on a coarse grid. We then construct a data-driven stochastic basis in stochastic collocation representation [11] under which the solutions of the effective stochastic equation have a compact representation for a broad range of forcing functions and/or boundary conditions.

We use the following stochastic elliptic equations with multiscale random coefficients as an example to illustrate the main idea of our approach:

$$-\nabla \cdot (a^{\varepsilon}(x,\omega)\nabla u^{\varepsilon}(x,\omega)) = f(x,\theta), \quad x \in D, \omega \in \Omega, \theta \in \Theta, \tag{1.1}$$

$$u^{\varepsilon}(x,\omega) = 0, \quad x \in \partial D, \tag{1.2}$$

where $D \in R^d$ is a bounded spatial domain, Ω is a sample space, and Θ is a parameter set, which is used in the multiquery setting. The multiscale information is described by the multiscale coefficient matrix $a^{\varepsilon}(x,\omega)$. We assume that $a^{\varepsilon}(x,\omega)$ is a symmetric, positive definite matrix satisfying $\lambda_{min} \geq \alpha > 0$ (λ_{min} is the smallest eigenvalue of $a^{\varepsilon}(x,\omega)$) for a.e. $x \in D$, $\omega \in \Omega$. For such coefficients, the solutions are only Hölder continuous. If $a^{\varepsilon}(x,\omega)$ is highly oscillatory, the solution will become highly oscillatory as well. The deterministic forcing function $f(x,\theta) \in L^2(D)$ is parameterized by θ, which is assumed to be resolved on a coarse grid. We would like to derive an effective stochastic equation of the form

$$-\nabla \cdot (a^*(x,\omega)\nabla u^*(x,\omega)) = f(x,\theta), \quad x \in D, \omega \in \Omega_s, \theta \in \Theta, \tag{1.3}$$

$$u^*(x,\omega) = 0, \quad x \in \partial D, \tag{1.4}$$

where Ω_s is the approximated sample space including all the stochastic collocation points. The key is how to construct an effective coefficient $a^*(x,\omega)$ so that the solution of the effective (1.1) approximates the original multiscale (1.1) with some desirable accuracy. We adopt the global upscaling technique proposed in [12] to construct $a^*(x,\omega)$. We proceed as follows.

We generate collocation points or samples according to the distribution information of the random parameters. On each collocation point, or sample $\omega_l \in \Omega_s$, the multiscale problem (1.1)–(1.2) becomes deterministic. We first solve the corresponding homogeneous problem to obtain the harmonic coordinates F. Then, we decompose the harmonic coordinates F into a smooth part g plus a highly oscillatory part χ so that g is invertible and χ is small. Although F does not need to be invertible, to motivate the method, we assume temporarily that F is invertible and express u as a function of F. One important property of the harmonic coordinates is that u as a function of F is about one order smoother than u as a function of x (see [35]). Thus, we can write the solution of (1.1) as $u^\varepsilon(F) = u^\varepsilon(g+\chi)$ and formally expand u^ε around g. By substituting the leading order expansion into the original equation (1.1), we obtain an effective equation of form (1.3) after ignoring the higher order terms involving χ. The effective coefficient $a^*(x,\omega_l)$ in (1.3) is defined in terms of $a^\varepsilon(x,\omega_l)$, g, and χ, i.e.,

$$a^*(x,\omega_l) = a^\varepsilon(x,\omega_l)\left(I + \frac{\partial \chi}{\partial x}(x,\omega_l)\frac{\partial x}{\partial g}(x,\omega_l)\right), \quad \omega_l \in \Omega_s, \tag{1.5}$$

where I is an identity matrix. Under some conditions, one can show that the solution to the effective (1.3) is in H^2, which is one order smoother than the original multiscale solution. Thus, we can solve the effective equation on a coarse mesh. Note that $a^*(x,\omega_l)$ is still multiscale, but the nonsmooth part of a^* is divergence free, i.e., $\nabla \cdot (a\frac{\partial F}{\partial x}) = 0$, which will be shown later in this paper. This property will help us to obtain an upscaled solution u_0. This is how we perform model reduction in the physical dimension.

To perform model reduction in the stochastic dimension, we adopt the Karhunen–Loève (KL) expansion [26, 29] for the stochastic coefficient or solution. It is well known that the KL expansion can generate an optimal basis in the sense that it minimizes the total mean squared error. In our method, the model reduction in the stochastic dimension consists of two essential parts: (1) a compact parameterization for the (smooth) effective coefficient $a^*(x,\omega)$ and (2) a problem-dependent compact basis to represent the stochastic solution to the effective (1.3).

In the first step of stochastic model reduction, we compute the truncated KL expansion of $a^*(x,\omega)$,

$$a^*_{ij}(x,\omega) \approx \bar{a}_{ij}(x) + \sum_{m=1}^{M} \sqrt{\lambda_{ij,m}}\xi_{ij,m}(\omega)\phi_{ij,m}(x), \quad 1 \le i,j \le d, \tag{1.6}$$

and save the KL expansion results. We have used the stochastic collocation samples (1.5) to estimate the covariance function of $a^*(x,\omega)$. This compact representation of the effective coefficient enables us to generate $a^*(x,\omega)$ very efficiently in the online stage.

In the second step of stochastic model reduction, we construct a data-driven stochastic basis by applying the DSM for the effective stochastic equation (1.3). We assume that $f(x,\theta)$ can be approximated by a finite-dimensional basis $f_k(x)$, i.e., $f(x,\theta) \approx \sum_{k=0}^{K} c_k(\theta) f_k(x)$. With such a parameterization of $f(x,\theta)$, we first solve (1.3) with $f_0(x)$ as a forcing function, and then use the KL expansion of the solution to construct the stochastic basis $\{A_i(\omega)\}_{i=0}^m$. For each query $f(x,\theta)$, we expand the solution $u^*(x,\omega)$ in terms of the stochastic basis $\{A_i(\omega)\}_{i=0}^m$, i.e.,

$$u^*(x,\omega) \equiv \sum_{i=0}^{m} A_i(\omega) u_i(x),$$

and use the stochastic Galerkin method to determine the expansion coefficient $u_i(x)$. An error analysis is used to evaluate the completeness of the data-driven basis $\{A_i(\omega)\}_{i=0}^m$. A greedy-type algorithm combined with a two-level preconditioning [21] is used to reduce the computational cost. First, we solve the error equation on the coarse grid for each trial function $f_k(x)$, $k = 1, 2, \ldots, K$. We identify the trial function f_{k^*}, which gives the maximum error, and solve the error equation again with this trial function on the fine grid. After that, the KL expansion of the error can be used to enrich the stochastic basis. This process is repeated until the maximum residual error is below the prescribed threshold δ. When this updating process terminates, we obtain a compact data-driven basis $\{A_i(\omega)\}_{i=0}^m$ for the effective stochastic equation (1.3) which applies to all forcing functions. The detailed implementation of this enriching algorithm depends on specific numerical representation of the stochastic basis, which will be elaborated in detail in section 4.

We use two different sets of *fine* and *coarse* grids in the multiscale model reduction method and the data-driven stochastic method, respectively. For the multiscale model reduction in the spatial dimension, the fine grid and coarse grid are chosen to resolve the multiscale (1.1) and the effective equation (1.3), respectively. However, in the data-driven stochastic method, we use a fine grid to resolve the effective equation (1.3), and we choose a much coarser grid to further reduce the computational cost in training the data-driven stochastic basis.

To clarify, let h_f^{MMR} and h_c^{MMR} denote the fine and coarse mesh sizes in the multiscale model reduction method, and let h_f^{DSM} and h_c^{DSM} denote the fine and coarse mesh sizes in the data-driven stochastic method. In this paper, we have chosen that $h_f^{MMR} < \varepsilon < h_c^{MMR} = h_f^{DSM} < h_c^{DSM}$. For instance, in Example 5 of section 6, the smallest scale of the elliptic coefficient is of order $\epsilon = 1/65$. We choose a 1024×1024 fine gird to resolve the multiscale problem, and a 64×64 coarse grid to compute the effective SPDE. To obtain the data-driven stochastic basis for this effective SPDE, we choose the 64×64 fine grid to calculate the stochastic basis, and a 16×16 grid to do the preconditioning and select the candidate force function.

In the online stage, we expand the solution of (1.3) in terms of the data-driven stochastic basis and solve a set of coupled deterministic PDEs to obtain the coefficients. We remark that deriving the effective stochastic equation (1.3) and constructing the data-driven basis $\{A_i(\omega)\}_{i=0}^m$ can be expensive if we solve (1.1) only once for a given forcing function. However, when we need to solve the same (1.1) many times with multiple forcing functions, the MsDSM in the online stage offers considerable computational savings, since our method takes advantage of the model reduction in both the stochastic and the physical dimensions.

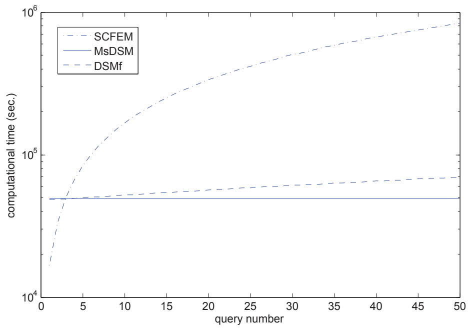


FIG. 1. *The computation time comparison. SCFEM: Stochastic collocation finite element method. MsDSM: Multiscale DSM on a coarse grid. DSMf: The DSM on a fine grid.*

We have carried out complexity analysis to compare the complexity of our method with that of the stochastic collocation finite element method (SCFEM) as well as with the DSM on a fine grid (DSMf). Our study shows that the computational saving of the MsDSM over the SCFEM is quite significant when the number of queries is large. To illustrate the main idea, we choose one particular example, which is Example 5 in section 6. In this problem, the smallest scale of the elliptic coefficient is of order $\epsilon = 1/65$. We need to use a 1024×1024 fine mesh to fully resolve this multiscale problem if we use either the SCFEM or the DSM. On the other hand, since the MsDSM solves the effective SPDE, we can use a 64×64 relative coarse mesh to achieve comparable accuracy. Our complexity analysis gives the following timing models for the three methods: (i) SCFEM: $t_{SCFEM} = 18620.01n$, (ii) MsDSM: $t_{MsDSM} = 49258.59 + 18.25n$, (iii) $t_{DSM} = 47700.90 + 438.62n$, where n is the number of queries. One can see that the online cost per query for MsDSM is the smallest (18.25 seconds) among the three methods. The first term in the timing model is the offline computational cost, which is absent for the SCFEM. We plot the CPU time comparison in the logarithmic scale in Figure 1. As we can see, the computational saving of the MsDSM over the SCFEM or the DSM is quite dramatic for n large. The MsDSM gives superior performance over the SCFEM even for a relatively small number of queries. More discussions can be found in sections 5 and 6.

The rest of the paper is organized as follows. In section 2, we give a brief introduction of the KL expansion and the gPC basis. In section 3, we review the derivation of the effective equation for a deterministic multiscale elliptic equation and its analytic results. We present our derivation of the MsDSM in section 4. In section 5, we perform complexity analysis and construct several timing models to illustrate the computational complexities of different methods. In section 6, we discuss some numerical implementation issues and present several numerical results to demonstrate the accuracy and effectiveness of our method. Finally, some concluding remarks are given in section 7.

2. Some preliminaries.

2.1. The Karhunen–Loève expansion. In the theory of stochastic processes, the KL expansion [26, 29] is a representation of a stochastic process as an infinite linear combination of orthogonal functions. The importance of the KL expansion is that it yields an optimal basis in the sense that it minimizes the total mean squared error.

Consider a probability space (Ω, F, P), whose event space is Ω and is equipped with σ-algebra F and probability measure P. Suppose $u(x,\omega)$, defined on a compact spatial domain $D \subseteq R^d$, is a second-order stochastic process, i.e., $u(x,\omega) \in L^2(D\times\Omega)$. Its KL expansion reads as

$$u(x,\omega) = \bar{u}(x) + \sum_{i=1}^{\infty} \sqrt{\lambda_i}\xi_i(\omega)\phi_i(x),$$

where $\bar{u}(x) = \mathbb{E}\left[u(x,\omega)\right]$, $\{\lambda_i, \phi_i(x)\}_{i=1}^{\infty}$ are the eigenpairs of the covariance kernel $C(x,y)$, i.e.,

$$\int_D C(x,y)\phi(y)\mathrm{d}y = \lambda\phi(x). \tag{2.1}$$

The covariance kernel $C(x,y)$ is defined as

$$C(x,y) = \mathbb{E}\left[(u(x,\omega) - \bar{u}(x))(u(y,\omega) - \bar{u}(y))\right]. \tag{2.2}$$

The random variables $\{\xi_i(\omega)\}_{i=1}^{\infty}$ are defined as

$$\xi_i(\omega) = \frac{1}{\sqrt{\lambda_i}} \int_D (u(x,\omega) - \bar{u}(x))\phi_i(x)\mathrm{d}x. \tag{2.3}$$

Moreover, these random variables $\{\xi_i(\omega)\}$ are of zero mean and are uncorrelated, i.e., $\mathbb{E}\left[\xi_i\right] = 0$, $\mathbb{E}\left[\xi_i\xi_j\right] = \delta_{ij}$. Generally, the eigenvalues λ_i are sorted in descending order and cluster at zero, and their decay rate depends on the regularity of the covariance kernel $C(x,y)$. It has been proven that algebraic decay rate, i.e., $\lambda_k = O(k^{-\gamma})$, is achieved asymptotically if the covariance kernel is of finite Sobolev regularity or exponential decay, i.e., $\lambda_k = O(e^{-\gamma k})$ for some $\gamma > 0$, if the covariance kernel is piecewise analytical [36]. In general, the decay rate depends on the correlation length of the stochastic solution. Small correlation length results in slow decay of the eigenvalues. In any case, an m-term truncated KL expansion converges in $L^2(D\times\Omega)$ to the original stochastic process $u(x,\omega)$ as m tends to infinity. Let ϵ_m denote the truncation error; we have

$$||\epsilon_m||^2_{L^2(D\times\Omega)} = \left\| \sum_{i=m+1}^{\infty} \sqrt{\lambda_i}\xi_i(\omega)\phi_i(x) \right\|^2_{L^2(D\times\Omega)} = \sum_{i=m+1}^{\infty} \lambda_i \to 0, \quad m \to \infty, \tag{2.4}$$

where we have used the biorthogonality of the KL expansion.

In practical computations, we truncate the KL expansion into its first m terms and obtain the following truncated KL expansion:

$$u(x,\omega) \approx \bar{u}(x) + \sum_{i=1}^{m} \sqrt{\lambda_i}\xi_i(\omega)\phi_i(x). \tag{2.5}$$

The truncation error analysis in (2.5) reveals the most important property of KL expansion. More specifically, given any integer m and orthonormal basis $\{\psi_i(x)\}_{i=1}^{m}$, we may approximate the stochastic process $u(x,\omega)$ by

$$u_{m,\psi}(x,\omega) = \bar{u}(x) + \sum_{i=1}^{m} \zeta_i(\omega)\psi_i(x), \tag{2.6}$$

where $\zeta_i(\omega)$, $i = 1, \ldots, m$, are the expansion coefficients. Among all m-term approximations using an orthonormal basis, the KL expansion given by (2.5) is the one that minimizes the total mean square error. In this sense, we say that the KL expansion gives the optimal (or the most compact) basis to represent the stochastic solution in the energy norm. Due to the biorthogonality of the KL expansion, we will call the stochastic part of the KL expansion the *data-driven basis* in the rest of this paper.

2.2. The generalized polynomial chaos basis. In many physical and engineering application problems, randomness generally comes from various independent sources, so randomness in SPDE (1.1) is often given in terms of independent random variables. We assume that the randomness in SPDE (1.1) is given in terms of r independent random variables, i.e., $\boldsymbol{\xi}(\omega) = (\xi_1(\omega), \xi_2(\omega), \ldots, \xi_r(\omega))$. Without loss of generality, we can further assume that such independent random variables have the same distribution function $\rho(x)$. We get $a^\varepsilon(x,\omega) = a^\varepsilon(x, \xi_1(\omega), \ldots, \xi_r(\omega))$. By the Doob–Dynkin lemma [34], the solution of (1.1) can still be represented by these random variables, i.e., $u^\varepsilon(x,\omega) = u^\varepsilon(x, \xi_1(\omega), ..., \xi_r(\omega))$.

Let $\{\mathrm{H}_i(\xi)\}_{i=1}^{\infty}$ denote the one-dimensional (1D), $\rho(\xi)$-orthogonal polynomials, i.e.,

$$\int_\Omega \mathrm{H}_i(\xi)\mathrm{H}_j(\xi)\rho(\xi)d\xi = \delta_{ij}.$$

For some commonly used distributions, such as the Gaussian distribution and the uniform distribution, such orthogonal polynomial sets are Hermite polynomials and Legendre polynomials, respectively. For general distributions, such polynomial sets can be obtained by numerical methods [39]. Furthermore, by a tensor product representation, we can use the 1D polynomial $\mathrm{H}_i(\xi)$ to construct an orthonormal basis $\mathbf{H}_{\boldsymbol{\alpha}}(\boldsymbol{\xi})$ of $\mathbb{L}^2(\Omega)$ as

$$\mathbf{H}_{\boldsymbol{\alpha}}(\boldsymbol{\xi}) = \prod_{i=1}^{r} \mathrm{H}_{\alpha_i}(\xi_i), \quad \boldsymbol{\alpha} \in \mathfrak{J}_r^\infty, \tag{2.7}$$

where $\boldsymbol{\alpha}$ is a multi-index and $\mathfrak{J}_r^\infty$ is a multi-index set of countable cardinality,

$$\mathfrak{J}_r^\infty = \{\boldsymbol{\alpha} = (\alpha_1, \alpha_2, \ldots, \alpha_r) \mid \alpha_i \geq 0, \alpha_i \in \mathbb{N}\}.$$

The zero multi-index corresponding to $\mathbf{H_0}(\boldsymbol{\xi}) = 1$ is used to represent the mean of the solution. Clearly, the cardinality of $\mathfrak{J}_r^\infty$ is infinite. In practical computation, we have to truncate it into a finite set. One possible choice is the set of polynomials whose total orders are at most p, i.e.,

$$\mathfrak{J}_r^p = \left\{\boldsymbol{\alpha} \,|\, \boldsymbol{\alpha} = (\alpha_1, \alpha_2, \ldots, \alpha_r), \alpha_i \geq 0, \alpha_i \in \mathbb{N}, |\alpha| = \sum_{i=1}^{r} \alpha_i \leq p\right\}. \tag{2.8}$$

The cardinality of $\mathfrak{J}_r^p$ in (2.8) or the number of polynomial basis functions, denoted by $N_p = |\mathfrak{J}_r^p|$, is equal to $\frac{(p+r)!}{p!r!}$. We may simply write such a truncated set as $\mathfrak{J}$ when no ambiguity arises. The orthonormal basis $\mathbf{H}_{\boldsymbol{\alpha}}(\boldsymbol{\xi})$ is the standard gPC basis; see [7, 24, 41, 25] for more details.

3. Model reduction for a multiscale elliptic equation. In this section, we give a brief review of the multiscale model reduction method [12]. We will illustrate this upscaling method through the deterministic multiscale elliptic equation

$$-\nabla \cdot (a^\varepsilon(x)\nabla u^\varepsilon(x)) = f(x), \quad x \in D, \tag{3.1}$$

$$u^\varepsilon(x) = 0, \quad x \in \partial D, \tag{3.2}$$

where $D \in R^d$ is a bounded spatial domain and multiscale coefficient matrix $a^\varepsilon(x)$ is a symmetric, positive definite matrix satisfying $\lambda_{min}(x) \geq \alpha > 0$ ($\lambda_{min}(x)$ is the smallest eigenvalue of $a^\varepsilon(x)$) for $x \in D$. For notational simplicity, in the rest of this section we omit the superscript ε. Let $F(x) = (F_1(x), \ldots, F_d(x))$ be the a-harmonic coordinates associated with (3.1) in dimension d. Then, F_k ($k = 1, \ldots, d$) satisfies the elliptic equation

$$\begin{cases} -\nabla \cdot (a(x)\nabla F_k(x)) = 0 & \text{in} \quad D, \\ F_k = x_k & \text{on} \quad \partial D, \end{cases} \tag{3.3}$$

where $x = (x_1, \ldots, x_d)$. Write $\tilde{u}_0 = u \circ F^{-1}$. It is well known that the solution u is smooth in terms of the harmonic coordinates, i.e., $\tilde{u}_0$ is smooth (see [35]). If we make a decomposition $F = g + \chi$ such that g is smooth and invertible, and χ is small with zero boundary conditions, then we obtain, by applying a formal Taylor expansion to $\tilde{u}_0$ and ignoring the higher order terms,

$$\tilde{u}_0(F) = \tilde{u}_0(g + \chi) \approx \tilde{u}_0(g) + \chi^T \nabla_g \tilde{u}_0(g). \tag{3.4}$$

Let $u_0(x) = \tilde{u}_0(g(x))$; then we get

$$u(x) = \tilde{u}_0(F) \approx u_0(x) + \chi^T \frac{\partial x}{\partial g} \nabla u_0(x). \tag{3.5}$$

Furthermore, we have

$$\nabla u(x) \approx \nabla u_0(x) + \frac{\partial \chi}{\partial x}\frac{\partial x}{\partial g}\nabla u_0(x) + \chi^T \nabla \left(\frac{\partial x}{\partial g}\nabla u_0(x)\right). \tag{3.6}$$

By substituting (3.6) into (3.1) and eliminating the small terms involving $O(\chi)$, we get a new PDE for u_0 as

$$-\nabla \cdot (a^*(x)\nabla u_0(x)) = f(x), \quad x \in D, \tag{3.7}$$

$$u_0(x) = 0, \quad x \in \partial D, \tag{3.8}$$

where $a^*(x) = a(x)(I + \frac{\partial \chi}{\partial x}\frac{\partial x}{\partial g})$ denotes the effective coefficient and I is the identity matrix. Although the above derivation is formal and approximate, we can prove that the $a^*(x)$ has some nice properties. In [12], error analysis is performed to show that the difference between the solution of the effective equation (3.7) and the solution of (3.1) can be bounded in the H^1 norm by the maximum norm of the oscillatory part of the harmonic coordinates. The main result can be summarized in the following theorem.

THEOREM 3.1. *Suppose u, F, and u_0 are weak solutions to* (3.1), (3.3), *and* (3.7), *respectively. Let $u_1 = \chi^T \frac{\partial x}{\partial g}\nabla u_0$, $F = g + \chi$, and $\chi = 0$ on $\partial\Omega$. Then we have*

$$\|u - u_0 - u_1\|_{H^1(\Omega)} \leq C \left\|\frac{\partial g}{\partial x}\right\|_{L^\infty(\Omega)} \|\chi\|_{L^\infty(\Omega)} \left\|\det\left(\frac{\partial x}{\partial g}\right)\right\|_{L^\infty(\Omega)} |\tilde{u}_0|_{H^2(\Omega)}, \tag{3.9}$$

where C is a constant that depends on n, Ω, and a. And $\tilde{u}_0 = u_0 \circ g^{-1}$.

In general, the oscillatory part of the harmonic coordinates is small. The smallness of the oscillatory part will depend on the regularity of the multiscale coefficient of the problem. In the case when the problem has scale separation and periodic structure, this method recovers the homogenized equation from the classical homogenization theory, and the oscillatory part of the harmonic coordinates can be proved to be small in the H^1 norm. One important advantage of this method is that one does not require the problem to have scale separation or periodic structures. Thus the method can be applied to solve more challenging problems arising from various physical applications.

Theorem 3.1 indicates that one can first solve (3.7) accurately on a coarse mesh to obtain u_0, and then approximate u by $u_0 + \chi^T \frac{\partial x}{\partial g} \nabla u_0$. This suggests the following steps to derive the effective equation.

Step 1. Solve (3.3) on a fine mesh to get harmonic coordinates F.

Step 2. Decompose $F = g + \chi$, where g is smooth and χ is small with $\chi i = 0$ on ∂D.

Step 3. Solve (3.7) on a coarse mesh to get u_0.

Step 4. Approximate u by $u_0 + \chi^T \frac{\partial x}{\partial g} \nabla u_0$.

The first and second steps are solved in the offline stage. We can store the information about g and χ so that we could compute u_0 efficiently for different f. The remaining steps are solved very efficiently on a coarse mesh in the online stage. In [12], the authors gave some guidelines on how to construct the decomposition of the harmonic coordinates. The first criterion is to make sure that g is smooth and invertible. The second criterion is to make χ small. A simple but effective way to construct g is to choose the nodal values of g at the coarse mesh points to be the local average of F around these coarse mesh points [12]. One can then interpolate g from the coarse mesh points to the fine mesh points using the linear finite element interpolation. Then χ is given by $\chi = F - g$. Since F is linear on the boundary, such a decomposition guarantees that $g = F$ on the boundary, which implies that $\chi = 0$ on $\partial\Omega$.

4. Multiscale data-driven stochastic method.

It is extremely challenging to solve the SPDE involving multiple scales. We not only need to use a very fine mesh to resolve the small scales of the solution in the physical dimension, but we also need to approximate the solution in the stochastic space whose dimension could be high. In applications, we often need to solve the same SPDE many times with multiple forcing functions or boundary conditions, which is known as the multiquery problem. It is computationally infeasible to solve a multiquery multiscale stochastic problem using traditional methods. In this paper, we propose a multiscale data-driven stochastic method (MsDSM) to reduce the computational complexity of solving the multiquery multiscale stochastic problem.

Our MsDSM consists of offline and online stages. In the offline stage, we generate collocation points or samples according to the distribution information of the random parameters [43]. On each collocation, or sample, point, the multiscale problem (1.1)-(1.2) becomes a deterministic one. We derive an effective stochastic equation that can be resolved on a coarse grid. We then construct a data-driven stochastic basis which gives a compact representation for the solutions of the effective stochastic equation for a broad range of forcing functions and/or boundary conditions. In the online stage, we represent the multiscale stochastic solution in terms of this data-driven stochastic basis, and we need only solve a small number of coupled deterministic PDEs. This leads to considerable computational savings when we need to solve the same multiscale stochastic PDE many times with multiple queries.

4.1. Derivation of the effective equation for a stochastic partial differential equation involving multiple scales. We use the stochastic elliptic equations (1.1)–(1.2) described in the introduction as an example to illustrate the main idea. It is important to note that the effective coefficient $a^*(x)$ derived in (3.7) depends only on the multiscale coefficient and decompositions of the harmonic coordinates, and does not depend on the forcing term. Therefore, we can apply this idea to upscale the multiscale stochastic coefficient in (1.1). For each collocation, or sample, point $\omega_l \in \Omega$, the multiscale problem (1.1)–(1.2) becomes a deterministic PDE. We first solve the corresponding homogeneous problem with specified boundary conditions to obtain the harmonic coordinates F. Then, we decompose the harmonic coordinates F into a smooth part g plus a highly oscillatory part χ: $F = g + \chi$. According to the analysis in section 3, the effective coefficient can be given in terms of $a^\varepsilon(x, \omega_l)$, g, and χ, i.e.,

$$a^*(x, \omega_l) = a^\varepsilon(x, \omega_l)\left(I + \frac{\partial \chi}{\partial x}(x, \omega_l)\frac{\partial x}{\partial g}(x, \omega_l)\right), \quad \omega_l \in \Omega_s, \tag{4.1}$$

where I is the identity matrix. The effective coefficient in (4.1) is valid for each sample ω_l in the sample space Ω_s. Looping over all the samples, we can obtain effective stochastic equations of the following form:

$$-\nabla \cdot (a^*(x, \omega)\nabla u^*(x, \omega)) = f(x, \theta), \quad x \in D, \omega \in \Omega_s, \theta \in \Theta, \tag{4.2}$$

$$u^*(x, \omega) = 0, \quad x \in \partial D. \tag{4.3}$$

According to the analysis in section 3, the solution to the effective equation (4.2) is one order smoother than the original multiscale equation (1.1). Thus, we can solve the effective equation (4.2) on a coarse mesh.

To save memory, we compute the M-term truncated KL expansion of each entry of the $a^*(x, \omega)$ in (4.2),

$$a^*_{ij}(x, \omega) \approx \bar{a}_{ij}(x) + \sum_{m=1}^{M} \sqrt{\lambda_{ij,m}}\xi_{ij,m}(\omega)\phi_{ij,m}(x), \quad 1 \le i, j \le d. \tag{4.4}$$

Then, we expand the stochastic basis $\xi_{ij,m}(\omega)$ in the gPC basis $\mathbf{H}_{\boldsymbol{\alpha}}(\boldsymbol{\xi})$ with the index given by (2.8), i.e.,

$$\xi_{ij,m}(\omega) = \sum_{\boldsymbol{\alpha} \in \mathfrak{J}} \xi_{ij,m\boldsymbol{\alpha}}\mathbf{H}_{\boldsymbol{\alpha}}(\boldsymbol{\xi}(\omega)), \quad 1 \le i, j \le d. \tag{4.5}$$

The expansion coefficient $\xi_{ij,m\boldsymbol{\alpha}}$ is given by $\xi_{ij,m\boldsymbol{\alpha}} = E[\xi_{ij,m}(\omega)\mathbf{H}_{\boldsymbol{\alpha}}(\boldsymbol{\xi}(\omega))]$. To compute the expectation in the KL expansion (4.4) or expansion coefficient (4.5), we choose the quadrature rules based on the sparse grids [43]. For instance,

$$\xi_{ij,m\boldsymbol{\alpha}} = E[\xi_{ij,m}(\omega)\mathbf{H}_{\boldsymbol{\alpha}}(\boldsymbol{\xi}(\omega))] \approx \sum_{l=1}^{L} \xi_{ij,m}(\omega_l)\mathbf{H}_{\boldsymbol{\alpha}}(\boldsymbol{\xi}(\omega_l))s_l, \quad \omega_l \in \Omega_s,$$

where s_l are the associated weights, and L is the number of sparse grids. If we choose Monte Carlo method to generate the samples, we use the sample average to compute the expectations.

In addition, we compute the correction term $(\chi^T \frac{\partial x}{\partial g})(x,\omega)$ in the offline stage. Actually, this can be done simultaneously when we derive the effective stochastic equation. For each collocation, or sample, point $\omega_l \in \Omega_s$, after we decompose the harmonic coordinates F into a smooth part g and a highly oscillatory part χ, we can compute the correction vector term $(\chi^T \frac{\partial x}{\partial g})(x,\omega_l)$. Based on these samples, we can calculate the N-term truncated KL expansion of the correction term $\chi^T \frac{\partial x}{\partial g}(x,\omega)$, i.e.,

$$\chi^T \frac{\partial x}{\partial g}(x,\omega) = \bar{c}(x) + \sum_{n=1}^{N} \sqrt{\lambda_n}\vartheta_n(\omega)\psi_n(x). \tag{4.6}$$

Furthermore, we expand the stochastic basis $\vartheta_n(\omega)$ in the gPC $\mathbf{H}_{\boldsymbol{\alpha}}(\boldsymbol{\xi})$ with the index given by (2.8), i.e.,

$$\vartheta_n(\omega) = \sum_{\boldsymbol{\alpha}\in\mathfrak{J}} \vartheta_{\boldsymbol{\alpha}n}\mathbf{H}_{\boldsymbol{\alpha}}(\boldsymbol{\xi}(\omega)), \tag{4.7}$$

where the expansion coefficient $\vartheta_{\boldsymbol{\alpha}n}$ is given by $\vartheta_{\boldsymbol{\alpha}n} = E[\vartheta_n(\omega)\mathbf{H}_{\boldsymbol{\alpha}}(\boldsymbol{\xi}(\omega))]$.

4.2. The data-driven stochastic basis for the effective stochastic equation. In this subsection, we consider the model reduction in the stochastic dimension for the effective equation (4.2). We first summarize the assumptions that we make for $a^\varepsilon(x,\omega)$ and $f(x,\theta)$ as follows:

- The coefficient $a^\varepsilon(x,\omega)$ in (1.1) is given in terms of r independent random variables, i.e., $a^\varepsilon(x,\omega) = a^\varepsilon(x,\boldsymbol{\xi}(\omega)) = a^\varepsilon(x,\xi_1(\omega),\dots,\xi_r(\omega))$. Therefore, by the Doob–Dynkin lemma, the harmonic coordinates as well as the effective coefficient $a^*(x,\omega)$ in (4.2) can still be represented by these random variables, i.e., $a^*(x,\omega) = a^*(x,\boldsymbol{\xi}(\omega)) = a^*(x,\xi_1(\omega),\dots,\xi_r(\omega))$.
- The order of the gPC basis $\mathbf{H}_{\boldsymbol{\alpha}}(\boldsymbol{\xi})$ is sufficiently high to properly represent the stochastic coefficients and solutions in (1.1) and (4.2).
- The force $f(x,\theta)$ in (4.2) can be expanded into a finite-dimensional basis $f_k(x)$, i.e., $f(x,\theta) \approx \sum_{k=0}^{K} c_k(\theta)f_k(x)$, and well-resolved on a coarse mesh, with mesh size $h \gg \varepsilon$.

We now begin our construction of the data-driven stochastic basis for the effective stochastic equation (4.2). In the data-driven method, we use a fine grid to discretize the effective equation (4.2). To reduce the offline computation, we choose a much coarser grid to select the candidate force function.

Due to limitations of space, we discuss only the data-driven stochastic basis in stochastic collocation representation. The DSM consists of two steps, i.e., initial leaning and update steps. See Figure 2 for the general framework of the DSM. We refer the reader to [11] for more details.

In the *initial learning step* of the DSM, we first use the stochastic collocation method to generate L collocation points ω_l according to the distribution of the coefficient $a^\varepsilon(x,\omega)$ in (1.1) as well as the associated weights s_l. Then, we solve (4.2),(4.3) with the random variable evaluated at the collocation grid points and $f_0(x)$ as the right-hand side,

$$-\nabla\cdot(a^*(x,\omega_l)\nabla u^*(x,\omega_l)) = f_0(x), \quad x\in D, l=1,\dots,L, \tag{4.8}$$

$$u(x,\omega_l) = 0, \quad x \in \partial D. \tag{4.9}$$

By solving (4.8)–(4.9), we can obtain the values of the stochastic solution $u^*(x,\omega;f_0)$ on the collocation points, i.e., $\{u^*(x,\omega_l;f_0)\}_{l=1}^{L}$. The m_1-term KL expansion of the

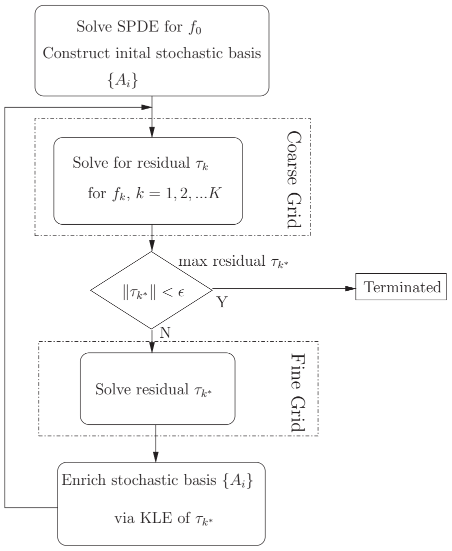

Solve SPDE for f_0
Construct inital stochastic basis $\{A_i\}$
Coarse Grid
Solve for residual τ_k for f_k, $k = 1, 2, ...K$
max residual τ_{k^*}
$\|\tau_{k^*}\| < \epsilon$
Y
Terminated
N
Fine Grid
Solve residual τ_{k^*}
Enrich stochastic basis $\{A_i\}$ via KLE of τ_{k^*}

FIG. 2. *Greedy stochastic basis enriching algorithm on a coarse-fine grid hierarchy.*

solution $u^*(x, \omega; f_0)$ gives the dominant components in the random space. We use the decaying property of eigenvalues to select parameter m_1; i.e., the number of stochastic basis m_1 can be chosen such that $\lambda_{m_1+1}/\lambda_1$ is smaller than some predefined threshold, say, 10^{-4}. We denote the truncated KL expansion as

$$u^*(x, \omega; f_0) \approx \bar{u}(x; f_0) + \sum_{i=1}^{m_1} \sqrt{\lambda_i} A_i(\omega)\phi_i(x; f_0). \tag{4.10}$$

We call the stochastic basis $\{A_i(\omega)\}_{i=0}^{m_1}$ in (4.10) the data-driven stochastic basis, where $A_0(\omega) = 1$. Furthermore, we would like to expand the stochastic basis $A_i(\omega)$ in a gPC basis $\mathbf{H}_{\boldsymbol{\alpha}}(\boldsymbol{\xi})$, i.e.,

$$A_i(\omega) = \sum_{\boldsymbol{\alpha}} A_{\boldsymbol{\alpha} i}\mathbf{H}_{\boldsymbol{\alpha}}(\boldsymbol{\xi}(\omega)). \tag{4.11}$$

The expansion coefficient $A_{\boldsymbol{\alpha} i}$ is given by

$$A_{\boldsymbol{\alpha} i} = E[A_i(\omega)\mathbf{H}_{\boldsymbol{\alpha}}(\boldsymbol{\xi}(\omega))] \approx \sum_{l=1}^{L} A_i(\omega_l)\mathbf{H}_{\boldsymbol{\alpha}}(\boldsymbol{\xi}(\omega_l))s_l, \quad \boldsymbol{\alpha} \in \mathfrak{J}, \tag{4.12}$$

where ω_l and s_l are the sparse grid points and the associated weights, respectively. We use the N_p-by-(m_1+1) matrix $\mathbf{A}$ to denote the expansion coefficient $A_{\boldsymbol{\alpha} i}$, which is essentially the data-driven stochastic basis in the stochastic collocation representation. In general, the stochastic basis constructed by using f_0 may not be adequate to give an accurate approximation of the SPDE for another right-hand side, $f(x, \theta)$. We need to supplement the stochastic basis by using multiple trial functions involving other f_k.

In the *preconditioning and update step* of our DSM, we propose a greedy-type algorithm and adopt a two-level preconditioning strategy [22] to enrich the stochastic basis. First, we perform an error analysis. Given a new right-hand side $f_1(x) =$

$f(x,\theta)$ for some choice of θ, we expand the solution in terms of the stochastic basis, $\{A_i(\omega)\}_{i=0}^{m_1}$,

$$u^*(x,\omega;f_1) \approx \bar{u}(x;f_1) + \sum_{i=1}^{m_1} A_i(\omega)u_i(x;f_1) \equiv \sum_{i=0}^{m_1} A_i(\omega)u_i(x;f_1). \tag{4.13}$$

In the rest of this subsection, we also use $u_i(x) \equiv u_i(x;f_1)$ for simplification. We use the standard stochastic Galerkin method to obtain the coefficient $u_i(x)$. Specifically, we substitute the expansion (4.13) into (4.2), multiply both sides by $A_j(\omega)$, and take expectations. This gives rise to a coupled PDE system for the expansion coefficient $u_i(x)$,

$$-\nabla \cdot (E[a^* A_i A_j]\nabla u_i) = f_1(x)E[A_j], \quad x \in D,\ j = 0,1,\ldots,m_1, \tag{4.14}$$

$$u_i(x) = 0, \quad x \in \partial D, \tag{4.15}$$

where Einstein summation is assumed. The term $E[a^* A_i A_j]$ can be calculated by the stochastic collocation method. Solving the coupled deterministic PDE system (4.14)–(4.15) by a finite element method (FEM) or a finite difference method (FDM), we obtain the expansion coefficient $\{u_i(x)\}_{i=0}^{m_1}$ and an approximate solution for $u^*(x,\omega;f_1)$ given by (4.13). We know that the exact solution can be written as

$$u^*(x,\omega;f_1) = \sum_{i=0}^{m_1} A_i(\omega)u_i(x;f_1) + \tau(x,\omega;f_1), \tag{4.16}$$

where $\tau(x,\omega;f_1)$ is the error. Simple calculations show that the error satisfies the following equation:

$$-\nabla \cdot (a^*(x,\omega)\nabla\tau(x,\omega;f_1)) = f_1(x) + \sum_{i=0}^{m_1} \nabla \cdot (a(x,\omega)A_i(\omega)\nabla u_i(x)). \tag{4.17}$$

For a different f_k, we can obtain a similar error equation for the error $\tau(x,\omega;f_k)$ by replacing f_1 by f_k in the above error equation. To verify the effectiveness of the stochastic basis, we solve the error (4.17) on a coarse grid for each $f_k(x)$, $k = 1,\ldots,K$, and obtain the error $\{\tau(x,\omega;f_k)\}_{k=1}^{K}$. If $\max_{1\le k\le K} ||\tau(x,\omega;f_k)|| < \delta$, then we consider this stochastic basis complete. Here, we choose $||\cdot||$ as the L^2 norm of the variance of the stochastic solution. Otherwise, we identify the maximum error $\tau_{k^*} = \max_{1\le k\le K} ||\tau(x,\omega;f_k)||$ and the corresponding trial function $f_{k^*}(x)$. Subsequently, we solve the residual (4.17) for this trial function $f_{k^*}(x)$ one more time on a fine grid. Again, we perform the KL expansion for the residual solution $\tau(x,\omega;f_{k^*})$, and extract several dominant components in the random space, and use them as a supplement to the current stochastic basis. To improve the numerical stability, we apply some stable orthogonalization procedures, such as the modified Gram–Schmidt process, to produce an orthogonal basis. We use $\{A_i(\omega)\}_{i=0}^{m_2}$ to denote the updated stochastic basis. This process is repeated until the maximum residual is below a prescribed threshold ϵ_0. We project the stochastic basis denoted by $\{A_i(\omega)\}_{i=0}^{m}$ into the generalized polynomial chaos basis according to (4.11), (4.12) and save only the N_p-by-$(m+1)$ matrix $\mathbf{A}$.

In the *online stage*, for each query $f(x,\theta)$ in (1.1), the corresponding stochastic solution $u^\varepsilon(x,\omega)$ can be approximated by the MsDSM solution in two steps. First, with our data-driven stochastic basis $\{A_i(\omega)\}_{i=0}^{m}$, we use the standard stochastic Galerkin method to solve the effective stochastic equation (4.2) to obtain $u^*(x,\omega)$.

Then we obtain the approximate solution by adding correction terms into $u^*(x,\omega)$, i.e.,

$$u^\varepsilon(x,\omega) \approx u_{MsDSM}(x,\omega) \equiv u^*(x,\omega) + \chi^T \frac{\partial x}{\partial g}(x,\omega)\nabla u^*(x,\omega). \tag{4.18}$$

The construction of the effective stochastic equation (4.2) and the data-driven stochastic basis $\{A_i(\omega)\}_{i=0}^m$ could be expensive. However, in a multiple query problem, the MsDSM offers considerably more computational savings than traditional methods because of the model reduction in both the physical and stochastic dimensions. We will demonstrate this through several numerical examples in section 6.

REMARK 4.1. *It is important to point out that since our methods involve the computation of global harmonic coordinates, the memory consumption becomes a serious issue when the ratio of the smallest scale and the largest scale in the stochastic multiscale problem* (1.1) *becomes extremely small. We have proposed a multiscale multilevel Monte Carlo (MsMLMC) method* [13]*, which is mainly based on the localized upscaling method and multilevel Monte Carlo method, to address this issue.*

4.3. The complete algorithm of the multiscale data-driven stochastic method. In this section, we give the complete algorithm of the MsDSM to solve the multiscale stochastic equation in a multiquery setting. Our method consists of offline and online stages. Since the online stage is pretty straightforward and was presented in section 4.2, we state only the offline computation algorithm as follows.

MsDSM offline computation.

- (I) (Preparations):
  - Set error threshold δ; divide the spatial domain D into different grids with size $h_f^{MMR} < \varepsilon < h_c^{MMR} = h_f^{DSM} < h_c^{DSM}$.
  - Approximate $f(x,\theta)$ by a finite-dimensional basis $\{f_k(x)\}_{k=0}^K$, that is, $f(x,\theta) \approx \sum_{k=0}^K c_k(\theta) f_k(x)$.
  - Generate the gPC basis $\mathbf{H}_{\boldsymbol{\alpha}}(\boldsymbol{\xi})$, the sparse grid points ω_l and its associated weights s_l, $l = 1, \ldots, L$.
- (II) (Derive the effective SPDE and calculate the correction term on the gird with size h_f^{MMR}):
  - Loop over all sparse grids, get harmonic coordinates, and obtain the effective SPDE (4.2).
  - Compute (4.4)–(4.5) to obtain a compact representation of the effective stochastic coefficient.
  - Compute the correction term ($\chi^T \frac{\partial x}{\partial g}$) as well as its KL expansion (4.6)–(4.7).
- (III) (Construct the DSM basis for effective SPDE):
  - Step III.1 (*Initial learning on the grid with size* h_f^{DSM}):
    * Solve (4.2) with $f_0(x)$ as a forcing function to obtain $u^*(x,\omega;f_0)$.
    * Calculate the truncated KL expansion of $u^*(x,\omega;f_0)$, and use the first m_1 terms of the stochastic modes to obtain the current data-driven basis $\{A_i(\omega)\}_{i=0}^{m_1}$, where $A_0(\omega) = 1$.
  - Step III.2 (*Preconditioning on the grid with size* h_c^{DSM}):
    * For each $f_k(x)$, solve (4.2) utilizing the current stochastic basis $\{A_i(\omega)\}_{i=0}^{m_1}$ and the stochastic Galerkin method to obtain the DSM solution $u_{DSM}^*(x,\omega;f_k)$.
    * For each $f_k(x)$, solve an error (4.17) to obtain the approximate residual error $\tau_k = \tau(x,\omega;f_k)$.

```
        * If $\max_{1\leq k\leq K} ||\tau_k|| < \delta$, then goto Step III.4; otherwise set
          $k^* = arg\max_{1\leq k\leq K} ||\tau_k||$ and $f_{k^*}(x)$, and goto Step III.3.
    – Step III.3 (*Update on the grid with size* $h_f^{DSM}$):
        * Solve the error equation associated with $f_{k^*}(x)$ to obtain the residual
          error $\tau_{k^*} = \tau(x,\omega; f_{k^*})$.
        * Enrich the current stochastic basis $\{A_i(\omega)\}_{i=0}^{m_1}$ by the KL expansion
          of $\tau_{k^*}$, and use $\{A_i(\omega)\}_{i=0}^{m_2}$ to denote the updated stochastic basis.
          Goto Step III.2.
    – Step III.4 (*Termination*):
        * Save the data-driven stochastic basis $\{A_i(\omega)\}_{i=0}^{m}$ and relevant sta-
          tistical quantities.
• (IV) (Save the relevant data):
    – Save the data-driven stochastic basis $\{A_i(\omega)\}_{i=0}^{m}$ and the KL expansion
      of correction term $(\chi^T \frac{\partial x}{\partial g})$.
```

5. Computational complexity analysis. The computational time of the MsDSM consists of both offline and online parts. The offline computation can be very expensive if we use a brute-force way to construct the data-driven basis. In this section, we will demonstrate through computational complexity analysis that the overhand time of offline computation is acceptable, and the online computation is "super fast." It is well known that the stochastic collocation method is very effective in solving an SPDE when the stochastic solution is smooth in the stochastic dimension. Therefore, we choose the stochastic collocation finite element method (SCFEM) as a benchmark, and compare the computational cost of the MsDSM and the SCFEM. We will compare the performance of the MsDSM and the DSM as well.

In [11], the authors have already done a thorough study and have explained why DSM is superior to the traditional methods, such as the gPC, stochastic collocation, and Monte Carlo methods in a multiquery setting. The same property still holds for the MsDSM, since it is designed with the same technique. In our numerical experiments, we find that the offline computational costs of the MsDSM and DSM have the same order of magnitude. However, due to the model reduction (upscaling) in the physical domain, the MsDSM offers more computational savings in the online stage than the DSM.

We will demonstrate this by solving a model problem, i.e., (1.1) on $D = [0,1] \times [0,1]$ with the coefficient given by

$$
\begin{aligned}
a^{\varepsilon}(x,\omega) = 0.1 &+ \xi_1(\omega)\frac{2+1.8\sin(2\pi x_1/\epsilon_1)}{2+1.8\sin(2\pi x_2/\epsilon_1)} \\
&+ \xi_2(\omega)\frac{2+1.8\sin(2\pi x_2/\epsilon_2)}{2+1.8\cos(2\pi x_1/\epsilon_2)} + \xi_3(\omega)\frac{2+1.8\cos(2\pi x_1/\epsilon_3)}{2+1.8\sin(2\pi x_2/\epsilon_3)},
\end{aligned}
\tag{5.1}
$$

where $\{\epsilon_i\}_{i=1}^3$ are multiscale parameters, and $\{\xi_i\}_{i=1}^3$ are independent uniform random variables in $[0,1]$.

Let N_h and J denote the number of the physical grid points and sparse grid points, respectively. We assume that in all tests, level six sparse grids in the SCFEM will give an accurate result. Therefore, we choose $J = 135$. All the simulations and comparisons were conducted on a single computing node with 16 GB memory at the Caltech Center for Advanced Computing Research (CACR).

Table 1
Computational time of the linear equation solver for one collocation point. (Time: Sec.)

$N_h = 32^2$	$N_h = 64^2$	$N_h = 128^2$	$N_h = 256^2$	$N_h = 512^2$	$N_h = 1024^2$
0.0065	0.0359	0.2577	1.6490	18.6875	140.0816

5.1. The computational cost of the stochastic collocation finite element method solver. We first show the computational cost of solving (1.1) once using the stochastic collocation method in Table 1 (the same result can be applied to a Monte Carlo solver). The SCFEM is very effective if the SPDE solution is smooth in the stochastic dimension; however, when the SPDE solution has multiscale features in the physical dimension, the SCFEM becomes very expensive as demonstrated in Table 1. For instance, it takes about 1.89×10^4 (135×140.0816) seconds to obtain a single query result on a 1024^2 mesh grid. Let t_{SCFEM} denote the computational time of the SCFEM solver for one forcing function; then t_{SCFEM} is approximately given by

$$t_{SCFEM} \approx 2.45 \times 10^{-7} J N_h^{1.5}. \tag{5.2}$$

5.2. The computational cost of the multiscale data-driven stochastic and the data-driven stochastic solvers. Both the MsDSM and the DSM consist of offline and online computational cost. In [11], the authors performed a complexity analysis for the DSM and compared it with other commonly used methods in the multiquery setting, such as the gPC, stochastic collocation, and Monte Carlo methods. They adopted the randomized singluar value decomposition (SVD) algorithm and a two-level preconditioning method to reduce the overhead time in the offline computing. They demonstrated through computational complexity analysis and numerical examples that with the help of all the cost-saving measures, the DSM is superior to the traditional method if one needs to solve (1.1) with a relatively small number of queries. The MsDSM inherits all these cost-saving measures when we construct the DSM basis for the effective SPDE. The only extra computational cost stems from the derivation of the effective SPDE and calculating its correction term. Roughly speaking, this part of the computational cost is equivalent to solving (1.1) with two forcing functions. In Table 2, we list the offline computational cost of the MsDSM and the DSM on a different mesh, where we fix the basis number $m = 7$. We also list the cost of SCFEM for one forcing function, where the CPU time on one collocation point is obtained by the time model (5.2) and J=135. One can see that the offline computational costs of the MsDSM and the DSM have the same order of magnitude. In addition, the offline computational cost of the MsDSM and the DSM is approximately equal to the cost of performing SCFEM for several different forcing functions.

We assume that the data-driven basis with seven modes gives sufficient approximation to the solution space. Let t_{DSMoff} and $t_{MsDSMoff}$ denote the computational time of DSM and MsDSM in the offline stage, respectively. Then, they are approximately given by

$$\begin{aligned} t_{DSMoff} &\approx 7.65 \times 10^{-5} N_h^{1.5}, \\ t_{MsDSMoff} &\approx 1.52 \times 10^{-4} N_h^{1.5}. \end{aligned} \tag{5.3}$$

Table 2
Computational time of the offline computation. (Time: Sec.) $m = 7$.

Grid number	$N_h = 336^2$	$N_h = 360^2$	$N_h = 384^2$
DSM	1790.1	2341.8	2640.1
MsDSM	2092.3	2466.6	3188.3
SCFEM	664.3	811.5	978.5

Grid number	$N_h = 408^2$	$N_h = 432^2$	$N_h = 456^2$	$N_h = 480^2$
DSM	3167.4	3870.5	4522.3	5138.9
MsDSM	3662.5	4305.8	4960.2	5673.5
SCFEM	1166.5	1376.7	1610.3	1868.4

In the online stage of the MsDSM or the DSM, we use the standard Galerkin method to solve (1.1). In the multiple query setting, the stiffness matrix S for the DSM or the MsDSM solver is fixed and the load vector b is different for each query. We can compute the Cholesky decomposition of S in advance, and the computational time is decided only by the forward and backward substitutions in solving the linear equation system. Actually, we can do the Cholesky decomposition of the stiffness matrix $S = LL^T$ in the offline stage, and save only the decomposition result L. The computational time of Cholesky decomposition is negligible compared with the training data-driven basis. Thus, we do not consider this part of the cost.

Let t_{fb} denote the time of forward and backward substitutions. In Table 3, we list the computation time of t_{fb} for different mesh grids and basis numbers. If we choose $m = 7$, then t_{fb} is approximately given by

$$t_{fb} \approx 1.27 \times 10^{-6} N_h^{1.4}. \tag{5.4}$$

Roughly speaking, if the MsDSM is applied on a coarse grid with a coarsening factor C in each direction, the speedup would be $\sim (C^2)^{1.4}$ in the online stage for each query. For example, if $C = 16$, the speedup is ~ 2352 ($256^{1.4}$). This essentially reveals the power of the upscaling method.

Remark 5.1. *We do not consider the computational time of adding correction terms to the MsDSM solution here. From numerical results in section* 6, *we can find that this part of the cost is also very small compared to the SCFEM solver.*

Remark 5.2. *The stiffness matrix S is a sparse positive definite matrix; however, the Cholesky decomposition matrix L is not sparse anymore. Before we perform the Cholesky decomposition, we reorder the matrix S using the approximate minimum degree (AMD) algorithm to ensure the least* fill-in. *However, when the scale of the stiffness matrix becomes large, the fill-in will become a serious problem, and the direct method will break down. We can find a good preconditioner, and design an effective iterative method, but this is beyond the scope of this paper and will be reported in our subsequent paper.*

6. Numerical examples.

In this section, we perform numerical experiments to test the performance and accuracy of the proposed MsDSM. We also demonstrate the computational efficiency of MsDSM over the traditional method, such as the stochastic collocation finite element method (SCFEM), in solving the multiquery problems with multiscale features. Finally, we compare the computational cost and accuracy of the MsDSM and the DSM in solving multiscale problems.

Table 3
Computational time of forward/back substitution. (Time: Sec.) m is the basis number. The data marked with an asterisk is obtained by extrapolation.

Grid Number	$N_h = 64^2$	$N_h = 128^2$	$N_h = 256^2$	$N_h = 512^2$	$N_h = 1024^2$
m=5	0.0626	0.4102	2.4672	17.0917	(*)114.5388
m=7	0.1281	0.8383	5.4933	(*)37.5849	(*)255.0034
m=9	0.2347	1.5620	10.3214	(*)66.6531	(*) 438.6207

6.1. Comparison of the MsDSM and the SCFEM.

Example 1. We consider the following stochastic elliptic equation with multiple scales on $D = [0,1] \times [0,1]$:

$$-\nabla \cdot (a^\varepsilon(x,y,\omega)\nabla u^\varepsilon(x,y,\omega)) = f(x,y,\theta), \quad (x,y) \in D, \omega \in \Omega, \theta \in \Theta, \tag{6.1}$$

$$u^\varepsilon(x,y,\omega) = 0, \quad (x,y) \in \partial D. \tag{6.2}$$

The multiscale information is described by the multiscale coefficient matrix $a^\varepsilon(x,y,\omega) = a_0^\varepsilon(x,y,\omega)\mathbf{I}_{2\times 2}$. The scalar function $a_0^\varepsilon(x,y,\omega)$ is given by

$$\begin{aligned} a_0^\varepsilon(x,y,\omega) = 0.1 &+ \frac{\xi_1(\omega)}{2 + 1.6\sin(2\pi(x-y)/\epsilon_1)} + \frac{\xi_2(\omega)}{4 + 1.8(\sin(2\pi x/\epsilon_2) + \sin(2\pi y/\epsilon_2))} \\ &+ \frac{\xi_3(\omega)}{10(2 + 1.8\sin(2\pi(x-0.5)/\epsilon_3))(2 + 1.8\sin(2\pi(y-0.5)/\epsilon_3))}, \end{aligned} \tag{6.3}$$

where $\epsilon_1 = 1/3$, $\epsilon_2 = 1/11$, and $\epsilon_3 = 1/19$, and $\{\xi_i\}_{i=1}^3$ are independent uniform random variables in $[0,1]$. In Figure 3, we plot four samples of the coefficient $a_0^\varepsilon(x,y,\omega)$. One can see that the coefficient oscillates very rapidly, which will generate small-scale features in the stochastic solution.

In our computations, we use the standard FEM to discretize the spatial dimension. We choose a 384×384 fine mesh to well resolve the *spatial dimension* of the stochastic solution $u^\varepsilon(x,y,\omega)$. Since the stochastic solution $u^\varepsilon(x,y,\omega)$ is smooth in the *stochastic dimension*, we use the sparse-grid based stochastic collocation method to discretize the stochastic dimension. First, we conduct a convergence study and find that the relative errors of mean and STD between the solutions obtained by level seven sparse grids in the SCFEM and higher-level sparse grids are smaller than 0.1% both in the L^2 and H^1 norms. Therefore, we choose level seven sparse grids with 207 points in the SCFEM and the MsDSM when we compare the computational cost of these two different methods. The reference solution is obtained by using higher-level sparse grids.

To implement the MsDSM, the coarse meshes are chosen to be 8×8, 16×16, 32×32, and 64×64, respectively, and we compare the results on different meshes and calculate the convergence rate. We remark that in the MsDSM, the forcing function $f(x,y,\theta)$ should be well-resolved by the coarse mesh; otherwise the numerical error will be large. We choose $\mathfrak{F} = \{\sin(k_i\pi x + \phi_i)\cos(l_i\pi y + \varphi_i)\}_{i=1}^{20}$, where k_i and l_i are uniformly distributed over the interval $[0,4]$, while ϕ_i and φ_i are uniformly distributed over the interval $[0,1]$ as the function class of the right-hand side in the preconditioning of the MsDSM method. We use this random training strategy to reduce the computational cost.

Multiquery results in the online stage. The MsDSM solver using 207 sparse grids in the computation produces $m = 7$ modes in the data-driven stochastic basis. In the online stage, we use them to solve the effective equation of the multiscale SPDE

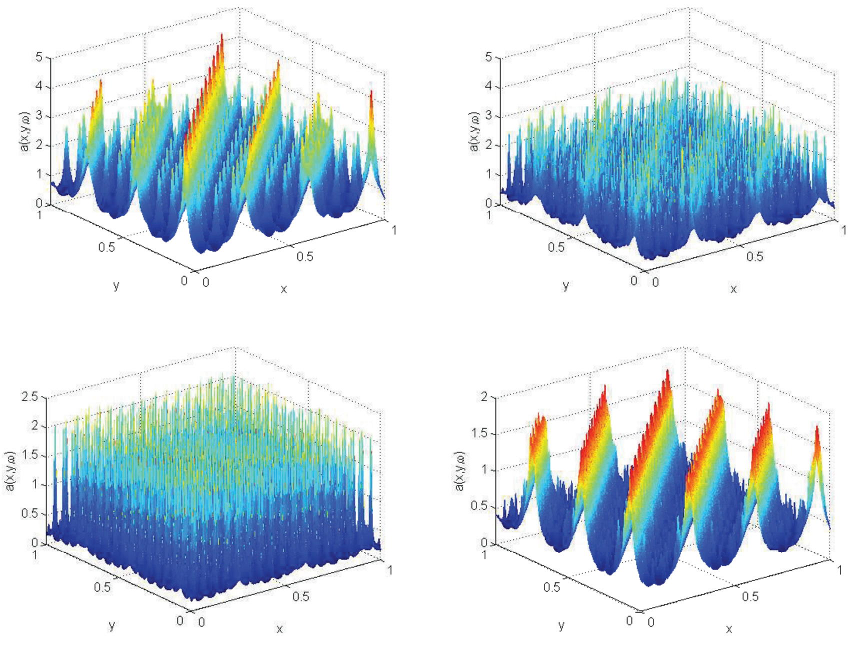


FIG. 3. *Some coefficient samples of* $a(x, y, \omega)$.

(6.1). We randomly generate 50 force functions of the form $f(x, y) \in \{\sin(k_i\pi x + l_i)\cos(m_i\pi x + n_i\}_{i=1}^{50}$, where k_i, l_i, m_i, and n_i are random numbers. In Figures 4 and 5, we show the relative errors of mean and STD of the MsDSM solution in the L^2 norm and the H^1 norm, respectively. In Tables 4 and 5, we list the mean of the relative errors for these 50 force functions on different coarse mesh grids. One can observe that the MsDSM solution converges approximately at a rate of $O(h^{1.5})$ in the L^2 norm and $O(h^1)$ in the H^1 norm.

In Figure 6, we show the mean and STD of the solution corresponding to $f(x, y) = \sin(3.1\pi x+0.2)\cos(0.5\pi y-0.3)$. It can be seen that the mean and STD of the MsDSM solution match the exact solution very well. We also show the contour plot of the mean of the solution corresponding to $f(x, y) = \sin(3.1\pi x + 0.2)\cos(0.5\pi y - 0.3)$ in Figure 7. One can see the heterogeneous structures of the multiscale solution.

To further test the performance of the MsDSM, we compare the solutions on a coarse grid obtained with and without numerical upscaling. Specifically, on the coarse grid with $N_c = 64$, we calculate the finite element solution of (6.1) (where the coefficient is chosen as the local average of $a^\varepsilon(x, y, \omega)$ around these coarse mesh grids) and interpolate these solutions from coarse grids to fine grids. We also calculate the MsDSM solution with small scale correction. In Figure 8, we plot the relative errors of mean and STD of these two solutions in the H^1 norm. Clearly, the MsDSM solver can effectively capture the small scale of the SPDE solutions.

Comparison of the MsDSM solver with the SCFEM solver. For the SCFEM solver, it will take 1648.38 seconds to solve (6.1) with one specific forcing term $f(x, y, \theta)$. Thus in a multiquery problem, if we need to solve (6.1) with n different forcing terms $f(x, y, \theta)$, the total computational cost will be $t_{SCFEM} = 1648.38n$. If we choose $N_c = 64$ in the MsDSM solver, the offline computation will cost 4732.66

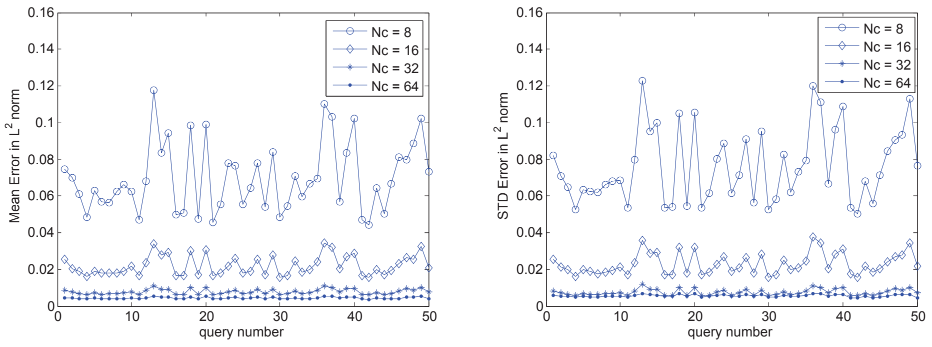


FIG. 4. *The mean and STD error of the MsDSM in the L^2 norm. N_c is the coarse grid number in each direction.*

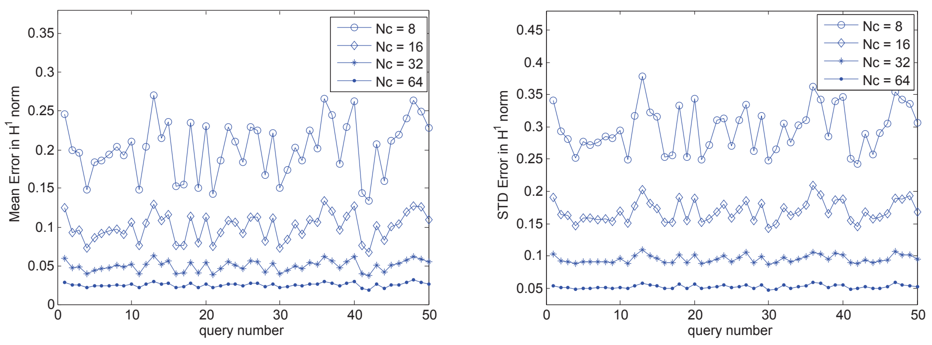


FIG. 5. *The mean and STD error of the MsDSM in the H^1 norm. N_c is the coarse grid number in each direction.*

seconds, which includes the computational time for deriving an effective SPDE, calculating correction term, and constructing DSM basis for the effective SPDE. In the online stage of the MsDSM, it takes 1.27 seconds to compute each query, and thus the total computational cost will be $t_{MsDSM} = 4732.66 + 1.27n$. We plot the total computational time in Figure 9. One can see that the MsDSM offers considerable computational savings over the SCFEM and is helpful if we need to solve the same SPDE many times with multiple forcing functions. Simple calculation shows that if we need to solve the original SPDE with more than three different forcing functions, the MsDSM will be superior to the SCFEM.

Example 2. In this example, we consider the SPDE (6.1)–(6.2) on $D = [0,1] \times [0,1]$ with the coefficient given by $a^{\varepsilon}(x,y,\omega) = a_0^{\varepsilon}(x,y,\omega)\mathbf{I}_{2\times 2}$. The scalar function $a_0^{\varepsilon}(x,y,\omega)$ is a random linear combination of five fixed coefficient fields plus a constant, i.e.,

$$a_0^{\varepsilon}(x,y,\omega) = \sum_{i=1}^{5} \xi_i(\omega) k_i(x,y) + 0.5, \tag{6.4}$$

where $\{\xi_i\}_{i=1}^5$ are independent uniform random variables in $[0,1]$, and $k_i(x,y)$, $i = 1,\ldots,5$, are fixed coefficient fields without scale separation. Specifically, $k_i(x,y) = |\theta_i(x,y)|$, where $\theta_i(x,y)$, $i = 1,\ldots,5$ are defined on 3×3, 5×5, 9×9, 17×17, and

TABLE 4
The relative error of mean in the L^2 and H^1 norms.

Mesh size	L^2	rate	H^1	rate
hc = $\frac{1}{8}$	0.0698		0.2253	
hc = $\frac{1}{16}$	0.0221	1.6592	0.1114	1.0161
hc = $\frac{1}{32}$	0.0078	1.5025	0.0552	1.0130
hc = $\frac{1}{64}$	0.0031	1.3312	0.0223	1.3076

TABLE 5
The relative error of STD in the L^2 and H^1 norms.

Mesh size	L^2	rate	H^1	rate
hc = $\frac{1}{8}$	0.0763		0.3305	
hc = $\frac{1}{16}$	0.0230	1.7300	0.1880	0.8139
hc = $\frac{1}{32}$	0.0076	1.5976	0.1059	0.8280
hc = $\frac{1}{64}$	0.0036	1.0780	0.0465	1.1874

31×31 grids over the domain D. For each grid cell, the value of $\theta_i(x, y)$ is normally distributed. In Figure 10, we show four samples of the coefficient $a_0^\varepsilon(x, y, \omega)$. One can see that the coefficients are very rough and do not satisfy scale separation or have any periodic structure. The implementations of the SCFEM and the MsDSM are exactly the same as in the previous example.

Multiquery results in the online stage. The MsDSM solver using 903 sparse grids in the computation produces $m = 8$ modes in the data-driven stochastic basis. In the online stage we use them to solve the effective equation of the multiscale SPDE (6.1). We randomly generate 50 force functions of the form $f(x, y) \in \{\sin(k_i\pi x + l_i)\cos(m_i\pi x + n_i\}_{i=1}^{50}$, where k_i, l_i, m_i, and n_i are random numbers. In Figures 11 and 12, we show the relative errors of mean and STD of the MsDSM solution in the L^2 norm and the H^1 norm, respectively. In Tables 6 and 7, we list the mean of the relative errors for these 50 force functions on different coarse mesh grids. One can observe that the MsDSM solution converges approximately at a rate of $O(h^{1.5})$ in the L^2 norm and $O(h^1)$ in the H^1 norm.

Comparison of the MsDSM solver with the SCFEM solver. For the SCFEM solver, it will take 7626.34 seconds to solve (6.1) with one specific forcing term $f(x, y, \theta)$. Thus in a multiquery problem, if we need to solve (6.1) with n different forcing terms $f(x, y, \theta)$, the total computational cost will be $t_{SCFEM} = 7626.34n$. If we choose $N_c = 64$ in the MsDSM solver, the offline computation will cost 21231.56 seconds. In the online stage of the MsDSM, it takes 1.82 seconds to compute one query, and thus the total computational cost will be $t_{MsDSM} = 21231.56 + 1.82n$. The MsDSM offers considerable computational savings over the SCFEM and is helpful if we need to solve the same SPDE with more than three different forcing functions.

Example 3. We consider the SPDE (6.1)–(6.2) on $D = [0, 1] \times [0, 1]$ with a high-contrast random coefficient. The elliptic coefficient is given by $a^\varepsilon(x, y, \omega) = a_0^\varepsilon(x, y, \omega)\mathbf{I}_{2\times 2}$, with $a_0^\varepsilon(x, y, \omega)$ given by a random high-contrast field. Specifically, $a_0^\varepsilon(x, y, \omega)$ is a random linear combination of inclusion fields and channel fields plus a constant, i.e.,

$$a_0^\varepsilon(x, y, \omega) = \sum_{i=1}^{3} \xi_i(\omega) k_i(x, y) + 1.0, \qquad (6.5)$$

where $\{\xi_i\}_{i=1}^3$ are independent uniform random variables in $[0, 1]$, $k_1(x, y)$ is an in-

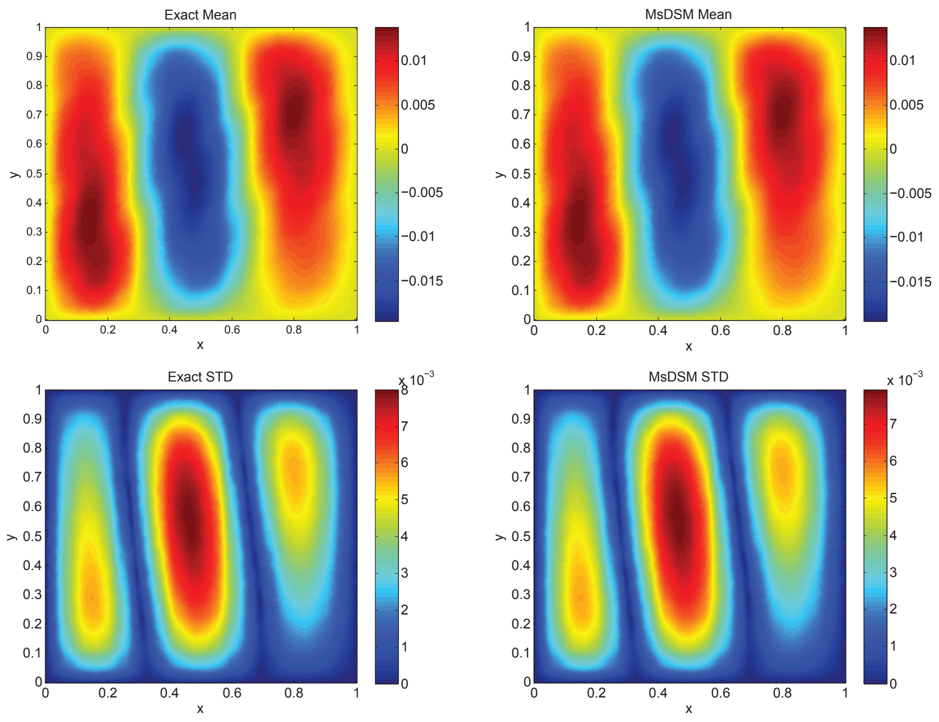


FIG. 6. *Profiles of the mean and STD solution.*

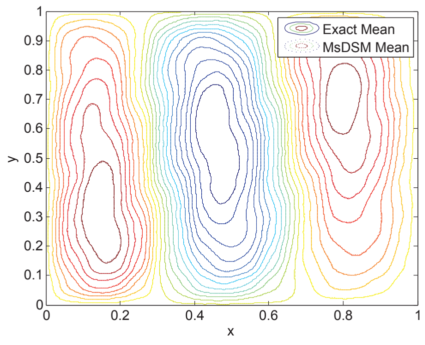


FIG. 7. *Contour plot of the mean solution.*

clusion field, and $k_2(x, y)$ and $k_3(x, y)$ are two channel fields. In Figure 13(a)–(c), we show the inclusion field and channel field, respectively, while in Figure 13(d)–(f) we show three samples of the coefficient $a_0^\varepsilon(x, y, \omega)$. One can see the diversity of the random high-contrast coefficients. This presents a challenging test problem for the MsDSM. The implementations of the SCFEM and the MsDSM are exactly the same as in the previous examples.

Multiquery results in the online stage. The MsDSM solver using 207 sparse grids in the computation produces $m = 10$ modes in the data-driven stochastic basis. In the online stage we use them to solve the effective equation of the multiscale SPDE

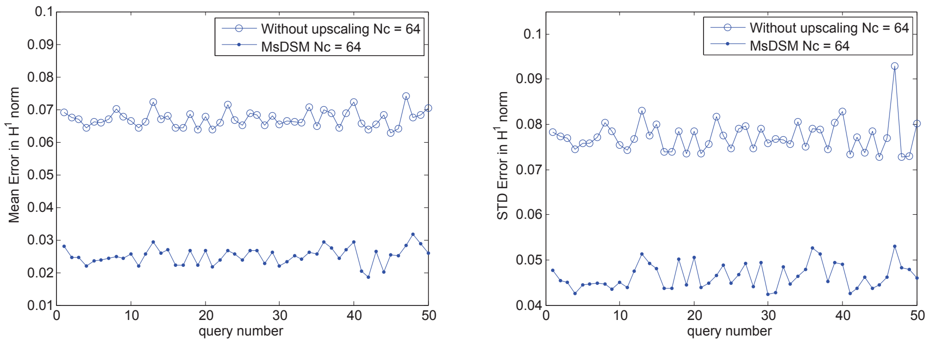


Fig. 8. *The effectiveness of the numerical upscaling.*

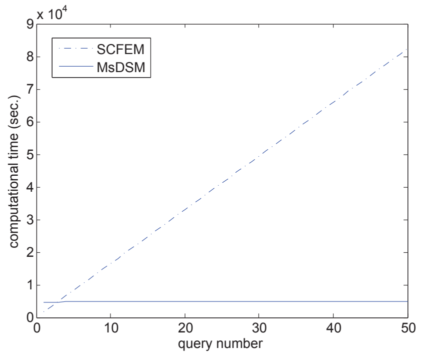


Fig. 9. *The computation time comparison.*

(6.1). We randomly generate 50 force functions of the form $f(x, y) \in \{\sin(k_i\pi x + l_i)\cos(m_i\pi x + n_i\}_{i=1}^{50}$, where k_i, l_i, m_i, and n_i are random numbers. In Figures 14 and 15, we show the relative errors of mean and STD of the MsDSM solution in the L^2 norm and theH^1 norm, respectively. One can observe that the MsDSM solution converges in both the L^2 norm and the H^1 norm.

In Figure 16, we show the mean and STD of the solution corresponding to $f(x, y) = \sin(0.7\pi x + 0.2)\cos(3.3\pi y + 0.2)$. It can be seen that the mean and STD of the MsDSM solution match those of the exact solution very well. Due to the inclusions and channels in the permeability field, the mean and STD of the stochastic solution possess some interesting structures. In Figure 17, we plot the STD error of the MsDSM solution. One can see that large uncertainty exists around the boundary of the inclusion or channel field.

Comparison of the MsDSM solver with the SCFEM solver. For the SCFEM solver, it will take 1648.04 seconds to solve (6.1) with one specific forcing term $f(x, y, \theta)$. Thus in a multiquery problem, if we need to solve (6.1) with n different forcing terms $f(x, y, \theta)$, the total computational cost will be $t_{SCFEM} = 1648.04n$. If we choose $N_c = 64$ in the MsDSM solver, the offline computation will cost 7782.68 seconds, which includes the computational time for deriving the effective SPDE, calculating the correction term, and the constructing DSM basis for the effective SPDE. In the online stage of the MsDSM, it takes 2.95 seconds to compute one query; thus the total computational cost will be $t_{MsDSM} = 7782.68 + 2.95n$. We plot the total

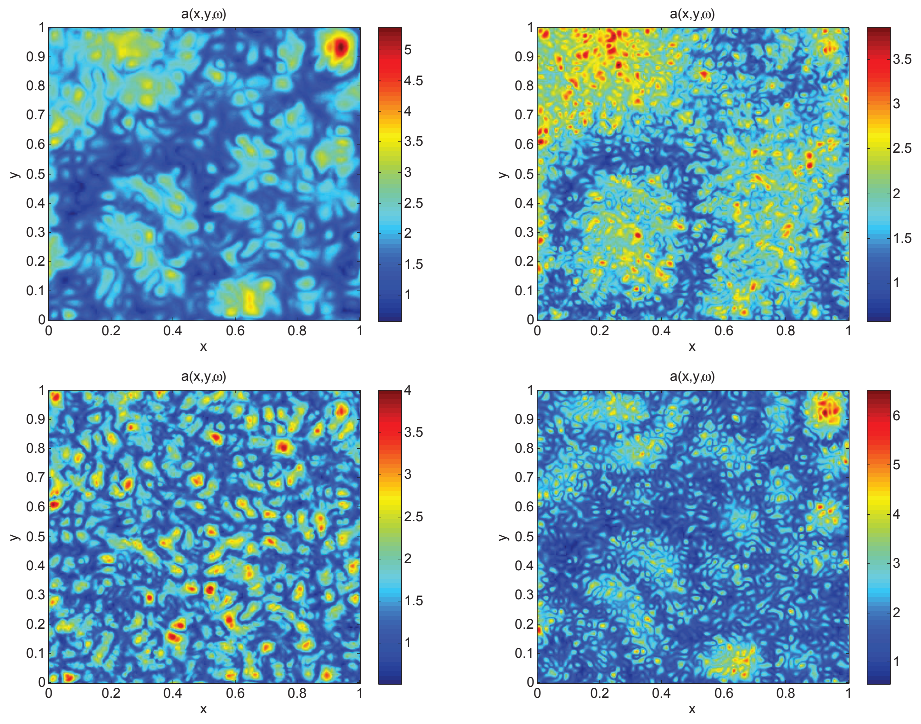


FIG. 10. *Some coefficient samples of* $a(x, y, \omega)$.

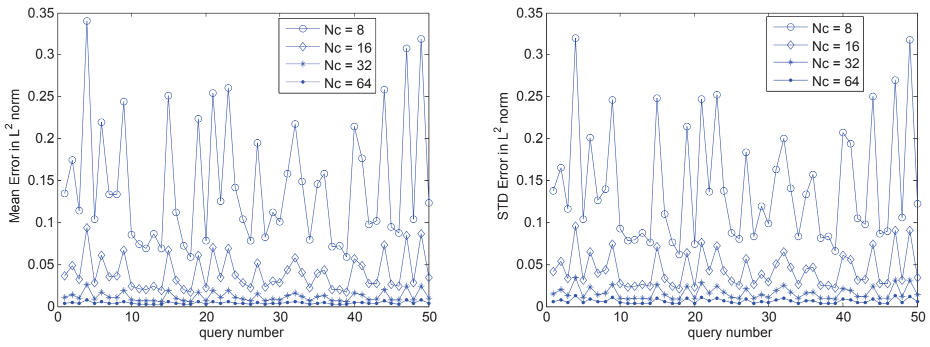


FIG. 11. *The mean and STD error of the MsDSM in the* L^2 *norm.* N_c *is the coarse grid number in each direction.*

computational time in Figure 18. One can see that the MsDSM offers considerable computational savings over the SCFEM and is helpful if we need to solve the same SPDE with more than five different forcing functions.

6.2. Comparison of the MsDSM and the DSM. As the authors have demonstrated in [11], the DSM offers considerable computational savings over some traditional methods such as the gPC, stochastic collocation, and Monte Carlo methods in solving the multiquery problem. Finally, we compare the computational cost and accuracy of the MsDSM and the DSM in solving multiscale problems.

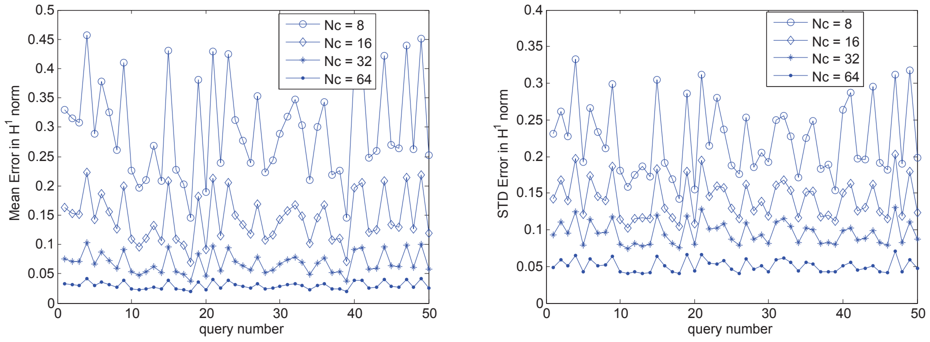


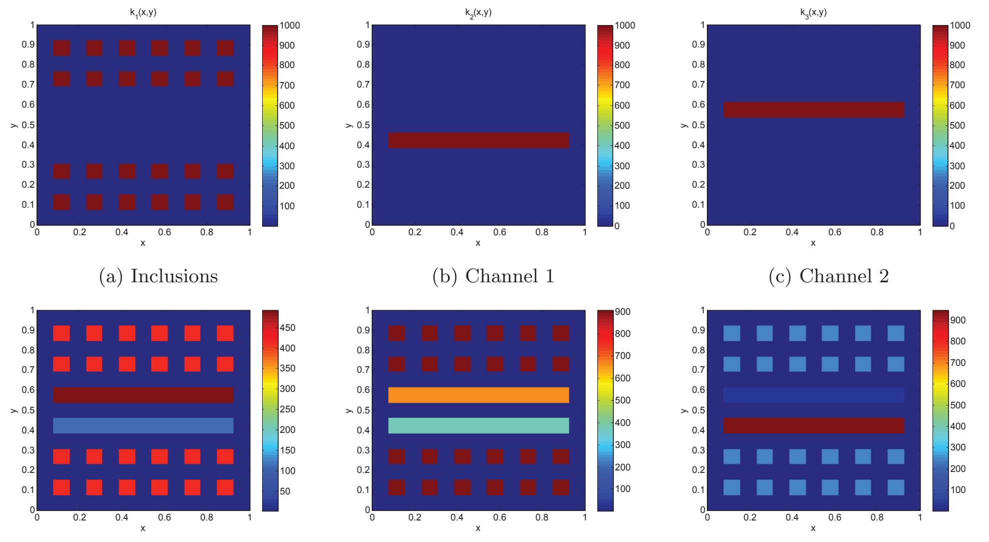
FIG. 12. *The mean and STD error of the MsDSM in the H^1 norm. N_c is the coarse grid number in each direction.*

TABLE 6
The relative error of mean in L^2 and H^1 norm.

Mesh size	L^2	Rate	H^1	Rate
hc $= \frac{1}{8}$	0.1889		0.3464	
hc $= \frac{1}{16}$	0.0521	1.8583	0.1684	1.0405
hc $= \frac{1}{32}$	0.0158	1.7214	0.0796	1.0811
hc $= \frac{1}{64}$	0.0051	1.6314	0.0339	1.2315

TABLE 7
The relative error of STD in L^2 and H^1 norms.

Mesh size	L^2	Rate	H^1	Rate
hc $= \frac{1}{8}$	0.1936		0.2764	
hc $= \frac{1}{16}$	0.0589	1.7167	0.1755	0.6553
hc $= \frac{1}{32}$	0.0216	1.4472	0.1018	0.7857
hc $= \frac{1}{64}$	0.0082	1.3973	0.0514	0.9859



(a) Inclusions (b) Channel 1 (c) Channel 2

(d) Coefficient 1 (e) Coefficient 2 (f) Coefficient 3

FIG. 13. *Inclusions, channels, and random coefficients.*

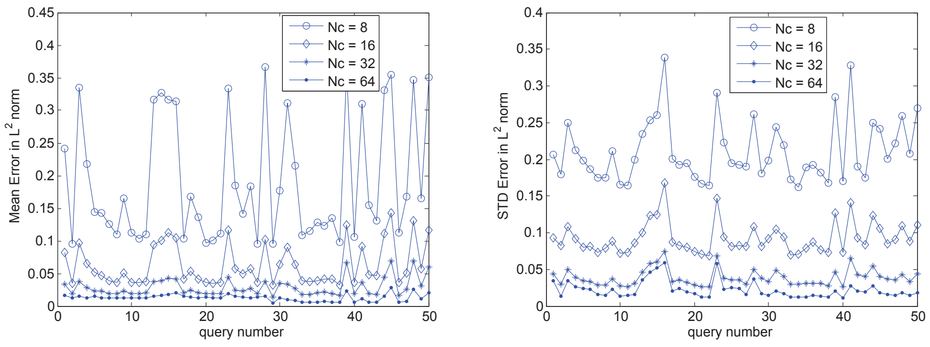


FIG. 14. *The mean and STD error of the MsDSM in the L^2 norm. N_c is the coarse grid number in each direction.*

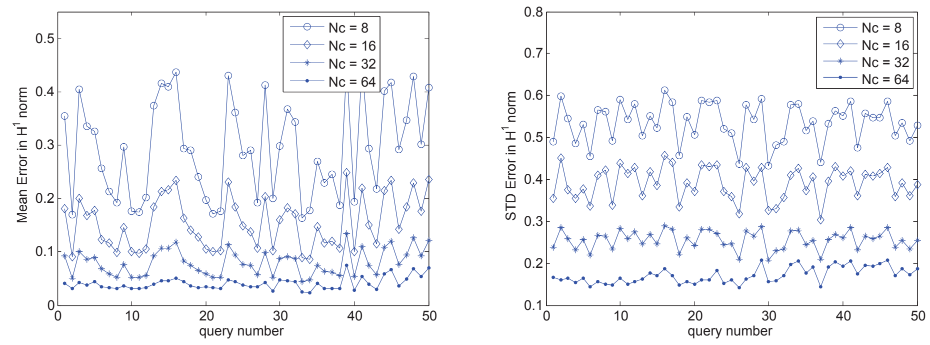


FIG. 15. *The mean and STD error of the MsDSM in the H^1 norm. N_c is the coarse grid number in each direction.*

Example 4. We consider the SPDE (6.1)–(6.2) on $D = [0,1] \times [0,1]$ with the coefficient given by

$$a_0^\varepsilon(x,y,\omega) = 0.1 + \xi_1(\omega)\frac{2+1.8\sin(2\pi x/\epsilon_1)}{2+1.8\sin(2\pi y/\epsilon_1)} + \xi_2(\omega)\frac{2+1.8\sin(2\pi y/\epsilon_2)}{2+1.8\cos(2\pi x/\epsilon_2)} + \xi_3(\omega)\frac{2+1.8\cos(2\pi x/\epsilon_3)}{2+1.8\sin(2\pi y/\epsilon_3)}, \tag{6.6}$$

where $\epsilon_1 = 1/3$, $\epsilon_2 = 1/11$, and $\epsilon_3 = 1/19$, and $\{\xi_i\}_{i=1}^3$ are independent uniform random variables in $[0,1]$.

In our computations, we use the standard FEM to discretize the spatial dimension. We choose a 384×384 fine mesh to well resolve the spatial dimension of the stochastic solution $u^\varepsilon(x,y,\omega)$. We choose level six sparse grids in the discretization of the stochastic dimension, which has 135 points. The reference solution is obtained by using higher-level sparse grids. The coarse mesh of the MsDSM is chosen to be 64×64. We implement the DSM on a 384×384 fine mesh and 64×64 coarse mesh, respectively.

The MsDSM generates $m = 7$ modes in the data-driven stochastic basis, while the DSM generates $m = 9$ modes. In the online stage we use them to solve the effective equation of the multiscale SPDE (6.1) with the coefficient given by (6.6). We randomly

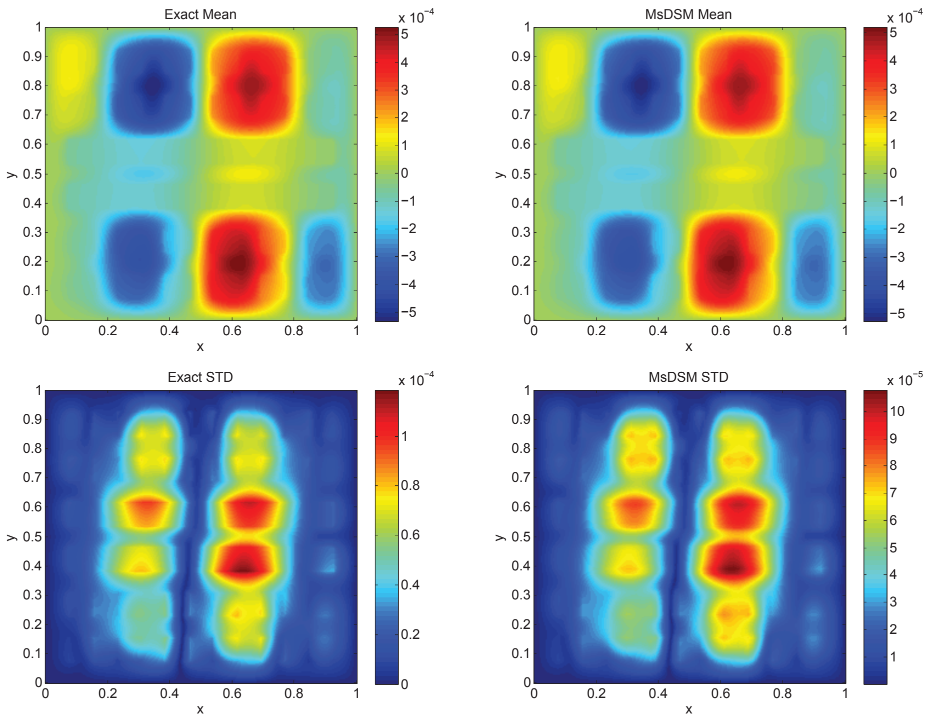


FIG. 16. *Profile of the mean and STD solution.*

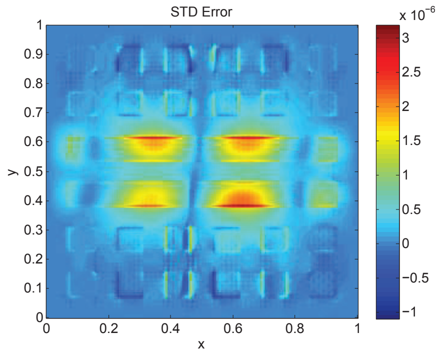


FIG. 17. *The error of the STD.*

generate 50 force functions of the form $f(x,y) \in \{\sin(k_i\pi x + l_i)\cos(m_i\pi x + n_i\}_{i=1}^{50}$, where k_i, l_i, m_i, and n_i are random numbers. In Figures 19 and 20, we show the relative errors of mean and STD of the MsDSM and DSM solutions in the L^2 norm and the H^1 norm, respectively. Here $DSMf$ and $DSMc$ denote the DSM solution obtained on the fine and coarse grids, respectively. We conclude that the accuracy of the MsDSM is comparable with that obtained by the DSM on the fine mesh grid. In addition, applying the DSM on a coarse mesh grid without any numerical upscaling will generate large errors in the numerical solution.

For the SCFEM solver, it will take 1132.94 seconds to solve (6.1) with one specific

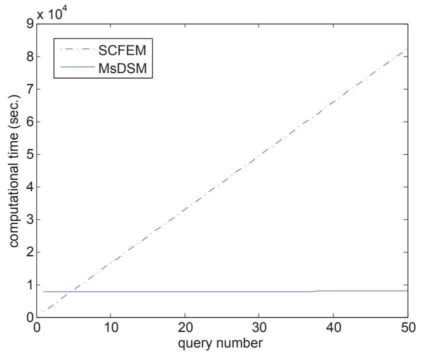


FIG. 18. *The computation time comparison.*

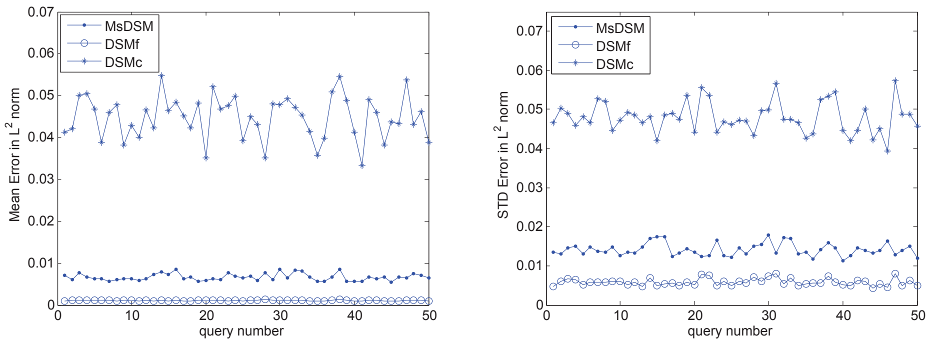


FIG. 19. *The mean and STD error of the MsDSM and the DSM in the L^2 norm.*

forcing term $f(x, y, \theta)$. Thus in a multiquery problem, if we need to solve (6.1) with n different forcing terms $f(x, y, \theta)$, the total computational cost will be $t_{SCFEM} = 1132.94n$. The offline computation of the MsDSM and the DSM costs 3254.17 and 2898.32 seconds, respectively. In the online stage of the MsDSM, it takes 1.89 seconds to compute one query, and thus the total computational cost will be $t_{MsDSM} = 3254.17+1.89n$. For the DSM solver on the fine grid, it takes 33.29 seconds to compute one query, and thus the total computational cost will be $t_{MsDSM} = 2898.32+33.29n$. It turns out that both the MsDSM and the DSM offer considerable computational savings over the SCFEM and are helpful if we need to solve the same SPDE with more than three different forcing functions. The offline cost of the MsDSM is more expensive than the DSM, since we have to derive the effective equation. However, the online cost will be much cheaper than the DSM because we solve the effective equation on a coarse grid.

Example 5. Finally, we consider the SPDE (6.1)–(6.2) on $D = [0, 1] \times [0, 1]$ with the coefficient given by (6.6). This time, we choose $\epsilon_1 = 1/3$, $\epsilon_2 = 1/19$, and $\epsilon_3 = 1/65$, and $\{\xi_i\}_{i=1}^3$ are independent uniform random variables in $[0, 1]$. We choose a 1024×1024 fine mesh to well resolve the spatial dimension of the stochastic solution $u^\varepsilon(x, y, \omega)$. We choose level six sparse grids in the discretization of the stochastic dimension, which has 135 points. The reference solution is obtained by using higher-level sparse grids. In this example, due to memory overflow, the DSM easily breaks down. However, MsDSM still works, owing to the upscaling in the physical dimension.

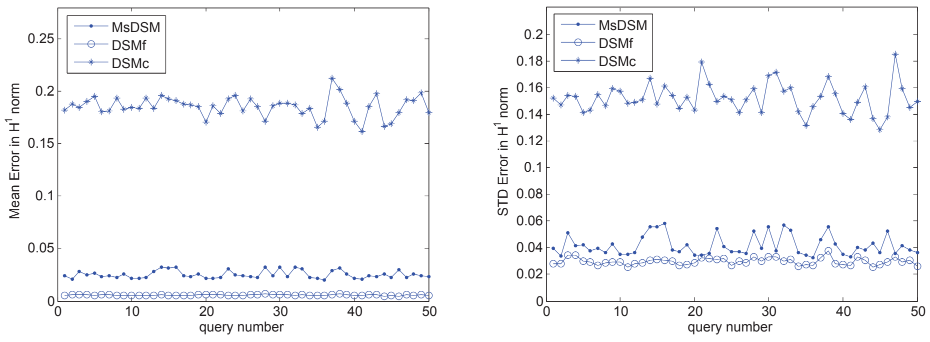


FIG. 20. *The mean and STD error of the MsDSM and the DSM in the H^1 norm.*

The MsDSM solver using 135 sparse grids in the computation produces $m = 8$ modes in the data-driven stochastic basis. In the online stage we use them to solve the effective equation of the multiscale SPDE (6.1). We randomly generate 50 force functions of the form $f(x, y) \in \{\sin(k_i\pi x + l_i)\cos(m_i\pi x + n_i\}_{i=1}^{50}$, where k_i, l_i, m_i, and n_i are random numbers. In Figure 21, we show the relative errors of mean and STD of the MsDSM solution in the L^2 norm and the H^1 norm, respectively.

For the SCFEM solver, it will take 18620.01 seconds to solve (6.1) with one specific forcing term $f(x, y, \theta)$. Thus in a multiple query problem, if we need to solve (6.1) with n different forcing term $f(x, y, \theta)$, the total computational cost will be $t_{SCFEM} = 18620.01n$. If we choose $N_c = 64$ in the MsDSM solver, the offline computation will cost 49258.59 seconds, which includes the computational time for deriving the effective SPDE, calculating the correction term, and constructing the DSM basis for the effective SPDE. In the online stage of the MsDSM, it takes 18.25 seconds to compute one query, and thus the total computational cost will be $t_{MsDSM} = 49258.59+18.25n$. MsDSM offers considerable computational savings over the SCFEM and is helpful if we need to solve the same SPDE with more than three different forcing functions. We conjecture that the time model obtained in section 5.2 may still be valid for the DSM. Actually, due to the fill-in, the real computation time and memory cost will be larger. The total computational cost for the DSM can be extrapolated as $t_{DSM} = 47700.90 + 438.62n$. We plotted the total computational time in Figure 1. One can observe that the MsDSM offers huge savings over other methods in solving multiscale problems.

REMARK 6.1. *When the input dimension of the random variables is high, the stochastic collocation method will be very expensive or even infeasible. In this case, most of the existing methods are expensive, especially for the problems with multiscale features. As demonstrated in* [11], *one can develop MsDSM with the ensemble representation, since the accuracy of the Monte Carlo method does not depend on the dimension of the input random variables. One can also adopt the adaptive ANOVA (analysis of variation) decomposition technique to decompose the original high-dimensional multiscale problem into a set of low-dimensional subproblems, and can apply the MsDSM on each subproblem accordingly; see* [44].

7. Conclusion remarks.

In this paper, we developed a novel multiscale data-driven stochastic method (MsDSM) to solve multiscale stochastic partial differential equations (SPDEs) in a multiquery setting. These SPDEs arise from various appli-

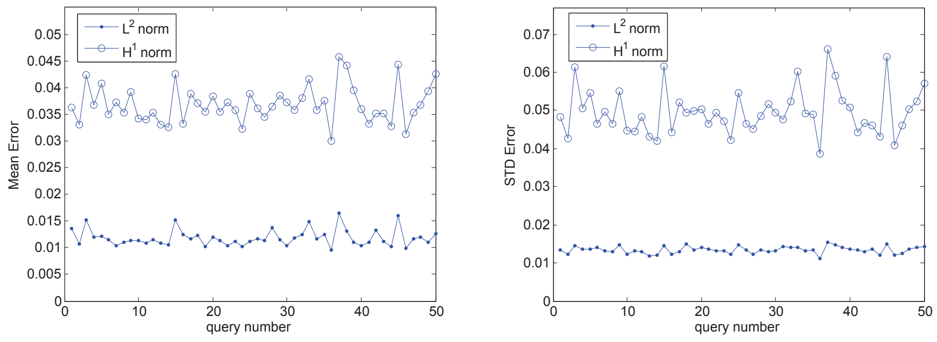


FIG. 21. *The mean and STD error of the MsDSM in the L^2 and H^1 norms.*

cations such as heterogeneous porous media flow problem in a water aquifer and oil reservoirs simulation. Our method consists of offline and online stages. In the offline stage, we first derive the effective SPDE on coarse grids by adopting the multiscale model reduction approach proposed in [12]. Then, we construct a data-driven stochastic basis $\{A_i(\omega)\}_{i=1}^m$ by using the Karhunen–Loève (KL) expansion and a two-level optimization approach based on multiple trial functions. In the online stage, we use the standard Galerkin projection method (with our data-driven stochastic basis) to solve the effective SPDE for a class of forcing functions. We can also improve the accuracy of the MsDSM solution by adding the small-scale correction term saved in the offline stage. By the model reduction from both stochastic and physical dimensions, the MsDSM offers considerable computational savings over some traditional methods such as the stochastic collocation finite element method (SCFEM).

We presented several numerical examples for the two-dimensional stochastic elliptic PDEs with random multiscale or high-contrast coefficients to demonstrate the accuracy and efficiency of the proposed method. These numerical examples indicate the following advantages of the proposed MsDSM: (1) by integrating the DSM with the multiscale model reduction, the MsDSM can effectively solve stochastic multiscale PDEs with desirable accuracy on a coarse grid; (2) the optimal data-driven stochastic basis can be used for the multiscale SPDEs with a class of deterministic forcing functions; (3) compared to classic numerical solvers such as the Monte Carlo method and SCFEM, the MsDSM offers considerable computational savings and is helpful if one needs to solve the same SPDE many times with multiple forcing functions.

It is important to point out that since our methods involve the computation of global harmonic coordinates, the memory consumption becomes a serious issue when the ratio of the smallest scale and the largest scale in the stochastic multiscale problem (1.1) is very small. In this case, we may use a localized upscaling method such as the MsFEM [18] developed by Hou and Wu. to reduce the memory consumption. We are currently adopting the localized upscaling method and multilevel Monte Carlo method in the study of this class of problems.

Acknowledgment. We thank Sydney Garstang for proofreading the manuscript.

REFERENCES

[1] M. ARNST AND R. GHANEM, *Probabilistic equivalence and stochastic model reduction in multi-*

scale analysis, Comput. Methods Appl. Mech. Engrg., 197 (2008), pp. 3584–3592.

[2] G. Allaire and R. Brizzi, *A multiscale finite element method for numerical homogenization*, Multiscale Model. Simul., 4 (2005), pp. 790–812.

[3] T. Arbogast, G. Pencheva, M. F. Wheeler, and I. Yotov, *A multiscale mortar mixed finite element method*, Multiscale Model. Simul., 6 (2007), pp. 319–346.

[4] I. Babuška and E. Osborn, *Generalized finite element methods: Their performance and their relation to mixed methods*, SIAM J. Numer. Anal., 20 (1983), pp. 510–536.

[5] I. Babuška, F. Nobile, and R. Tempone, *A stochastic collocation method for elliptic partial differential equations with random input data*, SIAM Rev., 52 (2010), pp. 317–355.

[6] H. J. Bungartz and M. Griebel, *Sparse grids*, Acta Numer., 13 (2004), pp. 147–269.

[7] R. H. Cameron and W. T. Martin, *The orthogonal development of non-linear functionals in series of Fourier-Hermite functionals*, Ann. of Math., 48 (1947), pp. 385–392.

[8] Z. Chen and T. Y. Hou, *A mixed multiscale finite element method for elliptic problems with oscillating coefficients*, Math. Comp., 72 (2002), pp. 541–576.

[9] M. L. Cheng, T. Y. Hou, and Z. W. Zhang, *A dynamically bi-orthogonal method for time-dependent stochastic partial differential equations* I: *Derivation and algorithms*, J. Comput. Phys., 242 (2013), pp. 843–868.

[10] M. L. Cheng, T. Y. Hou, and Z. W. Zhang, *A dynamically bi-orthogonal method for time-dependent stochastic partial differential equations* II: *Adaptivity and generalizations*, J. Comput. Phys., 242 (2013), pp. 753–776.

[11] M. L. Cheng, T. Y. Hou, M. Yan, and Z. W. Zhang, *A data-driven stochastic method for elliptic PDEs with random coefficients*, SIAM/ASA J. Uncertain. Quantif., 1 (2013), pp. 452–493.

[12] M. L. Ci, T. Y. Hou, and Z. Shi, *A multiscale model reduction method for partial differential equations*, ESAIM Math. Model. Numer. Anal., 48 (2014), pp. 449–474.

[13] M. L. Ci, M. B. Giles, T. Y. Hou, and Z. W. Zhang, *A multiscale multilevel Monte Carlo method for elliptic PDEs with random coefficients*, SIAM/ASA J. Uncertain. Quantif., submitted.

[14] C. C. Chu, I. Graham, and T. Y. Hou, *A New multiscale finite element method for high-contrast elliptic interface problems*, Math. Comp., 79 (2010), pp. 1915–1955.

[15] W. E and B. Engquist, *The heterogeneous multi-scale methods*, Commun. Math. Sci., 1 (2003), pp. 87–133.

[16] Y. Efendiev and T. Y. Hou, *Multiscale Finite Element Methods. Theory and Applications*, Springer, New York, 2009.

[17] H. D. Han and Z. W. Zhang, *Multiscale tailored finite point method for second order elliptic equations with rough or highly oscillatory coefficients*, Commun. Math. Sci., 10 (2012), pp. 945–976.

[18] T. Y. Hou and X. H. Wu, *A multiscale finite element method for elliptic problems in composite materials and porous media*, J. Comput. Phys., 134 (1997), pp. 169–189.

[19] T. Hughes, G. Feijoo, L. Mazzei, and J. Quincy, *The variational multiscale method—a paradigm for computational mechanics*, Comput. Methods Appl. Mech. Engrg, 166 (1998), pp. 3–24.

[20] P. Jenny, S. H. Lee, and H. Tchelepi, *Multi-scale finite volume method for elliptic problems in subsurface flow simulation*, J. Comput. Phys., 187 (2003), pp. 47–67.

[21] P. Dostert, Y. Efendiev, T. Y. Hou, and W. Luo, *Coarse gradient Langevin algorithms for dynamic data integration and uncertainty quantification*, J. Comput. Phys., 217 (2006), pp. 123–142.

[22] Y. Efendiev, T. Hou, and W. Luo, *Preconditioning Markov chain Monte Carlo simulations using coarse-scale models*, SIAM J. Sci. Comput., 28 (2006), pp. 776–803.

[23] B. Ganapathysubramanian and N. Zabaras, *Modelling diffusion in random heterogeneous media: Data-driven models, stochastic collocation and the variational multi-scale method*, J. Comput. Phys., 226 (2007), pp. 326–353.

[24] R. G. Ghanem and P. D. Spanos, *Stochastic Finite Elements: A Spectral Approach*, Springer-Verlag, New York, 1991.

[25] T. Y. Hou, W. Luo, B. Rozovskii, and H. Zhou, *Wiener Chaos expansions and numerical solutions of randomly forced equations of fluid mechanics*, J. Comput. Phys., 216 (2006), pp. 687–706.

[26] K. Karhunen, *Uber lineare Methoden in der Wahrscheinlichkeitsrechnung*, Ann. Acad. Sci. Fennicae. Ser. A. I. Math.-Phys., no. 37, 1947.

[27] I. G. Kevrekidis, C. W. Gear, J. M. Hyman, P. G. Kevrekidis, O. Runborg, and C. Theodoropoulos, *Equation-free, coarse-grained multiscale computation: Enabling microscopic simulators to perform system-level analysis*, Commun. Math. Sci., 1 (2003),

pp. 715–762.

[28] I. G. Kevrekidis and G. Samaey, *Equation-free multiscale computation: Algorithms and applications*, Annu. Rev. Phys. Chem., 60 (2009), pp. 321–344.

[29] M. Loeve, *Probability Theory*. II, 4th ed., Grad. Texts Math. 46, Springer-Verlag, New York, Heidelberg, 1978.

[30] X. Ma and N. Zabaras, *An adaptive hierarchical sparse grid collocation algorithm for the solution of stochastic differential equations*, J. Comput. Phys., 228 (2009), pp. 3084–3113.

[31] O. Le Maitre, *Uncertainty propagation using Wiener-Haar expansions*, J. Comput. Phys., 197 (2004), pp. 28–57.

[32] A. J. Majda and M. Branicki, *Lessons in uncertainty quantification for turbulent dynamical system*s, Discrete Contin. Dyn. Syst., 32 (2012), pp. 3133–3221.

[33] H. N. Najm, *Uncertainty quantification and polynomial chaos techniques in computational fluid dynamics*, Ann. Rev. Fluid Mech., 41 (2009), pp. 35–52.

[34] B. K. Oksendal, *Stochastic Differential Equations: An Introduction with Applications*, 6th ed., Springer, Berlin, 2003.

[35] H. Owhadi and L. Zhang, *Metric based upscaling*, Comm. Pure Appl. Math., 60 (2007), pp. 675–723.

[36] C. Schwab and R. A. Todor, *Karhunen-Loeve approximation of random fields by generalized fast multipole methods*, J. Comput. Phys., 217 (2006), pp. 100–122.

[37] S. Sirisup, G. E. Karniadakis, D. Xiu, and I. G. Kevrekidis, *Equation-free/Galerkin-free POD-assisted computation of incompressible flows*, J. Comput. Phys., 207 (2005), pp. 568–587.

[38] B. V. Asokan and N. Zabaras, *A stochastic variational multiscale method for diffusion in heterogeneous random media*, J. Comput. Phys., 218 (2006), pp. 654–676.

[39] X. Wan and G. E. Karniadakis, *An adaptive multi-element generalized polynomial chaos method for stochastic differential equations*, J. Comput. Phys., 209 (2005), pp. 617–642.

[40] N. Wiener, *The homogeneous chaos*, Amer. J. Math., 60 (1938), pp. 897–936.

[41] D. Xiu and G. E. Karniadakis, *The Wiener–Askey polynomial chaos for stochastic differential equations*, SIAM J. Sci. Comput., 24 (2002), pp. 614–644.

[42] D. Xiu and G. E. Karniadakis, *Modeling uncertainty in flow simulations via generalized polynomial chaos*, J. Comput. Phys., 187 (2003), pp. 137–167.

[43] D. Xiu and J. S. Hesthaven, *High-order collocation methods for differential equations with random inputs*, SIAM J. Sci. Comput., 27 (2005), pp. 1118–1139.

[44] Z. W. Zhang, X. Hu, T. Y. Hou, G. Lin, and M. Yan, *An adaptive ANOVA-based data-driven stochastic method for elliptic PDE with random coefficients*, Commun. Comput. Phys., 16 (2014), pp. 571–598.